Aaron J. Parry

Student ID:	0050025590
Course:	ENG4112 Research Project Part 2
Class:	90323
Assessment:	PROJECT PERFORMANCE
Due Date:	29-OCT-2009
Submitted To:	Faculty of Engineeri

5056027

Mark Given (Office use only): ___________ **Marker Signature:** ________________________________

Tutor Name:

Tutorial Group Number: **Room:**

Day: **Time:**

Campus:

STUDENT DECLARATION:

I hold a copy of this assignment which I can produce if the original is lost or damaged.

I hereby certify that no part of this assignment has been copied from any other student's work or from any other source except where due acknowledgement is made in the assignment. No part of this assignment has been written for me by any other person except where such collaboration has been authorised by the Course Team Leader or the examiner concerned.

Signature:________________________ **Date:**________________

Notes:

1. An Examiner or a Course Team Leader has, and may exercise, the right not to mark this assignment if the above declaration has not been signed.
2. If the above declaration is found to be false, further appropriate action will be taken.

Make sure that you have used the **CORRECT** Assignment Cover for your assignment.
Incorrect assignment cover sheets will cause delays.

Faculty of Engineering and Surveying

ENG4111 Research Project Part 1 & ENG4112 Research Project Part 2

DISSERTATION SUBMISSION FORM

STUDENT USE ONLY:

Student Name:	Aaron Parry
Student Number:	W0050025590
Program:	Bachelor of Engineering-- Electrical and Electronics
Mode:	On Campus
Project Title:	Remote Aerial Data Acquisition Concept -- RADAC
Supervisor/s:	Mr. Kerry Nufer and Dr. Alexander Kist

1 printed and soft bound assessment copy	x
Inclusion of a Title page	x
Inclusion of a Disclaimer page (it should be titled Limitations of Use)	x
Inclusion of a signed Certification page	x
Inclusion of Appendix A - Project Specification	x
2 CDs containing a single PDF file identical to the printed copy (refer Project Reference Book section 11.11)	x

I declare that I have submitted (Student to sign): ______________ Date: 29/10/2009

Assignment Label	Date Stamp	Library Copy
		Does the Supervisor give permission for a copy to be forwarded to the Library?*

* If a formal undertaking of Confidentiality has been given, then both CD copies will remain with the Supervisor (refer 6.4 of the Project Reference Book).

USQ collects personal information to assist the University in providing tertiary education and related ancillary services and to be able to contact you regarding enrolment, assessment and associated USQ services. Personal information will not be disclosed to third parties without your consent unless required by law.

USQ collects personal information to assist the University in providing tertiary education and related ancillary services and to be able to contact you regarding enrolment, assessment and associated USQ services. Personal information will not be disclosed to third parties without your consent unless required by law.

University of Southern Queensland

Faculty of Engineering & Surveying

Remote Aerial Data Acquisition & Capture Project (RADAC)

A dissertation submitted by

Aaron Parry

In fulfillment of the requirements of

ENG4111/2 Research Project

Towards the degree of

Bachelor of Electrical and Electronic Engineering

Submitted: October, 2009

Abstract

The RADAC Project encompasses the design and prototype implementation of a system for low-cost aerial data sensor acquisition. It includes a Ground Transponder Unit (GTU), and Aerial Interrogation System (AIS) mounted under an aircraft. The GTU captures and transmits water-meter readings; the AIS initiates' measurements and processes and displays the results. The proposed system is based on RF devices in association with a small low-cost single chip camera and microcontrollers.

During a consultancy to a large Queensland government authority which has approximately 8000 water meters in regional and remote parts of the state. It was realised that considerable savings could be made in the management of water resources and human resources needed to read these at three month intervals.

This project will calculate the RF subsystem performance in terms of gain, beamwidth, return loss, bandwidth, and matching of the antenna into the RF transceiver device Design Executive Data Acquisition System and Design and implement Interrogation Microcontroller and Design PCB for RF Transceiver and Design and calculate power usage and Power Supply design and calculate High-Gain Helical Antenna.

The solution based on using RF devices based on IEEE802.15.4 (IEEE 2006) in association with a small low cost single chip camera and microcontroller. Figure 1 shows the Block Diagram of the Ground Transponder Unit and symbolic AIS.

The potential saving in maintenance costs to industry by remotely taking measurements is significant enough to warrant further investigation with industry. There is the potential for its adaption in other resource sectors.

University of Southern
Queensland Faculty of
Engineering and Surveying

ENG4111/2 *Research Project*

Limitations of Use

The Council of the University of Southern Queensland, its Faculty of Engineering and Surveying, and the staff of the University of Southern Queensland, do not accept any responsibility for the truth, accuracy or completeness of material contained within or associated with this dissertation.

Persons using all or any part of this material do so at their own risk, and not at the risk of the Council of the University of Southern Queensland, its Faculty of Engineering and Surveying or the staff of the University of Southern Queensland.

This dissertation reports an educational exercise and has no purpose or validity beyond this exercise. The sole purpose of the course pair entitled "Research Project" is to contribute to the overall education within the student's chosen degree program. This document, the associated hardware, software, drawings, and other material set out in the associated appendices should not be used for any other purpose: if they are so used, it is entirely at the risk of the user.

Dean
Faculty of Engineering and Surveying

Certification

I certify that the ideas, designs and experimental work, results, analysis and conclusions set out in this dissertation are entirely my own effort, except where otherwise indicated and acknowledged.

I further certify that the work is original and has not been previously submitted for assessment in any other course or institution, except where specifically stated.

Aaron Parry

Student Number:
0050025590

Signature

29/10/2009

Date

Acknowledgments

I would like to acknowledge the efforts of my project supervisors, Mr. Kerry Nufer and Dr Alexander Kist, for their support and direction in working through this project.

My wife Sarah for her understanding and support, my children Storm, Skye, Kiara, and Ethan deserves thanks as well for putting up with me being away from home for extended periods.

Table of Contents

Abstract ii

Acknowledgments v

Table of Contents vi

Table of Figures ix

List of Tables x

Glossary of Terms & Definitions xi

Chapter 1 Introduction 1-1

1.1 Project Description 1-1

1.2 Intellectual Property 1-1

1.3 Scope of Project 1-1

1.5 Project Methodology 1-2

1.6 Background 1-2

1.7 The Problem 1-3

1.8 A Possible Solution 1-3

1.9 Project Goals 1-3

1.10 Dissertation Outline 1-5

Chapter 2 Description 2-1

2.1 Literature Review 2-1

2.2 Consequential effects/implications/ethics 2-2

2.3 Risk Assessment 2-2

2.4 Resource Planning 2-2

2.5 Conclusion 2-3

Chapter 3 RADAC DESIGN 3-1

3.1 RADAC Block Diagram 3-1

3.2 Aerial Interrogation System (AIS) 3-1

3.3 Ground Transponder Unit 3-2

3.4 RADAC Method of Operation 3-4

3.5 RADAC GTU Implementation Options 3-6
3.6 Dedicated Terrestrial Applications 3-6

Chapter 4 General Design 4-1

4.1 Simplicity 4-1
4.2 Adaptability 4-1
4.3 Cost 4-1
4.4 Ground Transponder Unit (GTU) Assembly 4-2
4.5 Printed Circuit Board 4-2
4.6 Microcontroller Unit (MCU) 4-2
4.7 Memory 4-5
4.8 Wireless Transceiver Module 4-6
4.9 Camera Module 4-7
4.10 Antenna 4-9
4.11 Radome 4-15
4.12 Radome Enclosure Adaptor 4-16
4.13 Installation 4-16
4.14 Aerial Interrogation System 4-17
4.15 Flooding Routing Protocol 4-18
4.16 Operating System 4-19
4.17 Software Architecture 4-20
4.18 Sensor 4-21
4.19 Network 4-23

Chapter 5 Specification 5-1

5.1 Common Specification 5-1
5.2 Ground Transponder Unit (GTU) 5-3
5.3 Aerial Interrogator Unit (AIS) 5-4

Chapter 6 Conclusions 6-1

Further work to be Completed 6-2

References I
Appendices i
Appendix A i

Project Specification ii

Appendix B i

Calculations i

Electrical Current Usage of Circuit i

Solar Panel Calculations i

Beam Width and Gain Calculations ii

Antenna Calculations and Losses vi

Component Analysis xi

Appendix C i

Camera Code i

Appendix D i

Datasheets i

C328R Datasheet .... Pg 88-91i

C328R User Manual .... Pg 92-104i

C328-EV232 EV BOARD .... Pg 105i

Zigbee Applications Framework Whitepaper .... Pg 106-112i

AT45DB321D Datasheet .... Pg 113-166i

ATMEGA644PV Datasheet .... Pg 167-194i

Helical_Feed_Antennas Datasheet .... Pg 195-217i

Survey of Helical Antennas Datasheet .... Pg 218-240i

The Axial Mode Helix _etd_CH2 Datasheet .... Pg 241-254i

XbeeProZB Datasheet .... Pg 255-392i

Manual_xb_oem-rf-modules_802.15.4_v1.xAx ....Pg 393-464i

Table of Figures

Figure 1 Aerial Interrogation System –Block Diagram 3-2

Figure 2 Ground Transponder Unit –Block Diagram...................... 3-4

Figure 3 MCU.. 4-3

Figure 4 ATmega644PV microcontroller 4-4

Figure 5 Master/Slave Mode Communication.............................. 4-5

Figure 6 Pin Configuration of Data Flash Module.......................... 4-6

Figure 7 Data Flow in UART Interface Environment...................... 4-7

Figure 8 Wireless Transceiver Module 4-7

Figure 9 Block Diagram of Camera Module 4-8

Figure 10 Camera 4-8

Figure 11 Overview of materials used and dimensions. 4-11

Figure 12 Stub Mounting 4-12

Figure 13 Almost finished helix antenna. 4-12

Figure 14 Finished 12 turn 2.4 GHz helix antenna 4-13

Figure 15 Return loss (dB) from 2300 - 2500 MHz..................... 4-13

Figure 16 Smith chart 2300 - 2500 MHz.................................. 4-14

Figure 17 Ground Transponder Unit (GTU) Assembly 4-15

Figure 18 Aerial Interrogation System.................................... 4-17

Figure 19 Layering Structure of the Operating System 4-19

Figure 20 Data Flow 4-21

Figure 21 Modules Involve In Sensor Part................................ 4-21

Figure 22 Format of Image Data Package 4-23

Figure 23 Modules Involve In Network Part.............................. 4-24

Figure 24 Format of Network Packet 4-24

Figure 25 Format of UART Data Frame 4-25

Figure 26 Flow Chart of the Receiving Data Mechanism.............. 4-26

Figure 27 Risk Assessment xii

Figure 28 Camera Module Interface Schematic............................xiv

List of Tables

Table 1 Command Lists of the Camera Module 4-22

Table 2 Electrical current requirements of hardware i

Table 3 Antenna Calculations and Losses...................................... vi

Table 4 Component Analysis xi

Glossary of Terms & Definitions

AIS	Aerial Interrogation System
GTU	Ground Transponder Unit
POC	Proof of Concept
RADAC	Remote Aerial Data Acquisition & Capture
RF	Radio Frequency
UAV	Unmanned Aerial Vehicles
JPEG	Joint Photographic Experts Group; Also the name of the imaging standard they created.
PSNR	Peak Signal-to-Noise Ratio, a common measure of image quality, measured in dB.
SNR	Signal-to-Noise Ratio, a ratio of a signal to the underlying noise, measured in dB.
dB	Decibels, a measure of ratios.
ISO	International Standards Organization
Baud rate	The rate at which data flows
BCD	Binary Coded Decimal
Binary	Number system consisting of zeros and ones only
Bit	Binary Digit- smallest unit of storage in a computer
Byte	A group of eight data bits
CASA	Civil Aviation Safety Authority
CASR	Civil Aviation Safety Regulations
RTS/CTS	Request to send/ clear to send
SCI	Asynchronous Serial Communications Interface
SCLK	Clock signal

Chapter 1 Introduction

1.1 Project Description

The RADAC Project encompasses the design and prototype implementation of a system for low-cost aerial sensor data acquisition. It includes a Ground Transponder Unit (GTU), and Aerial Interrogation System (AIS). The proposed system is based on RF devices in association with a small low-cost single chip camera and microcontrollers. The GTU captures and transmits water-meter readings; the AIS initiates' measurements and processes and displays the results.

1.2 Intellectual Property

The concepts, design specifications, functionality and operation of RADAC are the property of NUFER & Associates (Aust) Pty Ltd, and the acceptance of these IP rights is necessary.

1.3 Scope of Project

The scope is limited to Proof of Concept (POC) in which the general philosophy of the design is proven to work, as a first step to the development of a commercial design by others. To this extent, the POC shall focus on specific aspects of the design which are identified below under the Project Goals.

1.5 Project Methodology

Since the project at this time is a ‘Proof-of-Concept’ the following methodologies are used:

The hardware components in relation to the actual antennas, main PCB, and mounting brackets are to be determined such that these enable the manufacturability of these components to be evaluated as a precursor to a final design – particularly in cost.

The actual electronics modules for the RF transceivers, camera module, and microcontrollers, are to be off-the-shelf modules, which utilise standard industry interfaces and allow the system functionality and algorithms to be developed in the ‘basic’ language.

Once the parameters for the air-segment link are determined, the antennas will be developed on an identical two-off basis so that these can be tested into each other on a range, and then testing confirmed using a standard dipole against each unit individually.

All hardware components for the proof-of-concept design will be able to be sourced across the counter at any hardware retail outlet.

1.6 Background

It was realised recently during a consultancy to a large Queensland government authority that the potential existed for considerable savings in the management of water resources, through the mechanisation of water meters.

1.7 The Problem

The organisation in question has somewhere in the vicinity of 8000 water meters in regional and remote parts of the state as shown in the photograph below. The units represent a considerable cost in terms of replacement and the human resource needed to read these at approximately three month intervals.

Further to this the replacement of the meters with digital units would not necessarily result in many savings due to the ongoing cost of reading the devices. Fitting the units with (RF) radio frequency modules to interface into terrestrial radio network (where within range) would also comprise a significant cost – both in terms of initial capital cost and ongoing operational costs

1.8 A Possible Solution

Based on the continued evolution of low-cost RF devices and associated technologies, NUFER & Associates (Aust) Pty Ltd have determined that a possible solution based on IEEE802.15.4 RF based devices (Zigbee, XBee) in association with a small low cost single chip camera and microcontroller.

1.9 Project Goals

The goals of the Project, which shall be reflected in the deliverables, shall comprise of the following –

Confirmation of the specifications by which a temporary radio link for the transmission of data at a particular speed may be achieved between a ground sensor (the GTU) and an overhead interrogation system (the AIS), including that the correlation of time versus altitude and antenna gain for the AIS to acquire, upload, and close down the connection with the GTU:

Demonstration of the RF subsystem performance in terms of gain, beamwidth, return loss, beamwidth, bandwidth, and matching of the antenna into the RF transceiver device:

Demonstration of the required software algorithms for this process to occur in a reliable and stable manner whereby:

The AIS is able to initiate and obtain contact with the GTU based on the fact it is within the target area determined under part (a);
The AIS is able to confirm that the established connection with the GTU is unique (based on the identification of the GTU), stable and secure;
The AIS is able to, via the GTU, supervise the capture of an image from the in-built camera system of the water supply gauge;
The AIS is able to, via the GTU, supervise the successful upload of the image data captured from the step described above;
The AIS is able to, via the GTU, upload extraneous data including battery status and temperature from the GTU;
The AIS is able to close down the connection with the GTU in a controlled manner at the end of the upload process so that it can be reinitialised at any subsequent time.
The GTU is able to carry out routine background tasks including in-built integrity analysis interspersed with ‘sleep’ processes that minimise power consumption to the utmost minimum:

Demonstration of an in-built power scavenging process which uses micro-solar cell (MSC) technology to float charge the inbuilt low-cost Lithium Ion batteries.

Demonstration of RADAC functionality through the remote capture and display of a water-meter dial display on a computer screen under the MS Windows XP system, as a result of the user initiating the capture process.

1.10 Dissertation Outline

This Dissertation comprises of six chapters. The first chapter briefly describe about background, problem statement, objectives and scope of the project.

The second chapter reviews previous related research that had been conducted. This chapter states the information which helps to develop the project.

The third chapter is about RADAC design. This chapter provides information about the components used to prepare the RADAC design of the project and why such components are chosen for this project.

The fourth chapter reviews the General Design. The detailed explanation about the hardware including the mechanism involved, and the function of every component.
The fifth chapter describes the specifications set out at the outset of the project, including the needs and wants for the project.

The last chapter which is chapter six is the conclusion. There are also suggestions for future work in order to improve the project.

Chapter 2 Description

2.1 Literature Review

Research has been conducted into various other methods of remote data acquisition. Of the available techniques RADAC fulfils all requirements set out in the achievable outcomes. Factors have been mentioned earlier in the report detailing project outcomes and how these need to be implemented to achieve these goals.

Remote Data Acquisition through Internet Based Telemetry- This experiment is to develop and implement an Internet Based Telemetry system. Telemetry is science of automatically measuring or recording data from one or more instruments and transmitting them over a distance. This study researched the transmission of data remotely acquisitioned over the internet. (See References)

Meter Readout System –
A method is presented for remote meter reading that addresses upgrade needs for automatic periodical readout of meters lacking electrical output signals. The solution embodied in the invention is a self contained, fully enclosed, low energy solution with provisions for onsite visual meter inspections. It can be implemented in various applications, such as but not limited to, water meters, electricity meters, gas meters, tachometers, and other meters. Methods are incorporated that enables the user full control over the timing and rate of the data acquisition, and control over system performance and power consumption. (See References)
Automatic Meter Reading (AMR)-
System wide monitoring using Automatic Meter Reading (AMR) for meters and sub-meters for energy management in utility applications. The traditional utility meter displays energy usage as an accumulation

of counts presented to a display, which is used to calculate the monthly bill. It has applications within the electricity, gas and water utility industries for domestic, commercial and industrial applications. Sub metering is often used within a building, retail or industrial facility where it is desirable to measure power consumption for specific equipment, locations or sub-level accounts. (See References)

2.2 Consequential effects/implications/ethics

If RADAC were to be implemented within the industry the consequential effects on the industry would increase productivity, decrease running costs and possibly increase technical knowledge of the maintenance employees. Implementing RADAC could allow for reduction in labour costs therefore allowing maintenance staff to be invested in other areas of the industry.

2.3 Risk Assessment

RADAC has been designed with many factors in mind regarding safety issues. One of the issues which have been assessed as a high risk is the use of light aircraft given the possibility of and probability of an aircraft incident. Even though this was deemed unlikely the consequences of this were fatal. Other risks which were identified include manual handling, strain injury, and abrasions. (See Risk Assessment Matrix in Appendix B)

2.4 Resource Planning

The resources required for RADAC have been sourced off the shelf where possible. Sources for the material used in the radon have been sourced from the local hardware store. The microcontroller was sourced from a local electronics store. The enclosure is off the shelf as well as other miscellaneous tools and equipment. The RADAC project was born out of the realisation of possible savings in the capture of information from legacy water meters.

2.5 Conclusion

The alternatives to RADAC are too cumbersome, technically unviable, reliant on further technology, unadaptable to various applications. The alternatives do not offer compatibility to all types of water meters regardless of location and size. They also are unadaptable to given climate and accessibility issues. The alternative remote meter reading technologies biggest drawbacks is the costs to implement in comparison to RADAC. The companies offer some comparison to RADAC but do not fulfil all of the achievable outcomes with the budget constraints of this project and deliver the outcomes specified. There have been many projects encompassing the use of the many separate components which are used in this project but these do not use these technologies to perform the allotted tasks set out in this project.

Chapter 3 RADAC DESIGN

3.1 RADAC Block Diagram

In order to achieve the required functionality, the RADAC design brings a number of components together in a low-cost housing and associated antenna as described below. Essentially it comprises two parts – an Aerial Interrogation System (AIS), and Ground Transponder Unit (GTU).

3.2 Aerial Interrogation System (AIS)

As shown in Figure 1, the Aerial Interrogation System comprises four components –

Executive Data Acquisition System – Typically running on a laptop or similar computer platform with an integrated database which contains the executive program responsible for supervising data acquisition and storing same into a database.

Interrogation Microcontroller – For detailed supervision of the RF transceiver device including acquisition and release of the transponder along with responding with the lap-top serial link.

Single-Chip RF Transceiver – For actual implementation of the interrogation process and data acquisition into the microcontroller.

Power Supply/ Battery – For independent powering of the AIS – Note this may be derived from the aircraft power system if appropriate.

High-Gain Helical Antenna – For direction of the RF energy towards the transponder using a circularly polarised RF signal and therefore removing any potential effects of depolarisation between transceiver antennas and therefore optimising gain over the resulting link.

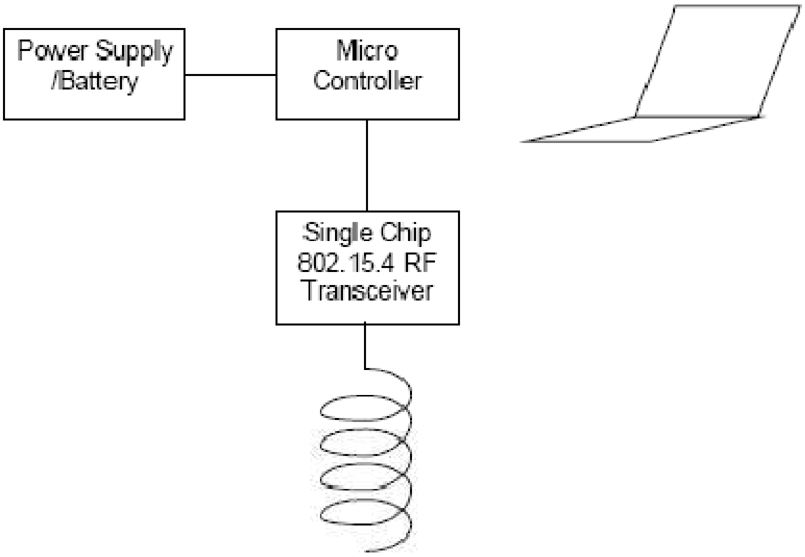


Figure 1 Aerial Interrogation System –Block Diagram

Following a session of water-meter message captures, the database would be processed to automatically generate evidentiary information for billing of water supply usage, and could include the actual image of the water meter concerned including the time and date of image capture as well as other information.

3.3 Ground Transponder Unit

A block diagram of the Ground Transponder Unit (GTU) is shown in Figure 2 and comprises five major components –

A High-Gain Helical Antenna – For direction of the RF energy towards the transponder using a circularly polarised RF signal and therefore removing any potential effects of depolarisation between transceiver antennas and therefore optimising gain over the resulting link.

Single Chip RF Transceiver – For communications and reception of commands from, and streaming of data back to the AIS; ZigBee, XBee uses IEEE 802.15.4 MAC and PHY layers aimed at simple, low-cost wireless communication networks, and lower power consumption than other wireless protocols such as Bluetooth due to its small size stack (about 28Kb). It also supports ease of installation, reliable data transfer, and short-range operation. Depending on the application, a system network can be designed as either a star or peer-to-peer topology which can be implemented as a mesh networking topology. They are determined by the controller, called the PAN coordinator.

Transponder Microcontroller – For Interpretation of AIS command data and subsequent supervision of response data stream following communications with the CMOS camera; it consists of a microcontroller unit - ATmega644PV is capable to operate with low power consumption. The MCU is active with 398uA current, 0.027uA in power-down mode, and 0.5uA in power-save mode. The operating voltage is from 1.8V to 5.5V. The sensor node operates with 3.3 V supply which is in the range of the MCU operating voltage.

CMOS Camera & Flash – Actual illumination and imaging of the meter display back to the supervising microcontroller. A C328 CMOS camera module was used for capturing an image. This component is designed for low cost, and low power solutions for high resolution image capture. It supports VGA/CIF/SIF/QCIF/ 160x128/80x64/3- 20x240 image resolutions. It also controlled with an RS-232 interface for setup and data transfer. The unit has 115.2Kbps bandwidth for transferring JPEG still pictures or 160x128 previews at 8bpp with

0.75-6 fps. It is operated using 3.3V, 60mA, and low standby current 100 µA.

Battery & Solar Cell – For ongoing float charging of the built-in battery and powering of the active electronic components contained there-in.

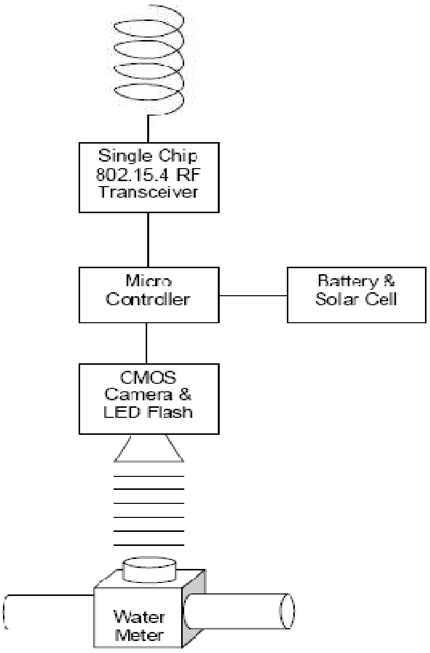


Figure 2 Ground Transponder Unit –Block Diagram

In actual operation the AIS is therefore able to interrogate GTUs and therefore obtain imaging of the water supply meter face, which is stored into the database of the supervising computer on-board the aircraft.

3.4 RADAC Method of Operation

Since water meters are scattered over large areas and are often difficult to access from the ground due to their location through locked gates and creek gullies etc. the use of an aerial vehicle for access and reading the water meter has many advantages.

In actual operation the following method of operation is therefore envisaged.

An aircraft would be fitted with an AIS and the coordinates of GTUs used as the basis for programming a flight path which incorporated a number of way points for the purpose of tracking over and uploading meter information.

In flight, the AIS would know where it was relative to each GTU based on independent information being obtained from a GPS receiver incorporated in the AIS.

On approaching the location of the GTUs the AIS would commence transmission of a ‘wake-up’ signal which would bring the GTU out of hibernation and ready for it to take a picture of the water meter dial.

On command, a picture would be taken of the water meter dial and the data uploaded to the AIS where the image would be stored in the database of the associated laptop computer.

Associated with this could be other information relating to the state-of-health of the GTU and any other information associated with the installation2 which may include temperature and weather records over the period since the last reading.
Once the information capture had been completed, the GTU is put back into hibernation, and the data saved into the data base of the laptop for analysis and report.

On completion of the information upload the aircraft is then able to track rapidly to the next site, or loiter in a holding pattern to recover other sites in the immediate area.

An extension to the GTU design would allow one transponder to become a ground mounted router, for capture of information from adjacent water meters using a local mesh, and therefore lower the cost of GTUs in a cluster situation.

3.5 RADAC GTU Implementation Options

The design of the RADAC GTU is not restricted to aerial information retrieval but may also be used for terrestrial applications or a mixture of both. In these cases a combination of antennas oriented in the required direction may be used to implement the design.

3.6 Dedicated Terrestrial Applications

In a dedicated terrestrial application Yagi antennas may be used to connect the site to a point of information collection in the vicinity of the site. Alternatively a combination of aerial space or helical antennas and local ground based communications antennas may be used for low-cost networking of data with a single router unit which is responsible for relaying data between an overhead AIS and the GTU router.

Chapter 4 General Design

It is important that the design take into account appropriate technologies and methodology of construction as part of manufacturing and installation on-site.

4.1 Simplicity

It is essential that the design be as outwardly simple as possible in that it comprises easily assembled structures and is easy to fit, and maintain in the field.

4.2 Adaptability

In addition the design must allow fitment to the large range of water meters in terms of host water pipe diameter, and gauge positioning so that a common kit-of-parts can be used for a significant part or the entire product.

4.3 Cost

The manufacturing cost must be low to leverage the increasingly low cost of electronic devices and make the product viable in the business case. At this time a manufacturing cost of $AUD50 is seen as the target for the Ground Transponder Unit (GTU) in large quantities.

In the case of the Aerial Interrogator System (AIS), the cost may be somewhat higher at around $AUD500 in recognition of the need to attain compliance for aviation installation.

4.4 Ground Transponder Unit (GTU) Assembly

At this time the GTU is seen as comprising 100mm PVC tube which contains the helical antenna for its entire length – connected at one end to a PCB within the lower end of the tube. (Possible change of enclosure material)

4.5 Printed Circuit Board

The upper side of the PCB will include the matching components and connection to the feed-point of the antenna. This feed point will extend through the PCB to the lower side which will be fitted with the RF transceiver chip and associated electronic components as well as the camera.

4.6 Microcontroller Unit (MCU)

MCU is the core component in every single sensor node. The functions of this component is to control the activity of the sensor node including data processing, serial communication which is controlling other devices through several processes including hardware interface, collecting data from devices or send data to the devices, acknowledgement and so on.

ATmega644PV, produced by Atmel Corporation is chosen as the MCU of the sensor node. It consists of two programmable serial USART required for interfacing with camera module and wireless transceiver module. Basically, USART operates by taking bytes of data and transmit it to another USART bit by bit sequentially. Another USART receives bits and reconstructs the bits to form complete bytes.

ATmega644PV is capable to operate with low power consumption. The MCU is active with 398uA current, 0.027uA in power-down mode, and 0.5uA in power-save mode. The operating voltage is from 1.8V to 5.5V. The sensor node operates with 3.3 V supply which is in the range of the MCU operating voltage.

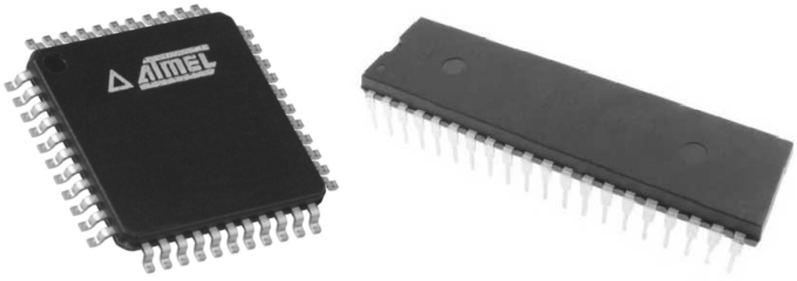


Figure 3 MCU

In addition, based on the device datasheet, the MCU consists of 64kbytes of In-System Self-Programmable Flash which compatible for higher demand of the RAM size especially for the uses of an operating system. The following Figure 4 shows the photo of ATmega644PV and showing the pin configuration of ATmega644PV microcontroller.

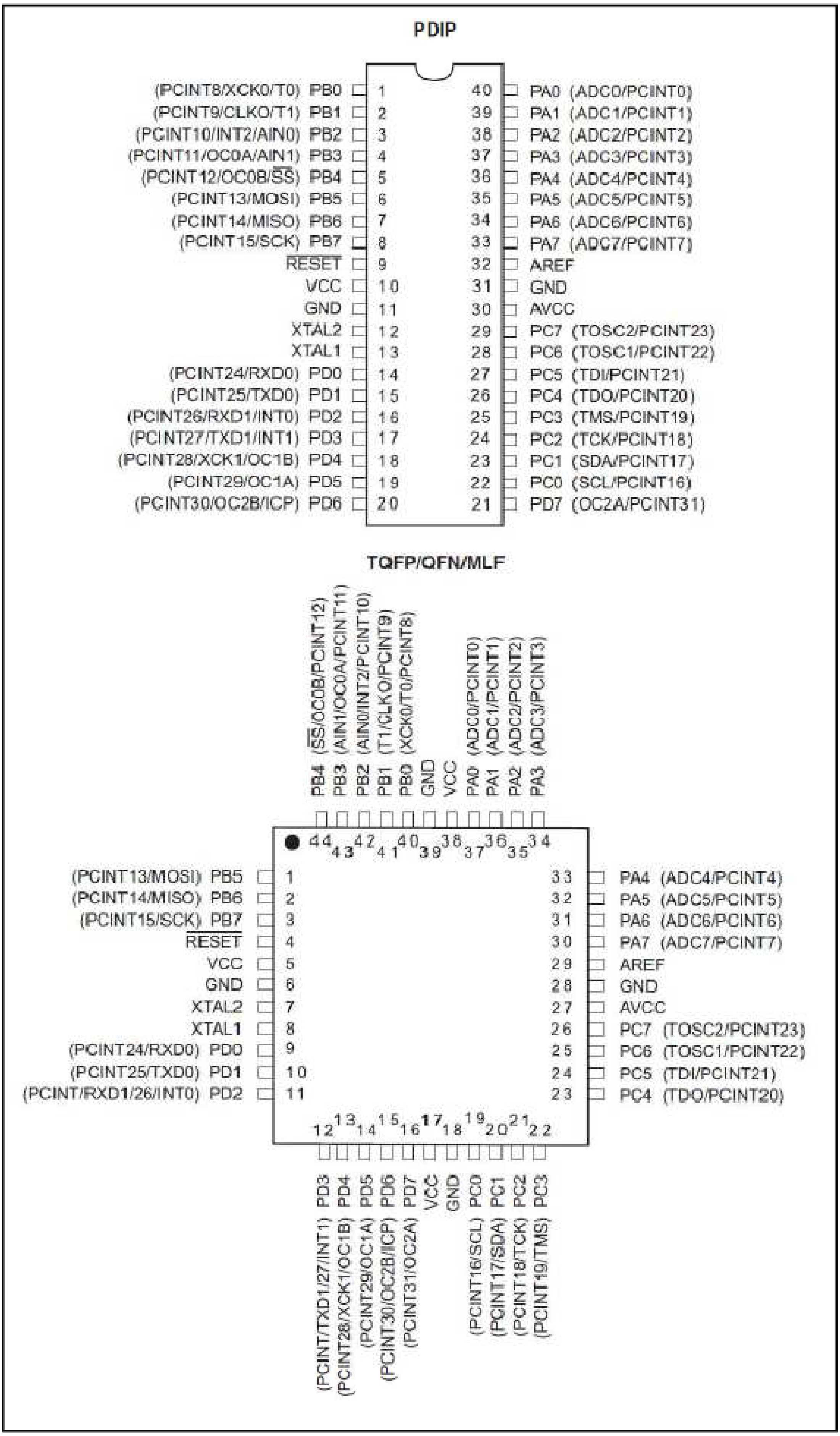


Figure 4 ATmega644PV microcontroller

4.7 Memory

The requirement of larger memory spaces to store temporary data from MCU is solved using AT45DB321D data flash produced by Atmel Corporation. The device operates with single supply from 2.7V to 3.6V. The 3.3V supply of the sensor node is in the range of the operating voltage of the data flash. The device also operates with low power dissipation where 7mA active read current, 25KA standby current and 5KA deep powers down.

The data flash communicates with MCU via full duplex synchronous serial data link known as Serial Peripheral Interface (SPI). The communications include input/output, device selection and clock is done through four pins connected to MCU which are Serial Input (SI), Serial Output (SO), Chip Select (CS) and Serial Clock (SCK). The communication is in Master/Slave mode where MCU function as the master while the data flash function as slave as shown in Figure 5 below.

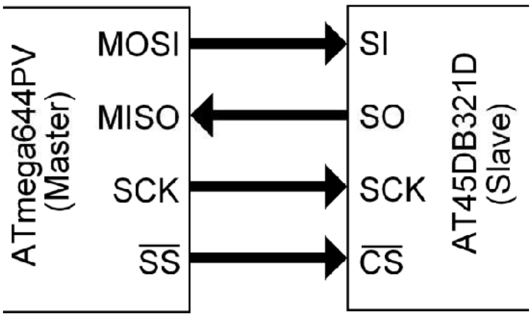


Figure 5 Master/Slave Mode Communication

The capacity of AT45DB321D data flash is 32 megabit or equal to 4 Mbytes. The device has user configurable page size whether 512 bytes per or 528 bytes per page. Based on the device datasheet, it consists of 34,603,008 bits of memory which are organized as 8,192 pages of 512 bytes or 528 bytes each depending on the page size configuration. The following figure, Figure 6 shows the pin configuration of AT45DB321D data flash.

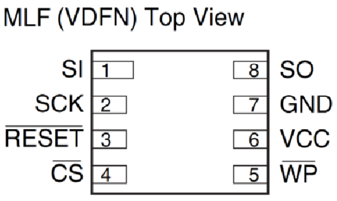


Figure 6 Pin Configuration of Data Flash Module

4.8 Wireless Transceiver Module

Communication between sensors node is done through the Zigbee, XBee OEM RF modules manufactured by Maxstream. It is manufactured to operate in compliance with IEEE 802.15.4 standard with several features in order to obtain low cost and low power sensor node specifications. Transmitting and receiving data are done within ISM 2.4GHz frequency band with 250kbps data rate.

The transceiver is able to operate in 4 modes which is transmit mode, receive mode, idle mode and sleep mode which lead to power saving. Furthermore, it operates at 3.3V with 45mA TX current, 50mA RX current and less than 10uA power down current. The transmitted power is 1mW which is equal to 0dBm. The signal coverage radius is up to 30m at urban or indoor and up to 100m at outdoor line-of-sight.

As stated previously, the wireless transceiver module communicates with MCU through USART. Figure 7 illustrates the system data flow in UART interface

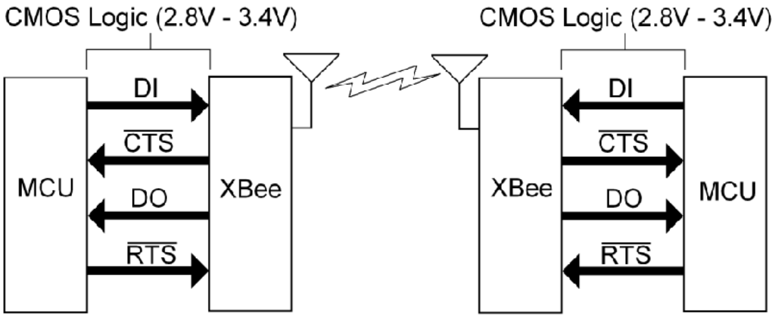


Figure 7 Data Flow in UART Interface Environment

environment. From the Figure, DI stand for Data In, DO stand for Data Out while both CTS and RTS are Hardware Flow Control. Figure 8 shows the photo of XBee Module.

Figure 8 Wireless Transceiver Module

4.9 Camera Module

In order to retrieve graphical information in the sensor field, C328R JPEG Camera Module is used. This camera is VGA camera module consists of JPEG compression engine or JPEG CODEC which called OV528 Serial Bridge. This module also has built-in serial type program memory to store a set of command interfacing to MCU. When the camera receives snapshot command from MCU, a single-frame still picture with VGA resolution will be captured. Then the picture will be compress by JPEG CODEC and it is sent to MCU through serial UART.

The features of C328R includes small in size where its dimension is 20x28mm, operation with 3.3V operating voltage and low power consumption which is 60mA. The UART interface speed can be up to 115.2kbps and able to generate several size of JPEG compression images from 80x64 pixels up to 640x480 pixels. Figure 9 illustrate the block diagram of the camera module and follow by Figure 10 showing the photo of C328R.

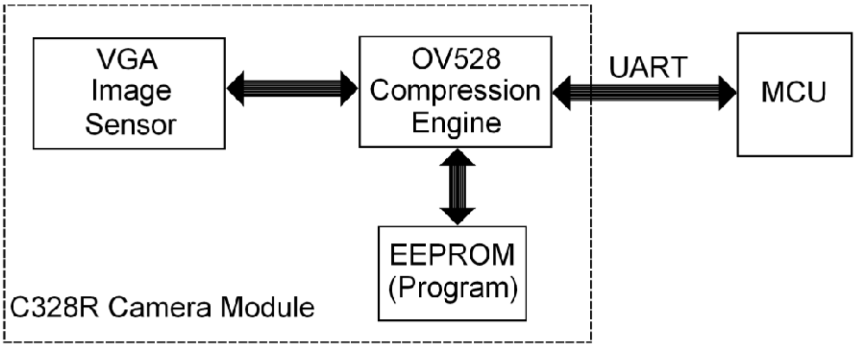


Figure 9 Block Diagram of Camera Module

Figure 10 Camera

4.10 Antenna

A helical antenna has been identified as the most appropriate for the design, owing to the fact that a reproducible gain device is required for manufacturing purposes. As an option a 4 x patch array would be an alternative suggestion, however, it is thought that this may be too costly and required too-large an area to achieve the same gain as the helical.

The helix antenna, invented in the late forties by John Kraus (W8JK), can be considered as the genius ultimate simplicity as far as antenna design is concerned. Especially for frequencies in the range 2 - 5 GHz this design is very easy, practical, and, non critical. This contribution describes how to produce a helix antenna for frequencies around 2.4 GHz which can be used for e.g. high speed packet radio (S5-PSK, 1.288 Mbit/s), 2.4 GHz wavelengths, and, amateur satellite (AO40). Developments in wavelengths equipment result in easy possibilities for high speed wireless internet access using the 802.11b (aka WiFi) standard. The helix antenna can be considered as a spring with ***N*** turns with a reflector. The circumference (***C)*** of a turn is approximately one wavelength (***l)***, and, the distance (***d)*** between the turns is approx. 0.25*C*. The size of the reflector (***R)*** is equal to *C* or *l*, and can be a circle or a square. The design yields circular polarization (CP), which can be either 'right hand' or 'left hand' (RHCP or LHCP respectively), depending upon how the helix is wound. To have maximum transfer of energy, both ends of the link must use the same polarization, unless you use a (passive) reflector in the radio path.

The gain (*G*) of the antenna, relative to an isotropic (dBi), can be estimated by:

$$G = 11.8 + 10 * \log\{(C/l)^{\wedge}2 * N * d\}\,\mathrm{dBi} \qquad (1)$$

4-9

According to Dr. Darrel Emerson of the National Radio Astronomy Observatory, the results from [1], also known as the 'Kraus formula', are 4 - 5 dB too optimistic. Dr. Ray Cross inserted the results from Emerson in an antenna analysis program called 'ASAP'. The characteristic impedance (*Z)* of the resulting 'transmission line' empirically seems to be:

$$\mathrm{Z} = 140 * (\mathrm{C/l})\ \mathrm{Ohm} \qquad (2)$$

Practical design for 2.43 GHz

$$\mathrm{l} = (0.3/2.43) = 0.1234567\ \mathrm{m}\ ;-)(\ 12.34\ \mathrm{cm}) \qquad (3)$$

$$\text{The diameter (D) of one turn} = (\mathrm{l/pi}) = 39.3\ \mathrm{mm} \qquad (4)$$

Standard PVC sewer pipe with an outer diameter of 40 mm is perfect for the job and can be obtained easily from a 'do it yourself' shop or a plumber. The helix will be wound with standard wire used to interconnect 240V AC outlets households. This wire has a colourized PVC isolation and a 1.5 mm thick copper core. Winding it around the PVC pipe will result in *D* = ca. 42 mm, due to the thickness of the isolation.

$$\text{With D} = 42\ \mathrm{mm}, \mathrm{C} = 42 * \mathrm{pi} = 132\ \mathrm{mm}\ (\text{which is } 1.07\ \mathrm{l}) \qquad (5)$$

$$\text{Now d} = 0.25\mathrm{C} = 0.25 * 132 = 33\ \mathrm{mm} \qquad (6)$$

For distances ranging from 100 m - 2.5 km **with line of sight**, 12 turns (*N* = 12) are sufficient. The length of the PVC pipe therefore will be 40 cm (3.24 *l*). Turn the wire around the PVC pipe and glue it with PVC glue or any other glue containing tetrahydrofurane (THF). The result will be a very solid helix wound along the pipe, see Figure 11 below.

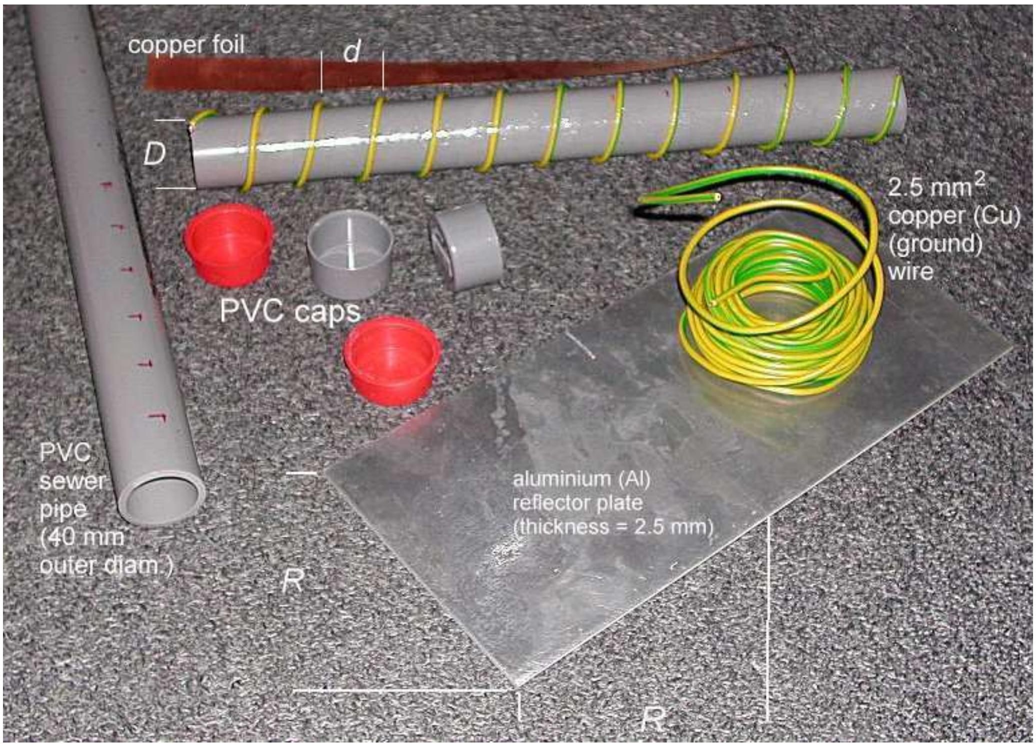


Figure 11 Overview of materials used and dimensions.

The impedance of the antenna, which is:

$$Z = 140 * (C/l) = 140 * \{(42 * pi)/123.4\} = 150\,Ohm \tag{7}$$

It requires a matching network in order to apply standard 50 Ohm UHF/SHF coax and connectors.
The use of a 1/4-wave matching stub with an impedance (*Zs*) of:

$$Zs = sqrt(Z1 * Z2) = sqrt(50 * 150) = 87\,Ohm \tag{8}$$

is very common. Due to the helix design, this equals 1/4 turn. My first thoughts were to empirically decrease *d* for the first and second turn and match the helix using the 'trial and error'-method, while measuring the results with a directional coupler, and signal generator. For details, see Figure 12 (below).

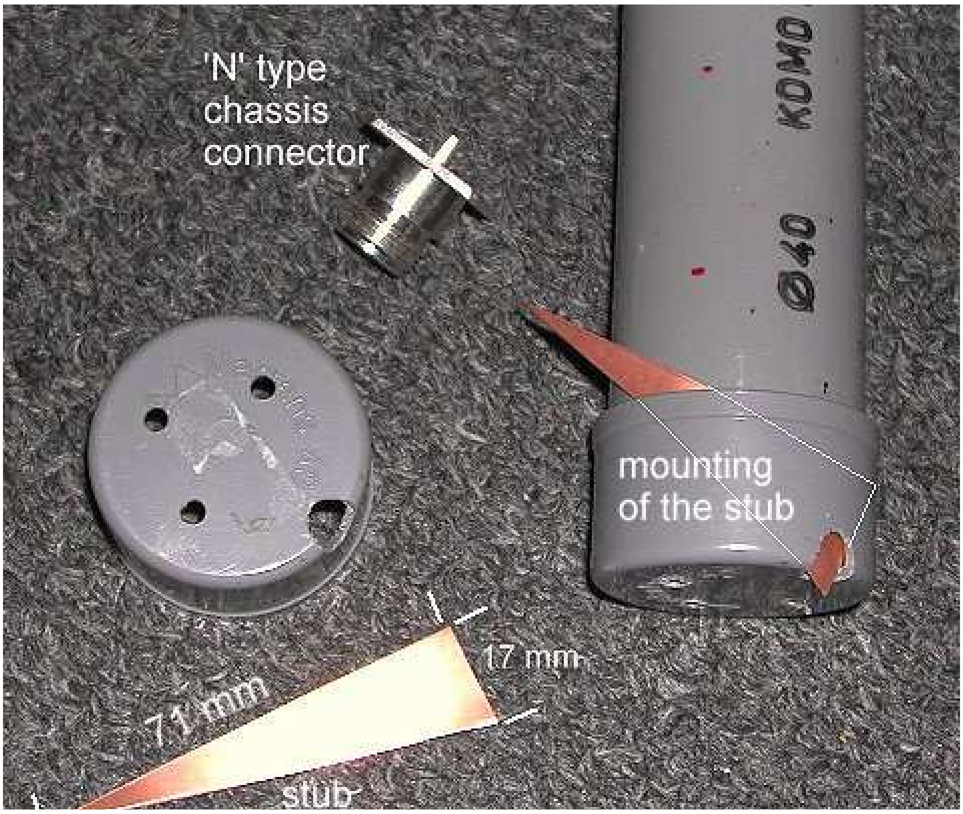


Figure 12 Stub Mounting

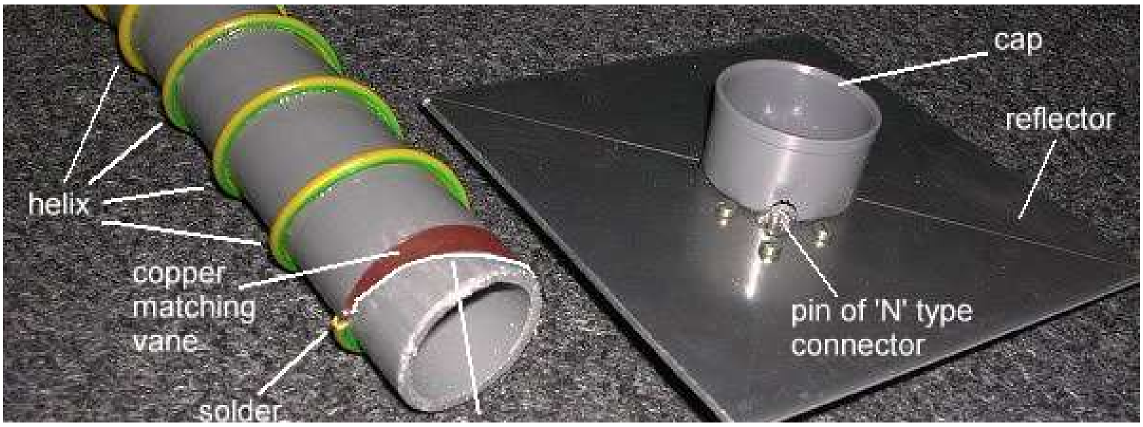


Figure 13 Almost finished helix antenna.

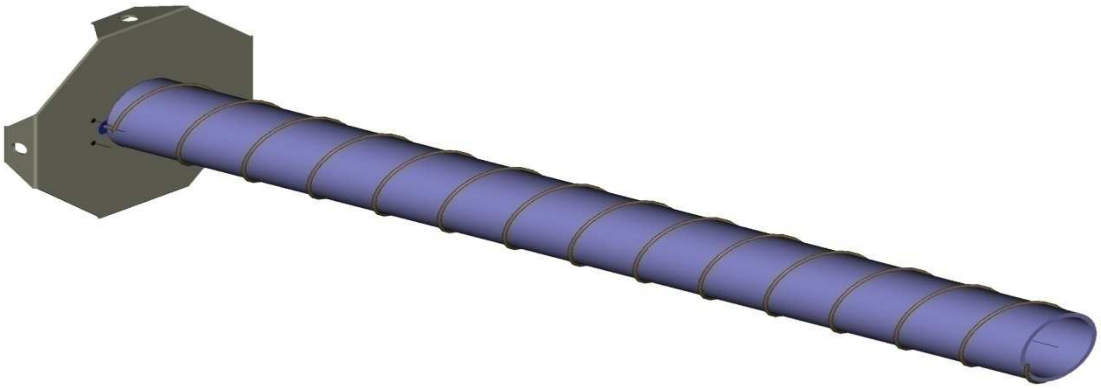


Figure 14 Finished 12 turn 2.4 GHz helix antenna

The antenna was swept and measured. The results are given below

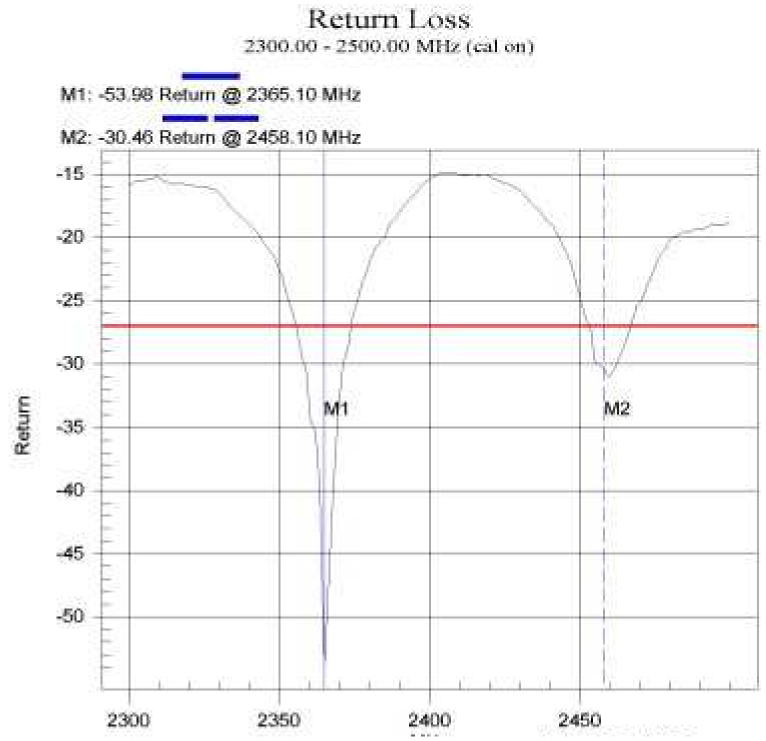


Figure 15 Return loss (dB) from 2300 - 2500 MHz

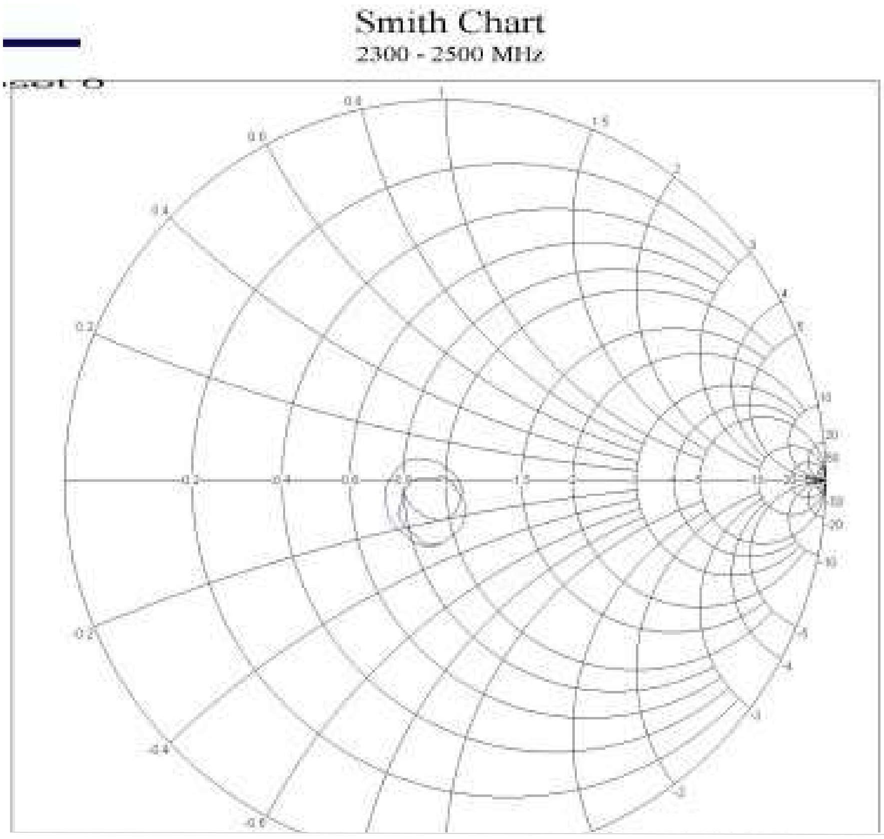


Figure 16 Smith chart 2300 - 2500 MHz

4.11 Radome

The antenna is enclosed in the PCB tube which not only supports the antenna but also provides the main package of the product, and is effectively the objective of the manufacturing process. A nominal length of 500 mm has been identified for the length of this housing; however shorter housings may be possible with lower gain. This aspect of the proposal must be checked during the design and correlated against the benefits – especially altitude or free-space loss and system gain.

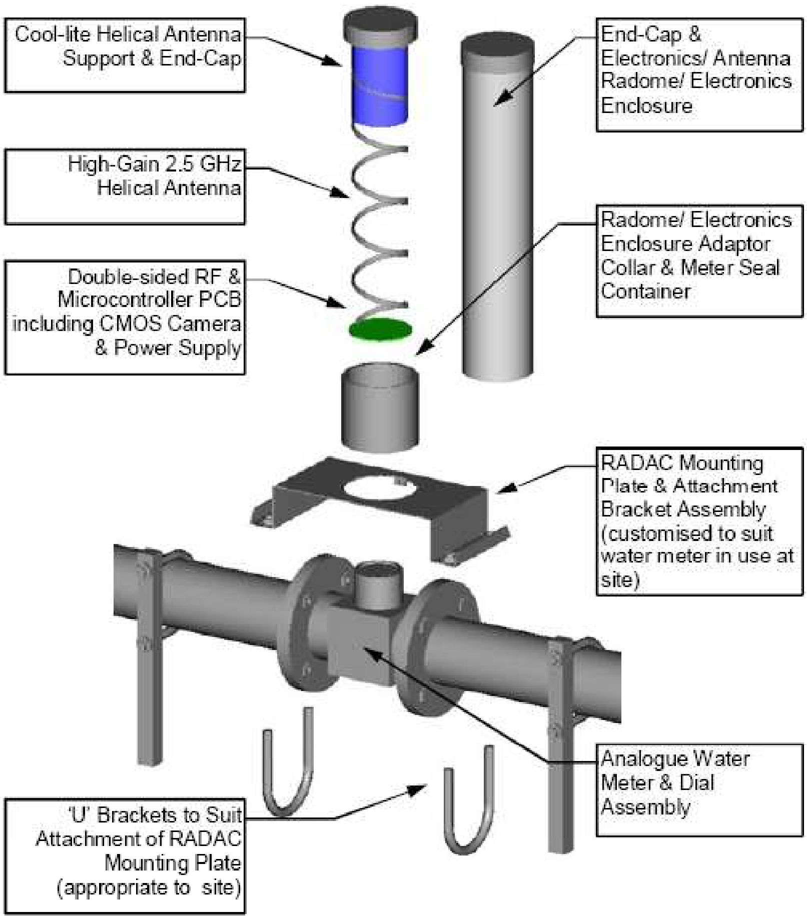

Cool-lite Helical Antenna Support & End-Cap
High-Gain 2.5 GHz Helical Antenna
Double-sided RF & Microcontroller PCB including CMOS Camera & Power Supply
End-Cap & Electronics/ Antenna Radome/ Electronics Enclosure
Radome/ Electronics Enclosure Adaptor Collar & Meter Seal Container
RADAC Mounting Plate & Attachment Bracket Assembly (customised to suit water meter in use at site)
Analogue Water Meter & Dial Assembly
'U' Brackets to Suit Attachment of RADAC Mounting Plate (appropriate to site)

Figure 17 Ground Transponder Unit (GTU) Assembly

4.12 Radome Enclosure Adaptor

The radome package together with the end-cap and electronics package at the bottom, is located into a radome enclosure adaptor, and fitted at a height and in such a direction as to facilitate illumination of the solar cell (not shown).

Generally, any installation would proceed with the installation of the mounting plate on the pipe above the water meter to produce a working platform for installation of the adaptor and subsequent antenna/PCB package.

4.13 Installation

When installed correctly the mounting plate is allowed to descend down and around the water meter dial assembly. To stop the ingress of vermin, a coiled spiral of high-density foam is placed around the meter dial, to create the seal.

With the adaptor secured, the combination radome /electronics module is then lowered over the coiled tape which is compacted to increase the resilience to outside vermin. The radome assemble is now lowered down and to an appropriate height for the camera to operate correctly. At this point the assembly is sealed, and put into operation. As discussed earlier, the radome may able to be shortened however this will depend heavily on the link budget objective, which is part of this design process.

4.14 Aerial Interrogation System

To a large extent the AIS is very similar to the GTU with the exception that this device is located upside down as shown in Figure 18 Proposed AIS Assembly. As with the GTU the antenna is connected directly to the PCB to minimise losses, but unlike the GTU, the whole assembly is complete. The assembly shown in actual production and for the prototype may comprise the antenna mounted on an enclosed metal box, and mounting plate for subsequent attachment to the under-belly of the aircraft, or internally where this the construction of the aircraft is of fibreglass.

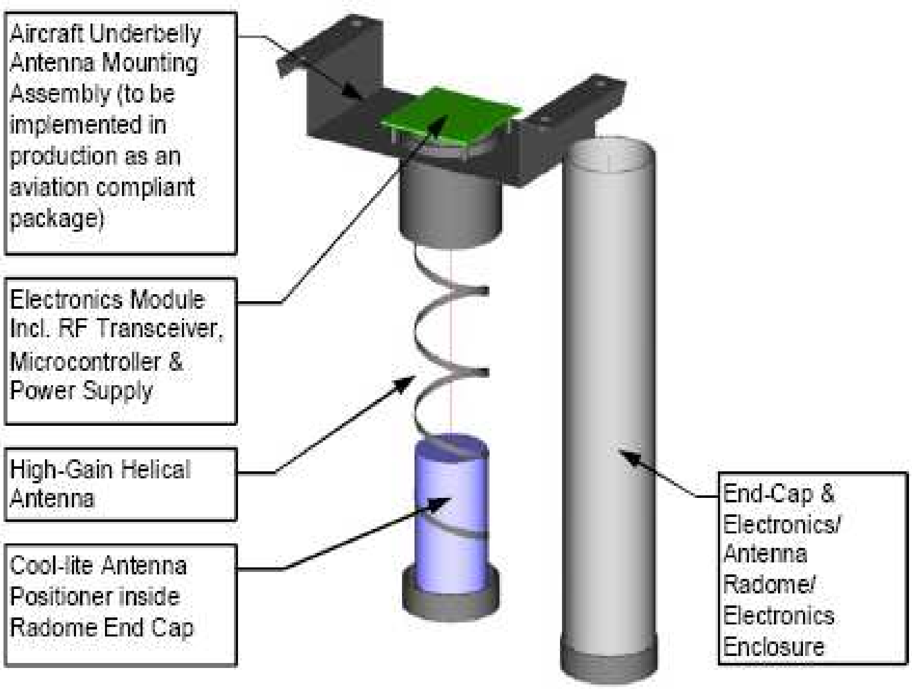

Aircraft Underbelly Antenna Mounting Assembly (to be implemented in production as an aviation compliant package)
Electronics Module Incl. RF Transceiver, Microcontroller & Power Supply
High-Gain Helical Antenna
Cool-lite Antenna Positioner inside Radome End Cap
End-Cap & Electronics/ Antenna Radome/ Electronics Enclosure

Figure 18 Aerial Interrogation System

4.15 Flooding Routing Protocol

Flooding is one of the common routing protocols used in wireless network. Basically the node in the network would have the ability to examine the packet received in order to determine where the packet should go. If the received packet is not destined to it, then the packet will be rebroadcasted. The protocol also is designed to be able to remember the packet that has been received so that when the same packet is received again, it can be simply dropped. Based on flooding routing protocol, the node will broadcast every single packet received. Hence, how about the network traffic in a field consists of large number of node where the nodes are broadcasting a packet? The network traffic will be highly congested and this is why the characteristic stated above must be acquired by the nodes in the network.

The flooding routing protocol has several advantages. Firstly, it is guaranteed that the packet sent reach at the destination because based on the protocol, all possible routes between source and destination will be tried. However, there should be at least one path that connecting the source and destination. Secondly, since all routes are tried, it is possible to find a minimum-hop route and shortest route that can be used to setup the virtual circuit route. Lastly, broadcast technique used will ensure that all nodes are visited. This is a useful attribute when there is a case where important information is need to be distributed to all nodes. The drawback of this technique is that it is too exhaustive. The retransmission occurred will greatly deplete the power source and shortened the network lifetime.

4.16 Operating System

Basically, OS in sensor node function as an interface between application and hardware which offer several services related to the applications of the sensor node. In this project, the OS developed by TRG, UTM which is called as ANOS is used.

The applications software is designed base on ANOS. Services provides by ANOS is accessed through application programming interfaces (API). After receiving command from user, the interface will be called by applications software to request certain services from the OS. Then, certain parameters will be passed and the result from related operation will be obtained. Figure 19 illustrates the layer structure of the OS.

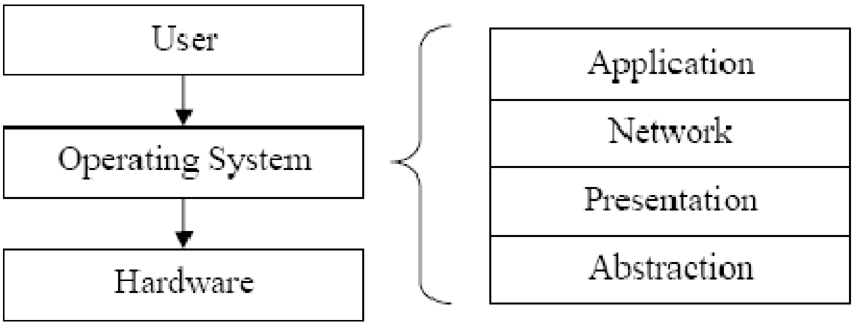


Figure 19 Layering Structure of the Operating System

The interfacing between user and hardware is made through four layers of the OS which are application layer, network layer, presentation layer and abstraction layer.

Application layer basically is the nearest layer that connects the end user to the system. Generally, there will be certain applications software that enable user to communicate, give commands to the sensor node and provide result to the user. In this project, the data is received from presentation layer. The application layer function is to provide the data to the end application software on the computer known as Graphic User Interface (GUI). GUI will process the received data, generate image from the data in JPEG format which is stored in computer hard drive and finally display the image at computer screen.

Network Layer in this project is designed for node-to-node and end-to-end packet delivery. Here, flooding routing protocol is used. Based on the conditions as stated in literature review, every sensor node will rebroadcast the received data until the data is delivered at the end node.

As stated before, information received by application layer came from presentation layer. The information is actually being formatted by presentation layer to a format recognized by application layer. In short, presentation layer could be analogous as translator who will translate the information received to something understood by application layer. Generally, this layer is where the encryption data (if any) is being read and reformatted for upper layer process.

The abstraction layer plays the role as a functions adapter between hardware and presentation layer and it is hardware dependence. Here, the presentation layer functions will fixed although the hardware use is change. This is because the abstraction layer will generalize the hardware functions to a specific command used by presentation layer. The uses of the abstraction layer will simplify the presentation layer functions and at the same time make the presentation layer to be hardware independence.

4.17 Software Architecture

The operating system is built up by three parts including sensor part, network part and application part as illustrated in Figure 20 below. Each part consists of several modules with certain function which will be discussed later. In general, sensor part is the part where the data is gathered, and network part is responsible for data delivery through the network while all data processing is done in application part.

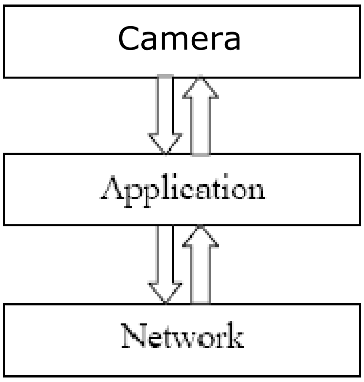


Figure 20 Data Flow

4.18 Sensor

Sensor part is responsible for image data collecting process which is communication with hardware (C328R camera module). The functions include issuing command, receiving data and passing data from sensor part to application part. The modules involve in sensor part are known as USART1, SERIAL1 and CAMERA. US ART 1 and SERIAL 1 is the kernel of the operating system. Both of the kernel functionality is basically to pass data from one module to another through certain process. Figure 21 below shows how the modules in sensor part are linked.

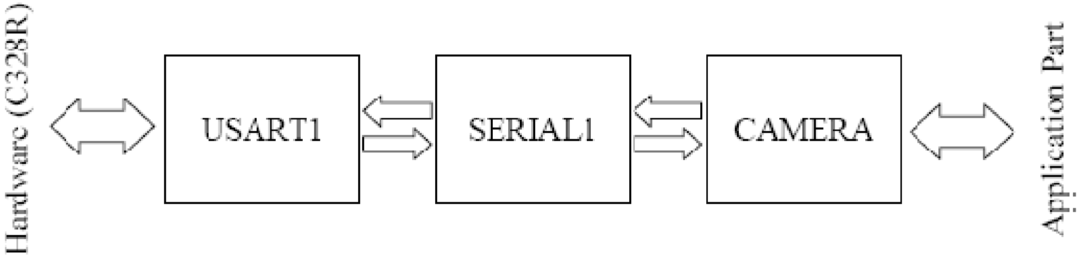


Figure 21 Modules Involve In Sensor Part

CAMERA module is the one that responsible for communication with the hardware. Basically the OS will communicate with C328R using 6 bytes command set which is started with sync command. Table 1 shows the list of command set for C328R.

Table 1 Command Lists of the Camera Module

Command	ID Number	Parameter 1	Parameter 2	Parameter 3	Parameter 4
Initial	AA01h	00h	Color Type	RAW Resolutio	JPEG Resolution
Get Picture	AA04h	Picture Type	00h	00h	00h
Snapshot	AA05h	Snapshot Type	Skip Frame	Skip Frame	00h
Set Package	AA06h	08h	Package Size Low	Package Size High	00h
Set Baud Rate	AA07h	1st Divider	2nd Divider	00h	00h
Reset	AA08h	Reset	00h	00h	xxh*
Power Off	AA09h	09h	09h	09h	09h
Data	AA0Ah	Data Type	Length Byte 0	Length Byte 1	Length Byte 2
SYNC	AA0Dh	00h	00h	00h	00h
ACK	AA0Eh	Command ID	ACK Counter	00h / Package	00h / Package
NAK	AA0Fh	00h	NAK Counter	Error Number	00h
Light Frequency	AA13h	Frequency Type	00h	00h	00h

The sync command will be sent repeatedly until acknowledgement is received from C328R. After that, initial command consists of several parameters which are Color Type, RAW Resolution and JPEG Resolution, will be sent to C328R. Then, Set Package Size command will be issued to C328R to set the size of packet that will be generated by the camera before it is sent to MCU. Figure 22 below illustrates the format of image data package that will be generated.

Byte 0			Byte N
ID (2 bytes)	Data Size (2 bytes)	Image Data (Package size - 6 bytes)	Verify Code (2 bytes)

Figure 22 Format of Image Data Package

After that, Get Picture command will be sent to the hardware. Once Get Picture command is received, C328R will start capturing image. The image data is passed to MCU byte by byte and the module responsible to receive the data is USART1 module. The byte data is received by USART1 through serial communication and it is directly sent to SERIAL1 module. At SERIAL1 module, the data will first be queued in buffer before it is collected by camera module. Here, the data must be buffered due to multitasking process where the MCU will only collect the data once the task that instruct it to do so is reached. Otherwise, the data must be buffered first. The data collected will be sent to CAMERA module and finally the application part will collect the data to be processed. The same process will be gone through by every single byte of data received from C328R.

4.19 Network

Network part is responsible for data transmission path setup which is referring to communication between nodes to the end node which is done via hardware wireless transceiver module (XBee OEM RF Module). The modules involve in network part are known as USART2, SERIAL2, XBEE and NETWORK. USART2 and SERIAL2 are physically having the same function as USART1 and SERIAL 1 in sensor part where they are kernel of the operating system that will passing data from one module to another through certain process. Figure 23 below shows how the modules in network part are linked.

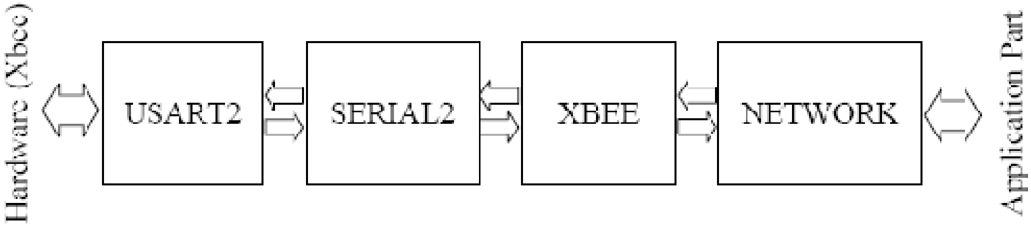


Figure 23 Modules Involve In Network Part

As stated before, the protocol used in the network is flooding protocol. Here XBEE and NETWORK modules will play their role in order to establish the flooding protocol since the protocol mechanism is apply in both modules. There are two data processing mechanisms in network part which are transmitting data mechanism and receiving data mechanism. Let's go through transmitting data mechanism first.

Transmitting data mechanism is the processing algorithms that start with collecting data from application part and send the data to hardware wireless transceiver to be transmitted over the network. Once a packet of data reaches at NETWORK module, it will be processed by appending the data packet with network protocol data unit (network PDU) as shows in Figure 24. After that, the packet will be sent to XBEE module. At this module, the packet will be appended with another PDU which is MAC PDU. Next, the packet will be passed to SERIAL2 module follow by USART2 module and finally the data packet will reach at hardware (XBee OEM) and it is then being transmitted through the network.

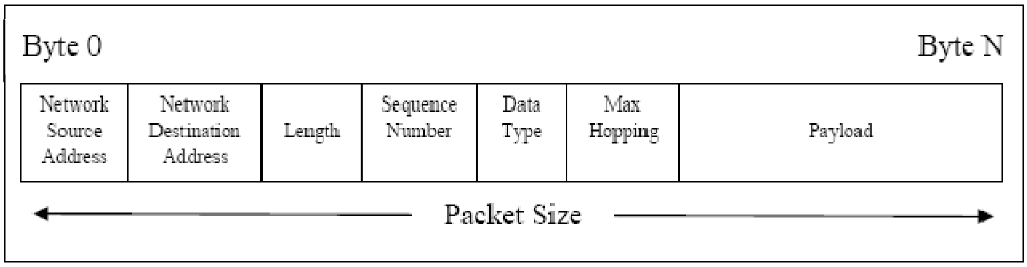
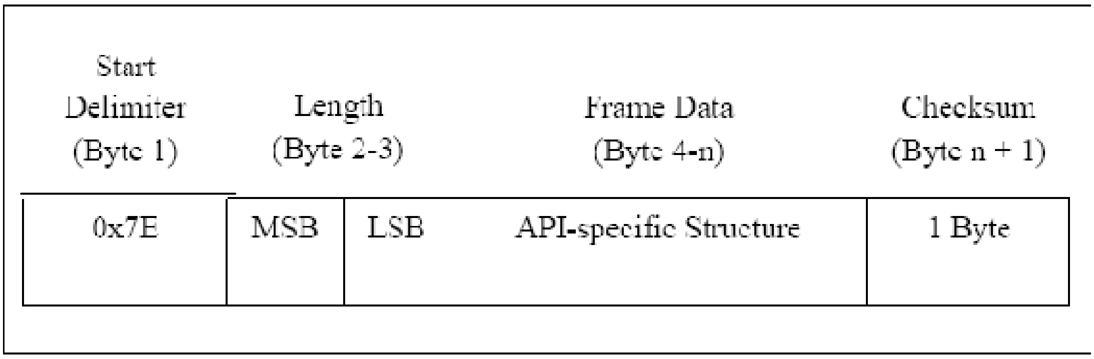


Figure 24 Format of Network Packet

At XBEE module, the data packet received from application part will be appended with MAC PDU according UART Data Frame Structure as illustrated in Figure 25. The frame structure is required for communication purpose where the hardware wireless transceiver (XBee OEM) is programmed to operate in Application Programming Interface (API) mode.



Start Delimiter (Byte 1)	Length (Byte 2-3)		Frame Data (Byte 4-n)	Checksum (Byte n + 1)
0x7E	MSB	LSB	API-specific Structure	1 Byte

Figure 25 Format of UART Data Frame

Receiving data mechanism is the processing algorithms started at USART2 module. When the transceiver of a node receives the data, it will send the data to MCU through USART2 module. Through the same process as discussed in sensor part for USART2 and SERIAL2, the data is next delivered to XBEE module. Here the module will examine the MAC destination address of the packet in order to verify whether the address is the same as its address or not. If the MAC destination address differs, then the packet will be dropped. However, if it is the same, then the packet will be passed to NETWORK module. At this module, the packet will be examined again by duplicate checker mechanism to ensure that the packet never reach at the node yet. If it does, then the packet will be dropped. After duplicate checker, the network destination address of the packet will be verified. If it is correct then the packet will be passed to application part. However, if there is wrong network destination address, then the packet will be rebroadcasted. Figure 26 shows the flow chart of the receiving data mechanism.

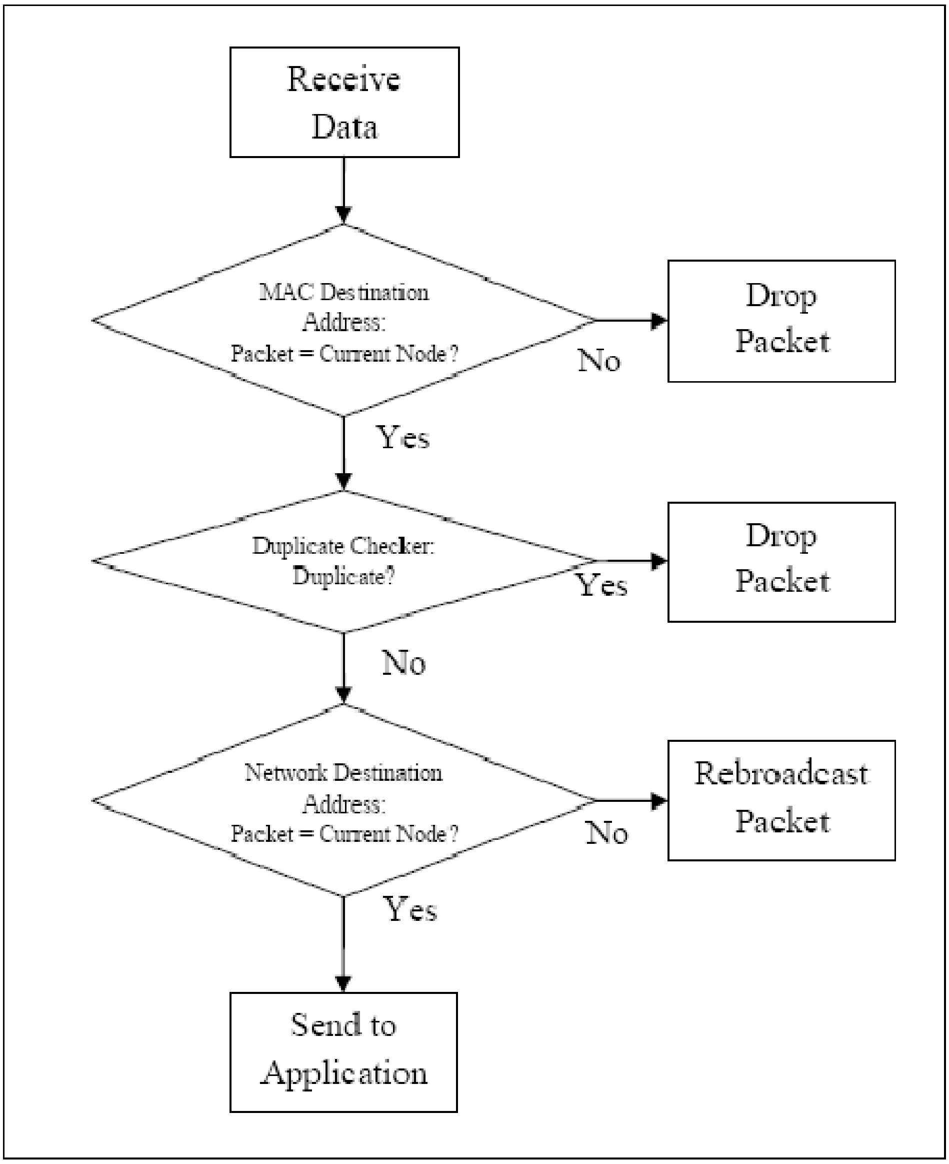

Receive Data
MAC Destination Address: Packet = Current Node ?
No
Drop Packet
Yes
Duplicate Checker: Duplicate?
Yes
Drop Packet
No
Network Destination Address: Packet = Current Node ?
No
Rebroadcast Packet
Yes
Send to Application

Figure 26 Flow Chart of the Receiving Data Mechanism

Chapter 5 Specification

The specification for RADAC units is as follows. Where a specification is not found for the particular component please refer to the common specification presented in Chapter 4.

5.1 Common Specification

Type: Remote RF data acquisition system incorporating low-power environmental energy scavenging systems for use in the ISM bands.

Frequency Band: 2.4000 GHz to 2.4835 GHz

Operating Channel: To be selected based on site and potential for self interference

Channel Bandwidth: Nominally 5 MHz

Antennas

Type: Helical with integral radome

Gain: 14 dBi (nominal)

Beamwidth: To be determined

Bandwidth: 2.4000 GHz to 2.4835 GHz

Polarisation: Left-Hand Circular

Impedance: 50 ohms (unbalanced)

Return Loss: 20 dB

Front-to-back Ratio: >30 dB

Protocol: Zigbee, XBee over IEEE802.15.4

Security: AES 128 Bit

Data Rate: 256 kbits/sec (simplex) minimum at maximum altitude

Altitude: 2,500 feet maximum (AGL) up to a maximum (AMSL) of 15,000 feet for AIS.

Environmental: Temperature: -10ºC to +65ºC

Humidity: Maximum of 98% non-condensing. Preferred due to the lower bandwidth requirements and inbuilt AES encryption.

5.2 Ground Transponder Unit (GTU)

Type: Integrated antenna, RF Transceiver, microcontroller and imaging system with solar power supply, radome mounted data acquisition transponder.

Electronics Subsystems

RF Transceiver: Zigbee, XBee

Microcontroller: ATMEGA644PV

Camera: C328R still image Jpeg camera

Camera Manufacturer: COMedia Ltd. http://www.comedia.com.hk/

Voltage: 3.6 Volts

Current: 25 mA max (Operation) 10 micro-amps (Sleep)

Dimensions

Integrated Radome Transponder: 500 mm x 100 mm (Height x Diameter)

Mounting Bracket Assembly: Water Meter Specific (to be determined)

Weight

Integrated Radome Transponder: Maximum of 2.5 Kg

Mounting Bracket Assembly: Water Meter Specific (to be determined)

5.3 Aerial Interrogator Unit (AIS)

Type: Integrated antenna, RF Transceiver, microcontroller communications management system, radome mounted transponder interrogation system and computer serial interface.

Electronics Subsystems

RF Transceiver: Zigbee, XBee

Microcontroller: ATMEGA644PV

Voltage: 3.6 Volts

Current: 25 mA max (Operation) 10 micro-amps (Sleep)

Dimensions

Integrated Radome Transponder: 500 mm x 100 mm (Height x Diameter)

Weight

Integrated Radome Transponder: Maximum of 2.5 Kg

Mounting Bracket Assembly: Aircraft Specific

Chapter 6 Conclusions

Confirmation of the specifications by which a temporary radio link for the transmission of data at a particular speed may be achieved between a ground sensor (the GTU) and an overhead interrogation system (the AIS), including that the correlation of time versus altitude and antenna gain for the AIS to acquire, upload, and close down the connection with the GTU has not been explored at this stage due to time constraints and circumstances out of my control.

Demonstration of the RF subsystem performance in terms of gain, beamwidth, return loss, beamwidth, bandwidth, and matching of the antenna into the RF transceiver device was conducted and explored and found to be within controlled limitations.

Demonstration of the required software algorithms for this process to occur in a reliable and stable manner whereby:

The AIS is able to initiate and obtain contact with the GTU based on the fact it is within the target area determined. The AIS is able to confirm that the established connection with the GTU is unique (based on the identification of the GTU), stable and secure;
The AIS is able to, via the GTU, supervise the capture of an image from the in-built camera system; The AIS is able to, via the GTU, supervise the successful upload of the image data captured from the step described above; The AIS is able to, via the GTU, upload extraneous data including battery status and temperature from the GTU was unable to be performed due unfinished coding. The AIS is able to close down the connection with the GTU in a controlled

manner at the end of the upload process so that it can be reinitialised at any subsequent time is unfinished and untested due to time constraints but has been established to be viable and attainable.

The GTU is able to carry out routine background tasks including in-built integrity analysis interspersed with ‘sleep’ processes that minimise power consumption to the utmost minimum is unfinished and untested due to time constraints but has been established to be viable and attainable.

Demonstration of an in-built power scavenging process which uses micro-solar cell (MSC) technology to float charge the inbuilt low-cost Lithium Ion batteries is unfinished and untested due to time constraints but has been established to be viable and attainable.

Demonstration of RADAC functionality through the remote capture and display of a water-meter dial display on a computer screen under the MS Windows XP system, as a result of the user initiating the capture process is unfinished and untested due to time constraints but has been established to be viable and attainable.

Further work to be Completed

Due to time constraints there is much further work to be completed on this project.

Demonstration of RADAC functionality through the remote capture and display of a water-meter dial display on a computer screen under the MS Windows XP system, as a result of the user initiating the capture process.

Data flow rates and low buffer capacity has been identified as future work to investigated before actual implementation of this project.

Manufacturing the PCB to fit inside the Radome enclosure needs to be completed.

The Graphic User Interface needs to be written for the expressed purpose of the project.

References

NUFER & Associates (Aust) Pty Ltd, Consulting Radio, Electrical & Electronics Engineers (2008)

http://www.nufer.com.au

http://www.perax.fr/en/images_en/p16xt.pdf

http://www.sutron.com/pdfs/LCRA_water_meter_project.pdf

http://www.neptunetg.com/arbmobile_waterstudies.cfm

http://www.tuc.nrao.edu/%7Edemerson/helixgain/helix.htm

http://home.att.net/%7Eray.l.cross/asap/asapexam/spread.html

http://www.comtechm2m.com/m2m-telemetry-solutions/amr-utility-telemetry.htm

http://bansky.net/blog/2008/03/jpeg-camera-and-micro-framework/comments.html

http://www.neptunetg.com/arbmobile_waterstudies.cfm

http://www.sutron.com/pdfs/LCRA_water_meter_project.pdf

http://www.perax.fr/en/images_en/p16xt.pdf

IEEE Standard 802.15.4-2006, IEEE Standard for Information technology

Zig (2006), XBee and XBee-PRO 2.4 GHz OEM RF Modules. Data sheet.

Leis, J. (2002), *Digital Signal Processing - A Matlab-Based Tutorial Approach,* Research Studies Press Ltd, England.

Kist, A. (2008), *ELE3305 Computer Systems and Communications Protocols Study Book,* University of Southern Queensland, Toowoomba.

USQ Publication (2008), *ELE4605 Fields and Waves Study Book,* University of Southern Queensland, Toowoomba.

USQ Publication (2005), *ELE2504 Electronic Design and Analysis Study Book,* University of Southern Queensland, Toowoomba.

Wade, Paul Helical Feed Antennas *W1GHZ ©2002* w1ghz@arrl.net

Technical Information for Choosing Solar Module (2006), PowerFilmInc.http://www.powerfilmsolar.com/products/oem_components/technical.htm.
The Axial Mode Helix - A Historical PerspectiveBarts_etd_CH2

Literature Review Sources

Remote Data Acquisition through Internet Based Telemetry Mohd Rizal Ahmad', Nasrutlah Khan', *C.* C. Hoong' and Naeenl Abas' 'Department of Electrical and Electronic Engineering, University Putra Malaysia, 43400 Serdang, Se1angor;Malaysia 'CECOS University, Peshawnr, Pakistan

http://www.wipo.int/pctdb/en/wo.jsp?WO=2006021963
Pub. No.: WO/2006/021963 International Application No.: PCT/IL2005/000923 Publication Date:02.03.2006 International Filing Date:26.08.2005
METER READOUT SYSTEM

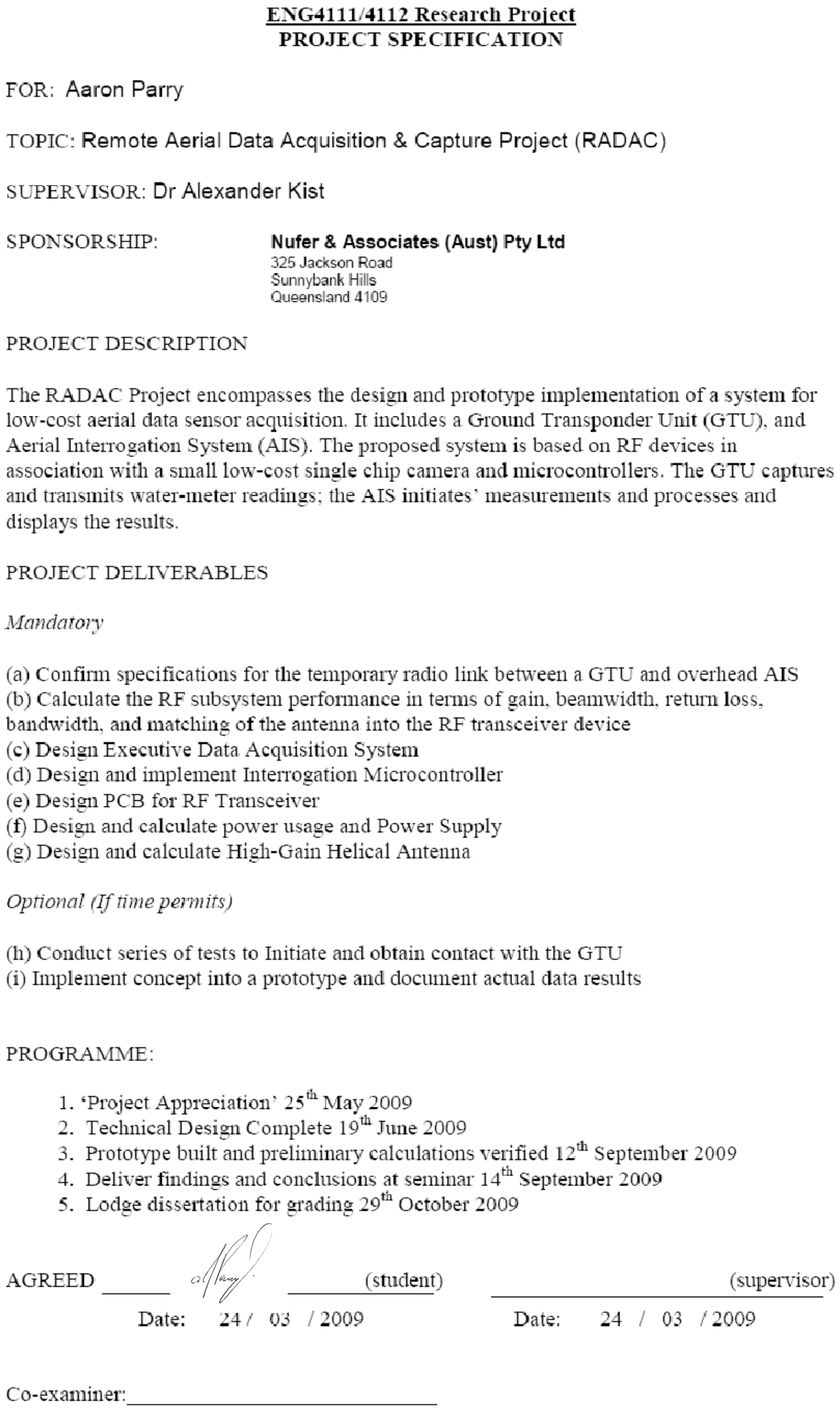
http://www.comtechm2m.com/m2m-telemetry-solutions/amr-utility-telemetry.htm
Automatic Meter Reading (AMR)
System wide monitoring using Automatic Meter Reading (AMR) for meters and sub-meters for energy management in utility applications.

Appendices

Appendices i
Appendix A i
Project Specification ii
Appendix B i
Calculations i
Electrical Current Usage of Circuit i
Solar Panel Calculations i
Beam Width and Gain Calculations ii
Antenna Calculations and Losses vi
Component Analysis xi
Appendix C i
Camera Code i
Appendix D i
Datasheets i
C328R Datasheet ..... Pg 88-91i
C328R User Manual ..... Pg 92-104i
C328-EV232 EV BOARD ..... Pg 105i
Zigbee Applications Framework Whitepaper ..... Pg 106-112i
AT45DB321D Datasheet ..... Pg 113-166i
ATMEGA644PV Datasheet ..... Pg 167-194i
Helical Feed Antennas Datasheet ..... Pg 195-217i
Survey of Helical Antennas Datasheet ..... Pg 218-240i
The Axial Mode Helix etd CH2 Datasheet ..... Pg 241-254i
XbeeProZB Datasheet ..... Pg 255-392i
Manual_xb_oem-rf-modules_802.15.4_v1.xAx ..... Pg 393-464i

Appendix A

Project Specification



ENG4111/4112 Research Project
PROJECT SPECIFICATION

FOR: Aaron Parry

TOPIC: Remote Aerial Data Acquisition & Capture Project (RADAC)

SUPERVISOR: Dr Alexander Kist

SPONSORSHIP: **Nufer & Associates (Aust) Pty Ltd**
325 Jackson Road
Sunnybank Hills
Queensland 4109

PROJECT DESCRIPTION

The RADAC Project encompasses the design and prototype implementation of a system for low-cost aerial data sensor acquisition. It includes a Ground Transponder Unit (GTU), and Aerial Interrogation System (AIS). The proposed system is based on RF devices in association with a small low-cost single chip camera and microcontrollers. The GTU captures and transmits water-meter readings; the AIS initiates' measurements and processes and displays the results.

PROJECT DELIVERABLES

Mandatory

(a) Confirm specifications for the temporary radio link between a GTU and overhead AIS
(b) Calculate the RF subsystem performance in terms of gain, beamwidth, return loss, bandwidth, and matching of the antenna into the RF transceiver device
(c) Design Executive Data Acquisition System
(d) Design and implement Interrogation Microcontroller
(e) Design PCB for RF Transceiver
(f) Design and calculate power usage and Power Supply
(g) Design and calculate High-Gain Helical Antenna

Optional (If time permits)

(h) Conduct series of tests to Initiate and obtain contact with the GTU
(i) Implement concept into a prototype and document actual data results

PROGRAMME:

1. 'Project Appreciation' 25th May 2009
2. Technical Design Complete 19th June 2009
3. Prototype built and preliminary calculations verified 12th September 2009
4. Deliver findings and conclusions at seminar 14th September 2009
5. Lodge dissertation for grading 29th October 2009

AGREED ______ ______ (student) ______ (supervisor)

Date: 24 / 03 / 2009 Date: 24 / 03 / 2009

Co-examiner: ______________________

Appendix B

Calculations

Electrical Current Usage of Circuit

The average current draw *(I_avg)* is calculated based on the current drawn values taken from the circuit.

Table 2 Electrical current requirements of hardware

Activity	Current Usage	Time/hour	Fraction of time	Current
Zigbee off	17 mA	50 minutes	0.833	14.17 mA
Zigbee on and idle	65 mA	~5 minutes	0.0833	5.42 mA
Zigbee on and receiving	75 mA	~5 minutes	0.0833	6.25 mA
Zigbee on and transmitting	230 mA	300 ms	0.000005	0.0015 mA
			Total	25.83 mA

Solar Panel Calculations

Using equations from *(Technical Information for Choosing Solar Module* 2006),

I_{app} is the duty cycle of the current draw of the application

L_{avg} is the average illumination on the solar module, and has been assumed to be 8 hours each day.

L_{avg} = 100% x equivalent hours of full sun per day

$$t = 100\% \; x \; 8/24 = 33.3\%$$

$$I \; module = Iapp \; x \; 100/ \; Lavg = 256.5 \; mA$$

Therefore a 256.5 mA solar panel would be required.

$$Imodule = Iapp \times 100$$

$$Lavg = 25.83 \times 100/33.3 = 77.49 \; mA$$

Therefore a 100 mA solar panel may be used to charge the batteries.

Beam Width and Gain Calculations

$At\ minimum\ AGL\ (2500\ Feet)\ and\ Minimum\ Frequency\ Band\ 2.4Ghz$

$(2500\ feet = 762m)$

$$Path\ loss = 32.4 + 20log0.763 + 20\ log2400$$
$$= 97.643 + 3dB = 100.643dB$$

$(3dB\ for\ edge)$

$(15000feet = 4572.014m)$

$At\ Maximum\ AMSL\ 15000feet\ and\ Maximum\ Frequency\ Band\ 2.4835\ Gigahertz$

$$Pathloss = 32.4 + 20log4.572014 + 20log2483.5$$
$$= 113.503 + 3dB = 116.503dB \qquad (3dB\ for\ edge)$$

$T_X\ output\ power\ dBm$

$P_L = Power\ level = 0 = 1mW$

$P = Power$

$$P_w = \frac{10^{\frac{PL}{10}dBm}}{100} \quad W$$

$$P_{mw} = 10^{\frac{PL}{10}dBm}$$

$RX\ Threshold\ frequency\ (PER)$ $\qquad PER = (Packet\ Error\ Rate)$

$\rho(PER) = 0.10$

$PER \approx BER$ $\qquad BER = (Baud\ Error\ Rate)$

ii

$Threshold = -88dBm = 1nw$ $\qquad = 1 * 10^{-3}(Voice)$

$= \frac{10^{-90}}{10} = 10^{-9}$ $\qquad = 1 * 10^{-6}(Data)$

$= 1 * \ 10^{-8}(Residual)$

$System\ Gain = TXPower - RXPower\ _{Threshold}$

$R_X Threshold \quad = \ -88dBn$

$T_X Power \qquad = 0dBm$

$System\ Gain = 0 — (-88) = 88dB$

$G_{TX} = 8dBd\ convert\ to\ dBi = 10.2dBi$

$G_{RX} = 8dBd\ convert\ to\ dBi = 10.2dBi$

$Range\ Gain = System\ Gain + G_{TX} + G_{RX}$

$= 88 + 10.2 + 10.2 = 108.4dB$

$Fade\ Margin\ (Excess\ Margin) FM$

$FM = Range\ Gain - Pass\ Loss\ (PL)$

$FMdB = System\ Gain + G_{TX} + G_{RX} - PL$

$= 108.4 - 100.643$

$= \ 7.757dB$(must be positive if not there is no radio link due to loss into noise)

iii

$at\ minimum\ height\ and\ frequency$

$= 108.4 - 116.503 = -8.103$

$at\ maximum\ AMSL\ and\ Maximum\ Frequency\ Band\ 2.4835$

$a\ maximum\ AMSL\ of\ 1.7986km\ or\ 6520.978feet$

$= 108.4 - 108.3999751 = 24.932udB$

$$Half\ power\ Beam\ width = \frac{52}{C_{\lambda}\sqrt{n5\lambda}} in\ degrees$$

$C\lambda = \pi D_{\lambda} = circumference\ of\ the\ winding$

$= \pi * 0.1 = 0.31416$

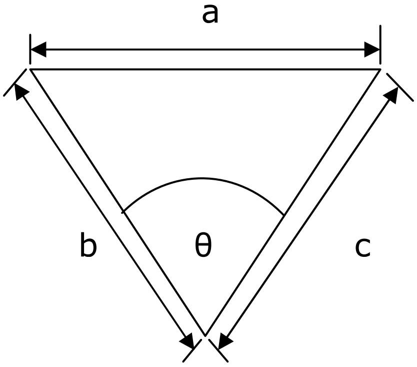
$D_{\lambda} = 100mm$

$S_{\lambda} = axial\ length\ of\ 1\ turn\ = 33.3mm = 0.03333m$

$n = how\ many\ turns\ = 5.5$

$$\theta = \frac{52}{0.31416\sqrt{5.5 * 0.03333}} \qquad = 386.5925 - 360 = 25.59degrees$$

$a^2 = b^2 + c^2 - 2bc\cos\theta$



$at\ minimum\ altitude$

$$a^2 = 762^2 + 762^2 - 2(762 * 762)cos26.59$$
$$= 0.107136 * 10^3\ 26.59$$

$$= 1.161288 * 10^6 - \; 1.038461 * 10^6$$

$$a^2 = 122.827 * 10^3$$

$$a \;= 350.466m$$

$at\ maximum\ altitude$

$$a^2 = 1798.6^2 + 1798.6^2 - 2(1798.6 * 1798.6)cos26.59$$

$$= 6.46992392 * 10^6 - 5.785615 * 10^6$$

$$a^2 = 684308.92$$

$$a \;= 827.229m$$

$$180Knots = 333.36\frac{km}{h} = 5.556\frac{km}{minute} = 0.0926\frac{km}{sec} = \frac{92.6m}{s}$$

$At\ minimum\ altitude\ Half\ power\ Beamwidth$

$$= 340.466metres$$

$Travelling\ at\ 180knots\ allows\ for\ T_X and\ R_X\ time$

$$= \frac{340.466}{92.6} = 3.6767seconds\ to\ T_X\ and\ R_X$$

$At\ maximum\ altitude\ Half\ power\ Beamwidth$

$$= 827.229m$$

$Travelling\ at\ 180knots\ allows\ for\ T_X\ and\ R_X\ time$

$$= \frac{827.229}{92.6} = 8.933seconds\ to\ T_X\ and\ R_X$$

v

Antenna Calculations and Losses

Table 3 Antenna Calculations and Losses

Frequency		Nominal Value	Accepted Value	Derived	Nominal Value	Accepted Value	Derived	Nominal Value	Accepted Value	Derived
		2450.000 MHz			2450.000 MHz			5800.000 MHz		
Wavelength				0.122 metres			0.122 metres			0.052 metres
Former Diameter (Dλ)										
	entered	50.0 mm			50.0 mm			25.0 mm		
	circumference			157.1 mm			157.1 mm			78.5 mm
Form Circumference (Cλ)										
	min	91.8 mm			91.8 mm			38.8 mm		
	max	162.9 mm			162.9 mm			68.8 mm		
		157.1 mm	157.1 mm		157.1 mm	157.1 mm		78.5 mm	78.5 mm	
Ground Plane Diameter (Gλ)										
	min	98.0 mm			98.0 mm			41.4 mm		
	max	134.7 mm			134.7 mm			56.9 mm		
	entered	134.0 mm	134.0 mm		134.0 mm	134.0 mm		50.0 mm	50.0 mm	
Axial Length of One Turn (Sλ)										
	min	33.4 mm			33.4 mm			16.7 mm		
	max	45.0 mm			45.0 mm			22.5 mm		
	entered	33.5 mm	33.5 mm		35.0 mm	35.0 mm		18.0 mm	18.0 mm	

Frequency		Nominal Value	Accepted Value	Derived	Nominal Value	Accepted Value	Derived	Nominal Value	Accepted Value	Derived
		2450.000 MHz			2450.000 MHz			5800.000 MHz		
Factor (gλ)										
	min	14.7 mm			14.7 mm			6.2 mm		
	max	15.9 mm			15.9 mm			6.7 mm		
	entered	15.0 mm	15.0 mm		15.0 mm	15.0 mm		6.7 mm	6.7 mm	
	entered	20			10			20		
Wire Diameter		4.0 mm								
Winding Diameter		56.2 mm								
Wire Length		3530.2 mm								
Length of Antenna										
				670.0 mm			350.0 mm			360.0 mm
Axial Ratio										
				1.025			1.05			1.025
Half Power Beamwidth										
				49.5 °			68.5 °			57.1 °
First Null Beamwidth										
				109.5 °			151.5 °			126.3 °
Gain										
				12.2 dBi			9.4 dBi			11.0 dBi
Transmit Power @ Module										
		10.0 dBm			18.0 dBm			18.0 dBm		

		Nominal Value	Accepted Value	Derived	Nominal Value	Accepted Value	Derived	Nominal Value	Accepted Value	Derived
Frequency		2450.000 MHz			2450.000 MHz			5800.000 MHz		
Module Cable & Connector Losses										
		0.0 dB			0.0 dB			0.0 dB		
RF Branching (Filter Loss)										
		0.0 dB			0.0 dB			0.0 dB		
RF Branching (Splitter/ Combiner Loss)										
		0.0 dB			0.0 dB			0.0 dB		
RF Branching (Isolator Losses)										
		0.0 dB			0.0 dB			0.0 dB		
Antenna Cable & Connector Losses										
		0.0 dB			0.0 dB			0.0 dB		
Alllowed EIRP										
		4.0 Watts		36.0 dBm	4.0 Watts		36.0 dBm	4.0 Watts		36.0 dBm
Trial EIRP										
				22.2 dBm			27.4 dBm			29.0 dBm
Attenuation Required										
		0.0 dB			0.0 dB			0.0 dB		
Actual EIRP										
				22.2 dBm			27.4 dBm			29.0 dBm
Path Length										
		1.000 km			0.500 km			0.500 km		

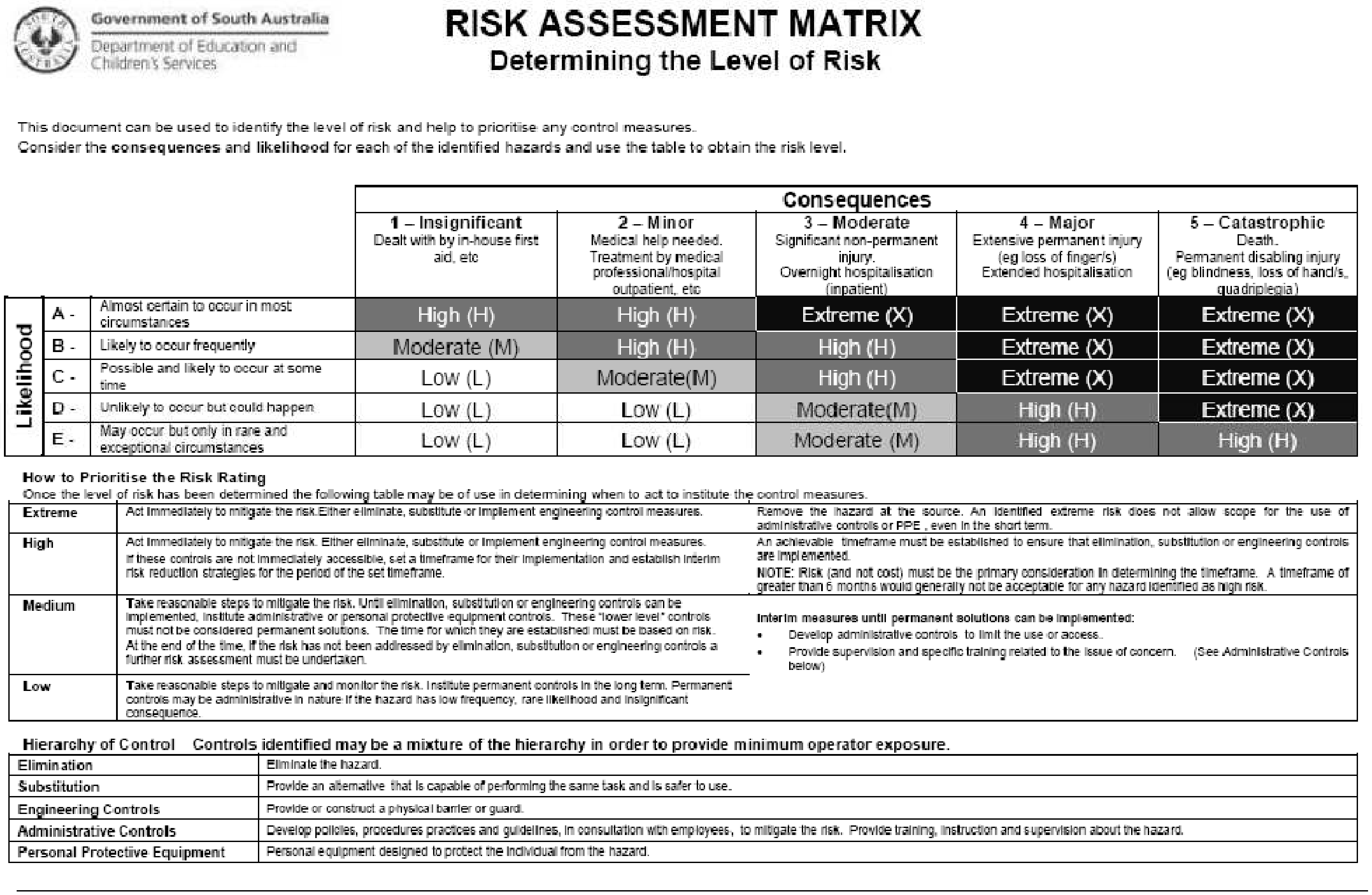
Frequency		Nominal Value	Accepted Value	Derived	Nominal Value	Accepted Value	Derived	Nominal Value	Accepted Value	Derived
		2450.000 MHz			2450.000 MHz			5800.000 MHz		
Mid Range Fresnel Clearance										
				3.3 metres			2.3 metres			1.5 metres
Path Loss										
				100.2 dB			94.2 dB			101.6 dB
Receiver Signal @ Antenna Connector										
				-65.7 dBm			-57.4 dBm			-61.7 dBm
Antenna Cable & Connector Losses										
		0.0 dB			0.0 dB			0.0 dB		
RF Branching (Filter Loss)										
		0.0 dB			0.0 dB			0.0 dB		
RF Branching (Splitter/ Combiner Loss)										
		0.0 dB			0.0 dB			0.0 dB		
Module Cable & Connector Losses										
		0.0 dB			0.0 dB			0.0 dB		
Receive Power @ Module										
				-65.7 dBm			-57.4 dBm			-61.7 dBm

Frequency		Nominal Value	Accepted Value	Derived	Nominal Value	Accepted Value	Derived	Nominal Value	Accepted Value	Derived
		2450.000 MHz			2450.000 MHz			5800.000 MHz		
Receiver Threshold (1% PER)										
				-70.0 dBm			-95.0 dBm			-95.0 dBm
Receiver Fade Margin										
				4.3 dB			37.6 dB			33.3 dB

Component Analysis

Table 4 Component Analysis

						Transceiver Option					
	Technology			**Power**		**Band**	**Technology**	**Mode**	**Power**		
VIDAUD	Analogue Video		00	0 dBm		0250	VIDAUD	TXO	10	0250-VIDAUD-TXO-10	
80211A	Wireless LAN		10	10 dBm		0250	VIDAUD	RXO	NA	0250-VIDAUD-RXO-NA	
80211B	Wireless LAN		18	18 dBm		0250	802154	HDX	20	0250-802154-HDX-20	
80211G	Wireless LAN		20	20 dBm		0250	ZIGBCO	HDX	20	0250-ZIGBCO-HDX-20	
ZIGBCO	Zigbee Coordinator		NA	Not Applic		0250	ZIGBRO	HDX	20	0250-ZIGBRO-HDX-20	
ZIGBRO	Zigbee Router					0250	ZIGBND	HDX	20	0250-ZIGBND-HDX-20	
ZIGBND	Zigbee End Device					0580	80211A	HDX	18	0580-80211A-HDX-18	
802154	802.15.4					0250	80211B	HDX	18	0250-80211B-HDX-18	
6LWPAN	6LowPan					0250	80211G	HDX	18	0250-80211G-HDX-18	
NORDSM	Nordic Semi										
						RF Branching Option					
	Direction					**Split**	**Transceivers**	**Mode**			
TXO	Transmit Only					1W	1	HDX		1W-1-HDX	Integrated & Split Configurations
RXO	Receive Only					1W	1	TXO		1W-1-TXO	Integrated & Split Configurations
FDX	Full Duplex					1W	1	RXO		1W-1-RXO	Integrated & Split Configurations
HDX	Half Duplex					2W	1	HDX		2W-1-HDX	Only Available With HDX
						3W	1	HDX		3W-1-HDX	Only Available With HDX
	Frequency Band					4W	1	HDX		4W-1-HDX	Only Available With HDX
0250	2.5 GHz										
0580	5.8 GHz					**Antenna Option**					
1050	10.5 GHz					**Band**	**Gain**	**Antenna**	**Construct**		
						0250	122I	HELX	INT	0250-122I-HELX-INT	
	Gain					0250	095I	HELX	INT	0250-095I-HELX-INT	
022I	2.2 dBi					0250	122I	HELX	EXT	0250-122I-HELX-EXT	
122I	12.2 dBi					0250	095I	HELX	EXT	0250-095I-HELX-EXT	
095I	9.5 dBi					0250	022I	CHIP	INT	0250-022I-CHIP-INT	
						0250	022I	MONO	EXT	0250-022I-MONO-EXT	
						Transceiver Option					
	Technology			**Power**		**Band**	**Technology**	**Mode**	**Power**		
VIDAUD	Analogue Video		00	0 dBm		0250	VIDAUD	TXO	11	0250-VIDAUD-TXO-11	
80211A	Wireless LAN		10	10 dBm		0250	VIDAUD	RXO	NA	0250-VIDAUD-RXO-NA	
80211B	Wireless LAN		18	18 dBm		0250	802154	HDX	20	0250-802154-HDX-20	
80211G	Wireless LAN		20	20 dBm		0250	ZIGBCO	HDX	20	0250-ZIGBCO-HDX-20	
ZIGBCO	Zigbee Coordinator		NA	Not Applic		0250	ZIGBRO	HDX	20	0250-ZIGBRO-HDX-20	
ZIGBRO	Zigbee Router					0250	ZIGBND	HDX	20	0250-ZIGBND-HDX-20	
ZIGBND	Zigbee End Device					0580	80211A	HDX	18	0580-80211A-HDX-18	
802155	802.15.5					0250	80211B	HDX	18	0250-80211B-HDX-18	
6LWPAN	6LowPan					0250	80211G	HDX	18	0250-80211G-HDX-18	
NORDSM	Nordic Semi										
						RF Branching Option					
	Direction					**Split**	**Transceivers**	**Mode**			
TXO	Transmit Only					1W	1	HDX		1W-1-HDX	Integrated & Split Configurations
RXO	Receive Only					1W	1	TXO		1W-1-TXO	Integrated & Split Configurations
FDX	Full Duplex					1W	1	RXO		1W-1-RXO	Integrated & Split Configurations
HDX	Half Duplex					2W	1	HDX		2W-1-HDX	Only Available With HDX



Government of South Australia
Department of Education and Children's Services

RISK ASSESSMENT MATRIX

Determining the Level of Risk

This document can be used to identify the level of risk and help to prioritise any control measures.
Consider the **consequences** and **likelihood** for each of the identified hazards and use the table to obtain the risk level.

			Consequences				
			1 – Insignificant Dealt with by in-house first aid, etc	**2 – Minor** Medical help needed. Treatment by medical professional/hospital outpatient, etc	**3 – Moderate** Significant non-permanent injury. Overnight hospitalisation (inpatient)	**4 – Major** Extensive permanent injury (eg loss of finger/s) Extended hospitalisation	**5 – Catastrophic** Death. Permanent disabling injury (eg blindness, loss of hand/s, quadriplegia)
Likelihood	A -	Almost certain to occur in most circumstances	High (H)	High (H)	Extreme (X)	Extreme (X)	Extreme (X)
	B -	Likely to occur frequently	Moderate (M)	High (H)	High (H)	Extreme (X)	Extreme (X)
	C -	Possible and likely to occur at some time	Low (L)	Moderate(M)	High (H)	Extreme (X)	Extreme (X)
	D -	Unlikely to occur but could happen	Low (L)	Low (L)	Moderate(M)	High (H)	Extreme (X)
	E -	May occur but only in rare and exceptional circumstances	Low (L)	Low (L)	Moderate (M)	High (H)	High (H)

How to Prioritise the Risk Rating
Once the level of risk has been determined the following table may be of use in determining when to act to institute the control measures.

Extreme	Act immediately to mitigate the risk.Either eliminate, substitute or implement engineering control measures.	Remove the hazard at the source. An identified extreme risk does not allow scope for the use of administrative controls or PPE , even in the short term.
High	Act immediately to mitigate the risk. Either eliminate, substitute or implement engineering control measures. If these controls are not immediately accessible, set a timeframe for their implementation and establish interim risk reduction strategies for the period of the set timeframe.	An achievable timeframe must be established to ensure that elimination, substitution or engineering controls are implemented. NOTE: Risk (and not cost) must be the primary consideration in determining the timeframe. A timeframe of greater than 6 months would generally not be acceptable for any hazard identified as high risk.
Medium	Take reasonable steps to mitigate the risk. Until elimination, substitution or engineering controls can be implemented, institute administrative or personal protective equipment controls. These "lower level" controls must not be considered permanent solutions. The time for which they are established must be based on risk. At the end of the time, if the risk has not been addressed by elimination, substitution or engineering controls a further risk assessment must be undertaken.	**Interim measures until permanent solutions can be implemented:** • Develop administrative controls to limit the use or access. • Provide supervision and specific training related to the issue of concern. (See Administrative Controls below)
Low	Take reasonable steps to mitigate and monitor the risk. Institute permanent controls in the long term. Permanent controls may be administrative in nature if the hazard has low frequency, rare likelihood and insignificant consequence.	

Hierarchy of Control Controls identified may be a mixture of the hierarchy in order to provide minimum operator exposure.

Elimination	Eliminate the hazard.
Substitution	Provide an alternative that is capable of performing the same task and is safer to use.
Engineering Controls	Provide or construct a physical barrier or guard.
Administrative Controls	Develop policies, procedures practices and guidelines, in consultation with employees, to mitigate the risk. Provide training, instruction and supervision about the hazard.
Personal Protective Equipment	Personal equipment designed to protect the individual from the hazard.

Figure 27 Risk Assessment

Government of South Australia
Department of Education and Children's Services

RISK ASSESSMENT SUMMARY

Topic: Date: Issue No. Review date:

Identify Hazards and subsequent Risks	Analyse Risks Evaluate Risks			Identify and evaluate existing risk controls			Further Risk Treatments
Hazards/Issues/Risks	Consequence	Likelihood	Risk level	What we are doing now to manage this risk.	Effectiveness of our strategies	New risk level	Further action needed Opportunities for improvement
Manual handling	3	C	H	Administrative Controls, Engineering Controls	Moderate	M	
EMF Radiation	3	C	H	Administrative Controls, Engineering Controls	Moderate	M	
Cuts and Abrasions	3	C	H	Administrative Controls, Engineering Controls	Moderate	M	

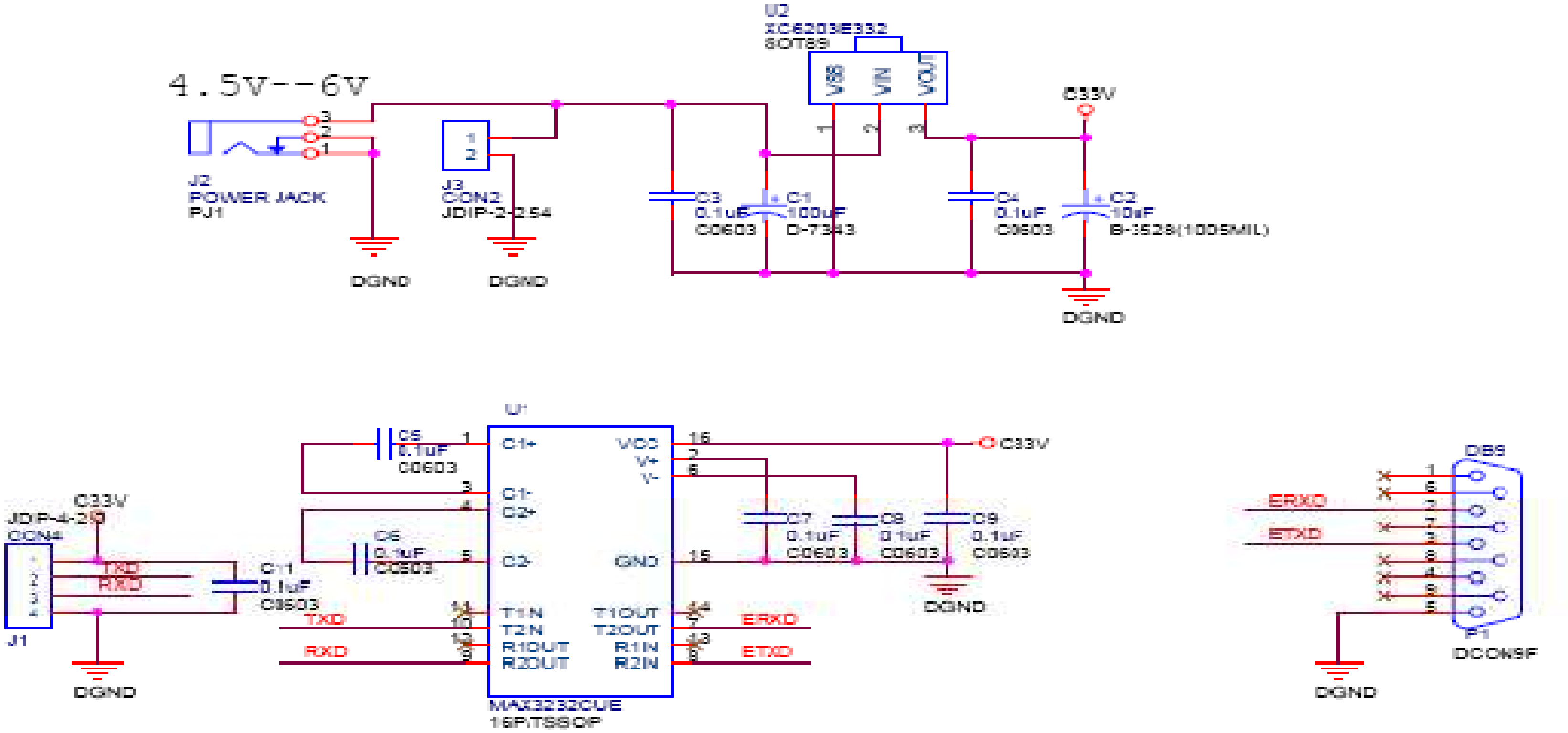


Figure 28 Camera Module Interface Schematic

Appendix C

Camera Code

Written in C#

```
using System;
using System.IO.Ports;
using System.Threading;
using System.Collections.Generic;



namespace CR328R
{
   /// <summary>
   /// Class from http://bansky.net/blog/2008/03/jpeg-camera-and-micro-framework/comments.html
   ///
   /// Modified by Rick Chronister to work with XBee directly and a desktop application
   /// Modified by Aaron Parry to work with XBee directly and a desktop application
   /// http://www.codeproject.com/KB/recipes/C328R.aspx
   /// </summary>
   public class C328R
   {
      public enum ColorType { GreyScale2 = 1, GreyScale4 = 2, GreyScale8 = 3, Color12 = 5, Color16 = 6, Jpeg = 7 };
      public enum PreviewResolution { R80x60 = 1, R160x120 = 3 };
      public enum JpegResolution { R80x64 = 1, R160x128 = 3, R320x240 = 5, R640x480 = 7 };
      public enum PictureType { Snapshot = 1, Preview = 2, Jpeg = 5 };
      public enum SnapshoteType { Compressed = 0, Uncompressed = 1 };
      public enum FrequencyType { F50Hz = 0, F60Hz = 1 };
      public enum BaudRate { Baud7200, Baud9600, Baud14400, Baud19200, Baud28800, Baud38400, Baud57600, Baud115200 };

      #region Command constants
      const byte CMD_PREFIX = 0xAA;
      const byte CMD_INITIAL = 0x01;
      const byte CMD_GETPICTURE = 0x04;
      const byte CMD_SNAPSHOT = 0x05;
      const byte CMD_PACKAGESIZE = 0x06;

      const byte CMD_BAUDRATE = 0x07;
      const byte CMD_RESET = 0x08;
      const byte CMD_POWEROFF = 0x09;

      const byte CMD_DATA = 0x0A;
      const byte CMD_SYNC = 0x0D;
      const byte CMD_ACK = 0x0E;

      const byte CMD_NAK = 0x0F;
      const byte CMD_LIGHTFREQ = 0x13;
      #endregion
```

```
    /// <summary>
    /// Size of the data package with image
    /// </summary>
    const int PACKAGE_SIZE = 128;//Reduced to buffer for xBee

    private SerialPort serialPort;
    private byte[] command = new byte[6];
    public List<string> comLog = new List<string>();
    public bool Debug;



    /// <summary>
    /// Create new C328R camera instance
    /// </summary>
    /// <param name="serialConfig">Configuration for serial port</param>
    public C328R(SerialPort SerialPort)
    {
      serialPort = SerialPort;
    }

    /// <summary>
    /// Tries to sync with camera
    /// </summary>
    /// <returns>True if succeeded</returns>
    public bool Sync()
    {
      // Create 'Sync' command
      command[0] = CMD_PREFIX;
      command[1] = CMD_SYNC;
      command[2] = 0;
      command[3] = 0;
      command[4] = 0;
      command[5] = 0;

      byte[] recvCommand = new byte[6];
      int i = 0;
      bool stat = false;

      while (true)
      {
        i++;
        if (i > 60)
        {
          stat = false;
          break;
        }

        //AddLog("Sending Sync Command " + i.ToString());
        stat = SendCommand(command);

        // Wait for ACK
        //AddLog("Waiting For ACK");
        stat = WaitForResponse(ref recvCommand, 200, false); //Reduced to
100ms
        if (!stat || recvCommand[1] != CMD_ACK || recvCommand[2] !=
CMD_SYNC)
        {
          continue;
```

```
            }

            // Wait for SYNC
            //AddLog("Waiting for SYNC");
            stat = WaitForResponse(ref recvCommand, 100, false);
            if (!stat || recvCommand[1] != CMD_SYNC)
            {
                continue;
            }

            //AddLog("Sending Sync ACK");
            stat = SendACK();

            break;
        }
        return stat;
    }

    /// <summary>
    /// Inititate camera with specific settings
    /// </summary>
    /// <param name="colorType">Color depth</param>
    /// <param name="previewResolution">Preview resolution</param>
    /// <param name="jpegResolution">Jpeg image resolution</param>
    /// <returns>True if succeeded</returns>
    public bool Initial(ColorType colorType, PreviewResolution previewResolution,
JpegResolution jpegResolution)
    {
        // Create the command
        command[0] = CMD_PREFIX;
        command[1] = CMD_INITIAL;
        command[2] = 0;
        command[3] = (byte)colorType;
        command[4] = (byte)previewResolution;
        command[5] = (byte)jpegResolution;

        // send 'Initial' command
        SendCommand(command);

        // Wait for ACK
        if (!ReceiveACK(CMD_INITIAL, 250))
            return false;

        // Create package size command
        int packsize = PACKAGE_SIZE;
        command[0] = CMD_PREFIX;
        command[1] = CMD_PACKAGESIZE;
        command[2] = 0x08;
        command[3] = (byte)packsize; // PACKAGE_SIZE Low byte
        command[4] = (byte)(packsize >> 8); // PACKAGE_SIZE High byte
        command[5] = 0x00;

        // Send 'Set Package Size' command
        SendCommand(command);

        // Wait for ACK
        if (!ReceiveACK(CMD_PACKAGESIZE, 250))
            return false;
        return true;
    }

    /// <summary>
    /// Make snapshot and keep it the buffer
    /// </summary>
    /// <param name="snapshotType">Snapshot type</param>
    /// <param name="skipFrameCounter">Number of dropped frame before
compression. 0 means current frame.</param>
    /// <returns>True if succeeded</returns>
    public bool Snapshot(SnapshoteType snapshotType, int skipFrameCounter)
    {
        // Create snapshot command
        command[0] = CMD_PREFIX;
        command[1] = CMD_SNAPSHOT;
        command[2] = (byte)snapshotType;
        command[3] = (byte)(skipFrameCounter);
        command[4] = (byte)(skipFrameCounter >> 8);
        command[5] = 0x00;

        // send 'Snapshot' command
        SendCommand(command);

        // Receive ACK
        if (!ReceiveACK(CMD_SNAPSHOT, 100))
            return false;

        return true;
    }

    /// <summary>
    /// Get Jpeg picture from camera
    /// </summary>
    /// <param name="pictureType">Picture type (Snapshot/Jpeg)</param>
    /// <param name="dataBuffer">Buffer for received data</param>
    /// <param name="processDelay">Time to process the image in camera.
Larger Jpegs takes about 1 sec.</param>
    /// <returns>True if succeeded</returns>
    public bool GetJpegPicture(PictureType pictureType, out byte[] dataBuffer, int
processDelay)
    {
        dataBuffer = new byte[0];
        int pictureDataSize;

        // Send 'Get Picture' command
        if (!GetPictureCommand(pictureType, processDelay, out pictureDataSize))
            return false;


        // init data buffer
        dataBuffer = new byte[pictureDataSize];
        int bufferPosition = 0;
        int packageCounter = 0;
        int errorCounter = 0;
        int packageID = 0;
        byte[] response = new byte[PACKAGE_SIZE];

        // Loop to read all data
        while (bufferPosition < dataBuffer.Length && errorCounter <= 15)
```

```
            {
                SendACK(packageCounter);
                Thread.Sleep(40);

                // Wait for data package
                Array.Clear(response, 0, PACKAGE_SIZE);

                bool stat = WaitForResponse(ref response, 250, true);

                // If data package received process it else increase error counter
                if (stat)
                {
                    // Get data size in packet
                    int packetSize = response[3] << 8;
                    packetSize |= response[2];

                    //Get package id
                    packageID = (int)response[0] + ((int)response[1] * 16);

                    //Validate package id we expected is the same as we recieved
                    if (packageID != packageCounter)
                    {
                        return false;
                        throw new Exception("Expecting package " +
packageCounter.ToString() + " but recieved " + packageID.ToString());
                    }


                    Array.Copy(response, 4, dataBuffer, bufferPosition, packetSize);

                    // Move buffer position and get ready for next package
                    bufferPosition += packetSize;
                    packageCounter++;
                }
                else
                    errorCounter++;
            }

            // Send final package ACK
            SendACK(packageCounter);

            if (errorCounter == 16)
            {
                return false;
            }

            return true;
        }

        /// <summary>
        /// Get raw picture from camera
        /// </summary>
        /// <param name="pictureType">Picture type (Snapshot/Preview)</param>
        /// <param name="dataBuffer">Buffer for received data</param>
        /// <param name="processDelay">Time to process the image in camera.
Larger Jpegs takes about 1 sec.</param>
        /// <returns>True if succeeded</returns>
        public bool GetRawPicture(PictureType pictureType, out byte[] dataBuffer, int
processDelay)
```

v

```
        {
            dataBuffer = new byte[0];
            int dataSize;

            // Send 'Get Picture' command
            if (!GetPictureCommand(pictureType, processDelay, out dataSize))
                return false;

            // init data buffer
            dataBuffer = new byte[dataSize];

            // Read whole image at once
            if (!WaitForResponse(ref dataBuffer, 2000, false))
                return false;

            // Send final package ACK
            SendACK(0x00);

            return true;
        }

        /// <summary>
        /// Set the light frequency of camera
        /// </summary>
        /// <param name="lightFrequency">Light frequency (50Hz / 60
Hz)</param>
        /// <returns>True if succeeded</returns>
        public bool LigtFrequency(FrequencyType lightFrequency)
        {
            // Create 'Light Frequency' command
            command[0] = CMD_PREFIX;
            command[1] = CMD_LIGHTFREQ;
            command[2] = (byte)lightFrequency;
            command[3] = 0x00;
            command[4] = 0x00;
            command[5] = 0x00;

            // Send 'Light Frequency' command
            SendCommand(command);

            // Receive ACK
            if (!ReceiveACK(CMD_LIGHTFREQ, 100))
                return false;

            return true;
        }

        /// <summary>
        /// Set communicatin speed that will be used by camera until physically power
off.
        /// </summary>
        /// <param name="baudRate">Baudrate</param>
        /// <returns>True if succeeded</returns>
        public bool SetBaudRate(BaudRate baudRate)
        {
            byte divider1;

            switch (baudRate)
            {
                case BaudRate.Baud7200:
                    divider1 = 0xFF;
                    break;
                case BaudRate.Baud9600:
                    divider1 = 0xBF;
                    break;
                case BaudRate.Baud14400:
                    divider1 = 0x7F;
                    break;
                case BaudRate.Baud19200:
                    divider1 = 0x5F;
                    break;
                case BaudRate.Baud28800:
                    divider1 = 0x3F;
                    break;
                case BaudRate.Baud38400:
                    divider1 = 0x2F;
                    break;
                case BaudRate.Baud57600:
                    divider1 = 0x1F;
                    break;
                case BaudRate.Baud115200:
                    divider1 = 0x0F;
                    break;
                default:
                    divider1 = 0xBF;
                    break;
            }

            // Create 'Set Baudrate' command
            command[0] = CMD_PREFIX;
            command[1] = CMD_BAUDRATE;
            command[2] = divider1; // Divider 1
            command[3] = 0x01; // Divider 2
            command[4] = 0x00;
            command[5] = 0x00;

            // Send 'Set Baudrate' command
            SendCommand(command);

            // Receive ACK
            if (!ReceiveACK(CMD_BAUDRATE, 100))
                return false;

            return true;
        }

        /// <summary>
        /// Reset the camera
        /// </summary>
        /// <param name="completeReset">True for complete reset, False for state
machine reset</param>
        /// <returns>True if succeeded<</returns>
        public bool Reset(bool completeReset)
        {
            // Create 'Reset' command
            command[0] = CMD_PREFIX;
            command[1] = CMD_RESET;
            command[2] = (byte)((completeReset == true) ? 0x00 : 0x01);
            command[3] = 0x00;
            command[4] = 0x00;
            command[5] = 0xFF;

            // Send 'Reset' command
            SendCommand(command);

            // Receive ACK
            if (!ReceiveACK(CMD_RESET, 250))
                return false;

            return true;
        }

        /// <summary>
        /// Power off the camera
        /// </summary>
        /// <returns>True if succeeded</returns>
        public bool PowerOff()
        {
            // Create 'Power Off' command
            command[0] = CMD_PREFIX;
            command[1] = CMD_POWEROFF;
            command[2] = 0x00;
            command[3] = 0x00;
            command[4] = 0x00;
            command[5] = 0x00;

            // Send 'Power Off' command
            SendCommand(command);

            // Receive ACK
            if (!ReceiveACK(CMD_POWEROFF, 100))
                return false;

            return true;
        }

        /// <summary>
        /// Waits for response from camera
        /// </summary>
        /// <param name="readBuffer">Buffer for response</param>
        /// <param name="timeout">Timeout in milliseconds</param>
        /// <returns>False if timeout occured</returns>
        private bool WaitForResponse(ref byte[] readBuffer, int timeout, bool
pictureData)
        {
            int bytesRead = 0;
            int ret;
            int packageSize = 0;

            serialPort.ReadTimeout = timeout;

            try
            {
                for (bytesRead = 0; bytesRead < readBuffer.Length; bytesRead++)
                {
                    ret = serialPort.ReadByte();
                    readBuffer[bytesRead] = (byte)ret;
                    if (readBuffer.Length > 6 && bytesRead == 3 )
                    {
                        //This is a picture data packet that contains the size
                        packageSize = readBuffer[3] << 8;
                        packageSize |= readBuffer[2];
                        packageSize += 6;
                    }

                    if (packageSize > 0 && (bytesRead == packageSize - 1))
                    {
                        //This is a picture data packet and we got all the bytes
                        return true;
                    }
                }
            }
            catch
            {
                AddLog("Response Timeout of " + timeout.ToString() + "s");
                return false;
            }

            return true;


        }

        /// <summary>
        /// Waits for response and parse it for ACK
        /// </summary>
        /// <param name="expectedACKCommand">Command to be
ACKnowlegde</param>
        /// <param name="timeout">Timeout in miliseconds</param>
        /// <exception>Throws exception with error code when NACK is received.
        /// See C328R user manula for more information about error codes.
        /// Error codes:
        /// Picture Type Error      01h   Parameter Error                  0Bh
        /// Picture Up Scale        02h   Send Register Timeout            0Ch
        /// Picture Scale Error     03h   Command ID Error                 0Dh
        /// Unexpected Reply        04h   Picture Not Ready                0Fh
        /// Send Picture Timeout    05h   Transfer Package Number Error    10h
        /// Unexpected Command      06h   Set Transfer Package Size Wrong  11h
        /// SRAM JPEG Type Error    07h   Command Header Error             F0h
        /// SRAM JPEG Size Error    08h   Command Length Error             F1h
        /// Picture Format Error    09h   Send Picture Error               F5h
        /// Picture Size Error      0Ah   Send Command Error               FFh
        /// </exception>
        /// <returns>True if ACK for expected command was received</returns>
        private bool ReceiveACK(byte expectedACKCommand, int timeout)
        {
            byte[] responseBuffer = new byte[6];
            bool stat = WaitForResponse(ref responseBuffer, timeout, false);

            // If NAK is received instead of ACK - raise exception
            if (stat && responseBuffer[1] == CMD_NAK)
            {
                throw new Exception("C328R Error " + responseBuffer[4]);
            }
            // If no ACK or ACK for different command received - return false
            if (!stat || responseBuffer[1] != CMD_ACK || responseBuffer[2] !=
expectedACKCommand)
            {
                return false;
            }

            return true;
        }

        /// <summary>
        /// Send ACK command
        /// </summary>
        /// <param name="packageId">Package Id for ACK</param>
        /// <returns></returns>
        private bool SendACK(int packageId)
        {
            byte[] ackCommand = new byte[6] { CMD_PREFIX, CMD_ACK, 0, 0,
(byte)packageId, (byte)(packageId >> 8) };
            return SendCommand(ackCommand);
        }

        /// <summary>
        /// Send ACK command
        /// </summary>
        /// <returns>True if succeeded</returns>
        private bool SendACK()
        {
            return SendACK(0x00);
        }

        /// <summary>
        /// Send generic command
        /// </summary>
        /// <param name="commandArray">Byte array with command and
arguments</param>
        /// <returns>True if succeeded</returns>
        private bool SendCommand(byte[] commandArray)
        {
            int len = commandArray.Length;

            //int send = serialPort.Write(commandArray, 0, len);
            try
            {
                serialPort.Write(commandArray, 0, len);
            }
            catch (Exception ex)
            {
                return false;
            }


            Thread.Sleep(10);
            return true;
            //return send == len;
        }

        /// <summary>
```

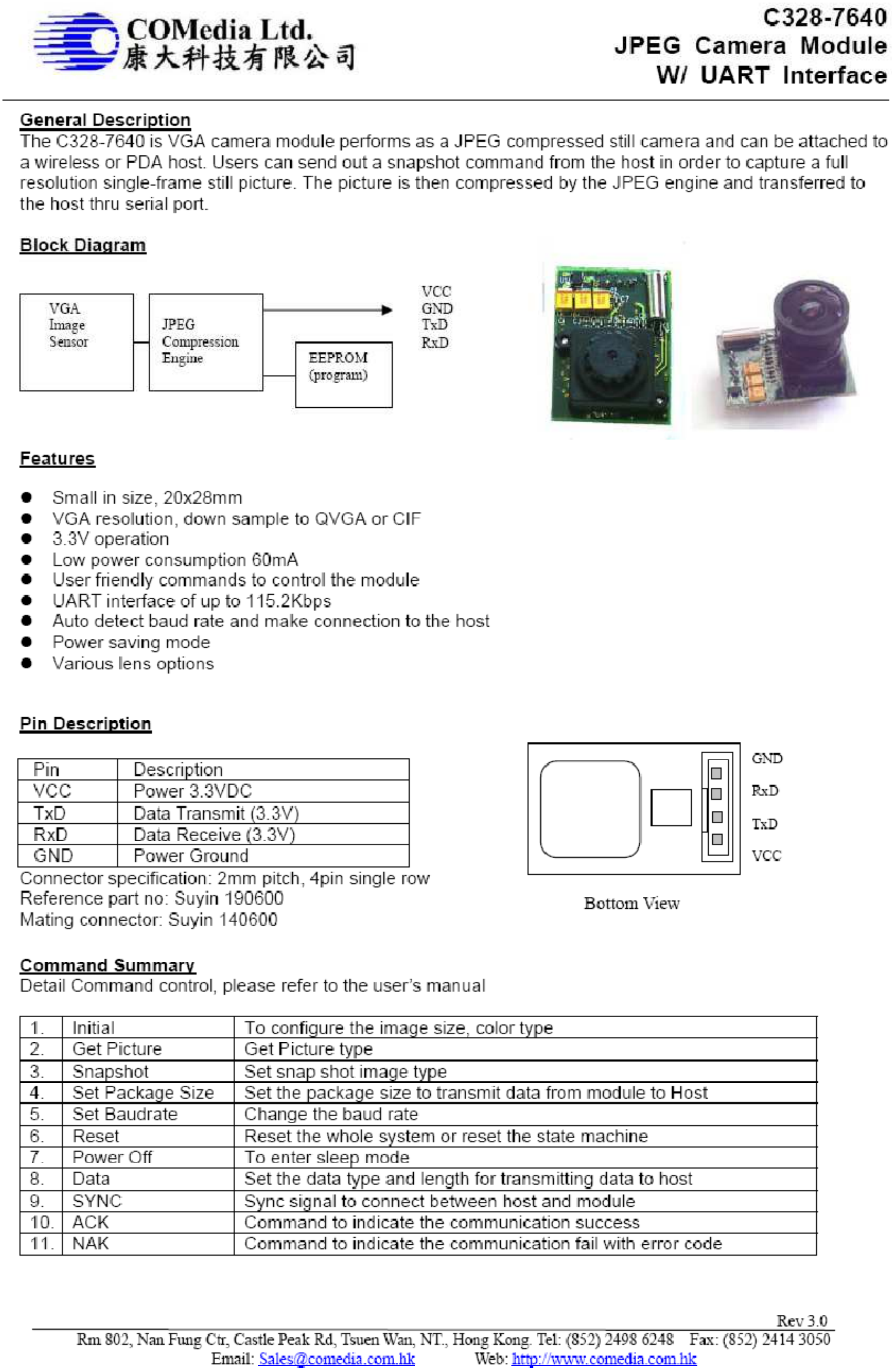
x

```
    /// Helper method: Send GetPicture command and receive expected data size
of the picture
    /// </summary>
    /// <param name="pictureType">Picture type
(Snapshot/Preview/Jpeg)</param>
    /// <param name="processDelay">Time to process the image in
camera</param>
    /// <param name="pictureDataSize">Picture data size</param>
    /// <returns>True if succeeded</returns>
    private bool GetPictureCommand(PictureType pictureType, int processDelay,
out int pictureDataSize)
    {
      pictureDataSize = 0;

      // Create 'Get Picture' command
      command[0] = CMD_PREFIX;
      command[1] = CMD_GETPICTURE;
      command[2] = (byte)pictureType;
      command[3] = 0x00;
      command[4] = 0x00;
      command[5] = 0x00;

      // Send 'Get Picture' command
      SendCommand(command);

      // Give camera time to proceed the image
      Thread.Sleep(processDelay);

      // Receive ACK
      if (!ReceiveACK(CMD_GETPICTURE, 1000))
        return false;

      // Receive DATA command, with inormations about image
      byte[] response = new byte[6];
      bool stat = WaitForResponse(ref response, 500, false);
      if (!stat || response[1] != CMD_DATA)
        return false;

      // Get dataSize from three bytes
      pictureDataSize = response[5] << 8;
      pictureDataSize |= response[4] << 8;
      pictureDataSize |= response[3];

      return true;
    }
    /// <summary>
    /// Used to determine what is going on with serial communication
    /// </summary>
    /// <param name="Message"></param>
    private void AddLog(string Message)
    {
      if (Debug)
      {
        comLog.Add(DateTime.Now.ToShortDateString() + " " +
DateTime.Now.ToLongTimeString() +  " : " + Message);
      }
    }
  }
}
```

Appendix D

Datasheets

C328R Datasheet	Pg 88-91
C328R User Manual	Pg 92-104
C328-EV232 EV BOARD	Pg 105
Zigbee Applications Framework Whitepaper	Pg 106-112
AT45DB321D Datasheet	Pg 113-166
ATMEGA644PV Datasheet	Pg 167-194
Helical_Feed_Antennas Datasheet	Pg 195-217
Survey of Helical Antennas Datasheet	Pg 218-239
The Axial Mode Helix _etd_CH2 Datasheet	Pg 240-254
XbeeProZB Datasheet	Pg 255-392
Manual_xb_oem-rf-modules_802.15.4_v1.xAx	Pg 393-464



COMedia Ltd.
康大科技有限公司

C328-7640
JPEG Camera Module
W/ UART Interface

General Description

The C328-7640 is VGA camera module performs as a JPEG compressed still camera and can be attached to a wireless or PDA host. Users can send out a snapshot command from the host in order to capture a full resolution single-frame still picture. The picture is then compressed by the JPEG engine and transferred to the host thru serial port.

Block Diagram

Features

- Small in size, 20x28mm
- VGA resolution, down sample to QVGA or CIF
- 3.3V operation
- Low power consumption 60mA
- User friendly commands to control the module
- UART interface of up to 115.2Kbps
- Auto detect baud rate and make connection to the host
- Power saving mode
- Various lens options

Pin Description

Pin	Description
VCC	Power 3.3VDC
TxD	Data Transmit (3.3V)
RxD	Data Receive (3.3V)
GND	Power Ground

Connector specification: 2mm pitch, 4pin single row
Reference part no: Suyin 190600
Mating connector: Suyin 140600

Bottom View

Command Summary

Detail Command control, please refer to the user's manual

1.	Initial	To configure the image size, color type
2.	Get Picture	Get Picture type
3.	Snapshot	Set snap shot image type
4.	Set Package Size	Set the package size to transmit data from module to Host
5.	Set Baudrate	Change the baud rate
6.	Reset	Reset the whole system or reset the state machine
7.	Power Off	To enter sleep mode
8.	Data	Set the data type and length for transmitting data to host
9.	SYNC	Sync signal to connect between host and module
10.	ACK	Command to indicate the communication success
11.	NAK	Command to indicate the communication fail with error code

Rev 3.0

Electrical Specification

V_{DD} = 3.3V+10%, TA = 0 to 25°C

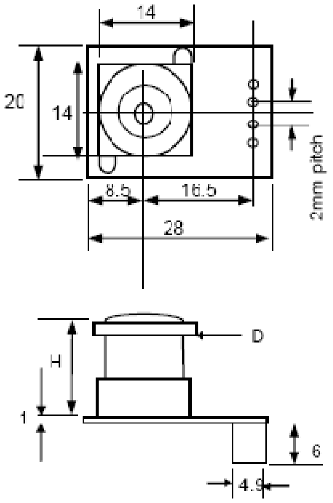
Symbol	Parameter	Condition	Min	Typ	Max	Unit
V_{DD}	DC supply voltage		3.0	3.3	3.6	V
Io	Normal Operation Current	Operating		60		mA
Is	Suspend Current	Suspend		100		uA
V_{IH}	High level input voltage	TTL	2.0			V
V_{IL}	Low level input voltage	TTL			0.8	V

Lens Specification

Description	C328-7640	C328-2225BW	C328-2520BW	C328-3620IR	C328-3620BW	C328-6016BW	C328-6016IR
F/#	2.8	2.5	2.0	2.0	2.0	1.6	1.6
Focal length (mm)	4.63	2.2	2.5	3.6	3.6	6.0	6.0
Field of View Diagonal (deg)	57	118	100	66	66	36	36
Filter Option IR-cut filter	Yes	NA	NA	Yes	NA	NA	Yes
Total height from PCB H (mm)	10	18	24	22	22	20	20
Diameter of lens cap D (mm)	9	18	15	14	14	14	14

Board Measurement

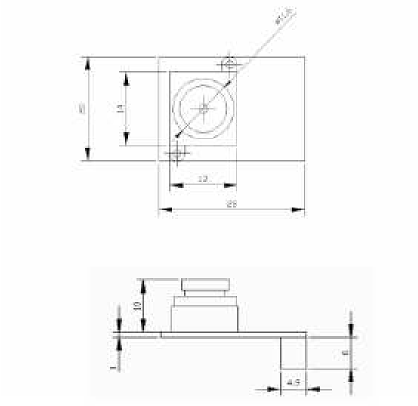
Note: All lens holder are with 14x14, thread of 12mmx0.5, height H will be varied from different lens spec.

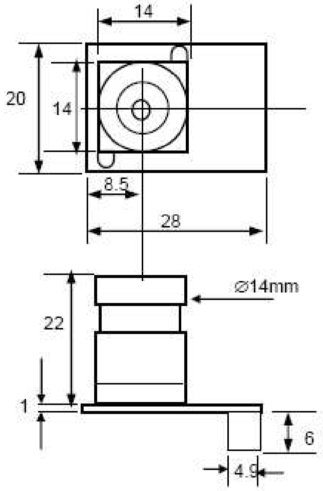


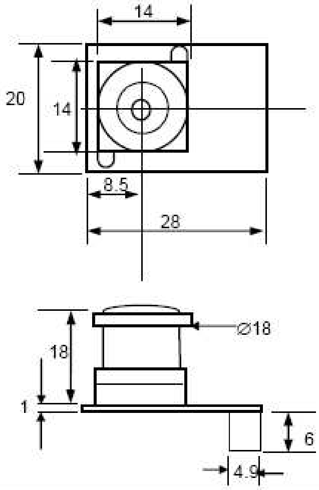
Note: In order to facilitate people for better understanding the communication with the module, we have developed an EV kit, C328-EV232, for user to run under PC Windows environment. However, this module is not designed for PC application. Such PC evaluation is only for better understanding of command control.

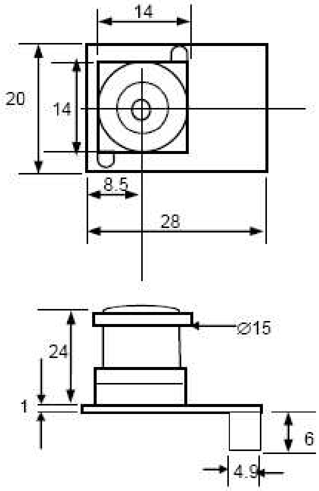
Information of Alternative Lens solution for C328

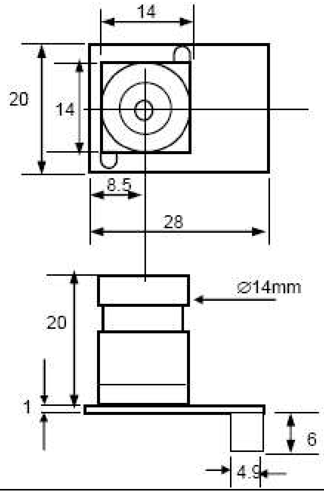
Note: we suggest to use lens of IR cut filter built-in for outdoor application.

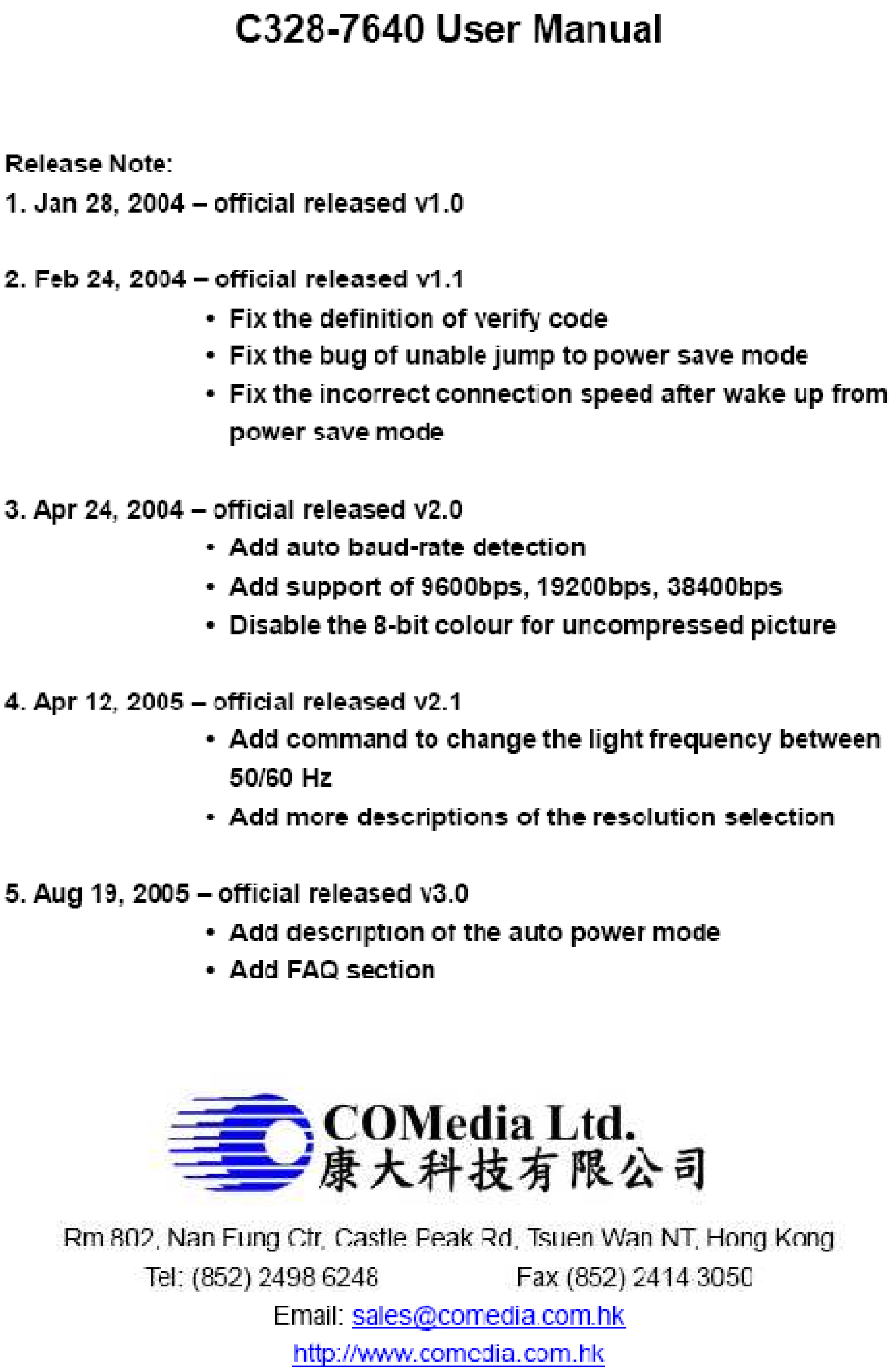
C328-7640 Standard configuration 		Detail spec and dimension, pls refer to the spec sheet of C328 F/No 2.8 f= 4.63mm FOV 57° diagonal IR cut filter built-in

C328-3620BW/IR
No IR cut filter on the lens
 	
		F/No 2.0
		f=3.6mm
		FOV=66° diagonal

C328-2225BW
No IR cut filter on the lens
 	
	
	
		F/No 2.5
		f=2.2mm
		FOV=118° diagonal

C328-2520BW
No IR cut filter on the lens
 	
	
	
		F/No 2.0
		f=2.5mm
		FOV=100° diagonal

C328-6016BW/IR
No IR cut filter on the lens
 	
	
	
		F/No 1.6
		f=6.0mm
		FOV=36° diagonal



C328-7640 User Manual

Release Note:

1. Jan 28, 2004 – official released v1.0

2. Feb 24, 2004 – official released v1.1
   - Fix the definition of verify code
   - Fix the bug of unable jump to power save mode
   - Fix the incorrect connection speed after wake up from power save mode

3. Apr 24, 2004 – official released v2.0
   - Add auto baud-rate detection
   - Add support of 9600bps, 19200bps, 38400bps
   - Disable the 8-bit colour for uncompressed picture

4. Apr 12, 2005 – official released v2.1
   - Add command to change the light frequency between 50/60 Hz
   - Add more descriptions of the resolution selection

5. Aug 19, 2005 – official released v3.0
   - Add description of the auto power mode
   - Add FAQ section

COMedia Ltd.
康大科技有限公司

Rm 802, Nan Fung Ctr, Castle Peak Rd, Tsuen Wan NT, Hong Kong
Tel: (852) 2498 6248 Fax (852) 2414 3050
Email: sales@comedia.com.hk
http://www.comedia.com.hk

General Description

The C328 module is a highly integrated serial camera board that can be attached to a wireless or PDA host performing as a video camera or a JPEG compressed still camera. It provides a serial interface (RS-232) and JPEG compression engine to act as a low cost and low powered camera module for high-resolution serial bus security system or PDA accessory applications.

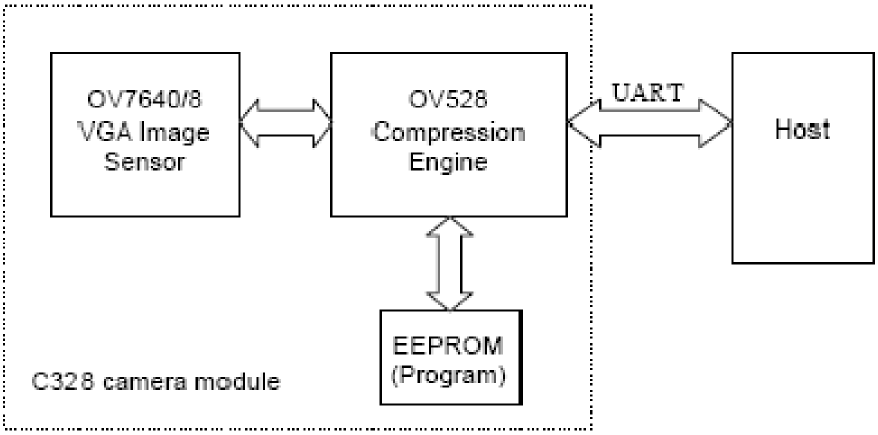


Figure 1 – System block diagram

Features

- ➢ Small in size, low cost and low powered (3.3V) camera module for high-resolution serial bus security system or PDA accessory applications.
- ➢ On-board EEPROM provides a command-based interface to external host via RS-232.
- ➢ UART: 115.2Kbps for transferring JPEG still pictures or 160x128 preview @8bpp with 0.75fps.
- ➢ On board OmniVision OV7640/8 VGA color sensor.
- ➢ Built-in JPEG CODEC for different resolutions.
- ➢ Built-in down sampling, clamping and windowing circuits for VGA, QVGA, 160x120 or 80x60 image resolutions.
- ➢ Built-in color conversion circuits for 2-bit gray, 4-bir gray, 8-bit gray, 12-bit RGB, 16-bit RGB or standard JPEG preview images.
- ➢ No external DRAM required.

System Configuration

1. Camera Sensor
   The C328-7640 module uses OmniVision OV7640/8 VGA color digital CameraChips with an 8-bit YCbCr interface.

2. OV528 Serial Bridge
   The OV528 Serial Bridge is a JPEG CODEC embedded controller chip that can compress and transfer image data from CameraChips to external device. The OV528 takes 8-bit YCbCr 422 progressive video data from an OV7640/8 CameraChip. The camera interface synchronizes with input video data and performs down sampling, clamping and windowing functions with desired resolution, as well as color conversion that is requested by the user through serial bus host commands.
   The JPEG CODEC can achieve higher compression ratio and better image quality for various image resolutions.

3. Program EEPROM
   A serial type program memory is built-in for C328-7640 to provide a set of user-friendly command interfacing to external host.

Board Layout

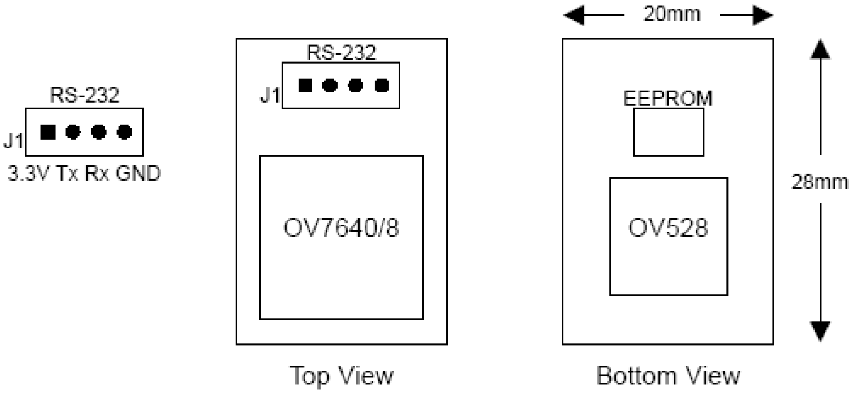


Figure 2 – C328-7640 board layout and serial interface pin

Serial Interface

1. Single Byte Timing Diagram
   A single byte RS-232 transmission consists of the start bit, 8-bit contents and the stop bit. A start bit is always 0, while a stop bit is always 1. LSB is sent out first and is right after the start bit.

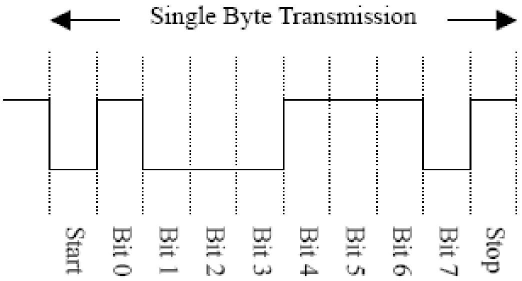


Figure 3 – RS-232 single byte timing diagram

2. Command Timing Diagram
   A single command consists of 6 continuous single byte RS-232 transmissions. The following is an example of SYNC (AA0D00000000h) command.

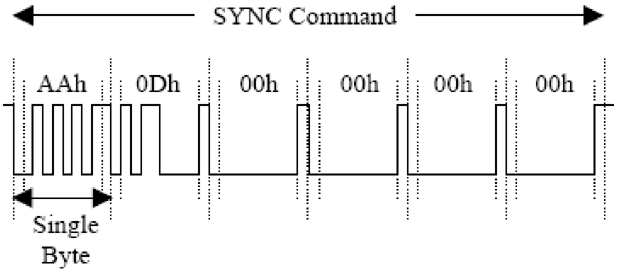


Figure 4 – RS-232 SYNC command timing diagram

Command Set

The C328-7640 module supports total 11 commands for interfacing to host as following:

Command	ID Number	Parameter1	Parameter2	Parameter3	Parameter4
Initial	AA01h	00h	Color Type	RAW Resolution (Still image only)	JPEG Resolution
Get Picture	AA04h	Picture Type	00h	00h	00h
Snapshot	AA05h	Snapshot Type	Skip Frame Low Byte	Skip Frame High Byte	00h
Set Package Size	AA06h	08h	Package Size Low Byte	Package Size High Byte	00h
Set Baudrate	AA07h	1st Divider	2nd Divider	00h	00h
Reset	AA08h	Reset Type	00h	00h	xxh*
Power Off	AA09h	00h	00h	00h	00h
Data	AA0Ah	Data Type	Length Byte 0	Length Byte 1	Length Byte 2
SYNC	AA0Dh	00h	00h	00h	00h
ACK	AA0Eh	Command ID	ACK counter	00h / Package ID Byte 0	00h / Package ID Byte 1
NAK	AA0Fh	00h	NAK counter	Error Number	00h
Light Frequency	AA13h	Frequency Type	00h	00h	00h

* If the parameter is 0xFF, the command is a special Reset command and the firmware responds to it immediately.

1. Initial (AA01h)

The host issues this command to configure the preview image size and color type. After receiving this command, the module will send out an ACK command to the host if the configuration success. Otherwise, an NACK command will be sent out.

1.1 Color Type

C328-7640 can support 7 different color types as follow:

2-bit Gray Scale	01h
4-bit Gray Scale	02h
8-bit Gray Scale	03h
12-bit Color	05h
16-bit Color	06h
JPEG	07h

1.2 Preview Resolution

80x60	01h
160x120	03h

1.3 JPEG Resolution

Since the Embedded JPEG Code can support only multiple of 16, the JPEG preview mode can support following image sizes. It is different from normal preview mode.

80x64	01h
160x128	03h
320x240	05h
640x480	07h

2. Get Picture (AA04h)

The host gets a picture from C328-7640 by sending this command.

2.1 Picture Type

Snapshot Picture	01h
Preview Picture	02h
JPEG Preview Picture	05h

3. Snapshot (AA05h)

C328-7640 keeps a single frame of JPEG still picture data in the buffer after receiving this command.

3.1 Snapshot Type

Compressed Picture	00h
Uncompressed Picture	01h

3.2 Skip Frame Counter

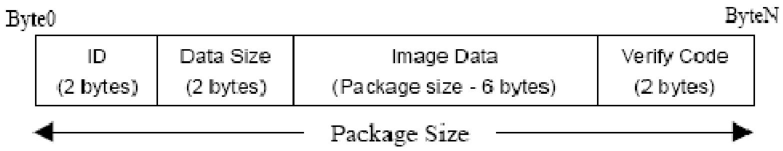
The number of dropped frames can be defined before compression occurs. "0" keeps the current frame, "1" captures the next frame, and so forth.

4. Set Package Size (AA06h)

The host issues this command to change the size of data package which is used to transmit JPEG image data from the C328-7640 to the host. This command should be issued before sending Snapshot command or Get Picture command to C328-7640. It is noted that the size of the last package varies for different image.

4.1 Package Size

The default size is 64 bytes and the maximum size is 512 bytes.



ID (2 bytes)	Data Size (2 bytes)	Image Data (Package size - 6 bytes)	Verify Code (2 bytes)

ID	-> Package ID, starts from zero for an image
Data Size	-> Size of image data in the package
Verify Code	-> Error detection code, equals to the lower byte of sum of the whole package data except the verify code field. The higher byte of this code is always zero. i.e. verify code = lowbyte(sum(byte[0] to byte[N-2]))

Note: As the transmission of uncompressed image is not in package mode, it is not necessary to set the package size for uncompressed image.

5. Set Baudrate (AA07h)

Set the C328-7640 baud rate by issuing this command. As the module can auto-detect the baud rate of the incoming command, host can make connection with one of the following baud rate in the table. The module will keep using the detected baud rate until physically power off

5.1 Baudrate Divider

Baudrate = 14.7456MHz / 2 x (2nd Divider + 1) / 2 x (1st Divider + 1)

Baudrate	1st Divider	2nd Divider	Baudrate	1st Divider	2nd Divider
7200 bps	ffh	01h	28800 bps	3fh	01h
9600 bps	bfh	01h	38400 bps	2fh	01h
14400 bps	7fh	01h	57600 bps	1fh	01h
19200 bps	5fh	01h	115200 bps	0fh	01h

6. Reset (AA08h)

The host reset C328-7640 by issuing this command.

6.1 Reset Type

"00h" resets the whole system. C328-7640 will reboot and reset all registers and state machines. "01h" resets state machines only.

7. Power Off (AA09h)

C328-7640 will go into sleep mode after receiving this command. SYNC command (AA0Dh) must be sent to wake up C328-7640 for certain period until receiving ACK command from C328-7640.

8. Data (AA0Ah)

C328-7640 issues this command for telling the host the type and the size of the image data which is ready for transmitting out to the host.

8.1 Data Type

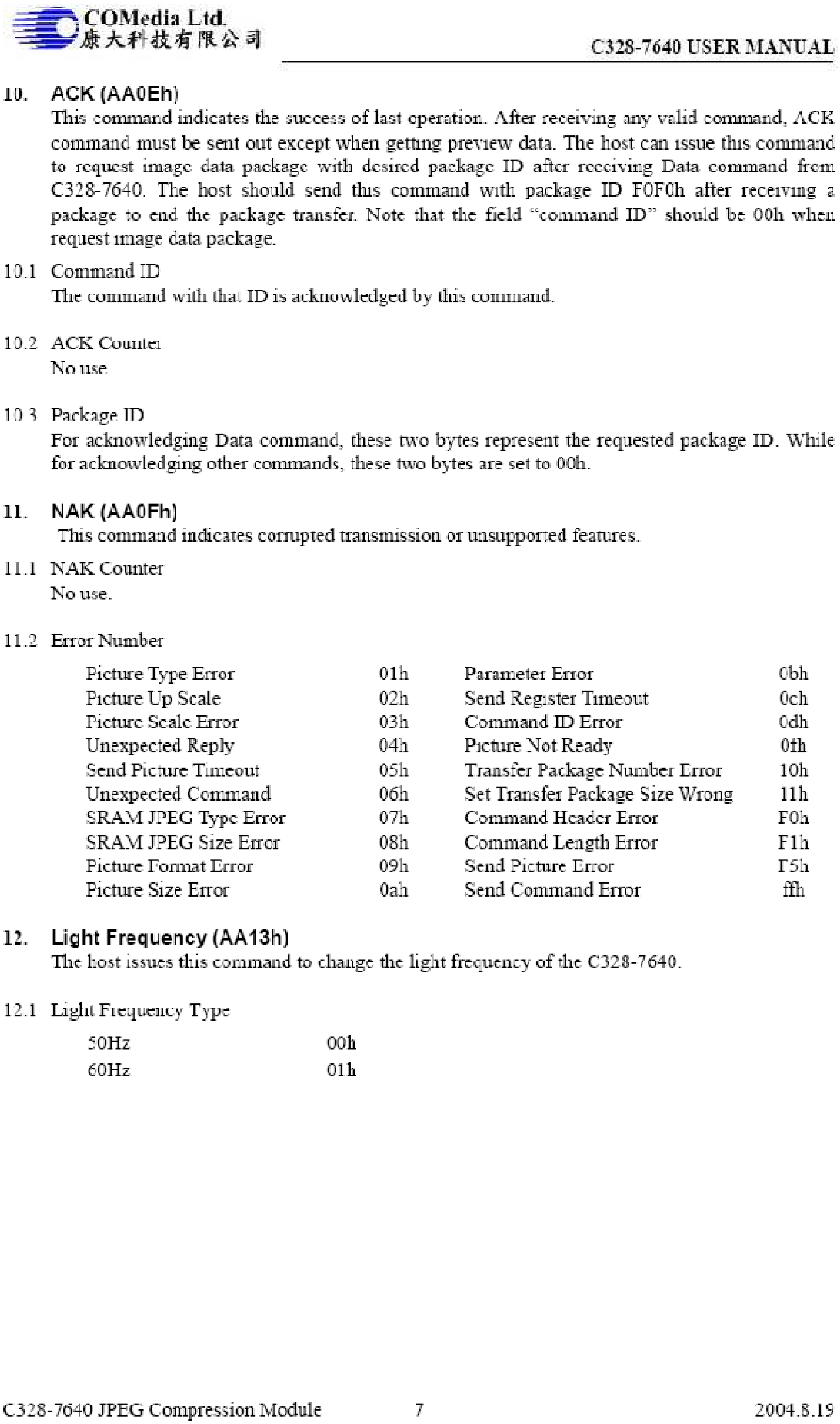
Snapshot Picture	01h
Preview Picture	02h
JPEG Preview Picture	05h

8.2 Length

These three bytes represent the length of data of the Snapshot Picture, Preview Picture or JPEG Preview Picture.

9. SYNC (AA0Dh)

Either the host or the C328-7640 can issue this command to make connection. An ACK command must be sent out after receiving this command.



10. ACK (AA0Eh)

This command indicates the success of last operation. After receiving any valid command, ACK command must be sent out except when getting preview data. The host can issue this command to request image data package with desired package ID after receiving Data command from C328-7640. The host should send this command with package ID F0F0h after receiving a package to end the package transfer. Note that the field "command ID" should be 00h when request image data package.

10.1 Command ID

The command with that ID is acknowledged by this command.

10.2 ACK Counter

No use.

10.3 Package ID

For acknowledging Data command, these two bytes represent the requested package ID. While for acknowledging other commands, these two bytes are set to 00h.

11. NAK (AA0Fh)

This command indicates corrupted transmission or unsupported features.

11.1 NAK Counter

No use.

11.2 Error Number

Picture Type Error	01h	Parameter Error	0bh
Picture Up Scale	02h	Send Register Timeout	0ch
Picture Scale Error	03h	Command ID Error	0dh
Unexpected Reply	04h	Picture Not Ready	0fh
Send Picture Timeout	05h	Transfer Package Number Error	10h
Unexpected Command	06h	Set Transfer Package Size Wrong	11h
SRAM JPEG Type Error	07h	Command Header Error	F0h
SRAM JPEG Size Error	08h	Command Length Error	F1h
Picture Format Error	09h	Send Picture Error	F5h
Picture Size Error	0ah	Send Command Error	ffh

12. Light Frequency (AA13h)

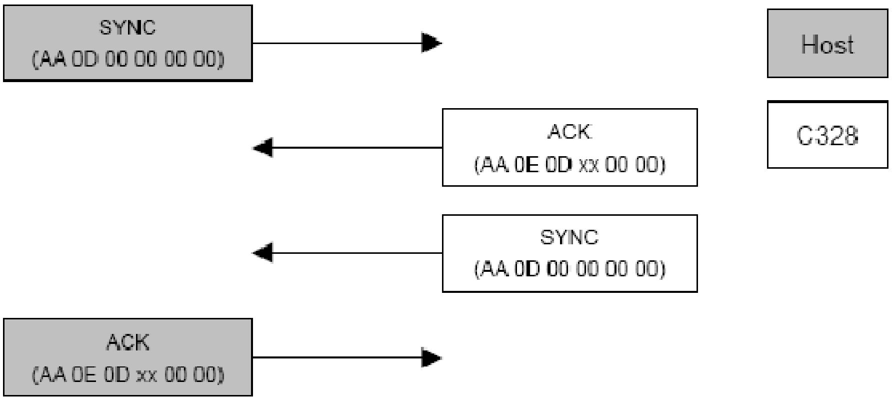
The host issues this command to change the light frequency of the C328-7640.

12.1 Light Frequency Type

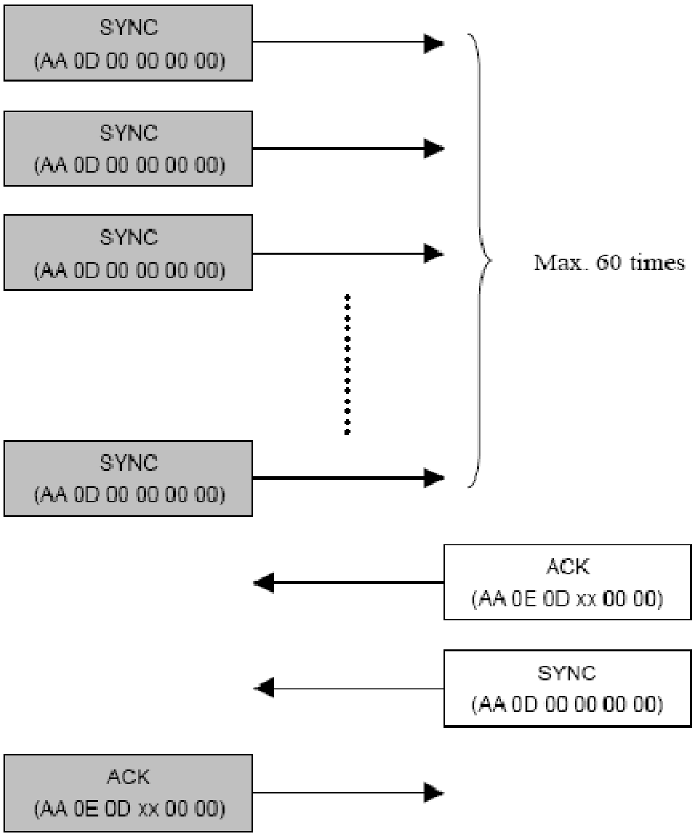
50Hz	00h
60Hz	01h

Command Protocol

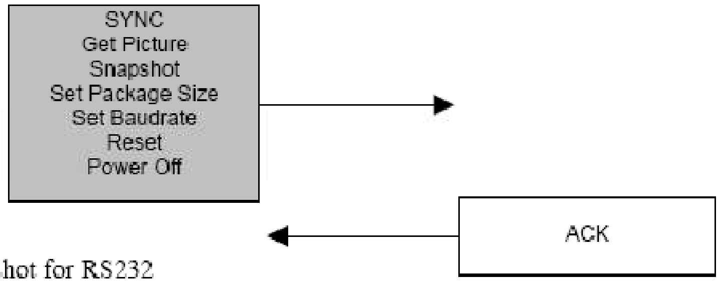
1. SYNC Command



2. Make Connection with C328-7640
Send the SYNC command (at 14400bps) until receiving ACK command from C328-7640 (usually an ACK command is receive after sending 25 times of SYNC command). This must be done after power up.



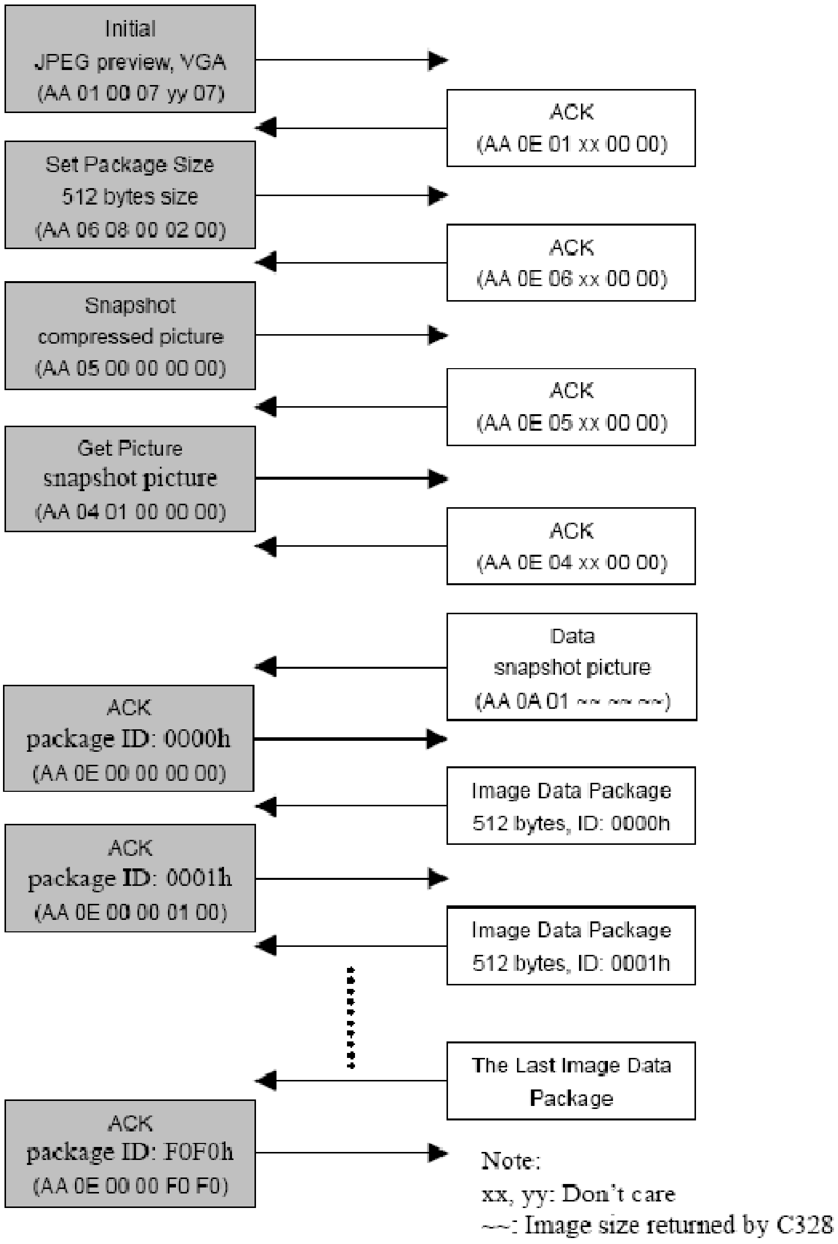
3. Initial, Get Picture, Snapshot, Set Package Size, Set Baudrate, Reset and Power Off Command



4. Getting a Snapshot for RS232

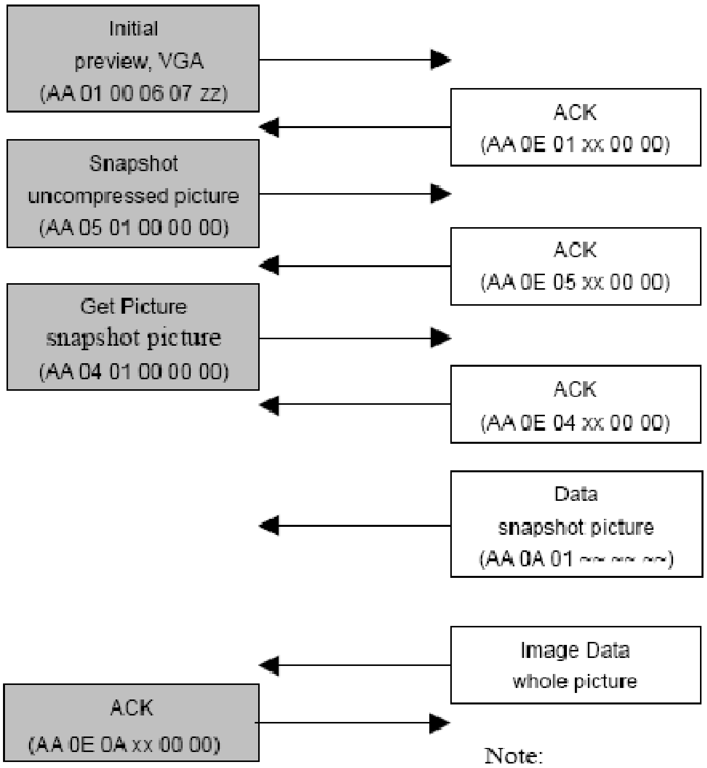
Make sure connection is made before the following communication.

4.1 JPEG Snapshot Picture (eg. 640x480 resolution)



Note:
xx, yy: Don't care
~~: Image size returned by C328

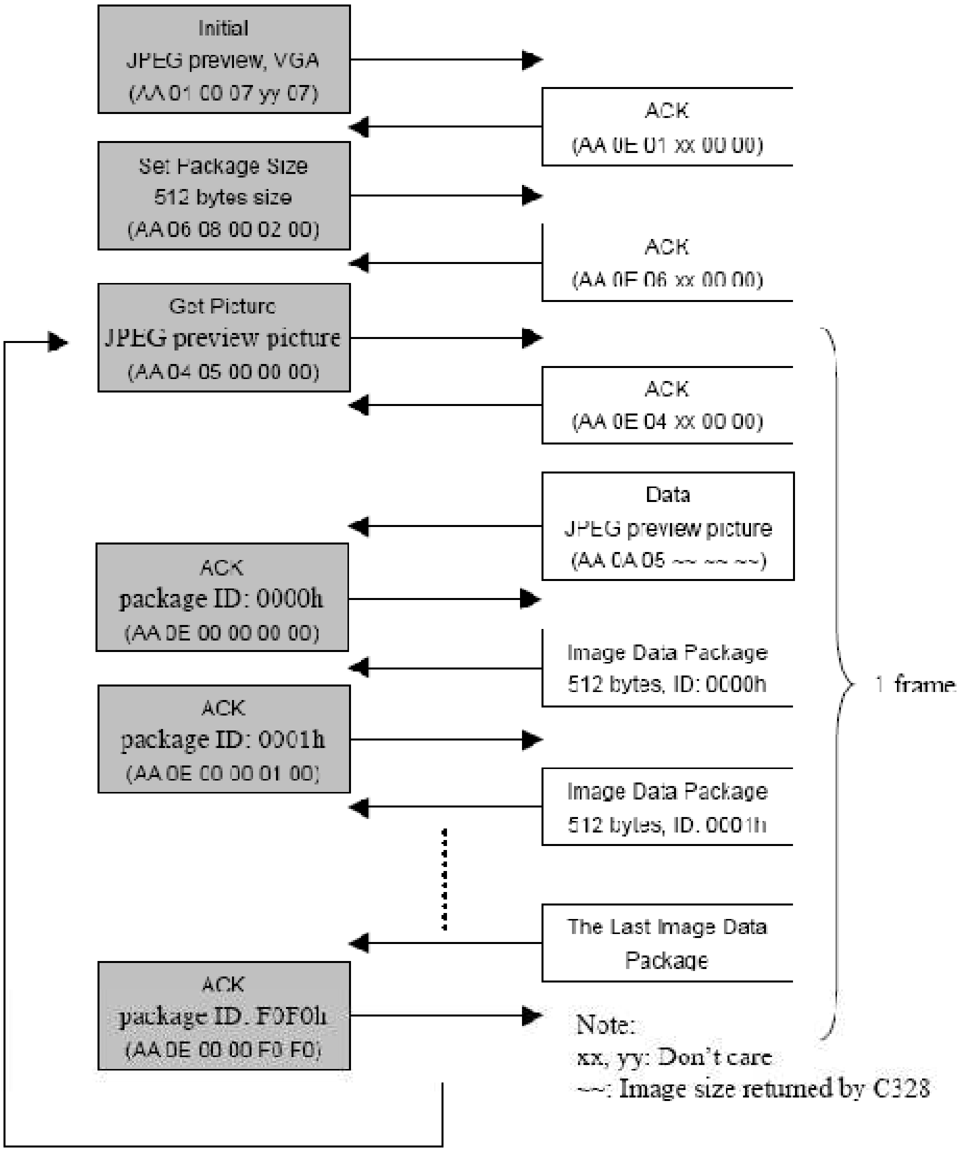
4.2 Snapshot Picture (uncompressed snapshot picture)



Note:
xx, zz : Don't care
~~: Image size returned by C328

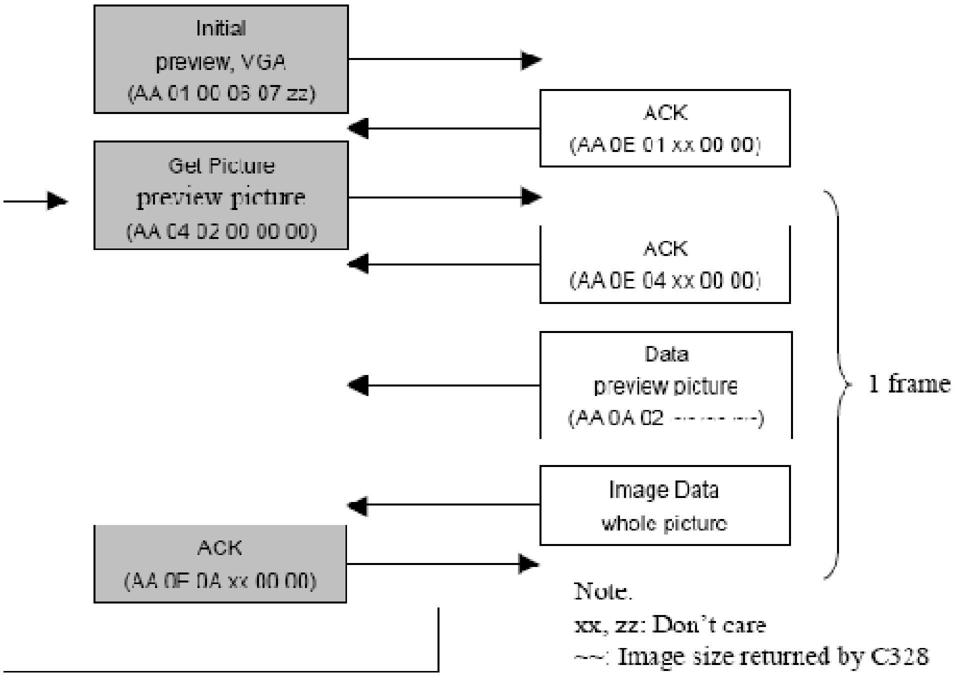
5. Getting JPEG preview pictures (video) for RS232
   Make sure connection is made before the following communication.

5.1 JPEG Preview Picture

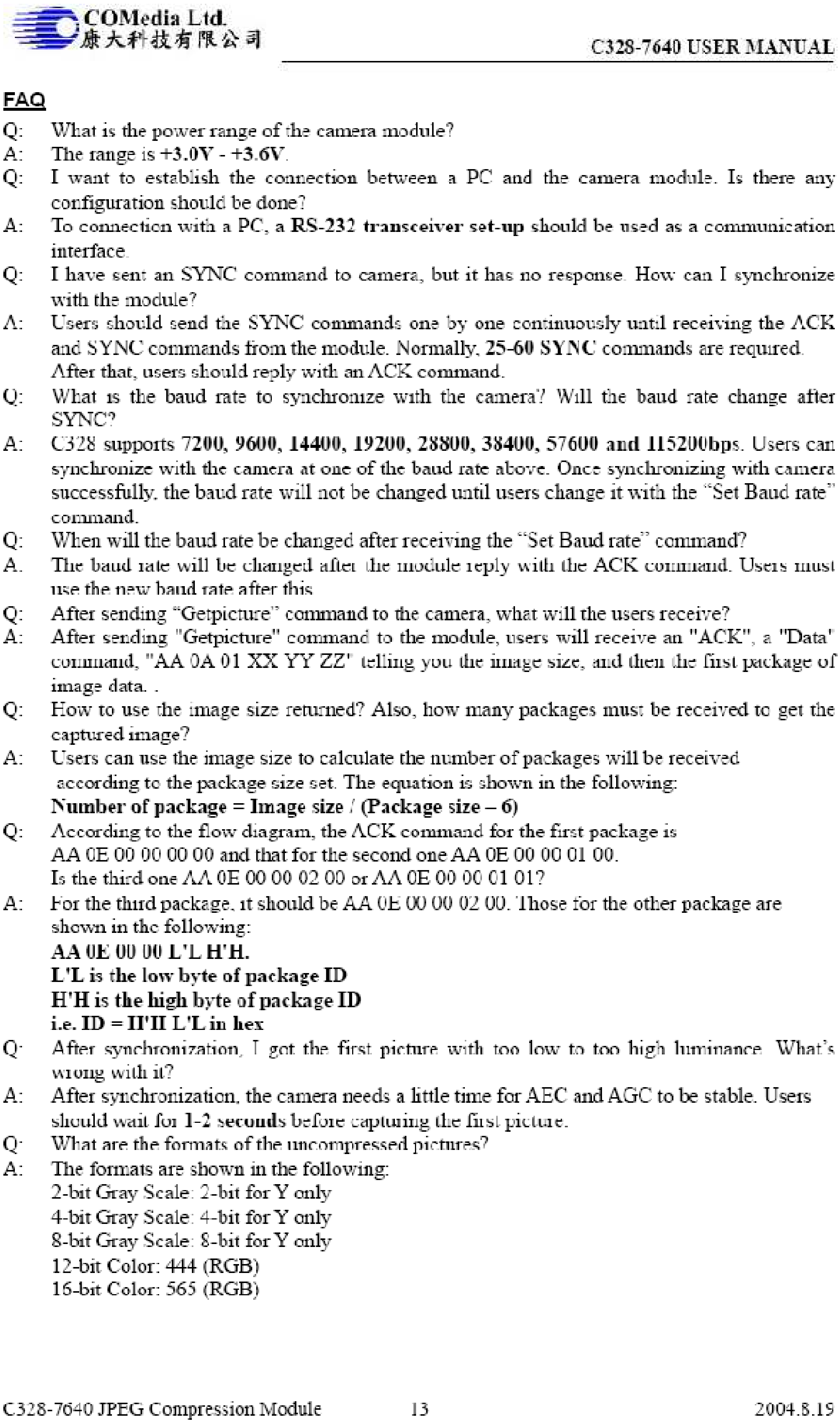


Note:
xx, yy: Don't care
~~: Image size returned by C328

5.2 Preview Picture (uncompressed preview picture)



Note.
xx, zz: Don't care
~~: Image size returned by C328



FAQ

Q: What is the power range of the camera module?

A: The range is **+3.0V - +3.6V**.

Q: I want to establish the connection between a PC and the camera module. Is there any configuration should be done?

A: To connection with a PC, a **RS-232 transceiver set-up** should be used as a communication interface.

Q: I have sent an SYNC command to camera, but it has no response. How can I synchronize with the module?

A: Users should send the SYNC commands one by one continuously until receiving the ACK and SYNC commands from the module. Normally, **25-60 SYNC** commands are required.
After that, users should reply with an ACK command.

Q: What is the baud rate to synchronize with the camera? Will the baud rate change after SYNC?

A: C328 supports **7200, 9600, 14400, 19200, 28800, 38400, 57600 and 115200bps**. Users can synchronize with the camera at one of the baud rate above. Once synchronizing with camera successfully, the baud rate will not be changed until users change it with the "Set Baud rate" command.

Q: When will the baud rate be changed after receiving the "Set Baud rate" command?

A. The baud rate will be changed after the module reply with the ACK command. Users must use the new baud rate after this

Q: After sending "Getpicture" command to the camera, what will the users receive?

A: After sending "Getpicture" command to the module, users will receive an "ACK", a "Data" command, "AA 0A 01 XX YY ZZ" telling you the image size, and then the first package of image data. .

Q: How to use the image size returned? Also, how many packages must be received to get the captured image?

A: Users can use the image size to calculate the number of packages will be received according to the package size set. The equation is shown in the following:
Number of package = Image size / (Package size – 6)

Q: According to the flow diagram, the ACK command for the first package is
AA 0E 00 00 00 00 and that for the second one AA 0E 00 00 01 00.
Is the third one AA 0E 00 00 02 00 or AA 0E 00 00 01 01?

A: For the third package, it should be AA 0E 00 00 02 00. Those for the other package are shown in the following:
AA 0E 00 00 L'L H'H.
L'L is the low byte of package ID
H'H is the high byte of package ID
i.e. ID = H'H L'L in hex

Q: After synchronization, I got the first picture with too low to too high luminance. What's wrong with it?

A: After synchronization, the camera needs a little time for AEC and AGC to be stable. Users should wait for **1-2 seconds** before capturing the first picture.

Q: What are the formats of the uncompressed pictures?

A: The formats are shown in the following:
2-bit Gray Scale: 2-bit for Y only
4-bit Gray Scale: 4-bit for Y only
8-bit Gray Scale: 8-bit for Y only
12-bit Color: 444 (RGB)
16-bit Color: 565 (RGB)

C328-EV232 EV BOARD

C328-EV232
EV BOARD

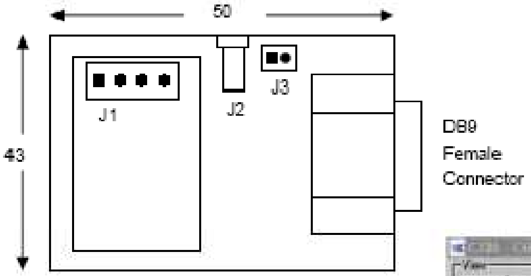
General Description

The C328-EV232 EV board is a simple application reference for the C328 Camera Module. It is built with RS232 transceivers to connect the C328 Module to a PC. A user-friendly PC driver is provided for user to evaluate the key functions of the C328 camera module.

Features

- DC adapter or Battery operate
- Direct connect to PC serial port
- PC driver provided
- OS Support: Win98/ME/XP
- Different image size setting
- Preview window
- Captured image window
- Light frequency setting

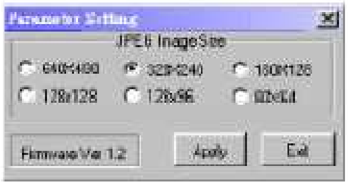
Board Layout



Connector Description

J1	4 pin header 2mm pitch for C328
J2	DC Power Jack, 5VDC input
J3	DC input, 3.6V or 4.5V
P1	9pin D type, female

PC Interface

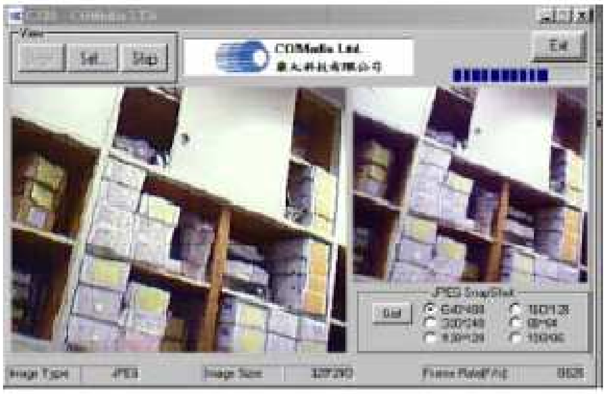


Parameter Setting

JPEG ImageSize

○ 640x480 ● 320x240 ○ 160x128
○ 128x128 ○ 128x96 ○ 80x64

Firmware Ver 1.2

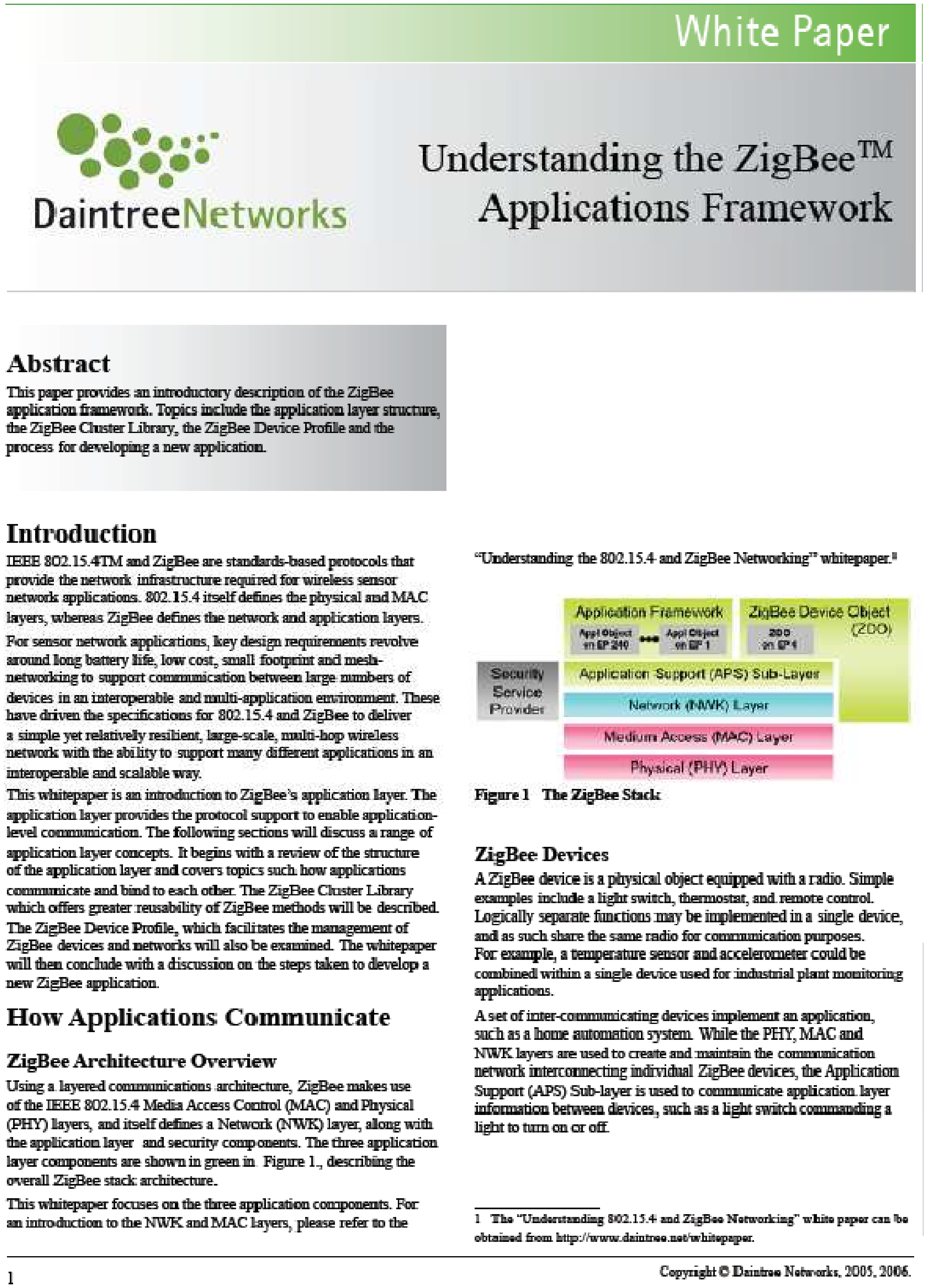


Kit Package

Parts	Qty	Description
EV232	1pc	EV board
RS232 Cable	1pc	Serial Cable
2-wire cable	1pc	Power cable for battery
DC adapter	1pc	AC/DC adapter with 5V
CD ROM	1pc	Document and PC driver

Update: May 10, 2004

Zigbee Applications Framework



White Paper

DaintreeNetworks

Understanding the ZigBee™ Applications Framework

Abstract

This paper provides an introductory description of the ZigBee application framework. Topics include the application layer structure, the ZigBee Cluster Library, the ZigBee Device Profile and the process for developing a new application.

Introduction

IEEE 802.15.4TM and ZigBee are standards-based protocols that provide the network infrastructure required for wireless sensor network applications. 802.15.4 itself defines the physical and MAC layers, whereas ZigBee defines the network and application layers.

For sensor network applications, key design requirements revolve around long battery life, low cost, small footprint and mesh-networking to support communication between large numbers of devices in an interoperable and multi-application environment. These have driven the specifications for 802.15.4 and ZigBee to deliver a simple yet relatively resilient, large-scale, multi-hop wireless network with the ability to support many different applications in an interoperable and scalable way.

This whitepaper is an introduction to ZigBee's application layer. The application layer provides the protocol support to enable application-level communication. The following sections will discuss a range of application layer concepts. It begins with a review of the structure of the application layer and covers topics such how applications communicate and bind to each other. The ZigBee Cluster Library which offers greater reusability of ZigBee methods will be described. The ZigBee Device Profile, which facilitates the management of ZigBee devices and networks will also be examined. The whitepaper will then conclude with a discussion on the steps taken to develop a new ZigBee application.

How Applications Communicate

ZigBee Architecture Overview

Using a layered communications architecture, ZigBee makes use of the IEEE 802.15.4 Media Access Control (MAC) and Physical (PHY) layers, and itself defines a Network (NWK) layer, along with the application layer and security components. The three application layer components are shown in green in Figure 1., describing the overall ZigBee stack architecture.

This whitepaper focuses on the three application components. For an introduction to the NWK and MAC layers, please refer to the "Understanding the 802.15.4 and ZigBee Networking" whitepaper.[1]

Application Framework
Appl Object on EP 240
Appl Object on EP 1
ZigBee Device Object (ZDO)
ZDO on EP 0
Security Service Provider
Application Support (APS) Sub-Layer
Network (NWK) Layer
Medium Access (MAC) Layer
Physical (PHY) Layer

Figure 1 The ZigBee Stack

ZigBee Devices

A ZigBee device is a physical object equipped with a radio. Simple examples include a light switch, thermostat, and remote control. Logically separate functions may be implemented in a single device, and as such share the same radio for communication purposes. For example, a temperature sensor and accelerometer could be combined within a single device used for industrial plant monitoring applications.

A set of inter-communicating devices implement an application, such as a home automation system. While the PHY, MAC and NWK layers are used to create and maintain the communication network interconnecting individual ZigBee devices, the Application Support (APS) Sub-layer is used to communicate application layer information between devices, such as a light switch commanding a light to turn on or off.

[1] The "Understanding 802.15.4 and ZigBee Networking" white paper can be obtained from http://www.daintree.net/whitepaper.

the attic switch to separate endpoints on the hallway/attic device, the same radio device can allow independent control of independent attached lights. The remote control, bound to the two hallway lights will independently control these lights, while the attic switch is independently bound to the attic light.

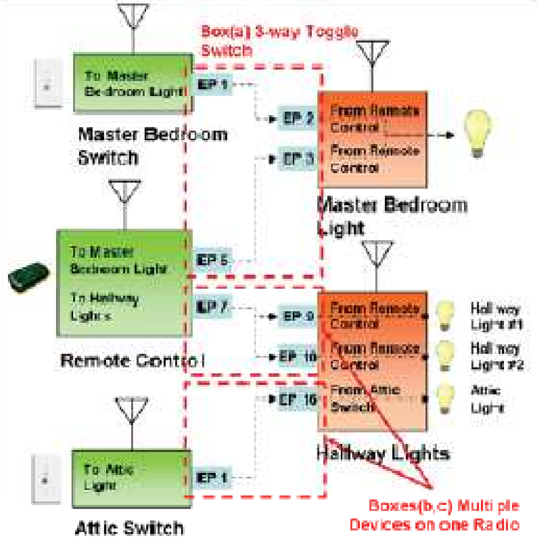


Figure 3 Complex Binding Example

Where are Bindings Stored?

The storage of binding information is an important design consideration. The most obvious place to store binding information is within the source device. For instance, a remote control could store the addresses and endpoint IDs of all the applications it needs to communicate with (see Figure 4). In this case, it uses *direct binding* by constructing a packet that contains all the information necessary to send that packet to its peer application. This is sometimes known as *source binding*, since the source device has the necessary binding information.

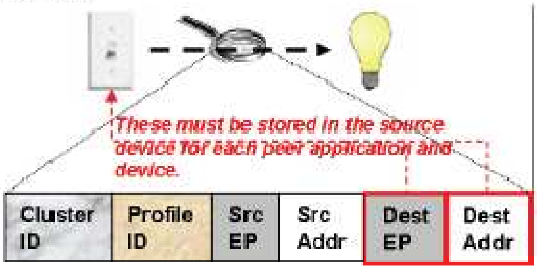

These must be stored in the source device for each peer application and device.
Cluster ID
Profile ID
Src EP
Src Addr
Dest EP
Dest Addr

Figure 4 Direct Binding[2]

Several limitations exist with direct binding. The device must have sufficient memory to store all this information about all its peer applications and devices. However, this might not be possible (or too expensive) for a simple device. When the device fails, the binding information stored within the device may be lost. Should its peer device be replaced or fail, updating this information in the device will be difficult.

One solution that ZigBee offers is to use a binding cache. This can be located in an intermediary device that provides a lookup table mapping the source endpoint and address to the corresponding destination endpoint and address (see Figure 5).

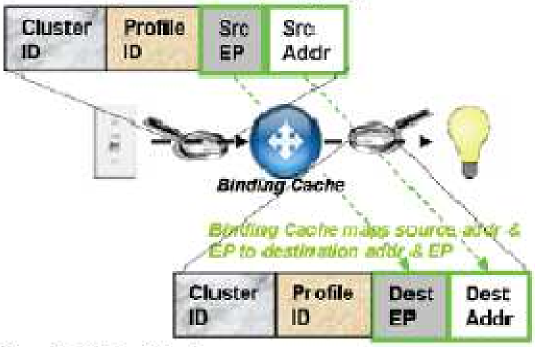


Figure 5 Indirect Binding

Indirect binding through a cache does add additional communication overhead, since all messages must go through the cache, rather than directly to the end device. However, indirect binding does eliminate the need for additional memory in the end device. It also enables more effective binding management. If a device needs to be replaced, only the binding cache needs to be notified rather than all devices that had a binding relationship with the device being replaced. Should the source device fail, a replacement device could simply notify the binding cache that it is replacing the previous device and reuse the bindings from the old device.

In the next revision of the ZigBee specification (v1.1)[3], a collection of new methods provide more substantial binding cache management capabilities. These include support for a secondary cache to backup and restore the primary cache in case of failure. Revision 1.1 also offers new methods to download binding information from a cache to an end device to enable the end device to use direct binding, thereby eliminating the need for communication through the cache, while still preserving the benefits of a cache.

ZigBee Cluster Library

Another significant addition to the ZigBee specification in revision 1.1 is the ZigBee Cluster Library (ZCL). As mentioned earlier, a cluster is a message, or a collection of similar messages. The ZCL offers cluster reusability by abstracting clusters used across many applications and placing them in a library encompassed by a particular *functional domain*. As an example, clusters for

[2] The fields shown in this diagram are a subset of the fields in the packet and not necessarily in the order they are communicated. They merely represent the critical fields for the transaction.

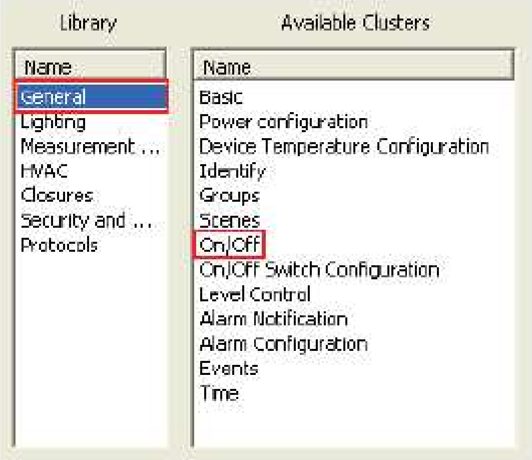
[3] At the time writing, this new revision is expected to be ratified in Q3, 2006. Note too that this document will use the term "v1.1" to describe this new revision although the ZigBee Alliance has not yet confirmed the use of this term for this new release.

controlling lighting devices, which are used in both residential and commercial *application domains* reside in a library associated with a lighting functional domain and may be used in both residential and commercial building automation, as well as other applications.

How Does The ZCL Work?

The ZCL defines a library[4] of clusters which may be utilized by any application. The designer simply extracts the needed clusters from the relevant libraries and assigns a cluster ID to each such cluster.

To demonstrate this, consider a fragment of a Profile Editor[5] shown in Figure 6. Here, we observe a list of available libraries from ZigBee Cluster Library (left panel) – General, Lighting, and so forth. For each library, a number of clusters are available. In Figure 6, the clusters for the selected library, the General library, are listed on the right panel – Basic, Power Configuration and so forth.



Library	Available Clusters
Name	Name
General	Basic
Lighting	Power configuration
Measurement ...	Device Temperature Configuration
HVAC	Identify
Closures	Groups
Security and ...	Scenes
Protocols	On/Off
	On/Off Switch Configuration
	Level Control
	Alarm Notification
	Alarm Configuration
	Events
	Time

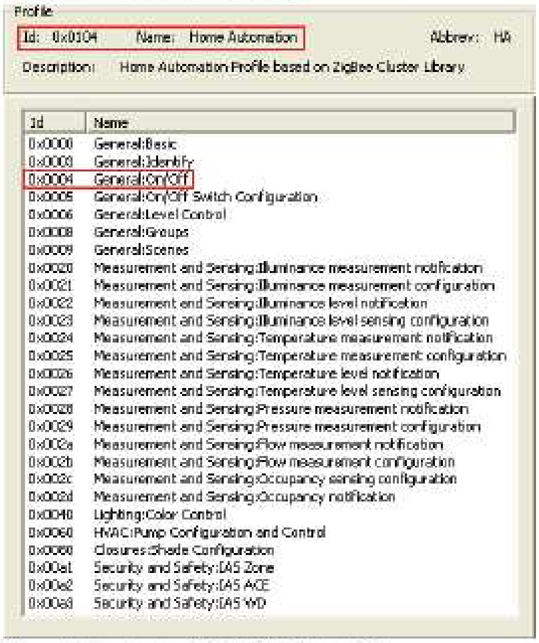
Figure 6 ZigBee Cluster Library

Application developers can select the clusters they require from ZCL to construct their new private application profile in its entirety if all necessary clusters are already defined in the library. Alternatively, some of the clusters from the library may be used as a base, and additional clusters may be added as required by the specific application.

As an example, consider the Home Automation profile. This profile is one of the first application profiles to use the ZCL. The clusters used by the Home Automation profile are shown in another fragment of the same Profile Editor in Figure 7. The cluster IDs assigned to each of these clusters are listed on the left column. To examine the details, use Figure 6 and Figure 7 as a reference and following the red rectangles – cluster ID 0x0004 of profile ID 0x0104 (Home Automation) is the On/Off control cluster, which is sourced from the General library's On/Off cluster.

It should be noted that the Home Automation application profile cluster list defines the complete set of clusters that are required by all supported device types in the Home Automation application. As a result, the list of clusters shown in Figure 7 encompasses devices ranging from simple light switches and lights, to heating and security systems.

By using this method, any new (standard or private) application profile can reuse and inherit clusters that were previously defined in the cluster library to get started rapidly.



Profile

Id: 0x0104 Name: Home Automation Abbrev: HA

Description: Home Automation Profile based on ZigBee Cluster Library

Id	Name
0x0000	General:Basic
0x0003	General:Identify
0x0004	General:On/Off
0x0005	General:On/Off Switch Configuration
0x0006	General:Level Control
0x0008	General:Groups
0x0009	General:Scenes
0x0020	Measurement and Sensing:Illuminance measurement notification
0x0021	Measurement and Sensing:Illuminance measurement configuration
0x0022	Measurement and Sensing:Illuminance level notification
0x0023	Measurement and Sensing:Illuminance level sensing configuration
0x0024	Measurement and Sensing:Temperature measurement notification
0x0025	Measurement and Sensing:Temperature measurement configuration
0x0026	Measurement and Sensing:Temperature level notification
0x0027	Measurement and Sensing:Temperature level sensing configuration
0x0028	Measurement and Sensing:Pressure measurement notification
0x0029	Measurement and Sensing:Pressure measurement configuration
0x002a	Measurement and Sensing:Flow measurement notification
0x002b	Measurement and Sensing:Flow measurement configuration
0x002c	Measurement and Sensing:Occupancy sensing configuration
0x002d	Measurement and Sensing:Occupancy notification
0x0040	Lighting:Color Control
0x0060	HVAC:Pump Configuration and Control
0x0080	Closures:Shade Configuration
0x00a1	Security and Safety:IAS Zone
0x00a2	Security and Safety:IAS ACE
0x00a3	Security and Safety:IAS WD

Figure 7 Home Automation Profile Cluster List

A More Detailed Look at the ZigBee Cluster Library – Device Descriptors

For a deeper look at the ZCL framework, consider the Home Automation example discussed earlier. As an application, Home Automation includes a number of different devices, including a basic *On/Off Switch* to control the lights, and *Pump Controls* to manage the heating system, amongst others.

An *On/Off Switch* to control a light requires little more than an *On/Off command*. Referring to Figure 7, the *General:On/Off cluster (Id = 0x0004)* is the main cluster used by such a device. This provides the command to turn a device on, off, or to toggle its on/off state. Remember, the device that actually receives this command from the switch will be the device that the switch is bound to (as discussed in earlier sections on binding).

However, the Home Automation profile also includes three other clusters with the "on/off switch", largely for management purposes. These are:

- *General:On/Off Switch Configuration cluster (Id = 0x0005)* which is used to configure the switch, for instance to be a toggle switch, as distinct from a pushbutton switch.

[4] ZigBee specifications call these libraries, "Functional Domains"

[5] Sourced from the Daintree Networks Profile Editor.

- *General:Basic cluster (Id = 0x0000)* which is used to store general information such the switch's revision number, manufacturer and physical location (if used)

- *General:Identify cluster (Id = 0x0003)* which is used to tag a device (or more likely, a collection of devices) for management usage (such as commissioning).

Collectively, these four clusters, define the required commands that must be implemented on the Home Automation *On/Off Switch*[6].

To complete the picture, consider Figure 8. In box (a) at the top, the relationship between the cluster library and an application profile is highlighted – the application profile extracts the required clusters from the libraries and assigns cluster IDs to them (designated by the number in the top-right box). A different profile may use the same cluster, but could assign a different cluster ID to that cluster. As a result, the combination of the profile ID and the cluster ID uniquely identifies a cluster within the context of that profile.

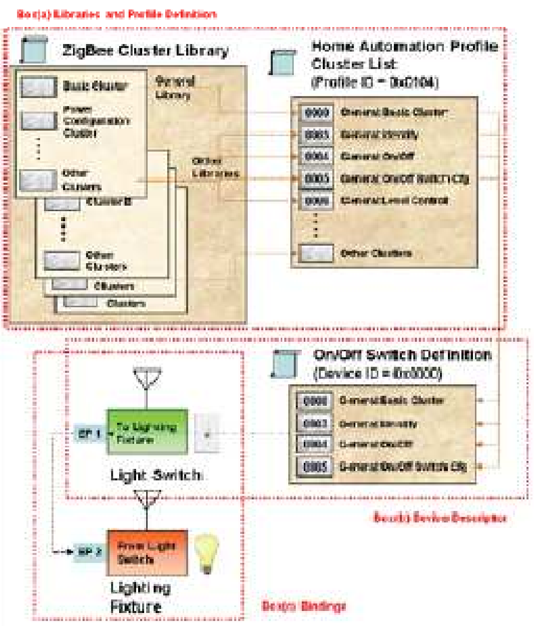


Figure 8 A Complete View of a ZigBee Application

In the bottom-right of Figure 8, individual devices (within the application profile) use a subset of the clusters from the application profile to define a device, in this case an *On/Off Switch*[7]. This device definition is then associated to a particular endpoint on the actual device. The combination of the device definition and the endpoint it is associated to is often known as the device descriptor, highlighted by box(b). A more complex device, such as a remote control could have device definitions attached to multiple endpoints, as mentioned earlier. And finally, using one of the binding techniques mentioned earlier, that endpoint (1) is bound to a corresponding endpoint (2) on another device (as shown in box(c)), which enables communication between the two devices. Specifically, the act of turning on the light will be handled by sending a cluster 0x0004 *(On/Off cluster)* message from endpoint 1 of the light switch to endpoint 2 of the light that it is bound to.

The ZigBee Device Profile

In the earlier discussion on bindings, references were made to commands being available to manage bindings. These commands are part of a suite of commands within the ZigBee Device Object, or ZDO. This object provides management commands common to all ZigBee applications and devices and, as shown in Figure 1, it is implemented on endpoint zero of all ZigBee devices[8]. The definitions of the clusters used by ZDO are described by the ZigBee Device Profile.

At the time of writing, the following management functions are supported by this profile.

- Device and Service Discovery
- Binding Management
- Network Management

These commands are over-the-air commands, enabling any ZigBee device or tool to use them for a variety of different purposes, including commissioning and management.

Device Discovery commands offer the means to determine what devices are on the network, their addresses and the list of their children devices. They can also provide information about the device, including whether it is coordinator, router or end device, manufacturer or product information and even its power source and current battery level.

Service Discovery commands, in contrast, enable the determination of services offered by devices. Using these commands, it possible to determine which endpoints are active on a device, what profiles are associated with each endpoint (i.e., the device descriptor, as mentioned earlier) and to match device descriptors between two devices.

Binding Management commands offer the means to manage bindings between devices. This is explored in greater detail in a sidebar.

Finally, *Network Management* commands provide a way to collect information and control devices for network management purposes. They provide information such as the list of ZigBee networks that a device is able to detect, the quality of the radio link with its neighbors, and the contents of its routing and binding tables. For control purposes, there are commands to instruct a device to join or leave the network.

A Closer Look At Binding Management

One of the more significant changes to the ZigBee specification in release 1.1 is binding management. In this release, a series of commands have been introduced to provide even more flexibility to binding management. Since these changes are relatively new and important, it is worth investigating them in greater detail. This discussion will also elaborate further on how ZigBee Device Profile commands can be used, starting with the existing binding commands,

[6] Some clusters are mandatory, others are optional.

[7] Devices would normally only support the necessary clusters for that device in order to minimize the amount of code required for that device.

[8] While support for the ZDO is required on all ZigBee devicecs, not all ZDO commands are mandatory.

followed by a description of several new and important commands.

The *End_Device_Bind* command is used to facilitate binding through external stimulus. As an example, two end devices may have binding buttons, which, when pressed would send an *End_Device_Bind* command to the binding cache, as shown in Figure 9. The binding cache then attempts to match the profile and cluster IDs[9] from the two devices if both commands were received within a preconfigured timeout period from each other. If a successful match is made, the binding cache will add this binding to the binding table. Thereafter, indirect binding, as described earlier, can occur.

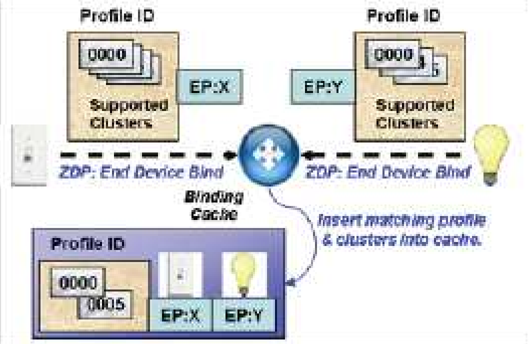


Figure 9 End Device Bind Command

When a device knows exactly which device it needs to bind itself to, it can do one of two things – send messages directly (*direct* or *source binding*), or it may choose to send that information to the binding cache (as shown in Figure 10), and thereafter, use *indirect binding* through the binding cache. In the latter case, the *Bind* command facilitates this. It should be noted that the *Bind command* can be sent by any device, including one of the two devices that are part of the binding, or a device that is not part of the binding.

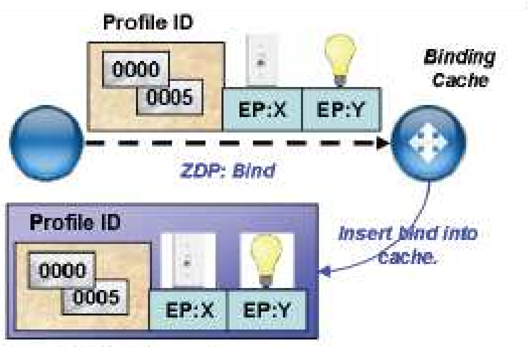


Figure 10 Bind Command

These two binding commands, supported in earlier releases of the ZigBee specification have been enhanced through the following additional commands:

- *Bind_Register*, which allows devices to register with the binding cache and download all bindings stored in the cache for that device.

- *Replace_Device*, which allow a new device (Y) to request that entries within the binding cache for an old device (X) be replaced by the new device (Y).

- *Backup_Bind_Table* and *Recover_Bind_Table* to backup the primary binding cache to the second binding cache, and to recover the information from the secondary binding cache.

These are a few of the new commands available to facilitate binding management and open up a variety of new options for managing bindings. To explain this, consider the scenario shown in in Figure 11. Starting in box (a), end devices are bound using some physical input (say, a button on each device and the remote control). Using the *End_Device_Bind* command, these bindings populate the binding cache.

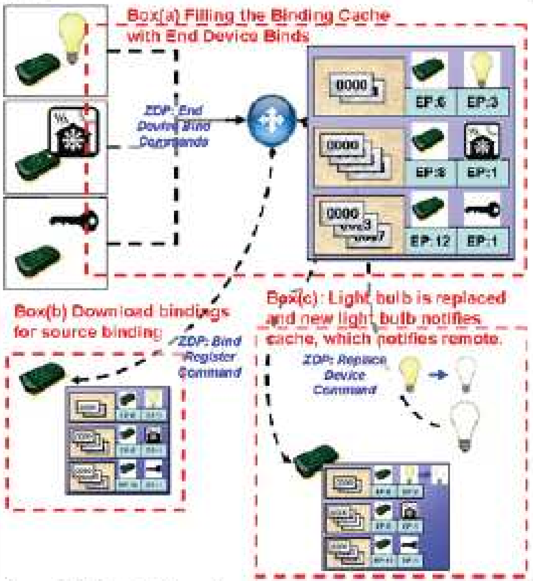

Box(a) Filling the Binding Cache with End Device Binds
ZDP: End Device Bind Commands
0000
EP:6
EP:3
0000
EP:8
EP:1
0000
EP:12
EP:1
Box(b) Download bindings for source binding
ZDP: Bind Register Command
Box(c): Light bulb is replaced and new light bulb notifies cache, which notifies remote.
ZDP: Replace Device Command

Figure 11 Binding Scenario

In box (b), the remote control then issues a Bind_Register command to download all bindings for which it is the source device. This then allows the remote control to directly communicate to other devices without communicating via the binding cache. Finally, in box (c), should the lighting device fail, a replacement device would be substituted, and it would issue a *Replace_Device* notification to the binding cache that the original lighting device should be replaced by the new lighting device. The binding cache would then note that the remote control currently stores its own copy of its bindings for source binding, and would then send a *Replace_Device* command to the remote control to have its local copy of the binding replaced as well.

This short discussion describes a specific binding management

[9] Cluster are asymmetric in the sense that communication from one device to another will, in general never occur in the reverse direction. As an example, a light switch can initiate a on/off command to the light bulb, but never the reverse. The ZigBee specification therefore describes supported clusters in terms input and output clusters, and when the binding cache matches clusters, it will match the input cluster of one device to the output of the other.

scenario for this specific application. Of course, each application will be different, but a range of binding command are available to facilitate many different scenarios.

Developing A New Application

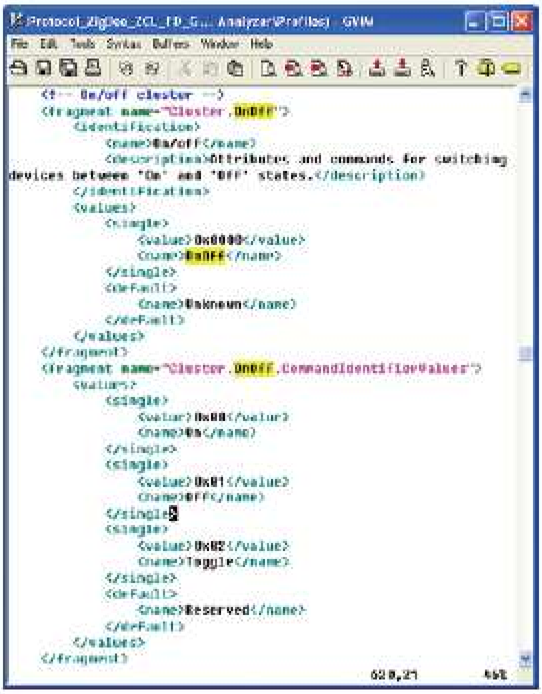
Building on earlier discussions, this section describes a typical process for developing a new application.

Defining and Implementing the Application Profile

The first step is to define the application profile. As part of this exercise, an application profile, along with device definitions are required to meet the specific requirements of the application. As mentioned in the discussion on the ZigBee Cluster Library, where possible this library should be used to leverage existing definitions and code available from the platform provider. As shown in Figure 8, the application profile, along with the device definition ultimately leads to device descriptors that are downloaded into devices.

Following the definition of the application profile is of course, the development of the necessary code for devices around these definitions. With standardization around the ZCL framework, development tools will be increasingly available using standard description languages such as XML, as well as for analysis and testing. This facilitates a single definition of the application profile that may propagate through the entire development, test and commissioning tool chain.

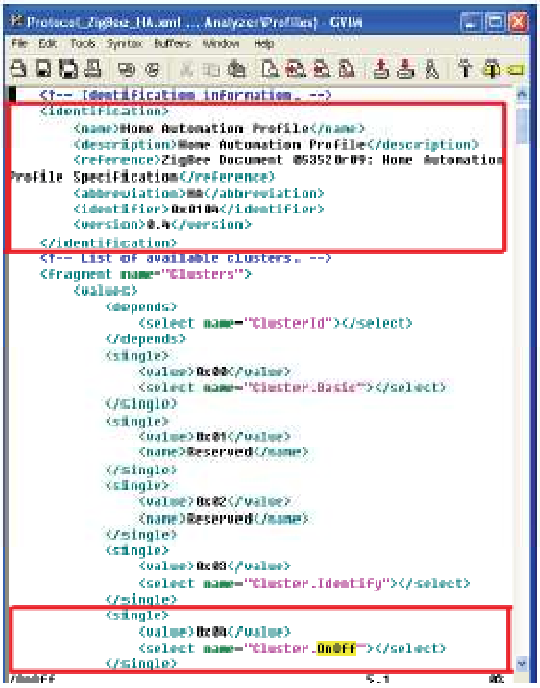
To demonstrate the use of XML to achieve this, consider Figure 12, which shows a fragment of XML code used to define the ZCL. In this fragment, the On/Off cluster from the ZCL General library (see Figure 6) is described. This defines the over-the-air message format, including the available commands (in the bottom half), such as On, Off and Toggle.


Protocol_ZigBee_ZCL_FD_G... Analyzer\Profiles) - GVIM
File Edit Tools Syntax Buffers Window Help

```
    <!-- On/off cluster -->
    <fragment name="Cluster.OnOff">
        <identification>
            <name>On/off</name>
            <description>Attributes and commands for switching
devices between 'On' and 'Off' states.</description>
        </identification>
        <values>
            <single>
                <value>0x0000</value>
                <name>OnOff</name>
            </single>
            <default>
                <name>Unknown</name>
            </default>
        </values>
    </fragment>
    <fragment name="Cluster.OnOff.CommandIdentifierValues">
        <values>
            <single>
                <value>0x00</value>
                <name>On</name>
            </single>
            <single>
                <value>0x01</value>
                <name>Off</name>
            </single>
            <single>
                <value>0x02</value>
                <name>Toggle</name>
            </single>
            <default>
                <name>Reserved</name>
            </default>
        </values>
    </fragment>
                                              620,21        46%
```

Figure 12   ZCL Definition in XML

A corresponding XML file then includes the definition of the Home Automation application profile (or a new application profile), as shown in Figure 13. Here, we observe, in the top red rectangle, the definition of the application profile, and in the bottom red rectangle, the assignment of the On/Off cluster from the ZCL to cluster ID 0x04.


Protocol_ZigBee_HA.xml ... Analyzer\Profiles) - GVIM
File Edit Tools Syntax Buffers Window Help
/OnOff 5,1

```
    <!-- Identification information. -->
    <identification>
        <name>Home Automation Profile</name>
        <description>Home Automation Profile</description>
        <reference>ZigBee Document 053520r09: Home Automation
Profile Specification</reference>
        <abbreviation>HA</abbreviation>
        <identifier>0x0104</identifier>
        <version>0.4</version>
    </identification>
    <!-- List of available clusters. -->
    <fragment name="Clusters">
        <values>
            <depends>
                <select name="ClusterId"></select>
            </depends>
            <single>
                <value>0x00</value>
                <select name="Cluster.Basic"></select>
            </single>
            <single>
                <value>0x01</value>
                <name>Reserved</name>
            </single>
            <single>
                <value>0x02</value>
                <name>Reserved</name>
            </single>
            <single>
                <value>0x03</value>
                <select name="Cluster.Identify"></select>
            </single>
            <single>
                <value>0x04</value>
                <select name="Cluster.OnOff"></select>
            </single>
```

Figure 13 HA Definition in XML

Proprietary vs. Standards-Based Application Profiles and Implications on Certification

Another design consideration is the importance of standards-based application profiles, as distinct from a proprietary (private) application profiles.

Where interoperability with devices from other manufacturers is required, standards-based application profiles need to be defined. The ZigBee Alliance's Application Framework Group provides the forum to enable interested parties to work together to achieve this. Accompanying this effort is a compliance program (recognizing the product as a *ZigBee Certified Product*) that is facilitated by the ZigBee Alliance to ensure interoperability between all devices that adopt such a standards-based application profile. Refer to the ZigBee Alliance for current information and policy concerning compliance programs.

Selection and Deployment of An Appropriate Commissioning Model

Some applications have minimal requirements for commissioning. Others have more complex requirements to fit into existing commissioning workflows. Commissioning incorporates both the device admission process (which includes security considerations and troubleshooting requirements, but was not discussed in this document), and includes selection of the binding management model, which was discussed earlier in this document.

Concluding Remarks

This paper has provided an introductory description of the ZigBee application framework. The applications framework will continue to evolve, in particular to encompass an ever expanding set of potential applications through new standard and private application profiles, and procedures and facilities for deploying and managing those applications.

About Daintree Networks

Daintree Networks is a leading provider of design verification tools for emerging wireless sensor and control networks and associated devices. Our products are essential tools used to expedite the development, certification and deployment of wireless sensor network devices and applications. Daintree's Sensor Network Analyzer, provides visualization, measurements and packet decodes for IEEE 802.15.4™ and ZigBee™ wireless communications. Figures in this white paper are screen shots taken from actual live networks. For more information on this and our other products, visit us at www.daintree.net, or email us at sales@daintree.net.

May 2006

Daintree Networks, Inc.
111 N. Market Street, Suite 648
San Jose, CA 95113 U.S.A.
www.daintree.net email: sales@daintree.net
+1 (408) 351-3645

32-megabit 2.7-volt DataFlash®

AT45DB321D

Preliminary

Features

- Single 2.7V - 3.6V Supply
- RapidS™ Serial Interface: 66 MHz Maximum Clock Frequency
  - SPI Compatible Modes 0 and 3
- User Configurable Page Size
  - 512 Bytes per Page
  - 528 Bytes per Page
- Page Program Operation
  - Intelligent Programming Operation
  - 8,192 Pages (512/528 Bytes/Page) Main Memory
- Flexible Erase Options
  - Page Erase (512 Bytes)
  - Block Erase (4 Kbytes)
  - Sector Erase (64 Kbytes)
  - Chip Erase (32 Mbits)
- Two SRAM Data Buffers (512/528 Bytes)
  - Allows Receiving of Data while Reprogramming the Flash Array
- Continuous Read Capability through Entire Array
  - Ideal for Code Shadowing Applications
- Low-power Dissipation
  - 7 mA Active Read Current Typical
  - 25 µA Standby Current Typical
  - 5 µA Deep Power Down Typical
- Hardware and Software Data Protection Features
  - Individual Sector
- Sector Lockdown for Secure Code and Data Storage
  - Individual Sector
- Security: 128-byte Security Register
  - 64-byte User Programmable Space
  - Unique 64-byte Device Identifier
- JEDEC Standard Manufacturer and Device ID Read
- 100,000 Program/Erase Cycles Per Page Minimum
- Data Retention – 20 Years
- Industrial Temperature Range
- Green (Pb/Halide-free/RoHS Compliant) Packaging Options

1. Description

The AT45DB321D is a 2.7-volt, serial-interface sequential access Flash memory ideally suited for a wide variety of digital voice-, image-, program code- and data-storage applications. The AT45DB321D supports RapidS serial interface for applications requiring very high speed operations. RapidS serial interface is SPI compatible for frequencies up to 66 MHz. Its 34,603,008 bits of memory are organized as 8,192 pages of 512 bytes or 528 bytes each. In addition to the main memory, the AT45DB321D also contains two SRAM buffers of 512/528 bytes each. The buffers allow the receiving of data while a page in the main Memory is being reprogrammed, as well as writing a continuous data stream. EEPROM emulation (bit or byte alterability) is easily handled with a self-contained three step read-modify-write operation. Unlike conventional Flash memories that are accessed randomly with multiple address lines and a parallel interface, the DataFlash uses a RapidS serial interface to sequentially access its data. The simple sequential access dramatically reduces active pin count, facilitates hardware layout, increases system reliability, minimizes switching noise, and reduces package size. The device is optimized for use in many commercial and industrial applications where high-density, low-pin count, low-voltage and low-power are essential.

To allow for simple in-system reprogrammability, the AT45DB321D does not require high input voltages for programming. The device operates from a single power supply, 2.7V to 3.6V, for both the program and read operations. The AT45DB321D is enabled through the chip select pin ($\overline{CS}$) and accessed via a three-wire interface consisting of the Serial Input (SI), Serial Output (SO), and the Serial Clock (SCK).

All programming and erase cycles are self-timed.

2. Pin Configurations and Pinouts

Figure 2-1. MLF and CASON Top View through Package

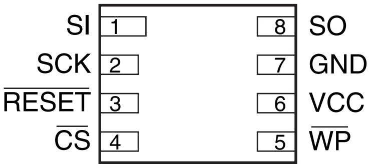


Figure 2-2. SOIC Top View

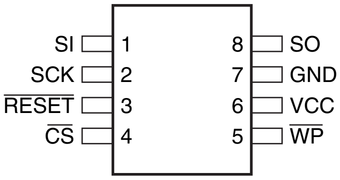
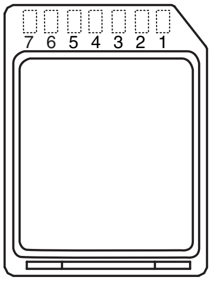
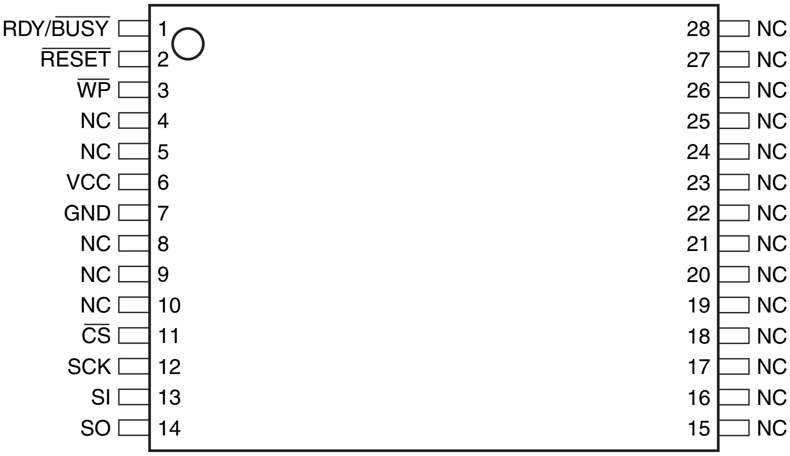


Figure 2-3. DataFlash Card[(1)] Top View through Package



Note: 1. See AT45DCB004D Datasheet.

Figure 2-4. TSOP Top View: Type 1

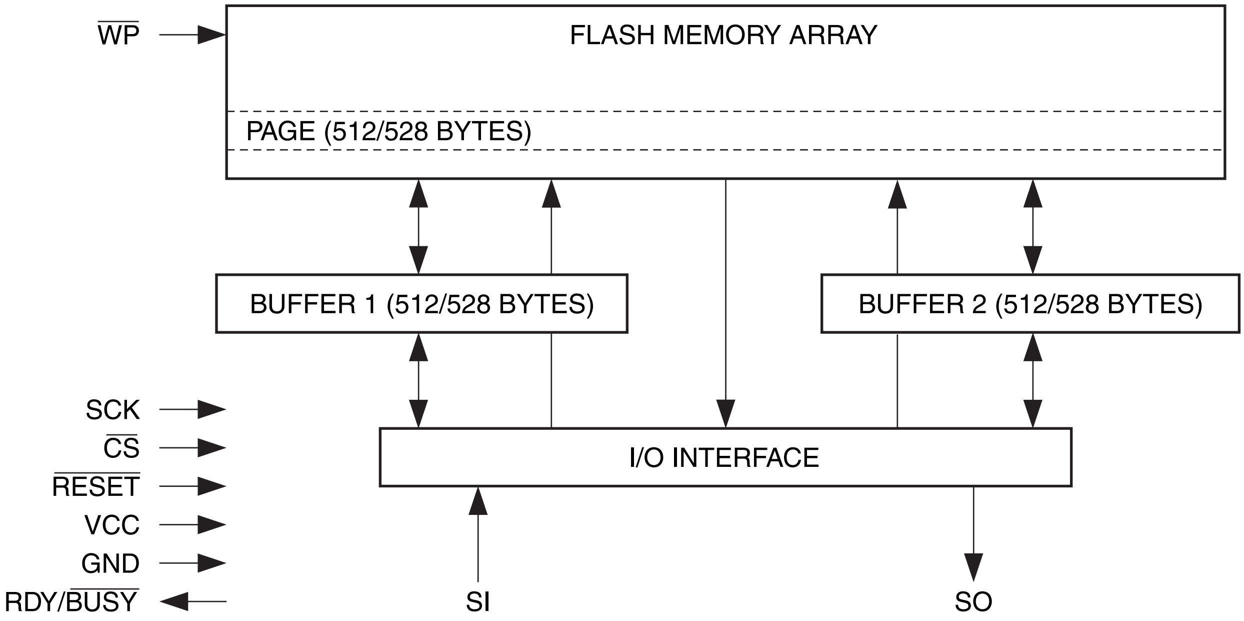


Note: TSOP package is not recommended for new designs. Future die shrinks will support 8-pin packages only.

Table 2-1. Pin Configurations

Symbol	Name and Function	Asserted State	Type
$\overline{\text{CS}}$	**Chip Select:** Asserting the $\overline{\text{CS}}$ pin selects the device. When the $\overline{\text{CS}}$ pin is deasserted, the device will be deselected and normally be placed in the standby mode (not Deep Power-Down mode), and the output pin (SO) will be in a high-impedance state. When the device is deselected, data will not be accepted on the input pin (SI). A high-to-low transition on the $\overline{\text{CS}}$ pin is required to start an operation, and a low-to-high transition is required to end an operation. When ending an internally self-timed operation such as a program or erase cycle, the device will not enter the standby mode until the completion of the operation.	Low	Input
SCK	**Serial Clock:** This pin is used to provide a clock to the device and is used to control the flow of data to and from the device. Command, address, and input data present on the SI pin is always latched on the rising edge of SCK, while output data on the SO pin is always clocked out on the falling edge of SCK.	–	Input
SI	**Serial Input:** The SI pin is used to shift data into the device. The SI pin is used for all data input including command and address sequences. Data on the SI pin is always latched on the rising edge of SCK.	–	Input
SO	**Serial Output:** The SO pin is used to shift data out from the device. Data on the SO pin is always clocked out on the falling edge of SCK.	–	Output
$\overline{\text{WP}}$	**Write Protect:** When the $\overline{\text{WP}}$ pin is asserted, all sectors specified for protection by the Sector Protection Register will be protected against program and erase operations regardless of whether the Enable Sector Protection command has been issued or not. The $\overline{\text{WP}}$ pin functions independently of the software controlled protection method. After the $\overline{\text{WP}}$ pin goes low, the content of the Sector Protection Register cannot be modified. If a program or erase command is issued to the device while the $\overline{\text{WP}}$ pin is asserted, the device will simply ignore the command and perform no operation. The device will return to the idle state once the $\overline{\text{CS}}$ pin has been deasserted. The Enable Sector Protection command and Sector Lockdown command, however, will be recognized by the device when the $\overline{\text{WP}}$ pin is asserted. The $\overline{\text{WP}}$ pin is internally pulled-high and may be left floating if hardware controlled protection will not be used. However, it is recommended that the $\overline{\text{WP}}$ pin also be externally connected to V_{CC} whenever possible.	Low	Input
$\overline{\text{RESET}}$	**Reset:** A low state on the reset pin ($\overline{\text{RESET}}$) will terminate the operation in progress and reset the internal state machine to an idle state. The device will remain in the reset condition as long as a low level is present on the $\overline{\text{RESET}}$ pin. Normal operation can resume once the $\overline{\text{RESET}}$ pin is brought back to a high level. The device incorporates an internal power-on reset circuit, so there are no restrictions on the $\overline{\text{RESET}}$ pin during power-on sequences. If this pin and feature are not utilized it is recommended that the $\overline{\text{RESET}}$ pin be driven high externally.	Low	Input
RDY/$\overline{\text{BUSY}}$	**Ready/Busy:** This open drain output pin will be driven low when the device is busy in an internally self-timed operation. This pin, which is normally in a high state (through an external pull-up resistor), will be pulled low during programming/erase operations, compare operations, and page-to-buffer transfers. The busy status indicates that the Flash memory array and one of the buffers cannot be accessed; read and write operations to the other buffer can still be performed.	–	Output
V_{CC}	**Device Power Supply:** The V_{CC} pin is used to supply the source voltage to the device. Operations at invalid V_{CC} voltages may produce spurious results and should not be attempted.	–	Power
GND	**Ground:** The ground reference for the power supply. GND should be connected to the system ground.	–	Ground

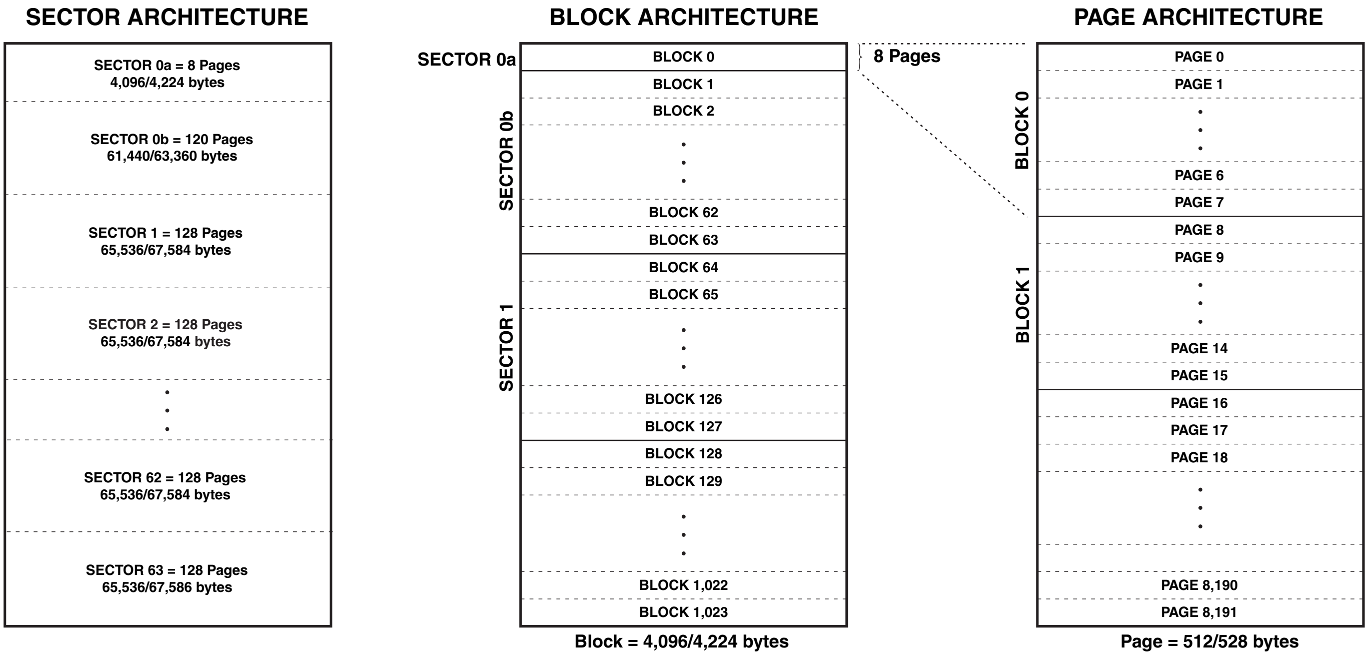
3. Block Diagram



4. Memory Array

To provide optimal flexibility, the memory array of the AT45DB321D is divided into three levels of granularity comprising of sectors, blocks, and pages. The “Memory Architecture Diagram” illustrates the breakdown of each level and details the number of pages per sector and block. All program operations to the DataFlash occur on a page by page basis. The erase operations can be performed at the chip, sector, block or page level.

Figure 4-1. Memory Architecture Diagram



5. Device Operation

The device operation is controlled by instructions from the host processor. The list of instructions and their associated opcodes are contained in Table 15-1 on page 28 through Table 15-7 on page 31. A valid instruction starts with the falling edge of $\overline{CS}$ followed by the appropriate 8-bit opcode and the desired buffer or main memory address location. While the $\overline{CS}$ pin is low, toggling the SCK pin controls the loading of the opcode and the desired buffer or main memory address location through the SI (serial input) pin. All instructions, addresses, and data are transferred with the most significant bit (MSB) first.

Buffer addressing for the DataFlash standard page size (528 bytes) is referenced in the datasheet using the terminology BFA9 - BFA0 to denote the 10 address bits required to designate a byte address within a buffer. Main memory addressing is referenced using the terminology PA12 - PA0 and BA9 - BA0, where PA12 - PA0 denotes the 13 address bits required to designate a page address and BA9 - BA0 denotes the 10 address bits required to designate a byte address within the page.

For “Power of 2” binary page size (512 bytes) the Buffer addressing is referenced in the datasheet using the conventional terminology BFA8 - BFA0 to denote the 9 address bits required to designate a byte address within a buffer. Main memory addressing is referenced using the terminology A21 - A0, where A21 - A9 denotes the 13 address bits required to designate a page address and A8 - A0 denotes the 9 address bits required to designate a byte address within a page.

6. Read Commands

By specifying the appropriate opcode, data can be read from the main memory or from either one of the two SRAM data buffers. The DataFlash supports RapidS protocols for Mode 0 and Mode 3. Please refer to the “Detailed Bit-level Read Timing” diagrams in this datasheet for details on the clock cycle sequences for each mode.

6.1 Continuous Array Read (Legacy Command: E8H): Up to 66 MHz

By supplying an initial starting address for the main memory array, the Continuous Array Read command can be utilized to sequentially read a continuous stream of data from the device by simply providing a clock signal; no additional addressing information or control signals need to be provided. The DataFlash incorporates an internal address counter that will automatically increment on every clock cycle, allowing one continuous read operation without the need of additional address sequences. To perform a continuous read from the DataFlash standard page size (528 bytes), an opcode of E8H must be clocked into the device followed by three address bytes (which comprise the 24-bit page and byte address sequence) and 4 don’t care bytes. The first 13 bits (PA12 - PA0) of the 23-bit address sequence specify which page of the main memory array to read, and the last 10 bits (BA9 - BA0) of the 23-bit address sequence specify the starting byte address within the page. To perform a continuous read from the binary page size (512 bytes), the opcode (E8H) must be clocked into the device followed by three address bytes and 4 don’t care bytes. The first 13 bits (A21 - A9) of the 22-bits sequence specify which page of the main memory array to read, and the last 9 bits (A8 - A0) of the 22-bits address sequence specify the starting byte address within the page. The don’t care bytes that follow the address bytes are needed to initialize the read operation. Following the don’t care bytes, additional clock pulses on the SCK pin will result in data being output on the SO (serial output) pin.

The $\overline{CS}$ pin must remain low during the loading of the opcode, the address bytes, the don’t care bytes, and the reading of data. When the end of a page in main memory is reached during a

Continuous Array Read, the device will continue reading at the beginning of the next page with no delays incurred during the page boundary crossover (the crossover from the end of one page to the beginning of the next page). When the last bit in the main memory array has been read, the device will continue reading back at the beginning of the first page of memory. As with crossing over page boundaries, no delays will be incurred when wrapping around from the end of the array to the beginning of the array.

A low-to-high transition on the $\overline{\text{CS}}$ pin will terminate the read operation and tri-state the output pin (SO). The maximum SCK frequency allowable for the Continuous Array Read is defined by the f_{CAR1} specification. The Continuous Array Read bypasses both data buffers and leaves the contents of the buffers unchanged.

6.2 Continuous Array Read (High Frequency Mode: 0BH): Up to 66 MHz

This command can be used with the serial interface to read the main memory array sequentially in high speed mode for any clock frequency up to the maximum specified by f_{CAR1}. To perform a continuous read array with the page size set to 528 bytes, the $\overline{\text{CS}}$ must first be asserted then an opcode 0BH must be clocked into the device followed by three address bytes and a dummy byte. The first 13 bits (PA12 - PA0) of the 23-bit address sequence specify which page of the main memory array to read, and the last 10 bits (BA9 - BA0) of the 23-bit address sequence specify the starting byte address within the page. To perform a continuous read with the page size set to 512 bytes, the opcode, 0BH, must be clocked into the device followed by three address bytes (A21 - A0) and a dummy byte. Following the dummy byte, additional clock pulses on the SCK pin will result in data being output on the SO (serial output) pin.

The CS pin must remain low during the loading of the opcode, the address bytes, and the reading of data. When the end of a page in the main memory is reached during a Continuous Array Read, the device will continue reading at the beginning of the next page with no delays incurred during the page boundary crossover (the crossover from the end of one page to the beginning of the next page). When the last bit in the main memory array has been read, the device will continue reading back at the beginning of the first page of memory. As with crossing over page boundaries, no delays will be incurred when wrapping around from the end of the array to the beginning of the array. A low-to-high transition on the CS pin will terminate the read operation and tri-state the output pin (SO). The maximum SCK frequency allowable for the Continuous Array Read is defined by the f_{CAR1} specification. The Continuous Array Read bypasses both data buffers and leaves the contents of the buffers unchanged.

6.3 Continuous Array Read (Low Frequency Mode: 03H): Up to 33 MHz

This command can be used with the serial interface to read the main memory array sequentially without a dummy byte up to maximum frequencies specified by f_{CAR2}. To perform a continuous read array with the page size set to 528 bytes, the $\overline{\text{CS}}$ must first be asserted then an opcode, 03H, must be clocked into the device followed by three address bytes (which comprise the 24-bit page and byte address sequence). The first 13 bits (PA12 - PA0) of the 23-bit address sequence specify which page of the main memory array to read, and the last 10 bits (BA9 - BA0) of the 23-bit address sequence specify the starting byte address within the page. To perform a continuous read with the page size set to 512 bytes, the opcode, 03H, must be clocked into the device followed by three address bytes (A21 - A0). Following the address bytes, additional clock pulses on the SCK pin will result in data being output on the SO (serial output) pin.

The CS pin must remain low during the loading of the opcode, the address bytes, and the reading of data. When the end of a page in the main memory is reached during a Continuous Array Read, the device will continue reading at the beginning of the next page with no delays incurred during the page boundary crossover (the crossover from the end of one page to the beginning of the next page). When the last bit in the main memory array has been read, the device will continue reading back at the beginning of the first page of memory. As with crossing over page boundaries, no delays will be incurred when wrapping around from the end of the array to the beginning of the array. A low-to-high transition on the CS pin will terminate the read operation and tri-state the output pin (SO). The Continuous Array Read bypasses both data buffers and leaves the contents of the buffers unchanged.

6.4 Main Memory Page Read

A main memory page read allows the user to read data directly from any one of the 8,192 pages in the main memory, bypassing both of the data buffers and leaving the contents of the buffers unchanged. To start a page read from the DataFlash standard page size (528 bytes), an opcode of D2H must be clocked into the device followed by three address bytes (which comprise the 24-bit page and byte address sequence) and 4 don't care bytes. The first 13 bits (PA12 - PA0) of the 23-bit address sequence specify the page in main memory to be read, and the last 10 bits (BA9 - BA0) of the 23-bit address sequence specify the starting byte address within that page. To start a page read from the binary page size (512 bytes), the opcode D2H must be clocked into the device followed by three address bytes and 4 don't care bytes. The first 13 bits (A21 - A9) of the 22-bits sequence specify which page of the main memory array to read, and the last 9 bits (A8 - A0) of the 22-bits address sequence specify the starting byte address within the page. The don't care bytes that follow the address bytes are sent to initialize the read operation. Following the don't care bytes, additional pulses on SCK result in data being output on the SO (serial output) pin. The $\overline{CS}$ pin must remain low during the loading of the opcode, the address bytes, the don't care bytes, and the reading of data. When the end of a page in main memory is reached, the device will continue reading back at the beginning of the same page. A low-to-high transition on the $\overline{CS}$ pin will terminate the read operation and tri-state the output pin (SO). The maximum SCK frequency allowable for the Main Memory Page Read is defined by the f_{SCK} specification. The Main Memory Page Read bypasses both data buffers and leaves the contents of the buffers unchanged.

6.5 Buffer Read

The SRAM data buffers can be accessed independently from the main memory array, and utilizing the Buffer Read Command allows data to be sequentially read directly from the buffers. Four opcodes, D4H or D1H for buffer 1 and D6H or D3H for buffer 2 can be used for the Buffer Read Command. The use of each opcode depends on the maximum SCK frequency that will be used to read data from the buffer. The D4H and D6H opcode can be used at any SCK frequency up to the maximum specified by f_{CAR1}. The D1H and D3H opcode can be used for lower frequency read operations up to the maximum specified by f_{CAR2}.

To perform a buffer read from the DataFlash standard buffer (528 bytes), the opcode must be clocked into the device followed by three address bytes comprised of 14 don't care bits and 10 buffer address bits (BFA9 - BFA0). To perform a buffer read from the binary buffer (512 bytes), the opcode must be clocked into the device followed by three address bytes comprised of 15 don't care bits and 9 buffer address bits (BFA8 - BFA0). Following the address bytes, one don't care byte must be clocked in to initialize the read operation. The $\overline{CS}$ pin must remain low during the loading of the opcode, the address bytes, the don't care byte, and the reading of data. When the end of a buffer is reached, the device will continue reading back at the beginning of the buffer. A low-to-high transition on the $\overline{CS}$ pin will terminate the read operation and tri-state the output pin (SO).

7. Program and Erase Commands

7.1 Buffer Write

Data can be clocked in from the input pin (SI) into either buffer 1 or buffer 2. To load data into the DataFlash standard buffer (528 bytes), a 1-byte opcode, 84H for buffer 1 or 87H for buffer 2, must be clocked into the device, followed by three address bytes comprised of 14 don't care bits and 10 buffer address bits (BFA9 - BFA0). The 10 buffer address bits specify the first byte in the buffer to be written. To load data into the binary buffers (512 bytes each), a 1-byte opcode 84H for buffer 1 or 87H for buffer 2, must be clocked into the device, followed by three address bytes comprised of 15 don't care bits and 9 buffer address bits (BFA8 - BFA0). The 9 buffer address bits specify the first byte in the buffer to be written. After the last address byte has been clocked into the device, data can then be clocked in on subsequent clock cycles. If the end of the data buffer is reached, the device will wrap around back to the beginning of the buffer. Data will continue to be loaded into the buffer until a low-to-high transition is detected on the $\overline{\text{CS}}$ pin.

7.2 Buffer to Main Memory Page Program with Built-in Erase

Data written into either buffer 1 or buffer 2 can be programmed into the main memory. A 1-byte opcode, 83H for buffer 1 or 86H for buffer 2, must be clocked into the device. For the DataFlash standard page size (528 bytes), the opcode must be followed by three address bytes consist of 1 don't care bit, 13 page address bits (PA12 - PA0) that specify the page in the main memory to be written and 10 don't care bits. To perform a buffer to main memory page program with built-in erase for the binary page size (512 bytes), the opcode 83H for buffer 1 or 86H for buffer 2, must be clocked into the device followed by three address bytes consisting of 2 don't care bits 13-page address bits (A21 - A9) that specify the page in the main memory to be written and 9 don't care bits. When a low-to-high transition occurs on the $\overline{\text{CS}}$ pin, the part will first erase the selected page in main memory (the erased state is a logic 1) and then program the data stored in the buffer into the specified page in main memory. Both the erase and the programming of the page are internally self-timed and should take place in a maximum time of t_{EP}. During this time, the status register and the RDY/$\overline{\text{BUSY}}$ pin will indicate that the part is busy.

7.3 Buffer to Main Memory Page Program without Built-in Erase

A previously-erased page within main memory can be programmed with the contents of either buffer 1 or buffer 2. A 1-byte opcode, 88H for buffer 1 or 89H for buffer 2, must be clocked into the device. For the DataFlash standard page size (528 bytes), the opcode must be followed by three address bytes consist of 1 don't care bit, 13 page address bits (PA12 - PA0) that specify the page in the main memory to be written and 10 don't care bits. To perform a buffer to main memory page program without built-in erase for the binary page size (512 bytes), the opcode 88H for buffer 1 or 89H for buffer 2, must be clocked into the device followed by three address bytes consisting of 2 don't care bits, 13 page address bits (A21 - A9) that specify the page in the main memory to be written and 9 don't care bits. When a low-to-high transition occurs on the $\overline{\text{CS}}$ pin, the part will program the data stored in the buffer into the specified page in the main memory. It is necessary that the page in main memory that is being programmed has been previously erased using one of the erase commands (Page Erase or Block Erase). The programming of the page is internally self-timed and should take place in a maximum time of t_P. During this time, the status register and the RDY/$\overline{\text{BUSY}}$ pin will indicate that the part is busy.

7.4 Page Erase

The Page Erase command can be used to individually erase any page in the main memory array allowing the Buffer to Main Memory Page Program to be utilized at a later time. To perform a page erase in the DataFlash standard page size (528 bytes), an opcode of 81H must be loaded into the device, followed by three address bytes comprised of 1 don't care bit, 13 page address bits (PA12 - PA0) that specify the page in the main memory to be erased and 10 don't care bits. To perform a page erase in the binary page size (512 bytes), the opcode 81H must be loaded into the device, followed by three address bytes consist of 2 don't care bits, 13 page address bits (A21 - A9) that specify the page in the main memory to be erased and 9 don't care bits. When a low-to-high transition occurs on the $\overline{\text{CS}}$ pin, the part will erase the selected page (the erased state is a logical 1). The erase operation is internally self-timed and should take place in a maximum time of t_{PE}. During this time, the status register and the RDY/$\overline{\text{BUSY}}$ pin will indicate that the part is busy.

7.5 Block Erase

A block of eight pages can be erased at one time. This command is useful when large amounts of data has to be written into the device. This will avoid using multiple Page Erase Commands. To perform a block erase for the DataFlash standard page size (528 bytes), an opcode of 50H must be loaded into the device, followed by three address bytes comprised of 1 don't care bit, 10 page address bits (PA12 -PA3) and 13 don't care bits. The 10 page address bits are used to specify which block of eight pages is to be erased. To perform a block erase for the binary page size (512 bytes), the opcode 50H must be loaded into the device, followed by three address bytes consisting of 2 don't care bits, 10 page address bits (A21 - A12) and 12 don't care bits. The 10 page address bits are used to specify which block of eight pages is to be erased. When a low-to-high transition occurs on the $\overline{\text{CS}}$ pin, the part will erase the selected block of eight pages. The erase operation is internally self-timed and should take place in a maximum time of t_{BE}. During this time, the status register and the RDY/$\overline{\text{BUSY}}$ pin will indicate that the part is busy.

Table 7-1. Block Erase Addressing

PA12/ A21	PA11/ A20	PA10/ A19	PA9/ A18	PA8/ A17	PA7/ A16	PA6/ A15	PA5/ A14	PA4/ A13	PA3/ A12	PA2/ A11	PA1/ A10	PA0/ A9	Block
0	0	0	0	0	0	0	0	0	0	X	X	X	0
0	0	0	0	0	0	0	0	0	1	X	X	X	1
0	0	0	0	0	0	0	0	1	0	X	X	X	2
0	0	0	0	0	0	0	0	1	1	X	X	X	3
•	•	•	•	•	•	•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•	•	•	•	•	•	•
1	1	1	1	1	1	1	1	0	0	X	X	X	1020
1	1	1	1	1	1	1	1	0	1	X	X	X	1021
1	1	1	1	1	1	1	1	1	0	X	X	X	1022
1	1	1	1	1	1	1	1	1	1	X	X	X	1023

7.6 Sector Erase

The Sector Erase command can be used to individually erase any sector in the main memory. There are 64 sectors and only one sector can be erased at one time. To perform sector 0a or sector 0b erase for the DataFlash standard page size (528 bytes), an opcode of 7CH must be loaded into the device, followed by three address bytes comprised of 1 don't care bit, 10 page address bits (PA12 - PA3) and 13 don't care bits. To perform a sector 1-63 erase, the opcode 7CH must be loaded into the device, followed by three address bytes comprised of 1 don't care bit, 4 page address bits (PA12 - PA9) and 19 don't care bits. To perform sector 0a or sector 0b erase for the binary page size (512 bytes), an opcode of 7CH must be loaded into the device, followed by three address bytes comprised of 2 don't care bit and 10 page address bits (A21 - A12) and 12 don't care bits. To perform a sector 1-63 erase, the opcode 7CH must be loaded into the device, followed by three address bytes comprised of 2 don't care bits and 4 page address bits (A21 - A18) and 18 don't care bits. The page address bits are used to specify any valid address location within the sector which is to be erased. When a low-to-high transition occurs on the $\overline{CS}$ pin, the part will erase the selected sector. The erase operation is internally self-timed and should take place in a maximum time of t_{SE}. During this time, the status register and the RDY/$\overline{BUSY}$ pin will indicate that the part is busy.

Table 7-2. Sector Erase Addressing

PA12/ A21	PA11/ A20	PA10/ A19	PA9/ A18	PA8/ A17	PA7/ A16	PA6/ A15	PA5/ A14	PA4/ A13	PA3/ A12	PA2/ A11	PA1/ A10	PA0/ A9	Sector
0	0	0	0	0	0	0	0	0	0	X	X	X	0a
0	0	0	0	0	0	0	0	0	1	X	X	X	0b
0	0	0	0	0	1	X	X	X	X	X	X	X	1
0	0	0	0	1	0	X	X	X	X	X	X	X	2
•	•	•	•	•	•	•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•	•	•	•	•	•	•
1	1	1	1	0	0	X	X	X	X	X	X	X	60
1	1	1	1	0	1	X	X	X	X	X	X	X	61
1	1	1	1	1	0	X	X	X	X	X	X	X	62
1	1	1	1	1	1	X	X	X	X	X	X	X	63

7.7 Chip Erase[(1)]

The entire main memory can be erased at one time by using the Chip Erase command.

To execute the Chip Erase command, a 4-byte command sequence C7H, 94H, 80H and 9AH must be clocked into the device. Since the entire memory array is to be erased, no address bytes need to be clocked into the device, and any data clocked in after the opcode will be ignored. After the last bit of the opcode sequence has been clocked in, the $\overline{CS}$ pin can be deasserted to start the erase process. The erase operation is internally self-timed and should take place in a time of t_{CE}. During this time, the Status Register will indicate that the device is busy.

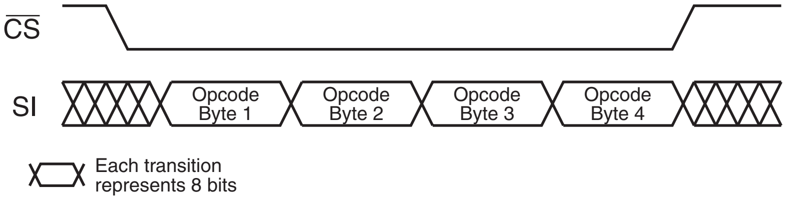
The Chip Erase command will not affect sectors that are protected or locked down; the contents of those sectors will remain unchanged. Only those sectors that are not protected or locked down will be erased.

Note: 1. Refer to the errata regarding Chip Erase on page 53.

The WP pin can be asserted while the device is erasing, but protection will not be activated until the internal erase cycle completes.

Command	Byte 1	Byte 2	Byte 3	Byte 4
Chip Erase	C7H	94H	80H	9AH

Figure 7-1. Chip Erase



Note: 1. Refer to the errata regarding Chip Erase on page 53.

7.8 Main Memory Page Program Through Buffer

This operation is a combination of the Buffer Write and Buffer to Main Memory Page Program with Built-in Erase operations. Data is first clocked into buffer 1 or buffer 2 from the input pin (SI) and then programmed into a specified page in the main memory. To perform a main memory page program through buffer for the DataFlash standard page size (528 bytes), a 1-byte opcode, 82H for buffer 1 or 85H for buffer 2, must first be clocked into the device, followed by three address bytes. The address bytes are comprised of 1 don't care bit, 13 page address bits, (PA12 - PA0) that select the page in the main memory where data is to be written, and 10 buffer address bits (BFA9 - BFA0) that select the first byte in the buffer to be written. To perform a main memory page program through buffer for the binary page size (512 bytes), the opcode 82H for buffer 1 or 85H for buffer 2, must be clocked into the device followed by three address bytes consisting of 2 don't care bits, 13 page address bits (A21 - A9) that specify the page in the main memory to be written, and 9 buffer address bits (BFA8 - BFA0) that selects the first byte in the buffer to be written. After all address bytes are clocked in, the part will take data from the input pins and store it in the specified data buffer. If the end of the buffer is reached, the device will wrap around back to the beginning of the buffer. When there is a low-to-high transition on the CS pin, the part will first erase the selected page in main memory to all 1s and then program the data stored in the buffer into that memory page. Both the erase and the programming of the page are internally self-timed and should take place in a maximum time of t_{EP}. During this time, the status register and the RDY/BUSY pin will indicate that the part is busy.

8. Sector Protection

Two protection methods, hardware and software controlled, are provided for protection against inadvertent or erroneous program and erase cycles. The software controlled method relies on the use of software commands to enable and disable sector protection while the hardware controlled method employs the use of the Write Protect (WP) pin. The selection of which sectors that are to be protected or unprotected against program and erase operations is specified in the nonvolatile Sector Protection Register. The status of whether or not sector protection has been enabled or disabled by either the software or the hardware controlled methods can be determined by checking the Status Register.

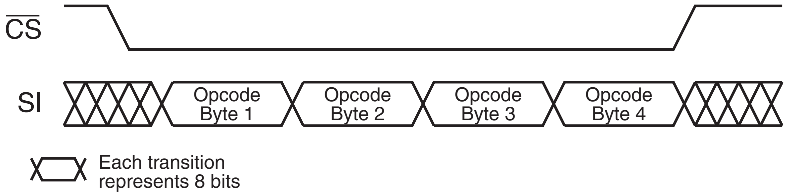
8.1 Software Sector Protection

8.1.1 Enable Sector Protection Command

Sectors specified for protection in the Sector Protection Register can be protected from program and erase operations by issuing the Enable Sector Protection command. To enable the sector protection using the software controlled method, the $\overline{CS}$ pin must first be asserted as it would be with any other command. Once the $\overline{CS}$ pin has been asserted, the appropriate 4-byte command sequence must be clocked in via the input pin (SI). After the last bit of the command sequence has been clocked in, the $\overline{CS}$ pin must be deasserted after which the sector protection will be enabled.

Command	Byte 1	Byte 2	Byte 3	Byte 4
Enable Sector Protection	3DH	2AH	7FH	A9H

Figure 8-1. Enable Sector Protection

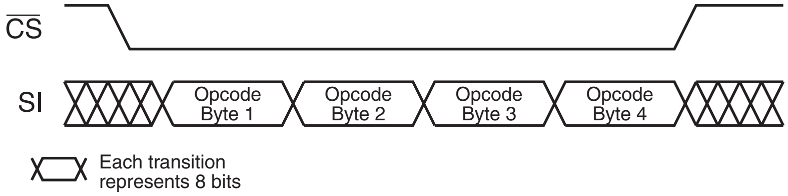


8.1.2 Disable Sector Protection Command

To disable the sector protection using the software controlled method, the $\overline{CS}$ pin must first be asserted as it would be with any other command. Once the $\overline{CS}$ pin has been asserted, the appropriate 4-byte sequence for the Disable Sector Protection command must be clocked in via the input pin (SI). After the last bit of the command sequence has been clocked in, the $\overline{CS}$ pin must be deasserted after which the sector protection will be disabled. The $\overline{WP}$ pin must be in the deasserted state; otherwise, the Disable Sector Protection command will be ignored.

Command	Byte 1	Byte 2	Byte 3	Byte 4
Disable Sector Protection	3DH	2AH	7FH	9AH

Figure 8-2. Disable Sector Protection



8.1.3 Various Aspects About Software Controlled Protection

Software controlled protection is useful in applications in which the $\overline{WP}$ pin is not or cannot be controlled by a host processor. In such instances, the $\overline{WP}$ pin may be left floating (the $\overline{WP}$ pin is internally pulled high) and sector protection can be controlled using the Enable Sector Protection and Disable Sector Protection commands.

If the device is powered cycled, then the software controlled protection will be disabled. Once the device is powered up, the Enable Sector Protection command should be reissued if sector protection is desired and if the $\overline{WP}$ pin is not used.

9. Hardware Controlled Protection

Sectors specified for protection in the Sector Protection Register and the Sector Protection Register itself can be protected from program and erase operations by asserting the $\overline{\text{WP}}$ pin and keeping the pin in its asserted state. The Sector Protection Register and any sector specified for protection cannot be erased or reprogrammed as long as the $\overline{\text{WP}}$ pin is asserted. In order to modify the Sector Protection Register, the $\overline{\text{WP}}$ pin must be deasserted. If the $\overline{\text{WP}}$ pin is permanently connected to GND, then the content of the Sector Protection Register cannot be changed. If the $\overline{\text{WP}}$ pin is deasserted, or permanently connected to V_{CC}, then the content of the Sector Protection Register can be modified.

The $\overline{\text{WP}}$ pin will override the software controlled protection method but only for protecting the sectors. For example, if the sectors were not previously protected by the Enable Sector Protection command, then simply asserting the $\overline{\text{WP}}$ pin would enable the sector protection within the maximum specified t_{WPE} time. When the $\overline{\text{WP}}$ pin is deasserted; however, the sector protection would no longer be enabled (after the maximum specified t_{WPD} time) as long as the Enable Sector Protection command was not issued while the $\overline{\text{WP}}$ pin was asserted. If the Enable Sector Protection command was issued before or while the $\overline{\text{WP}}$ pin was asserted, then simply deasserting the $\overline{\text{WP}}$ pin would not disable the sector protection. In this case, the Disable Sector Protection command would need to be issued while the $\overline{\text{WP}}$ pin is deasserted to disable the sector protection. The Disable Sector Protection command is also ignored whenever the $\overline{\text{WP}}$ pin is asserted.

A noise filter is incorporated to help protect against spurious noise that may inadvertently assert or deassert the $\overline{\text{WP}}$ pin.

The table below details the sector protection status for various scenarios of the $\overline{\text{WP}}$ pin, the Enable Sector Protection command, and the Disable Sector Protection command.

Figure 9-1. $\overline{\text{WP}}$ Pin and Protection Status

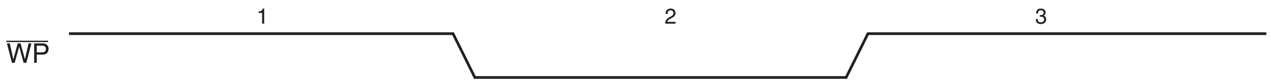


Table 9-1. $\overline{\text{WP}}$ Pin and Protection Status

Time Period	$\overline{\text{WP}}$ Pin	Enable Sector Protection Command	Disable Sector Protection Command	Sector Protection Status	Sector Protection Register
1	High	Command Not Issued Previously – Issue Command	X Issue Command –	Disabled Disabled Enabled	Read/Write Read/Write Read/Write
2	Low	X	X	Enabled	Read Only
3	High	Command Issued During Period 1 or 2 – Issue Command	Not Issued Yet Issue Command –	Enabled Disabled Enabled	Read/Write Read/Write Read/Write

9.1 Sector Protection Register

The nonvolatile Sector Protection Register specifies which sectors are to be protected or unprotected with either the software or hardware controlled protection methods. The Sector Protection Register contains 64 bytes of data, of which byte locations 0 through 63 contain values that specify whether sectors 0 through 63 will be protected or unprotected. The Sector Protection Register is user modifiable and must first be erased before it can be reprogrammed. Table 9-3 illustrates the format of the Sector Protection Register.:

Table 9-2. Sector Protection Register

Sector Number	0 (0a, 0b)	1 to 63
Protected	See Table 9-3	FFH
Unprotected		00H

Table 9-3. Sector 0 (0a, 0b)

	0a	0b			
	(Pages 0-7)	(Pages 8-127)			
	Bit 7, 6	Bit 5, 4	Bit 3, 2	Bit 1, 0	Data Value
Sectors 0a, 0b Unprotected	00	00	xx	xx	0xH
Protect Sector 0a (Pages 0-7)	11	00	xx	xx	CxH
Protect Sector 0b (Pages 8-127)	00	11	xx	xx	3xH
Protect Sectors 0a (Pages 0-7), 0b (Pages 8-127)[1]	11	11	xx	xx	FxH

Note: 1. The default value for bytes 0 through 63 when shipped from Atmel is 00H.
x = don't care.

9.1.1 Erase Sector Protection Register Command

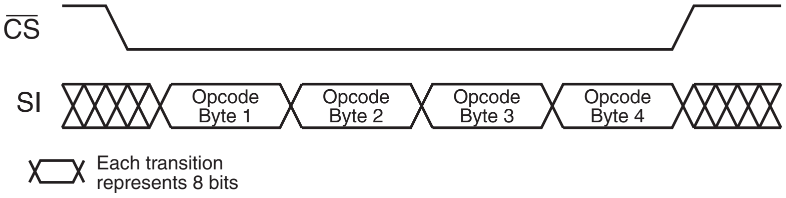
In order to modify and change the values of the Sector Protection Register, it must first be erased using the Erase Sector Protection Register command.

To erase the Sector Protection Register, the $\overline{CS}$ pin must first be asserted as it would be with any other command. Once the $\overline{CS}$ pin has been asserted, the appropriate 4-byte opcode sequence must be clocked into the device via the SI pin. The 4-byte opcode sequence must start with 3DH and be followed by 2AH, 7FH, and CFH. After the last bit of the opcode sequence has been clocked in, the $\overline{CS}$ pin must be deasserted to initiate the internally self-timed erase cycle. The erasing of the Sector Protection Register should take place in a time of t_{PE}, during which time the Status Register will indicate that the device is busy. If the device is powered-down before the completion of the erase cycle, then the contents of the Sector Protection Register cannot be guaranteed.

The Sector Protection Register can be erased with the sector protection enabled or disabled. Since the erased state (FFH) of each byte in the Sector Protection Register is used to indicate that a sector is specified for protection, leaving the sector protection enabled during the erasing of the register allows the protection scheme to be more effective in the prevention of accidental programming or erasing of the device. If for some reason an erroneous program or erase command is sent to the device immediately after erasing the Sector Protection Register and before the register can be reprogrammed, then the erroneous program or erase command will not be processed because all sectors would be protected.

Command	Byte 1	Byte 2	Byte 3	Byte 4
Erase Sector Protection Register	3DH	2AH	7FH	CFH

Figure 9-2. Erase Sector Protection Register



9.1.2 Program Sector Protection Register Command

Once the Sector Protection Register has been erased, it can be reprogrammed using the Program Sector Protection Register command.

To program the Sector Protection Register, the $\overline{CS}$ pin must first be asserted and the appropriate 4-byte opcode sequence must be clocked into the device via the SI pin. The 4-byte opcode sequence must start with 3DH and be followed by 2AH, 7FH, and FCH. After the last bit of the opcode sequence has been clocked into the device, the data for the contents of the Sector Protection Register must be clocked in. As described in Section 9.1, the Sector Protection Register contains 64 bytes of data, so 64 bytes must be clocked into the device. The first byte of data corresponds to sector 0, the second byte corresponds to sector 1, and so on with the last byte of data corresponding to sector 63.

After the last data byte has been clocked in, the $\overline{CS}$ pin must be deasserted to initiate the internally self-timed program cycle. The programming of the Sector Protection Register should take place in a time of t_P, during which time the Status Register will indicate that the device is busy. If the device is powered-down during the program cycle, then the contents of the Sector Protection Register cannot be guaranteed.

If the proper number of data bytes is not clocked in before the $\overline{CS}$ pin is deasserted, then the protection status of the sectors corresponding to the bytes not clocked in can not be guaranteed. For example, if only the first two bytes are clocked in instead of the complete 62 bytes, then the protection status of the last 62 sectors cannot be guaranteed. Furthermore, if more than 64 bytes of data is clocked into the device, then the data will wrap back around to the beginning of the register. For instance, if 65 bytes of data are clocked in, then the 65th byte will be stored at byte location 0 of the Sector Protection Register.

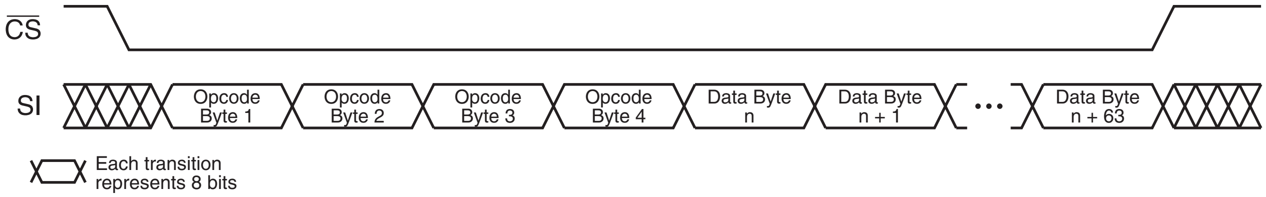
If a value other than 00H or FFH is clocked into a byte location of the Sector Protection Register, then the protection status of the sector corresponding to that byte location cannot be guaranteed. For example, if a value of 17H is clocked into byte location 2 of the Sector Protection Register, then the protection status of sector 2 cannot be guaranteed.

The Sector Protection Register can be reprogrammed while the sector protection enabled or disabled. Being able to reprogram the Sector Protection Register with the sector protection enabled allows the user to temporarily disable the sector protection to an individual sector rather than disabling sector protection completely.

The Program Sector Protection Register command utilizes the internal SRAM buffer 1 for processing. Therefore, the contents of the buffer 1 will be altered from its previous state when this command is issued.

Command	Byte 1	Byte 2	Byte 3	Byte 4
Program Sector Protection Register	3DH	2AH	7FH	FCH

Figure 9-3. Program Sector Protection Register



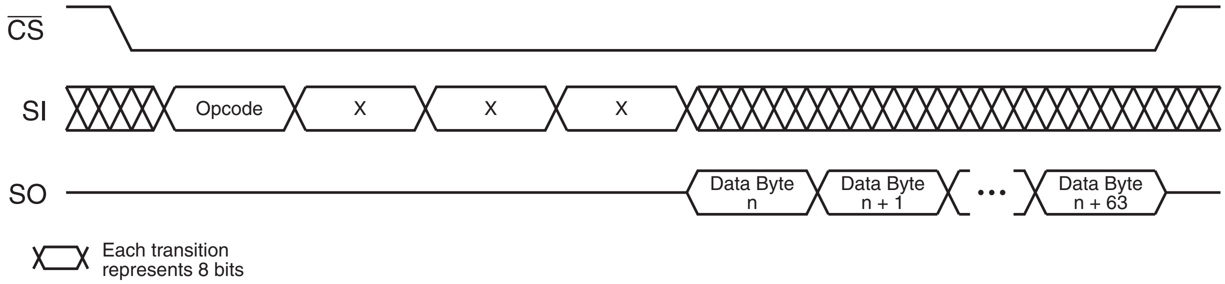
9.1.3 Read Sector Protection Register Command

To read the Sector Protection Register, the CS pin must first be asserted. Once the CS pin has been asserted, an opcode of 32H and 3 dummy bytes must be clocked in via the SI pin. After the last bit of the opcode and dummy bytes have been clocked in, any additional clock pulses on the SCK pins will result in data for the content of the Sector Protection Register being output on the SO pin. The first byte corresponds to sector 0 (0a, 0b), the second byte corresponds to sector 1 and the last byte (byte 64) corresponds to sector 63. Once the last byte of the Sector Protection Register has been clocked out, any additional clock pulses will result in undefined data being output on the SO pin. The CS must be deasserted to terminate the Read Sector Protection Register operation and put the output into a high-impedance state.

Command	Byte 1	Byte 2	Byte 3	Byte 4
Read Sector Protection Register	32H	xxH	xxH	xxH

Note: xx = Dummy Byte

Figure 9-4. Read Sector Protection Register



9.1.4 Various Aspects About the Sector Protection Register

The Sector Protection Register is subject to a limit of 10,000 erase/program cycles. Users are encouraged to carefully evaluate the number of times the Sector Protection Register will be modified during the course of the applications' life cycle. If the application requires that the Sector Protection Register be modified more than the specified limit of 10,000 cycles because the application needs to temporarily unprotect individual sectors (sector protection remains enabled while the Sector Protection Register is reprogrammed), then the application will need to limit this practice. Instead, a combination of temporarily unprotecting individual sectors along with disabling sector protection completely will need to be implemented by the application to ensure that the limit of 10,000 cycles is not exceeded.

10. Security Features

10.1 Sector Lockdown

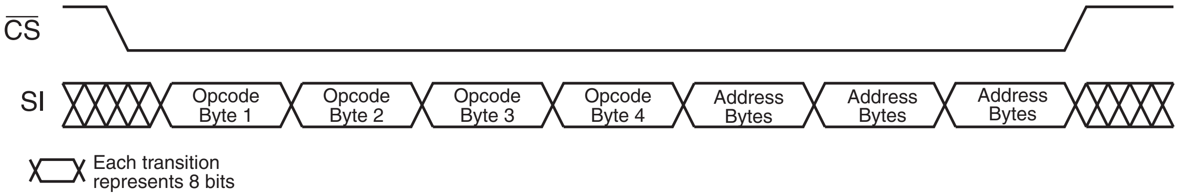
The device incorporates a Sector Lockdown mechanism that allows each individual sector to be permanently locked so that it becomes read only. This is useful for applications that require the ability to permanently protect a number of sectors against malicious attempts at altering program code or security information. **Once a sector is locked down, it can never be erased or programmed, and it can never be unlocked.**

To issue the Sector Lockdown command, the $\overline{CS}$ pin must first be asserted as it would be for any other command. Once the $\overline{CS}$ pin has been asserted, the appropriate 4-byte opcode sequence must be clocked into the device in the correct order. The 4-byte opcode sequence must start with 3DH and be followed by 2AH, 7FH, and 30H. After the last byte of the command sequence has been clocked in, then three address bytes specifying any address within the sector to be locked down must be clocked into the device. After the last address bit has been clocked in, the $\overline{CS}$ pin must then be deasserted to initiate the internally self-timed lockdown sequence.

The lockdown sequence should take place in a maximum time of t_P, during which time the Status Register will indicate that the device is busy. If the device is powered-down before the completion of the lockdown sequence, then the lockdown status of the sector cannot be guaranteed. In this case, it is recommended that the user read the Sector Lockdown Register to determine the status of the appropriate sector lockdown bits or bytes and reissue the Sector Lockdown command if necessary.

Command	Byte 1	Byte 2	Byte 3	Byte 4
Sector Lockdown	3DH	2AH	7FH	30H

Figure 10-1. Sector Lockdown



10.1.1 Sector Lockdown Register

Sector Lockdown Register is a nonvolatile register that contains 64 bytes of data, as shown below:

Sector Number	0 (0a, 0b)	1 to 63
Locked	See Below	FFH
Unlocked		00H

Table 10-1. Sector 0 (0a, 0b)

	0a	0b			
	(Pages 0-7)	(Pages 8-127)			
	Bit 7, 6	Bit 5, 4	Bit 3, 2	Bit 1, 0	Data Value
Sectors 0a, 0b Unlocked	00	00	00	00	00H
Sector 0a Locked (Pages 0-7)	11	00	00	00	C0H
Sector 0b Locked (Pages 8-127)	00	11	00	00	30H
Sectors 0a, 0b Locked (Pages 0-127)	11	11	00	00	F0H

10.1.2 Reading the Sector Lockdown Register

The Sector Lockdown Register can be read to determine which sectors in the memory array are permanently locked down. To read the Sector Lockdown Register, the $\overline{CS}$ pin must first be asserted. Once the $\overline{CS}$ pin has been asserted, an opcode of 35H and 3 dummy bytes must be clocked into the device via the SI pin. After the last bit of the opcode and dummy bytes have been clocked in, the data for the contents of the Sector Lockdown Register will be clocked out on the SO pin. The first byte corresponds to sector 0 (0a, 0b) the second byte corresponds to sector 1 and the last byte (byte 16) corresponds to sector 15. After the last byte of the Sector Lockdown Register has been read, additional pulses on the SCK pin will simply result in undefined data being output on the SO pin.

Deasserting the $\overline{CS}$ pin will terminate the Read Sector Lockdown Register operation and put the SO pin into a high-impedance state.

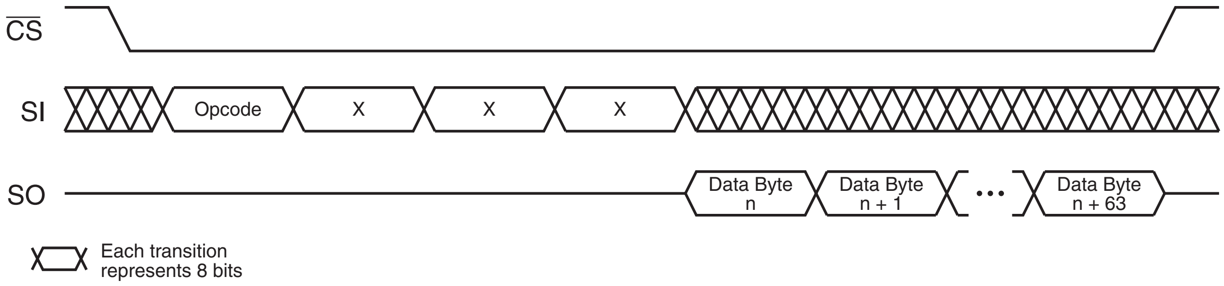
Table 10-2 details the values read from the Sector Lockdown Register.

Table 10-2. Sector Lockdown Register

Command	Byte 1	Byte 2	Byte 3	Byte 4
Read Sector Lockdown Register	35H	xxH	xxH	xxH

Note: xx = Dummy Byte

Figure 10-2. Read Sector Lockdown Register



10.2 Security Register

The device contains a specialized Security Register that can be used for purposes such as unique device serialization or locked key storage. The register is comprised of a total of 128 bytes that is divided into two portions. The first 64 bytes (byte locations 0 through 63) of the Security Register are allocated as a one-time user programmable space. Once these 64 bytes have been programmed, they cannot be reprogrammed. The remaining 64 bytes of the register (byte locations 64 through 127) are factory programmed by Atmel and will contain a unique value for each device. The factory programmed data is fixed and cannot be changed.

Table 10-3. Security Register

	Security Register Byte Number									
	0	**1**	**• • •**	**62**	**63**	**64**	**65**	**• • •**	**126**	**127**
Data Type	One-time User Programmable					Factory Programmed By Atmel				

10.2.1 Programming the Security Register

The user programmable portion of the Security Register does not need to be erased before it is programmed.

To program the Security Register, the $\overline{\text{CS}}$ pin must first be asserted and the appropriate 4-byte opcode sequence must be clocked into the device in the correct order. The 4-byte opcode sequence must start with 9BH and be followed by 00H, 00H, and 00H. After the last bit of the opcode sequence has been clocked into the device, the data for the contents of the 64-byte user programmable portion of the Security Register must be clocked in.

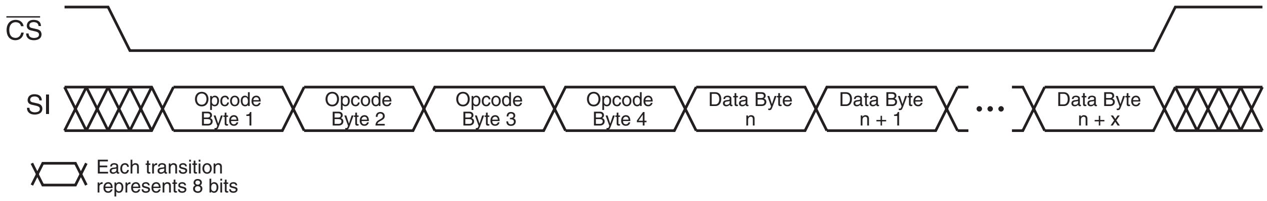
After the last data byte has been clocked in, the $\overline{\text{CS}}$ pin must be deasserted to initiate the internally self-timed program cycle. The programming of the Security Register should take place in a time of t_P, during which time the Status Register will indicate that the device is busy. If the device is powered-down during the program cycle, then the contents of the 64-byte user programmable portion of the Security Register cannot be guaranteed.

If the full 64 bytes of data is not clocked in before the $\overline{\text{CS}}$ pin is deasserted, then the values of the byte locations not clocked in cannot be guaranteed. For example, if only the first two bytes are clocked in instead of the complete 64 bytes, then the remaining 62 bytes of the user programmable portion of the Security Register cannot be guaranteed. Furthermore, if more than 64 bytes of data is clocked into the device, then the data will wrap back around to the beginning of the register. For instance, if 65 bytes of data are clocked in, then the 65th byte will be stored at byte location 0 of the Security Register.

The user programmable portion of the Security Register can only be programmed one time. Therefore, it is not possible to only program the first two bytes of the register and then program the remaining 62 bytes at a later time.

The Program Security Register command utilizes the internal SRAM buffer 1 for processing. Therefore, the contents of the buffer 1 will be altered from its previous state when this command is issued.

Figure 10-3. Program Security Register

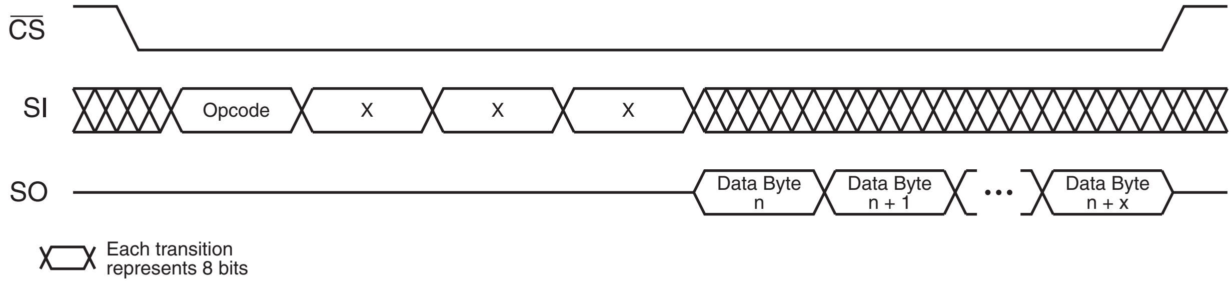


10.2.2 Reading the Security Register

The Security Register can be read by first asserting the $\overline{CS}$ pin and then clocking in an opcode of 77H followed by three dummy bytes. After the last don't care bit has been clocked in, the content of the Security Register can be clocked out on the SO pins. After the last byte of the Security Register has been read, additional pulses on the SCK pin will simply result in undefined data being output on the SO pins.

Deasserting the $\overline{CS}$ pin will terminate the Read Security Register operation and put the SO pins into a high-impedance state.

Figure 10-4. Read Security Register



11. Additional Commands

11.1 Main Memory Page to Buffer Transfer

A page of data can be transferred from the main memory to either buffer 1 or buffer 2. To start the operation for the DataFlash standard page size (528 bytes), a 1-byte opcode, 53H for buffer 1 and 55H for buffer 2, must be clocked into the device, followed by three address bytes comprised of 1 don't care bit, 13-page address bit (PA12 - PA0), which specify the page in main memory that is to be transferred, and 10 don't care bits. To perform a main memory page to buffer transfer for the binary page size (512 bytes), the opcode 53H for buffer 1 or 55H for buffer 2, must be clocked into the device followed by three address bytes consisting of 2 don't care bits, 13-page address bits (A21 - A9) which specify the page in the main memory that is to be transferred, and 9 don't care bits. The $\overline{CS}$ pin must be low while toggling the SCK pin to load the opcode and the address bytes from the input pin (SI). The transfer of the page of data from the main memory to the buffer will begin when the $\overline{CS}$ pin transitions from a low to a high state. During the transfer of a page of data (t_{XFR}), the status register can be read or the RDY/$\overline{BUSY}$ can be monitored to determine whether the transfer has been completed.

11.2 Main Memory Page to Buffer Compare

A page of data in the main memory can be compared to the data in buffer 1 or buffer 2. To initiate the operation for DataFlash standard page size, a 1-byte opcode, 60H for buffer 1 and 61H for buffer 2, must be clocked into the device, followed by three address bytes consisting of 1 don’t care bit, 13-page address bits (PA12 - PA0) that specify the page in the main memory that is to be compared to the buffer, and 10 don’t care bits. To start a main memory page to buffer compare for a binary page size, the opcode 60H for buffer 1 or 61H for buffer 2, must be clocked into the device followed by three address bytes consisting of 2 don’t care bits, 13 page address bits (A21 - A9) that specify the page in the main memory that is to be compared to the buffer, and 9 don’t care bits. The $\overline{\text{CS}}$ pin must be low while toggling the SCK pin to load the opcode and the address bytes from the input pin (SI). On the low-to-high transition of the $\overline{\text{CS}}$ pin, the data bytes in the selected main memory page will be compared with the data bytes in buffer 1 or buffer 2. During this time (t_{COMP}), the status register and the RDY/$\overline{\text{BUSY}}$ pin will indicate that the part is busy. On completion of the compare operation, bit 6 of the status register is updated with the result of the compare.

11.3 Auto Page Rewrite

This mode is only needed if multiple bytes within a page or multiple pages of data are modified in a random fashion within a sector. This mode is a combination of two operations: Main Memory Page to Buffer Transfer and Buffer to Main Memory Page Program with Built-in Erase. A page of data is first transferred from the main memory to buffer 1 or buffer 2, and then the same data (from buffer 1 or buffer 2) is programmed back into its original page of main memory. To start the rewrite operation for the DataFlash standard page size (528 bytes), a 1-byte opcode, 58H for buffer 1 or 59H for buffer 2, must be clocked into the device, followed by three address bytes comprised of 1 don’t care bit, 13-page address bits (PA12-PA0) that specify the page in main memory to be rewritten and 10 don’t care bits. To initiate an auto page rewrite for a binary page size (512 bytes), the opcode 58H for buffer 1 or 59H for buffer 2, must be clocked into the device followed by three address bytes consisting of 2 don’t care bits, 13 page address bits (A21 - A9) that specify the page in the main memory that is to be written and 9 don’t care bits. When a low-to-high transition occurs on the $\overline{\text{CS}}$ pin, the part will first transfer data from the page in main memory to a buffer and then program the data from the buffer back into same page of main memory. The operation is internally self-timed and should take place in a maximum time of t_{EP}. During this time, the status register and the RDY/$\overline{\text{BUSY}}$ pin will indicate that the part is busy.

If a sector is programmed or reprogrammed sequentially page by page, then the programming algorithm shown in Figure 25-1 (page 45) is recommended. Otherwise, if multiple bytes in a page or several pages are programmed randomly in a sector, then the programming algorithm shown in Figure 25-2 (page 46) is recommended. Each page within a sector must be updated/rewritten at least once within every 10,000 cumulative page erase/program operations in that sector.

11.4 Status Register Read

The status register can be used to determine the device’s ready/busy status, page size, a Main Memory Page to Buffer Compare operation result, the Sector Protection status or the device density. The Status Register can be read at any time, including during an internally self-timed program or erase operation. To read the status register, the $\overline{\text{CS}}$ pin must be asserted and the opcode of D7H must be loaded into the device. After the opcode is clocked in, the 1-byte status register will be clocked out on the output pin (SO), starting with the next clock cycle. The data in the status register, starting with the MSB (bit 7), will be clocked out on the SO pin during the next eight clock cycles. After the one byte of the status register has been clocked out, the sequence will repeat itself (as long as $\overline{\text{CS}}$ remains low and SCK is being toggled). The data in the status register is constantly updated, so each repeating sequence will output new data.

Ready/busy status is indicated using bit 7 of the status register. If bit 7 is a 1, then the device is not busy and is ready to accept the next command. If bit 7 is a 0, then the device is in a busy state. Since the data in the status register is constantly updated, the user must toggle SCK pin to check the ready/busy status. There are several operations that can cause the device to be in a busy state: Main Memory Page to Buffer Transfer, Main Memory Page to Buffer Compare, Buffer to Main Memory Page Program, Main Memory Page Program through Buffer, Page Erase, Block Erase, Sector Erase, Chip Erase and Auto Page Rewrite.

The result of the most recent Main Memory Page to Buffer Compare operation is indicated using bit 6 of the status register. If bit 6 is a 0, then the data in the main memory page matches the data in the buffer. If bit 6 is a 1, then at least one bit of the data in the main memory page does not match the data in the buffer.

Bit 1 in the Status Register is used to provide information to the user whether or not the sector protection has been enabled or disabled, either by software-controlled method or hardware-controlled method. A logic 1 indicates that sector protection has been enabled and logic 0 indicates that sector protection has been disabled.

Bit 0 in the Status Register indicates whether the page size of the main memory array is configured for “power of 2” binary page size (512 bytes) or DataFlash standard page size (528 bytes). If bit 0 is a 1, then the page size is set to 512 bytes. If bit 0 is a 0, then the page size is set to 528 bytes.

The device density is indicated using bits 5, 4, 3, and 2 of the status register. For the AT45DB321D, the four bits are 1101 The decimal value of these four binary bits does not equate to the device density; the four bits represent a combinational code relating to differing densities of DataFlash devices. The device density is not the same as the density code indicated in the JEDEC device ID information. The device density is provided only for backward compatibility.

Table 11-1. Status Register Format

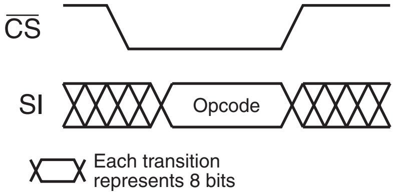
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
RDY/$\overline{\text{BUSY}}$	COMP	1	1	0	1	PROTECT	PAGE SIZE

12. Deep Power-down

After initial power-up, the device will default in standby mode. The Deep Power-down command allows the device to enter into the lowest power consumption mode. To enter the Deep Power-down mode, the $\overline{CS}$ pin must first be asserted. Once the $\overline{CS}$ pin has been asserted, an opcode of B9H command must be clocked in via input pin (SI). After the last bit of the command has been clocked in, the $\overline{CS}$ pin must be de-asserted to initiate the Deep Power-down operation. After the $\overline{CS}$ pin is de-asserted, the will device enter the Deep Power-down mode within the maximum t_{EDPD} time. Once the device has entered the Deep Power-down mode, all instructions are ignored except for the Resume from Deep Power-down command.

Command	Opcode
Deep Power-down	B9H

Figure 12-1. Deep Power-down

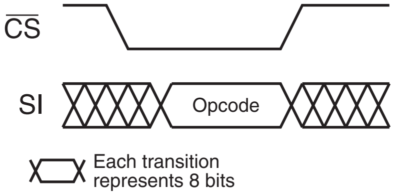


12.1 Resume from Deep Power-down

The Resume from Deep Power-down command takes the device out of the Deep Power-down mode and returns it to the normal standby mode. To Resume from Deep Power-down mode, the $\overline{CS}$ pin must first be asserted and an opcode of ABH command must be clocked in via input pin (SI). After the last bit of the command has been clocked in, the $\overline{CS}$ pin must be de-asserted to terminate the Deep Power-down mode. After the $\overline{CS}$ pin is de-asserted, the device will return to the normal standby mode within the maximum t_{RDPD} time. The $\overline{CS}$ pin must remain high during the t_{RDPD} time before the device can receive any commands. After resuming form Deep Power-down, the device will return to the normal standby mode.

Command	Opcode
Resume from Deep Power-down	ABH

Figure 12-2. Resume from Deep Power-Down



13. “Power of 2” Binary Page Size Option

“Power of 2” binary page size Configuration Register is a user-programmable nonvolatile register that allows the page size of the main memory to be configured for binary page size (512 bytes) or DataFlash standard page size (528 bytes). **The “power of 2” page size is a one-time programmable configuration register and once the device is configured for “power of 2” page size, it cannot be reconfigured again.** The devices are initially shipped with the page size set to 528 bytes.

For the binary “power of 2” page size to become effective, the following steps must be followed:

1. Program the one-time programmable configuration resister using opcode sequence 3DH, 2AH, 80H and A6H (please see Section 13.1).
2. Power cycle the device (i.e. power down and power up again).
3. The page for the binary page size can now be programmed.

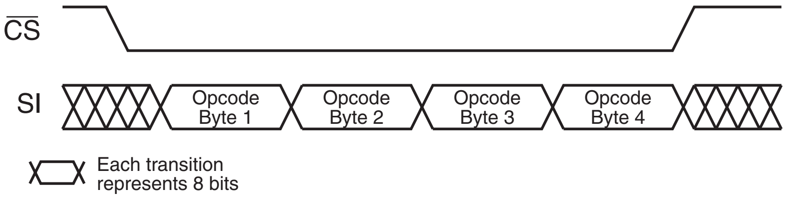
If the above steps are not followed to set the page size prior to page programming, incorrect data during a read operation may be encountered.

13.1 Programming the Configuration Register

To program the Configuration Register for “power of 2” binary page size, the $\overline{CS}$ pin must first be asserted as it would be with any other command. Once the $\overline{CS}$ pin has been asserted, the appropriate 4-byte opcode sequence must be clocked into the device in the correct order. The 4-byte opcode sequence must start with 3DH and be followed by 2AH, 80H, and A6H. After the last bit of the opcode sequence has been clocked in, the $\overline{CS}$ pin must be deasserted to initiate the internally self-timed program cycle. The programming of the Configuration Register should take place in a time of t_P, during which time the Status Register will indicate that the device is busy. The device must be power cycled after the completion of the program cycle to set the “power of 2” page size. If the device is powered-down before the completion of the program cycle, then setting the Configuration Register cannot be guaranteed. However, the user should check bit 0 of the status register to see whether the page size was configured for binary page size. If not, the command can be re-issued again.

Command	Byte 1	Byte 2	Byte 3	Byte 4
Power of Two Page Size	3DH	2AH	80H	A6H

Figure 13-1. Erase Sector Protection Register



14. Manufacturer and Device ID Read

Identification information can be read from the device to enable systems to electronically query and identify the device while it is in system. The identification method and the command opcode comply with the JEDEC standard for “Manufacturer and Device ID Read Methodology for SPI Compatible Serial Interface Memory Devices”. The type of information that can be read from the device includes the JEDEC defined Manufacturer ID, the vendor specific Device ID, and the vendor specific Extended Device Information.

To read the identification information, the $\overline{CS}$ pin must first be asserted and the opcode of 9FH must be clocked into the device. After the opcode has been clocked in, the device will begin outputting the identification data on the SO pin during the subsequent clock cycles. The first byte that will be output will be the Manufacturer ID followed by two bytes of Device ID information. The fourth byte output will be the Extended Device Information String Length, which will be 00H indicating that no Extended Device Information follows. As indicated in the JEDEC standard, reading the Extended Device Information String Length and any subsequent data is optional.

Deasserting the $\overline{CS}$ pin will terminate the Manufacturer and Device ID Read operation and put the SO pin into a high-impedance state. The $\overline{CS}$ pin can be deasserted at any time and does not require that a full byte of data be read.

14.1 Manufacturer and Device ID Information

14.1.1 Byte 1 – Manufacturer ID

Hex Value	JEDEC Assigned Code							
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
1FH	0	0	0	1	1	1	1	1

Manufacturer ID	1FH = Atmel

14.1.2 Byte 2 – Device ID (Part 1)

Hex Value	Family Code			Density Code				
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
27H	0	0	1	0	0	1	1	1

Family Code	001 = DataFlash
Density Code	00111 = 32-Mbit

14.1.3 Byte 3 – Device ID (Part 2)

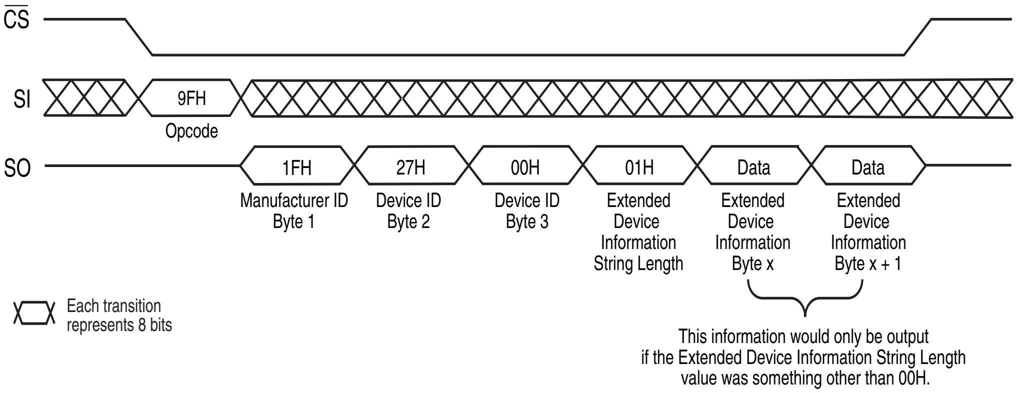
Hex Value	MLC Code			Product Version Code				
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
00H	0	0	0	0	0	0	0	1

MLC Code	000 = 1-bit/Cell Technology
Product Version	00001 = Second Version

14.1.4 Byte 4 – Extended Device Information String Length

Hex Value	Byte Count							
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
00H	0	0	0	0	0	0	0	0

Byte Count	00H = 0 Bytes of Information


CS
SI
9FH
Opcode
SO
1FH
Manufacturer ID Byte 1
27H
Device ID Byte 2
00H
Device ID Byte 3
01H
Extended Device Information String Length
Data
Extended Device Information Byte x
Data
Extended Device Information Byte x + 1
Each transition represents 8 bits
This information would only be output if the Extended Device Information String Length value was something other than 00H.

Note: Based on JEDEC publication 106 (JEP106), Manufacturer ID data can be comprised of any number of bytes. Some manufacturers may have Manufacturer ID codes that are two, three or even four bytes long with the first byte(s) in the sequence being 7FH. A system should detect code 7FH as a "Continuation Code" and continue to read Manufacturer ID bytes. The first non-7FH byte would signify the last byte of Manufacturer ID data. For Atmel (and some other manufacturers), the Manufacturer ID data is comprised of only one byte.

14.2 Operation Mode Summary

The commands described previously can be grouped into four different categories to better describe which commands can be executed at what times.

Group A commands consist of:

1. Main Memory Page Read
2. Continuous Array Read
3. Read Sector Protection Register
4. Read Sector Lockdown Register
5. Read Security Register

Group B commands consist of:

1. Page Erase
2. Block Erase
3. Sector Erase
4. Chip Erase
5. Main Memory Page to Buffer 1 (or 2) Transfer
6. Main Memory Page to Buffer 1 (or 2) Compare
7. Buffer 1 (or 2) to Main Memory Page Program with Built-in Erase
8. Buffer 1 (or 2) to Main Memory Page Program without Built-in Erase
9. Main Memory Page Program through Buffer 1 (or 2)
10. Auto Page Rewrite

Group C commands consist of:

1. Buffer 1 (or 2) Read
2. Buffer 1 (or 2) Write
3. Status Register Read
4. Manufacturer and Device ID Read

Group D commands consist of:

1. Erase Sector Protection Register
2. Program Sector Protection Register
3. Sector Lockdown
4. Program Security Register

If a Group A command is in progress (not fully completed), then another command in Group A, B, C, or D should not be started. However, during the internally self-timed portion of Group B commands, any command in Group C can be executed. The Group B commands using buffer 1 should use Group C commands using buffer 2 and vice versa. Finally, during the internally self-timed portion of a Group D command, only the Status Register Read command should be executed.

15. Command Tables

Table 15-1. Read Commands

Command	Opcode
Main Memory Page Read	D2H
Continuous Array Read (Legacy Command)	E8H
Continuous Array Read (Low Frequency)	03H
Continuous Array Read (High Frequency)	0BH
Buffer 1 Read (Low Frequency)	D1H
Buffer 2 Read (Low Frequency)	D3H
Buffer 1 Read	D4H
Buffer 2 Read	D6H

Table 15-2. Program and Erase Commands

Command	Opcode
Buffer 1 Write	84H
Buffer 2 Write	87H
Buffer 1 to Main Memory Page Program with Built-in Erase	83H
Buffer 2 to Main Memory Page Program with Built-in Erase	86H
Buffer 1 to Main Memory Page Program without Built-in Erase	88H
Buffer 2 to Main Memory Page Program without Built-in Erase	89H
Page Erase	81H
Block Erase	50H
Sector Erase	7CH
Chip Erase	C7H, 94H, 80H, 9AH
Main Memory Page Program Through Buffer 1	82H
Main Memory Page Program Through Buffer 2	85H

Table 15-3. Protection and Security Commands

Command	Opcode
Enable Sector Protection	3DH + 2AH + 7FH + A9H
Disable Sector Protection	3DH + 2AH + 7FH + 9AH
Erase Sector Protection Register	3DH + 2AH + 7FH + CFH
Program Sector Protection Register	3DH + 2AH + 7FH + FCH
Read Sector Protection Register	32H
Sector Lockdown	3DH + 2AH + 7FH + 30H
Read Sector Lockdown Register	35H
Program Security Register	9BH + 00H + 00H + 00H
Read Security Register	77H

Table 15-4. Additional Commands

Command	Opcode
Main Memory Page to Buffer 1 Transfer	53H
Main Memory Page to Buffer 2 Transfer	55H
Main Memory Page to Buffer 1 Compare	60H
Main Memory Page to Buffer 2 Compare	61H
Auto Page Rewrite through Buffer 1	58H
Auto Page Rewrite through Buffer 2	59H
Deep Power-down	B9H
Resume from Deep Power-down	ABH
Status Register Read	D7H
Manufacturer and Device ID Read	9FH

Table 15-5. Legacy Commands[1]

Command	Opcode
Buffer 1 Read	54H
Buffer 2 Read	56H
Main Memory Page Read	52H
Continuous Array Read	68H
Status Register Read	57H

Note: 1. These legacy commands are not recommended for new designs.

Table 15-6. Detailed Bit-level Addressing Sequence for Binary Page Size (512 Bytes)

Page Size = 512 bytes									Address Byte								Address Byte								Address Byte								Additional Don't Care Bytes
Opcode	Opcode								Reserved	Reserved	A21	A20	A19	A18	A17	A16	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	
03h	0	0	0	0	0	0	1	1	x	x	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	N/A
0Bh	0	0	0	0	1	0	1	1	x	x	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	1
50h	0	1	0	1	0	0	0	0	x	x	A	A	A	A	A	A	A	A	A	A	x	x	x	x	x	x	x	x	x	x	x	x	N/A
53h	0	1	0	1	0	0	1	1	x	x	A	A	A	A	A	A	A	A	A	A	A	A	A	x	x	x	x	x	x	x	x	x	N/A
55h	0	1	0	1	0	1	0	1	x	x	A	A	A	A	A	A	A	A	A	A	A	A	A	x	x	x	x	x	x	x	x	x	N/A
58h	0	1	0	1	1	0	0	0	x	x	A	A	A	A	A	A	A	A	A	A	A	A	A	x	x	x	x	x	x	x	x	x	N/A
59h	0	1	0	1	1	0	0	1	x	x	A	A	A	A	A	A	A	A	A	A	A	A	A	x	x	x	x	x	x	x	x	x	N/A
60h	0	1	1	0	0	0	0	0	x	x	A	A	A	A	A	A	A	A	A	A	A	A	A	x	x	x	x	x	x	x	x	x	N/A
61h	0	1	1	0	0	0	0	1	x	x	A	A	A	A	A	A	A	A	A	A	A	A	A	x	x	x	x	x	x	x	x	x	N/A
77h	0	1	1	1	0	1	1	1	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	N/A
7Ch	0	1	1	1	1	1	0	0	x	x	A	A	A	A	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	N/A
81h	1	0	0	0	0	0	0	1	x	x	A	A	A	A	A	A	A	A	A	A	A	A	A	X	x	x	x	x	x	x	x	x	N/A
82h	1	0	0	0	0	0	1	0	x	x	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	N/A
83h	1	0	0	0	0	0	1	1	x	x	A	A	A	A	A	A	A	A	A	A	A	A	A	X	x	x	x	x	x	x	x	x	N/A
84h	1	0	0	0	0	1	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	A	A	A	A	A	A	A	A	A	N/A
85h	1	0	0	0	0	1	0	1	x	x	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	N/A
86h	1	0	0	0	0	1	1	0	x	x	A	A	A	A	A	A	A	A	A	A	A	A	A	x	x	x	x	x	x	x	x	x	N/A
87h	1	0	0	0	0	1	1	1	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	A	A	A	A	A	A	A	A	A	N/A
88h	1	0	0	0	1	0	0	0	x	x	A	A	A	A	A	A	A	A	A	A	A	A	A	x	x	x	x	x	x	x	x	x	N/A
89h	1	0	0	0	1	0	0	1	x	x	A	A	A	A	A	A	A	A	A	A	A	A	A	x	x	x	x	x	x	x	x	x	N/A
9Fh	1	0	0	1	1	1	1	1	N/A								N/A								N/A								N/A
B9h	1	0	1	1	1	0	0	1	N/A								N/A								N/A								N/A
ABh	1	0	1	0	1	0	1	1	N/A								N/A								N/A								N/A
D1h	1	1	0	1	0	0	0	1	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	A	A	A	A	A	A	A	A	A	N/A
D2h	1	1	0	1	0	0	1	0	x	x	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	4
D3h	1	1	0	1	0	0	1	1	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	A	A	A	A	A	A	A	A	A	N/A
D4h	1	1	0	1	0	1	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	A	A	A	A	A	A	A	A	A	1
D6h	1	1	0	1	0	1	1	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	A	A	A	A	A	A	A	A	A	1
D7h	1	1	0	1	0	1	1	1	N/A								N/A								N/A								N/A
E8h	1	1	1	0	1	0	0	0	x	x	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	4

Notes: x = Don't Care

Table 15-7. Detailed Bit-level Addressing Sequence for DataFlash Standard Page Size (528 Bytes)

Page Size = 528 bytes									Address Byte								Address Byte								Address Byte								
Opcode	**Opcode**								**Reserved**	**PA12**	**PA11**	**PA10**	**PA9**	**PA8**	**PA7**	**PA6**	**PA5**	**PA4**	**PA3**	**PA2**	**PA1**	**PA0**	**BA9**	**BA8**	**BA7**	**BA6**	**BA5**	**BA4**	**BA3**	**BA2**	**BA1**	**BA0**	**Additional Don't Care Bytes**
03h	0	0	0	0	0	0	1	1	x	P	P	P	P	P	P	P	P	P	P	P	P	P	B	B	B	B	B	B	B	B	B	B	N/A
0Bh	0	0	0	0	1	0	1	1	x	P	P	P	P	P	P	P	P	P	P	P	P	P	B	B	B	B	B	B	B	B	B	B	1
50h	0	1	0	1	0	0	0	0	x	P	P	P	P	P	P	P	P	P	P	x	x	x	x	x	x	x	x	x	x	x	x	x	N/A
53h	0	1	0	1	0	0	1	1	x	P	P	P	P	P	P	P	P	P	P	P	P	P	x	x	x	x	x	x	x	x	x	x	N/A
55h	0	1	0	1	0	1	0	1	x	P	P	P	P	P	P	P	P	P	P	P	P	P	x	x	x	x	x	x	x	x	x	x	N/A
58h	0	1	0	1	1	0	0	0	x	P	P	P	P	P	P	P	P	P	P	P	P	P	x	x	x	x	x	x	x	x	x	x	N/A
59h	0	1	0	1	1	0	0	1	x	P	P	P	P	P	P	P	P	P	P	P	P	P	x	x	x	x	x	x	x	x	x	x	N/A
60h	0	1	1	0	0	0	0	0	x	P	P	P	P	P	P	P	P	P	P	P	P	P	x	x	x	x	x	x	x	x	x	x	N/A
61h	0	1	1	0	0	0	0	1	x	P	P	P	P	P	P	P	P	P	P	P	P	P	x	x	x	x	x	x	x	x	x	x	N/A
77h	0	1	1	1	0	1	1	1	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	N/A
7Ch	0	1	1	1	1	1	0	0	x	P	P	P	P	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	N/A
81h	1	0	0	0	0	0	0	1	x	P	P	P	P	P	P	P	P	P	P	P	P	P	x	x	x	x	x	x	x	x	x	x	N/A
82h	1	0	0	0	0	0	1	0	x	P	P	P	P	P	P	P	P	P	P	P	P	P	B	B	B	B	B	B	B	B	B	B	N/A
83h	1	0	0	0	0	0	1	1	x	P	P	P	P	P	P	P	P	P	P	P	P	P	x	x	x	x	x	x	x	x	x	x	N/A
84h	1	0	0	0	0	1	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	B	B	B	B	B	B	B	B	B	B	N/A
85h	1	0	0	0	0	1	0	1	x	P	P	P	P	P	P	P	P	P	P	P	P	P	B	B	B	B	B	B	B	B	B	B	N/A
86h	1	0	0	0	0	1	1	0	x	P	P	P	P	P	P	P	P	P	P	P	P	P	x	x	x	x	x	x	x	x	x	x	N/A
87h	1	0	0	0	0	1	1	1	x	x	x	x	x	x	x	x	x	x	x	x	x	x	B	B	B	B	B	B	B	B	B	B	N/A
88h	1	0	0	0	1	0	0	0	x	P	P	P	P	P	P	P	P	P	P	P	P	P	x	x	x	x	x	x	x	x	x	x	N/A
89h	1	0	0	0	1	0	0	1	x	P	P	P	P	P	P	P	P	P	P	P	P	P	x	x	x	x	x	x	x	x	x	x	N/A
9Fh	1	0	0	1	1	1	1	1	N/A								N/A								N/A								N/A
B9h	1	0	1	1	1	0	0	1	N/A								N/A								N/A								N/A
ABh	1	0	1	0	1	0	1	1	N/A								N/A								N/A								N/A
D1h	1	1	0	1	0	0	0	1	x	x	x	x	x	x	x	x	x	x	x	x	x	x	B	B	B	B	B	B	B	B	B	B	N/A
D2h	1	1	0	1	0	0	1	0	x	P	P	P	P	P	P	P	P	P	P	P	P	P	B	B	B	B	B	B	B	B	B	B	4
D3h	1	1	0	1	0	0	0	1	x	x	x	x	x	x	x	x	x	x	x	x	x	x	B	B	B	B	B	B	B	B	B	B	N/A
D4h	1	1	0	1	0	1	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	B	B	B	B	B	B	B	B	B	B	1
D6h	1	1	0	1	0	1	1	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	B	B	B	B	B	B	B	B	B	B	1
D7h	1	1	0	1	0	1	1	1	N/A								N/A								N/A								N/A
E8h	1	1	1	0	1	0	0	0	x	P	P	P	P	P	P	P	P	P	P	P	P	P	B	B	B	B	B	B	B	B	B	B	4

Notes: P = Page Address Bit B = Byte/Buffer Address Bit x = Don't Care

16. Power-on/Reset State

When power is first applied to the device, or when recovering from a reset condition, the device will default to Mode 3. In addition, the output pin (SO) will be in a high impedance state, and a high-to-low transition on the $\overline{CS}$ pin will be required to start a valid instruction. The mode (Mode 3 or Mode 0) will be automatically selected on every falling edge of $\overline{CS}$ by sampling the inactive clock state.

16.1 Initial Power-up/Reset Timing Restrictions

At power up, the device must not be selected until the supply voltage reaches the V_{CC} (min.) and further delay of t_{VCSL}. During power-up, the internal Power-on Reset circuitry keeps the device in reset mode until the V_{CC} rises above the Power-on Reset threshold value (V_{POR}). At this time, all operations are disabled and the device does not respond to any commands. After power up is applied and the V_{CC} is at the minimum operating voltage V_{CC} (min.), the t_{VCSL} delay is required before the device can be selected in order to perform a read operation.

Similarly, the t_{PUW} delay is required after the V_{CC} rises above the Power-on Reset threshold value (V_{POR}) before the device can perform a write (Program or Erase) operation. After initial power-up, the device will default in Standby mode.

Symbol	Parameter	Min	Typ	Max	Units
t_{VCSL}	V_{CC} (min.) to Chip Select low	70			µs
t_{PUW}	Power-Up Device Delay before Write Allowed			20	ms
V_{POR}	Power-ON Reset Voltage	1.5		2.5	V

17. System Considerations

The RapidS serial interface is controlled by the clock SCK, serial input SI and chip select $\overline{CS}$ pins. These signals must rise and fall monotonically and be free from noise. Excessive noise or ringing on these pins can be misinterpreted as multiple edges and cause improper operation of the device. The PC board traces must be kept to a minimum distance or appropriately terminated to ensure proper operation. If necessary, decoupling capacitors can be added on these pins to provide filtering against noise glitches.

As system complexity continues to increase, voltage regulation is becoming more important. A key element of any voltage regulation scheme is its current sourcing capability. Like all Flash memories, the peak current for DataFlash occur during the programming and erase operation. The regulator needs to supply this peak current requirement. An under specified regulator can cause current starvation. Besides increasing system noise, current starvation during programming or erase can lead to improper operation and possible data corruption.

18. Electrical Specifications

Table 18-1. Absolute Maximum Ratings*

Temperature under Bias	-55° C to +125° C
Storage Temperature	-65° C to +150° C
All Input Voltages (including NC Pins) with Respect to Ground	-0.6V to +6.25V
All Output Voltages with Respect to Ground	-0.6V to V_{CC} + 0.6V

*NOTICE: Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions beyond those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

Table 18-2. DC and AC Operating Range

		AT45DB321D
Operating Temperature (Case)	Ind.	-40° C to 85° C
V_{CC} Power Supply		2.7V to 3.6V

Table 18-3. DC Characteristics

Symbol	Parameter	Condition	Min	Typ	Max	Units
I_{DP}	Deep Power-down Current	$\overline{CS}$, $\overline{RESET}$, $\overline{WP}$ = V_{IH}, all inputs at CMOS levels		5	10	µA
I_{SB}	Standby Current	$\overline{CS}$, $\overline{RESET}$, $\overline{WP}$ = V_{IH}, all inputs at CMOS levels		25	50	µA
I_{CC1}[(1)]	Active Current, Read Operation	f = 20 MHz; I_{OUT} = 0 mA; V_{CC} = 3.6V		7	10	mA
		f = 33 MHz; I_{OUT} = 0 mA; V_{CC} = 3.6V		8	12	mA
		f = 50 MHz; I_{OUT} = 0 mA; V_{CC} = 3.6V		10	14	mA
		f = 66 MHz; I_{OUT} = 0 mA; V_{CC} = 3.6V		11	15	mA
I_{CC2}	Active Current, Program/Erase Operation	V_{CC} = 3.6V		12	17	mA
I_{LI}	Input Load Current	V_{IN} = CMOS levels			1	µA
I_{LO}	Output Leakage Current	$V_{I/O}$ = CMOS levels			1	µA
V_{IL}	Input Low Voltage				V_{CC} x 0.3	V
V_{IH}	Input High Voltage		V_{CC} x 0.7			V
V_{OL}	Output Low Voltage	I_{OL} = 1.6 mA; V_{CC} = 2.7V			0.4	V
V_{OH}	Output High Voltage	I_{OH} = -100 µA	V_{CC} - 0.2V			V

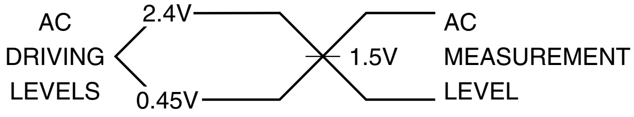
Notes: 1. I_{CC1} during a buffer read is 20 mA maximum @ 20 MHz.

2. All inputs are 5 volts tolerant.

Table 18-4. AC Characteristics – RapidS/Serial Interface

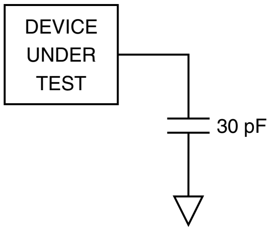
		AT45DB321D			
Symbol	**Parameter**	**Min**	**Typ**	**Max**	**Units**
f_{SCK}	SCK Frequency			66	MHz
f_{CAR1}	SCK Frequency for Continuous Array Read			66	MHz
f_{CAR2}	SCK Frequency for Continuous Array Read (Low Frequency)			33	MHz
t_{WH}	SCK High Time	6.8			ns
t_{WL}	SCK Low Time	6.8			ns
t_{SCKR}(1)	SCK Rise Time, Peak-to-Peak (Slew Rate)	0.1			V/ns
t_{SCKF}(1)	SCK Fall Time, Peak-to-Peak (Slew Rate)	0.1			V/ns
t_{CS}	Minimum $\overline{\text{CS}}$ High Time	50			ns
t_{CSS}	$\overline{\text{CS}}$ Setup Time	5			ns
t_{CSH}	$\overline{\text{CS}}$ Hold Time	5			ns
t_{CSB}	$\overline{\text{CS}}$ High to RDY/$\overline{\text{BUSY}}$ Low			100	ns
t_{SU}	Data In Setup Time	2			ns
t_{H}	Data In Hold Time	3			ns
t_{HO}	Output Hold Time	0			ns
t_{DIS}	Output Disable Time			6	ns
t_{V}	Output Valid			6	ns
t_{WPE}	$\overline{\text{WP}}$ Low to Protection Enabled			1	µs
t_{WPD}	$\overline{\text{WP}}$ High to Protection Disabled			1	µs
t_{EDPD}	$\overline{\text{CS}}$ High to Deep Power-down Mode			3	µs
t_{RDPD}	$\overline{\text{CS}}$ High to Standby Mode			35	µs
t_{XFR}	Page to Buffer Transfer Time			200	µs
t_{comp}	Page to Buffer Compare Time			200	µs
t_{EP}	Page Erase and Programming Time (512/528 bytes)		17	40	ms
t_{P}	Page Programming Time (512/528 bytes)		3	6	ms
t_{PE}	Page Erase Time (512/528 bytes)		15	35	ms
t_{BE}	Block Erase Time (4,096/4,224 bytes)		45	100	ms
t_{CE}	Chip Erase Time		TBD	TBD	s
t_{SE}	Sector Erase Time (262,144/270,336 bytes)		1.6	5	s
t_{RST}	$\overline{\text{RESET}}$ Pulse Width	10			µs
t_{REC}	$\overline{\text{RESET}}$ Recovery Time			1	µs

19. Input Test Waveforms and Measurement Levels



t_R, t_F < 2 ns (10% to 90%)

20. Output Test Load

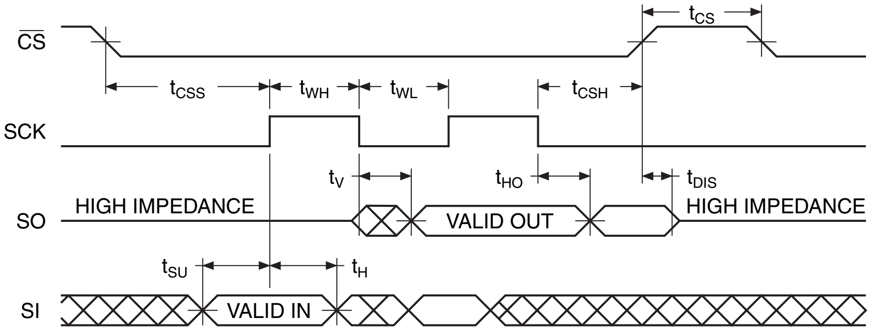


21. AC Waveforms

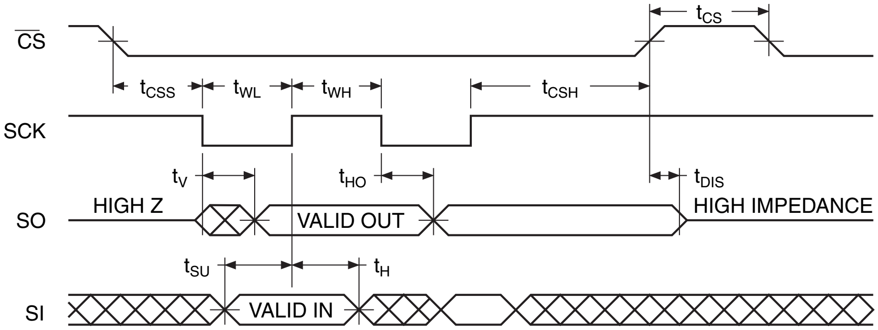
Six different timing waveforms are shown on page 36. Waveform 1 shows the SCK signal being low when $\overline{\text{CS}}$ makes a high-to-low transition, and waveform 2 shows the SCK signal being high when $\overline{\text{CS}}$ makes a high-to-low transition. In both cases, output SO becomes valid while the SCK signal is still low (SCK low time is specified as t_{WL}). Timing waveforms 1 and 2 conform to RapidS serial interface but for frequencies up to 66 MHz. Waveforms 1 and 2 are compatible with SPI Mode 0 and SPI Mode 3, respectively.

Waveform 3 and waveform 4 illustrate general timing diagram for RapidS serial interface. These are similar to waveform 1 and waveform 2, except that output SO is not restricted to become valid during the t_{WL} period. These timing waveforms are valid over the full frequency range (maximum frequency = 66 MHz) of the RapidS serial case.

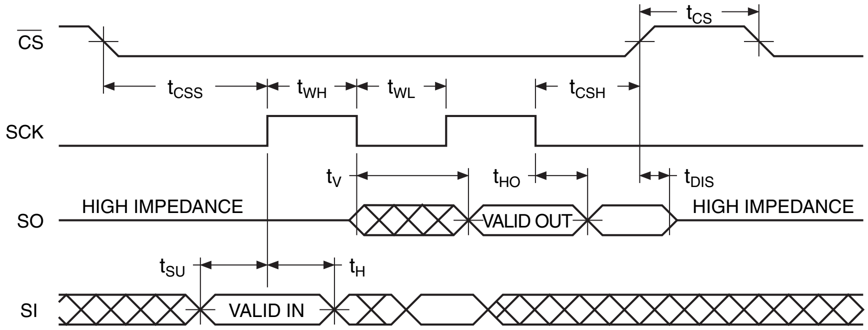
21.1 Waveform 1 – SPI Mode 0 Compatible (for frequencies up to 66 MHz)



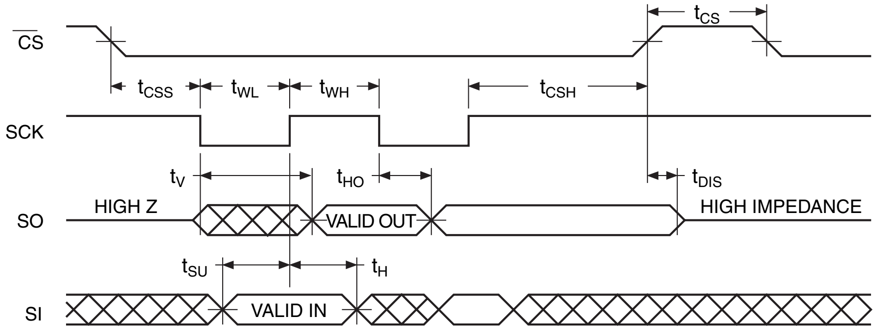
21.2 Waveform 2 – SPI Mode 3 Compatible (for frequencies up to 66 MHz)



21.3 Waveform 3 – RapidS Mode 0 (F_{MAX} = 66 MHz)



21.4 Waveform 4 – RapidS Mode 3 (F_{MAX} = 66 MHz)

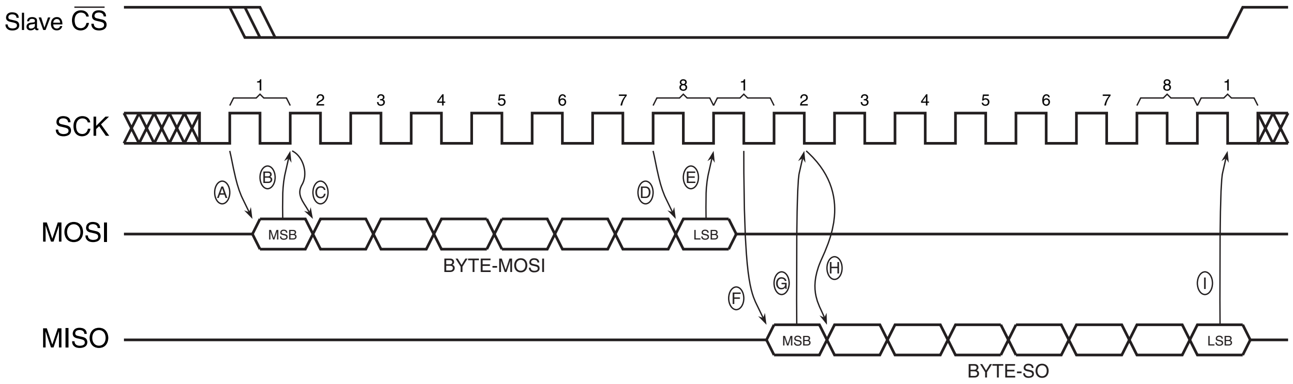


21.5 Utilizing the RapidS™ Function

To take advantage of the RapidS function's ability to operate at higher clock frequencies, a full clock cycle must be used to transmit data back and forth across the serial bus. The DataFlash is designed to always clock its data out on the falling edge of the SCK signal and clock data in on the rising edge of SCK.

For full clock cycle operation to be achieved, when the DataFlash is clocking data out on the falling edge of SCK, the host controller should wait until the next falling edge of SCK to latch the data in. Similarly, the host controller should clock its data out on the rising edge of SCK in order to give the DataFlash a full clock cycle to latch the incoming data in on the next rising edge of SCK.

Figure 21-1. RapidS Mode

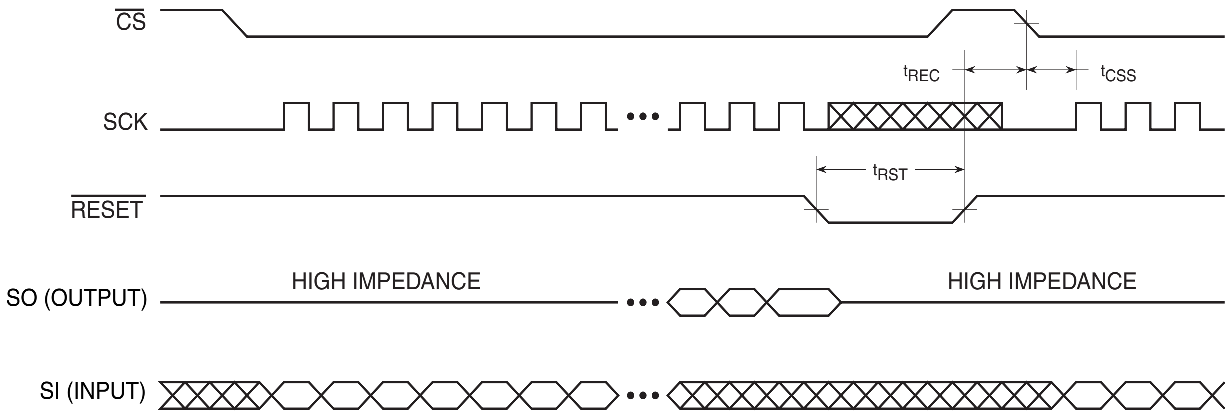


MOSI = Master Out, Slave In
MISO = Master In, Slave Out
The Master is the host controller and the Slave is the DataFlash

The Master always clocks data out on the rising edge of SCK and always clocks data in on the falling edge of SCK.
The Slave always clocks data out on the falling edge of SCK and always clocks data in on the rising edge of SCK.

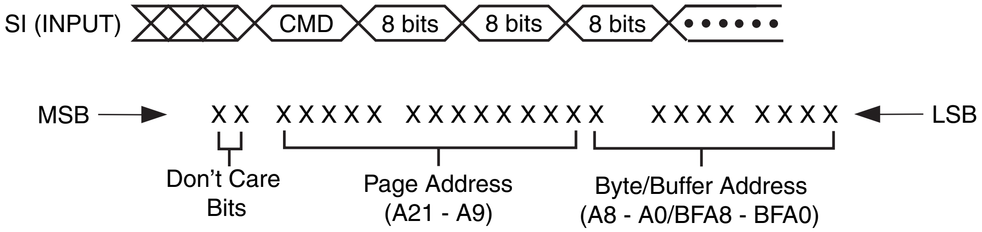
A. Master clocks out first bit of BYTE-MOSI on the rising edge of SCK.
B. Slave clocks in first bit of BYTE-MOSI on the next rising edge of SCK.
C. Master clocks out second bit of BYTE-MOSI on the same rising edge of SCK.
D. Last bit of BYTE-MOSI is clocked out from the Master.
E. Last bit of BYTE-MOSI is clocked into the slave.
F. Slave clocks out first bit of BYTE-SO.
G. Master clocks in first bit of BYTE-SO.
H. Slave clocks out second bit of BYTE-SO.
I. Master clocks in last bit of BYTE-SO.

21.6 Reset Timing

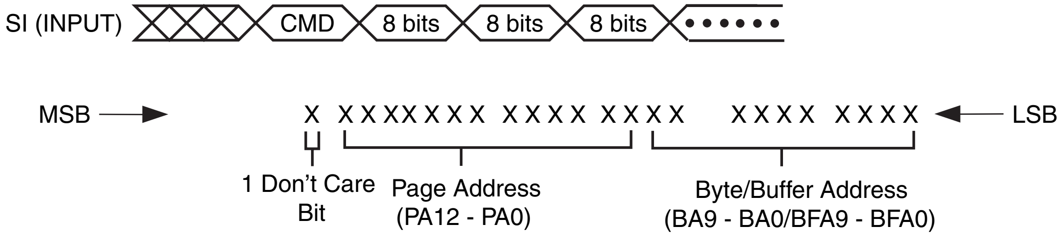


Note: The CS signal should be in the high state before the RESET signal is deasserted.

21.7 Command Sequence for Read/Write Operations for Page Size 512 Bytes (Except Status Register Read, Manufacturer and Device ID Read)

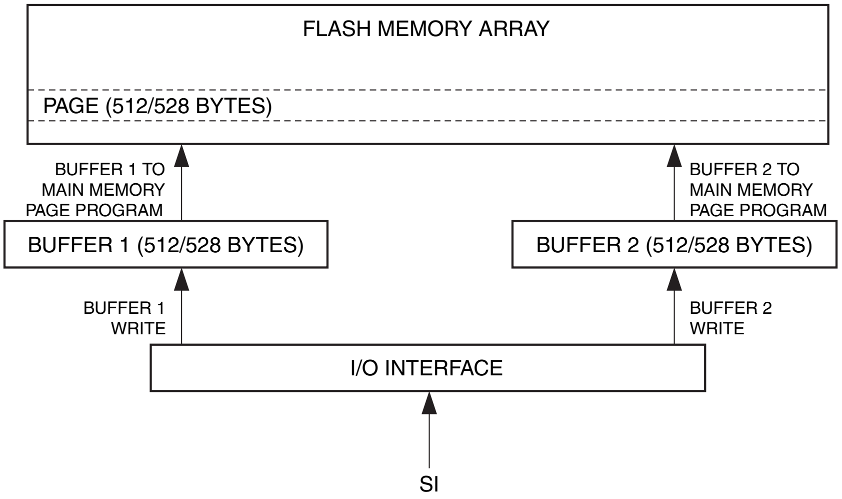

SI (INPUT)
CMD
8 bits
8 bits
8 bits
MSB
X X   X X X X X   X X X X X X X X X   X X X X   X X X X
LSB
Don't Care Bits
Page Address (A21 - A9)
Byte/Buffer Address (A8 - A0/BFA8 - BFA0)

21.8 Command Sequence for Read/Write Operations for Page Size 528 Bytes (Except Status Register Read, Manufacturer and Device ID Read)

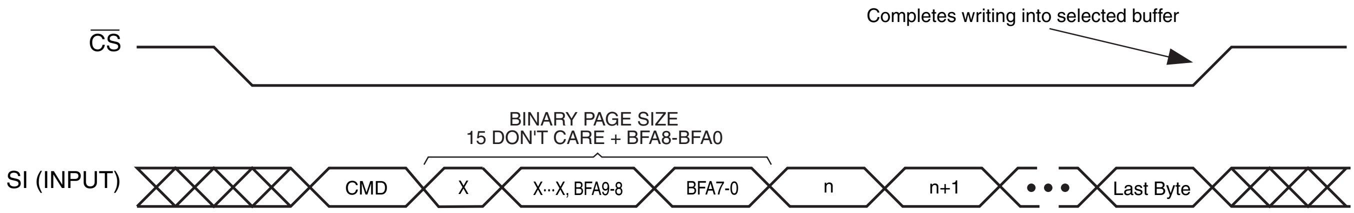

SI (INPUT)
CMD
8 bits
8 bits
8 bits
MSB
X   X X X X X X X   X X X X   X X X X   X X X X   X X X X
LSB
1 Don't Care Bit
Page Address (PA12 - PA0)
Byte/Buffer Address (BA9 - BA0/BFA9 - BFA0)

22. Write Operations

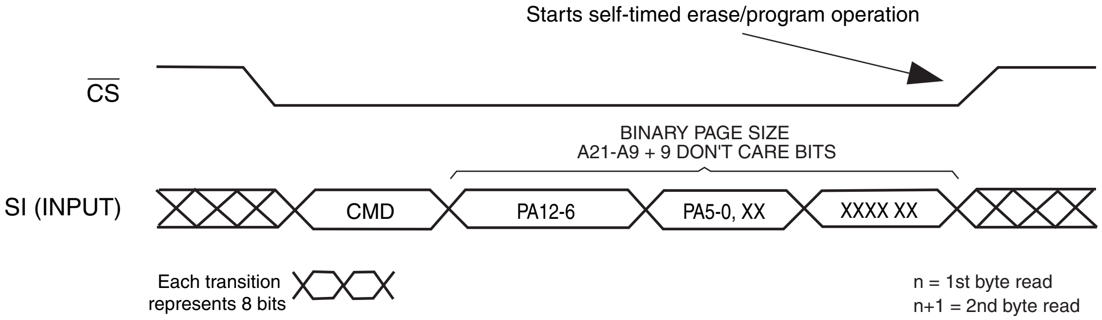
The following block diagram and waveforms illustrate the various write sequences available.



22.1 Buffer Write

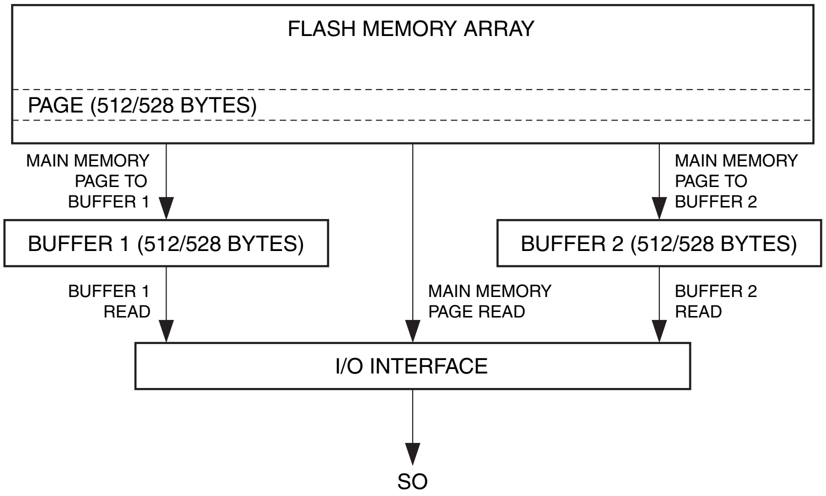


22.2 Buffer to Main Memory Page Program (Data from Buffer Programmed into Flash Page)

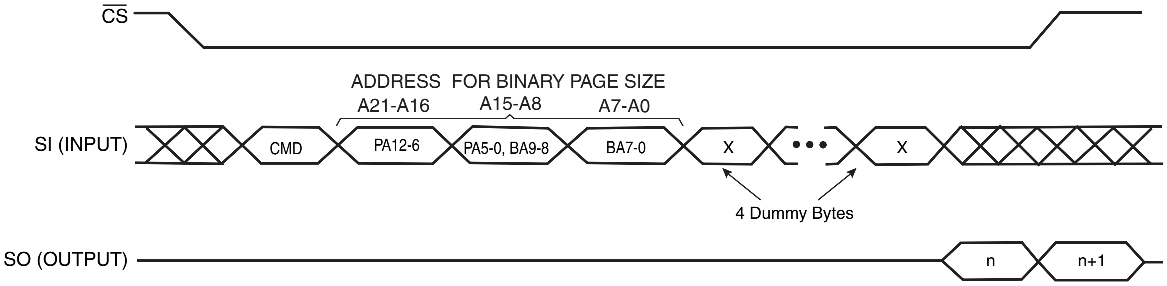


23. Read Operations

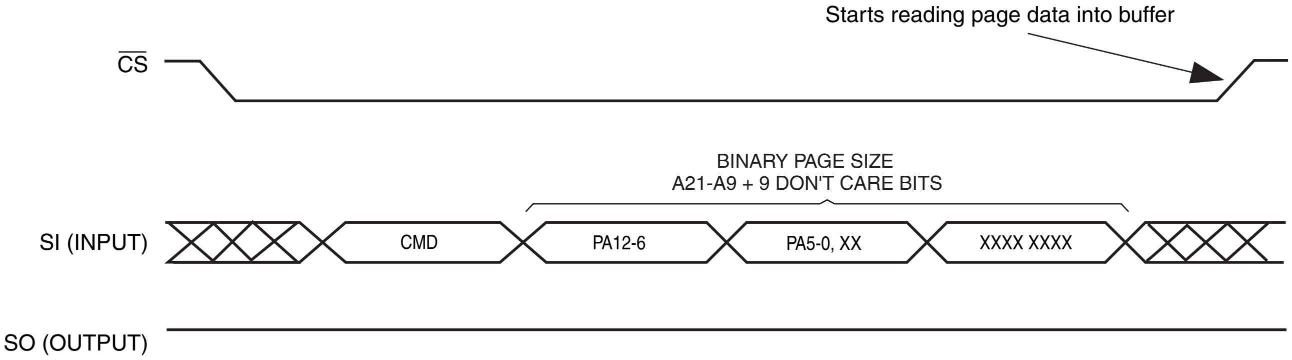
The following block diagram and waveforms illustrate the various read sequences available.



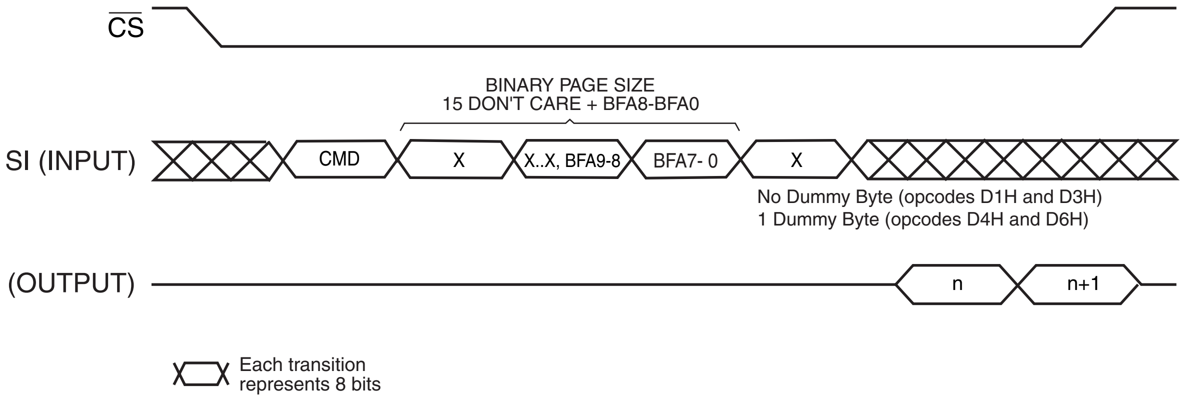
23.1 Main Memory Page Read



23.2 Main Memory Page to Buffer Transfer (Data from Flash Page Read into Buffer)

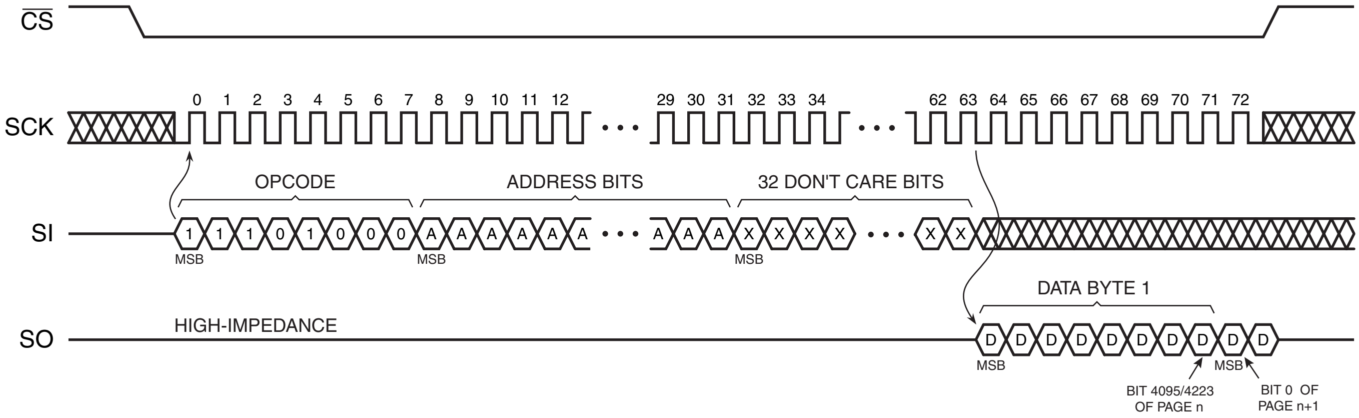


23.3 Buffer Read

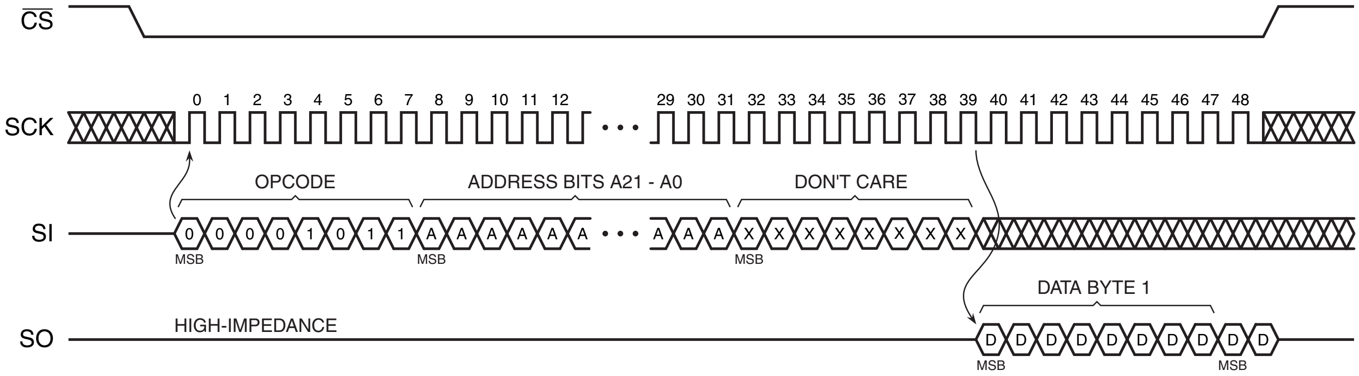


24. Detailed Bit-level Read Waveform – RapidS Serial Interface Mode 0/Mode 3

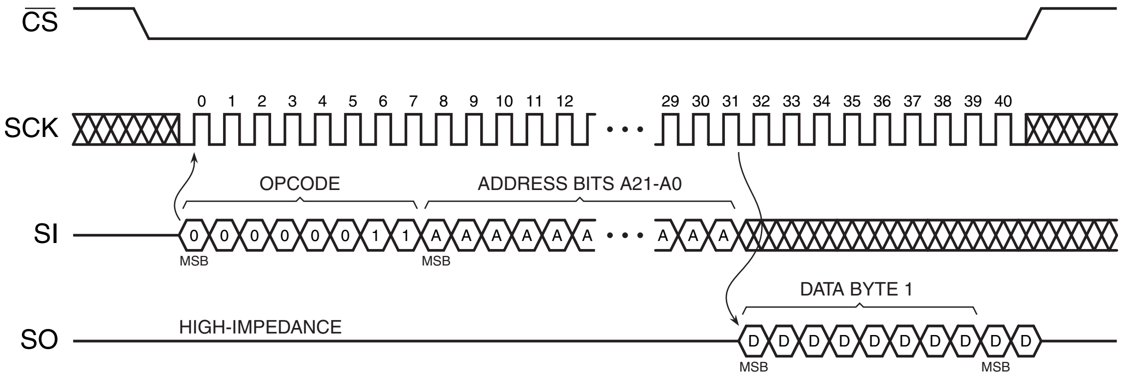
24.1 Continuous Array Read (Legacy Opcode E8H)



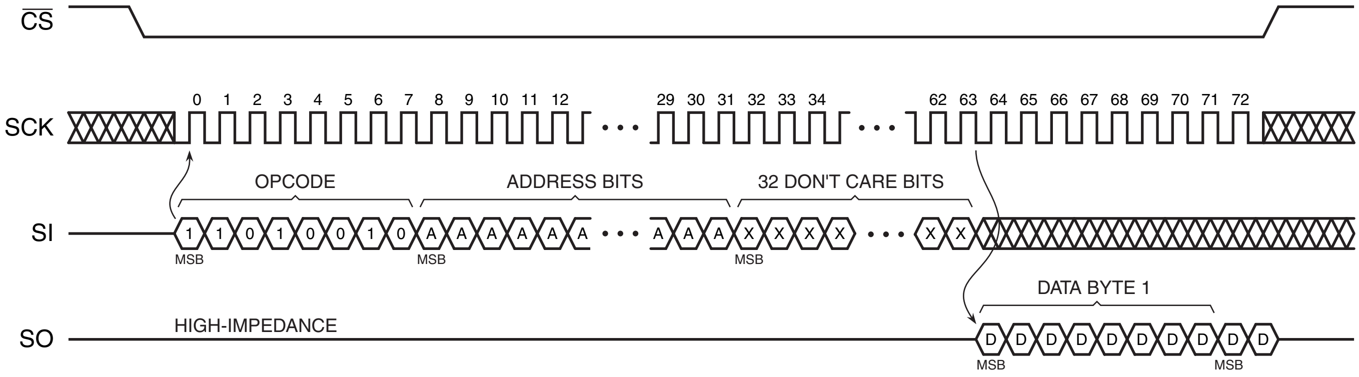
24.2 Continuous Array Read (Opcode 0BH)



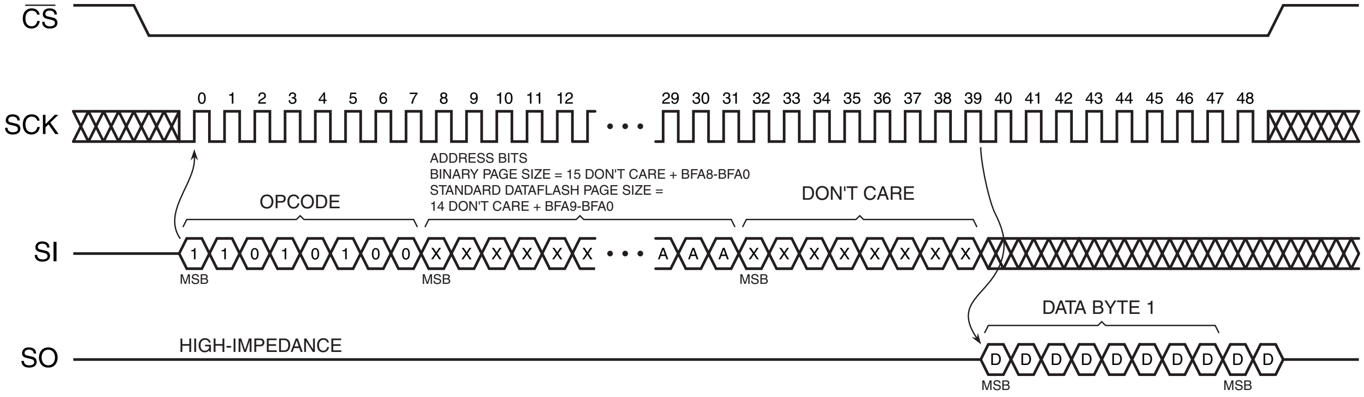
24.3 Continuous Array Read (Low Frequency: Opcode 03H)



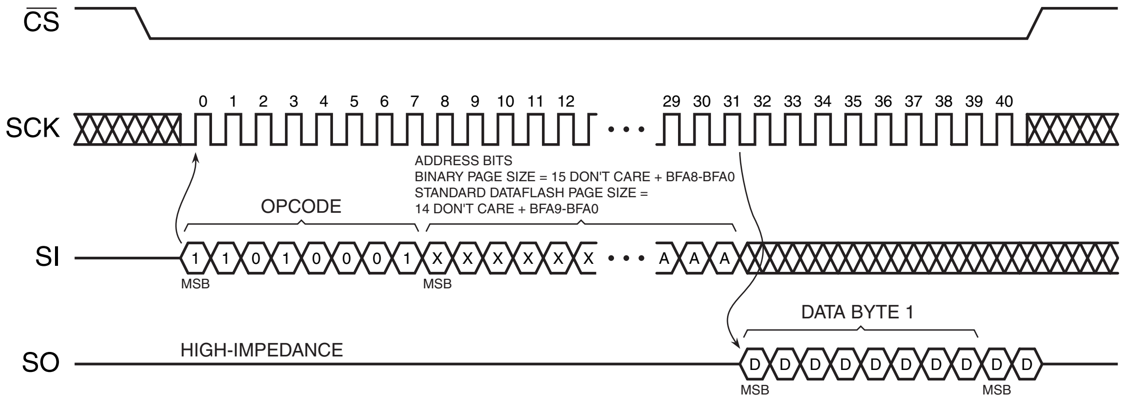
24.4 Main Memory Page Read (Opcode: D2H)



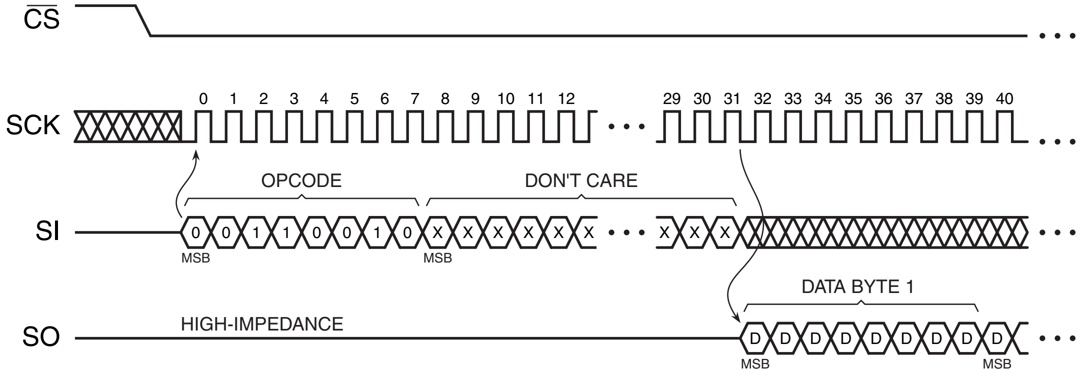
24.5 Buffer Read (Opcode D4H or D6H)



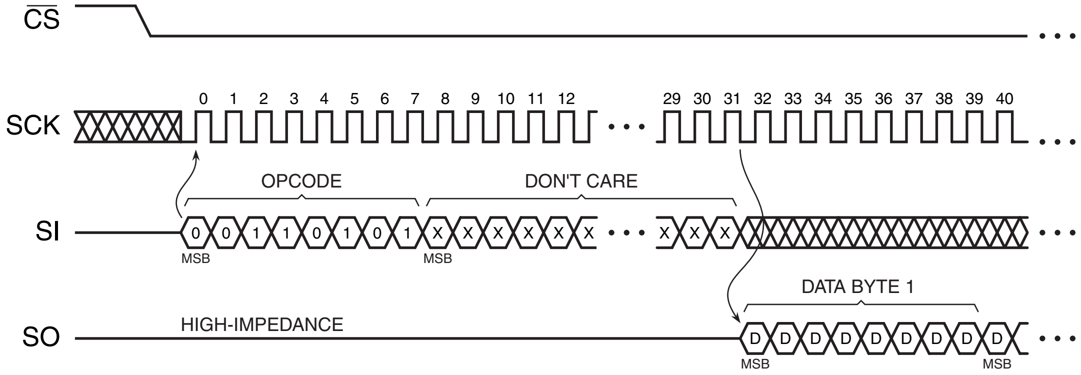
24.6 Buffer Read (Low Frequency: Opcode D1H or D3H)



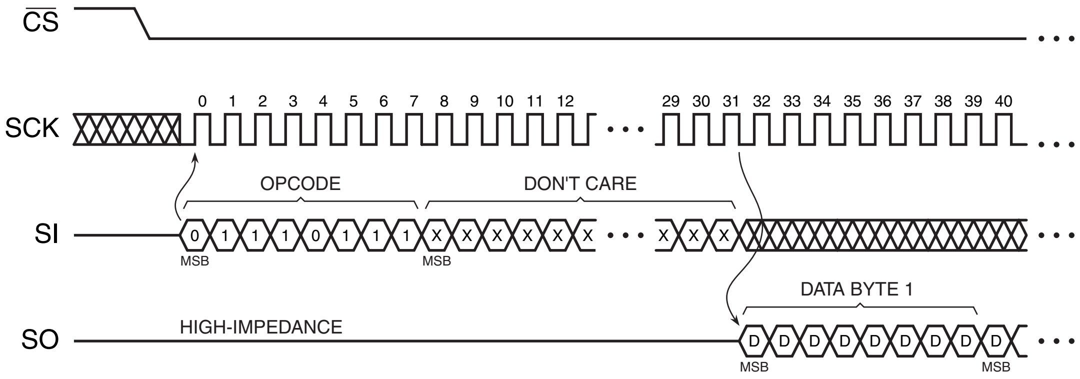
24.7 Read Sector Protection Register (Opcode 32H)



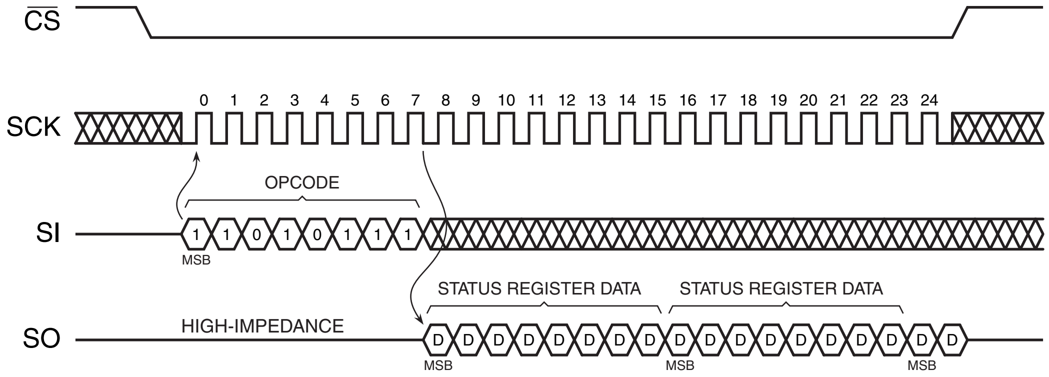
24.8 Read Sector Lockdown Register (Opcode 35H)



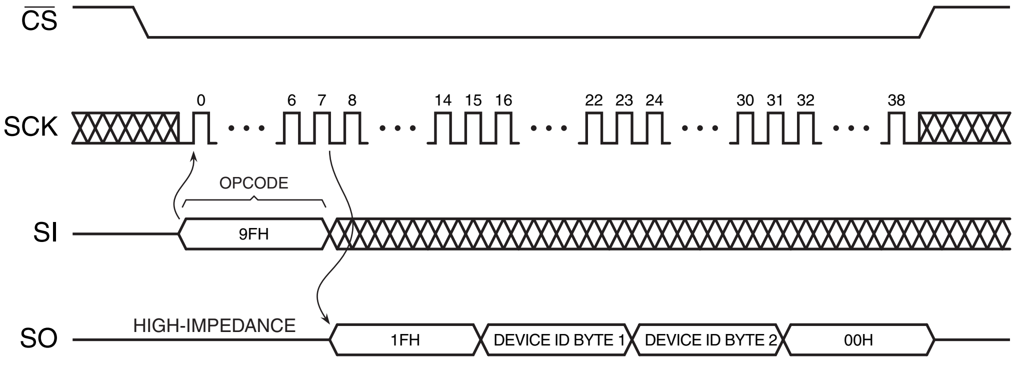
24.9 Read Security Register (Opcode 77H)



24.10 Status Register Read (Opcode D7H)



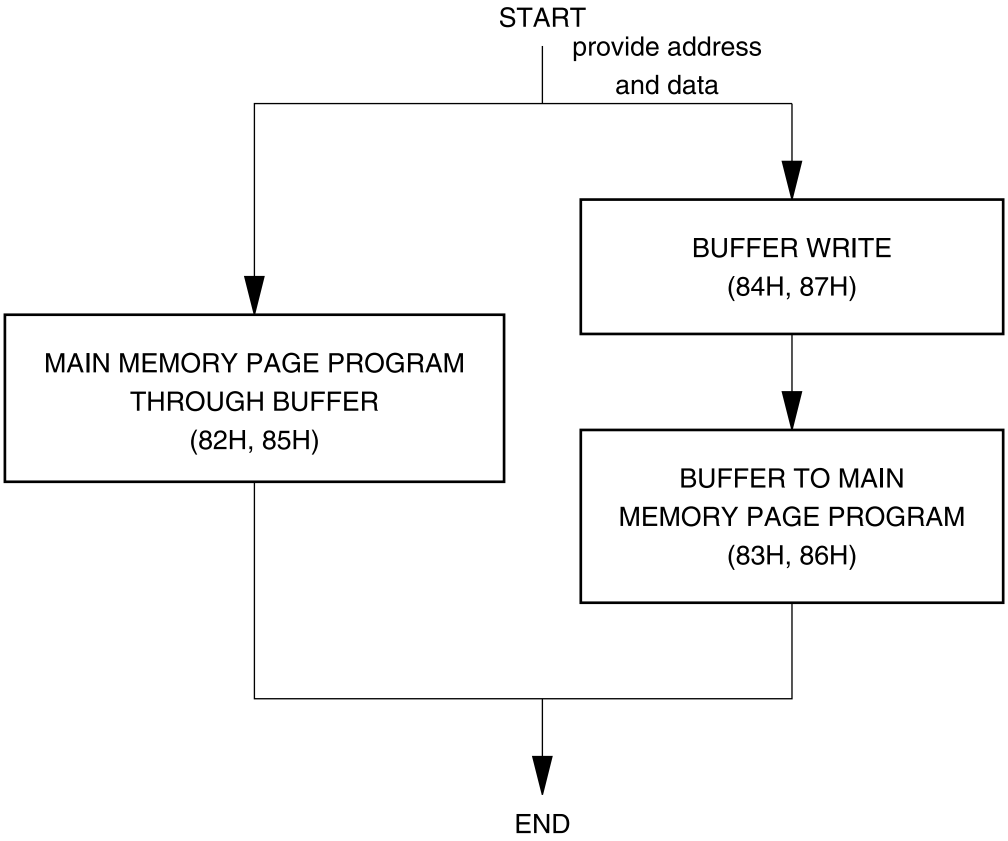
24.11 Manufacturer and Device Read (Opcode 9FH)



Note: Each transition shown for SI and SO represents one byte (8 bits)

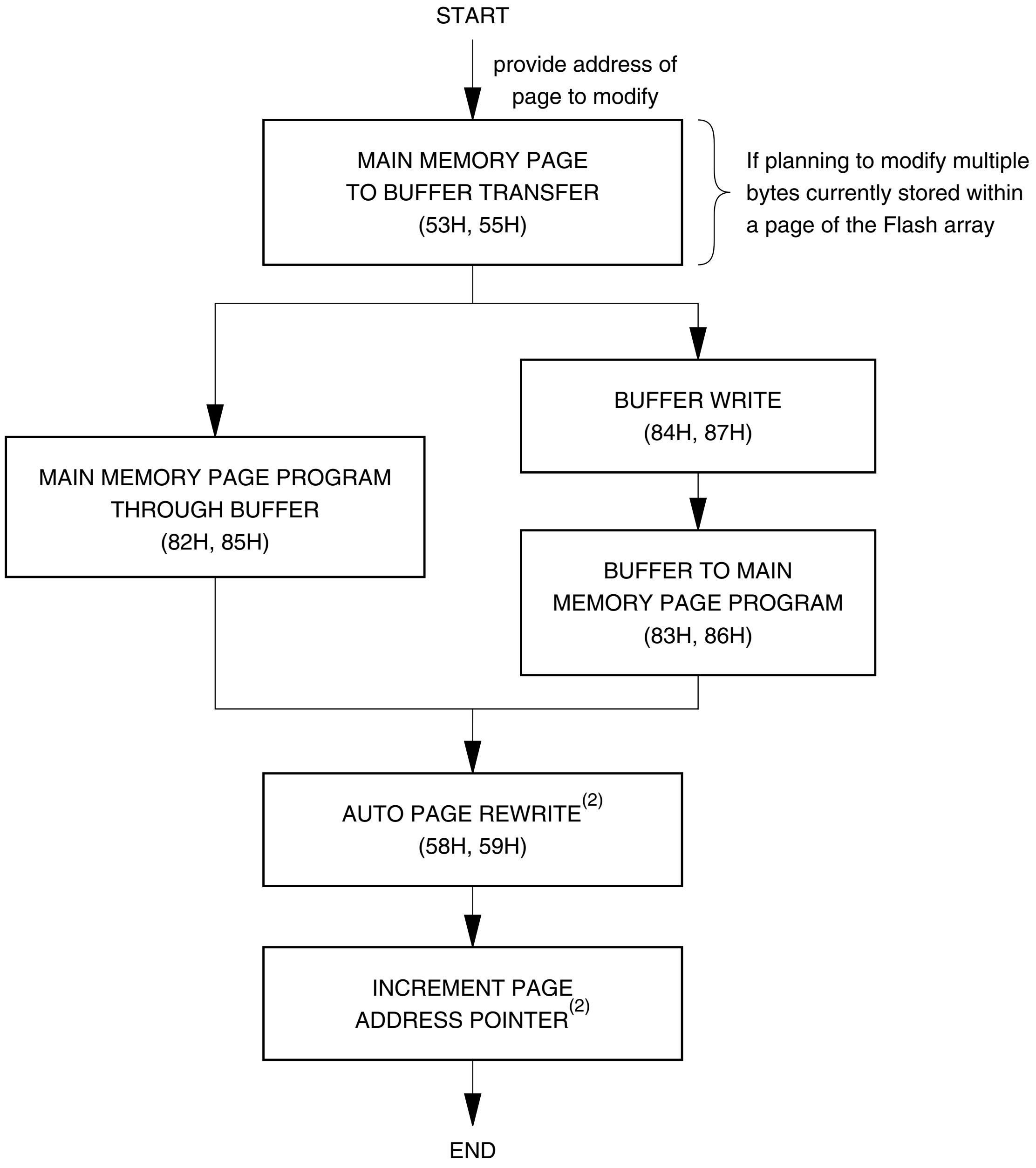
25. Auto Page Rewrite Flowchart

Figure 25-1. Algorithm for Programming or Reprogramming of the Entire Array Sequentially



Notes:
1. This type of algorithm is used for applications in which the entire array is programmed sequentially, filling the array page-by-page.
2. A page can be written using either a Main Memory Page Program operation or a Buffer Write operation followed by a Buffer to Main Memory Page Program operation.
3. The algorithm above shows the programming of a single page. The algorithm will be repeated sequentially for each page within the entire array.

Figure 25-2. Algorithm for Randomly Modifying Data


START
provide address of page to modify
MAIN MEMORY PAGE TO BUFFER TRANSFER (53H, 55H)
If planning to modify multiple bytes currently stored within a page of the Flash array
BUFFER WRITE (84H, 87H)
MAIN MEMORY PAGE PROGRAM THROUGH BUFFER (82H, 85H)
BUFFER TO MAIN MEMORY PAGE PROGRAM (83H, 86H)
AUTO PAGE REWRITE(2) (58H, 59H)
INCREMENT PAGE ADDRESS POINTER(2)
END

Notes:
1. To preserve data integrity, each page of a DataFlash sector must be updated/rewritten at least once within every 10,000 cumulative page erase and program operations.
2. A Page Address Pointer must be maintained to indicate which page is to be rewritten. The Auto Page Rewrite command must use the address specified by the Page Address Pointer.
3. Other algorithms can be used to rewrite portions of the Flash array. Low-power applications may choose to wait until 10,000 cumulative page erase and program operations have accumulated before rewriting all pages of the sector. See application note AN-4 (“Using Atmel’s Serial DataFlash”) for more details.

26. Ordering Information

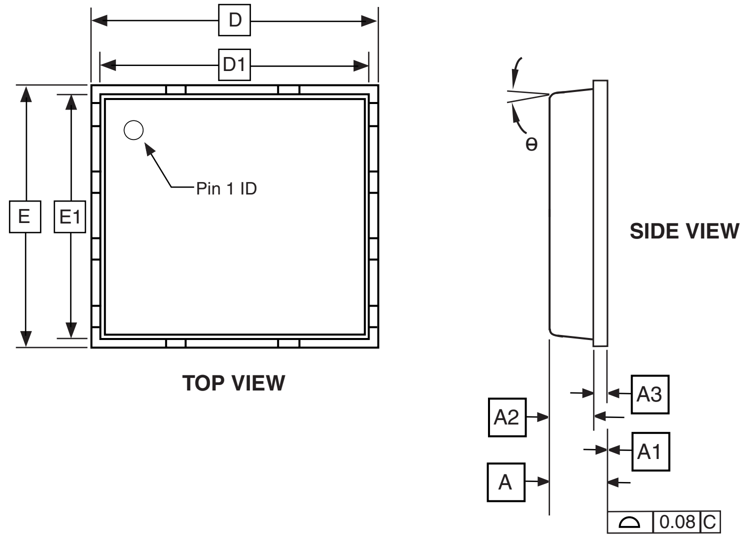
26.1 Green Package Options (Pb/Halide-free/RoHS Compliant)

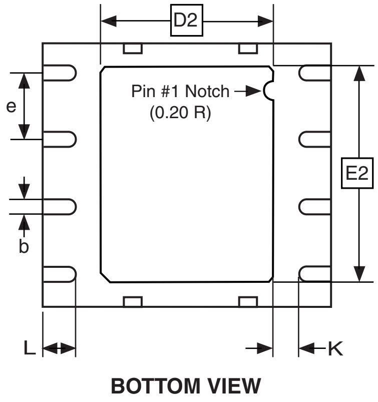
f_{SCK} (MHz)	I_{CC} (mA)		Ordering Code	Package	Operation Range
	Active	Standby			
66	15	0.05	AT45DB321D-MU	8M1-A	Industrial (-40° C to 85° C)
			AT45DB321D-SU	8S2	
			AT45DB321D-TU	28T	
			AT45DB321D-MWU	8MW	

Package Type	
8M1-A	8-contact, 6 mm x 5 mm, Very Thin Micro Lead-frame Package (MLF)
8MW	8-contact, 8 mm x 6 mm, Very Thin Micro Lead-frame Package (MLF)
8S2	8-lead, 0.209" wide, Plastic Gull Wing Small Outline Package (EIAJ SOIC)
28T	28-lead, 8 mm x 13.4 mm, Plastic Thin Small Outline Package, Type I (TSOP)

27. Packaging Information

27.1 8M1-A – MLF





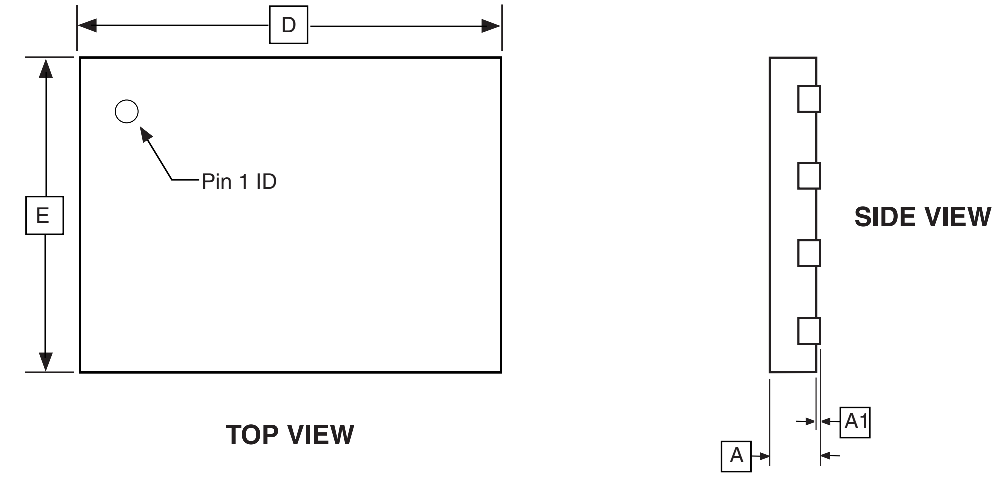
COMMON DIMENSIONS
(Unit of Measure = mm)

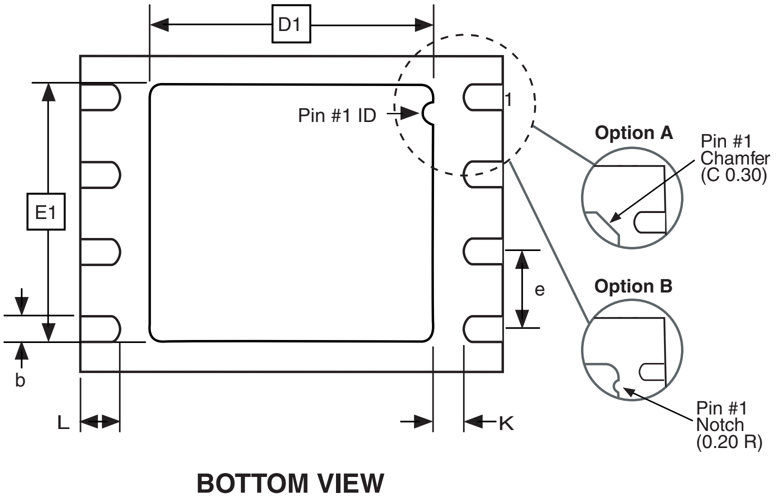
SYMBOL	MIN	NOM	MAX	NOTE
A	–	0.85	1.00	
A1	–	–	0.05	
A2		0.65 TYP		
A3		0.20 TYP		
b	0.35	0.40	0.48	
D		6.00 TYP		
D1		5.75 TYP		
D2	3.20	3.40	3.60	
E		5.00 TYP		
E1		4.75 TYP		
E2	3.80	4.00	4.20	
e		1.27		
L	0.50	0.60	0.75	
θ			12°	
K		1.30 REF		

12/6/04

	TITLE	DRAWING NO.	REV.
2325 Orchard Parkway San Jose, CA 95131	**8M1-A,** 8-lead, 6 x 5 x 1.00 mm Body, Very Thin Dual Flat Package No Lead (MLF)	8M1-A	A

27.2 8MW – MLF





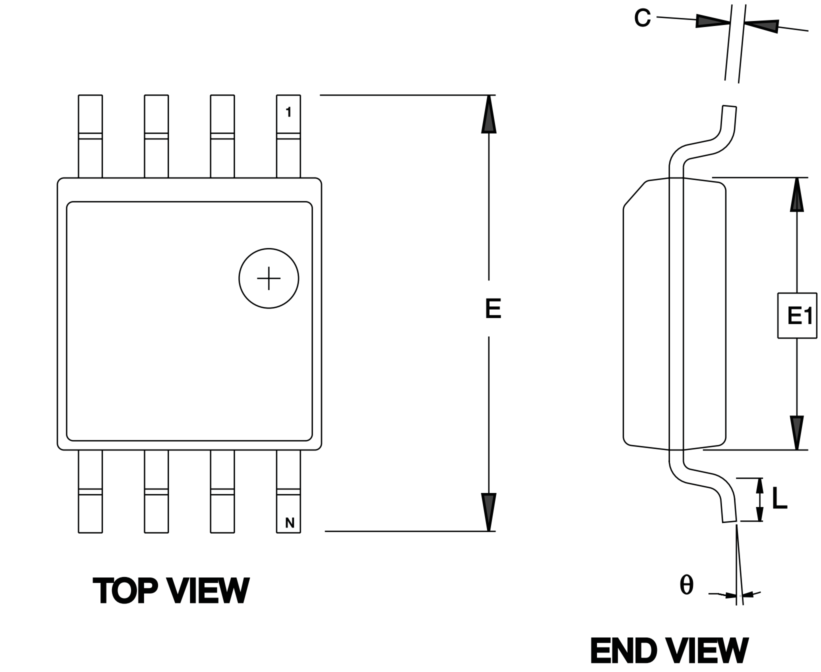
COMMON DIMENSIONS
(Unit of Measure = mm)

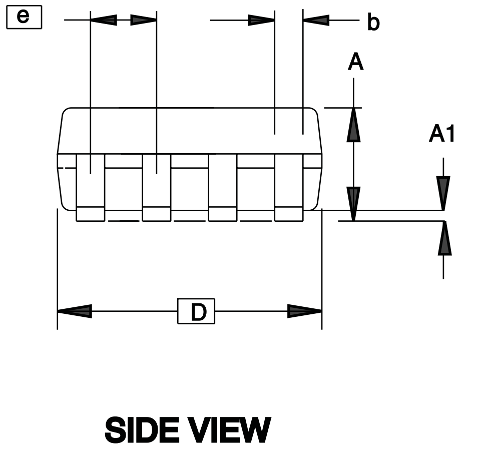
SYMBOL	MIN	NOM	MAX	NOTE
A	–	–	1.00	
A1	–	–	0.05	
b	0.35	0.40	0.48	
D	7.90	8.00	8.10	
D1	6.30	6.40	6.50	
E	5.90	6.00	6.10	
E1	4.70	4.80	4.90	
e		1.27		
L	0.45	0.50	0.55	
K		0.30 REF		

5/25/06

2325 Orchard Parkway San Jose, CA 95131	**TITLE** **8MW,** 8-pad, 8 x 6 x 1.0 mm Body, Very Thin Dual Flat Package No Lead (MLF)	**DRAWING NO.** 8MW	**REV.** B

27.3 8S2 – EIAJ SOIC





COMMON DIMENSIONS
(Unit of Measure = mm)

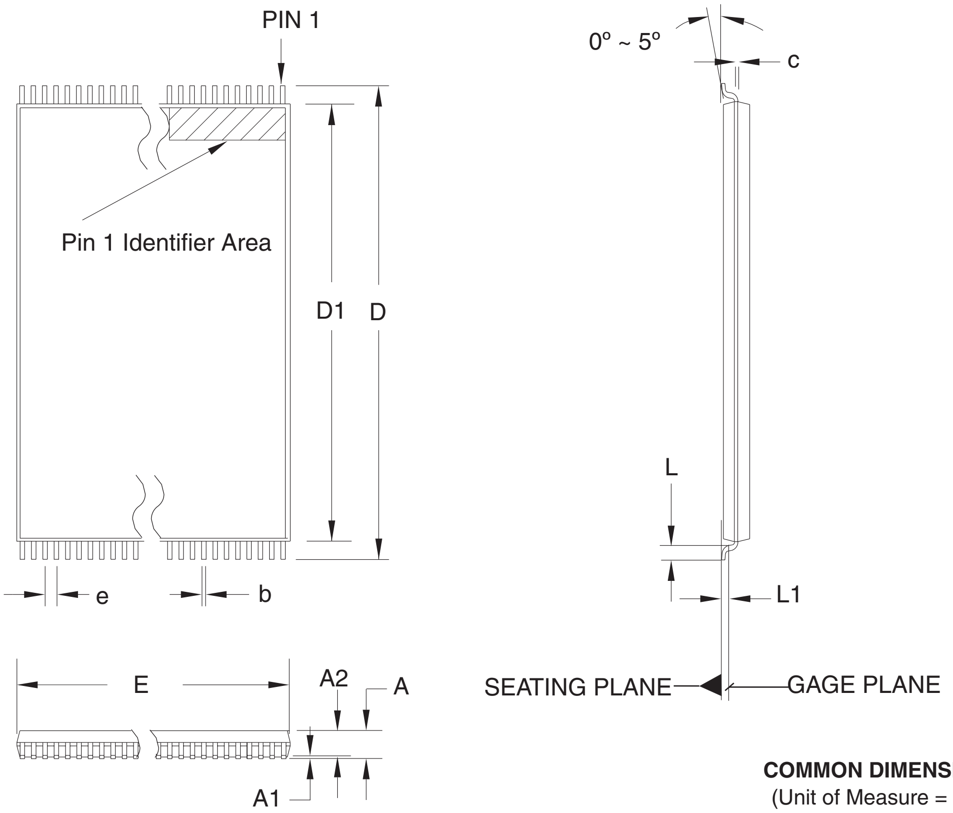
SYMBOL	MIN	NOM	MAX	NOTE
A	1.70		2.16	
A1	0.05		0.25	
b	0.35		0.48	5
C	0.15		0.35	5
D	5.13		5.35	
E1	5.18		5.40	2, 3
E	7.70		8.26	
L	0.51		0.85	
θ	0°		8°	
e	1.27 BSC			4

Notes: 1. This drawing is for general information only; refer to EIAJ Drawing EDR-7320 for additional information.
2. Mismatch of the upper and lower dies and resin burrs are not included.
3. It is recommended that upper and lower cavities be equal. If they are different, the larger dimension shall be regarded.
4. Determines the true geometric position.
5. Values b,C apply to plated terminal. The standard thickness of the plating layer shall measure between 0.007 to .021 mm.

4/7/06

	TITLE	DRAWING NO.	REV.
2325 Orchard Parkway San Jose, CA 95131	**8S2**, 8-lead, 0.209" Body, Plastic Small Outline Package (EIAJ)	8S2	D

27.4 28T – TSOP, Type 1



COMMON DIMENSIONS
(Unit of Measure = mm)

SYMBOL	MIN	NOM	MAX	NOTE
A	–	–	1.20	
A1	0.05	–	0.15	
A2	0.90	1.00	1.05	
D	13.20	13.40	13.60	
D1	11.70	11.80	11.90	Note 2
E	7.90	8.00	8.10	Note 2
L	0.50	0.60	0.70	
L1	0.25 BASIC			
b	0.17	0.22	0.27	
c	0.10	–	0.21	
e	0.55 BASIC			

Notes:
1. This package conforms to JEDEC reference MO-183.
2. Dimensions D1 and E do not include mold protrusion. Allowable protrusion on E is 0.15 mm per side and on D1 is 0.25 mm per side.
3. Lead coplanarity is 0.10 mm maximum.

12/06/02

	TITLE	DRAWING NO.	REV.
2325 Orchard Parkway San Jose, CA 95131	**28T**, 28-lead (8 x 13.4 mm) Plastic Thin Small Outline Package, Type I (TSOP)	28T	C

28. Revision History

Revision Level – Release Date	History
A – November 2005	Initial Release
B – January 2006	Added 5 x 6 mm MLF package. Added text, in “Programming the Configuration Register”, to indicate that power cycling is required to switch to “power of 2” page size after the opcode enable has been executed. Corrected typographical error regarding the opcode for chip erase in “Program and Erase Commands” table.
C – March 2006	Added Preliminary. Changed the sector size from 256K bytes to 64K bytes. Added the “Legacy Commands” table.
D – April 2006	Added 6 x 8 mm MLF package. Changed the sector size of 0a and 0b to 8 pages and 120 pages respectively. Changed the Product Version Code to 00001.
E – July 2006	Corrected typographical errors.
F – August 2006	Added errata regarding Chip Erase. Added AT45DB321D-SU to ordering information and corresponding 8S2 package.
G – September 2006	Removed “not recommended for new designs” note from ordering information for 8MW package.
H – February 2007	Added AT45DB321D-CNU to ordering information and corresponding 8CN3 package. Removed “not recommended for new designs” comment from 8MW package drawing.
I – August 2007	Added additional text to “power of 2” binary page size option. Changed t_{VSCL} from 50 µs to 70 µs. Changed t_{RDPD} from 30 µs to 35 µs. Changed t_{XFR} and t_{COMP} values from 400 µs to 200 µs. Removed AT45DB321D-CNU from ordering information and corresponding 8CN3 package.

29. Errata

29.1 Chip Erase

29.1.1 Issue

In a certain percentage of units, the Chip Erase feature may not function correctly and may adversely affect device operation. Therefore, it is recommended that the Chip Erase commands (opcodes C7H, 94H, 80H, and 9AH) not be used.

29.1.2 Workaround

Use Block Erase (opcode 50H) as an alternative. The Block Erase function is not affected by the Chip Erase issue.

29.1.3 Resolution

The Chip Erase feature may be fixed with a new revision of the device. Please contact Atmel for the estimated availability of devices with the fix.

Headquarters

Atmel Corporation
2325 Orchard Parkway
San Jose, CA 95131
USA
Tel: 1(408) 441-0311
Fax: 1(408) 487-2600

International

Atmel Asia
Room 1219
Chinachem Golden Plaza
77 Mody Road Tsimshatsui
East Kowloon
Hong Kong
Tel: (852) 2721-9778
Fax: (852) 2722-1369

Atmel Europe
Le Krebs
8, Rue Jean-Pierre Timbaud
BP 309
78054 Saint-Quentin-en-
Yvelines Cedex
France
Tel: (33) 1-30-60-70-00
Fax: (33) 1-30-60-71-11

Atmel Japan
9F, Tonetsu Shinkawa Bldg.
1-24-8 Shinkawa
Chuo-ku, Tokyo 104-0033
Japan
Tel: (81) 3-3523-3551
Fax: (81) 3-3523-7581

Product Contact

Web Site
www.atmel.com

Technical Support
dataflash@atmel.com

Sales Contact
www.atmel.com/contacts

Literature Requests
www.atmel.com/literature

Disclaimer: The information in this document is provided in connection with Atmel products. No license, express or implied, by estoppel or otherwise, to any intellectual property right is granted by this document or in connection with the sale of Atmel products. **EXCEPT AS SET FORTH IN ATMEL'S TERMS AND CONDITIONS OF SALE LOCATED ON ATMEL'S WEB SITE, ATMEL ASSUMES NO LIABILITY WHATSOEVER AND DISCLAIMS ANY EXPRESS, IMPLIED OR STATUTORY WARRANTY RELATING TO ITS PRODUCTS INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. IN NO EVENT SHALL ATMEL BE LIABLE FOR ANY DIRECT, INDIRECT, CONSEQUENTIAL, PUNITIVE, SPECIAL OR INCIDENTAL DAMAGES (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF PROFITS, BUSINESS INTERRUPTION, OR LOSS OF INFORMATION) ARISING OUT OF THE USE OR INABILITY TO USE THIS DOCUMENT, EVEN IF ATMEL HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.** Atmel makes no representations or warranties with respect to the accuracy or completeness of the contents of this document and reserves the right to make changes to specifications and product descriptions at any time without notice. Atmel does not make any commitment to update the information contained herein. Unless specifically provided otherwise, Atmel products are not suitable for, and shall not be used in, automotive applications. Atmel's products are not intended, authorized, or warranted for use as components in applications intended to support or sustain life.

Features

- High-performance, Low-power AVR® 8-bit Microcontroller
- Advanced RISC Architecture
  - 131 Powerful Instructions – Most Single-clock Cycle Execution
  - 32 x 8 General Purpose Working Registers
  - Fully Static Operation
  - Up to 20 MIPS Throughput at 20 MHz
  - On-chip 2-cycle Multiplier
- High Endurance Non-volatile Memory segments
  - 16/32/64K Bytes of In-System Self-programmable Flash program memory
  - 512B/1K/2K Bytes EEPROM
  - 1/2/4K Bytes Internal SRAM
  - Write/Erase Cycles: 10,000 Flash/ 100,000 EEPROM
  - Data retention: 20 years at 85°C/100 years at 25°C[1]
  - Optional Boot Code Section with Independent Lock Bits
    In-System Programming by On-chip Boot Program
    True Read-While-Write Operation
  - Programming Lock for Software Security
- JTAG (IEEE std. 1149.1 Compliant) Interface
  - Boundary-scan Capabilities According to the JTAG Standard
  - Extensive On-chip Debug Support
  - Programming of Flash, EEPROM, Fuses, and Lock Bits through the JTAG Interface
- Peripheral Features
  - Two 8-bit Timer/Counters with Separate Prescalers and Compare Modes
  - One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode
  - Real Time Counter with Separate Oscillator
  - Six PWM Channels
  - 8-channel, 10-bit ADC
    Differential mode with selectable gain at 1x, 10x or 200x
  - Byte-oriented Two-wire Serial Interface
  - Two Programmable Serial USART
  - Master/Slave SPI Serial Interface
  - Programmable Watchdog Timer with Separate On-chip Oscillator
  - On-chip Analog Comparator
  - Interrupt and Wake-up on Pin Change
- Special Microcontroller Features
  - Power-on Reset and Programmable Brown-out Detection
  - Internal Calibrated RC Oscillator
  - External and Internal Interrupt Sources
  - Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby and Extended Standby
- I/O and Packages
  - 32 Programmable I/O Lines
  - 40-pin PDIP, 44-lead TQFP, 44-pad VQFN/QFN/MLF (ATmega164P/324P/644P)
  - 44-pad DRQFN (ATmega164P)
- Operating Voltages
  - 1.8 - 5.5V for ATmega164P/324P/644PV
  - 2.7 - 5.5V for ATmega164P/324P/644P
- Speed Grades
  - ATmega164P/324P/644PV: 0 - 4MHz @ 1.8 - 5.5V, 0 - 10MHz @ 2.7 - 5.5V
  - ATmega164P/324P/644P: 0 - 10MHz @ 2.7 - 5.5V, 0 - 20MHz @ 4.5 - 5.5V
- Power Consumption at 1 MHz, 1.8V, 25°C for ATmega164P/324P/644PV
  - Active: 0.4 mA
  - Power-down Mode: 0.1µA
  - Power-save Mode: 0.6µA (Including 32 kHz RTC)

Note: 1. See "Data Retention" on page 8.

8-bit AVR® Microcontroller with 16/32/64K Bytes In-System Programmable Flash

ATmega164P/V ATmega324P/V ATmega644P/V

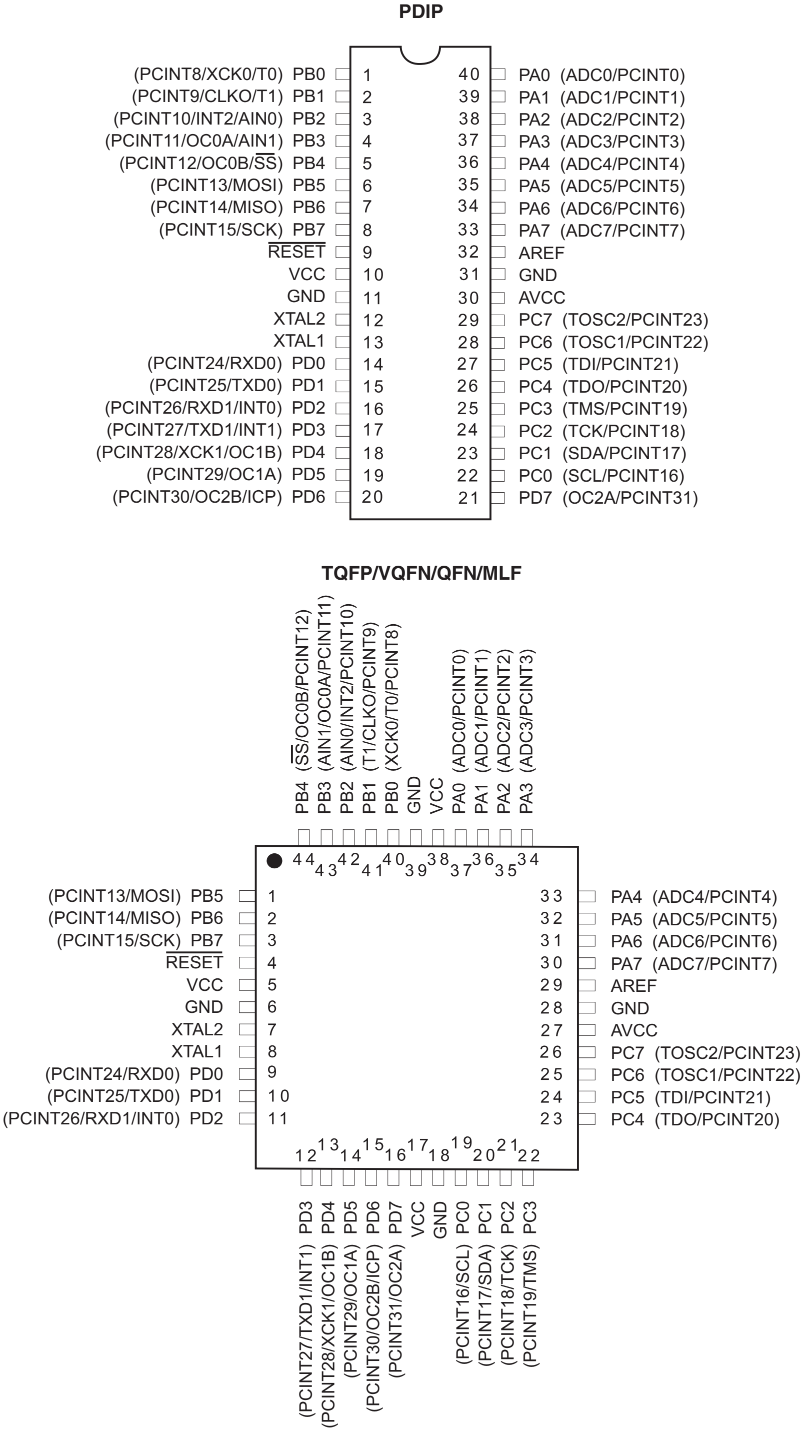
Summary

8011MS–AVR–08/09

1. Pin Configurations

1.1 Pinout - PDIP/TQFP/VQFN/QFN/MLF

Figure 1-1. Pinout ATmega164P/324P/644P



PDIP

TQFP/VQFN/QFN/MLF

Note: The large center pad underneath the VQFN/QFN/MLF package should be soldered to ground on the board to ensure good mechanical stability.

1.2 Pinout - DRQFN

Figure 1-2. DRQFN - Pinout ATmega164P

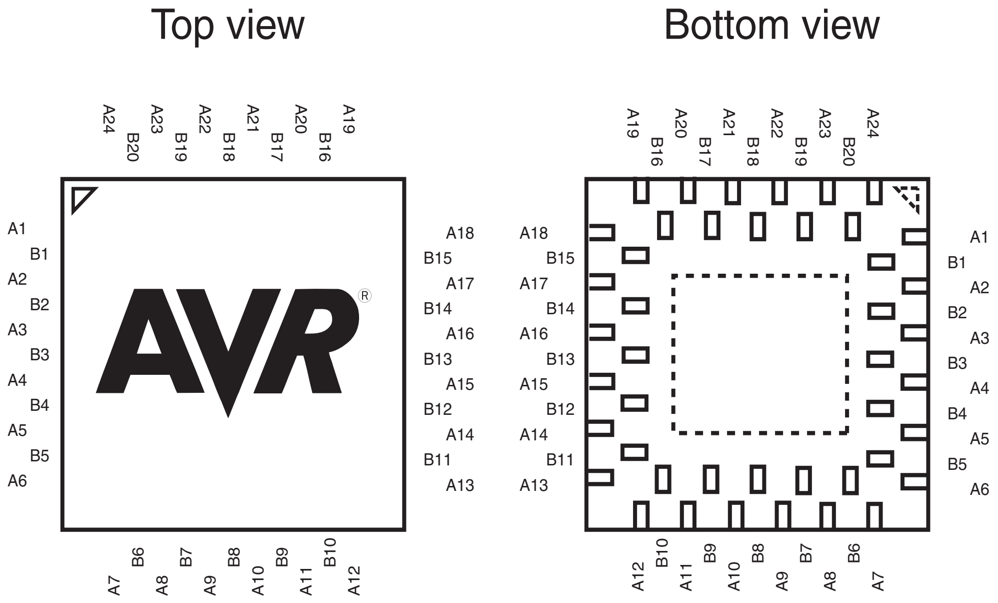

Top view
Bottom view
A24 B20 A23 B19 A22 B18 A21 B17 A20 B16 A19
A1 B1 A2 B2 A3 B3 A4 B4 A5 B5 A6
A18 B15 A17 B14 A16 B13 A15 B12 A14 B11 A13
A7 B6 A8 B7 A9 B8 A10 B9 A11 B10 A12
AVR
A19 B16 A20 B17 A21 B18 A22 B19 A23 B20 A24
A18 B15 A17 B14 A16 B13 A15 B12 A14 B11 A13
A1 B1 A2 B2 A3 B3 A4 B4 A5 B5 A6
A12 B10 A11 B9 A10 B8 A9 B7 A8 B6 A7

Table 1-1. DRQFN - Pinout ATmega164P/324P

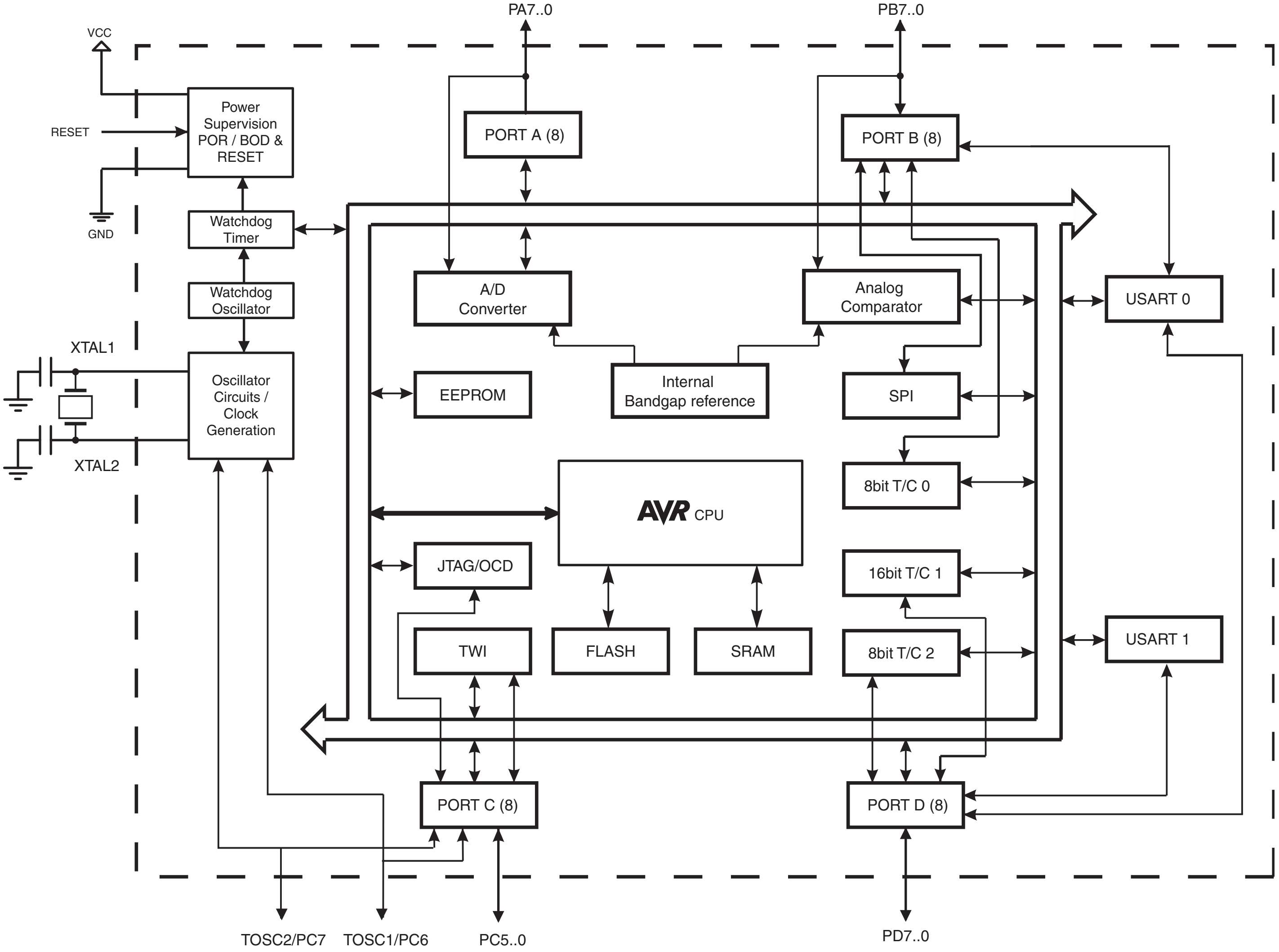
A1	PB5	**A7**	PD3	**A13**	PC4	**A19**	PA3
B1	PB6	**B6**	PD4	**B11**	PC5	**B16**	PA2
A2	PB7	**A8**	PD5	**A14**	PC6	**A20**	PA1
B2	$\overline{\text{RESET}}$	**B7**	PD6	**B12**	PC7	**B17**	PA0
A3	VCC	**A9**	PD7	**A15**	AVCC	**A21**	VCC
B3	GND	**B8**	VCC	**B13**	GND	**B18**	GND
A4	XTAL2	**A10**	GND	**A16**	AREF	**A22**	PB0
B4	XTAL1	**B9**	PC0	**B14**	PA7	**B19**	PB1
A5	PD0	**A11**	PC1	**A17**	PA6	**A23**	PB2
B5	PD1	**B10**	PC2	**B15**	PA5	**B20**	PB3
A6	PD2	**A12**	PC3	**A18**	PA4	**A24**	PB4

2. Overview

The ATmega164P/324P/644P is a low-power CMOS 8-bit microcontroller based on the AVR enhanced RISC architecture. By executing powerful instructions in a single clock cycle, the ATmega164P/324P/644P achieves throughputs approaching 1 MIPS per MHz allowing the system designer to optimize power consumption versus processing speed.

2.1 Block Diagram

Figure 2-1. Block Diagram



The AVR core combines a rich instruction set with 32 general purpose working registers. All the 32 registers are directly connected to the Arithmetic Logic Unit (ALU), allowing two independent registers to be accessed in one single instruction executed in one clock cycle. The resulting architecture is more code efficient while achieving throughputs up to ten times faster than conventional CISC microcontrollers.

The ATmega164P/324P/644P provides the following features: 16/32/64K bytes of In-System Programmable Flash with Read-While-Write capabilities, 512B/1K/2K bytes EEPROM, 1/2/4K bytes SRAM, 32 general purpose I/O lines, 32 general purpose working registers, Real Time Counter (RTC), three flexible Timer/Counters with compare modes and PWM, 2 USARTs, a byte oriented 2-wire Serial Interface, a 8-channel, 10-bit ADC with optional differential input stage with programmable gain, programmable Watchdog Timer with Internal Oscillator, an SPI serial port, IEEE std. 1149.1 compliant JTAG test interface, also used for accessing the On-chip Debug system and programming and six software selectable power saving modes. The Idle mode stops the CPU while allowing the SRAM, Timer/Counters, SPI port, and interrupt system to continue functioning. The Power-down mode saves the register contents but freezes the Oscillator, disabling all other chip functions until the next interrupt or Hardware Reset. In Power-save mode, the asynchronous timer continues to run, allowing the user to maintain a timer base while the rest of the device is sleeping. The ADC Noise Reduction mode stops the CPU and all I/O modules except Asynchronous Timer and ADC, to minimize switching noise during ADC conversions. In Standby mode, the Crystal/Resonator Oscillator is running while the rest of the device is sleeping. This allows very fast start-up combined with low power consumption. In Extended Standby mode, both the main Oscillator and the Asynchronous Timer continue to run.

The device is manufactured using Atmel's high-density nonvolatile memory technology. The On-chip ISP Flash allows the program memory to be reprogrammed in-system through an SPI serial interface, by a conventional nonvolatile memory programmer, or by an On-chip Boot program running on the AVR core. The boot program can use any interface to download the application program in the application Flash memory. Software in the Boot Flash section will continue to run while the Application Flash section is updated, providing true Read-While-Write operation. By combining an 8-bit RISC CPU with In-System Self-Programmable Flash on a monolithic chip, the Atmel ATmega164P/324P/644P is a powerful microcontroller that provides a highly flexible and cost effective solution to many embedded control applications.

The ATmega164P/324P/644P AVR is supported with a full suite of program and system development tools including: C compilers, macro assemblers, program debugger/simulators, in-circuit emulators, and evaluation kits.

2.2 Comparison Between ATmega164P, ATmega324P and ATmega644P

Table 2-1. Differences between ATmega164P and ATmega644P

Device	Flash	EEPROM	RAM
ATmega164P	16 Kbyte	512 Bytes	1 Kbyte
ATmega324P	32 Kbyte	1 Kbyte	2 Kbyte
ATmega644P	64 Kbyte	2 Kbyte	4 Kbyte

2.3 Pin Descriptions

2.3.1 VCC

Digital supply voltage.

2.3.2 GND

Ground.

2.3.3 Port A (PA7:PA0)

Port A serves as analog inputs to the Analog-to-digital Converter.

Port A also serves as an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port A output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port A pins that are externally pulled low will source current if the pull-up resistors are activated. The Port A pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port A also serves the functions of various special features of the ATmega164P/324P/644P as listed on page 81.

2.3.4 Port B (PB7:PB0)

Port B is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port B output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port B pins that are externally pulled low will source current if the pull-up resistors are activated. The Port B pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port B also serves the functions of various special features of the ATmega164P/324P/644P as listed on page 83.

2.3.5 Port C (PC7:PC0)

Port C is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port C output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port C pins that are externally pulled low will source current if the pull-up resistors are activated. The Port C pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port C also serves the functions of the JTAG interface, along with special features of the ATmega164P/324P/644P as listed on page 86.

2.3.6 Port D (PD7:PD0)

Port D is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port D output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port D pins that are externally pulled low will source current if the pull-up resistors are activated. The Port D pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port D also serves the functions of various special features of the ATmega164P/324P/644P as listed on page 88.

2.3.7 $\overline{\text{RESET}}$

Reset input. A low level on this pin for longer than the minimum pulse length will generate a reset, even if the clock is not running. The minimum pulse length is given in ”System and Reset Characteristics” on page 332. Shorter pulses are not guaranteed to generate a reset.

2.3.8 XTAL1

Input to the inverting Oscillator amplifier and input to the internal clock operating circuit.

2.3.9 XTAL2

Output from the inverting Oscillator amplifier.

2.3.10 AVCC

AVCC is the supply voltage pin for Port A and the Analog-to-digital Converter. It should be externally connected to V_{CC}, even if the ADC is not used. If the ADC is used, it should be connected to V_{CC} through a low-pass filter.

2.3.11 AREF

This is the analog reference pin for the Analog-to-digital Converter.

3. About

3.1 Resources

A comprehensive set of development tools, application notes and datasheetsare available for download on http://www.atmel.com/avr.

3.2 About Code Examples

This documentation contains simple code examples that briefly show how to use various parts of the device. Be aware that not all C compiler vendors include bit definitions in the header files and interrupt handling in C is compiler dependent. Please confirm with the C compiler documentation for more details.

The code examples assume that the part specific header file is included before compilation. For I/O registers located in extended I/O map, "IN", "OUT", "SBIS", "SBIC", "CBI", and "SBI" instructions must be replaced with instructions that allow access to extended I/O. Typically "LDS" and "STS" combined with "SBRS", "SBRC", "SBR", and "CBR".

3.3 Data Retention

Reliability Qualification results show that the projected data retention failure rate is much less than 1 PPM over 20 years at 85°C or 100 years at 25°C.

4. Register Summary

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Page
(0xFF)	Reserved	-	-	-	-		-	-	-	
(0xFE)	Reserved	-	-	-	-	-	-	-	-	
(0xFD)	Reserved	-	-	-	-	-	-	-	-	
(0xFC)	Reserved	-	-	-	-	-	-	-	-	
(0xFB)	Reserved	-	-	-	-		-	-	-	
(0xFA)	Reserved	-	-	-	-	-	-	-	-	
(0xF9)	Reserved	-	-	-	-		-	-	-	
(0xF8)	Reserved	-	-	-	-	-	-	-	-	
(0xF7)	Reserved	-	-	-	-	-	-	-	-	
(0xF6)	Reserved	-	-	-	-	-	-	-	-	
(0xF5)	Reserved	-	-	-	-		-	-	-	
(0xF4)	Reserved	-	-	-	-	-	-	-	-	
(0xF3)	Reserved	-	-	-	-	-	-	-	-	
(0xF2)	Reserved	-	-	-	-	-	-	-	-	
(0xF1)	Reserved	-	-	-	-		-	-	-	
(0xF0)	Reserved	-	-	-	-	-	-	-	-	
(0xEF)	Reserved	-	-	-	-		-	-	-	
(0xEE)	Reserved	-	-	-	-	-	-	-	-	
(0xED)	Reserved	-	-	-	-	-	-	-	-	
(0xEC)	Reserved	-	-	-	-	-	-	-	-	
(0xEB)	Reserved	-	-	-	-		-	-	-	
(0xEA)	Reserved	-	-	-	-	-	-	-	-	
(0xE9)	Reserved	-	-	-	-	-	-	-	-	
(0xE8)	Reserved	-	-	-	-	-	-	-	-	
(0xE7)	Reserved	-	-	-	-		-	-	-	
(0xE6)	Reserved	-	-	-	-	-	-	-	-	
(0xE5)	Reserved	-	-	-	-	-	-	-	-	
(0xE4)	Reserved	-	-	-	-	-	-	-	-	
(0xE3)	Reserved	-	-	-	-		-	-	-	
(0xE2)	Reserved	-	-	-	-	-	-	-	-	
(0xE1)	Reserved	-	-	-	-		-	-	-	
(0xE0)	Reserved	-	-	-	-		-	-	-	
(0xDF)	Reserved	-	-	-	-	-	-	-	-	
(0xDE)	Reserved	-	-	-	-	-	-	-	-	
(0xDD)	Reserved	-	-	-	-	-	-	-	-	
(0xDC)	Reserved	-	-	-	-		-	-	-	
(0xDB)	Reserved	-	-	-	-	-	-	-	-	
(0xDA)	Reserved	-	-	-	-	-	-	-	-	
(0xD9)	Reserved	-	-	-	-	-	-	-	-	
(0xD8)	Reserved	-	-	-	-	-	-	-	-	
(0xD7)	Reserved	-	-	-	-	-	-	-	-	
(0xD6)	Reserved	-	-	-	-	-	-	-	-	
(0xD5)	Reserved	-	-	-	-	-	-	-	-	
(0xD4)	Reserved	-	-	-	-	-	-	-	-	
(0xD3)	Reserved	-	-	-	-	-	-	-	-	
(0xD2)	Reserved	-	-	-	-	-	-	-	-	
(0xD1)	Reserved	-	-	-	-	-	-	-	-	
(0xD0)	Reserved	-	-	-	-	-	-	-	-	
(0xCF)	Reserved	-	-	-	-	-	-	-	-	
(0xCE)	UDR1	USART1 I/O Data Register								190
(0xCD)	UBRR1H	-	-	-	-	USART1 Baud Rate Register High Byte				194/207
(0xCC)	UBRR1L	USART1 Baud Rate Register Low Byte								194/207
(0xCB)	Reserved	-	-	-	-	-	-	-	-	
(0xCA)	UCSR1C	UMSEL11	UMSEL10	-	-	-	UDORD1	UCPHA1	UCPOL1	192/206
(0xC9)	UCSR1B	RXCIE1	TXCIE1	UDRIE1	RXEN1	TXEN1	UCSZ12	RXB81	TXB81	191/205
(0xC8)	UCSR1A	RXC1	TXC1	UDRE1	FE1	DOR1	UPE1	U2X1	MPCM1	190/205
(0xC7)	Reserved	-	-	-	-	-	-	-	-	
(0xC6)	UDR0	USART0 I/O Data Register								190
(0xC5)	UBRR0H	-	-	-	-	USART0 Baud Rate Register High Byte				194/207
(0xC4)	UBRR0L	USART0 Baud Rate Register Low Byte								194/207
(0xC3)	Reserved	-	-	-	-	-	-	-	-	
(0xC2)	UCSR0C	UMSEL01	UMSEL00	-	-	-	UDORD0	UCPHA0	UCPOL0	192/206
(0xC1)	UCSR0B	RXCIE0	TXCIE0	UDRIE0	RXEN0	TXEN0	UCSZ02	RXB80	TXB80	191/205

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Page
(0xC0)	UCSR0A	RXC0	TXC0	UDRE0	FE0	DOR0	UPE0	U2X0	MPCM0	190/205
(0xBF)	Reserved	-	-	-	-	-	-	-	-	
(0xBE)	Reserved	-	-	-	-	-	-	-	-	
(0xBD)	TWAMR	TWAM6	TWAM5	TWAM4	TWAM3	TWAM2	TWAM1	TWAM0	-	236
(0xBC)	TWCR	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	-	TWIE	233
(0xBB)	TWDR	2-wire Serial Interface Data Register								235
(0xBA)	TWAR	TWA6	TWA5	TWA4	TWA3	TWA2	TWA1	TWA0	TWGCE	236
(0xB9)	TWSR	TWS7	TWS6	TWS5	TWS4	TWS3	-	TWPS1	TWPS0	235
(0xB8)	TWBR	2-wire Serial Interface Bit Rate Register								233
(0xB7)	Reserved	-	-	-	-	-	-	-	-	
(0xB6)	ASSR	-	EXCLK	AS2	TCN2UB	OCR2AUB	OCR2BUB	TCR2AUB	TCR2BUB	158
(0xB5)	Reserved	-	-	-	-	-	-	-	-	
(0xB4)	OCR2B	Timer/Counter2 Output Compare Register B								158
(0xB3)	OCR2A	Timer/Counter2 Output Compare Register A								158
(0xB2)	TCNT2	Timer/Counter2 (8 Bit)								157
(0xB1)	TCCR2B	FOC2A	FOC2B	-	-	WGM22	CS22	CS21	CS20	156
(0xB0)	TCCR2A	COM2A1	COM2A0	COM2B1	COM2B0	-	-	WGM21	WGM20	153
(0xAF)	Reserved	-	-	-	-	-	-	-	-	
(0xAE)	Reserved	-	-	-	-	-	-	-	-	
(0xAD)	Reserved	-	-	-	-	-	-	-	-	
(0xAC)	Reserved	-	-	-	-	-	-	-	-	
(0xAB)	Reserved	-	-	-	-	-	-	-	-	
(0xAA)	Reserved	-	-	-	-	-	-	-	-	
(0xA9)	Reserved	-	-	-	-	-	-	-	-	
(0xA8)	Reserved	-	-	-	-	-	-	-	-	
(0xA7)	Reserved	-	-	-	-	-	-	-	-	
(0xA6)	Reserved	-	-	-	-	-	-	-	-	
(0xA5)	Reserved	-	-	-	-	-	-	-	-	
(0xA4)	Reserved	-	-	-	-	-	-	-	-	
(0xA3)	Reserved	-	-	-	-	-	-	-	-	
(0xA2)	Reserved	-	-	-	-	-	-	-	-	
(0xA1)	Reserved	-	-	-	-	-	-	-	-	
(0xA0)	Reserved	-	-	-	-	-	-	-	-	
(0x9F)	Reserved	-	-	-	-	-	-	-	-	
(0x9E)	Reserved	-	-	-	-	-	-	-	-	
(0x9D)	Reserved	-	-	-	-	-	-	-	-	
(0x9C)	Reserved	-	-	-	-	-	-	-	-	
(0x9B)	Reserved	-	-	-	-	-	-	-	-	
(0x9A)	Reserved	-	-	-	-	-	-	-	-	
(0x99)	Reserved	-	-	-	-	-	-	-	-	
(0x98)	Reserved	-	-	-	-	-	-	-	-	
(0x97)	Reserved	-	-	-	-	-	-	-	-	
(0x96)	Reserved	-	-	-	-	-	-	-	-	
(0x95)	Reserved	-	-	-	-	-	-	-	-	
(0x94)	Reserved	-	-	-	-	-	-	-	-	
(0x93)	Reserved	-	-	-	-	-	-	-	-	
(0x92)	Reserved	-	-	-	-	-	-	-	-	
(0x91)	Reserved	-	-	-	-	-	-	-	-	
(0x90)	Reserved	-	-	-	-	-	-	-	-	
(0x8F)	Reserved	-	-	-	-	-	-	-	-	
(0x8E)	Reserved	-	-	-	-	-	-	-	-	
(0x8D)	Reserved	-	-	-	-	-	-	-	-	
(0x8C)	Reserved	-	-	-	-	-	-	-	-	
(0x8B)	OCR1BH	Timer/Counter1 - Output Compare Register B High Byte								137
(0x8A)	OCR1BL	Timer/Counter1 - Output Compare Register B Low Byte								137
(0x89)	OCR1AH	Timer/Counter1 - Output Compare Register A High Byte								137
(0x88)	OCR1AL	Timer/Counter1 - Output Compare Register A Low Byte								137
(0x87)	ICR1H	Timer/Counter1 - Input Capture Register High Byte								138
(0x86)	ICR1L	Timer/Counter1 - Input Capture Register Low Byte								138
(0x85)	TCNT1H	Timer/Counter1 - Counter Register High Byte								137
(0x84)	TCNT1L	Timer/Counter1 - Counter Register Low Byte								137
(0x83)	Reserved	-	-	-	-	-	-	-	-	
(0x82)	TCCR1C	FOC1A	FOC1B	-	-	-	-	-	-	136
(0x81)	TCCR1B	ICNC1	ICES1	-	WGM13	WGM12	CS12	CS11	CS10	135
(0x80)	TCCR1A	COM1A1	COM1A0	COM1B1	COM1B0	-	-	WGM11	WGM10	133
(0x7F)	DIDR1	-	-	-	-	-	-	AIN1D	AIN0D	240

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Page
(0x7E)	DIDR0	ADC7D	ADC6D	ADC5D	ADC4D	ADC3D	ADC2D	ADC1D	ADC0D	260
(0x7D)	Reserved	-	-	-	-	-	-	-	-	
(0x7C)	ADMUX	REFS1	REFS0	ADLAR	MUX4	MUX3	MUX2	MUX1	MUX0	256
(0x7B)	ADCSRB	-	ACME	-	-	-	ADTS2	ADTS1	ADTS0	239
(0x7A)	ADCSRA	ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0	258
(0x79)	ADCH	ADC Data Register High byte								259
(0x78)	ADCL	ADC Data Register Low byte								259
(0x77)	Reserved	-	-	-	-	-	-	-	-	
(0x76)	Reserved	-	-	-	-	-	-	-	-	
(0x75)	Reserved	-	-	-	-	-	-	-	-	
(0x74)	Reserved	-	-	-	-	-	-	-	-	
(0x73)	PCMSK3	PCINT31	PCINT30	PCINT29	PCINT28	PCINT27	PCINT26	PCINT25	PCINT24	71
(0x72)	Reserved	-	-	-	-	-	-	-	-	
(0x71)	Reserved	-	-	-	-	-	-	-	-	
(0x70)	TIMSK2	-	-	-	-	-	OCIE2B	OCIE2A	TOIE2	159
(0x6F)	TIMSK1	-	-	ICIE1	-	-	OCIE1B	OCIE1A	TOIE1	138
(0x6E)	TIMSK0	-	-	-	-	-	OCIE0B	OCIE0A	TOIE0	110
(0x6D)	PCMSK2	PCINT23	PCINT22	PCINT21	PCINT20	PCINT19	PCINT18	PCINT17	PCINT16	71
(0x6C)	PCMSK1	PCINT15	PCINT14	PCINT13	PCINT12	PCINT11	PCINT10	PCINT9	PCINT8	71
(0x6B)	PCMSK0	PCINT7	PCINT6	PCINT5	PCINT4	PCINT3	PCINT2	PCINT1	PCINT0	72
(0x6A)	Reserved	-	-	-	-	-	-	-	-	
(0x69)	EICRA	-	-	ISC21	ISC20	ISC11	ISC10	ISC01	ISC00	68
(0x68)	PCICR	-	-	-	-	PCIE3	PCIE2	PCIE1	PCIE0	70
(0x67)	Reserved	-	-	-	-	-	-	-	-	
(0x66)	OSCCAL	Oscillator Calibration Register								41
(0x65)	Reserved	-	-	-	-	-	-	-	-	
(0x64)	PRR	PRTWI	PRTIM2	PRTIM0	PRUSART1	PRTIM1	PRSPI	PRUSART0	PRADC	49
(0x63)	Reserved	-	-	-	-	-	-	-	-	
(0x62)	Reserved	-	-	-	-	-	-	-	-	
(0x61)	CLKPR	CLKPCE	-	-	-	CLKPS3	CLKPS2	CLKPS1	CLKPS0	41
(0x60)	WDTCSR	WDIF	WDIE	WDP3	WDCE	WDE	WDP2	WDP1	WDP0	60
0x3F (0x5F)	SREG	I	T	H	S	V	N	Z	C	11
0x3E (0x5E)	SPH	SP15	SP14	SP13	SP12	SP11	SP10	SP9	SP8	12
0x3D (0x5D)	SPL	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0	12
0x3C (0x5C)	Reserved	-	-	-	-	-	-	-	-	
0x3B (0x5B)	RAMPZ	-	-	-	-	-	-	-	RAMPZ0	15
0x3A (0x5A)	Reserved	-	-	-	-	-	-	-	-	
0x39 (0x59)	Reserved	-	-	-	-	-	-	-	-	
0x38 (0x58)	Reserved	-	-	-	-	-	-	-	-	
0x37 (0x57)	SPMCSR	SPMIE	RWWSB	SIGRD	RWWSRE	BLBSET	PGWRT	PGERS	SPMEN	292
0x36 (0x56)	Reserved	-	-	-	-	-	-	-	-	
0x35 (0x55)	MCUCR	JTD	BODS	BODSE	PUD	-	-	IVSEL	IVCE	92/276
0x34 (0x54)	MCUSR	-	-	-	JTRF	WDRF	BORF	EXTRF	PORF	59/276
0x33 (0x53)	SMCR	-	-	-	-	SM2	SM1	SM0	SE	48
0x32 (0x52)	Reserved	-	-	-	-	-	-	-	-	
0x31 (0x51)	OCDR	On-Chip Debug Register								266
0x30 (0x50)	ACSR	ACD	ACBG	ACO	ACI	ACIE	ACIC	ACIS1	ACIS0	258
0x2F (0x4F)	Reserved	-	-	-	-	-	-	-	-	
0x2E (0x4E)	SPDR	SPI 0 Data Register								171
0x2D (0x4D)	SPSR	SPIF0	WCOL0	-	-	-	-	-	SPI2X0	170
0x2C (0x4C)	SPCR	SPIE0	SPE0	DORD0	MSTR0	CPOL0	CPHA0	SPR01	SPR00	169
0x2B (0x4B)	GPIOR2	General Purpose I/O Register 2								29
0x2A (0x4A)	GPIOR1	General Purpose I/O Register 1								29
0x29 (0x49)	Reserved	-	-	-	-	-	-	-	-	
0x28 (0x48)	OCR0B	Timer/Counter0 Output Compare Register B								110
0x27 (0x47)	OCR0A	Timer/Counter0 Output Compare Register A								109
0x26 (0x46)	TCNT0	Timer/Counter0 (8 Bit)								109
0x25 (0x45)	TCCR0B	FOC0A	FOC0B	-	-	WGM02	CS02	CS01	CS00	108
0x24 (0x44)	TCCR0A	COM0A1	COM0A0	COM0B1	COM0B0	-	-	WGM01	WGM00	110
0x23 (0x43)	GTCCR	TSM	-	-	-	-	-	PSRASY	PSR5SYNC	160
0x22 (0x42)	EEARH	-	-	-	-	EEPROM Address Register High Byte				24
0x21 (0x41)	EEARL	EEPROM Address Register Low Byte								24
0x20 (0x40)	EEDR	EEPROM Data Register								24
0x1F (0x3F)	EECR	-	-	EEPM1	EEPM0	EERIE	EEMPE	EEPE	EERE	24
0x1E (0x3E)	GPIOR0	General Purpose I/O Register 0								29
0x1D (0x3D)	EIMSK	-	-	-	-	-	INT2	INT1	INT0	69

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Page
0x1C (0x3C)	EIFR	-	-	-	-	-	INTF2	INTF1	INTF0	69
0x1B (0x3B)	PCIFR	-	-	-	-	PCIF3	PCIF2	PCIF1	PCIF0	70
0x1A (0x3A)	Reserved	-	-	-	-	-	-	-	-	
0x19 (0x39)	Reserved	-	-	-	-	-	-	-	-	
0x18 (0x38)	Reserved	-	-	-	-	-	-	-	-	
0x17 (0x37)	TIFR2	-	-	-	-	-	OCF2B	OCF2A	TOV2	160
0x16 (0x36)	TIFR1	-	-	ICF1	-	-	OCF1B	OCF1A	TOV1	139
0x15 (0x35)	TIFR0	-	-	-	-	-	OCF0B	OCF0A	TOV0	110
0x14 (0x34)	Reserved	-	-	-	-	-	-	-	-	
0x13 (0x33)	Reserved	-	-	-	-	-	-	-	-	
0x12 (0x32)	Reserved	-	-	-	-	-	-	-	-	
0x11 (0x31)	Reserved	-	-	-	-	-	-	-	-	
0x10 (0x30)	Reserved	-	-	-	-	-	-	-	-	
0x0F (0x2F)	Reserved	-	-	-	-	-	-	-	-	
0x0E (0x2E)	Reserved	-	-	-	-	-	-	-	-	
0x0D (0x2D)	Reserved	-	-	-	-	-	-	-	-	
0x0C (0x2C)	Reserved	-	-	-	-	-	-	-	-	
0x0B (0x2B)	PORTD	PORTD7	PORTD6	PORTD5	PORTD4	PORTD3	PORTD2	PORTD1	PORTD0	93
0x0A (0x2A)	DDRD	DDD7	DDD6	DDD5	DDD4	DDD3	DDD2	DDD1	DDD0	93
0x09 (0x29)	PIND	PIND7	PIND6	PIND5	PIND4	PIND3	PIND2	PIND1	PIND0	93
0x08 (0x28)	PORTC	PORTC7	PORTC6	PORTC5	PORTC4	PORTC3	PORTC2	PORTC1	PORTC0	93
0x07 (0x27)	DDRC	DDC7	DDC6	DDC5	DDC4	DDC3	DDC2	DDC1	DDC0	93
0x06 (0x26)	PINC	PINC7	PINC6	PINC5	PINC4	PINC3	PINC2	PINC1	PINC0	93
0x05 (0x25)	PORTB	PORTB7	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0	92
0x04 (0x24)	DDRB	DDB7	DDB6	DDB5	DDB4	DDB3	DDB2	DDB1	DDB0	92
0x03 (0x23)	PINB	PINB7	PINB6	PINB5	PINB4	PINB3	PINB2	PINB1	PINB0	92
0x02 (0x22)	PORTA	PORTA7	PORTA6	PORTA5	PORTA4	PORTA3	PORTA2	PORTA1	PORTA0	92
0x01 (0x21)	DDRA	DDA7	DDA6	DDA5	DDA4	DDA3	DDA2	DDA1	DDA0	92
0x00 (0x20)	PINA	PINA7	PINA6	PINA5	PINA4	PINA3	PINA2	PINA1	PINA0	92

Notes:

1. For compatibility with future devices, reserved bits should be written to zero if accessed. Reserved I/O memory addresses should never be written.
2. I/O registers within the address range $00 - $1F are directly bit-accessible using the SBI and CBI instructions. In these registers, the value of single bits can be checked by using the SBIS and SBIC instructions.
3. Some of the status flags are cleared by writing a logical one to them. Note that the CBI and SBI instructions will operate on all bits in the I/O register, writing a one back into any flag read as set, thus clearing the flag. The CBI and SBI instructions work with registers 0x00 to 0x1F only.
4. When using the I/O specific commands IN and OUT, the I/O addresses $00 - $3F must be used. When addressing I/O registers as data space using LD and ST instructions, $20 must be added to these addresses. The ATmega164P/324P/644P is a complex microcontroller with more peripheral units than can be supported within the 64 location reserved in Opcode for the IN and OUT instructions. For the Extended I/O space from $60 - $FF, only the ST/STS/STD and LD/LDS/LDD instructions can be used.

5. Instruction Set Summary

Mnemonics	Operands	Description	Operation	Flags	#Clocks
ARITHMETIC AND LOGIC INSTRUCTIONS					
ADD	Rd, Rr	Add two Registers	Rd ← Rd + Rr	Z,C,N,V,H	1
ADC	Rd, Rr	Add with Carry two Registers	Rd ← Rd + Rr + C	Z,C,N,V,H	1
ADIW	Rdl,K	Add Immediate to Word	Rdh:Rdl ← Rdh:Rdl + K	Z,C,N,V,S	2
SUB	Rd, Rr	Subtract two Registers	Rd ← Rd - Rr	Z,C,N,V,H	1
SUBI	Rd, K	Subtract Constant from Register	Rd ← Rd - K	Z,C,N,V,H	1
SBC	Rd, Rr	Subtract with Carry two Registers	Rd ← Rd - Rr - C	Z,C,N,V,H	1
SBCI	Rd, K	Subtract with Carry Constant from Reg.	Rd ← Rd - K - C	Z,C,N,V,H	1
SBIW	Rdl,K	Subtract Immediate from Word	Rdh:Rdl ← Rdh:Rdl - K	Z,C,N,V,S	2
AND	Rd, Rr	Logical AND Registers	Rd ← Rd • Rr	Z,N,V	1
ANDI	Rd, K	Logical AND Register and Constant	Rd ← Rd • K	Z,N,V	1
OR	Rd, Rr	Logical OR Registers	Rd ← Rd v Rr	Z,N,V	1
ORI	Rd, K	Logical OR Register and Constant	Rd ← Rd v K	Z,N,V	1
EOR	Rd, Rr	Exclusive OR Registers	Rd ← Rd ⊕ Rr	Z,N,V	1
COM	Rd	One's Complement	Rd ← 0xFF − Rd	Z,C,N,V	1
NEG	Rd	Two's Complement	Rd ← 0x00 − Rd	Z,C,N,V,H	1
SBR	Rd,K	Set Bit(s) in Register	Rd ← Rd v K	Z,N,V	1
CBR	Rd,K	Clear Bit(s) in Register	Rd ← Rd • (0xFF - K)	Z,N,V	1
INC	Rd	Increment	Rd ← Rd + 1	Z,N,V	1
DEC	Rd	Decrement	Rd ← Rd − 1	Z,N,V	1
TST	Rd	Test for Zero or Minus	Rd ← Rd • Rd	Z,N,V	1
CLR	Rd	Clear Register	Rd ← Rd ⊕ Rd	Z,N,V	1
SER	Rd	Set Register	Rd ← 0xFF	None	1
MUL	Rd, Rr	Multiply Unsigned	R1:R0 ← Rd x Rr	Z,C	2
MULS	Rd, Rr	Multiply Signed	R1:R0 ← Rd x Rr	Z,C	2
MULSU	Rd, Rr	Multiply Signed with Unsigned	R1:R0 ← Rd x Rr	Z,C	2
FMUL	Rd, Rr	Fractional Multiply Unsigned	R1:R0 ← (Rd x Rr) << 1	Z,C	2
FMULS	Rd, Rr	Fractional Multiply Signed	R1:R0 ← (Rd x Rr) << 1	Z,C	2
FMULSU	Rd, Rr	Fractional Multiply Signed with Unsigned	R1:R0 ← (Rd x Rr) << 1	Z,C	2
BRANCH INSTRUCTIONS					
RJMP	k	Relative Jump	PC ← PC + k + 1	None	2
IJMP		Indirect Jump to (Z)	PC ← Z	None	2
JMP	k	Direct Jump	PC ← k	None	3
RCALL	k	Relative Subroutine Call	PC ← PC + k + 1	None	4
ICALL		Indirect Call to (Z)	PC ← Z	None	4
CALL	k	Direct Subroutine Call	PC ← k	None	5
RET		Subroutine Return	PC ← STACK	None	5
RETI		Interrupt Return	PC ← STACK	I	5
CPSE	Rd,Rr	Compare, Skip if Equal	if (Rd = Rr) PC ← PC + 2 or 3	None	1/2/3
CP	Rd,Rr	Compare	Rd − Rr	Z, N,V,C,H	1
CPC	Rd,Rr	Compare with Carry	Rd − Rr − C	Z, N,V,C,H	1
CPI	Rd,K	Compare Register with Immediate	Rd − K	Z, N,V,C,H	1
SBRC	Rr, b	Skip if Bit in Register Cleared	if (Rr(b)=0) PC ← PC + 2 or 3	None	1/2/3
SBRS	Rr, b	Skip if Bit in Register is Set	if (Rr(b)=1) PC ← PC + 2 or 3	None	1/2/3
SBIC	P, b	Skip if Bit in I/O Register Cleared	if (P(b)=0) PC ← PC + 2 or 3	None	1/2/3
SBIS	P, b	Skip if Bit in I/O Register is Set	if (P(b)=1) PC ← PC + 2 or 3	None	1/2/3
BRBS	s, k	Branch if Status Flag Set	if (SREG(s) = 1) then PC←PC+k + 1	None	1/2
BRBC	s, k	Branch if Status Flag Cleared	if (SREG(s) = 0) then PC←PC+k + 1	None	1/2
BREQ	k	Branch if Equal	if (Z = 1) then PC ← PC + k + 1	None	1/2
BRNE	k	Branch if Not Equal	if (Z = 0) then PC ← PC + k + 1	None	1/2
BRCS	k	Branch if Carry Set	if (C = 1) then PC ← PC + k + 1	None	1/2
BRCC	k	Branch if Carry Cleared	if (C = 0) then PC ← PC + k + 1	None	1/2
BRSH	k	Branch if Same or Higher	if (C = 0) then PC ← PC + k + 1	None	1/2
BRLO	k	Branch if Lower	if (C = 1) then PC ← PC + k + 1	None	1/2
BRMI	k	Branch if Minus	if (N = 1) then PC ← PC + k + 1	None	1/2
BRPL	k	Branch if Plus	if (N = 0) then PC ← PC + k + 1	None	1/2
BRGE	k	Branch if Greater or Equal, Signed	if (N ⊕ V= 0) then PC ← PC + k + 1	None	1/2
BRLT	k	Branch if Less Than Zero, Signed	if (N ⊕ V= 1) then PC ← PC + k + 1	None	1/2
BRHS	k	Branch if Half Carry Flag Set	if (H = 1) then PC ← PC + k + 1	None	1/2
BRHC	k	Branch if Half Carry Flag Cleared	if (H = 0) then PC ← PC + k + 1	None	1/2
BRTS	k	Branch if T Flag Set	if (T = 1) then PC ← PC + k + 1	None	1/2
BRTC	k	Branch if T Flag Cleared	if (T = 0) then PC ← PC + k + 1	None	1/2
BRVS	k	Branch if Overflow Flag is Set	if (V = 1) then PC ← PC + k + 1	None	1/2

Mnemonics	Operands	Description	Operation	Flags	#Clocks
BRVC	k	Branch if Overflow Flag is Cleared	if (V = 0) then PC ← PC + k + 1	None	1/2
BRIE	k	Branch if Interrupt Enabled	if (I = 1) then PC ← PC + k + 1	None	1/2
BRID	k	Branch if Interrupt Disabled	if (I = 0) then PC ← PC + k + 1	None	1/2
BIT AND BIT-TEST INSTRUCTIONS					
SBI	P,b	Set Bit in I/O Register	I/O(P,b) ← 1	None	2
CBI	P,b	Clear Bit in I/O Register	I/O(P,b) ← 0	None	2
LSL	Rd	Logical Shift Left	Rd(n+1) ← Rd(n), Rd(0) ← 0	Z,C,N,V	1
LSR	Rd	Logical Shift Right	Rd(n) ← Rd(n+1), Rd(7) ← 0	Z,C,N,V	1
ROL	Rd	Rotate Left Through Carry	Rd(0)←C,Rd(n+1)← Rd(n),C←Rd(7)	Z,C,N,V	1
ROR	Rd	Rotate Right Through Carry	Rd(7)←C,Rd(n)← Rd(n+1),C←Rd(0)	Z,C,N,V	1
ASR	Rd	Arithmetic Shift Right	Rd(n) ← Rd(n+1), n=0..6	Z,C,N,V	1
SWAP	Rd	Swap Nibbles	Rd(3..0)←Rd(7..4),Rd(7..4)←Rd(3..0)	None	1
BSET	s	Flag Set	SREG(s) ← 1	SREG(s)	1
BCLR	s	Flag Clear	SREG(s) ← 0	SREG(s)	1
BST	Rr, b	Bit Store from Register to T	T ← Rr(b)	T	1
BLD	Rd, b	Bit load from T to Register	Rd(b) ← T	None	1
SEC		Set Carry	C ← 1	C	1
CLC		Clear Carry	C ← 0	C	1
SEN		Set Negative Flag	N ← 1	N	1
CLN		Clear Negative Flag	N ← 0	N	1
SEZ		Set Zero Flag	Z ← 1	Z	1
CLZ		Clear Zero Flag	Z ← 0	Z	1
SEI		Global Interrupt Enable	I ← 1	I	1
CLI		Global Interrupt Disable	I ← 0	I	1
SES		Set Signed Test Flag	S ← 1	S	1
CLS		Clear Signed Test Flag	S ← 0	S	1
SEV		Set Twos Complement Overflow.	V ← 1	V	1
CLV		Clear Twos Complement Overflow	V ← 0	V	1
SET		Set T in SREG	T ← 1	T	1
CLT		Clear T in SREG	T ← 0	T	1
SEH		Set Half Carry Flag in SREG	H ← 1	H	1
CLH		Clear Half Carry Flag in SREG	H ← 0	H	1
DATA TRANSFER INSTRUCTIONS					
MOV	Rd, Rr	Move Between Registers	Rd ← Rr	None	1
MOVW	Rd, Rr	Copy Register Word	Rd+1:Rd ← Rr+1:Rr	None	1
LDI	Rd, K	Load Immediate	Rd ← K	None	1
LD	Rd, X	Load Indirect	Rd ← (X)	None	2
LD	Rd, X+	Load Indirect and Post-Inc.	Rd ← (X), X ← X + 1	None	2
LD	Rd, - X	Load Indirect and Pre-Dec.	X ← X - 1, Rd ← (X)	None	2
LD	Rd, Y	Load Indirect	Rd ← (Y)	None	2
LD	Rd, Y+	Load Indirect and Post-Inc.	Rd ← (Y), Y ← Y + 1	None	2
LD	Rd, - Y	Load Indirect and Pre-Dec.	Y ← Y - 1, Rd ← (Y)	None	2
LDD	Rd,Y+q	Load Indirect with Displacement	Rd ← (Y + q)	None	2
LD	Rd, Z	Load Indirect	Rd ← (Z)	None	2
LD	Rd, Z+	Load Indirect and Post-Inc.	Rd ← (Z), Z ← Z+1	None	2
LD	Rd, -Z	Load Indirect and Pre-Dec.	Z ← Z - 1, Rd ← (Z)	None	2
LDD	Rd, Z+q	Load Indirect with Displacement	Rd ← (Z + q)	None	2
LDS	Rd, k	Load Direct from SRAM	Rd ← (k)	None	2
ST	X, Rr	Store Indirect	(X) ← Rr	None	2
ST	X+, Rr	Store Indirect and Post-Inc.	(X) ← Rr, X ← X + 1	None	2
ST	- X, Rr	Store Indirect and Pre-Dec.	X ← X - 1, (X) ← Rr	None	2
ST	Y, Rr	Store Indirect	(Y) ← Rr	None	2
ST	Y+, Rr	Store Indirect and Post-Inc.	(Y) ← Rr, Y ← Y + 1	None	2
ST	- Y, Rr	Store Indirect and Pre-Dec.	Y ← Y - 1, (Y) ← Rr	None	2
STD	Y+q,Rr	Store Indirect with Displacement	(Y + q) ← Rr	None	2
ST	Z, Rr	Store Indirect	(Z) ← Rr	None	2
ST	Z+, Rr	Store Indirect and Post-Inc.	(Z) ← Rr, Z ← Z + 1	None	2
ST	-Z, Rr	Store Indirect and Pre-Dec.	Z ← Z - 1, (Z) ← Rr	None	2
STD	Z+q,Rr	Store Indirect with Displacement	(Z + q) ← Rr	None	2
STS	k, Rr	Store Direct to SRAM	(k) ← Rr	None	2
LPM		Load Program Memory	R0 ← (Z)	None	3
LPM	Rd, Z	Load Program Memory	Rd ← (Z)	None	3
LPM	Rd, Z+	Load Program Memory and Post-Inc	Rd ← (Z), Z ← Z+1	None	3
ELPM		Extended Load Program Memory	R0 ← (RAMPZ:Z)	None	3
ELPM	Rd, Z	Extended Load Program Memory	Rd ← (Z)	None	3
ELPM	Rd, Z+	Extended Load Program Memory	Rd ← (RAMPZ:Z), RAMPZ:Z ←RAMPZ:Z+1	None	3

Mnemonics	Operands	Description	Operation	Flags	#Clocks
SPM		Store Program Memory	(Z) ← R1:R0	None	-
IN	Rd, P	In Port	Rd ← P	None	1
OUT	P, Rr	Out Port	P ← Rr	None	1
PUSH	Rr	Push Register on Stack	STACK ← Rr	None	2
POP	Rd	Pop Register from Stack	Rd ← STACK	None	2
MCU CONTROL INSTRUCTIONS					
NOP		No Operation		None	1
SLEEP		Sleep	(see specific descr. for Sleep function)	None	1
WDR		Watchdog Reset	(see specific descr. for WDR/timer)	None	1
BREAK		Break	For On-chip Debug Only	None	N/A

6. Ordering Information

6.1 ATmega164P

Speed (MHz)[3]	Power Supply	Ordering Code	Package[1]	Operational Range
10	1.8 - 5.5V	ATmega164PV-10AU[2] ATmega164PV-10PU[2] ATmega164PV-10MU[2]	44A 40P6 44M1	Industrial (-40°C to 85°C)
20	2.7 - 5.5V	ATmega164P-20AU[2] ATmega164P-20PU[2] ATmega164P-20MU[2]	44A 40P6 44M1	

Notes:
1. This device can also be supplied in wafer form. Please contact your local Atmel sales office for detailed ordering information and minimum quantities.
2. Pb-free packaging, complies to the European Directive for Restriction of Hazardous Substances (RoHS directive). Also Halide free and fully Green.
3. For Speed vs. V_{CC} see ”Speed Grades” on page 329.

Package Type	
44A	44-lead, Thin (1.0 mm) Plastic Gull Wing Quad Flat Package (TQFP)
40P6	40-pin, 0.600” Wide, Plastic Dual Inline Package (PDIP)
44M1	44-pad, 7 x 7 x 1.0 mm body, lead pitch 0.50 mm, Thermally Enhanced Plastic Very Thin Quad Flat No-Lead (VQFN)

6.2 ATmega324P

Speed (MHz)[3]	Power Supply	Ordering Code	Package[1]	Operational Range
10	1.8 - 5.5V	ATmega324PV-10AU[2] ATmega324PV-10PU[2] ATmega324PV-10MU[2]	44A 40P6 44M1	Industrial (-40°C to 85°C)
20	2.7 - 5.5V	ATmega324P-20AU[2] ATmega324P-20PU[2] ATmega324P-20MU[2]	44A 40P6 44M1	

Notes:
1. This device can also be supplied in wafer form. Please contact your local Atmel sales office for detailed ordering information and minimum quantities.
2. Pb-free packaging, complies to the European Directive for Restriction of Hazardous Substances (RoHS directive). Also Halide free and fully Green.
3. For Speed vs. V_{CC} see "Speed Grades" on page 329.

Package Type	
44A	44-lead, Thin (1.0 mm) Plastic Gull Wing Quad Flat Package (TQFP)
40P6	40-pin, 0.600" Wide, Plastic Dual Inline Package (PDIP)
44M1	44-pad, 7 x 7 x 1.0 mm Body, lead pitch 0.50 mm, Thermally Enhanced Plastic Very Thin Quad Flat No-Lead (VQFN)

6.3 ATmega644P

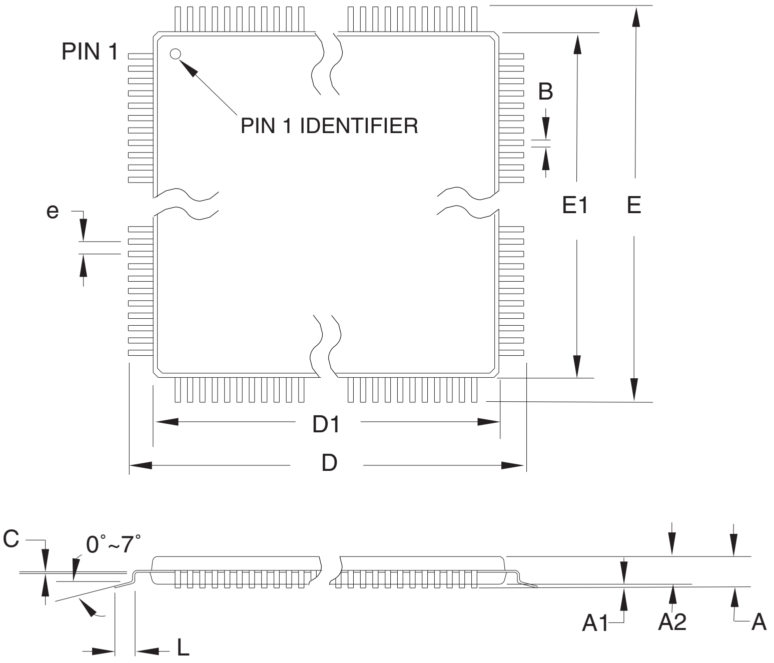
Speed (MHz)[3]	Power Supply	Ordering Code	Package[1]	Operational Range
10	1.8 - 5.5V	ATmega644PV-10AU[2] ATmega644PV-10PU[2] ATmega644PV-10MU[2]	44A 40P6 44M1	Industrial (-40°C to 85°C)
20	2.7 - 5.5V	ATmega644P-20AU[2] ATmega644P-20PU[2] ATmega644P-20MU[2]	44A 40P6 44M1	

Notes:
1. This device can also be supplied in wafer form. Please contact your local Atmel sales office for detailed ordering information and minimum quantities.
2. Pb-free packaging, complies to the European Directive for Restriction of Hazardous Substances (RoHS directive). Also Halide free and fully Green.
3. For Speed vs. V_{CC} see "Speed Grades" on page 329.

Package Type	
44A	44-lead, Thin (1.0 mm) Plastic Gull Wing Quad Flat Package (TQFP)
40P6	40-pin, 0.600" Wide, Plastic Dual Inline Package (PDIP)
44M1	44-pad, 7 x 7 x 1.0 mm body, lead pitch 0.50 mm, Thermally Enhanced Plastic Very Thin Quad Flat No-Lead (VQFN)

7. Packaging Information

7.1 44A



COMMON DIMENSIONS
(Unit of Measure = mm)

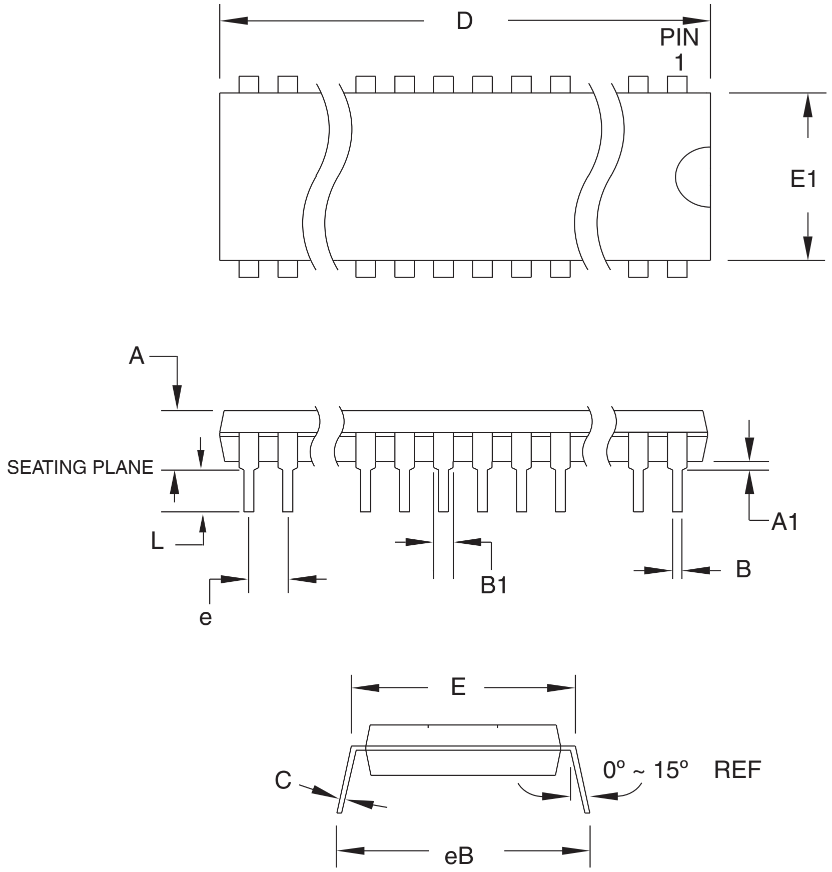
SYMBOL	MIN	NOM	MAX	NOTE
A	–	–	1.20	
A1	0.05	–	0.15	
A2	0.95	1.00	1.05	
D	11.75	12.00	12.25	
D1	9.90	10.00	10.10	Note 2
E	11.75	12.00	12.25	
E1	9.90	10.00	10.10	Note 2
B	0.30	–	0.45	
C	0.09	–	0.20	
L	0.45	–	0.75	
e	0.80 TYP			

Notes:
1. This package conforms to JEDEC reference MS-026, Variation ACB.
2. Dimensions D1 and E1 do not include mold protrusion. Allowable protrusion is 0.25 mm per side. Dimensions D1 and E1 are maximum plastic body size dimensions including mold mismatch.
3. Lead coplanarity is 0.10 mm maximum.

10/5/2001

	TITLE	DRAWING NO.	REV.
ATMEL 2325 Orchard Parkway San Jose, CA 95131	**44A,** 44-lead, 10 x 10 mm Body Size, 1.0 mm Body Thickness, 0.8 mm Lead Pitch, Thin Profile Plastic Quad Flat Package (TQFP)	44A	B

7.2 40P6



COMMON DIMENSIONS
(Unit of Measure = mm)

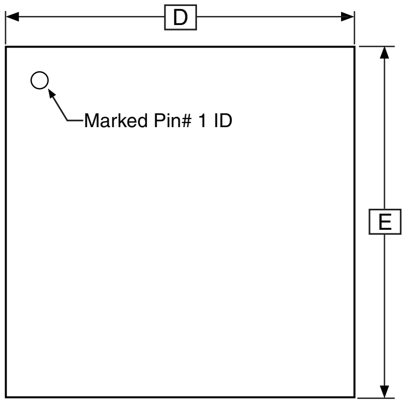
SYMBOL	MIN	NOM	MAX	NOTE
A	–	–	4.826	
A1	0.381	–	–	
D	52.070	–	52.578	Note 2
E	15.240	–	15.875	
E1	13.462	–	13.970	Note 2
B	0.356	–	0.559	
B1	1.041	–	1.651	
L	3.048	–	3.556	
C	0.203	–	0.381	
eB	15.494	–	17.526	
e	2.540 TYP			

Notes:
1. This package conforms to JEDEC reference MS-011, Variation AC.
2. Dimensions D and E1 do not include mold Flash or Protrusion. Mold Flash or Protrusion shall not exceed 0.25 mm (0.010").

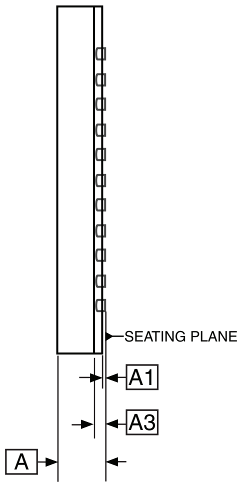
09/28/01

	TITLE	DRAWING NO.	REV.
2325 Orchard Parkway San Jose, CA 95131	**40P6**, 40-lead (0.600"/15.24 mm Wide) Plastic Dual Inline Package (PDIP)	40P6	B

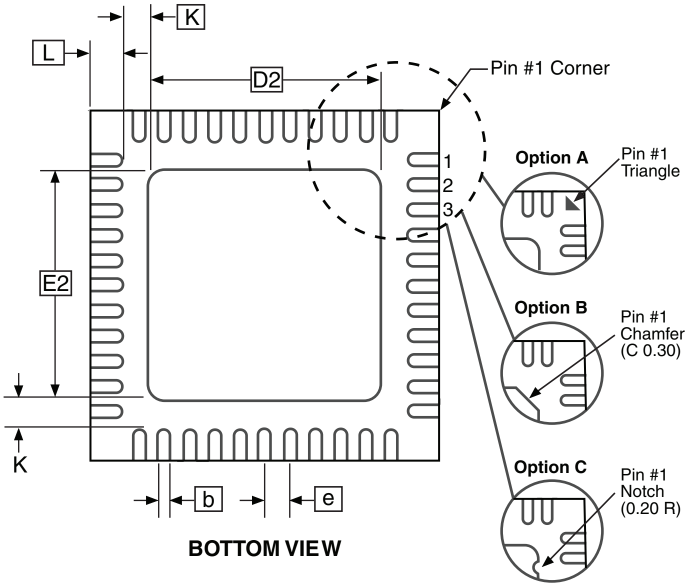
7.3 44M1



TOP VIEW



SIDE VIEW



BOTTOM VIEW

Note: JEDEC Standard MO-220, Fig. 1 (SAW Singulation) VKKD-3.

COMMON DIMENSIONS
(Unit of Measure = mm)

SYMBOL	MIN	NOM	MAX	NOTE
A	0.80	0.90	1.00	
A1	–	0.02	0.05	
A3		0.20 REF		
b	0.18	0.23	0.30	
D	6.90	7.00	7.10	
D2	5.00	5.20	5.40	
E	6.90	7.00	7.10	
E2	5.00	5.20	5.40	
e		0.50 BSC		
L	0.59	0.64	0.69	
K	0.20	0.26	0.41	

9/26/08

	TITLE	GPC	DRAWING NO.	REV.
Package Drawing Contact: packagedrawings@atmel.com	**44M1,** 44-pad, 7 x 7 x 1.0 mm Body, Lead Pitch 0.50 mm, 5.20 mm Exposed Pad, Thermally Enhanced Plastic Very Thin Quad Flat No Lead Package (VQFN)	ZWS	44M1	H

8. Errata

8.1 ATmega164P

8.1.1 Rev. A

No known Errata.

8.2 ATmega324P

8.2.1 Rev. A

No known Errata.

8.3 ATmega644P

8.3.1 Rev. A

Not sampled.

8.3.2 Rev. B

No known Errata.

9. Datasheet Revision History

Please note that the referring page numbers in this section are referred to this document. The referring revision in this section are referring to the document revision.

9.1 Rev. 8011M- 08/09

1. Updated ”Features” on page 1.
2. Removed VFBGA - pinout from ”Pin Configurations” on page 2.
3. Updated ”System Control and Reset” on page 50.
4. Updated Input Hysteresis Unit (V) in the “Typical Characteristics”.
5. Updated ”Ordering Information” on page 420. Removed 44MC and 49C2 packages.
6. Updated ”Packaging Information” on page 423.

9.2 Rev. 8011L- 02/09

1. Updated ”Features” on page 1 by inserting a table note 1.
2. Merged Sections 3.1, 3.2 and 3.3 in one section ”About” on page 9.
3. Updated the front page by removing “Preliminary”.
4. Updated the ”DC Characteristics” on page 326 by removing V_{IL3}/V_{IH3} and V_{OL3}/V_{OH3} and the table note 5.
5. Updated the table note1 of the Table 25-6 on page 332.
6. Updated ”Typical Characteristics” on page 339.
6. Updated ”Typical Characteristics” on page 339

9.3 Rev. 8011K- 09/08

1. Updated ”Features” on page 1, ”Pin Configurations” on page 2 and ”Ordering Information” on page 15 according to the updated 44M1 package drawing.
2. Updated V_{OL} in the table of ”DC Characteristics” on page 326.
3. Updated t_{RST} and t_{BOD} unites in the table of ”System and Reset Characteristics” on page 332.
4. Updated typical values for ATmega324P and ATmega644P in the tables of ”DC Characteristics” on page 326.
5. Replaced the package drawing ”44M1” on page 426 by a rev H update.
2. Added 49-ball VFBGA pinout for ATmega164P/324P in ”Pinout - VFBGA” on page 4.
6. Added 49-ball VFBGA (49C2) to ”Packaging Information” on page 19.

9.4 Rev. 8011J- 09/08

1. Updated ATmega644P ”Errata” on page 428.
2. Added 49-ball VFBGA pinout for ATmega164P/324P in ”Pinout - VFBGA” on page 4.
6. Added 49-ball VFBGA (49C2) to ”Packaging Information” on page 425.

9.5 Rev. 8011I- 05/08

1. Updated description in ”AVCC” on page 7.
2. Updated ”Stack Pointer” on page 14.
3. Updated Data Memory Map addresses, Figure 7-2 on page 21.
4. Updated description of use of external capacitors in ”Low Frequency Crystal Oscillator” on page 35.
5. Updated typo in”Alternate Functions of Port C” on page 86.
6. Updated bit description in ”TWSR – TWI Status Register” on page 235.
7. Updated typo in ”Programming via the JTAG Interface” on page 313.
8. Updated conditions for V_{OL} in the table of ”DC Characteristics” on page 326.
9. Updated ”External Clock Drive” on page 331.
10. Updated conditions for V_{INT2} in Table 27-11 (Single Ended channels) in ”ADC Characteristics” on page 336.
11. Updated Minimum Reference Voltage in Table 27-12 (Differential channels) in ”ADC Characteristics” on page 336.
12. Updated bit bit field typos in ”Register Summary” on page 414.
2. Added 49-ball VFBGA pinout for ATmega164P/324P in ”Pinout - VFBGA” on page 4.
6. Added 49-ball VFBGA (49C2) to ”Packaging Information” on page 425.

9.6 Rev. 8011H- 04/08

1. Added 44-pad DRQFN pinout for ATmega164P in ”Pinout - DRQFN” on page 3.
2. Added 49-ball VFBGA pinout for ATmega164P/324P in ”Pinout - VFBGA” on page 4.
2. Added note to ”Address Match Unit” on page 215.
3. Updated ATmega164P ”Ordering Information” on page 421.
4. Added 44-lead QFN (44MC) to ”Packaging Information” on page 424.
6. Added 49-ball VFBGA (49C2) to ”Packaging Information” on page 425.

9.7 Rev. 8011G- 08/07

1. Updated ”Features” on page 1
2. Added ”Data Retention” on page 9.
3. Updated ”SPH and SPL – Stack Pointer High and Stack pointer Low” on page 15.
4. LCD reference removed from table note in ”Sleep Modes” on page 43.
5. Updated code example in ”Bit 0 – IVCE: Interrupt Vector Change Enable” on page 66.

6. Removed reference to External Memory Interface in ”Alternate Functions of Port A” on page 81.
7. Updated ”Data Reception – The USART Receiver” on page 181.
8. Updated ”ADCSRB – ADC Control and Status Register B” on page 239.
9. Updated overview in ”ADC - Analog-to-digital Converter” on page 241.
10. Added ”ATmega644P Typical Characteristic” on page 389.
11. Updated Figure 28-31 on page 355, Figure 28-32 on page 356,Figure 28-33 on page 356
12. Updated notes in Table 8-3 on page 33.Table 8-8 on page 36, Table 8-9 on page 37, and Table 8-11 on page 38.
13. Updated Table 13-7 on page 85, Table 13-8 on page 85, Table 13-10 on page 87, Table 13-11 on page 88, Table 13-14 on page 91, Table 27-1 on page 328,Table 27-2 on page 328,Table 27-5 on page 331, Table 27-9 on page 333, and Table 27-12 on page 337
14. Updated ”ATmega324P DC Characteristics” on page 328 and ”ATmega644P DC Characteristics” on page 329.
15. Updated Table 27-7 on page 332 and Table 8-13 on page 38.

9.8 Rev. 8011F- 04/07

1. Updated ”Watchdog Timer Configuration” on page 60.

9.9 Rev. 8011E - 04/07

1. Updated ”GTCCR – General Timer/Counter Control Register” on page 160.
2. Updated ”EECR – The EEPROM Control Register” on page 24.

9.10 Rev. 8011D - 02/07

1. Updated ”Pinout ATmega164P/324P/644P” on page 2.
2. Updated ”Power-down Mode” on page 45.
3. Updated note in Table 12-1 on page 69.
4. Updated Table 24-1 on page 273.
5. Updated ”Boot Size Configuration[(1)]” on page 290.
6. Updated V_{OL} limits in ”DC Characteristics” on page 326.
7. Updated note 3 and 4 in ”DC Characteristics” on page 326.
8. Added note to ”ATmega164P DC Characteristics” on page 328.
9. Added note to ”ATmega324P DC Characteristics” on page 328.
10. Updated Figure 28-13 on page 346 and Figure 28-60 on page 371.

9.11 Rev. 8011C - 10/06

1. Updated ”DC Characteristics” on page 326.

9.12 Rev. 8011B - 09/06

1. Updated ”DC Characteristics” on page 326.

9.13 Rev. 8011A - 08/06

1. Initial revision.

Headquarters

Atmel Corporation
2325 Orchard Parkway
San Jose, CA 95131
USA
Tel: 1(408) 441-0311
Fax: 1(408) 487-2600

International

Atmel Asia
Unit 1-5 & 16, 19/F
BEA Tower, Millennium City 5
418 Kwun Tong Road
Kwun Tong, Kowloon
Hong Kong
Tel: (852) 2245-6100
Fax: (852) 2722-1369

Atmel Europe
Le Krebs
8, Rue Jean-Pierre Timbaud
BP 309
78054 Saint-Quentin-en-
Yvelines Cedex
France
Tel: (33) 1-30-60-70-00
Fax: (33) 1-30-60-71-11

Atmel Japan
9F, Tonetsu Shinkawa Bldg.
1-24-8 Shinkawa
Chuo-ku, Tokyo 104-0033
Japan
Tel: (81) 3-3523-3551
Fax: (81) 3-3523-7581

Product Contact

Web Site
www.atmel.com

Technical Support
avr@atmel.com

Sales Contact
www.atmel.com/contacts

Literature Requests
www.atmel.com/literature

Disclaimer: The information in this document is provided in connection with Atmel products. No license, express or implied, by estoppel or otherwise, to any intellectual property right is granted by this document or in connection with the sale of Atmel products. **EXCEPT AS SET FORTH IN ATMEL'S TERMS AND CONDITIONS OF SALE LOCATED ON ATMEL'S WEB SITE, ATMEL ASSUMES NO LIABILITY WHATSOEVER AND DISCLAIMS ANY EXPRESS, IMPLIED OR STATUTORY WARRANTY RELATING TO ITS PRODUCTS INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. IN NO EVENT SHALL ATMEL BE LIABLE FOR ANY DIRECT, INDIRECT, CONSEQUENTIAL, PUNITIVE, SPECIAL OR INCIDENTAL DAMAGES (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF PROFITS, BUSINESS INTERRUPTION, OR LOSS OF INFORMATION) ARISING OUT OF THE USE OR INABILITY TO USE THIS DOCUMENT, EVEN IF ATMEL HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.** Atmel makes no representations or warranties with respect to the accuracy or completeness of the contents of this document and reserves the right to make changes to specifications and product descriptions at any time without notice. Atmel does not make any commitment to update the information contained herein. Unless specifically provided otherwise, Atmel products are not suitable for, and shall not be used in, automotive applications. Atmel's products are not intended, authorized, or warranted for use as components in applications intended to support or sustain life.

Helical Feed Antennas

Paul Wade W1GHZ ©2002
w1ghz@arrl.net

Helical antennas have long been popular in applications from VHF to microwaves requiring circular polarization, since they have the unique property of naturally providing circularly polarized radiation. One area that takes advantage of this property is satellite communications. Where more gain is required than can be provided by a helical antenna alone, a helical antenna can also be used as a feed for a parabolic dish for higher gains. As we shall see, the helical antenna can be an excellent feed for a dish, with the advantage of circular polarization. One limitation is that the usefulness of the circular polarization is limited since it cannot be easily reversed to the other sense, left-handed to right-handed or vice-versa.

Helical Antennas

John Kraus, W8JK, is the originator of the helical-beam antenna; as he puts it[1], "which I devised in 1946". His 1950 book, *Antennas*[2], is the classic source of information. The recent third edition[3], *Antennas for All Applications*, has significant additional information.

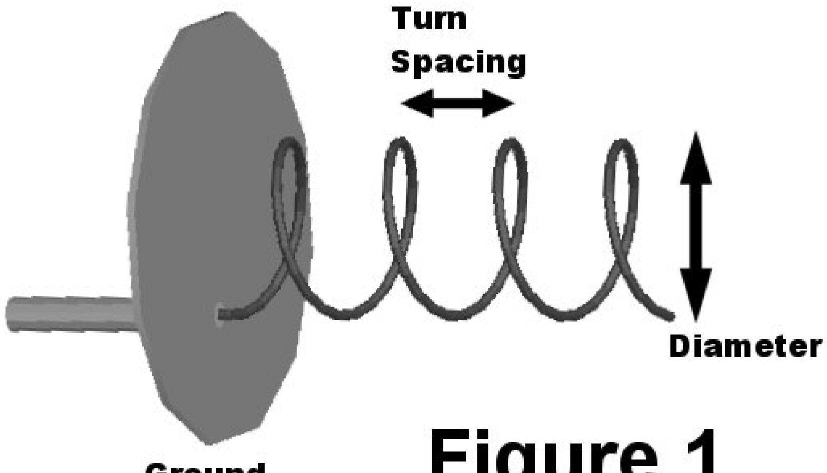


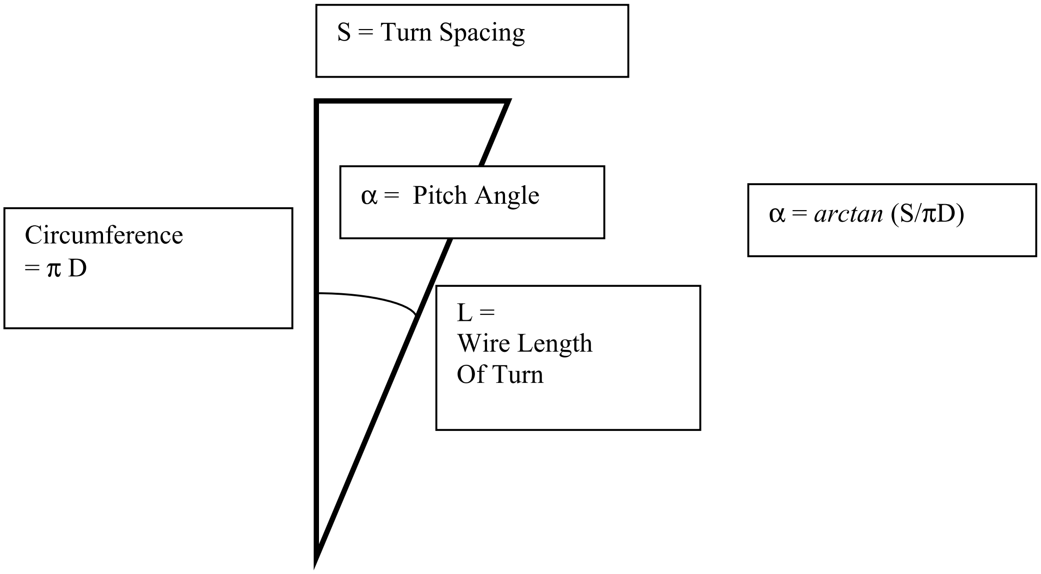
Figure 1

A sketch of a typical helical antenna is shown in Figure 1. The radiating element is a helix of wire, driven at one end and radiating along the axis of the helix. A ground plane at the driven end makes the radiation unidirectional from the far (open) end. There are also configurations that radiate perpendicular to the axis, with an omnidirectional pattern. The familiar “rubber ducky” uses this configuration; we all know that it is a relatively poor antenna, so we shall only consider the axial-mode configuration.

Typical helix dimensions for an axial-mode helical antenna have a helix circumference of one wavelength at the center frequency, with a helix pitch of 12 to 14 degrees. Kraus defines the pitch angle α as:

$$\alpha = tan^{-1}\frac{s}{\pi D}$$

where s is the spacing from turn to turn and D is the diameter, the circumference divided by π. The triangle below illustrates the relationships between the circumference, diameter, pitch, turn spacing, and wire length for each turn:



The ground plane diameter is typically 0.94λ in diameter at the center frequency, but many other configurations have been used, including square plates, wire grids, cavities, and loops[1]. The 3 dB beamwidth for a helix with **n** turns is approximately[2]:

$BW_{3dB} = \dfrac{52}{C_\lambda\sqrt{n \cdot s_\lambda}}$ degrees, where the circumference, $\mathbf{C_\lambda}$, and the turn spacing, $\mathbf{s_\lambda}$, are in wavelengths.

The gain of the helical antenna is also proportional to the number of turns. The gain curves in Kraus' 1950 book[2], and many others, show the gain increasing with helix length with no apparent limit. However, experiments with long helical antennas are invariably disappointing. Darrel Emerson, AA7FV, made a series of **NEC2** simulations of various length helical antennas and showed[4,5] that the gain approaches a limit of about 15 dB, for a length of around 7 wavelengths. The 2002 Kraus book[3] shows similar experimental data. For higher gains, arrays of multiple helixes are needed, or other types of antennas.

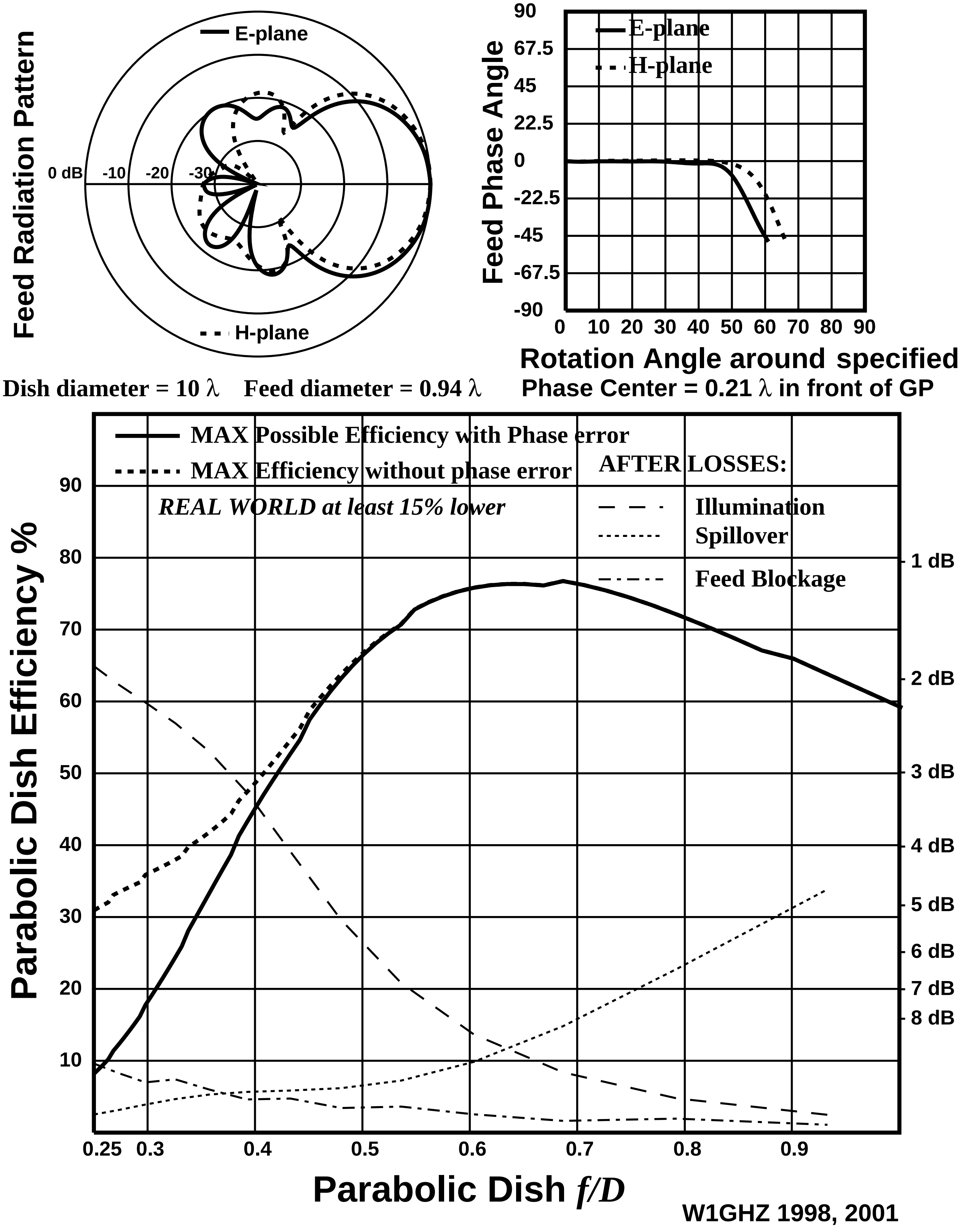
Almost all helical antennas have been made with uniform diameter and turn spacing. K2RIW once suggested that long helical antennas might require variations in diameter and spacing over the length of the antenna, just as optimized long Yagi-Uda antennas require variable element lengths and spacings for very high gain.

Some of the AMSAT satellites and others require more than 15 dB gain with circular polarization for good reception. Until someone finds an optimization that yields higher gain from a long helix, some other antenna type is needed; a parabolic dish is often a good choice. While a large dish can provide gains upward of 30 dB, a small dish can easily provide the 20 to 25 dB gain needed for many satellite applications. The beamwidth of a small dish is broader than the beam of a large dish, making tracking less difficult. Of course, the dish needs a feed antenna, and a short helix is a good choice for circular polarization. A small offset dish is very attractive, since the feed blockage, which degrades small dish performance, is greatly reduced.

Helical antennas are relatively broadband, typically useful over a range of frequencies relative to the helix circumference of $3/4\lambda$ to $4/3\lambda$, or roughly a 60% bandwidth. Most of the microwave ham bands are spaced by about this much, so there might be the possibility of covering two bands with one helical antenna, one band at the lower limit of the antenna bandwidth and the other at the upper limit. However, we shall see that for a feed antenna, the radiation patterns are much more useful near the center of the range. Thus, the main advantage of the broadband characteristic of the helical antenna is that the dimensions are not critical.

Helix Feed 4 turns 12.5˚ with 0.94λ GP diameter at 2.4 GHz

Figure 2



Helical feed antennas

A parabolic dish reflector typically requires a feed antenna with a rather large beamwidth, 90º or more. From the beamwidth formula above, only a short helix of a few turns is needed. Figure 2 shows the radiation pattern provided by a typical short helix, 4 turns with a 12.5º pitch and a ground plane of 0.94λ diameter. The calculated dish efficiency with this helix as a feed is very good, about 77%, at a center frequency of 2.4 GHz, with best **f/D** around 0.69, just about right for an offset-fed dish. Thus, we might expect a real efficiency >60% feeding a reasonably sized (>10λ) offset dish. A three-dimensional view of the radiation pattern, in Figure 3, shows a reasonably clean pattern with relatively small sidelobes; adjacent shades of gray have a difference in amplitude of 2 dB. The backlobes of most helical antennas, like the one in Figure 3, seem to have a twisted asymmetric shape.

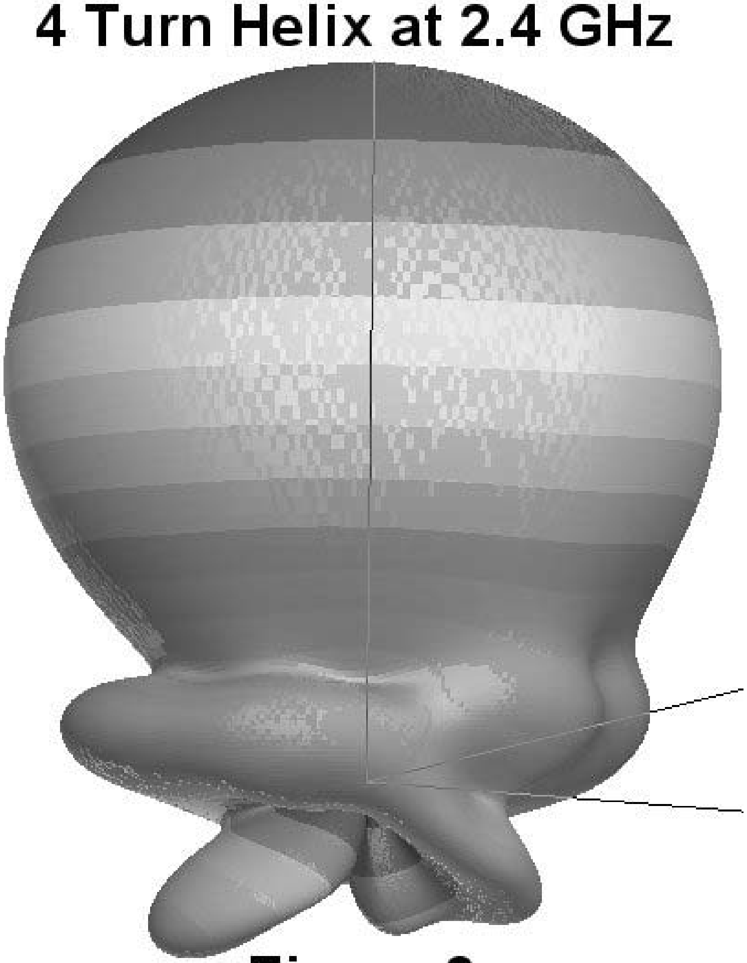
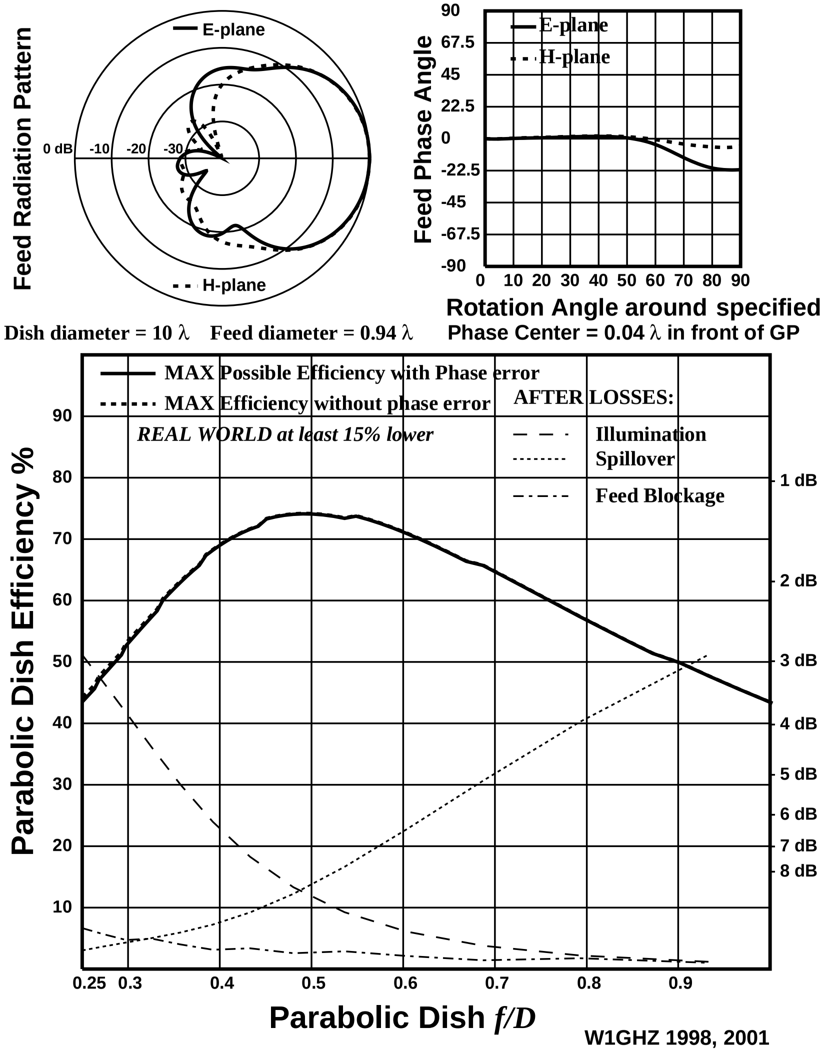


Figure 3

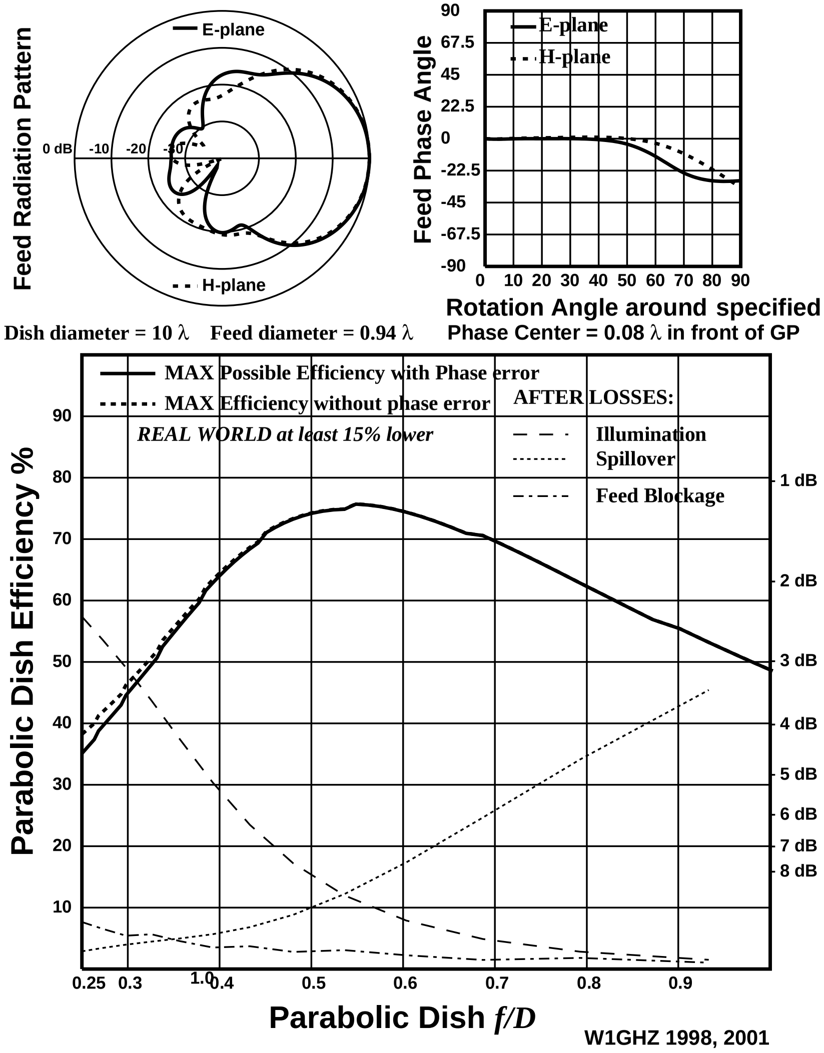
Helix 4 turns 12.5˚, 0.94λ GP, at 1.8 GHz

Figure 4a



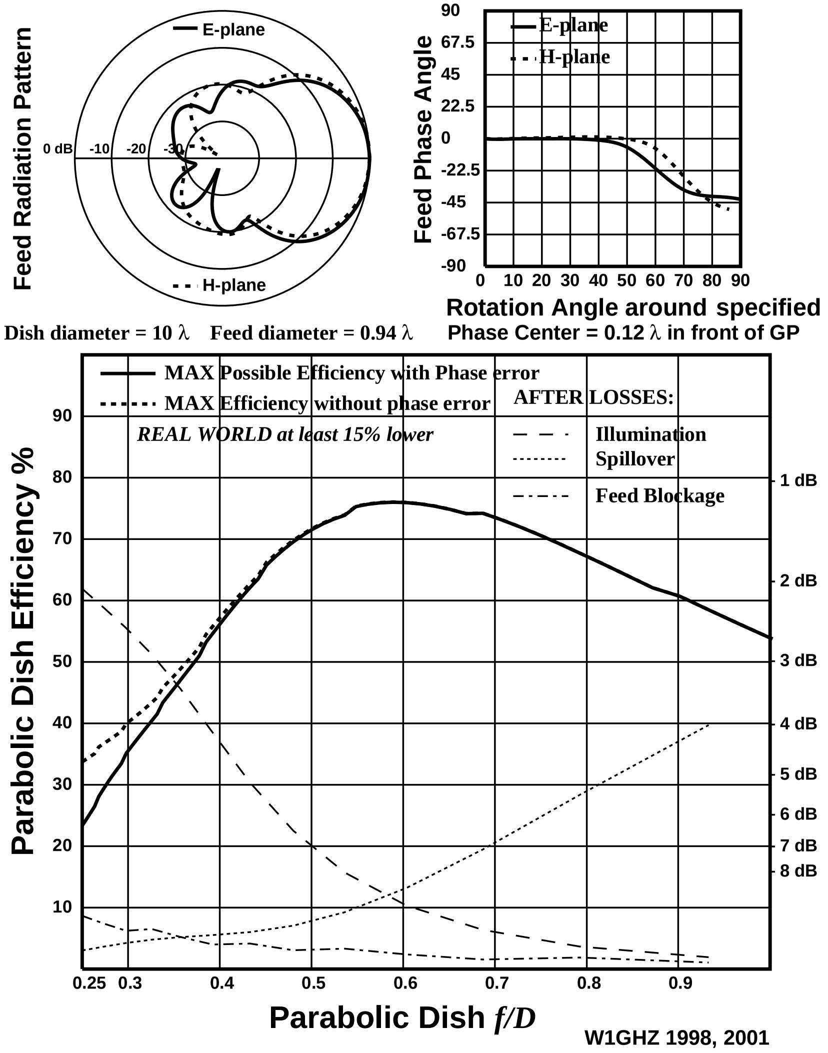
Helix 4 turn 12.5˚, 0.94λ GP, at 2.0 GHz

Figure 4b



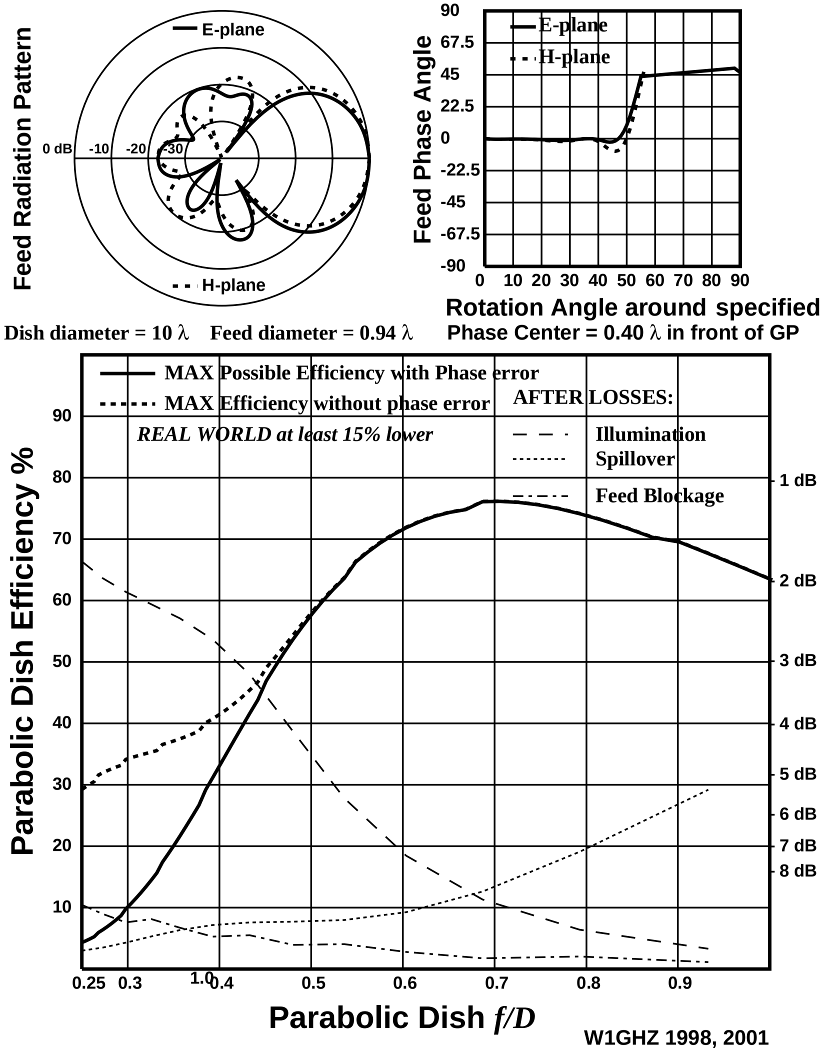
Helix 4 turns 12.5˚, 0.94λ GP, at 2.2 GHz

Figure 4c



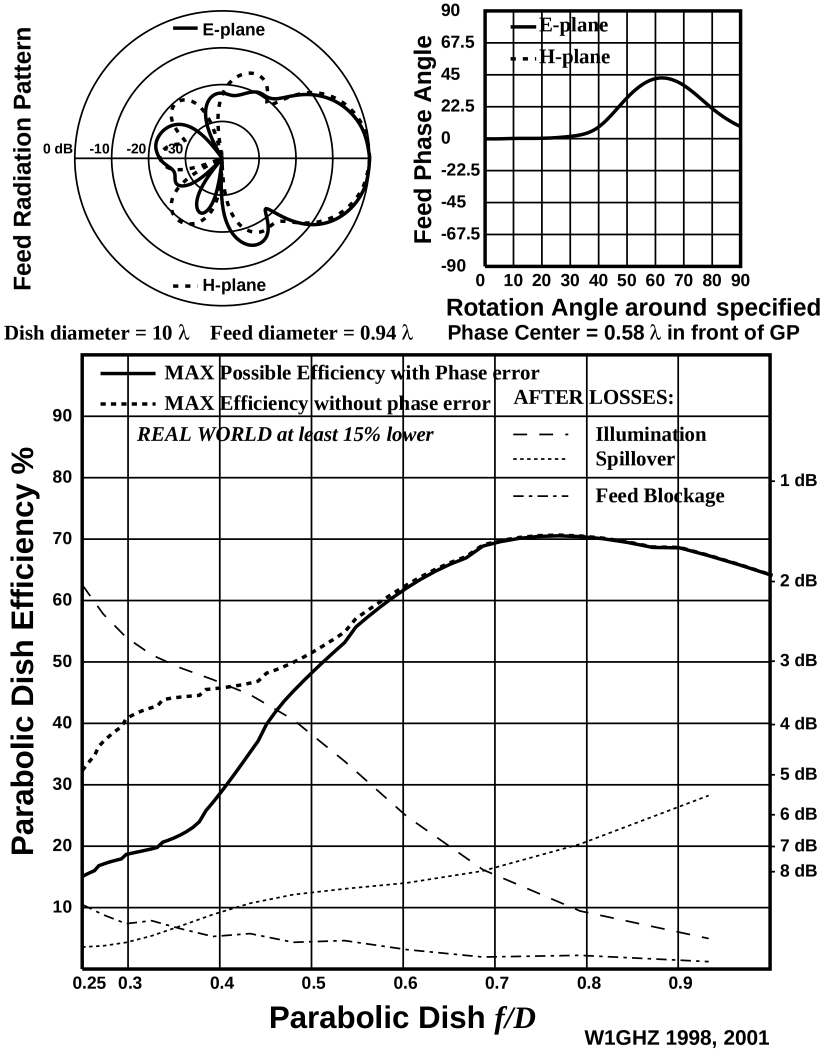
Helix 4 turns 12.5˚, 0.94λ GP, at 2.6 GHz

Figure 4d



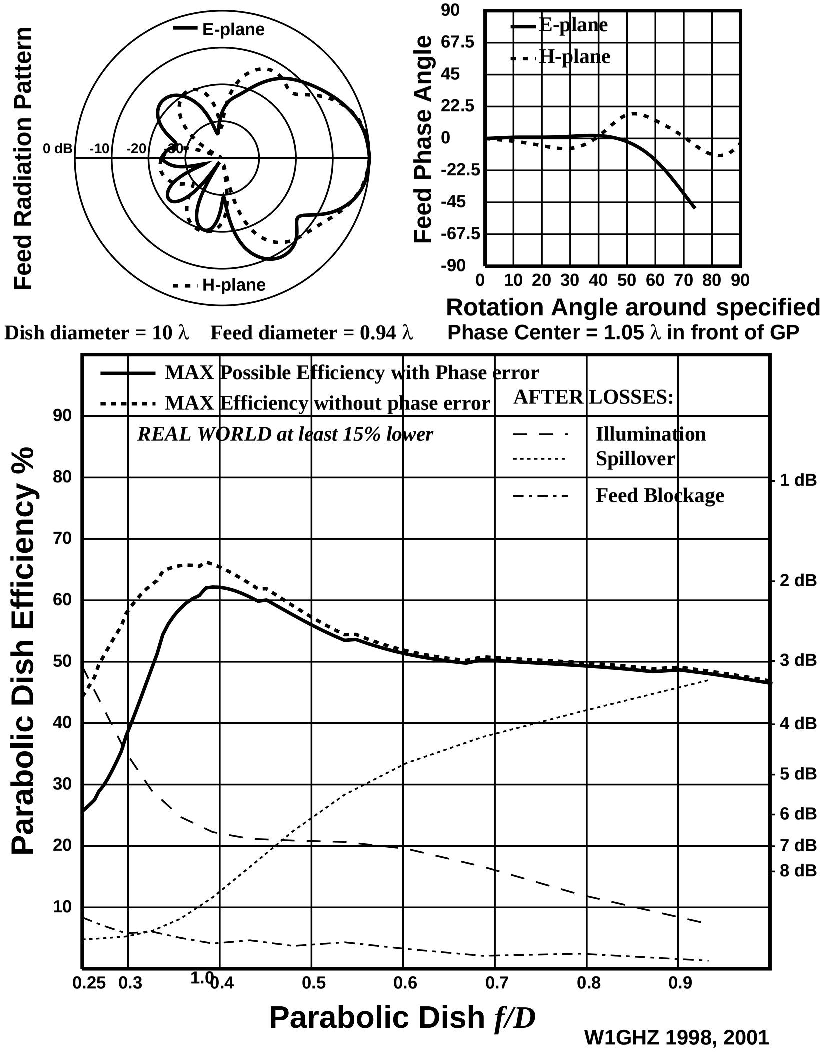
Helix 4 turns 12.5˚, 0.94λ GP, at 2.8 GHz

Figure 4e



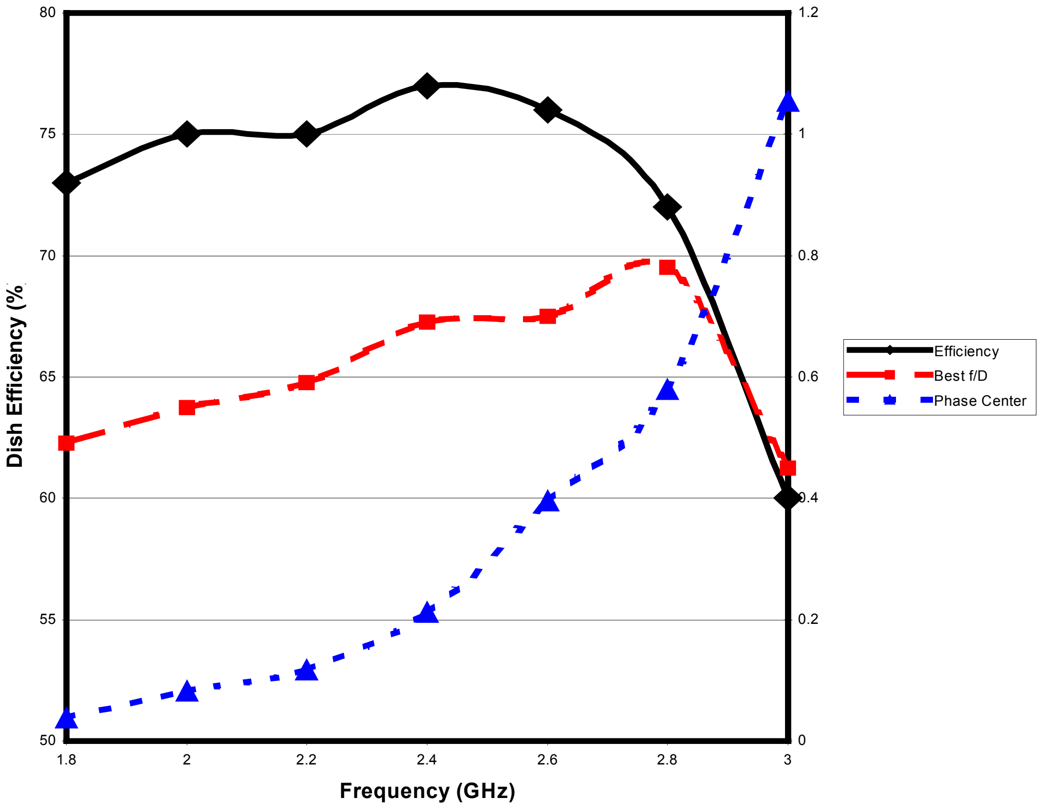
Helix 4 turns 12.5˚, 0.94λ GP, at 3.0 GHz

Figure 4f



The bandwidth of the helical feed can be seen from calculated radiation patterns over a 50% bandwidth, from 1.8 to 3.0 GHz, shown in Figure 4.

Figure 5: Helix feed - 4 turns, 12.5 deg, 0.94 lambda GP



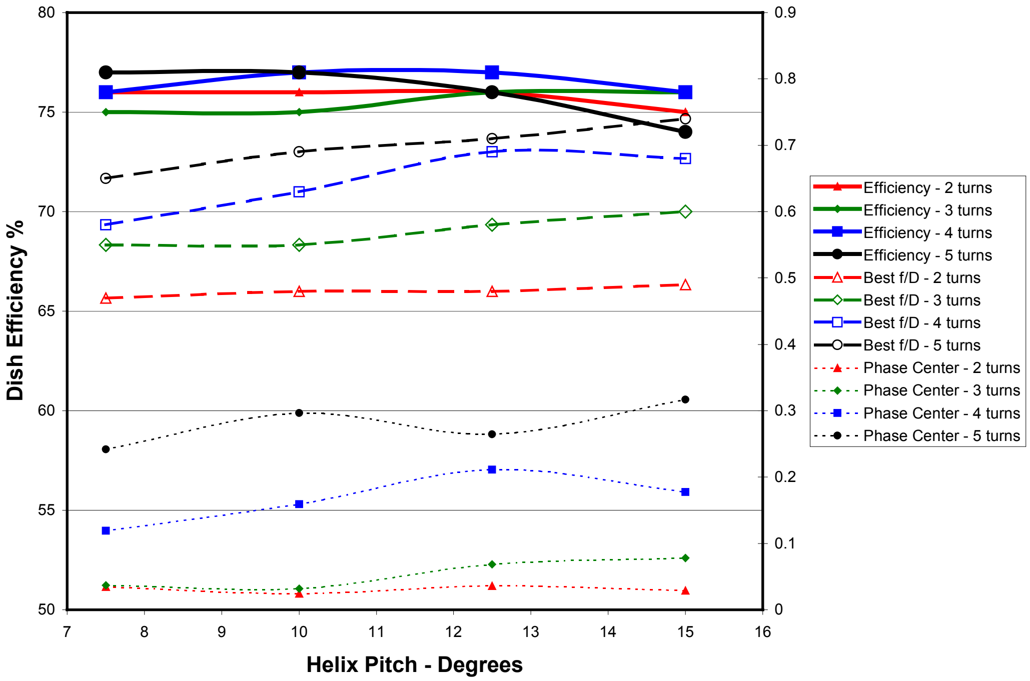
The calculated efficiency remains high from 2.0 to 2.6 GHz, about a 25% bandwidth. At the ends of the range, the efficiency falls off and the patterns deteriorate, with higher sidelobe levels, particularly at the higher-frequency end. Best **f/D** also varies with frequency. Figure 5 is a graph of efficiency and best **f/D** vs. frequency. Phase center is also plotted – it is reasonable constant over the lower half of the frequency range, but moves rapidly at the higher end of the range. We can conclude that this helical feed would work well on a single band, but would not provide good performance on any two adjacent ham bands.

The radiation patterns in Figures 2,3, and 4 were calculated using a 3D program; I have used both Zeland **Fidelity**[6] and Ansoft **HFSS**[7] programs to calculate helical antenna patterns. While both programs are quite expensive, the free **NEC2**[8] program will also do an excellent job on helical antennas; it simply lacks the graphical input and output capabilities. All the patterns in this paper are calculated, but many hams have reported good experimental results with helical feeds, so we have some assurance of validity.

Since my first helical feed calculations were so promising, I ran calculations over a range of dimensions to see if there are optimum combinations. By a serendipitous accident, I found that helix pitches much smaller than the recommended optimum pitch of 12 to 14 degrees seemed to work well, so I expanded the range to include pitches from 7.5 to 15 degrees and lengths from 2 to 5 turns. The helix dimensions were targeted for a center frequency of 2.4 GHz, and patterns calculated over 1.8 to 3.0 GHz.

The results, rather than finding any optimum, suggest that the helical antenna is a very forgiving feed – near the center design frequency, almost any dimensions will work to some degree. Figure 6 plots the efficiency, optimum *f*/D, and phase center at the center frequency for all of the helical feeds, and all are very good. Only the optimum *f*/D, varies; as expected, the narrower beamwidth of a longer helix provides the narrower illumination angle needed for a larger **f/D**. None of the combinations showed a significantly larger bandwidth than Figure 5.

Figure 6: Helix Feeds at 2.4 GHz

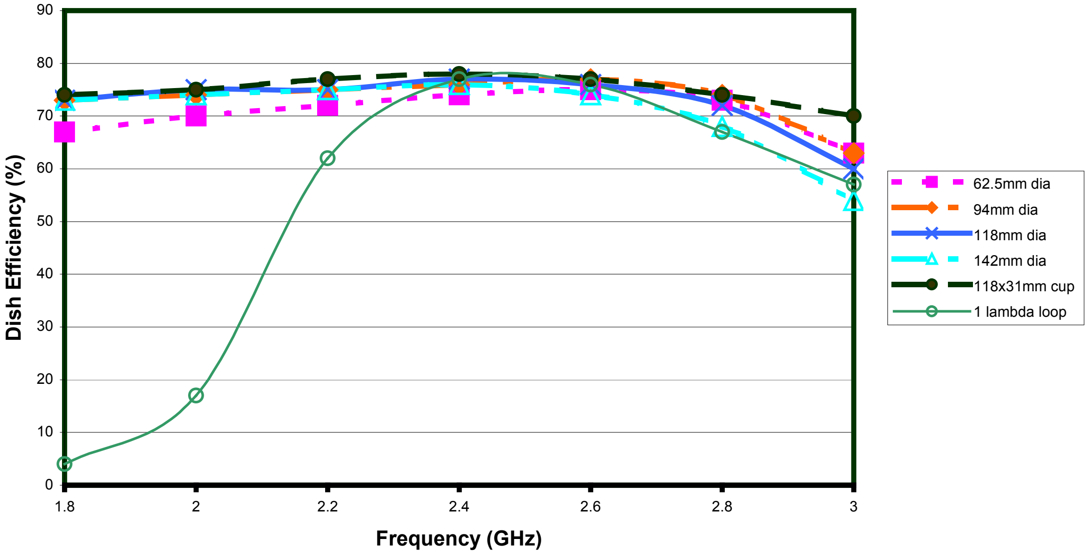


Since all the helical feeds are good, the design procedure is simple: pick a combination that is best for the ***f*/D** of your dish, and wind the helix. Then put the phase center of the feed at the focus of the dish. Figure 6 plots the phase center for all combinations at the center frequency. All are just in front of the ground plane, with the larger pitches a bit farther out. Since an offset-fed dish is more forgiving of phase-center error, placing the ground plane at the focus should be close enough. (For the finicky: the wire diameter for all calculations was 3 mm, and the helix started 4 mm in front of the ground plane, so that the beginning of the first turn was clear of the ground plane.)

Ground Plane variations

All of the variations of helix length and pitch shown in Figure 6 had a constant ground plane (GP) size, 118 mm, or 0.94λ diameter at the center frequency. Varying this diameter by ±20%, so that the diameter is 0.94λ at the highest or lowest frequency, had little effect, as shown in Figure 7.

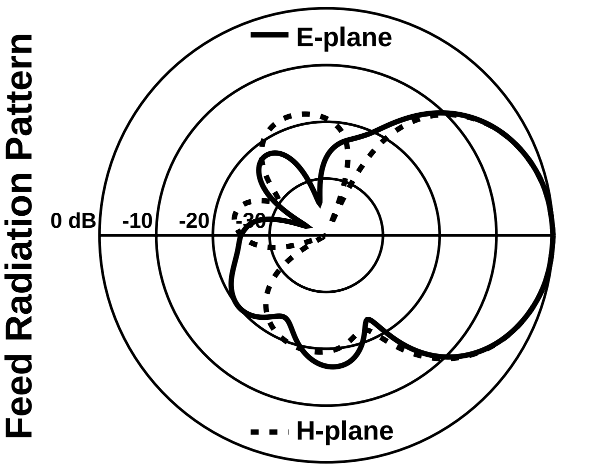
Figure 7: Helix feed (4t 12.5deg) with varying GP

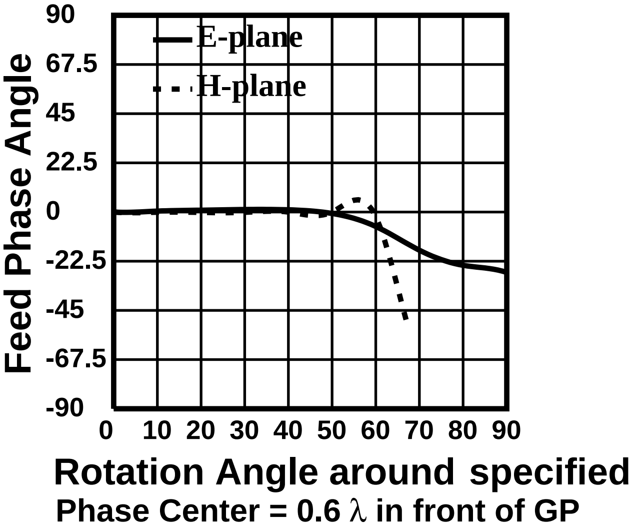


Reducing the diameter to 0.5λ lowered the efficiency slightly, while a cavity groundplane 0.94λ in diameter and λ/4 deep increased it slightly. The phase center and optimum ***f*/D** show only small variations near the center frequency. The only significant difference was found when the ground plane is replaced by a loop[1] 1λ in circumference, with a second loop behind it. The loop still provides high dish efficiency near the center frequency, but the bandwidth is much narrower, ~20%.

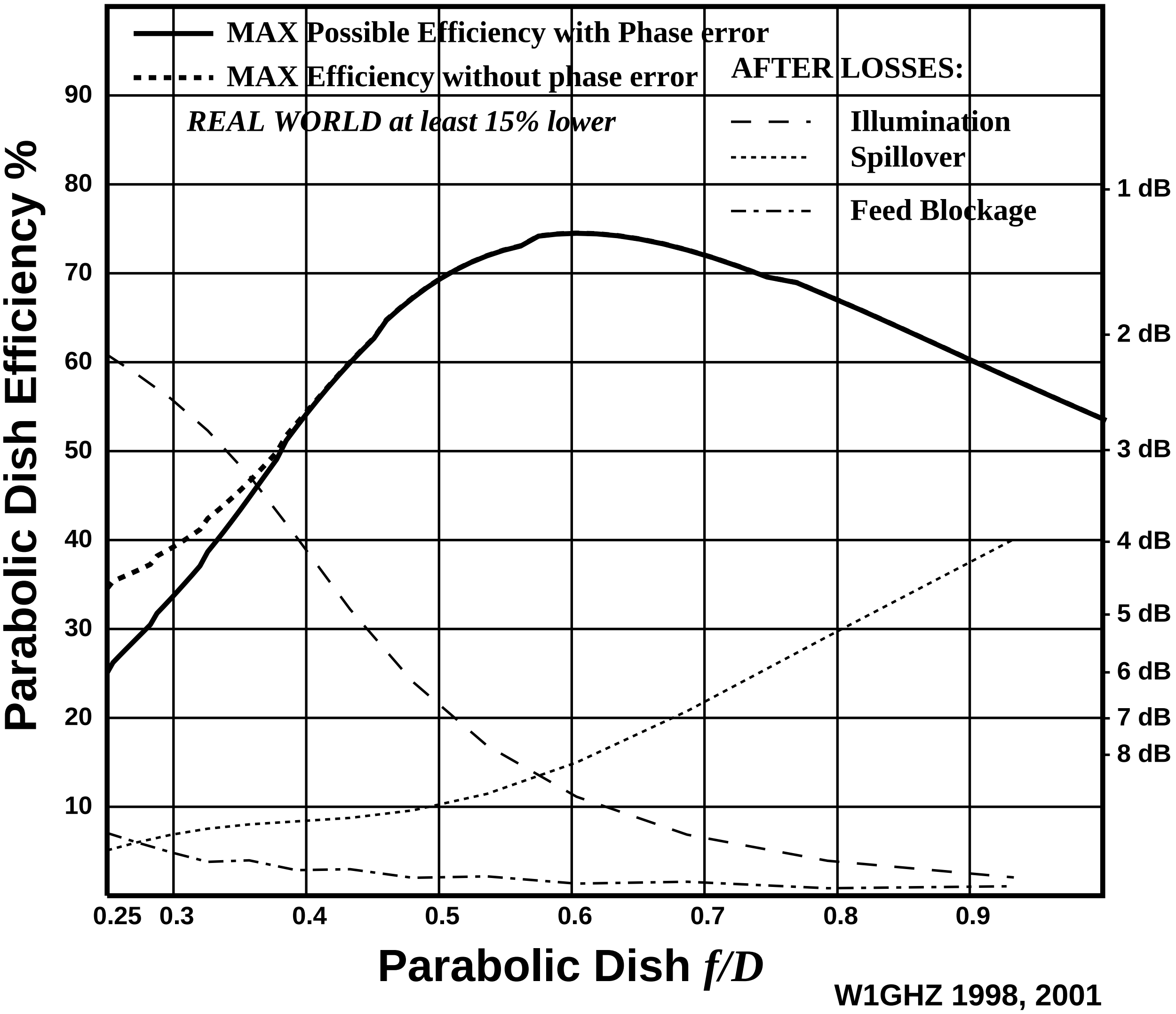
K5OE 2.4 GHz helix 5 turns, 13.43˚, 100x50mm cavity GP

Figure 8





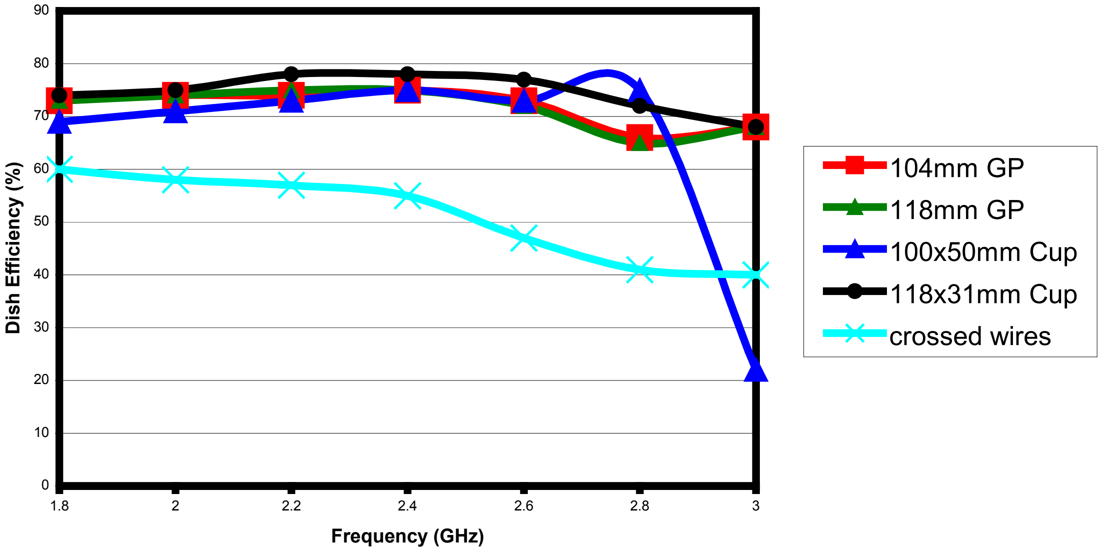
Dish diameter = 10 λ   Feed diameter = 0.8 λ



W1GHZ 1998, 2001

The cavity groundplane was suggested by K5OE, who used the dimensions suggested by Kraus[3]: 0.75λ diameter and 0.375λ deep, on a 5 turn helix with a pitch of 13.43°. This calculated patterns and efficiency curves for this helical antenna are shown in Figure 8. Efficiency is very good, but the bandwidth, shown in Figure 9, is narrower than with a flat ground plane.

Figure 9: K5OE 5-turn 13.4 degree Helix Feed - Efficiency

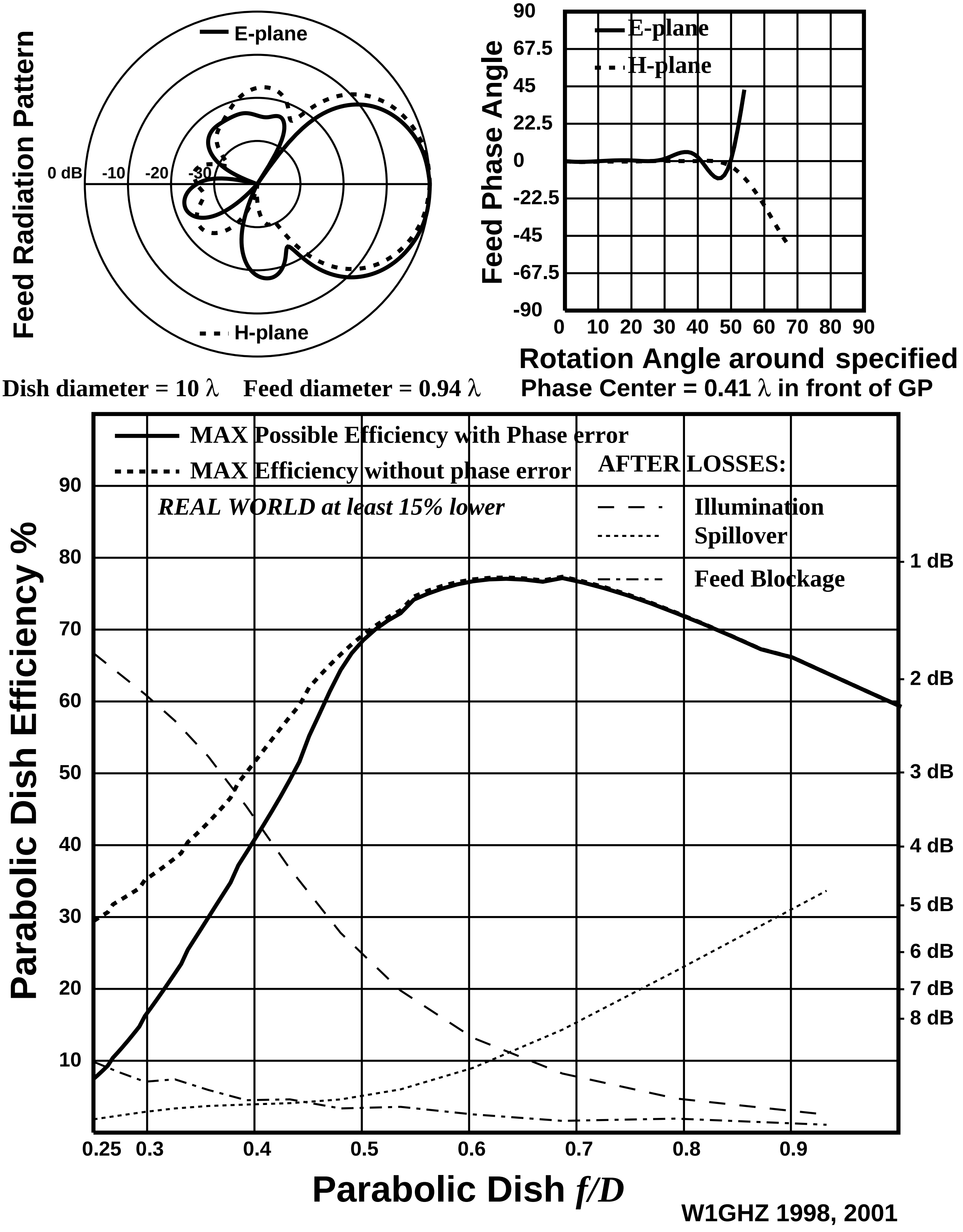


I tried some ground plane variations with this helix also. One was a larger, shallower cavity, 0.94λ in diameter and λ/4 deep. This provided slightly higher efficiency with much better bandwidth, as shown in Figure 9, and in the pattern and efficiency curves, Figure 10. Both cavity ground planes were slightly better than flat ones of the same diameter, also shown in Figure 9. The cavity ground planes reduce side and back lobes so that the efficiency is increased slightly, but the optimum ***f*/D** decrease – the effective length of the helix is only the part outside of the cavity. To feed an offset dish with a cavity-GP helix, we must increase the length to compensate – in this case, from about 4 turns to 6 turns with the deeper cavity or 5 turns with the shallower one. Figure 11 shows the radiation pattern and high efficiency of these two helix feed antennas.

A final ground plane experiment was a simple crossed wires 0.94λ long, like the reflector on a crossed Yagi-Uda antenna. The efficiency curve for the crossed-wire GP is significantly lower than the others in Figure 9.

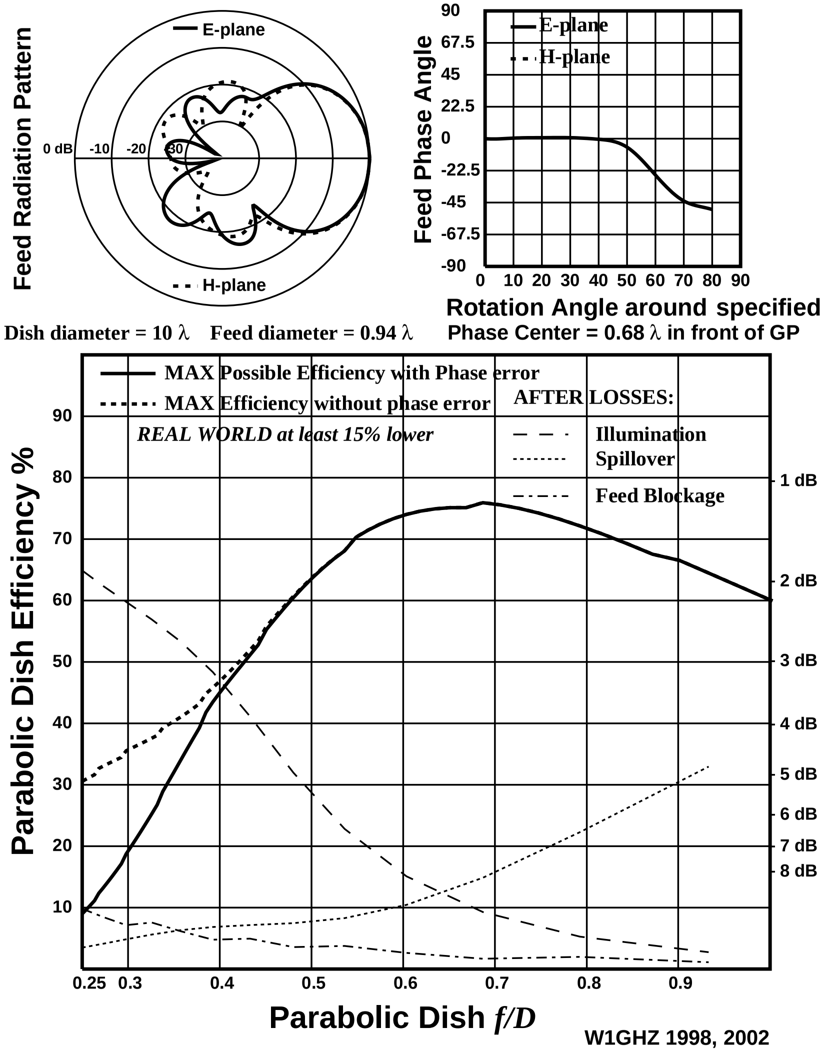
Helix 5 turns 13.43˚, 118x31mm cavity GP, at 2.4 GHz

Figure 10



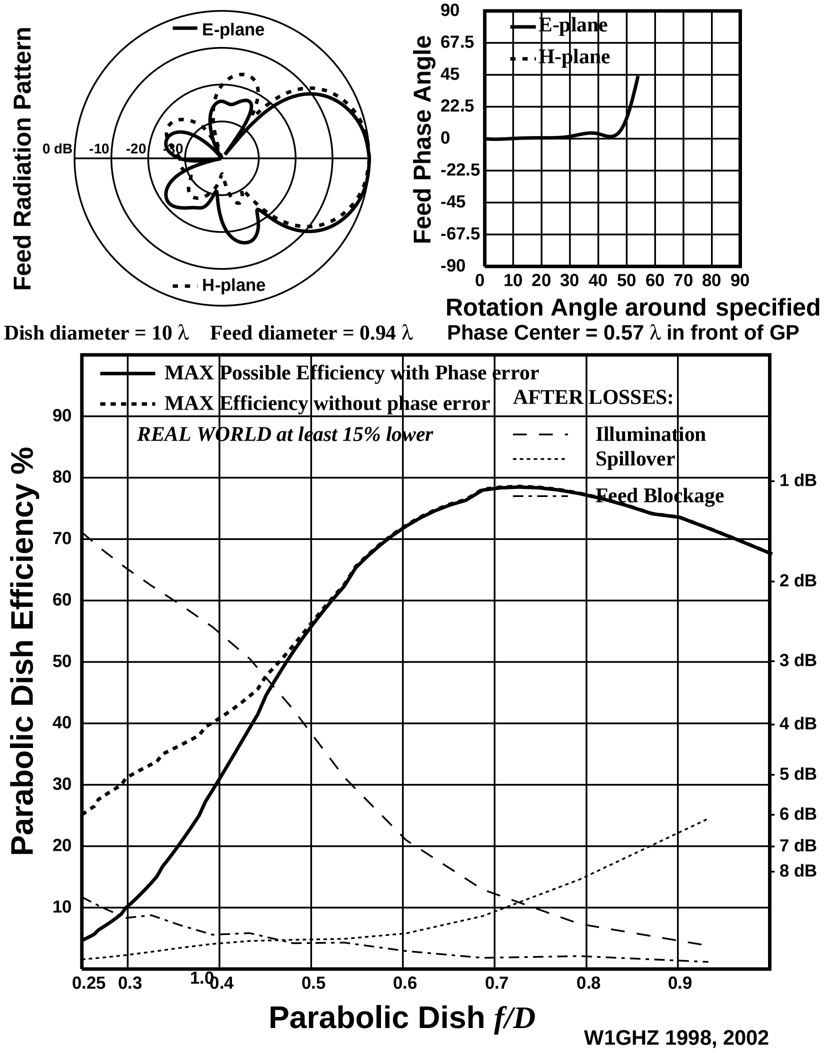
K5OE helix 6 turns, 13.43˚, 100x50mm cavity GP at 2.4 GHz

Figure 11a



K5OE helix 6 turns, 13.43˚, 118x31mm cavity GP at 2.4 GHz

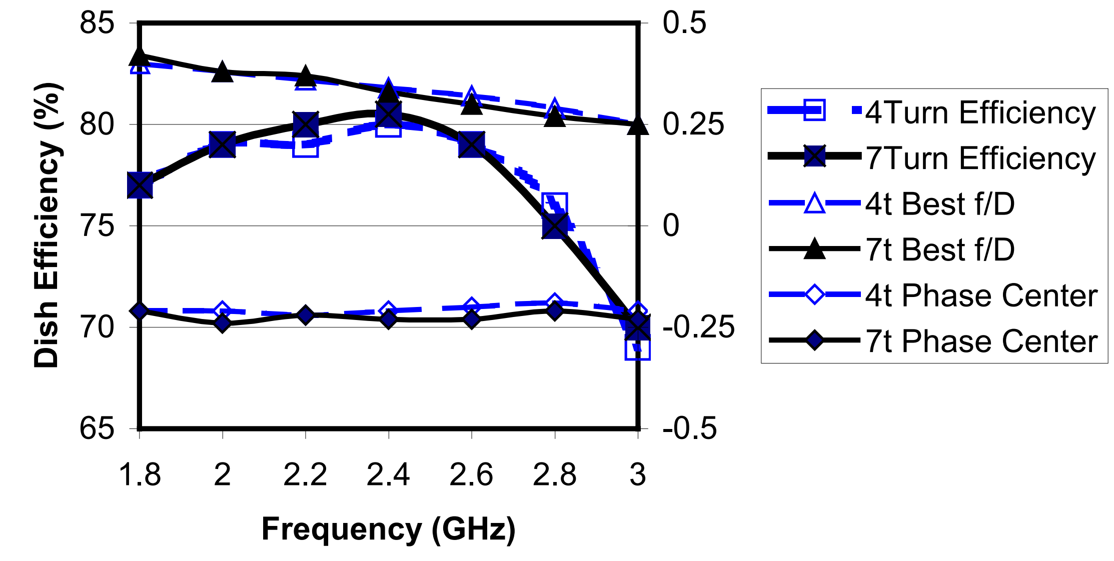
Figure 11b



Deep dishes

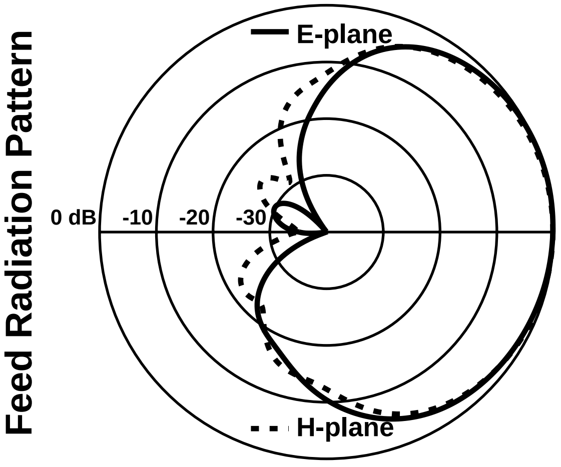
All the calculated helical feeds are only suitable for shallow dishes or offset-fed dishes, with **f/D** > 0.5, while most prime-focus dishes are deeper, with **f/D** = 0.4 or smaller. For shallow dishes, a different form of helix is needed. One possibility is a backfire helix[9], with a small loop instead of a ground plane – the loop is smaller in diameter than the helix diameter, like a director on a loop-Yagi. The radiation peak is toward the end with loop, and the beam is broader than a helix with ground plane. Figure 12 is the radiation pattern and calculated efficiency for a 7-turn helix with 14º pitch, with a loop 0.29λ in diameter. Calculated efficiency is 80% for an **f/D** = 0.33. Efficiency remains high at other frequencies, while best **f/D** decreases with increasing frequency, as shown in Figure 13. Thus, it might be possible to match the reflector **f/D** by dimensioning the helix for a different center frequency. The circular polarization of the backfire helix is reversed from the polarization sense of the same helix with a larger ground plane, radiating forward.

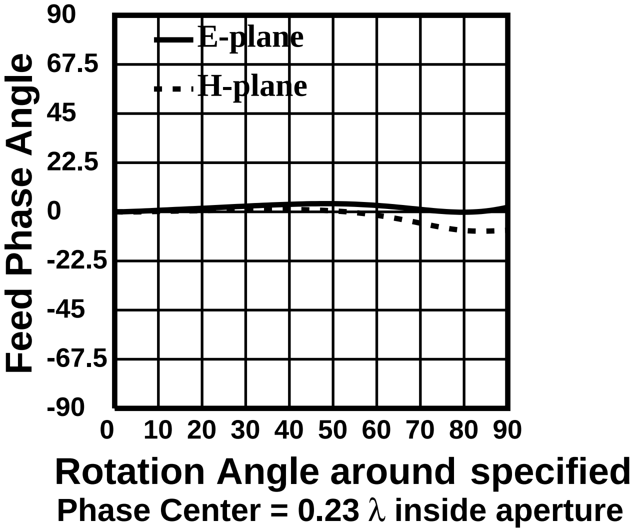
Figure 13: Backfire Helix Feed - 14 deg, 0.29 lambda GP



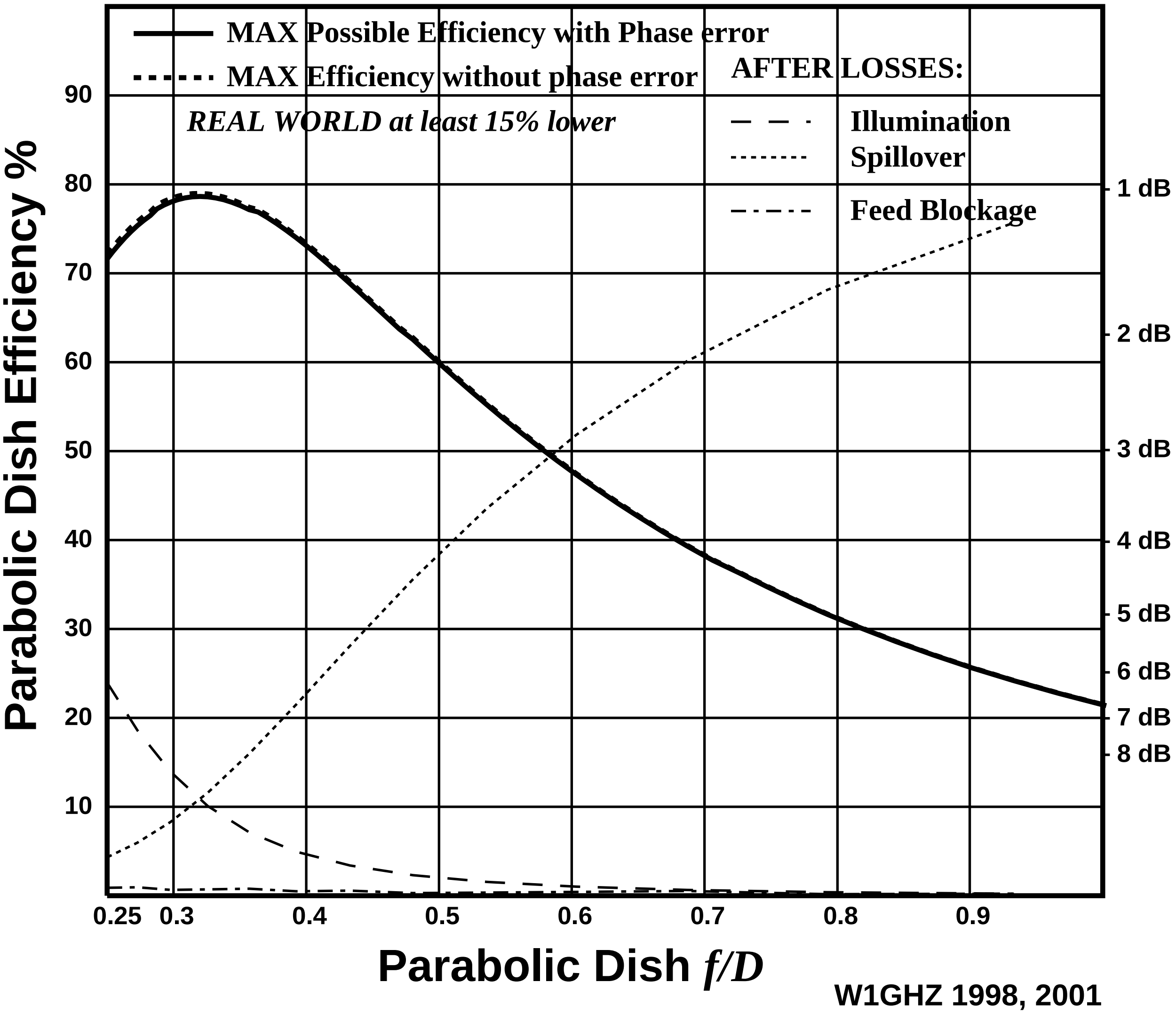
Backfire helix feed, 7 turns 14˚, 0.29λ GP, at 2.4 GHz

Figure 12



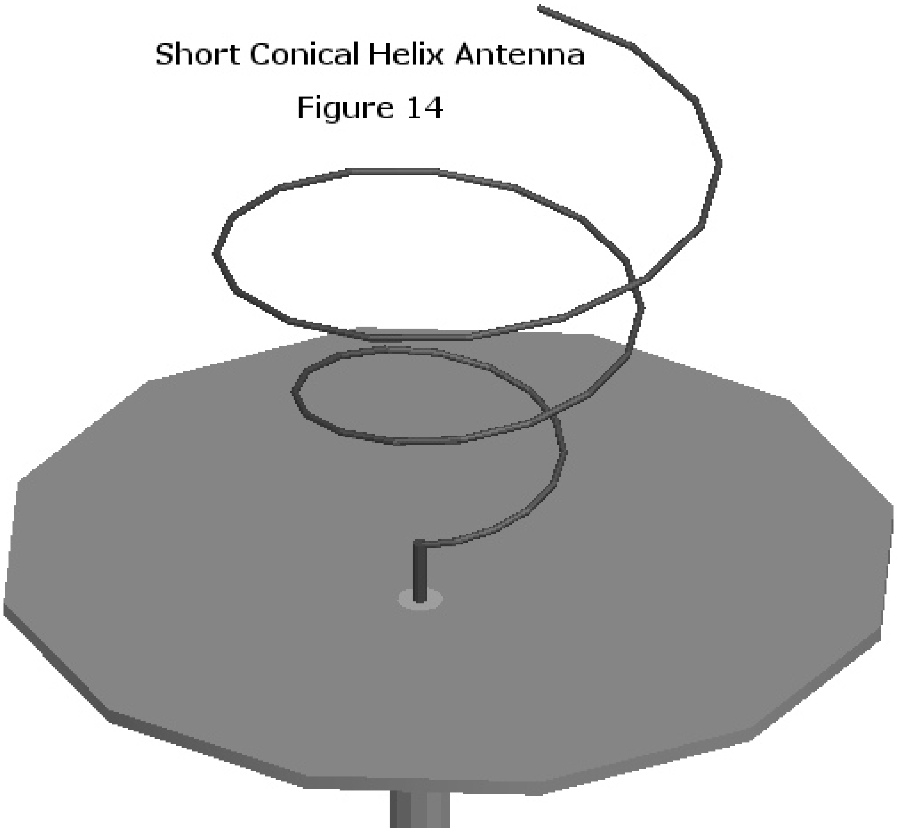


Dish diameter = 10 λ   Feed diameter = 0.5 λ



W1GHZ 1998, 2001

Another feed for deep dishes is the short conical helix[10], with the helix diameter continuously increasing with distance from the ground plane, as shown in Figure 14. I scaled the 4 GHz feed from the original paper to 2.4 GHz, and changed the infinite ground plane to a more realizable 0.94λ in diameter. This makes a pretty good feed for an ***f*/D** around 0.4, usable for many common prime-focus dishes. The calculated radiation patterns and efficiency are shown in Figure 15 at a frequency of 2.0 GHz, where the performance seemed best. Efficiency was good from 1.6 to 2.4 GHz, but circular polarization was good over a much narrower bandwidth, from about 1.8 to 2.2 GHz. If you experiment with a short conical helix feed, be sure to check the polarization circularity at the operating frequency.



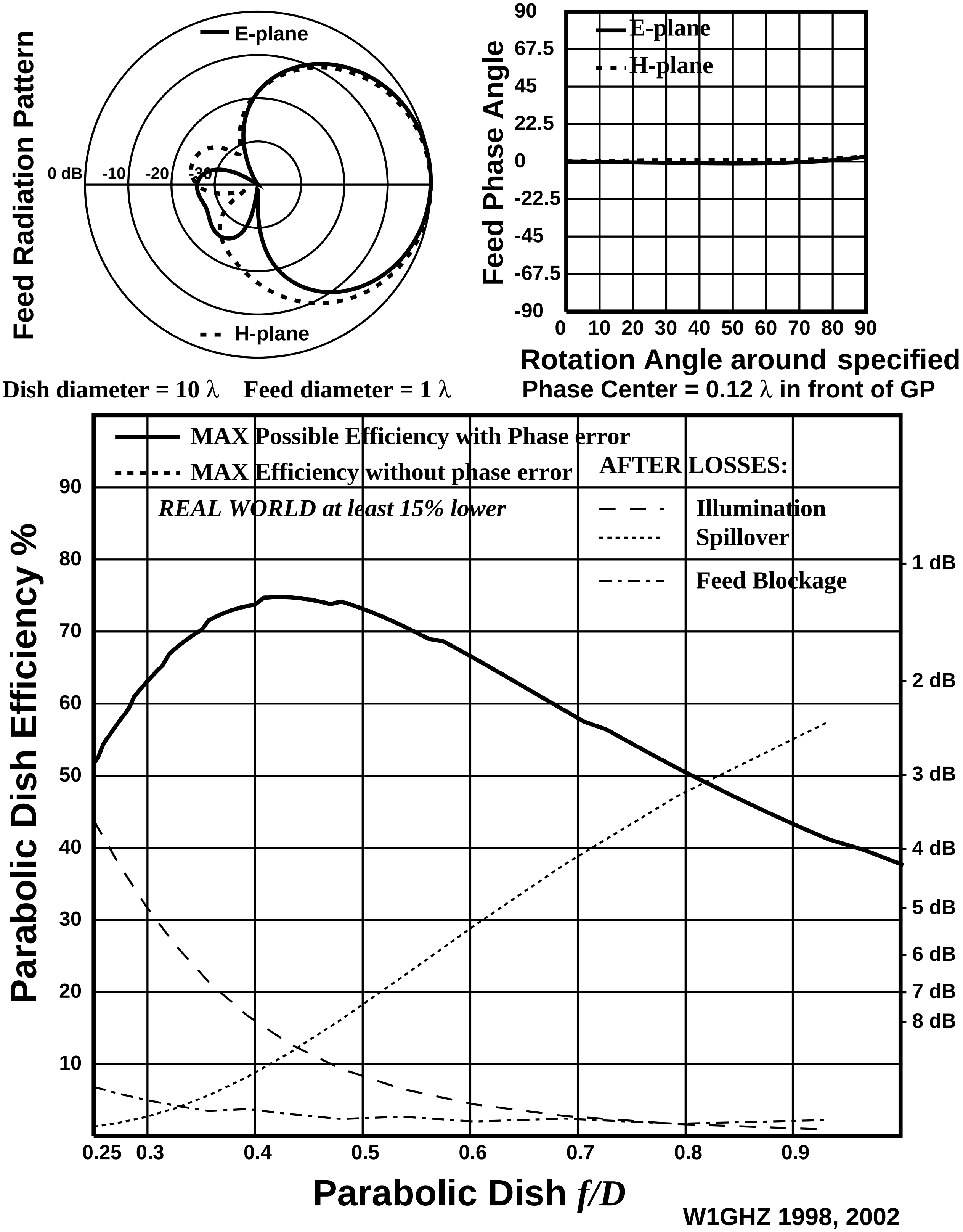
Short Conical Helix Antenna

Figure 14

Another possible feed for deep dishes might be a quadrifilar helix. I don't have any patterns for these feeds yet.

Short Conical Helix, 90˚, at 2.0 GHz

Figure 15



Dish diameter = 10 λ   Feed diameter = 1 λ

Phase Center = 0.12 λ in front of GP

W1GHZ 1998, 2002

Mechanical considerations

In most cases, a helix made of copper or aluminum wire is not self-supporting, particularly in New England weather. Many helical antenna photographs show a support in the center: one version has a metal center pole with periodic supports for the helix. Another variation winds the wire, like Figure 16, or a flat tape, on a dielectric support. Kraus[3] says the dielectric shifts the operating bandwidth to lower frequencies, so that a smaller helix is needed for a given frequency.

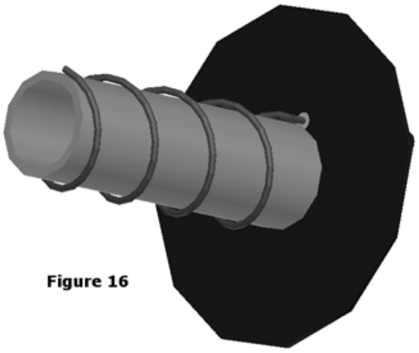
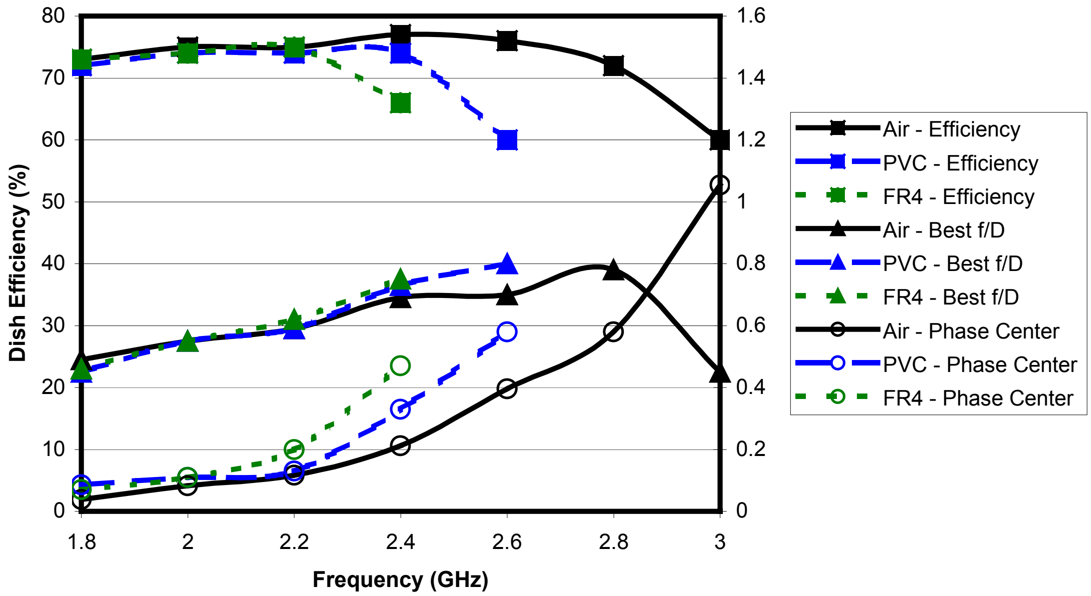


Figure 16

Plastic tubing is readily available in PVC and Fiberglass (FR4), so I calculated patterns for a 4 turn, 12.5° pitch helix with each of these materials. The wall thickness was 3mm, or about 1/8 inch.

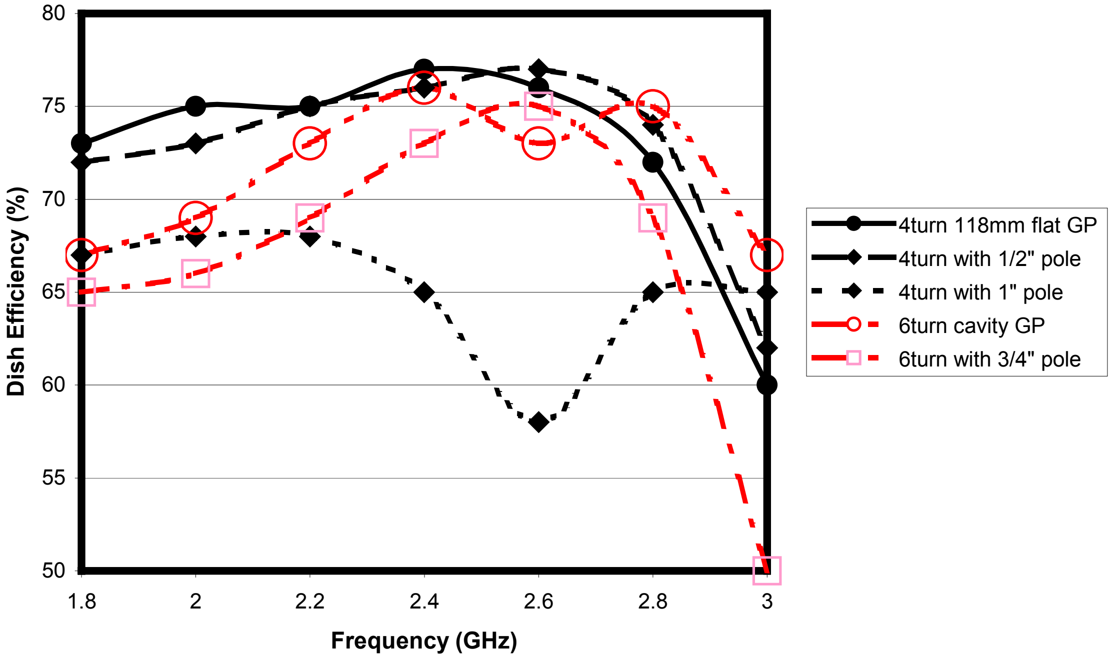
Figure 17: Helix feed with dielectric support tube



The efficiency with the two dielectric tubes is compared to a helix with no support in Figure 17 and shows a definite decrease in the maximum frequency, about 13% for the PVC and about 20% for the fiberglass. Thus, the size of a helix antenna using these support tubes should probably be reduced accordingly.

I also calculated patterns with a metal center pole, assuming that the support points are small enough to ignore. The 4-turn, 12.5º helix of Figure 4 showed little change with a ½" (12.7 mm) diameter pole inside the 40 mm diameter helix, but a 1" (25.4 mm) diameter pole significantly reduced the efficiency. Figure 18 adds curves for both poles to Figure 4. The length of the pole had little effect, so the pole can be short, just supporting the helix, or long enough to support the feed on the dish.

Figure 18: Helix feeds with central support pole



A second example adds a ¾" (19 mm) pole to the K5OE helical feed, with little change in efficiency, as shown in Figure 18.

We can conclude that a central pole with a diameter less than half the helix diameter does not significantly degrade performance, as long as the supports for the helix wire are small and infrequent.

Feed impedance

A typical helical antenna has an input impedance of around 140 ohms. Kraus[3] gives a nominal impedance of $Z = 140C_\lambda$ with axial feed. This is a resistive impedance only at one frequency, probably near the center frequency. Matching the impedance to 50 ohms over a broad bandwidth would be more difficult than simply matching it well for a ham band. A simple quarter-wave matching section with a Zo ~ 84 ohms should do the trick for a single band. The matching section[10] is often part of the helix: a quarter-wave of wire close to the ground plane before the first turn starts. It could also be on the other side of the ground plane, to separate impedance matching from the radiating element.

Polarization

Circular polarization has two possible senses: right-hand (RHCP) and left-hand (LHCP). Since a helix cannot switch polarization, it is important to get it right: by the IEEE definition[3], RHCP results when the helix is wound as though it were to fit in the threads of a large screw with normal right-hand threads. Note that the classical optics definition of polarization is opposite to the IEEE definition.

More important for a feed is that the sense of the polarization reverses on reflection, so that for a dish to radiate RHCP polarization requires a feed with LHCP. For EME, reflection from the moon also reverses circular polarization, so that the echo returns with polarization reversed from the transmitted polarization. A helical feed used for EME would not be able to receive its own echoes because of cross-polarization loss.

Summary

The helical antenna is an excellent feed for circular polarization. It is broadband and dimensions are not critical, and the patterns are well-suited to illumination of offset dishes. It is a particularly good feed for small offset dishes for satellite applications.

References

1. Kraus, J.D., (W8JK), "A Helical-Beam Antenna Without a Ground Plane," *IEEE Antennas and Propagation Magazine*, April 1995, p. 45.
2. Kraus, J.D., *Antennas*, McGraw-Hill, 1950.
3. Kraus, J.D. & Marhefka, R.J., *Antennas: for All Applications, third edition*, McGraw-Hill, 2002.
4. Emerson, D., AA7FV, "The Gain of the Axial-Mode Helix Antenna," *Antenna Compendium Volume 4*, ARRL, 1995, pp. 64-68.
5. http://ourworld.compuserve.com/homepages/demerson/helix.htm or http://www.tuc.nrao.edu/~demerson/helixgain/helix.htm

6. www.zeland.com
7. www.ansoft.com
8. The *unofficial* Numerical Electromagnetic Code (NEC) Archives, by WB6TPU, http://www.qsl.net/wb6tpu/swindex.html
9. Nakano, H., Yamauchi, J., & Mimaki, H., "Backfire Radiation from a Monofilar Helix with a Small Ground Plane," *IEEE Transactions on Antennas and Propagation*, October 1988, pp. 1359-1364.
10. Nakano, H., Mikawa, T., & Yamauchi, J., "Investigation of a Short Conical Helix Antenna," *IEEE Transactions on Antennas and Propagation*, October 1985, pp. 1157-1160.
11. Kraus, J.D., "A 50-Ohm Impedance for Helical Beam Antennas," *IEEE Transactions on Antennas and Propagation*, November 1977, p. 913.

2. Survey of Helical Antennas

2.1 Introduction

The helical antenna is a hybrid of two simple radiating elements, the dipole and loop antennas. A helix becomes a linear antenna when its diameter approaches zero or pitch angle goes to 90°. On the other hand, a helix of fixed diameter can be seen as a loop antenna when the spacing between the turns vanishes $(a = 0^\circ)$.

Helical antennas have been widely used as simple and practical radiators over the last five decades due to their remarkable and unique properties. The rigorous analysis of a helix is extremely complicated. Therefore, radiation properties of the helix, such as gain, far-field pattern, axial ratio, and input impedance have been investigated using experimental methods, approximate analytical techniques, and numerical analyses. Basic radiation properties of helical antennas are reviewed in this chapter.

The geometry of a conventional helix is shown in Figure 2.1a. The parameters that describe a helix are summarized below.

D = diameter of helix
S = spacing between turns
N = number of turns
C = circumference of helix = pD
A = total axial length = NS
a = pitch angle

If one turn of the helix is unrolled, as shown in Figure 2.1(b), the relationships between S, C, a and the length of wire per turn, L, are obtained as:

$$S = L \sin a = C \tan a$$

$$L = (S^2 + C^2)^{1/2} = (S^2 + p^2 D^2)^{1/2}$$

2.2 Modes of Operation

2.2.1 Transmission Modes

An infinitely long helix may be modeled as a transmission line or waveguide supporting a finite number of modes. If the length of one turn of the helix is small compared to the wavelength, $L << l$, the lowest transmission mode, called the T_0 mode, occurs. Figure 2.2a shows the charge distribution for this mode.

When the helix circumference, C, is of the order of about one wavelength $(C \approx 1l)$, the second-order transmission mode, referred to as the T_1 mode, occurs. The charge distribution associated with the T_1 mode can be seen in Figure 2.2b. Higher-order modes can be obtained by increasing of the ratio of circumference to wavelength and varying the pitch angle.

2.2.2 Radiation Modes

When the helix is limited in length, it radiates and can be used as an antenna. There are two radiation modes of important practical applications, the normal mode and the axial mode. Important properties of normal-mode and axial-mode helixes are summarized below.

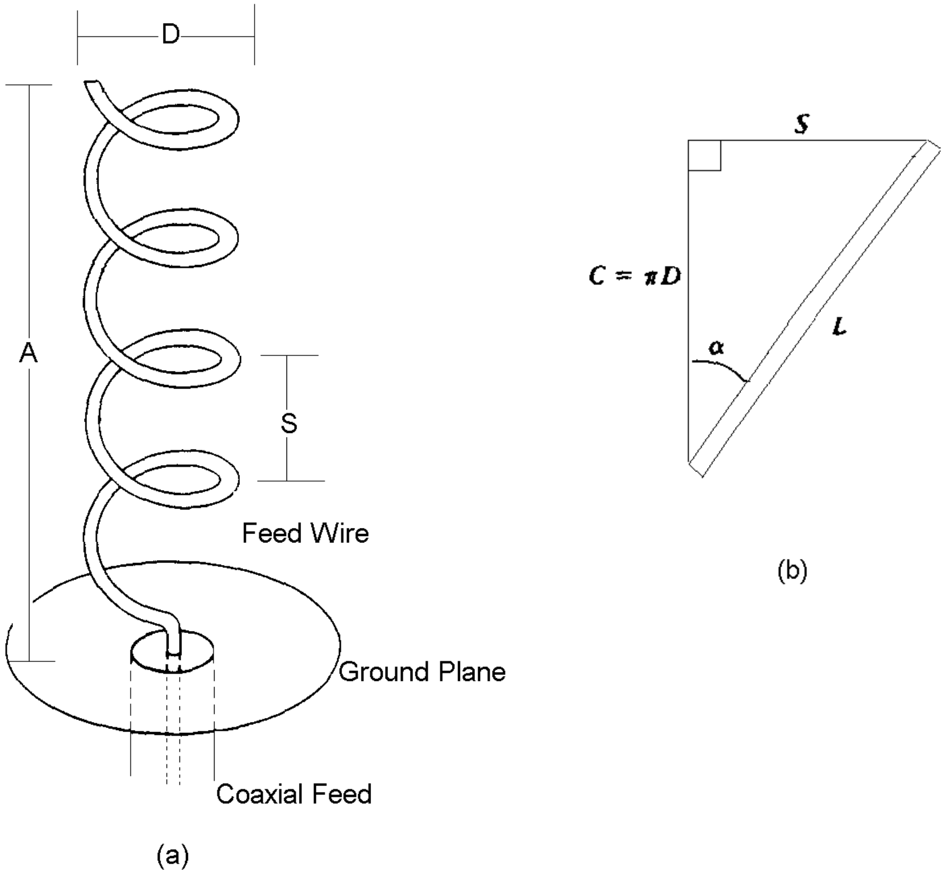


Figure 2.1 (a) Geometry of helical antenna; (b) Unrolled turn of helical antenna

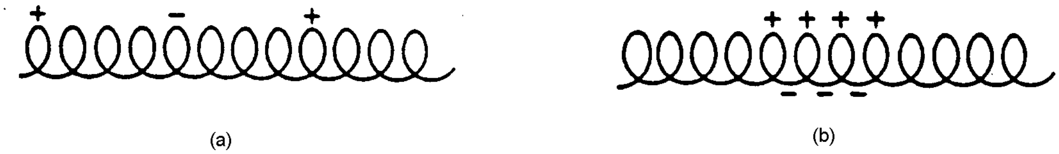


Figure 2.2 Instantaneous charge distribution for transmission modes: (a) The lowest-order mode (T_0); (b) The second-order mode (T_1)

2.2.2.1 Normal Mode

For a helical antenna with dimensions much smaller than wavelength $(NL << l)$, the current may be assumed to be of uniform magnitude and with a constant phase along the helix [5]. The maximum radiation occurs in the plane perpendicular to the helix axis, as shown in Figure 2.3a. This mode of operation is referred to as the “normal mode”. In general, the radiation field of this mode is elliptically polarized in all directions. But, under particular conditions, the radiation field can be circularly polarized. Because of its small size compared to the wavelength, the normal-mode helix has low efficiency and narrow bandwidth.

2.2.2.2 Axial Mode

When the circumference of a helix is of the order of one wavelength, it radiates with the maximum power density in the direction of its axis, as seen in Figure 2.3b. This radiation mode is referred to as “axial mode”. The radiation field of this mode is nearly circularly polarized about the axis. The sense of polarization is related to the sense of the helix winding.

In addition to circular polarization, this mode is found to operate over a wide range of frequencies. When the circumference (C) and pitch angle (a) are in the ranges of $\frac{3}{4} < \frac{C}{l} < \frac{4}{3}$ and $12^\circ < a < 15^\circ$ [6], the radiation characteristics of the axial-mode helix remain relatively constant. As stated in [7] ,“if the impedance and the pattern of an antenna do not change significantly over about one octave $(\frac{f_u}{f_l} = 2)$ or more, we will classify it as a broadband antenna”. It is noted that the ratio of the upper frequency to the lower frequency of the axial-mode helix is equal to $\frac{f_u}{f_l} = \frac{4/3}{3/4} = 1.78$. This is close to the definition of broadband antennas. For the reason that the axial-mode helix possesses a

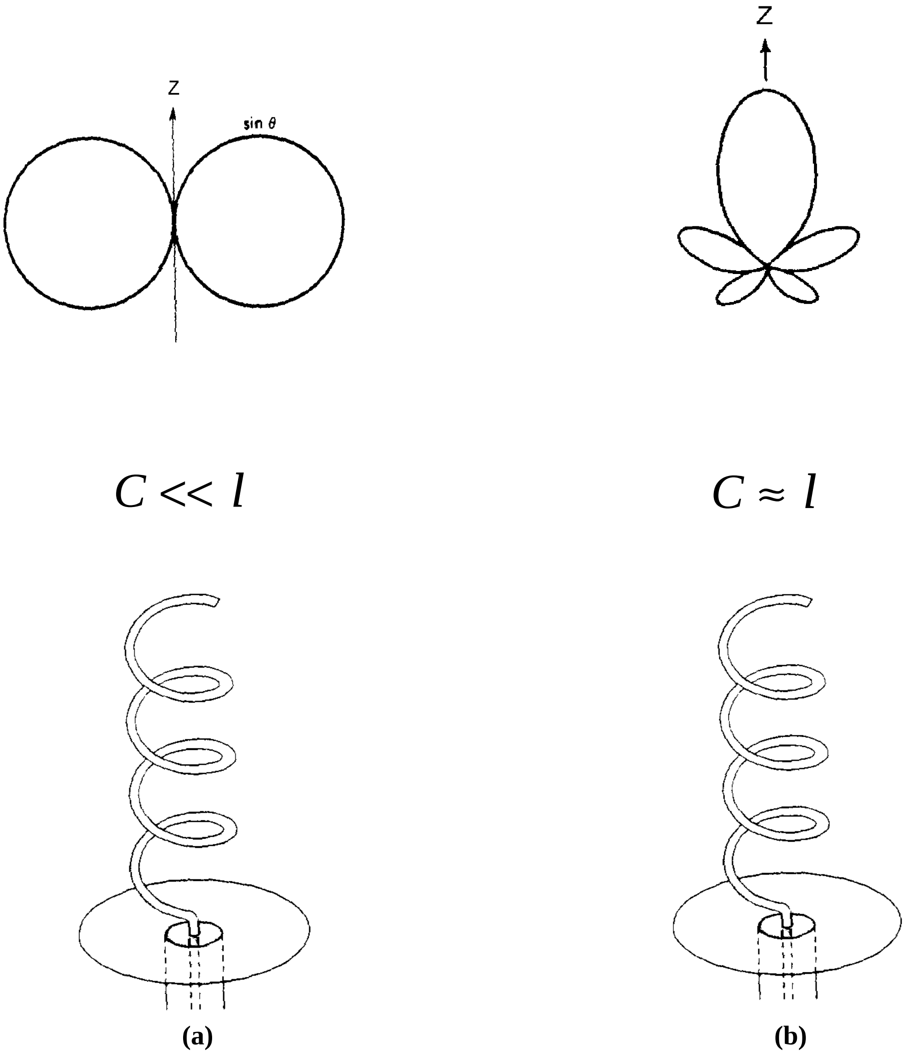


Figure 2.3 Radiation patterns of helix: (a) Normal mode; (b) Axial mode

number of interesting properties, including wide bandwidth and circularly polarized radiation, it has found many important applilcations in communication systems.

2.3 Analysis of Helix

Unlike the dipole and loop antennas, the helix has a complicated geometry. There are no exact solutions that describe the behavior of a helix. However, using experimental methods and approximate analytical or numerical techniques, it is possible to study the radiation properties of this antenna with sufficient accuracy. This section briefly discusses the analysis of normal-mode and axial-mode helices.

2.3.1 Normal-Mode Helix

The analysis of a normal-mode helix is based on a uniform current distribution over the length of the helix. Furthermore, the helix may be modeled as a series of small loop and short dipole antennas as shown in Figure 2.4. The length of the short dipole is the same as the spacing between turns of the helix, while the diameter of the loop is the same as the helix diameter.

Since the helix dimensions are much smaller than wavelength, the far-field pattern is independent of the number of turns. It is possible to calculate the total far-field of the normal-mode helix by combining the fields of a small loop and a short dipole connected in series. Doing so, the result for the electric field is expressed as [6]

$$\vec{\mathrm{E}} = jh\frac{kI_0 e^{-jkr}}{4pr}\sin q\,(S\hat{q} - j\frac{p^2 D^2}{2l}\hat{f})\,, \qquad (2.1)$$

where $k = \frac{2p}{l}$ is the propagation constant, $h = \sqrt{\frac{m}{e}}$ is the intrinsic impedance of the medium, and I_0 is a current amplitude. As noted in (2.1), the q and f components of the field are in phase quadrature. Generally, the polarization of this mode is elliptical with an axial ratio given by

$$AR = \frac{\left|E_q\right|}{\left|E_f\right|} = \frac{2Sl}{p^2 D^2} . \qquad (2.2)$$

The normal-mode helix will be circularly polarized if the condition $AR = 1$ is satisfied. As seen from (2.2), this condition is satisfied if the diameter of the helix and the spacing between the turns are related as

$$C = \sqrt{2Sl} . \qquad (2.3)$$

It is noted that the polarization of this mode is the same in all directions except along the z-axis where the field is zero. It is also seen from (2.1) that the maximum radiation occurs at $q = 90^\circ$; that is, in a plane normal to the helix axis.

2.3.2 Axial-Mode Helix

Unlike the case of a normal-mode helix, simple analytical solutions for the axial-mode helix do not exist. Thus, radiation properties and current distributions are obtained using experimental and approximate analytical or numerical methods.

The current distribution of a typical axial-mode helix is shown in Figure 2.5 [5]. As noted, the current distribution can be divided into two regions. Near the feed region, the current attenuates smoothly to a minimum, while the current amplitude over the remaining length of the helix is relatively uniform. Since the near-feed region is small compared to the length of the helix, the current can be approximated as a travelling wave of constant amplitude. Using this approximation, the far-field pattern of the axial-mode helix can be analytically determined. There are two methods for the analysis of far-field pattern. In the first method, an N-turn helix is considered as an array of N elements with an element spacing equal to S. The total field pattern is then obtained by multiplying the pattern of one turn of the helix by the array factor. The result is

$$F(q) = c_0 \cos q \, \frac{\sin(N\, {}^{y}\!/_{2})}{\sin({}^{y}\!/_{2})}, \qquad (2.4)$$

where c_0 is a constant coefficient and $y = kS\cos q + a$. Here, a is the phase shift between successive elements and is given as

$$a = -2p - \frac{p}{N}. \qquad (2.5)$$

In (2.4), $\cos q$ is the element pattern and $\frac{\sin({}^{Ny}\!/_{2})}{\sin({}^{y}\!/_{2})}$ is the array factor for a uniform array of N equally-spaced elements. As noted from (2.5), the Hansen-Woodyard condition is satisfied. This condition is necessary in order to achieve agreement between the measured and calculated patterns.

In a second method, the total field is directly calculated by integrating the contributions of the current elements from one end of the helix to another. The current is assumed to be a travelling wave of constant amplitude. The current distribution at an arbitrary point on the helix is written as [6]

$$\vec{\mathrm{I}}(l) = \mathrm{I}_0 \exp(-jgf')\hat{\mathrm{I}}, \qquad (2.6)$$

where

$l =$ the length of wire from the beginning of the helix to an arbitrary point

$g = \dfrac{wL_T}{pcf'_m}$

$L_T =$ the total length of the helix

$p =$ phase velocity of wave propagation along the helix relative to the velocity of light,c

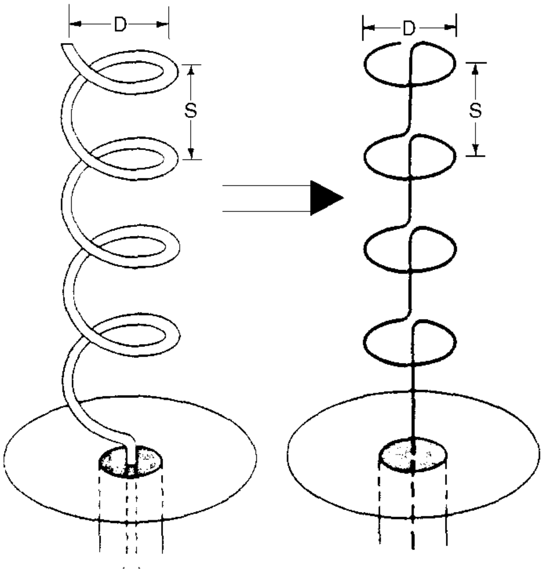


Figure 2.4 Approximating the geometry of normal-mode helix

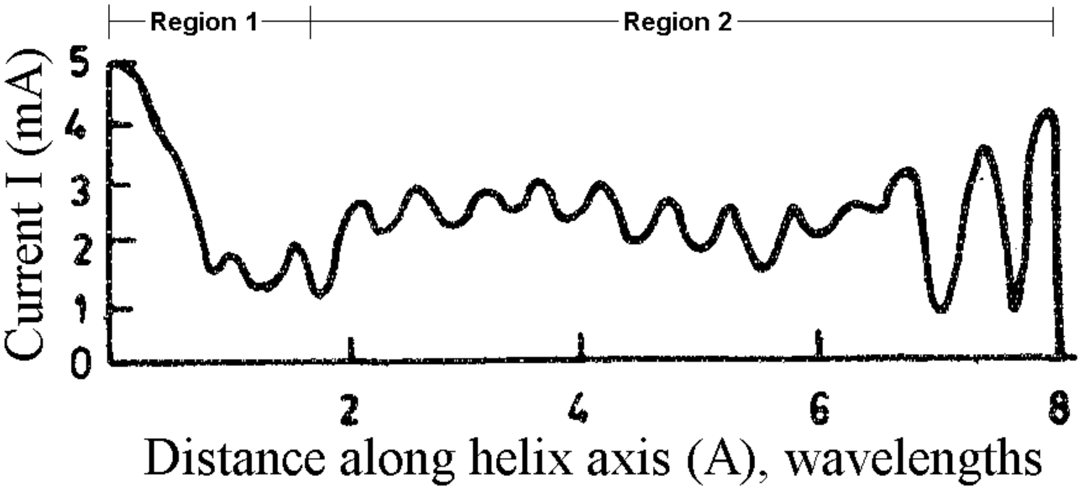


Figure 2.5 Measured current distribution on axial-mode helix [5]

$$= \frac{1}{\sin a + \left[{(2N+1)}\big/{N} \right] \left(l \cos a \right)\big/{C}}$$ (according to Hansen-Woodyard condition)

$$= 2pN$$

$f' =$ azimuthal coordinate of an arbitrary point

$\hat{I} =$ unit vector along the wire

$= -\hat{x}\sin f' + \hat{y}\cos f' + \hat{z}\sin a$

The magnetic vector potential at an arbitrary point in space is obtained as [6]

$$\vec{A}(\bar{r}) = \frac{ma I_0 \exp(-jkr)}{4pr} \int_0^{f'_m} \exp[ju\cos(f - f')]\exp(jdf')\hat{I}df' \text{ ,(2.7)}$$

Where $u = ka\sin q$

$a =$ radius of the helix

$d = B - g$

$B = ka\cos q \tan a$

Finally, the far-field components of the electric field, E_q and E_f , can be expressed as

$$E_q = -jw[(A_x\cos f + A_y\sin f)\cos q + A_z\sin q], \qquad (2.8)$$

$$E_f = -jw(A_y\cos f - A_x\sin f). \qquad (2.9)$$

2.3.3 Empirical Relations for Radiation Properties of Axial-Mode Helix

Approximate expressions for radiation properties of an axial-mode helix have also been obtained empirically. A summary of the empirical formulas for radiation characteristics is presented below. These formulas are valid when $12^\circ < a < 15^\circ$, $\frac{3}{4} < C_l < \frac{4}{3}$ and $N > 3$.

An approximate directivity expression is given as [1]

$$D \qquad C_l{}^2 NS \;\;, \qquad (2.10)$$

C_l and S_l are, respectively, the circumference and spacing between turns of the helix normalized to the free space wavelength (l). Since the axial-mode helix is nearly lossless, the directivity and the gain expressions are approximately the same.

In 1980, King and Wong [8] reported that Kraus's gain formula (2.10) overestimates the actual gain and proposed a new gain expression using a much larger experimental data base. The new expression is given as

$$G_P = 8.3\left(\frac{pD}{l_P}\right)^{\sqrt{N+2}-1}\left(\frac{NS}{l_P}\right)^{0.8}\left[\frac{\tan 12.5^\circ}{\tan a}\right]^{\sqrt{N}/2}, \qquad (2.11)$$

where l_P is the free-space wavelength at peak gain.

In 1995, Emerson [9] proposed a simple empirical expression for the maximum gain based on numerical modeling of the helix. This expression gives the maximum gain in dB as a function of length normalized to wavelength $(\overline{L}_T = {}^{L_T}\!/_{l})$.

$$G_{\max}\,(\mathrm{dB}) = 10.25 + 1.22\overline{L}_T - 0.0726\overline{L}_T{}^2 \,. \qquad (2.12)$$

Equation (2.12), when compared with the results from experimental and theoretical analyses, gives the gain reasonably accurately.

Half-Power Beamwidth

The empirical formula for the half-power beamwidth is [1]

$$HPBW = \frac{52}{C_l\sqrt{NS_l}} \quad \text{(degrees).} \qquad (2.13)$$

A more accurate formula was later presented by King and Wong using a larger experimental data base [10]. This result is

$$HPBW = \frac{61.5\left(\frac{2N}{N+5}\right)^{0.6}}{\left(\frac{pD}{l}\right)^{\sqrt{N}/4}\left(\frac{NS}{l}\right)^{0.7}}\left(\frac{\tan a}{\tan 12.5^{\circ}}\right)^{\sqrt{N}/4} \text{ (degrees).} \qquad (2.14)$$

Input Impedance

Since the current distribution on the axial-mode helix is assumed to be a travelling wave of constant amplitude (Section 2.3.2), its terminal impedance is nearly purely resistive and is constant with frequency. The empirical formula for the input impedance is

$$R = 140C_l \qquad \text{(ohms).} \qquad (2.15)$$

The input impedance, however, is sensitive to feed geometry. Our numerical modeling of the helix indicated that (2.15) is at best a crude approximation of the input impedance.

Bandwidth

Based on the work of King and Wong [8], an empirical expression for gain bandwidth, as a frequency ratio, has been developed:

$$\frac{f_U}{f_L} \approx 1.07\left(\frac{0.91}{G/G_P}\right)^{4/(3\sqrt{N})}, \qquad (2.16)$$

where f_U and f_L are the upper and lower frequencies, respectively, G_P is the peak gain from equation (2.11), and G is the gain drop with respect to the peak gain.

2.3.4 Optimum Performance of Helix

Many different configurations of the helix have been examined in search of an optimum performance entailing largest gain, widest bandwidth, and/or an axial ratio closest to unity. The helix parameters that result in an optimum performance are summarized in Table 2.1. There are some helices with parameters outside the ranges in Table 2.1 that exhibit unique properties. However, such designs are not regarded as optimum, because not all radiation characteristics meet desired specifications. A summary of the effects of various parameters on the performance of helix is presented below [2].

Table 2.1 Parameter ranges for optimum performance of helix

Parameter	Optimum Range
Circumference	$\frac{3}{4} l < C < \frac{4}{3} l$
Pitch angle	$11^\circ < a < 14^\circ$
Number of turns	$3 < N < 15$
Wire diameter	Negligible effect
Ground plane diameter	At least $\frac{1}{2} l$

Circumference

As shown in Figure 2.6, it is noted that the optimum circumference for achieving the peak gain is around $1.1l$ and is relatively independent of the length of the helix. Other results show that the peak gain smoothly drops as the diameter of the helix decreases (Figure 2.7). Since other parameters of the helix also affect its properties, a circumference of $1.1l$ is viewed as a good estimate for an optimum performance.

Pitch Angle

Keeping the circumference and the length of a helix fixed, the gain increases smoothly when the pitch angle is reduced, as seen in Figure 2.8. However, the reduction

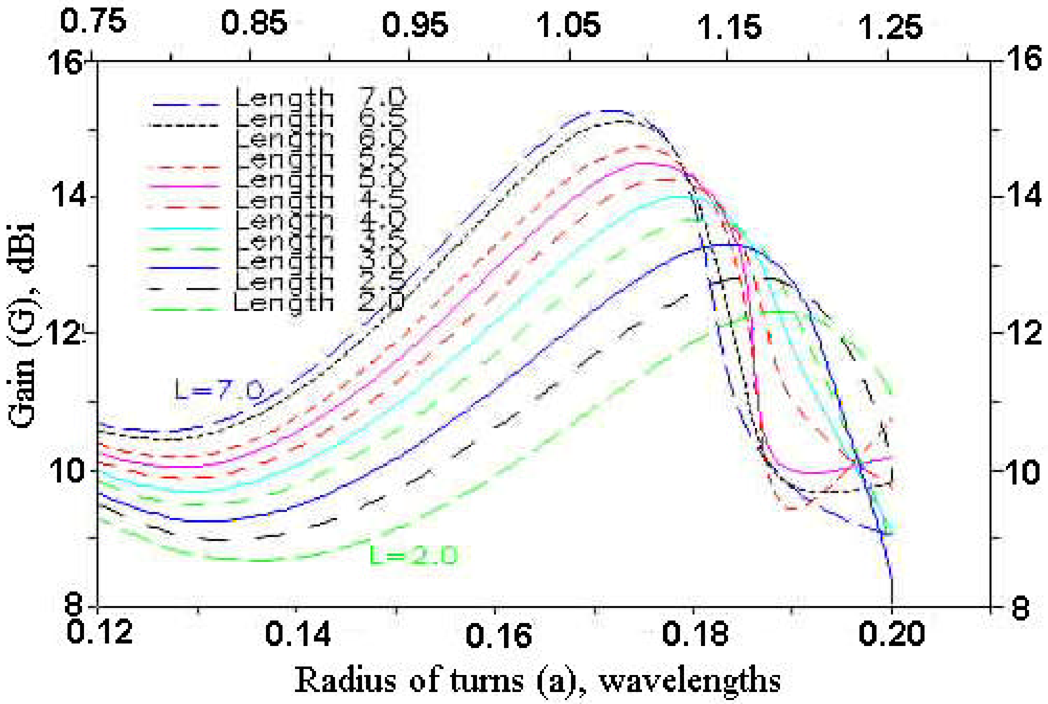


Figure 2.6 Gain of helix for different lengths as function of normalized circumference (C_l) [9]

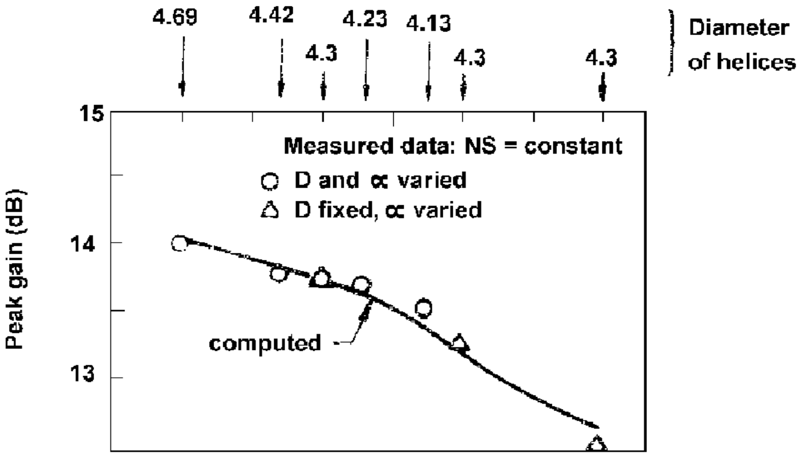


Figure 2.7 Peak gain of various diameter as D and a varied (circles), D fixed and a varied (triangle) [8].

of pitch angle is limited by the bandwidth performance. That is, a narrower bandwidth is obtained for a helix with a smaller pitch angle. For this reason, it has been generally agreed that the optimal pitch angle for the axial-mode helix is about 12.5°.

Number of Turns

Many properties, such as gain, axial ratio, and beamwidth, are affected by the number of turns. Figure 2.9 shows the variation of gain versus the number of turns. It is noted that as the number of turns increases, the gain increases too. The increase in gain is simply explained using the uniformly excited equally-spaced array theory. However, the gain does not increase linearly with the number of turns, and, for very large number of turns, adding more turns has little effect. Also, as shown in Figure 2.10, the beamwidth becomes narrower for larger number of turns. Although adding more turns improves the gain, it makes the helix larger, heavier, and more costly. Practical helices have between 6 and 16 turns. If high gain is required, array of helices may be used.

Conductor Diameter

This parameter does not significantly affect the radiation properties of the helix. For larger conductor diameters, slightly wider bandwidths are obtained. Also, thicker conductors can be used for supporting a longer antenna.

Ground Plain

The effect of ground plain on radiation characteristics of the helix is negligible since the backward traveling waves incident upon it are very weak [7]. Nevertheless, a ground plane with a diameter of one-half wavelength at the lowest frequency is usually recommended.

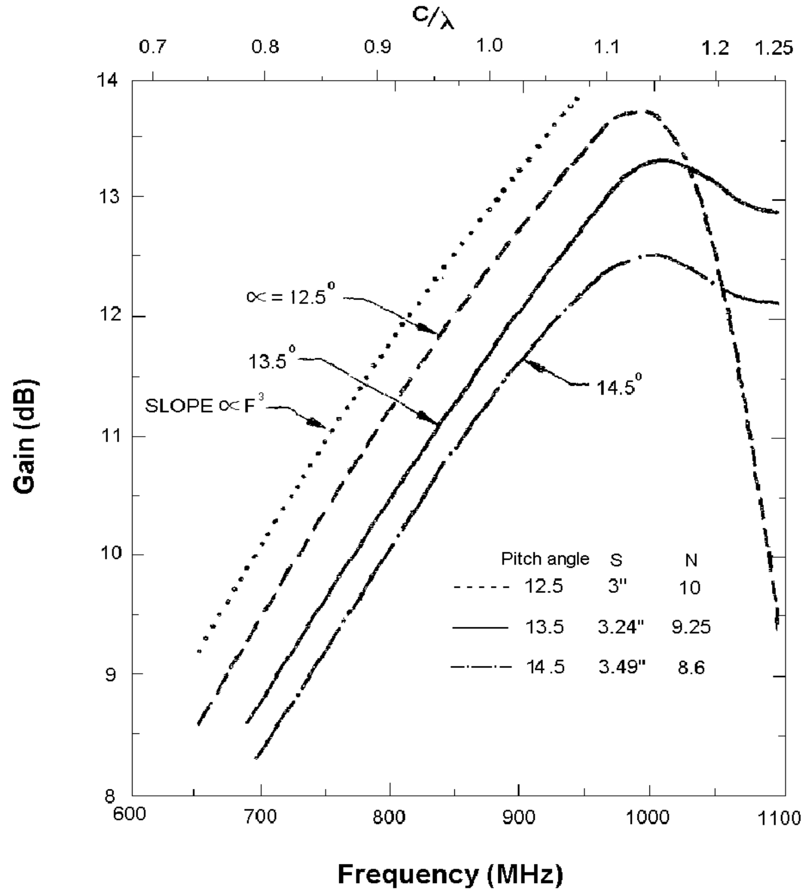


Figure 2.8 **Gain versus frequency of 30.8-inch length and 4.3-inch diameter helix for different pitch angles [8].**

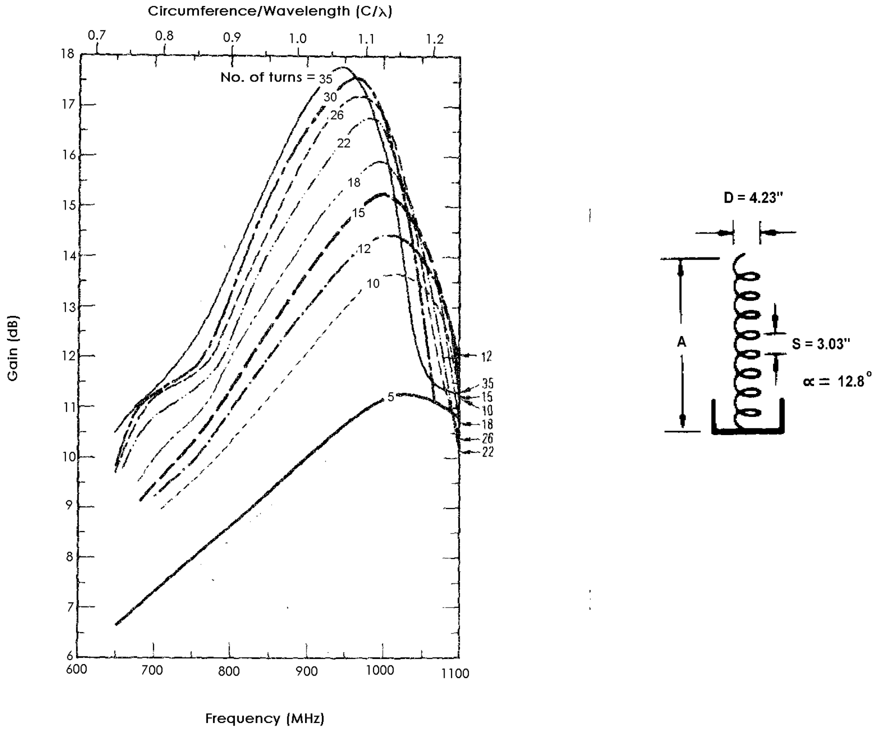


Figure 2.9 Gain versus frequency for 5 to 35-turn helical antennas with 4.23-inch diameter [8]

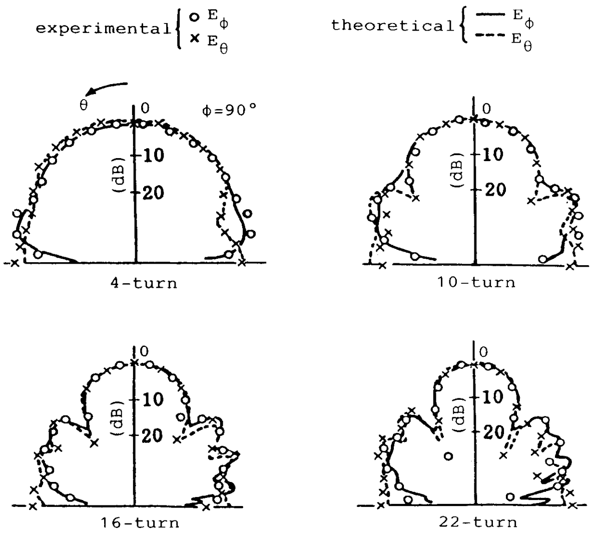


Figure 2.10 Radiation patterns for various helical turns of helices with $a = 12^\circ$ and $C = 10cm.$ at 3 GHz [12].

2.4 Modified Helices

Various modifications of the conventional helical antenna have been proposed for the purpose of improving its radiation characteristics. A summary of these modifications is presented below.

2.4.1 Helical Antenna with Tapered End

Nakano and Yamauchi [11] have proposed a modified helix in which the open end section is tapered as illustrated in Figure 2.11. This structure provides significant improvement in the axial ratio over a wide bandwidth. According to them, the axial ratio improves as the cone angle q_t is increased. For a helix with pitch angle of 12.5° and 6 turns followed by few tapered turns, they obtained an axial ratio of 1:1.3 over a frequency range of 2.6 to 3.5 GHz.

2.4.2 Printed Resonant Quadrifilar Helix

Printed resonant quadrifilar helix is a modified form of the resonant quadrifilar helix antenna first proposed by Kilgus [13]. The structure of this helix consists of 4 microstrips printed spirally around a cylindrical surface. The feed end is connected to the opposite radial strips as seen in Figure 2.12. The advantage of this antenna is a broad beam radiation pattern (half-power beamwidth $>145^\circ$). Additionally, its compact size and light weight are attractive to many applications especially for GPS systems [14].

2.4.3 Stub-Loaded Helix

To reduce the size of a helix operating in the axial mode, a novel geometry referred to as stub-loaded helix has been recently proposed [15]. Each turn contains four stubs as illustrated in Figure 2.13. The stub-loaded helix provides comparable radiation properties to the conventional helix with the same number of turns, while offering an approximately 4:1 reduction in the physical size.

2.4.4 Monopole-Helix Antenna

This antenna consists of a helix and a monopole, as shown in Figure 2.14, [16]. The purpose of this modified antenna is to maintain operation at two different frequencies, applicable to dual-band cellular phone systems operating in two different frequency bands (900 MHz for GSM and 1800 MHz for DCS1800).

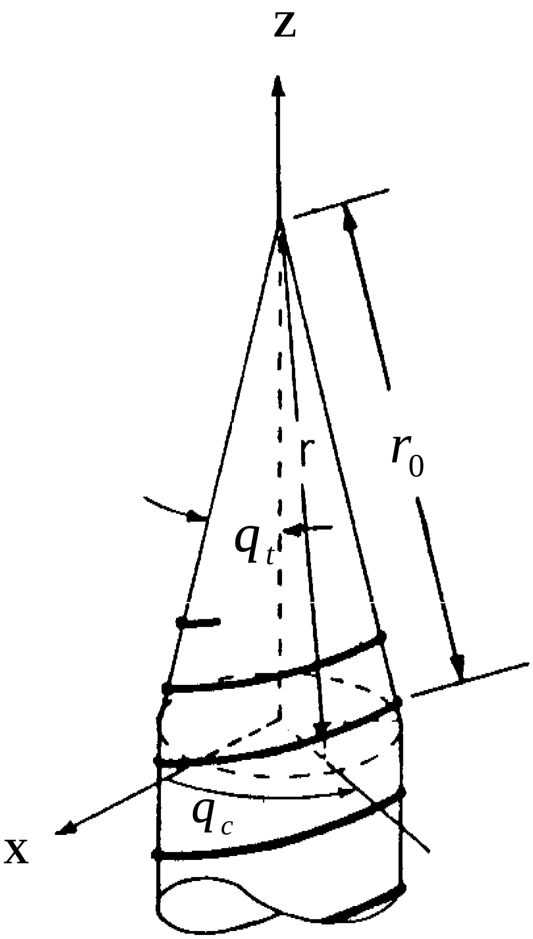


Figure 2.11 Tapered helical antenna configuration.[11].

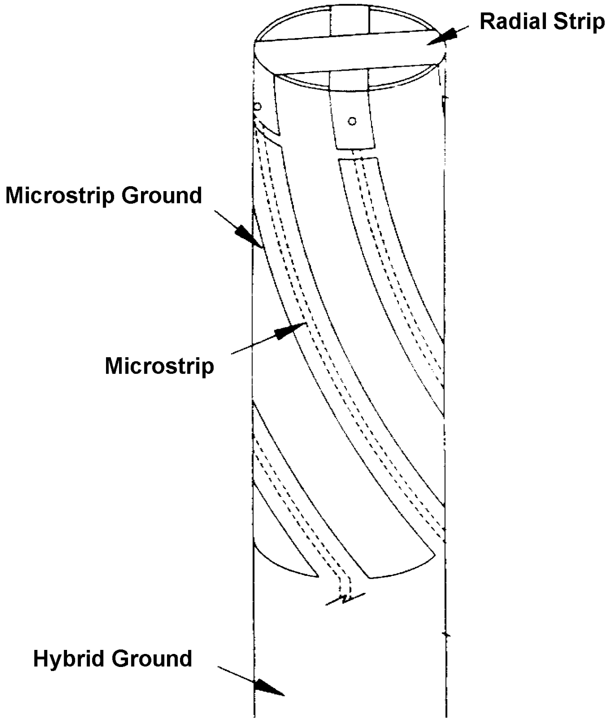


Figure 2.12 $\frac{1}{2}$ turn half-wavelength printed resonant quadrifilar helix [14].

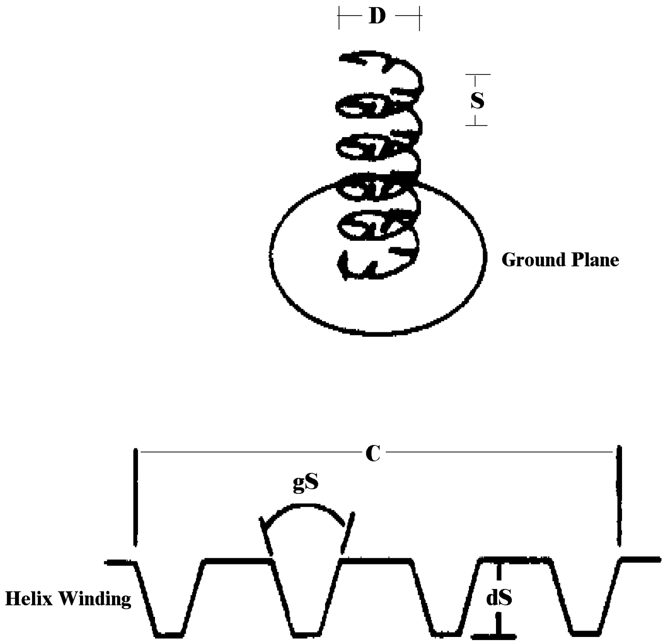


Figure 2.13 Stub-loaded helix configuration [15].

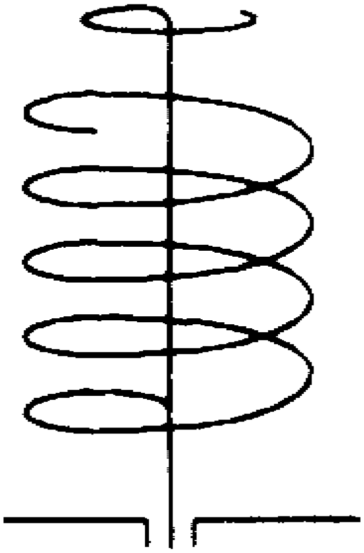


Figure 2.14 Monopole-helix antenna [16].

Chapter 2. The Axial Mode Helix - A Historical Perspective

2.1 Early Helix Development

The first recognition that an interaction between an electron beam and a traveling wave was possible appeared in work by Haeff [1936, 1941] in 1933. In patents filed in that year, Haeff described electron beam deflection tubes, with many of the features of helix traveling wave tubes, that could be used for oscillographs or detectors. In Haeff's devices, an RF signal propagating along a helical winding was used to deflect a nearby electron beam. This occurs through interaction between the fields produced by the RF signal propagating on the helix and the electrons in the electron beam. There is no indication that Haeff was able to produce amplification.

In 1940, Lindenblad [1942] filed a patent that described a helix traveling wave amplifier similar to the helix traveling wave tube. In Lindenblad's amplifier, the signal was applied to the beam using a grid in the electron gun. Later applications applied the RF signal to the helix. He was the first to explain how synchronous interaction between an electron beam and an RF signal on a helix could produce amplification.

Kompfner at Birmingham University, apparently unaware of Lindenblad's work, developed the first traveling wave tube (TWT) in 1943 [Gittins, 1965]. His goal was to produce an amplifier with sensitivity and noise figure comparable to the best state-of-the-art crystal mixer receivers at the time. Because of war secrecy, his work was not published until 1946 [Kompfner, 1946]. The first public presentation of the British wartime work occurred at the Fourth Institute of Radio Engineer's Electron Tube Conference at Yale University, June 27-28, 1946.

At the same conference work by J. R. Pierce and L.M. Field of Bell Labs was also presented. Later work by Pierce and Field would produce refinements to the TWT that would make it a more efficient and practical device. The small signal theory of the TWT was developed by Pierce and published in 1947 in ***Proc. I.R.E.*** [Pierce and Field, 1947]. Pierce's book ***Traveling Wave Tubes*** [Pierce, 1950], published in 1950, is considered to be the standard reference in the field.

2.1.1 Helical Structures in Traveling Wave Tubes

The traveling wave tube (TWT) is a type of microwave device known as a linear beam tube. There are two dominate types of TWT structures: helix and cavity coupled. The helix TWT uses a helical structure to interact with an electron beam directed down the helix axis, and is of primary interest here. Figure 2.1 [Gilmour, 1994] shows the basic structure of a helix TWT.

In order to explain the interactions between the electron beam and the RF signal on the helix, it is useful to consider a single-wire transmission line over a ground plane as shown in Figure 2.2 [Gilmour, 1994]. The instantaneous RF charges and electric field patterns of the single wire transmission line are shown in Figure 2.2. The RF magnetic field reacts only very weakly with the electron beam, so it is not considered. Assuming an infinitely long lossless line and a generator on the left of Figure 2.2, then the charges and fields shown travel from left to right with a constant amplitude. The velocity of propagation is equal to the speed of light and independent of frequency, i.e. the transmission line is non-dispersive.

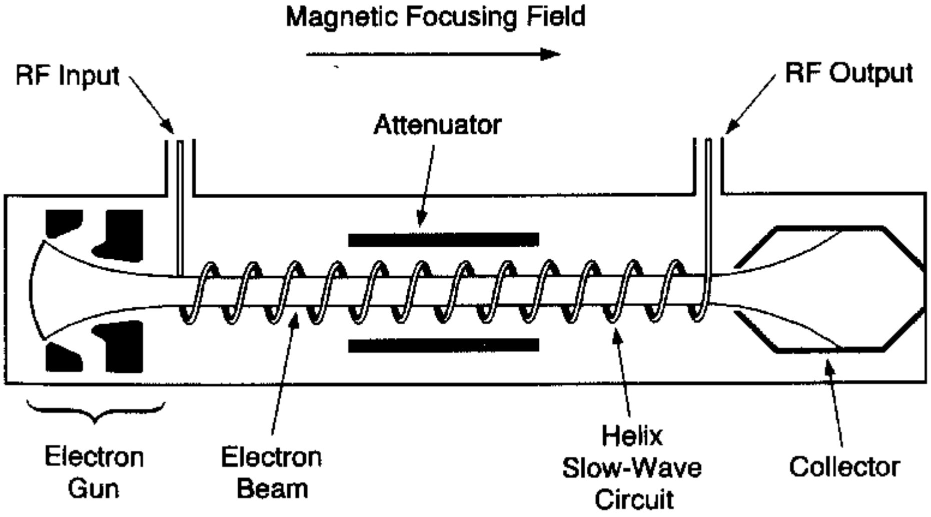


Figure 2.1 Basic helix traveling wave tube amplifier. Reprinted with permission from Principles of Traveling Wave Tubes, by A. S. Gilmour, Jr., Artech House Publishers, Norwood, MA, USA. www.artechhouse.com [Gilmour, 1994]

Now, if the single-wire transmission line is formed into a helix with a circumference much smaller than a wavelength, as shown in Figure 2.3 [Gilmour, 1994], then the RF signal will travel along the helix at a velocity nearly the speed of light. The axial velocity of the RF signal in the helix axis direction will be reduced by an amount equal to the helix pitch angle, creating a slow wave structure. The primary difference between the fields of the single-wire and helix transmission lines is that the helical line has a large axial component of the electric field along the helix axis.

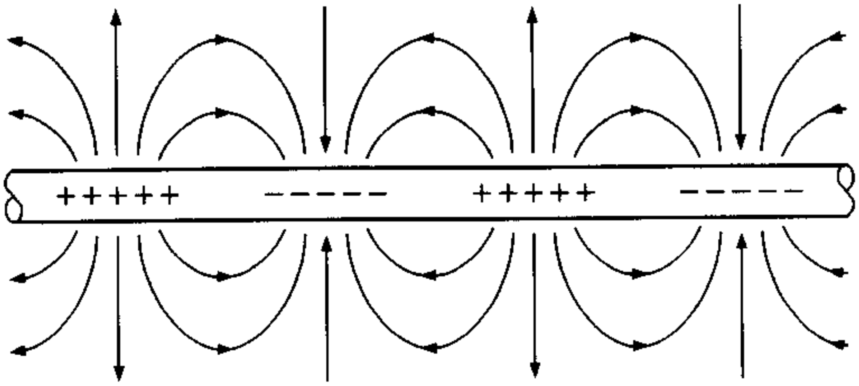


Figure 2.2 RF charge (+ and -) and electric field patterns (solid lines) for a single-wire transmission line above a ground plane. Reprinted with permission from Principles of Traveling Wave Tubes, by A. S. Gilmour, Jr., Artech House Publishers, Norwood, MA, USA. www.artechhouse.com [Gilmour, 1994]

When an electron beam is injected along the helix axis, electrons will be accelerated or decelerated depending upon which part of the axial electric field they interact with. For example, electrons will be accelerated toward the regions marked A in Figure 2.3 [Gilmour, 1994] and decelerated toward the regions marked B. The result is that an electron bunch will form around the regions marked A. This effect is called velocity modulation. The bunching of electrons in the electron beam will in turn interact with the electrons in the RF current flowing along the helix causing them to be repelled. If the velocity of the electron beam and the axial velocity of the helix current are the same, a synchronous interaction between helix current and electron beam current occurs which results in an exponential growth of the circuit voltage [Gilmour, 1994]. The result is amplification of the driving signal as it progresses along the helix. This occurs due to the transfer of energy from the electrons to the wave associated with the RF signal.

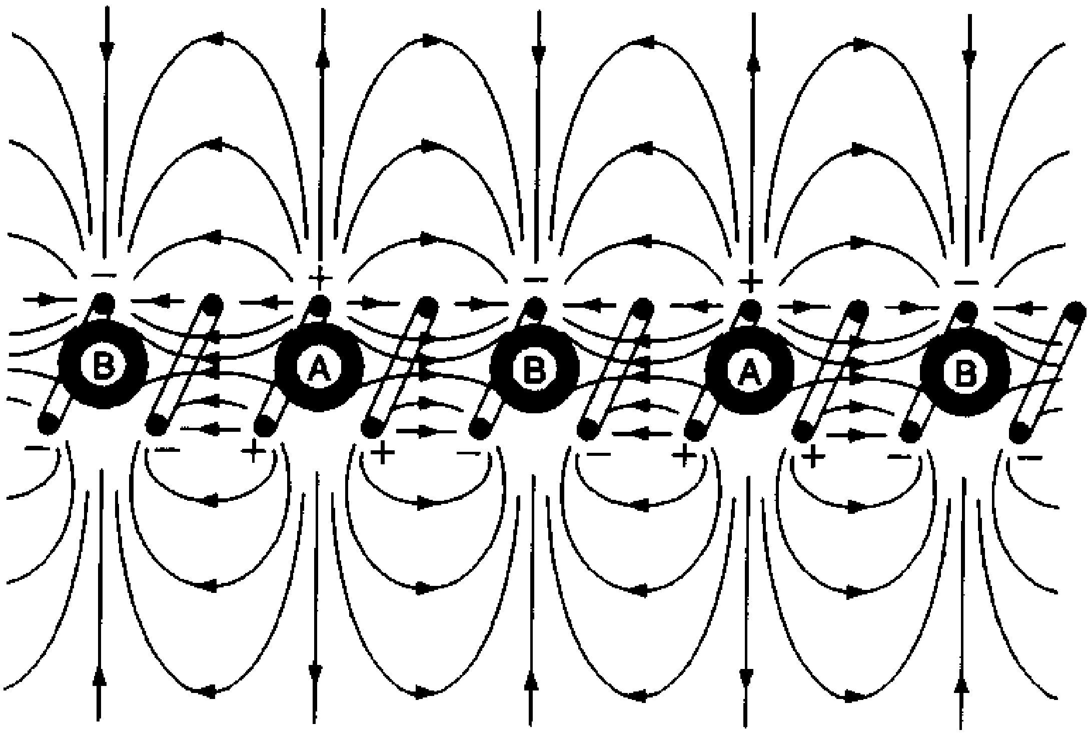


Figure 2.3. RF charge and electric field patterns for a helix. Reprinted with permission from Principles of Traveling Wave Tubes, by A. S. Gilmour, Jr., Artech House Publishers, Norwood, MA, USA. www.artechhouse.com [Gilmour, 1994]

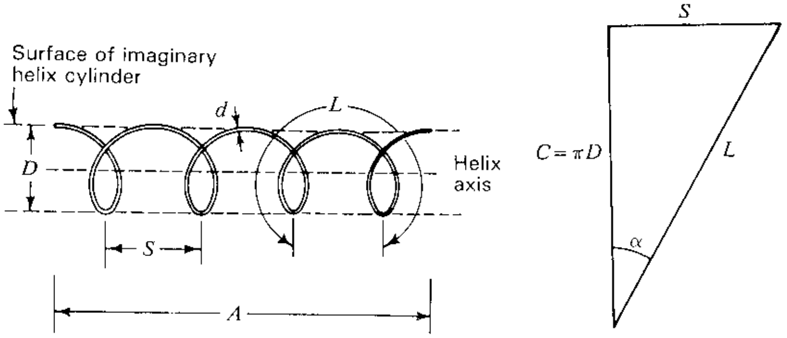
2.1.2 Development of the Helical Antenna

Shortly after the Electron Tube Conference at Yale University, Dr. Paul Raines visited Ohio State University in November 1946 and gave a lecture on traveling wave tube amplifiers [Kraus, 1976]. Dr. John Kraus, an OSU professor, was in attendance. After the lecture Kraus spoke with Raines and asked him if he thought the helix might be made to operate as an antenna. Raines responded that he had tried and the helix would not work as an antenna.

Kraus surmised that the helix may not have worked because it was too small in diameter. Kraus, using a 2.5 GHz oscillator as a source, wound a 7-turn helix with a 4 cm diameter, making for a circumference of 12.5 cm, approximately one wavelength at the source frequency of 2.5 GHz. Using a crystal detector attached to a small fan dipole as a receiving antenna, Kraus was able to verify that his helix produced endfire radiation along the axis of the helix and that the radiation was circularly polarized. This began an exhaustive program of research by Kraus into the properties of the endfire, or axial mode, helix.

The first published work on the helical antenna was in 1947 [Kraus, 1947]. Since then hundreds of papers have been published on the helical antenna detailing the theory of operation, modifications to improve its performance, and variations on the basic helix design. One source exploring many of the helix variations and their performance characteristics is the excellent book by Nakano [1987] which also includes an extensive list of references to other sources.

Figure 2.4 shows the basic geometry of a helical antenna as defined by Kraus [1988]. The defining parameters of the conventional helix are the helix diameter, D, the helix circumference, C, the turn-to-turn spacing, S, the pitch angle of the turns, α, and the axial length, A. The diameter, and hence circumference, primarily determine the frequency of operation of the helix. The circumference of the helix is approximately equal to the wavelength of the center frequency of operation of the helix. The pitch angle and axial length of the helix affect the gain. There is a range of pitch angles which correspond to optimum gain for a given axial length. The longer the axial length, the greater the forward gain of the helix. However, like most traveling wave structures, a point of diminishing returns on gain improvement with length is reached fairly quickly.



(a) (b)

Figure 2.4 Basic helix geometry defining diameter (D), turn-to-turn spacing (S), axial length (A), circumference (C), turn length (L), and pitch angle (α). In (b) the relationships between S, C, D, L and α are shown for a single turn that has been stretched out flat. [Kraus, 1988]

2.2 Variations on the Helix Antenna

The axial mode helix in the form discovered by Kraus was the starting point for many different kinds of traveling wave antennas. This section reviews some of the more prominent ones.

2.2.1 Quadrifilar Helix

The quadrifilar helix antenna, sometimes referred to as a volute antenna, consists of four helical windings oriented 90° with respect to one another, as shown in Figure 2.5. The basic quadrifilar helix, developed by Kilgus [1968, 1969, 1970, 1975], is a unique and versatile antenna. By proper selection of the helix parameters, a wide range of radiation pattern characteristics can be are obtained and good circular polarization can be achieved over a large percentage of the pattern. Although it is commonly used as a spacecraft antenna, the quadrifilar helix can also make an excellent ground station antenna.

The most common configuration of the quadrifilar helix is the half-turn, resonant quadrifilar, as shown in Figure 2.5. This configuration is instructive in understanding the operation of the quadrifilar. Quadrifilar helices are often confused with the more common conventional helix antenna, most likely due to the helical shape of the windings of both antennas. But the there are critical physical and electrical differences between the two.

The conventional helix consists of one, and occasionally more, helical windings, usually with multiple turns. The windings are usually fed in-phase. Their circumference is approximately one-wavelength at the operating frequency and typically have pitch angles (measured from turn-to-turn) of 10° to 15°. The axial mode of operation is supporting a traveling wave along the helical winding to produce an endfire radiation pattern. The conventional helix is usually fed against a groundplane, typically at least one-quarter wavelength in diameter.

The quadrifilar helix consists of four helical windings, spaced 90° with respect to the adjacent winding, enclosing a common volume. The opposing windings are usually joined to form a pair of orthogonal bifilar helices. The windings are fed in phase quadrature between adjacent windings. Besides the phasing, the most significant differences between the conventional helix and the quadrifilar helix are the helix diameter and pitch angle.

Depending on the desired pattern characteristics, the diameter and pitch angle of the quadrifilar windings can vary over a relatively large range. But generally, the diameter is much smaller and its pitch angle is large in comparison to the conventional helix.

For short quadrifilar helices (one turn or less) the radiation pattern is essentially endfire, but is very broad. Half-power beamwidths of greater than 90° are typical, and good circular polarization over a very large percentage of the pattern is expected. As the number of turns on a quadrifilar is increased, the pattern becomes more broadside with the beam peak moving toward the horizon and the beamwidth becomes narrower. As this occurs, the peak gain increases, but the portion of the pattern over which good circular polarization is produced narrows, but is usually located around the beam peak. Omnidirectional azimuthal coverage is maintained in all of these configurations. Figure 2.6 illustrates the change in elevation pattern as the number of turns of the quadrifilar increases.

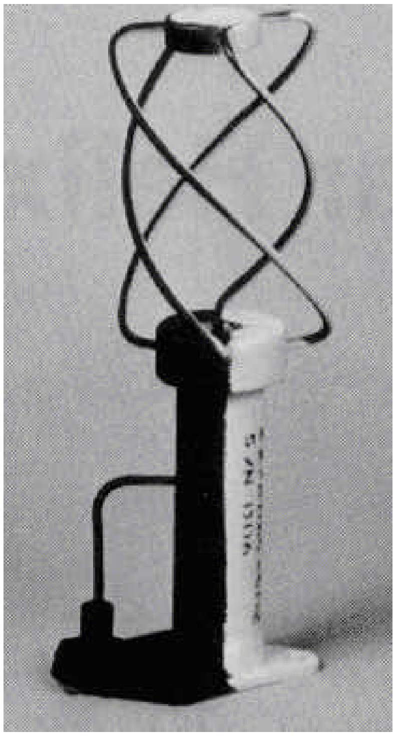


Figure 2.5. Typical quadrifilar helix or volute antenna [Maxwell, 1990]

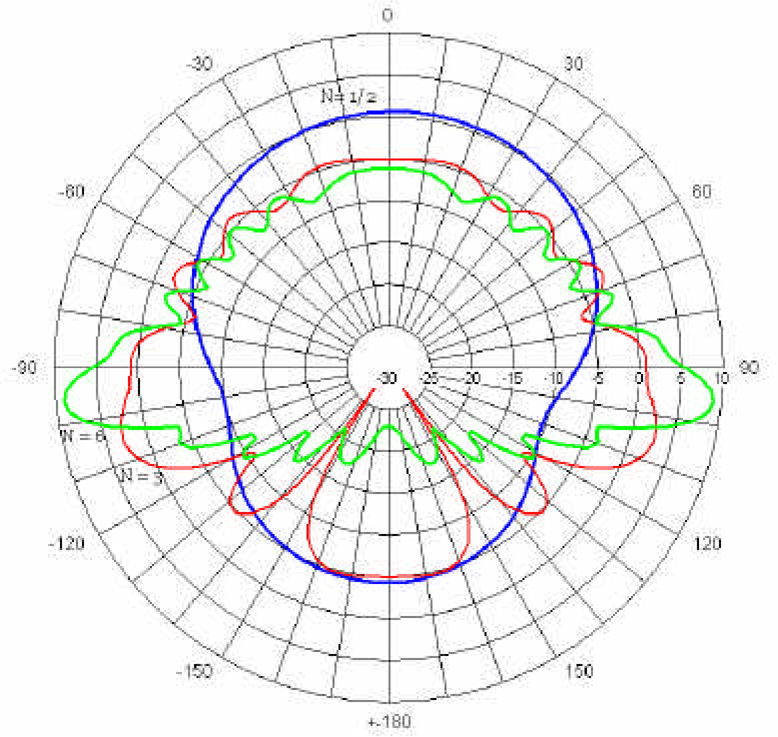


Figure 2.6 NEC simulated radiation patterns of quadrifilar helices with N = 1/2, N = 3, and N = 6 turns showing how the main lobe of the pattern moves from on-axis to broadside of the quadrifilar as the number of turns is increased. The helix axis is aligned with $\theta = 0°$, the same orientation as in Figure 2.5.

2.2.2 Spherical Helix

The traditional helix antenna consists of a helical winding on a cylindrical surface. Safaai-Jazi and Cardoso [1996] proposed a helical antenna that is formed on a spherical surface. The spherical helix consists of a helical winding with constant spacing between turns formed on a spherical surface. Their study indicated that the spherical helix has some interesting properties that are distinctly different from conventional helices.

Over the range where the circumference of the sphere is $0.75\lambda < C < 2.0\lambda$, the spherical helix radiates in an endfire mode producing circular polarization. The gain of the antenna does not vary significantly with the number of turns used, unlike a cylindrical helix. While at first this may be a surprising result, it should not be since, regardless of the number of turns used, the volume of the antenna is being held constant. The on-axis gain of the sperical helix operating over this frequency range was measured as 9 dB with 3 dB and 10 dB beamwidths of 60° and 110°, respectively. The axial mode circumference range of $0.75\lambda < C < 2.0\lambda$ corresponds to an bandwidth of approximately 91%.

Over certain narrow ranges within the circumference range cited above, the spherical helix produces circular polarization over a broad beamwidth. Specifically, for 4-turn and 10-turn helices with C = 1.15λ and 1.25λ, respectively, the patterns remained circularly polarized over a beamwidth of approximately 120°. It would appear that the axial ratio performance would define the operational bandwidth of the spherical helix. However, there were no specifics given on its expected axial ratio bandwidth.

When the circumference of the sphere is in the range $2.0\lambda < C < 2.8\lambda$, a null of approximately 10 dB develops in the pattern along the axis of the helix. The maximum gain of the bifurcated main lobe is approximately 7 dB and the beams have 3 dB beamwidths of approximately 33°. The polarization produced in this axial-null mode is generally elliptical, but may be circular over narrow frequency ranges. Again, no specifics were given in [Safaai-Jazi and Cardoso, 1996].

The unique characteristics of the spherical helix suggest that it might be a good candidate for use as a mobile antenna for low earth orbiting (LEO) satellite systems. Circular polarization over a broad beamwidth is a highly desired but sorely lacking characteristic in current antenna designs used in LEO systems.

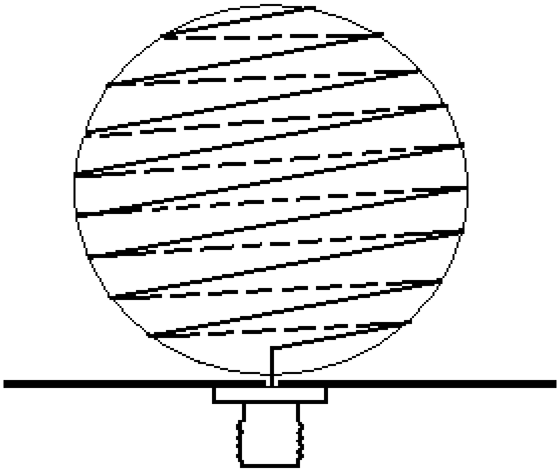


Figure 2.7. Spherical helix antenna with groundplane.

2.2.3 Zig-zag Antenna

Imagine a traditional helical structure that is flattened it into a planar structure. The result would be a wire antenna formed in a zig-zag pattern, see Figure 2.8. Cumming [1955] first investigated this antenna structure followed by Sengupta [1958].

The zig-zag antenna is a form of traveling wave antenna that when properly designed produces a strong axial beam with very low sidelobes. The gain of the antenna is a function of the Vee length (2L in Figure 2.8), the V angle (2α in Figure 2.8), and the number of Vee's. Results reported by Sengupta indicate that zig-zag antenna has a usable bandwidth of approximately 10%, based on the pattern behavior. When the operating frequency is approximately 10% above the center frequency of the antenna, its pattern forms a split beam with a null on axis, similar to an axial mode helix operated in its second mode.

The front-to-back ratio and sidelobe characteristics of the zig-zag antenna are comparable to a Yagi-Uda design of similar length, but with a wider bandwidth due to the traveling wave nature of the antenna. The polarization of the zig-zag antenna is linear, as would be expected of a planar, endfire array. Sengupta's [1958] investigations were prompted by the desire to use the zig-zag antenna in a VHF radiotelescope array being built at the University of Toronto. It appears that this antenna has little other application despite its features of high gain, simple construction, and relatively wide bandwidth.

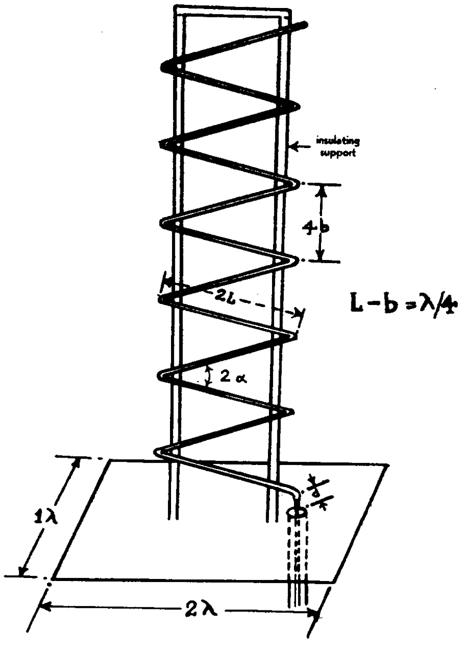


Figure 2.8. Diagram showing the zig-zag antenna with feeding arrangement [Sengupta, 1958]

2.2.4 Helix-Fed Dielectric Rod Antenna

Various attempts have been made to use dielectric loading, usually within the core of an axial mode helix, to reduce the helix size, as shown in Figure 2.9. These attempts have usually met with limited success most likely due to incompatibilities between helical and dielectric antennas. Namely, due to the high permittivity of the dielectric, the electric fields are concentrated within the material, reducing the amount of radiation that occurs.

However, dielectric rod antennas have been shown to produce high directivity when properly designed and fed. Combining a dielectric rod antenna with a small helix as a suitable feed combines the desirable characteristics of both [Hui, et al., 1996]. The helix launches a circularly polarized wave into the dielectric rod which guides and focuses the wave down the rod. Both helix and dielectric rod antenna are slow wave structures, hence there seems to be some basic compatibility in their operation. The result of the combination is a circularly polarized antenna with improved directivity. However, there are no claims of any improvements in gain or in size reduction.

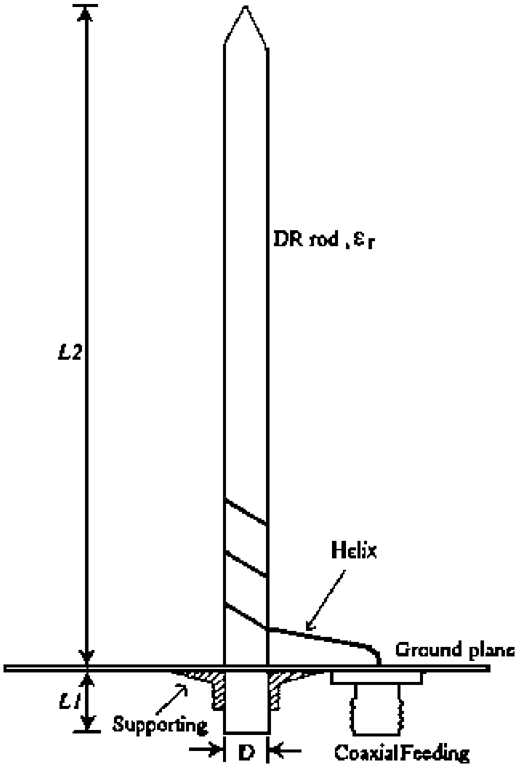


Figure 2.9. Cylindrical dielectric rod antenna fed by a short helix [Hui, et al, 1996]

2.2.5 Internally Matched Helix Antenna

Wong and Chen [1991] presented a design for reduced size variant of the axial mode helix antenna called the internally matched helical beam antenna. The internally matched helical beam antenna shown in Figure 2.10 consists of a helical winding on top of a cylindrical groundplane separated by a thin dielectric layer.

The helical winding over the groundplane forms a transmission line structure. By adjusting the width of the helical winding, the impedance of the transmission line is adjusted to provide a match to 50 Ω allowing direct feed of the antenna. The distributed inductance and capacitance of the transmission line structure result in a velocity of propagation along the winding less than the speed of light. This slow wave structure results in a size reduction for the helix. Wong and Chen claim an 86% reduction in the diameter of the helix using this technique.

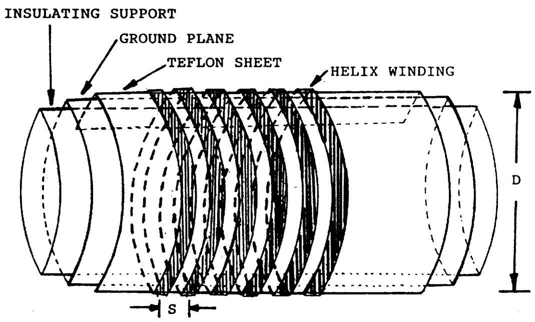


Figure 2.10 The schematic diagram of an internally matched helical antenna; S = spacing between the turns and D = diameter of the helix or the insulating cylinder [Wong and Chen, 1991].

The principle of using a microstrip transmission line structure in order to create an antenna seems counterintuitive. Well designed transmission line structures do not radiate well, as is desired for transmission. Significant radiation usually occurs when either the spacing between the transmission line and groundplane becomes electrically large or the width of the transmission line becomes a significant fraction of a wavelength, as in the case of microstrip patch antennas. The internally matched helix can excite a surface traveling wave along its structure, but one would expect that such a wave would remain tightly coupled to the microstrip structure resulting in low radiation efficiency. Wong and Chen [1991] show radiation patterns for their test antenna which demonstrate directivity, but make no claims about gain. They report an estimated radiation efficiency of more than 70%, but it is not entirely clear how they arrived at this number. An investigation by Spall, et al. [1994] failed to reproduce the results of Wong and Chen. The proposed directivity was observed but gain was very low.

2.2.6 Slow-Wave Helix

Based in part on the idea of the zig-zap spiral antenna, a helix geometry using zig-zag windings was developed by Spall, et al. [1994]. The Slow-Wave helix, shown in Figure 2.11, consists of a helical winding with a regular, evenly spaced zig-zag pattern along the axial direction of the antenna. This antenna design was originated by Spall and Stutzman at Virginia Tech. The zig-zag winding creates a slow wave transmission line structure for the helix. The phase velocity within the winding is lowered resulting in a shorter wavelength within the helix. Initial investigations indicated a diameter size reduction of almost two over a conventional axial mode helix [Spall, et al., 1994].

A later study consisting of an extensive parameter study using numerical modeling revealed some significant limitations to the Slow-Wave helix geometry [Barts and Stutzman, 1996]. The diameter reduction of the Slow-Wave helix was almost a factor of two; additionally, the length was less than one-half that of a conventional helix for the same number of turns. There was, however, a significant reduction in gain compared to a conventional helix. In modeling comparisons between five-turn Slow-Wave and conventional helices, it was observed that the Slow-Wave helix exhibited between 1 and 3 dB less gain than the conventional helix over their range of operation. Additionally, the Slow-Wave helix model exhibited a relatively narrow frequency range over which the axial ratio was low. The 3-dB axial ratio bandwidth was approximately 10%.

The Slow-Wave helix was developed as part of an effort to reduce the size of helical antennas for use in UHF military satellite systems. While it met the size reduction requirements and its gain reduction was tolerable, its limited bandwidth removed it from consideration. The limitations of the Slow-Wave helix were the motivation that led to the development of the Stub Loaded Helix. The Slow-Wave helix still holds promise for applications where moderate bandwidth is acceptable. The previous studies of the Slow-Wave helix were by no means exhaustive. The basic concept seems valid but the geometry requires optimizing in order to achieve maximum performance.

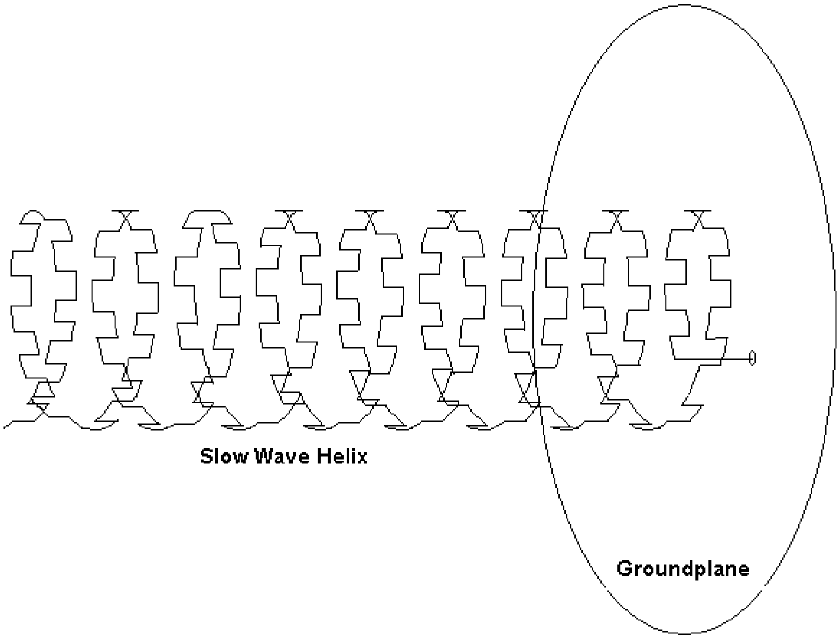


Figure 2.11 Geometry of the Slow-Wave helix

2.2.7 The Helicone Antenna

The helicone antenan is a full size helical antenna placed inside a large cylindrical horn; an example is illustrated in Figure 2.12. The helicone was developed by Carver [1967] as a variation of the helix with extremely low back and sidelobes and increased directivity over a stand alone helix. The helicone's bandwidth and axial ratio properties are superior to those of a conical horn excited from a circularly polarized waveguide operating in the TE_{11} mode. The impedance behavior of the helicone is very similar to that of the helix alone, being predominately resistive with a small amount of reactance across the operating bandwidth. The extremely low back and sidelobe levels of the helicone make it an attractive choice for radio astronomy where noise from the warm earth can increase the antenna noise temperature. Due to the low sidelobe levels, mutual coupling is minimal. Thus, using helicones in an array would be relatively straight forward.

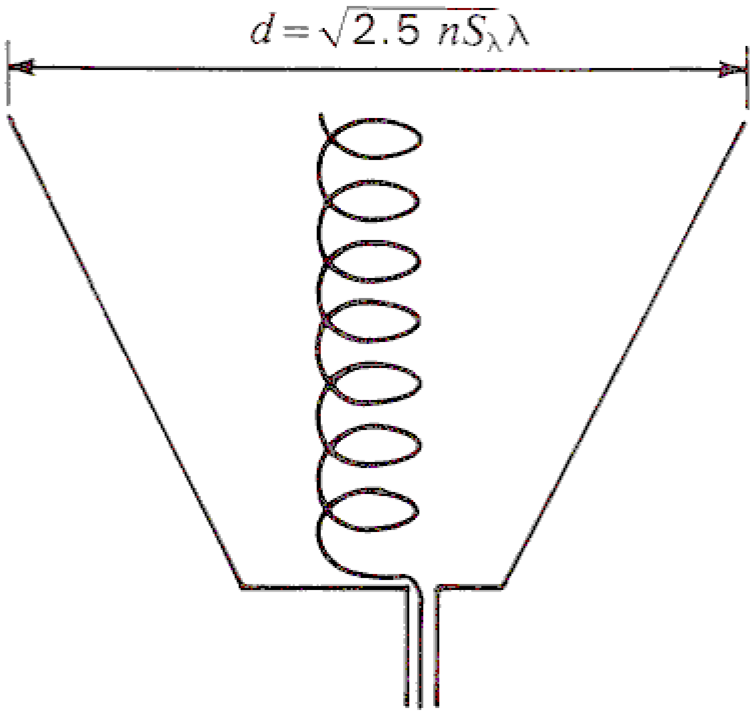


Figure 2.12. The Helicone antenna after Carver [1967]. Image from Kraus [1988].

XBee®/XBee-PRO® ZB RF Modules

ZigBee RF Modules by Digi International

Firmware Versions:

- 20xx - Coordinator - AT/Transparent Operation
- 21xx - Coordinator - API Operation
- 22xx - Router - AT/Transparent Operation
- 23xx - Router - API Operation
- 28xx - End Device - AT/Transparent Operation
- 29xx - End Device - API Operation

Digi International Inc.
11001 Bren Road East
Minnetonka, MN 55343
877 912-3444 or 952 912-3444
http://www.digi.com

90000976_C
3/3/2009

Technical Support: Phone: (801) 765-9885 Worldwide
(866) 765-9885 toll-free U.S.A. & Canada

Live Chat: www.digi.com

Contents

Overview 6

Key Features 7

Worldwide Acceptance 7

Specifications 7

Mechanical Drawings 9

SIF Header Interface 10

Mounting Considerations 10

Pin Signals 11

EM250 Pin Mappings 12

Design Notes 12

Power Supply Design 12

Recommended Pin Connections 12

Board Layout 13

Electrical Characteristics 13

RF Module Operation 14

Serial Communications 14

UART Data Flow 14

Serial Buffers 14

Serial Flow Control 15

Serial Interface Protocols 16

Modes of Operation 18

Idle Mode 18

Transmit Mode 18

Receive Mode 19

Command Mode 19

Sleep Mode 20

XBee ZigBee Networks 21

Introduction to ZigBee 21

ZigBee Stack Layers 21

Networking Concepts 21

Device Types 21

PAN ID 22

Operating Channel 23

ZigBee Application Layers: In Depth 23

Application Support Sublayer (APS) 23

Coordinator Operation 24

Forming a Network 24

Channel Selection 25

PAN ID Selection 25

Security Policy 25

Persistent Data 25

XBee ZB Coordinator Startup 25

Permit Joining 26

Resetting the Coordinator 26

Leaving a Network 27

Replacing a Coordinator (Security Disabled Only) 27

Example: Starting a Coordinator 28

Example: Replacing a Coordinator (security disabled) 28

Router Operation 28

Joining a Network 28

Discovering ZigBee Networks 28

Joining a Network 29

Authentication 29

Persistent Data 29

XBee ZB Router Joining 29

Permit Joining 31

Joining Always Enabled 31

Joining Temporarily Enabled 31

Resetting the Router 31

Leaving a Network 31

Example: Joining a Network 32

End Device Operation 32

Joining a Network 32

Discovering ZigBee Networks 32

Joining a Network 33

Parent Child Relationship 33

End Device Capacity 33

Authentication 33

Persistent Data 33

Orphan Scans 33

XBee: ZB End Device Joining 34

Parent Connectivity 35

Resetting the End Device 35

Leaving a Network 35

Example: Joining a Network 35

Channel Scanning 36

Managing Multiple ZigBee Networks 36

PAN ID Filtering 36

Preconfigured Security Keys 37

Permit Joining 37

Application Messaging 37

Data Transmission, Addressing, and Routing 38

Addressing 38

64-bit Device Addresses 38

16-bit Device Addresses 38

Contents

Application Layer Addressing 38
Data Transmission 38
Broadcast Transmissions 38
Unicast Transmissions 39
Address Table 40
Address Discovery 40
Data Transmission Examples 40
RF Packet Routing 42
Link Status Transmission 42
Encrypted Transmissions 48
Improving Routing Efficiency with the API 48
Address Table 48
Maximum RF Payload Size 49
Management Transmissions 49
ZigBee Device Objects (ZDO) 49
Sending a ZDO Command 50
Receiving ZDO Commands and Responses 50
Transmission Timeouts 52
Transmitting to Remote Router or Coordinator 52
Transmitting to Child End Device 52
Transmitting to Remote End Device 53
Transmission Examples 53
Security 55
Security Modes 55
ZigBee Security Model 55
Network Layer Security 55
Frame Counter 56
Message Integrity Code 56
Network Layer Encryption and Decryption 56
Network Key Updates 57
APS Layer Security 57
Message integrity Code 57
APS Link Keys 57
APS Layer Encryption and Decryption 58
Network and APS Layer Encryption 58
Trust Center 58
Forming and Joining a Secure Network 58
Implementing Security on the XBee 59
Enabling Security 59
Setting the Network Security Key 59
Setting the APS Trust Center Link Key 59
Using a Trust Center 59
XBee Security Examples 60
Example 1: Forming a network with security (pre-configured link keys) 60
Example 2: Forming a network with security (obtaining keys during joining) 60
Managing End Devices 62
End Device Operation 62
Parent Operation 62
End Device Poll Timeouts 63
Packet Buffer Usage 63
Non-Parent Device Operation 64
XBee End Device Configuration 64
Pin Sleep 64
Cyclic Sleep 66
Transmitting RF Data 69
Receiving RF Data 70
IO Sampling 70
Waking End Devices with the Commissioning Pushbutton 70
Parent Verification 70
Rejoining 70
XBee Router/Coordinator Configuration 71
RF Packet Buffering Timeout 71
Child Poll Timeout 71
Transmission Timeout 71
Putting it all Together 72
Short Sleep Periods 72
Extended Sleep Periods 72
Sleep Examples 72
Network Commissioning and Diagnostics 74
Device Configuration 74
Device Placement 74
Link Testing 74
RSSI Indicators 75
Device Discovery 75
Commissioning Pushbutton and Associate LED 75
Commissioning Pushbutton 76
Associate LED 77
XBee Analog and Digital IO Lines 79
IO Configuration 79
IO Sampling 79
Queried Sampling 81
Periodic IO Sampling 81
Change Detection Sampling 81
RSSI PWM 81

Contents

IO Examples 82

API Operation 83

API Frame Specifications 83

API Examples 85

API UART Exchanges 86

AT Commands 86

Transmitting and Receiving RF Data 86

Remote AT Commands 86

Source Routing 87

Supporting the API 87

API Frames 87

AT Command 87

AT Command - Queue Parameter Value 88

ZigBee Transmit Request 88

Explicit Addressing ZigBee Command Frame 90

Remote AT Command Request 92

Create Source Route 93

AT Command Response 94

Modem Status 94

ZigBee Transmit Status 95

ZigBee Receive Packet 96

ZigBee Explicit Rx Indicator 97

ZigBee IO Data Sample Rx Indicator 98

XBee Sensor Read Indicator 99

Node Identification Indicator 101

Remote Command Response 102

Over-the-Air Firmware Update Status 103

Route Record Indicator 104

Sending ZigBee Device Objects (ZDO) Commands with the API 105

Sending ZigBee Cluster Library (ZCL) Commands with the API 107

Sending Public Profile Commands with the API 109

XBee Command Reference Tables 113

OEM Support 121

X-CTU Configuration Tool 121

Customizing XBee ZB Firmware 121

Design Considerations for Digi Drop-In Networking 121

XBee Bootloader 121

Programming XBee Modules 122

Serial Firmware Updates 122

Invoke XBee Bootloader 122

Send Firmware Image 122

SIF Firmware Updates 123

Writing Custom Firmware 123

Regulatory Compliance 123

Enabling GPIO 1 and 2 123

Detecting XBee vs XBee-PRO 124

Ensuring Optimal Output Power 124

Improving Low Power Current Consumption 125

XBee (non-PRO) Initialization: 125

When sleeping (end devices): 125

When waking from sleep (end devices): 125

Appendix A: Definitions 127

Appendix A: Agency Certifications 129

Appendix B:Migrating from ZNet 2.5 to XBee ZB 134

Appendix C:XBee ZB Firmware Matrix 135

Overview of Features 135

Appendix D:Additional Information 137

1. Overview

The XBee/XBee-PRO ZB RF Modules are designed to operate within the ZigBee protocol and support the unique needs of low-cost, low-power wireless sensor networks. The modules require minimal power and provide reliable delivery of data between remote devices.

The modules operate within the ISM 2.4 GHz frequency band and are compatible with the following:

- XBee RS-232 Adapter
- XBee RS-232 PH (Power Harvester) Adapter
- XBee RS-485 Adapter
- XBee Analog I/O Adapter
- XBee Digital I/O Adapter
- XBee Sensor Adapter
- XBee USB Adapter
- XStick
- ConnectPort X Gateways
- XBee Wall Router.

The XBee/XBee-PRO ZB firmware release can be installed on XBee ZNet or ZB modules. The XBee ZB firmware is based on the EmberZNet 3.x ZigBee PRO Feature Set mesh networking stack, while the XBee ZNet 2.5 firmware is based on Ember's proprietary "designed for ZigBee" mesh stack (EmberZNet 2.5.x). ZB and ZNet 2.5 firmware are similar in nature, but not over-the-air compatible. Devices running ZNet 2.5 firmware cannot talk to devices running the ZB firmware.

Key Features

High Performance, Low Cost

XBee

- Indoor/Urban: up to 133’ (40 m)
- Outdoor line-of-sight: up to 400’ (120 m)
- Transmit Power: 2 mW (3 dBm)
- Receiver Sensitivity: -96 dBm

XBee-PRO

- Indoor/Urban: up to 300’ (90 m), 200' (60 m) for International variant
- Outdoor line-of-sight: up to 1 mile (1600 m), 2500' (750 m) for International variant
- Transmit Power: 50mW (17dBm), 10mW (10dBm) for International variant
- Receiver Sensitivity: -102 dBm

Advanced Networking & Security

Retries and Acknowledgements

DSSS (Direct Sequence Spread Spectrum)

Each direct sequence channel has over 65,000 unique network addresses available

Point-to-point, point-to-multipoint and peer-to-peer topologies supported

Self-routing, self-healing and fault-tolerant mesh networking

Low Power

XBee

- TX Peak Current: 40 mA (@3.3 V)
- RX Current: 40 mA (@3.3 V)
- Power-down Current: < 1 uA

XBee-PRO

- TX Peak Current: 295mA (170mA for international variant)
- RX Current: 45 mA (@3.3 V)
- Power-down Current: < 10 uA

Easy-to-Use

No configuration necessary for out-of box RF communications

AT and API Command Modes for configuring module parameters

Small form factor

Extensive command set

Free X-CTU Software
(Testing and configuration software)

Free & Unlimited Technical Support

Worldwide Acceptance

FCC Approval (USA) Refer to Appendix A for FCC Requirements.
Systems that contain XBee®/XBee-PRO® ZB RF Modules inherit Digi Certifications.

ISM (Industrial, Scientific & Medical) **2.4 GHz frequency band**

Manufactured under **ISO 9001:2000** registered standards

XBee®/XBee-PRO® ZB RF Modules are optimized for use in **US**, **Canada**, **Australia, Israel and Europe** (contact MaxStream for complete list of agency approvals).

Specifications

Table 1-01. Specifications of the XBee®/XBee-PRO® ZB OEM RF Module

Specification	XBee	XBee-PRO
Performance		
Indoor/Urban Range	up to 133 ft. (40 m)	Up to 300 ft. (90 m), up to 200 ft (60 m) international variant
Outdoor RF line-of-sight Range	up to 400 ft. (120 m)	Up to 1 mile (1600 m), up to 2500 ft (750 m) international variant
Transmit Power Output	2mW (+3dBm), boost mode enabled 1.25mW (+1dBm), boost mode disabled	50mW (+17 dBm) 10mW (+10 dBm) for International variant
RF Data Rate	250,000 bps	250,000 bps
Serial Interface Data Rate (software selectable)	1200 - 230400 bps (non-standard baud rates also supported)	1200 - 230400 bps (non-standard baud rates also supported)
Receiver Sensitivity	-96 dBm, boost mode enabled -95 dBm, boost mode disabled	-102 dBm

Table 1-01. Specifications of the XBee®/XBee-PRO® ZB OEM RF Module

Specification	XBee	XBee-PRO
Power Requirements		
Supply Voltage	2.1 - 3.6 V	3.0 - 3.4 V
Operating Current (Transmit, max output power)	40mA (@ 3.3 V, boost mode enabled) 35mA (@ 3.3 V, boost mode disabled)	295mA (@3.3 V), 170mA (@3.3 V) international variant
Operating Current (Receive))	40mA (@ 3.3 V, boost mode enabled) 38mA (@ 3.3 V, boost mode disabled)	45 mA (@3.3 V)
Idle Current (Receiver off)	15mA	15mA
Power-down Current	< 1 uA @ 25^{o}C	< 10 uA @ 25^{o}C
General		
Operating Frequency Band	ISM 2.4 GHz	ISM 2.4 GHz
Dimensions	0.960” x 1.087” (2.438cm x 2.761cm)	0.960 x 1.297 (2.438cm x 3.294cm)
Operating Temperature	-40 to 85º C (industrial)	-40 to 85º C (industrial)
Antenna Options	Integrated Whip, Chip, RPSMA, or U.FL Connector*	Integrated Whip, Chip, RPSMA, or U.FL Connector*
Networking & Security		
Supported Network Topologies	Point-to-point, Point-to-multipoint, Peer-to-peer, and Mesh	Point-to-point, Point-to-multipoint, Peer-to-peer, and Mesh
Number of Channels	16 Direct Sequence Channels	14 Direct Sequence Channels
Addressing Options	PAN ID and Addresses, Cluster IDs and Endpoints (optional)	PAN ID and Addresses, Cluster IDs and Endpoints (optional)
Agency Approvals		
United States (FCC Part 15.247)	FCC ID: OUR-XBEE2	FCC ID: MCQ-XBEEPRO2
Industry Canada (IC)	IC: 4214A-XBEE2	IC: 1846A-XBEEPRO2
Europe (CE)	ETSI	ETSI
Australia	C-Tick	C-Tick
Japan	R201WW07215214	R201WW08215142
RoHS	Compliant	Compliant

Mechanical Drawings

Figure 1-01. Mechanical drawings of the XBee®/XBee-PRO® ZB OEM RF Modules (antenna options not shown)

.

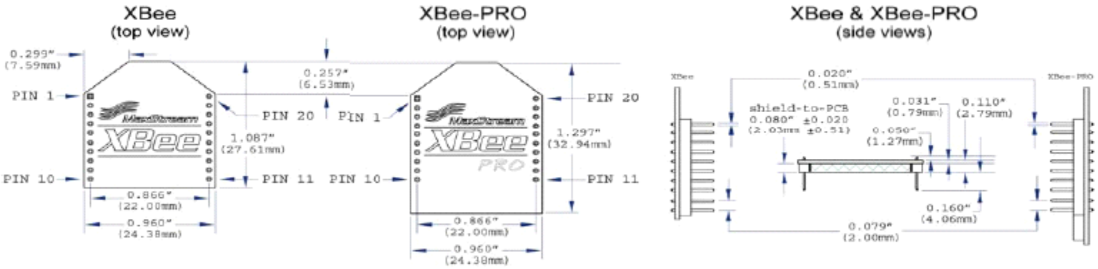
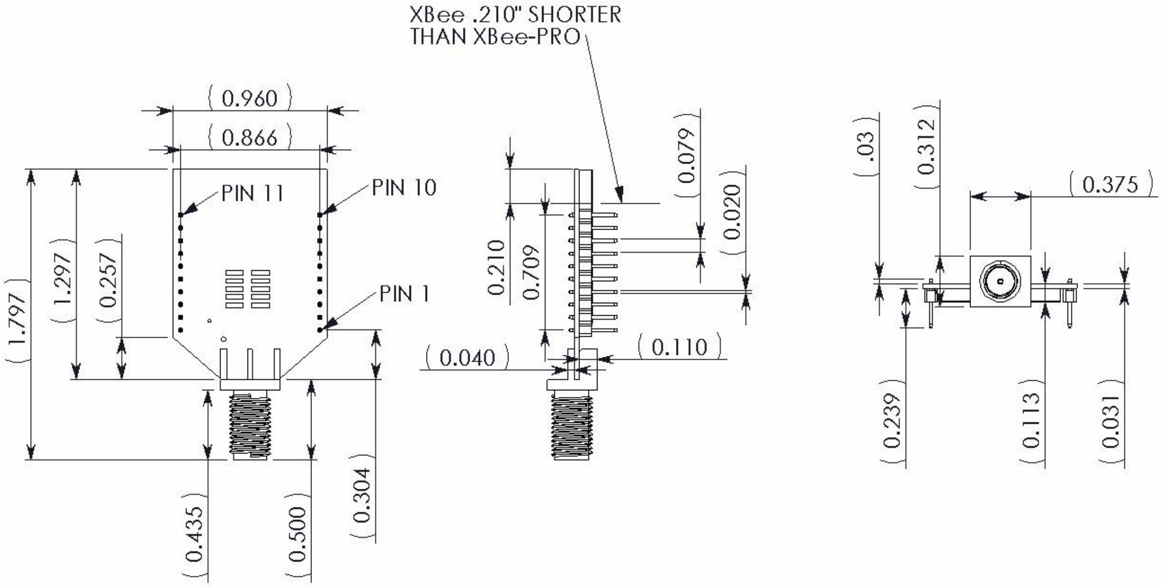
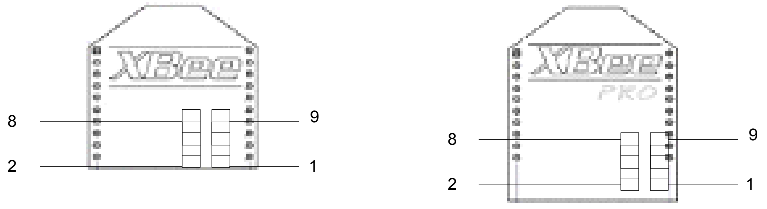


Figure 1-02. Mechanical Drawings for the RPSMA Variant



SIF Header Interface

The XBee/XBee-PRO ZB modules include a SIF programming header that can be used with Ember's programming tools to upload custom firmware images onto the XBee module. The SIF header orientation and pinout are shown below.



This figure shows the orientation of the insight port header .

Pin Number	Pin Name
1	VBRD
2	SIF-MISO
3	Ground
4	SIF-MOSI
5	Ground
6	SIF-CLOCK
7	SIF-LOAD
8	RESET
9	PTI-EN
10	PTI-DATA

A male header can be populated on the XBee that mates with Ember's 2x5 ribbon cable. The male header and ribbon cables are available from Samtec:

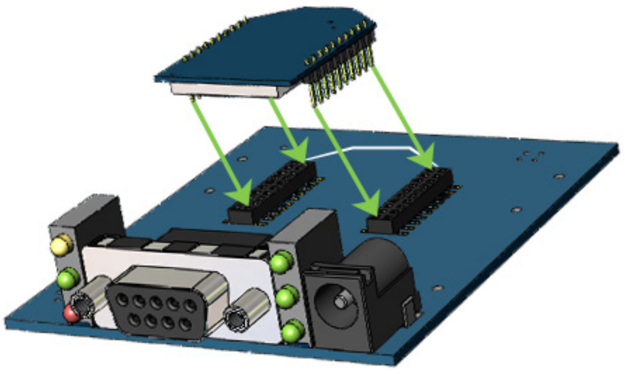
2x5 Male Header - FTSH-105-01-F-DV-K

2x5 Ribbon Cable - FFSD-05-D-12.00-01-N

Mounting Considerations

The XBee modules were designed to mount into a receptacle (socket) and therefore does not require any soldering when mounting it to a board. The XBee-PRO Development Kits contain RS-232 and USB interface boards which use two 20-pin receptacles to receive modules.

Figure 1-03. XBee-PRO Module Mounting to an RS-232 Interface Board.



The receptacles used on Digi development boards are manufactured by Century Interconnect. Several other manufacturers provide comparable mounting solutions; however, Digi currently uses the following receptacles:

- Through-hole single-row receptacles - Samtec P/N: MMS-110-01-L-SV (or equivalent)
- Surface-mount double-row receptacles - Century Interconnect P/N: CPRMSL20-D-0-1 (or equivalent)

- Surface-mount single-row receptacles - Samtec P/N: SMM-110-02-SM-S

Digi also recommends printing an outline of the module on the board to indicate the orientation the module should be mounted.

Pin Signals

Figure 1-04. XBee®/XBee-PRO® ZB RF Module Pin Number

(top sides shown - shields on bottom)

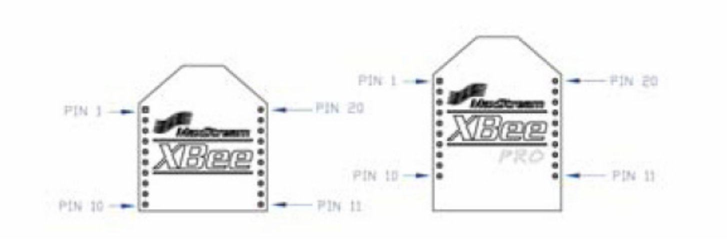


Table 1-02. Pin Assignments for the XBee-PRO Modules
(Low-asserted signals are distinguished with a horizontal line above signal name.)

Pin #	Name	Direction	Description
1	VCC	-	Power supply
2	DOUT	Output	UART Data Out
3	DIN / $\overline{\textbf{CONFIG}}$	Input	UART Data In
4	DIO12	Either	Digital I/O 12
5	$\overline{\textbf{RESET}}$	Input	Module Reset (reset pulse must be at least 200 ns)
6	PWM0 / RSSI / DIO10	Either	PWM Output 0 / RX Signal Strength Indicator / Digital IO
7	DIO11	Either	Digital I/O 11
8	[reserved]	-	Do not connect
9	$\overline{\text{DTR}}$ / SLEEP_RQ/ DIO8	Either	Pin Sleep Control Line or Digital IO 8
10	GND	-	Ground
11	DIO4	Either	Digital I/O 4
12	$\overline{\textbf{CTS}}$ / DIO7	Either	Clear-to-Send Flow Control or Digital I/O 7. CTS, if enabled, is an output.
13	ON / $\overline{\textbf{SLEEP}}$	Output	Module Status Indicator or Digital I/O 9
14	VREF	Input	Not used on this module. For compatibility with other XBee modules, we recommend connecting this pin to a voltage reference if Analog sampling is desired. Otherwise, connect to GND.
15	Associate / DIO5	Either	Associated Indicator, Digital I/O 5
16	$\overline{\textbf{RTS}}$ / DIO6	Either	Request-to-Send Flow Control, Digital I/O 6. RTS, if enabled, is an input.
17	AD3 / DIO3	Either	Analog Input 3 or Digital I/O 3
18	AD2 / DIO2	Either	Analog Input 2 or Digital I/O 2
19	AD1 / DIO1	Either	Analog Input 1 or Digital I/O 1
20	AD0 / DIO0 / Commissioning Button	Either	Analog Input 0, Digital IO 0, or Commissioning Button

- Signal Direction is specified with respect to the module
- See Design Notes section below for details on pin connections.
- PWM functionality not currently supported.

EM250 Pin Mappings

The following table shows how the EM250 pins are used on the XBee.

EM250 Pin Number	XBee Pin Number	Other Usage
13 (Reset)	5	Connected to pin 8 on 2x5 SIF header.
19 (GPIO 11)	16	
20 (GPIO 12)	12	
21 (GPIO 0)	15	
22 (GPIO 1)		**XBee** Tied to ground (module identification) **XBee-PRO** Low-asserting shutdown line for output power compensation circuitry.
24 (GPIO 2)		**XBee** Not connected. Configured as output low. **XBee-PRO** Powers the output power compensation circuitry.
25 (GPIO 3)	13	
26 (GPIO 4 / ADC 0)	20	Connected to pin 9 on 2x5 SIF header.
27 (GPIO 5 / ADC 1)	19	Connected to pin 10 on 2x5 SIF header.
29 (GPIO 6 /ADC 2)	18	
30 (GPIO 7 / ADC 3	17	
31 (GPIO 8)	4	
32 (GPIO 9)	2	
33 (GPIO 10)	3	
34 (SIF_CLK)		Connected to pin 6 on 2x5 SIF header.
35 (SIF_MISO)		Connected to pin 2 on 2x5 SIF header.
36 (SIF_MOSI)		Connected to pin 4 on 2x5 SIF header.
37 (SIF_LOAD)		Connected to pin 7 on 2x5 SIF header.
40 (GPIO 16)	7	
41 (GPIO 15)	6	
42 (GPIO 14)	9	
43 (GPIO 13)	11	

Design Notes

The XBee modules do not specifically require any external circuitry or specific connections for proper operation. However, there are some general design guidelines that are recommended for help in troubleshooting and building a robust design.

Power Supply Design

Poor power supply can lead to poor radio performance especially if the supply voltage is not kept within tolerance or is excessively noisy. To help reduce noise a 1.uF and 8.2pF capacitor are recommended to be placed as near to pin1 on the PCB as possible. If using a switching regulator for your power supply, switching frequencies above 500kHz are preferred. Power supply ripple should be limited to a maximum 250mV peak to peak.

Recommended Pin Connections

The only required pin connections are VCC, GND, DOUT and DIN. To support serial firmware updates, VCC, GND, DOUT, DIN, RTS, and DTR should be connected.

All unused pins should be left disconnected. All inputs on the radio can be pulled high with 30k internal pull-up resistors using the PR software command. No specific treatment is needed for unused outputs.

Other pins may be connected to external circuitry for convenience of operation including the Associate LED pin (pin 15) and the Commissioning pin (pin 20). The Associate LED pin will flash differently depending on the state of the module to the network, and a pushbutton attached to pin 20 can enable various join functions without having to send UART commands. Please see the commissioning pushbutton and associate LED section in chapter 7 for more details. The source and sink capabilities are limited to 4mA for all pins on the module.

The VRef pin (pin 14) is not used on this module. For compatibility with other XBee modules, we recommend connecting this pin to a voltage reference if analog sampling is desired. Otherwise, connect to GND.

Board Layout

XBee modules do not have any specific sensitivity to nearby processors, crystals or other PCB components. Other than mechanical considerations, no special PCB placement is required for integrating XBee radios. In general, Power and GND traces should be thicker than signal traces and be able to comfortably support the maximum currents.

The radios are also designed to be self sufficient and work with the integrated and external antennas without the need for additional ground planes on the host PCB. Large ground planes on a host PCB should not adversely affect maximum range, but they may affect radiation patterns of onboard XBee antennas.

Electrical Characteristics

Table 1-03. DC Characteristics of the XBee-PRO (VCC = 3.0 - 3.4 VDC).

Symbol	Parameter	Condition	Min	Typical	Max	Units
VIL	Input Low Voltage	All Digital Inputs	-	-	0.2 * VCC	V
V_{IH}	Input High Voltage	All Digital Inputs	0.8 * VCC	-	-	V
V_{OL}	Output Low Voltage	I_{OL} = 2 mA, VCC >= 2.7 V	-	-	0.18*VCC	V
V_{OH}	Output High Voltage	I_{OH} = -2 mA, VCC >= 2.7 V	0.82*VCC	-	-	V
II_{IN}	Input Leakage Current	V_{IN} = VCC or GND, all inputs, per pin	-	-	0.5uA	uA

2. RF Module Operation

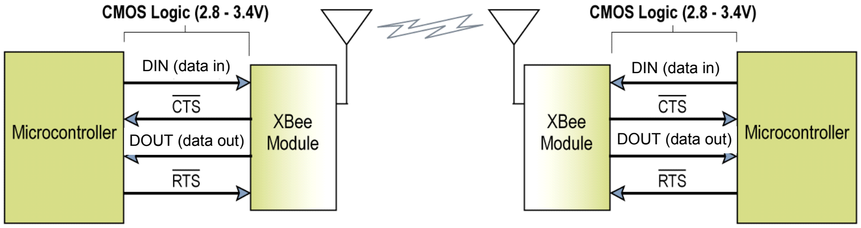
Serial Communications

The XBee OEM RF Modules interface to a host device through a logic-level asynchronous serial port. Through its serial port, the module can communicate with any logic and voltage compatible UART; or through a level translator to any serial device (For example: Through a Digi proprietary RS-232 or USB interface board).

UART Data Flow

Devices that have a UART interface can connect directly to the pins of the RF module as shown in the figure below.

Figure 2-01. System Data Flow Diagram in a UART-interfaced environment
(Low-asserted signals distinguished with horizontal line over signal name.)

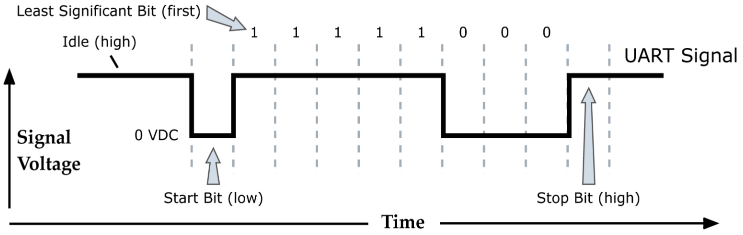


Serial Data

Data enters the module UART through the DIN (pin 3) as an asynchronous serial signal. The signal should idle high when no data is being transmitted.

Each data byte consists of a start bit (low), 8 data bits (least significant bit first) and a stop bit (high). The following figure illustrates the serial bit pattern of data passing through the module.

Figure 2-02. UART data packet 0x1F (decimal number "31") as transmitted through the RF module
Example Data Format is 8-N-1 (bits - parity - # of stop bits)

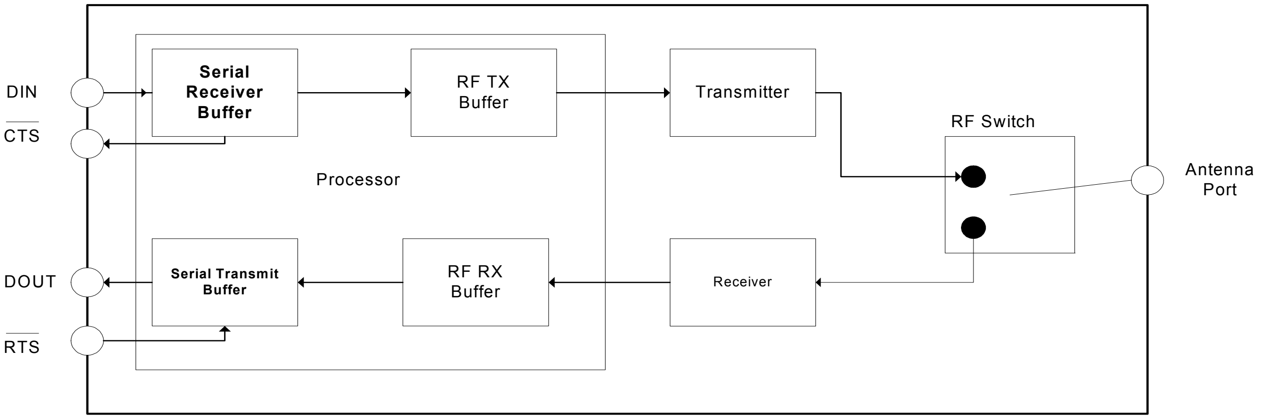


The module UART performs tasks, such as timing and parity checking, that are needed for data communications. Serial communications depend on the two UARTs to be configured with compatible settings (baud rate, parity, start bits, stop bits, data bits).

Serial Buffers

The XBee modules maintain small buffers to collect received serial and RF data, which is illustrated in the figure below. The serial receive buffer collects incoming serial characters and holds them until they can be processed. The serial transmit buffer collects data that is received via the RF link that will be transmitted out the UART.

Figure 2-03. Internal Data Flow Diagram



Serial Receive Buffer

When serial data enters the RF module through the DIN Pin (pin 3), the data is stored in the serial receive buffer until it can be processed. Under certain conditions, the module may not be able to process data in the serial receive buffer immediately. If large amounts of serial data are sent to the module, CTS flow control may be required to avoid overflowing the serial receive buffer.

Cases in which the serial receive buffer may become full and possibly overflow:

1.If the module is receiving a continuous stream of RF data, the data in the serial receive buffer will not be transmitted until the module is no longer receiving RF data.

2.If the module is transmitting an RF data packet, the module may need to discover the destination address or establish a route to the destination. After transmitting the data, the module may need to retransmit the data if an acknowledgment is not received, or if the transmission is a broadcast. These issues could delay the processing of data in the serial receive buffer.

Serial Transmit Buffer

When RF data is received, the data is moved into the serial transmit buffer and sent out the UART. If the serial transmit buffer becomes full enough such that all data in a received RF packet won’t fit in the serial transmit buffer, the entire RF data packet is dropped.

Cases in which the serial transmit buffer may become full resulting in dropped RF packets

1. If the RF data rate is set higher than the interface data rate of the module, the module could receive data faster than it can send the data to the host.

2. If the host does not allow the module to transmit data out from the serial transmit buffer because of being held off by hardware flow control.

Serial Flow Control

The RTS and CTS module pins can be used to provide RTS and/or CTS flow control. CTS flow control provides an indication to the host to stop sending serial data to the module. RTS flow control allows the host to signal the module to not send data in the serial transmit buffer out the uart. RTS and CTS flow control are enabled using the D6 and D7 commands.

CTS Flow Control

If CTS flow control is enabled (D7 command), when the serial receive buffer is 17 bytes away from being full, the module de-asserts CTS (sets it high) to signal to the host device to stop sending serial data. CTS is re-asserted after the serial receive buffer has 34 bytes of space.

$\overline{\text{RTS}}$ Flow Control

If RTS flow control is enabled (D6 command), data in the serial transmit buffer will not be sent out the DOUT pin as long as $\overline{\text{RTS}}$ is de-asserted (set high). The host device should not de-assert $\overline{\text{RTS}}$ for long periods of time to avoid filling the serial transmit buffer. If an RF data packet is received, and the serial transmit buffer does not have enough space for all of the data bytes, the entire RF data packet will be discarded.

Serial Interface Protocols

The XBee modules support both transparent and API (Application Programming Interface) serial interfaces.

Transparent Operation

When operating in transparent mode, the modules act as a serial line replacement. All UART data received through the DIN pin is queued up for RF transmission. When RF data is received, the data is sent out through the DOUT pin. The module configuration parameters are configured using the AT command mode interface.

Data is buffered in the serial receive buffer until one of the following causes the data to be packetized and transmitted:

- No serial characters are received for the amount of time determined by the RO (Packetization Timeout) parameter. If RO = 0, packetization begins when a character is received.
- The Command Mode Sequence (GT + CC + GT) is received. Any character buffered in the serial receive buffer before the sequence is transmitted.
- The maximum number of characters that will fit in an RF packet is received

RF modules that contain the following firmware versions will support Transparent Mode:
20xx (AT coordinator), 22xx (AT router), and 28xx (AT end device).

API Operation

API operation is an alternative to transparent operation. The frame-based API extends the level to which a host application can interact with the networking capabilities of the module. When in API mode, all data entering and leaving the module is contained in frames that define operations or events within the module.

Transmit Data Frames (received through the DIN pin (pin 3)) include:

- RF Transmit Data Frame
- Command Frame (equivalent to AT commands)

Receive Data Frames (sent out the DOUT pin (pin 2)) include:

- RF-received data frame
- Command response
- Event notifications such as reset, associate, disassociate, etc.

The API provides alternative means of configuring modules and routing data at the host application layer. A host application can send data frames to the module that contain address and payload information instead of using command mode to modify addresses. The module will send data frames to the application containing status packets; as well as source, and payload information from received data packets.

The API operation option facilitates many operations such as the examples cited below:

->Transmitting data to multiple destinations without entering Command Mode

->Receive success/failure status of each transmitted RF packet

->Identify the source address of each received packet

RF modules that contain the following firmware versions will support API operation: 21xx (API coordinator), 23xx (API router), and 29xx (API end device).

A Comparison of Transparent and API Operation

The following table compares the advantages of transparent and API modes of operation:

Transparent Operation Features	
Simple Interface	All received serial data is transmitted unless the module is in command mode.
Easy to support	It is easier for an application to support transparent operation and command mode
API Operation Features	
Easy to manage data transmissions to multiple destinations	Transmitting RF data to multiple remotes only requires changing the address in the API frame. This process is much faster than in transparent operation where the application must enter AT command mode, change the address, exit command mode, and then transmit data. Each API transmission can return a transmit status frame indicating the success or reason for failure.
Received data frames indicate the sender's address	All received RF data API frames indicate the source address.
Advanced ZigBee addressing support	API transmit and receive frames can expose ZigBee addressing fields including source and destination endpoints, cluster ID and profile ID. This makes it easy to support ZDO commands and public profile traffic.
Advanced networking diagnostics	API frames can provide indication of IO samples from remote devices, and node identification messages.
Remote Configuration	Set / read configuration commands can be sent to remote devices to configure them as needed using the API.

As a general rule of thumb, API firmware is recommended when a device:

- sends RF data to multiple destinations
- sends remote configuration commands to manage devices in the network
- receives IO samples from remote devices
- receives RF data packets from multiple devices, and the application needs to know which device sent which packet
- must support multiple ZigBee endpoints, cluster IDs, and/or profile IDs
- uses the ZigBee Device Profile services.

If the above conditions do not apply (i.e. a sensor node, router, or a simple application), then AT firmware might be suitable. It is acceptable to use a mixture of devices running API and AT firmware in a network.

Modes of Operation

Idle Mode

When not receiving or transmitting data, the RF module is in Idle Mode. The module shifts into the other modes of operation under the following conditions:

- Transmit Mode (Serial data in the serial receive buffer is ready to be packetized)
- Receive Mode (Valid RF data is received through the antenna)
- Sleep Mode (End Devices only)
- Command Mode (Command Mode Sequence is issued)

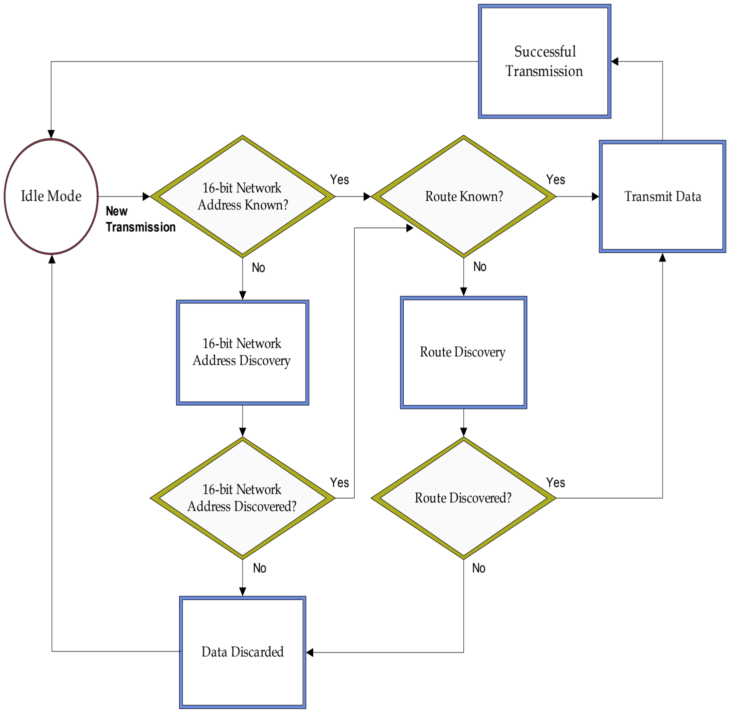
Transmit Mode

When serial data is received and is ready for packetization, the RF module will exit Idle Mode and attempt to transmit the data. The destination address determines which node(s) will receive the data.

Prior to transmitting the data, the module ensures that a 16-bit network address and route to the destination node have been established.

If the destination 16-bit network address is not known, network address discovery will take place. If a route is not known, route discovery will take place for the purpose of establishing a route to the destination node. If a module with a matching network address is not discovered, the packet is discarded. The data will be transmitted once a route is established. If route discovery fails to establish a route, the packet will be discarded.

Figure 2-04. Transmit Mode Sequence



When data is transmitted from one node to another, a network-level acknowledgement is transmitted back across the established route to the source node. This acknowledgement packet indicates to the source node that the data packet was received by the destination node. If a network acknowledgement is not received, the source node will re-transmit the data.

It is possible in rare circumstances for the destination to receive a data packet, but for the source to not receive the network acknowledgment. In this case, the source will retransmit the data, which could cause the destination to receive the same data packet multiple times. The XBee modules do not filter out duplicate packets. The application should include provisions to address this potential issue

See Data Transmission and Routing in chapter 4 for more information.

Receive Mode

If a valid RF packet is received, the data is transferred to the serial transmit buffer.

Command Mode

To modify or read RF Module parameters, the module must first enter into Command Mode - a state in which incoming serial characters are interpreted as commands. Refer to the API Mode section in Chapter 9 for an alternate means of configuring modules.

AT Command Mode

To Enter AT Command Mode:
Send the 3-character command sequence "+++" and observe guard times before and after the command characters. [Refer to the "Default AT Command Mode Sequence" below.]

Default AT Command Mode Sequence (for transition to Command Mode):

- No characters sent for one second [GT (Guard Times) parameter = 0x3E8]
- Input three plus characters ("+++") within one second [CC (Command Sequence Character) parameter = 0x2B.]
- No characters sent for one second [GT (Guard Times) parameter = 0x3E8]

Once the AT command mode sequence has been issued, the module sends an "OK\r" out the DOUT pin. The "OK\r" characters can be delayed if the module has not finished transmitting received serial data.

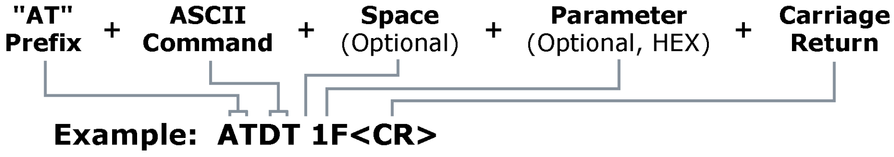
When command mode has been entered, the command mode timer is started (CT command), and the module is able to receive AT commands on the DIN pin.

All of the parameter values in the sequence can be modified to reflect user preferences.

NOTE: Failure to enter AT Command Mode is most commonly due to baud rate mismatch. By default, the BD (Baud Rate) parameter = 3 (9600 bps).

To Send AT Commands:
Send AT commands and parameters using the syntax shown below.

Figure 2-05. Syntax for sending AT Commands



To read a parameter value stored in the RF module's register, omit the parameter field.

The preceding example would change the RF module Destination Address (Low) to "0x1F". To store the new value to non-volatile (long term) memory, subsequently send the WR (Write) command.

For modified parameter values to persist in the module’s registry after a reset, changes must be saved to non-volatile memory using the WR (Write) Command. Otherwise, parameters are restored to previously saved values after the module is reset.

Command Response

When a command is sent to the module, the module will parse and execute the command. Upon successful execution of a command, the module returns an “OK” message. If execution of a command results in an error, the module returns an “ERROR” message.

Applying Command Changes

Any changes made to the configuration command registers through AT commands will not take effect until the changes are applied. For example, sending the BD command to change the baud rate will not change the actual baud rate until changes are applied. Changes can be applied in one of the following ways:

- The AC (Apply Changes) command is issued.
- AT command mode is exited.

To Exit AT Command Mode:

1. Send the ATCN (Exit Command Mode) command (followed by a carriage return).

   [OR]

2. If no valid AT Commands are received within the time specified by CT (Command Mode Timeout) Command, the RF module automatically returns to Idle Mode.

For an example of programming the RF module using AT Commands and descriptions of each configurable parameter, refer to the "Examples" and "XBee Command Reference Tables" chapters.

Sleep Mode

Sleep modes allow the RF module to enter states of low power consumption when not in use. The XBee RF modules support both pin sleep (sleep mode entered on pin transition) and cyclic sleep (module sleeps for a fixed time). XBee sleep modes are discussed in detail in section 6.

3. XBee ZigBee Networks

Introduction to ZigBee

ZigBee is an open global standard built on the IEEE 802.15.4 Mac/Phy. ZigBee defines a network layer above the 802.15.4 layers to support advanced mesh routing capabilities. The ZigBee specification is developed by a growing consortium of companies that make up the ZigBee Alliance. The Alliance is made up of over 300 members, including semiconductor, module, stack, and software developers.

ZigBee Stack Layers

The ZigBee stack consists of several layers including the PHY, MAC, Network, Application Support Sublayer (APS), and ZigBee Device Objects (ZDO) layers. Technically, an Application Framework (AF) layer also exists, but will be grouped with the APS layer in remaining discussions. The ZigBee layers are shown in the figure below.

A description of each layer appears in the following table:

ZigBee Layer	Description
PHY	Defines the physical operation of the ZigBee device including receive sensitivity, channel rejection, output power, number of channels, chip modulation, and transmission rate specifications. Most ZigBee applications operate on the 2.4 GHz ISM band at a 250kbps data rate. See the IEEE 802.15.4 specification for details.
MAC	Manages RF data transactions between neighboring devices (point to point). The MAC includes services such as transmission retry and acknowledgment management, and collision avoidance techniques (CSMA-CA).
Network	Adds routing capabilities that allows RF data packets to traverse multiple devices (multiple "hops") to route data from source to destination (peer to peer).
APS (AF)	Application layer that defines various addressing objects including profiles, clusters, and endpoints.
ZDO	Application layer that provides device and service discovery features and advanced network management capabilities.

Networking Concepts

Device Types

ZigBee defines three different device types: coordinator, router, and end devices.

Node Types / Sample of a Basic ZigBee Network Topology

A **coordinator** has the following characteristics: it

- Selects a channel and PAN ID (both 64-bit and 16-bit) to start the network
- Can allow routers and end devices to join the network
- Can assist in routing data
- Cannot sleep--should be mains powered.

A **router** has the following characteristics: it

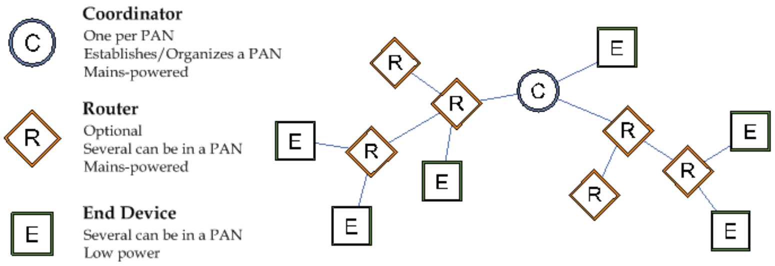
- Must join a ZigBee PAN before it can transmit, receive, or route data

- After joining, can allow routers and end devices to join the network
- After joining, can assist in routing data
- Cannot sleep--should be mains powered.

A **end device** has the following characteristics: it

- Must join a ZigBee PAN before it can transmit or receive data
- Cannot allow devices to join the network
- Must always transmit and receive RF data through its parent. Cannot route data.
- Can enter low power modes to conserve power and can be battery-powered.

An example of such a network is shown below:



In ZigBee networks, the coordinator must select a PAN ID (64-bit and 16-bit) and channel to start a network. After that, it behaves essentially like a router. The coordinator and routers can allow other devices to join the network and can route data.

After an end device joins a router or coordinator, it must be able to transmit or receive RF data through that router or coordinator. The router or coordinator that allowed an end device to join becomes the "parent" of the end device. Since the end device can sleep, the parent must be able to buffer or retain incoming data packets destined for the end device until the end device is able to wake and receive the data.

PAN ID

ZigBee networks are called personal area networks or PANs. Each network is defined with a unique PAN identifier (PAN ID). This identifier is common among all devices of the same network. ZigBee devices are either preconfigured with a PAN ID to join, or they can discovery nearby networks and select a PAN ID to join.

ZigBee supports both a 64-bit and a 16-bit PAN ID. Both PAN IDs are used to uniquely identify a network. Devices on the same ZigBee network must share the same 64-bit and 16-bit PAN IDs. If multiple ZigBee networks are operating within range of each other, each should have unique PAN IDs.

The 16-bit PAN ID is used as a MAC layer addressing field in all RF data transmissions between devices in a network. However, due to the limited addressing space of the 16-bit PAN ID (65,535 possibilities), there is a possibility that multiple ZigBee networks (within range of each other) could use the same 16-bit PAN ID. To resolve potential 16-bit PAN ID conflicts, the ZigBee Alliance created a 64-bit PAN ID.

The 64-bit PAN ID (also called the extended PAN ID), is intended to be a unique, non-duplicated value. When a coordinator starts a network, it can either start a network on a preconfigured 64-bit PAN ID, or it can select a random 64-bit PAN ID. The 64-bit PAN ID is used during joining; if a device has a preconfigured 64-bit PAN ID, it will only join a network with the same 64-bit PAN ID. Otherwise, a device could join any detected PAN and inherit the PAN ID from the network when it joins. The 64-bit PAN ID is included in all ZigBee beacons and is used in 16-bit PAN ID conflict resolution.

Routers and end devices are typically configured to join a network with any 16-bit PAN ID as long as the 64-bit PAN ID is valid. Coordinators typically select a random 16-bit PAN ID for their network.

Since the 16-bit PAN ID only allows up to 65,535 unique values, and since the 16-bit PAN ID is randomly selected, provisions exist in ZigBee to detect if two networks (with different 64-bit PAN IDs) are operating on the same 16-bit PAN ID. If such a conflict is detected, the ZigBee stack can perform PAN ID conflict resolution to change the 16-bit PAN ID of the network in order to resolve the conflict. See the ZigBee specification for details.

To summarize, ZigBee routers and end devices should be configured with the 64-bit PAN ID of the network they want to join. They typically acquire the 16-bit PAN ID when they join a network.

Operating Channel

ZigBee utilizes direct-sequence spread spectrum modulation and operates on a fixed channel. The 802.15.4 PHY defines 16 operating channels in the 2.4 GHz frequency band. XBee modules support all 16 channels and XBee-PRO modules support 14 of the 16 channel

ZigBee Application Layers: In Depth

This section provides a more in-depth look at the ZigBee application stack layers (APS, ZDO) including a discussion on ZigBee endpoints, clusters, and profiles. Much of the material in this section can introduce unnecessary details of the ZigBee stack that are not required in many cases.

Skip this section if

- The XBee does not need to interoperate or talk to non-Digi ZigBee devices
- The XBee simply needs to send data between devices.

Read this section if

- The XBee may talk to non-Digi ZigBee devices
- The XBee requires network management and discovery capabilities of the ZDO layer
- The XBee is designed to operate in a public application profile (smart energy, home automation, etc.)

Application Support Sublayer (APS)

The APS layer in ZigBee adds support for endpoints, cluster IDs and application profiles. A brief discussion of each follows.

Application Profiles

Application profiles specify various device descriptions including required functionality for various devices. The collection of device descriptions forms an application profile. Application profiles can be defined as "Public" or "Private" profiles. Private profiles are defined by a manufacturer whereas public profiles are defined, developed, and maintained by the ZigBee Alliance. Each application profile has a unique profile identifier assigned by the ZigBee Alliance.

Examples of public profiles include:

- Home Automation
- Smart Energy
- Commercial Building Automation

The Smart Energy profile, for example, defines various device types including an energy service portal, load controller, thermostat, in-home display, etc. The Smart Energy profile defines required functionality for each device type. For example, a load controller must respond to a defined command to turn a load on or off. By defining standard communication protocols and device functionality, public profiles allow interoperable ZigBee solutions to be developed by independent manufacturers.

Digi XBee ZB firmware operates on a private profile called the Digi Drop-In Networking profile. However, the API firmware in the module can be used in many cases to talk to devices in public profiles or non-Digi private profiles. See the API chapter for details.

Clusters

A cluster is an application message type defined within a profile. Clusters are used to specify a unique function, service, or action. For example, the following are some clusters defined in the home automation profile:

- On/Off - Used to switch devices on or off (lights, thermostats, etc)
- Level Control - Used to control devices that can be set to a level between on and off
- Color Control - Controls the color of color capable devices.

Each cluster has an associated 2-byte cluster identifier (cluster ID). The cluster ID is included in all application transmissions. Clusters often have associated request and response messages. For example, a smart energy gateway (service portal) might send a load control event to a load controller in order to schedule turning on or off an appliance. Upon executing the event, the load controller would send a load control report message back to the gateway.

Devices that operate in an application profile (private or public) must respond correctly to all required clusters. For example, a light switch that will operate in the home automation public profile must correctly implement the On/Off and other required clusters in order to interoperate with other home automation devices. The ZigBee Alliance has defined a ZigBee Cluster Library (ZCL) that contains definitions or various general use clusters that could be implemented in any profile.

XBee modules implement various clusters in the Digi private profile. In addition, the API can be used to send or receive messages on any cluster ID (and profile ID or endpoint). See the Explicit Addressing ZigBee Command API frame in chapter 3 for details.

Endpoints

The APS layer includes supports for endpoints. An endpoint can be thought of as a running application, similar to a TCP/IP port. A single device can support one or more endpoints. Each application endpoint is identified by a 1-byte value, ranging from 1 to 240. Each defined endpoint on a device is tied to an application profile. A device could, for example, implement one endpoint that supports a Smart Energy load controller, and another endpoint that supports other functionality on a private profile.

ZigBee Device Profile

Profile ID 0x0000 is reserved for the ZigBee Device Profile. This profile is implemented on all ZigBee devices. The ZigBee Device Profile defines a number of device and service discovery features and network management capabilities. Endpoint 0 is a reserved endpoint that supports the ZigBee Device Profile. This endpoint is called the ZigBee Device Objects (ZDO) endpoint.

ZigBee Device Objects (ZDO)

The ZDO (endpoint 0) supports the discovery and management capabilities of the ZigBee Device Profile. A complete listing of all ZDP services is included in the ZigBee specification. Each service has an associated cluster ID.

The XBee ZB firmware allows applications to easily send ZDO messages to devices in the network using the API. See the "Management Transmissions" section in chapter 4 for details.

Coordinator Operation

Forming a Network

The coordinator is responsible for selecting the channel, PAN ID (16-bit and 64-bit), security policy, and stack profile for a network. Since a coordinator is the only device type that can start a network, each ZigBee network must have one coordinator. After the coordinator has started a network, it can allow new devices to join the network. It can also route data packets and communicate with other devices on the network.

To ensure the coordinator starts on a good channel and unused PAN ID, the coordinator performs a series of scans to discover any RF activity on different channels (energy scan) and to discover any nearby operating PANs (PAN scan). The process for selecting the channel and PAN ID are described in the following sections.

Channel Selection

When starting a network, the coordinator must select a "good" channel for the network to operate on. To do this, it performs an energy scan on multiple channels (frequencies) to detect energy levels on each channel. Channels with excessive energy levels are removed from its list of potential channels to start on.

PAN ID Selection

After completing the energy scan, the coordinator scans its list of potential channels (remaining channels after the energy scan) to obtain a list of neighboring PANs. To do this, the coordinator sends a beacon request (broadcast) transmission on each potential channel. All nearby coordinators and routers (that have already joined a ZigBee network) will respond to the beacon request by sending a beacon back to the coordinator. The beacon contains information about the PAN the device is on, including the PAN identifiers (16-bit and 64-bit). This scan (collecting beacons on the potential channels) is typically called an active scan or PAN scan.

After the coordinator completes the channel and PAN scan, it selects a random channel and unused 16-bit PAN ID to start on.

Security Policy

The security policy determines which devices are allowed to join the network, and which device(s) can authenticate joining devices. See Chapter 5 for a detailed discussion of various security policies.

Persistent Data

Once a coordinator has started a network, it retains the following information through power cycle or reset events:

- PAN ID
- Operating channel
- Security policy and frame counter values
- Child table (end device children that are joined to the coordinator).

The coordinator will retain this information indefinitely until it leaves the network. When the coordinator leaves a network and starts a new network, the previous PAN ID, operating channel, and child table data are lost.

XBee ZB Coordinator Startup

The following commands control the coordinator network formation process.

Network formation commands used by the coordinator to form a network.

Command	Description
ID	Used to determine the 64-bit PAN ID. If set to 0 (default), a random 64-bit PAN ID will be selected.
SC	Determines the scan channels bitmask (up to 16 channels) used by the coordinator when forming a network. The coordinator will perform an energy scan on all enabled SC channels. It will then perform a PAN ID scan and then form the network on one of the SC channels.
SD	Set the scan duration period. This value determines how long the coordinator performs an energy scan or PAN ID scan on a given channel.
ZS	Set the ZigBee stack profile for the network.
EE	Enable or disable security in the network.

NK	Set the network security key for the network. If set to 0 (default), a random network security key will be used.
KY	Set the trust center link key for the network. If set to 0 (default), a random link key will be used.
EO	Set the security policy for the network.

Once the coordinator starts a network, the network configuration settings and child table data persist through power cycles as mentioned in the "Persistent Data" section.

When the coordinator has successfully started a network, it

- Allows other devices to join the network for a time. (see NJ command.)
- Sets AI=0
- Starts blinking the Associate LED
- Sends an API modem status frame ("coordinator started") out the UART (API firmware only).

These behaviors are configurable using the following commands:

Command	Description
NJ	Sets the permit-join time on the coordinator, measured in seconds.
D5	Enables the Associate LED functionality.
LT	Sets the Associate LED blink time when joined. Default is 1 blink per second.

If any of the command values innthe network formation commands table changes, the coordinator will leave its current network and start a new network, possibly on a different channel. Note that command changes must be applied (AC or C

N command) before taking effect.

Permit Joining

The permit joining attribute on the coordinator is configurable with the NJ command. NJ can be configured to always allow joining, or to allow joining for a short time.

Joining Always Enabled

If NJ=0xFF (default), joining is permanently enabled. This mode should be used carefully. Once a network has been deployed, the application should strongly consider disabling joining to prevent unwanted joins from occurring.

Joining Temporarily Enabled

If NJ < 0xFF, joining will be enabled only for a number of seconds, based on the NJ parameter. The timer is started once the XBee joins a network. Joining will not be re-enabled if the module is power cycled or reset. The following mechanisms can restart the permit-joining timer:

- Changing NJ to a different value (and applying changes with the AC or CN commands)
- Pressing the commissioning button twice (enables joining for 1 minute)
- Issuing the CB command with a parameter of 2 (software emulation of a 2 button press - enables joining for 1 minute).

Resetting the Coordinator

When the coordinator is reset or power cycled, it checks its PAN ID, operating channel and stack profile against the network configuration settings (ID, SC, ZS). It also verifies the saved security policy against the security configuration commands (EE, NK, KY). If the coordinator's PAN ID, operating channel, stack profile, or security policy is not valid based on its network formation commands (table xyz), then the coordinator will leave the network and attempt to form a new network based on its network formation command values.

To prevent the coordinator from leaving an existing network, the WR command should be issued after all network formation commands have been configured in order to retain these settings through power cycle or reset events.

Leaving a Network

There are a couple of mechanisms that will cause the coordinator to leave its current PAN and start a new network based on its network formation parameter values. These include the following:

- Change the ID command such that the current 64-bit PAN ID is invalid.
- Change the SC command such that the current channel (CH) is not included in the channel mask.
- Change the ZS or any of the security command values (excluding NK).
- Issue the NR0 command to cause the coordinator to leave.
- Issue the NR1 command to send a broadcast transmission, causing all devices in the network to leave and migrate to a different channel.
- Press the commissioning button 4 times or issue the CB command with a parameter of 4.

Note that changes to ID, SC, ZS, and security command values only take effect when changes are applied (AC or CN commands).

Replacing a Coordinator (Security Disabled Only)

In rare occasions, it may become necessary to replace an existing coordinator in a network with a new physical device. If security is not enabled in the network, a replacement XBee coordinator can be configured with the PAN ID (16-bit and 64-bit), channel, and stack profile settings of a running network in order to replace an existing coordinator.

NOTE: Having two coordinators on the same channel, stack profile, and PAN ID (16-bit and 64-bit) can cause problems in the network and should be avoided. When replacing a coordinator, the old coordinator should be turned off before starting the new coordinator.

To replace a coordinator, the following commands should be read from a device on the network:

AT Command	Description
OP	Read the operating 64-bit PAN ID.
OI	Read the operating 16-bit PAN ID.
CH	Read the operating channel.
ZS	Read the stack profile.

Each of the commands listed above can be read from any device on the network. (These parameters will be the same on all devices in the network.) After reading these commands from a device on the network, these parameter values should be programmed into the new coordinator using the following commands.

AT Command	Description
ID	Set the 64-bit PAN ID to match the read OP value.
II	Set the initial 16-bit PAN ID to match the read OI value.
SC	Set the scan channels bitmask to enable the read operating channel (CH command). For example, if the operating channel is 0x0B, set SC to 0x0001. If the operating channel is 0x17, set SC to 0x1000.

ZS	Set the stack profile to match the read ZS value.

Note: II is the initial 16-bit PAN ID. Under certain conditions, the ZigBee stack can change the 16-bit PAN ID of the network. For this reason, the II command cannot be saved using the WR command. Once II is set, the coordinator leaves the network and starts on the 16-bit PAN ID specified by II.

Example: Starting a Coordinator

1. Set SC and ID to the desired scan channels and PAN ID values. (The defaults should suffice.)

2. If SC or ID is changed from the default, issue the WR command to save the changes.

3. If SC or ID is changed from the default, apply changes (make SC and ID changes take effect) either by sending the AC command or by exiting AT command mode.

4. The Associate LED will start blinking once the coordinator has selected a channel and PAN ID.

5. The API Modem Status frame ("Coordinator Started") is sent out the UART (API firmware only).

6. Reading the AI command (association status) will return a value of 0, indicating a successful startup.

7. Reading the MY command (16-bit address) will return a value of 0, the ZigBee-defined 16-bit address of the coordinator

After startup, the coordinator will allow joining based on its NJ value.

Example: Replacing a Coordinator (security disabled)

1. Read the OP, OI, CH, and ZS commands on the running coordinator.

2. Set the ID, SC, and ZS parameters on the new coordinator, followed by WR command to save these parameter values.

3. Turn off the running coordinator.

4. Set the II parameter on the new coordinator to match the read OI value on the old coordinator.

5. Wait for the new coordinator to start (AI=0).

Router Operation

Joining a Network

Routers must discover and join a valid ZigBee network before they can participate in a ZigBee network. After a router has joined a network, it can allow new devices to join the network. It can also route data packets and communicate with other devices on the network.

Discovering ZigBee Networks

To discover nearby ZigBee networks, the router performs a PAN (or active) scan, just like the coordinator does when it starts a network. During the PAN scan, the router sends a beacon request (broadcast) transmission on the first channel in its scan channels list. All nearby coordinators and routers operating on that channel (that are already part of a ZigBee network) respond to the beacon request by sending a beacon back to the router. The beacon contains information about the PAN the nearby device is on, including the PAN identifier (PAN ID), and whether or not joining is allowed. The router evaluates each beacon received on the channel to determine if a valid PAN is found. A router considers a PAN to be valid if the PAN:

- Has a valid 64-bit PAN ID (PAN ID matches ID if ID > 0)
- Has the correct stack profile (ZS command)
- Is allowing joining.

If a valid PAN is not found, the router performs the PAN scan on the next channel in its scan channels list and continues scanning until a valid network is found, or until all channels have been scanned. If all channels have been scanned and a valid PAN was not discovered, all channels will be scanned again.

The ZigBee Alliance requires certified solutions not send beacon request messages to frequently. To meet certification requirements, the XBee firmware attempts 9 scans per minute for the first 5 minutes, and 3 scans per minute thereafter. If a valid PAN is within range of a joining router, it should typically be discovered within a few seconds.

Joining a Network

Once the router discovers a valid network, it sends an association request to the device that sent a valid beacon requesting a join on the ZigBee network. The device allowing the join then sends an association response frame that either allows or denies the join.

When a router joins a network, it receives a 16-bit address from the device that allowed the join. The 16-bit address is randomly selected by the device that allowed the join.

Authentication

In a network where security is enabled, the router must then go through an authentication process. See the Security chapter for a discussion on security and authentication.

After the router is joined (and authenticated, in a secure network), it can allow new devices to join the network.

Persistent Data

Once a router has joined a network, it retains the following information through power cycle or reset events:

- PAN ID
- Operating channel
- Security policy and frame counter values
- Child table (end device children that are joined to the coordinator).

The router will retain this information indefinitely until it leaves the network. When the router leaves a network, the previous PAN ID, operating channel, and child table data are lost.

XBee ZB Router Joining

When the router is powered on, if it is not joined to a valid ZigBee network, it immediately attempts to find and join a valid ZigBee network.

Note: The DJ command can be set to 1 to disable joining. The DJ parameter cannot be written with WR, so a power cycle always clears the DJ setting.

The following commands control the router joining process.

Command	Description
ID	Sets the 64-bit PAN ID to join. Setting ID=0 allows the router to join any 64-bit PAN ID.
SC	Set the scan channels bitmask that determines which channels a router will scan to find a valid network. SC on the router should be set to match SC on the coordinator. For example, setting SC to 0x281 enables scanning on channels 0x0B, 0x12, and 0x14, in that order.
SD	Set the scan duration, or time that the router will listen for beacons on each channel.
ZS	Set the stack profile on the device.
EE	Enable or disable security in the network. This must be set to match the EE value (security policy) of the coordinator.
KY	Set the trust center link key. If set to 0 (default), the link key is expected to be obtained (unencrypted) during joining.

Once the router joins a network, the network configuration settings and child table data persist through power cycles as mentioned in the "Persistent Data" section previously. If joining fails, the status of the last join attempt can be read in the AI command register.

If any of the above command values change, when command register changes are applied (AC or CN commands), the router will leave its current network and attempt to discover and join a new valid network.

When a ZB router has successfully joined a network, it:

- Allows other devices to join the network for a time
- Sets AI=0
- Starts blinking the Associate LED
- Sends an API modem status frame ("associated") out the UART (API firmware only).

These behaviors are configurable using the following commands:

Command	Description
NJ	Sets the permit-join time on the router, or the time that it will allow new devices to join the network, measured in seconds. If NJ=0xFF, permit joining will always be enabled.
D5	Enables the Associate LED functionality.
LT	Sets the Associate LED blink time when joined. Default is 2 blinks per second (router).

Permit Joining

The permit joining attribute on the router is configurable with the NJ command. NJ can be configured to always allow joining, or to allow joining for a short time.

Joining Always Enabled

If NJ=0xFF (default), joining is permanently enabled. This mode should be used carefully. Once a network has been deployed, the application should strongly consider disabling joining to prevent unwanted joins from occurring.

Joining Temporarily Enabled

If NJ < 0xFF, joining will be enabled only for a number of seconds, based on the NJ parameter. The timer is started once the XBee joins a network. Joining will not be re-enabled if the module is power cycled or reset. The following mechanisms can restart the permit-joining timer:

- Changing NJ to a different value (and applying changes with the AC or CN commands)
- Pressing the commissioning button twice (enables joining for 1 minute)
- Issuing the CB command with a parameter of 2 (software emulation of a 2 button press - enables joining for 1 minute).
- Causing the router to leave and rejoin the network.

Resetting the Router

When the router is reset or power cycled, it checks its PAN ID, operating channel and stack profile against the network configuration settings (ID, SC, ZS). It also verifies the saved security policy is valid based on the security configuration commands (EE, KY). If the router's PAN ID, operating channel, stack profile, or security policy is invalid, the router will leave the network and attempt to join a new network based on its network joining command values.

To prevent the router from leaving an existing network, the WR command should be issued after all network joining commands have been configured in order to retain these settings through power cycle or reset events.

Leaving a Network

There are a couple of mechanisms that will cause the router to leave its current PAN and attempt to discover and join a new network based on its network joining parameter values (see table xyz). These include the following:

- Change the ID command such that the current 64-bit PAN ID is invalid.
- Change the SC command such that the current channel (CH) is not included in the channel mask.

- Change the ZS or any of the security command values.
- Issue the NR0 command to cause the router to leave.
- Issue the NR1 command to send a broadcast transmission, causing all devices in the network to leave and migrate to a different channel.
- Press the commissioning button 4 times or issue the CB command with a parameter of 4.

Note that changes to ID, SC, ZS, and security command values only take effect when when changes are applied (AC or CN commands).

Example: Joining a Network

After starting a coordinator (that is allowing joins), the following steps will cause an XBee router to join the network:

1. Set ID to the desired 64-bit PAN ID, or to 0 to join any PAN.

2. Set SC to the list of channels to scan to find a valid network.

3. If SC or ID is changed from the default, apply changes (make SC and ID changes take effect) by issuing the AC or CN command.

4. The Associate LED will start blinking once the router has joined a PAN.

5. If the Associate LED is not blinking, the AI command can be read to determine the cause of join failure.

6. Once the router has joined, the OP and CH commands will indicate the operating 64-bit PAN ID and channel the router joined.

7. The MY command will reflect the 16-bit address the router received when it joined.

8. The API Modem Status frame ("Associated") is sent out the UART (API firmware only).

9. The joined router will allow other devices to join for a time based on its NJ setting.

End Device Operation

Joining a Network

Similar to routers, end devices must also discover and join a valid ZigBee network before they can participate in a network. After an end device has joined a network, it can communicate with other devices on the network. Since end devices are intended to be battery powered and therefore support low power (sleep) modes, end devices cannot allow other devices to join, nor can they route data packets.

Discovering ZigBee Networks

End devices go through the same process as routers to discover networks by issuing a PAN scan. After sending the broadcast beacon request transmission, the end device listens for a short time in order to receive beacons sent by nearby routers and coordinators on the same channel. The end device evaluates each beacon received on the channel to determine if a valid PAN is found. An end device considers a PAN to be valid if the PAN:

- Has a valid 64-bit PAN ID (PAN ID matches ID if ID > 0)
- Has the correct stack profile (ZS command)
- Is allowing joining
- Has capacity for additional end devices (see End Device Capacity section below).

If a valid PAN is not found, the end device performs the PAN scan on the next channel in its scan channels list and continues this process until a valid network is found, or until all channels have been scanned. If all channels have been scanned and a valid PAN was not discovered, the end device may enter a low power sleep state and scan again later.

If scanning all SC channels fails to discover a valid PAN, XBee ZB modules will attempt to enter a low power state and will retry scanning all SC channels after the module wakes from sleeping. If the module cannot enter a low power state, it will retry scanning all channels, similar to the router.

To meet ZigBee Alliance requirements, the end device will attempt up to 9 scans per minute for the first 5 minutes, and 3 scans per minute thereafter.

Note - The XBee ZB end device will not enter sleep until it has completed scanning all SC channels for a valid network.

Joining a Network

Once the end device discovers a valid network, it joins the network, similar to a router, by sending an association request (to the device that sent a valid beacon) to request a join on the ZigBee network. The device allowing the join then sends an association response frame that either allows or denies the join.

When an end device joins a network, it receives a 16-bit address from the device that allowed the join. The 16-bit address is randomly selected by the device that allowed the join.

Parent Child Relationship

Since an end device may enter low power sleep modes and not be immediately responsive, the end device relies on the device that allowed the join to receive and buffer incoming messages in its behalf until it is able to wake and receive those messages. The device that allowed an end device to join becomes the parent of the end device, and the end device becomes a child of the device that allowed the join.

End Device Capacity

Routers and coordinators maintain a table of all child devices that have joined called the child table. This table is a finite size and determines how many end devices can join. If a router or coordinator has at least one unused entry in its child table, the device is said to have end device capacity. In other words, it can allow one or more additional end devices to join. ZigBee networks should have sufficient routers to ensure adequate end device capacity.

In ZB firmware, the NC command (number of remaining end device children) can be used to determine how many additional end devices can join a router or coordinator. If NC returns 0, then the router or coordinator device has no more end device capacity. (Its child table is full.)

Also of note, since routers cannot sleep, there is no equivalent need for routers or coordinators to track joined routers. Therefore, there is no limit to the number of routers that can join a given router or coordinator device. (There is no "router capacity" metric.)

Authentication

In a network where security is enabled, the end device must then go through an authentication process. See chapter 5 for a discussion on security and authentication.

Persistent Data

The end device can retain its PAN ID, operating channel, and security policy information through a power cycle. However, since end devices rely heavily on a parent, the end device does an orphan scan to try and contact its parent. If the end device does not receive an orphan scan response (called a coordinator realignment command), it will leave the network and try to discover and join a new network. When the end device leaves a network, the previous PAN ID and operating channel settings are lost.

Orphan Scans

When an end device comes up from a power cycle, it performs an orphan scan to verify it still has a valid parent. The orphan scan is sent as a broadcast transmission and contains the 64-bit address of the end device. Nearby routers and coordinator devices that receive the broadcast check their child tables for an entry that contains the end device's 64-bit address. If an entry is found with a matching 64-bit address, the device sends a coordinator realignment command to the end device that includes the end device's 16-bit address, 16-bit PAN ID, operating channel, and the parent's 64-bit and 16-bit addresses.

If the orphaned end device receives a coordinator realignment command, it is considered joined to the network. Otherwise, it will attempt to discover and join a valid network.

XBee: ZB End Device Joining

When an end device is powered on, if it is not joined to a valid ZigBee network, or if the orphan scan fails to find a parent, it immediately attempts to find and join a valid ZigBee network.

Note: The DJ command can be set to 1 to disable joining. The DJ parameter cannot be written with WR, so a power cycle always clears the DJ setting.

Similar to a router, the following commands control the end device joining process.

Network joining commands used by an end device to join a network.

Command	Description
ID	Sets the 64-bit PAN ID to join. Setting ID=0 allows the router to join any 64-bit PAN ID.
SC	Set the scan channels bitmask that determines which channels an end device will scan to find a valid network. SC on the end device should be set to match SC on the coordinator and routers in the desired network. For example, setting SC to 0x281 enables scanning on channels 0x0B, 0x12, and 0x14, in that order.
SD	Set the scan duration, or time that the end device will listen for beacons on each channel.
ZS	Set the stack profile on the device.
EE	Enable or disable security in the network. This must be set to match the EE value (security policy) of the coordinator.
KY	Set the trust center link key. If set to 0 (default), the link key is expected to be obtained (unencrypted) during joining.

Once the end device joins a network, the network configuration settings can persist through power cycles as mentioned in the "Persistent Data" section previously. If joining fails, the status of the last join attempt can be read in the AI command register.

If any of these command values changes, when command register changes are applied, the end device will leave its current network and attempt to discover and join a new valid network.

When a ZB end device has successfully started a network, it

- Sets AI=0.
- Starts blinking the Associate LED
- Sends an API modem status frame ("associated") out the UART (API firmware only)
- Attempts to enter low power modes.

These behaviors are configurable using the following commands:

Command	Description
D5	Enables the Associate LED functionality.
LT	Sets the Associate LED blink time when joined. Default is 2 blinks per second (end devices).
SM, SP, ST, SN, SO	Parameters that configure the sleep mode characteristics. (See Managing End Devices chapter for details.)

Parent Connectivity

The XBee ZB end device sends regular poll transmissions to its parent when it is awake. These poll transmissions query the parent for any new received data packets. The parent always sends a MAC layer acknowledgment back to the end device. The acknowledgment indicates if the parent has data for the end device or not.

If the end device does not receive an acknowledgment for 3 consecutive poll requests, it considers itself disconnected from its parent and will attempt to discover and join a valid ZigBee network. See "Managing End Devices" chapter for details.

Resetting the End Device

When the end device is reset or power cycled, if the orphan scan successfully locates a parent, the end device then checks its PAN ID, operating channel and stack profile against the network configuration settings (ID, SC, ZS). It also verifies the saved security policy is valid based on the security configuration commands (EE, KY). If the end device's PAN ID, operating channel, stack profile, or security policy is invalid, the end device will leave the network and attempt to join a new network based on its network joining command values.

To prevent the end device from leaving an existing network, the WR command should be issued after all network joining commands have been configured in order to retain these settings through power cycle or reset events.

Leaving a Network

There are a couple of mechanisms that will cause the router to leave its current PAN and attempt to discover and join a new network based on its network joining parameter values. These include the following:

- The ID command changes such that the current 64-bit PAN ID is invalid.
- The SC command changes such that the current operating channel (CH) is not included in the channel mask.
- The ZS or any of the security command values change.
- The NR0 command is issued to cause the end device to leave.
- The NR1 command is issued to send a broadcast transmission, causing all devices in the network to leave and migrate to a different channel.
- The commissioning button is pressed 4 times or the CB command is issued with a parameter of 4.
- The end device's parent is powered down or the end device is moved out of range of the parent such that the end device fails to receive poll acknowledgment messages.

Note that changes to command values only take effect when changes are applied (AC or CN commands).

Example: Joining a Network

After starting a coordinator (that is allowing joins), the following steps will cause an XBee end device to join the network

1. Set ID to the desired 64-bit PAN ID, or to 0 to join any PAN.

2. Set SC to the list of channels to scan to find a valid network.

3. If SC or ID is changed from the default, apply changes (make SC and ID changes take effect) by issuing the AC or CN command.

4. The Associate LED will start blinking once the end device has joined a PAN.

5. If the Associate LED is not blinking, the AI command can be read to determine the cause of join failure.

6. Once the end device has joined, the OP and CH commands will indicate the operating 64-bit PAN ID and channel the end device joined.

7. The MY command will reflect the 16-bit address the router received when it joined.

8. The API Modem Status frame ("Associated") is sent out the UART (API firmware only).

9. The joined end device will attempt to enter low power sleep modes based on its sleep configuration commands (SM, SP, SN, ST, SO).

Channel Scanning

As mentioned previously, routers and end devices must scan one or more channels to discover a valid network to join. When a join attempt begins, the XBee sends a beacon request transmission on the lowest channel specified in the SC (scan channels) command bitmask. If a valid PAN is found on the channel, the XBee will attempt to join the PAN on that channel. Otherwise, if a valid PAN is not found on the channel, it will attempt scanning on the next higher channel in the SC command bitmask. The XBee will continue to scan each channel (from lowest to highest) in the SC bitmask until a valid PAN is found or all channels have been scanned. Once all channels have been scanned, the next join attempt will start scanning on the lowest channel specified in the SC command bitmask.

For example, if the SC command is set to 0x400F, the XBee would start scanning on channel 11 (0x0B) and scan until a valid beacon is found, or until channels 11, 12, 13, 14, and 25 have been scanned (in that order).

Once an XBee router or end device joins a network on a given channel, if the XBee is told to leave (see "Leaving a Network" section), it will leave the channel it joined on and continue scanning on the next higher channel in the SC bitmask.

For example, if the SC command is set to 0x400F, and the XBee joins a PAN on channel 12 (0x0C), if the XBee leaves the channel, it will start scanning on channel 13, followed by channels 14 and 25 if a valid network is not found. Once all channels have been scanned, the next join attempt will start scanning on the lowest channel specified in the SC command bitmask.

Managing Multiple ZigBee Networks

In some applications, multiple ZigBee networks may exist in proximity of each other. The application may need provisions to ensure the XBee joins the desired network. There are a number of features in ZigBee to manage joining among multiple networks. These include the following:

- PAN ID Filtering
- Preconfigured Security Keys
- Permit Joining
- Application Messaging

PAN ID Filtering

The XBee can be configured with a fixed PAN ID by setting the ID command to a non-zero value. If the PAN ID is set to a non-zero value, the XBee will only join a network with the same PAN ID.

Preconfigured Security Keys

Similar to PAN ID filtering, this method requires a known security key be installed on a router to ensure it will join a ZigBee network with the same security key. If the security key (KY command) is set to a non-zero value, and if security is enabled (EE command), an XBee router or end device will only join a network with the same security key.

Permit Joining

The Permit Joining parameter can be disabled in a network to prevent unwanted devices from joining. When a new device must be added to a network, permit-joining can be enabled for a short time on the desired network. In the XBee firmware, joining is disabled by setting the NJ command to a value less than 0xFF on all routers and coordinator devices. Joining can be enabled for a short time using the commissioning push-button (see Network Commissioning chapter for details) or the CB command.

Application Messaging

If the above mechanisms are not feasible, the application could build in a messaging framework between the coordinator and devices that join its network. For example, the application code in joining devices could send a transmission to the coordinator after joining a network, and wait to receive a defined reply message. If the application does not receive the expected response message after joining, the application could force the XBee to leave and continue scanning. Forcing the XBee to leave will cause the XBee to continue scanning on the next higher channel in the SC bitmask.

4. Data Transmission, Addressing, and Routing

Addressing

All ZigBee devices have two different addresses, a 64-bit and a 16-bit address. The characteristics of each are described below.

64-bit Device Addresses

The 64-bit address is a unique device address assigned during manufacturing. This address is unique to each physical device. The 64-bit address includes a 3-byte Organizationally Unique Identifier (OUI) assigned by the IEEE. The 64-bit address is also called the extended address.

16-bit Device Addresses

A device receives a 16-bit address when it joins a ZigBee network. For this reason, the 16-bit address is also called the "network address". The 16-bit address of 0x0000 is reserved for the coordinator. All other devices receive a randomly generated address from the router or coordinator device that allows the join. The 16-bit address can change under certain conditions:

- An address conflict is detected where two devices are found to have the same 16-bit address
- A device leaves the network and later joins (it can receive a different address)

All ZigBee transmissions are sent using the source and destination 16-bit addresses. The routing tables on ZigBee devices also use 16-bit addresses to determine how to route data packets through the network. However, since the 16-bit address is not static, it is not a reliable way to identify a device.

To solve this problem, the 64-bit destination address is often included in data transmissions to guarantee data is delivered to the correct destination. The ZigBee stack can discover the 16-bit address, if unknown, before transmitting data to a remote.

Application Layer Addressing

ZigBee devices can support multiple application profiles, cluster IDs, and endpoints. (See "ZigBee Application Layers - In Depth" in chapter 3.) Application layer addressing allows data transmissions to be addressed to specific profile IDs, cluster IDs, and endpoints. Application layer addressing is useful if an application must

- Interoperate with other ZigBee devices outside of the Digi application profile
- Utilize service and network management capabilities of the ZDO
- Operate on a public application profile such as Home Controls or Smart Energy.

The API firmware provides a simple yet powerful interface that can easily send data to any profile ID, endpoint, and cluster ID combination on any device in a ZigBee network.

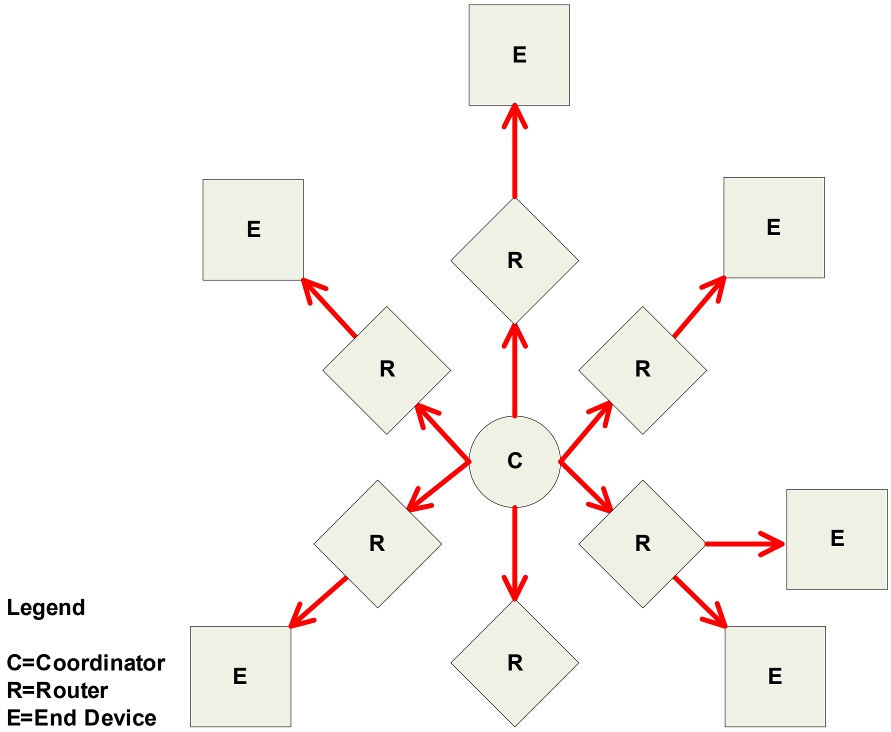
Data Transmission

ZigBee data packets can be sent as either unicast or broadcast transmissions. Unicast transmissions route data from one source device to one destination device, whereas broadcast transmissions are sent to many or all devices in the network.

Broadcast Transmissions

Broadcast transmissions within the ZigBee protocol are intended to be propagated throughout the entire network such that all nodes receive the transmission. To accomplish this, all devices that receive a broadcast transmission will retransmit the packet 3 times.

eBroadcast Data Transmission



Each node that transmits the broadcast will also create an entry in a local broadcast transmission table. This entry is used to keep track of each received broadcast packet to ensure the packets are not endlessly transmitted. Each entry persists for 8 seconds. The broadcast transmission table holds 8 entries.

For each broadcast transmission, the ZigBee stack must reserve buffer space for a copy of the data packet. This copy is used to retransmit the packet as needed.Large broadcast packets will require more buffer space.

Since broadcast transmissions are retransmitted by each device in the network, broadcast messages should be used sparingly.

Unicast Transmissions

Unicast transmissions are sent from one source device to another destination device. The destination device could be an immediate neighbor of the source, or it could be several hops away.

As mentioned previously, each device in a ZigBee network has both a 16-bit (network) address and a 64-bit (extended) address. Unicast transmissions are always addressed and routed to the 16-bit address of the destination. However, to ensure data is received by the correct device, the destination 64-bit address is often included in the RF transmission. If a receiving device has a matching 16-bit address, but not a matching 64-bit address, it will drop the packet and obtain a new 16-bit address.

XBee ZB firmware requires that data be sent to the 64-bit address of the destination device. However, since the actual RF transmission requires 16-bit addressing, the 16-bit address will be discovered by the XBee if unknown.

Address Table

Each ZigBee device maintains an address table that maps a 64-bit address to a 16-bit address. When a transmission is addressed to a 64-bit address, the ZigBee stack searches the address table for an entry with a matching 64-bit address, in hopes of determining the destination's 16-bit address. If a known 16-bit address is not found, the ZigBee stack will perform address discovery to discover the device's current 16-bit address.

Sample Address Table

64-bit Address	16-bit Address
0013 A200 4000 0001	0x4414
0013 A200 400A 3568	0x1234
0013 A200 4004 1122	0xC200
0013 A200 4002 1123	0xFFFE (unknown)

Address Discovery

Address discovery is a service provided in the ZigBee Device Profile that can discover the 16-bit address of a given device, based on its 64-bit address. To discover a 16-bit address of a remote, the device initiating the discovery sends a broadcast address discovery transmission. The address discovery includes the 64-bit address of the remote device whose 16-bit address is being requested.

All nodes that receive this transmission check the 64-bit address in the payload and compare it to their own 64-bit address. If the addresses match, the device sends a response packet back to the initiator. This response includes the remote's 16-bit address.

When the discovery response is received, the initiator will then transmit the data.

Data Transmission Examples

AT Firmware

To send a data packet in AT firmware, the DH and DL commands must be set to match the 64-bit address of the destination device. DH must match the upper 4-bytes, and DL must match the lower 4 bytes. Since the coordinator always receives a 16-bit address of 0x0000, a 64-bit address of 0x0000000000000000 is defined as the coordinator's address (in ZB firmware). The default values of DH and DL are 0x00, which sends data to the coordinator.

Example 1: Send a transmission to the coordinator.

(In this example, a '\r' refers to a carriage return character.)

A router or end device can send data in two ways. First, set the destination address (DH and DL commands) to 0x00.

1. Enter command mode ('+++')
2. After receiving an OK\r, issue the following commands:
   a. ATDH0\r
   b. ATDL0\r
   c. ATCN\r
3. Verify that each of the 3 commands returned an OK\r response.

4. After setting these command values, all serial characters will be sent as a unicast transmission to the coordinator.

Alternatively, if the coordinator's 64-bit address is known, DH and DL can be set to the coordinator's 64-bit address. Suppose the coordinator's address is 0x0013A200404A2244.

1. Enter command mode ('+++')

2. After receiving an OK\r, issue the following commands:

   a. ATDH13A200\r

   b. ATDL404A2244\

   c. ATCN\r

3. Verify that each of the 3 commands returned an OK\r response.

4. After setting these command values, all serial characters will be sent as a unicast transmission to the coordinator.

API Firmware

Use the transmit request, or explicit transmit request frame (0x10 and 0x11 respectively) to send data to the coordinator. The 64-bit address can either be set to 0x0000000000000000, or to the 64-bit address of the coordinator. The 16-bit address should be set to 0xFFFE when using the 64-bit address of all 0x00s.

To send an ascii "1" to the coordinator's 0x00 address, the following API frame can be used:

```
7E 00 0F 10 01  0000 0000 0000 0000 FFFE 00 00 31 C0
```

If the explicit transmit frame is used, the cluster ID should be set to 0x0011, the profile ID to 0xC105, and the source and destination endpoints to 0xE8 (recommended defaults for data transmissions in the Digi profile.) The same transmission could be sent using the following explicit transmit frame:

```
7E 00 15 11 01  0000 0000 0000 0000 FFFE E8 E8 0011 C105 00 00 31 18
```

Notice the 16-bit address is set to 0xFFFE. This is required when sending to a 64-bit address of 0x00s.

Now suppose the coordinator's 64-bit address is 0x0013A200404A2244. The following transmit request API frame (0x10) will send an ASCII "1" to the coordinator:

```
7E 00 0F 10  01 0013 A200 404A 2244 0000 0000 31 18
```

Example 2: Send a broadcast transmission.

(In this example, a '\r' refers to a carriage return character.)

Perform the following steps to configure a broadcast transmission:

1. Enter command mode ('+++')

2. After receiving an OK\r, issue the following commands:

   a. ATDH0\r

   b. ATDLffff\r

   c. ATCN\r

3. Verify that each of the 3 commands returned an OK\r response

4. After setting these command values, all serial characters will be sent as a broadcast transmission.

API Firmware

This example will use the transmit request API frame (0x10) to send an ASCII "1" in a broadcast transmission.

To send an ascii "1" as a broadcast transmission, the following API frame can be used:

```
7E 00 0F 10 01  0000 0000 0000 FFFF FFFE 00 00 31 C2
```

Notice the destination 16-bit address is set to 0xFFFE for broadcast transmissions.

RF Packet Routing

Unicast transmissions may require some type of routing. ZigBee includes several different ways to route data, each with its own advantages and disadvantages. These are summarized in the table below.

Routing Approach	Description	When to Use
Ad hoc On-demand Distance Vector (AODV) Mesh Routing	Routing paths are created between source and destination, possibly traversing multiple nodes ("hops"). Each device knows who to send data to next to eventually reach the destination	Use in networks where data does not need to be routed to many different destinations. The routing table is a finite size and each destination address requires one entry
Many-to-One Routing	A single broadcast transmission configured reverse routes on all devices into the device that sends the broadcast	Useful when many remote devices must send data to a single gateway or collector device.
Source Routing	Data packets include the entire route the packet should traverse to get from source to destination	Improves routing efficiency in large networks (over 40 remote devices)

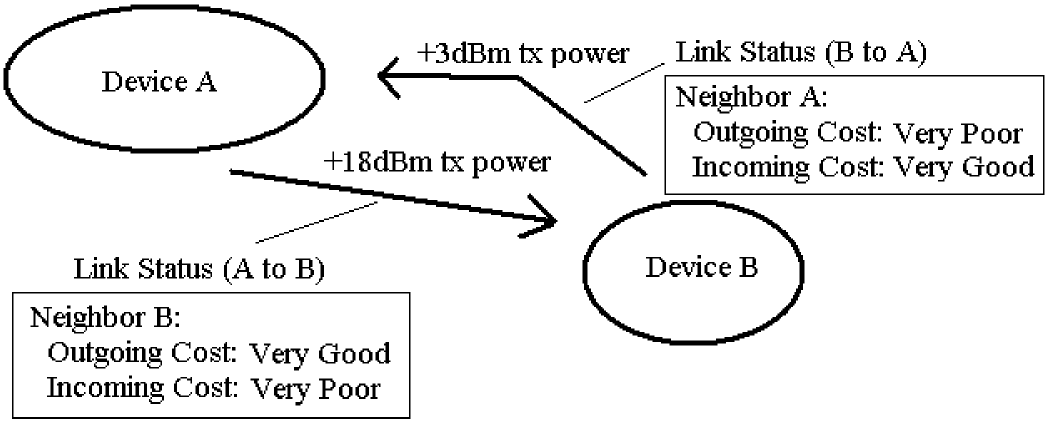
Note – End devices do not make use of these routing protocols. Rather, an end device sends a unicast transmission to its parent and allows the parent to route the data packet in its behalf.

Link Status Transmission

Before discussing the various routing protocols, it is worth understanding the primary mechanism in ZigBee for establishing reliable bi-directional links. This mechanism is especially useful in networks that may have a mixture of devices with varying output power and/or receiver sensitivity levels.

Each coordinator or router device periodically sends a link status message. This message is sent as a 1-hop broadcast transmission, received only by one-hop neighbors. The link status message contains a list of neighboring devices and incoming and outgoing link qualities for each neighbor. Using these messages, neighboring devices can determine the quality of a bi-directional link with each neighbor and use that information to select a route that works well in both directions.

For example, consider a network of two neighboring devices that send periodic link status messages. Suppose that the output power of device A is +18dBm, and the output power of device B is +3dBm (considerably less than the output power of device A). The link status messages might indicate the following:

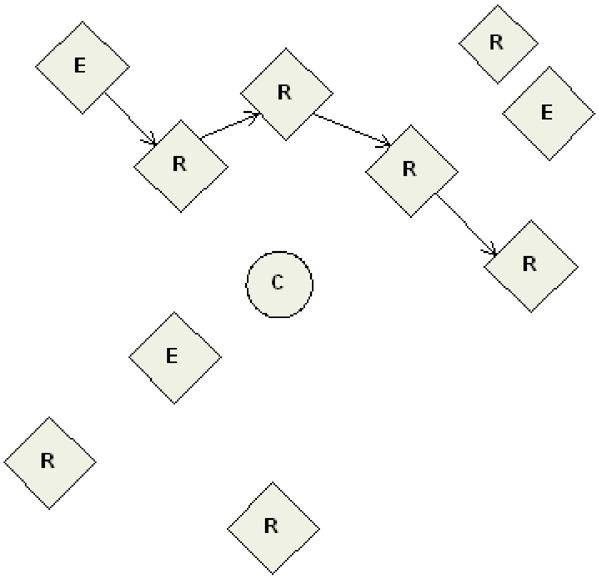


This mechanism enables devices A and B to recognize that the link is not reliable in both directions and select a different neighbor when establishing routes. (Such links are called asymmetric links, meaning the link quality is not similar in both directions.)

AODV Mesh Routing

ZigBee employs mesh routing to establish a route between the source device and the destination. Mesh routing allows data packets to traverse multiple nodes (hops) in a network to route data from a source to a destination. Routers and coordinators can participate in establishing routes between source and destination devices using a process called route discovery. The Route discovery process is based on the AODV (Ad-hoc On-demand Distance Vector routing) protocol.

Figure 4-06. Sample Transmission Through a Mesh Network



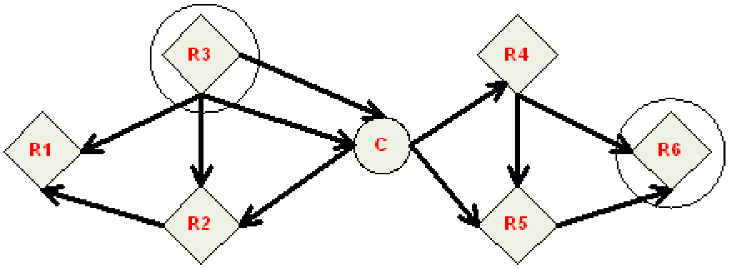
AODV (Ad-hoc On-demand Distance Vector) Routing Algorithm

Routing under the AODV protocol is accomplished using tables in each node that store the next hop (intermediary node between source and destination nodes) for a destination node. If a next hop is not known, route discovery must take place in order to find a path. Since only a limited number of routes can be stored on a Router, route discovery will take place more often on a large network with communication between many different nodes.

Node	Destination Address	Next Hop Address
R3	Router 6	Coordinator
C	Router 6	Router 5
R5	Router 6	Router 6

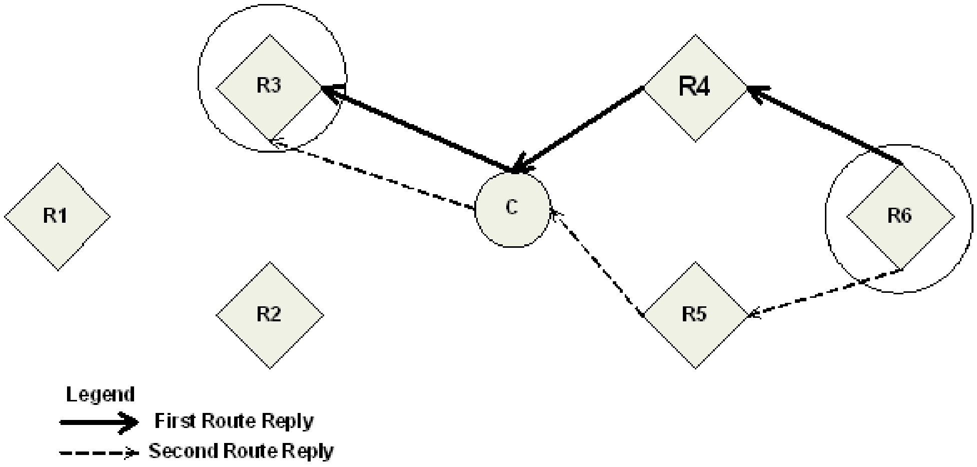
When a source node must discover a route to a destination node, it sends a broadcast route request command. The route request command contains the source network address, the destination network address and a path cost field (a metric for measuring route quality). As the route request command is propagated through the network (refer to the Broadcast Transmission), each node that re-broadcasts the message updates the path cost field and creates a temporary entry in its route discovery table.

Figure 4-07. Sample Route Request (Broadcast) Transmission Where R3 is Trying to Discover a Route to R6t



When the destination node receives a route request, it compares the 'path cost' field against previously received route request commands. If the path cost stored in the route request is better than any previously received, the destination node will transmit a route reply packet to the node that originated the route request. Intermediate nodes receive and forward the route reply packet to the source node (the node that originated route request).

Figure 4-08. Route Reply Sample Route Reply (Unicast) Where R6 Sends a Route Reply to R3.



Note: R6 could send multiple replies if it identifies a better route.

Retries and Acknowledgments

ZigBee includes acknowledgment packets at both the Mac and Application Support (APS) layers. When data is transmitted to remote device, it may traverse multiple hops to reach the destination. As data is transmitted from one node to its neighbor, an acknowledgment packet (Ack) is transmitted in the opposite direction to indicate that the transmission was successfully received. If the Ack is not received, the transmitting device will retransmit the data, up to 4 times. This Ack is called the Mac layer acknowledgment.

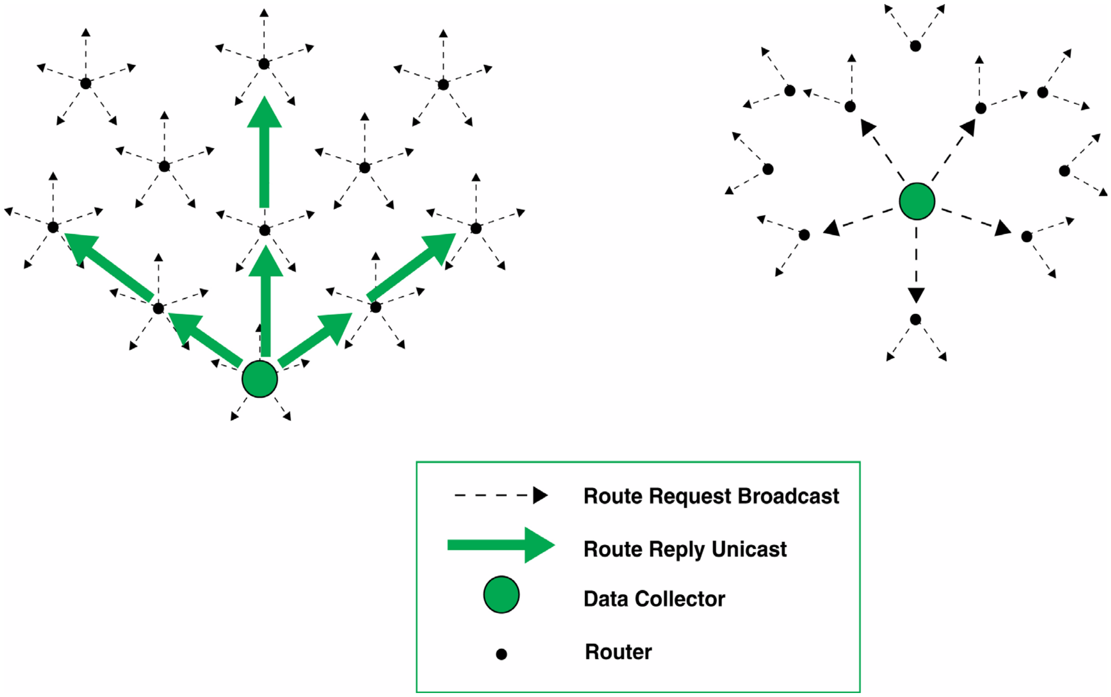
In addition, the device that originated the transmission expects to receive an acknowledgment packet (Ack) from the destination device. This Ack will traverse the same path that the data traversed, but in the opposite direction. If the originator fails to receive this Ack, it will retransmit the data, up to 2 times until an Ack is received. This Ack is called the ZigBee APS layer acknowledgment.

Refer to the ZigBee specification for more details.

Many-to-One Routing

In networks where many devices must send data to a central collector or gateway device, AODV mesh routing requires significant overhead. If every device in the network had to discovery a route before it could send data to the data collector, the network could easily become inundated with broadcast route discovery messages.

Many-to-one routing is an optimization for these kinds of networks. Rather than require each device to do its own route discovery, a single many-to-one broadcast transmission is sent from the data collector to establish reverse routes on all devices. This is shown in the figure below. The left side shows the many broadcasts the devices can send when they create their own routes. The right side shows the benefits of many-to-one routing where a single broadcast creates reverse routes on all routers.



The many-to-one broadcast is a route request message with the target discovery address set to the address of the data collector. Devices that receive this route request create a reverse many-to-one routing table entry to create a path back to the coordinator. The ZigBee stack on a device uses historical link quality information about each neighbor to select a reliable neighbor for the reverse route.

When a device sends data to a data collector, it finds a many-to-one routing table entry and transmits the data - no route discovery is required on the remotes. The many-to-one route request should be sent periodically to update and refresh the reverse routes in the network.

Applications that require multiple data collectors can also use many-to-one routing. If more than one data collector device sends a many-to-one broadcast, devices will create one reverse routing table entry for each collector.

In ZB firmware, the AR command is used to enable many-to-one broadcasting on a device. The AR command sets a time interval (measured in 10 second units) for sending the many to one broadcast transmission. (See the command table for details.)

Source Routing

In applications where a device must transmit data to many remotes, AODV routing would require performing one route discovery for each destination device to establish a route. Also, if there are more destination devices than there are routing table entries, established AODV routes would be overwritten with new routes, causing route discoveries to occur more regularly. This could result in larger packet delays and poor network performance.

ZigBee source routing helps solve these problems. In contrast to many-to-one routing that establishes routing paths from many devices to one data collector, source routing allows the collector to store and specify routes for many remotes.

Acquiring Source Routes

Source routing should be used in conjunction with many-to-one routing to obtain optimal routes. (See "Many-to-One Routing" section.)

When a remote device sends a transmission using a many-to-one route, it first sends a ZigBee Route Record command frame. The Route Record is sent along the entire many-to-one route until it reaches the data collector. As the Route Record traverses multiple hops, it appends the 16-bit address of each hop into the payload. When the Route Record reaches the data collector, it contains the destination address, and all devices that packet was routed through to reach the collector. The application can save these routes into a source routing table and use the route later to send data to the remote.

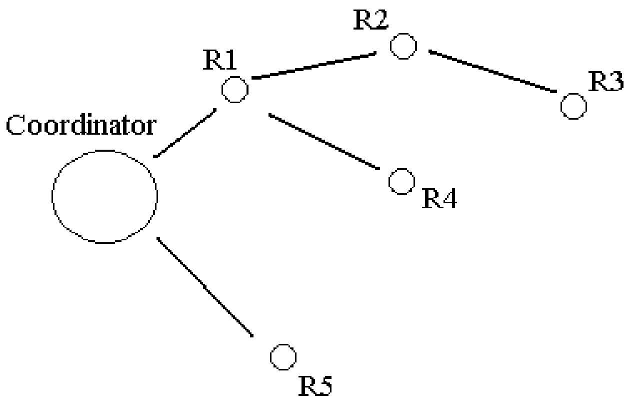
By maintaining source routes to all devices in a network, a device can eliminate the need to perform broadcast route discoveries to establish a route with each device in the network. This feature greatly expands the scalability of ZigBee networks.

To implement source routing in ZB firmware, the AR command must be set less than 0xFF on a data collector device (the device implementing source routing) to send periodic many-to-one route request transmissions. Upon receipt of the many-to-one route request transmissions, remote devices will transmit a route record command before each unicast transmission to the data collector. Upon receipt of a route record command, the Route Record Indicator API frame (0xA1) is sent out the UART of the data collector, indicating a new source route was received.

Note: API firmware should be used to support source routing.

To send a source routed transmission, the application should send a Create Source Route API frame (0x21) to the XBee to create a source route in its internal source route table. After sending the Create Source Route API frame, the application can send data transmission or remote command request frames as needed to the same destination, or any destination in the source route. Once data must be sent to a new destination (a destination not included in the last source route), the application should first send a new Create Source Route API frame. The XBee can buffer one source route that includes up to 10 hops (excluding source and destination).

For example, suppose a network exists with a coordinator and 5 routers (R1, R2, R3, R4, R5) with known source routes as shown below.



To send a source-routed packet to R3, the application must send a Create Source Route API frame (0x21) to the XBee, with a destination of R3, and 2 hops (R1 and R2). If the 64- bit address of R3 is 0x0013A200 404a1234 and the 16-bit addresses of R1, R2, and R3 are:

Device	16-bit address
R1	0xAABB
R2	0xCCDD
R3	0xEEFF

Then the Create Source Route API frame would be:

7E 0012 21 00 0013A200 404A1234 EEFF 00 02 CCDD AABB 5C

Where:

0x0012 - length

0x21 - API ID (create source route)

0x00 - frame ID (set to 0 always)

0x0013A200 404A1234 - 64-bit address of R3 (destination)

0xEEFF - 16-bit address of R3 (destination)

0x00 - Route options (set to 0)

0x02 - Number of intermediate devices in the source route

0xCCDD - Address of furthest device (1-hop from target)

0xAABB - Address of next-closer device

0x5C - Checksum (0xFF - SUM(all bytes after length))

After sending the Create Source Route frame, data can be sent to R1, R2, or R3 and the same source route will be used. However, if data must be sent to R4 or R5, a new Create Source Route frame should be sent to the XBee prior to the transmissions.

Retries and Acknowledgments

ZigBee includes acknowledgment packets at both the Mac and Application Support (APS) layers. When data is transmitted to remote device, it may traverse multiple hops to reach the destination. As data is transmitted from one node to its neighbor, an acknowledgment packet (Ack) is transmitted in the opposite direction to indicate that the transmission was successfully received. If the Ack is not received, the transmitting device will retransmit the data, up to 4 times. This Ack is called the Mac layer acknowledgment.

In addition, the device that originated the transmission expects to receive an acknowledgment packet (Ack) from the destination device. This Ack will traverse the same path that the data traversed, but in the opposite direction. If the originator fails to receive this Ack, it will retransmit the data, up to 2 times until an Ack is received. This Ack is called the ZigBee APS layer acknowledgment.

Refer to the ZigBee specification for more details.

Encrypted Transmissions

Encrypted transmissions are routed similar to non-encrypted transmissions with one exception. As an encrypted packet propagates from one device to another, each device decrypts the packet using the network key, and authenticates the packet by verifying packet integrity. It then re-encrypts the packet with its own source address and frame counter values, and sends the message to the next hop. This process adds some overhead latency to unicast transmissions, but it helps prevent replay attacks. See chapter 5 for details.

Improving Routing Efficiency with the API

If a network has one or more devices that need to transmit to more than 10 different devices, these applications can benefit by implementing an address table or source routing table and interacting with the API firmware. The benefits of these features are described below.

Address Table

Devices that transmit data to more that 10 remotes can see significant routing improvements by supporting an address table that maps a 64-bit address to a 16-bit address, and by using the API to send data. An example address table was shown previously in table xyz. Maintaining an address table significantly reduces the number of 16-bit address discoveries that must take place to route data to the appropriate 16-bit address.

If an application will support an address table, the size should ideally be larger than the maximum number of destination addresses the device will communicate with. Each entry in the address table should contain a 64-bit destination address and its last known 16-bit address.

When sending a transmission to a destination 64-bit address, the application should search the address table for a matching 64-bit address. If a match is found, the 16-bit address should be populated into the 16-bit address field of the API frame. If a match is not found, the 16-bit address should be set to 0xFFFE (unknown) in the API transmit frame.

The API provides indication of a remote device's 16-bit address in the following frames:

- All receive data frames
  Rx Data (0x90)
  Rx Explicit Data (0x91)
  IO Sample Data (0x92)
  Node Identification Indicator (0x95)
  Etc.
- Transmit status frame (0x8B)

The application should always update the 16-bit address in the address table when one of these frames is received to ensure the table has the most recently known 16-bit address. If a transmission failure occurs, the application should set the 16-bit address in the table to 0xFFFE (unknown).

Maximum RF Payload Size

XBee ZB firmware includes a command (ATNP) that returns the maximum number of RF payload bytes that can be sent in a unicast transmission. Querying the NP command, like most other commands, returns a HEXADECIMAL value. This number will change based on whether security is enabled or not. If security is enabled (EE command), the maximum number of RF payload bytes decreases since security requires added overhead.

Broadcast transmissions can support 8 bytes more than unicast transmissions.

If source routing is used, the addresses in the source route must fit into the RF payload space. For example, if NP returns 84 bytes, and a source route must traverse 3 intermediate hops (3 16-bit addresses), the total number of bytes that can be sent in one RF packet is 78.

Management Transmissions

ZigBee defines a ZigBee Device Objects layer (ZDO) that can provide device and service discovery and network management capabilities. This layer is described below.

ZigBee Device Objects (ZDO)

The ZigBee Device Objects (ZDO) is supported to some extent on all ZigBee devices. The ZDO is an endpoint that implements services described in the ZigBee Device Profile in the ZigBee specification. Each service has an assigned cluster ID, and most service requests have an associated response. The following table describes some common ZDO services.

Cluster Name	Cluster ID	Description
Network Address Request	0x0000	Request a 16-bit address of the radio with a matching 64-bit address (required parameter).
Active Endpoints Request	0x0005	Request a list of endpoints from a remote device.
LQI Request	0x0031	Request data from a neighbor table of a remote device.
Routing Table Request	0x0032	Request to retrieve routing table entries from a remote device.
Network Address Response	0x8000	Response that includes the 16-bit address of a device.
LQI Response	0x8031	Response that includes neighbor table data from a remote device.
Routing Table Response	0x8032	Response that includes routing table entry data from a remote device.

Refer to the ZigBee specification for a detailed description of all ZigBee Device Profile services.

Sending a ZDO Command

To send a ZDO command, an explicit transmit API frame must be used and formatted correctly. The source and destination endpoints must be set to 0, and the profile ID must be set to 0. The cluster ID must be set to match the cluster ID of the appropriate service. For example, to send an active endpoints request, the cluster ID must be set to 0x0005.

The first byte of payload in the API frame is an application sequence number (transaction sequence number) that can be set to any single byte value. This same value will be used in the first byte of the ZDO response. All remaining payload bytes must be set as required by the ZDO. All multi-byte values must be sent in little endian byte order.

Receiving ZDO Commands and Responses

In XBee ZB firmware, ZDO commands can easily be sent using the API. In order to receive incoming ZDO commands, receiver application addressing must be enabled with the AO command. (See examples later in this section.) Not all incoming ZDO commands are passed up to the application.

When a ZDO message is received on endpoint 0 and profile ID 0, the cluster ID indicates the type of ZDO message that was received. The first byte of payload is generally a sequence number that corresponds to a sequence number of a request. The remaining bytes are set as defined by the ZDO. Similar to a ZDO request, all multi-byte values in the response are in little endian byte order.

Example 1: Send a ZDO LQI Request to read the neighbor table contents of a remote.

Looking at the ZigBee specification, the cluster ID for an LQI Request is 0x0031, and the payload only requires a single byte (start index). This example will send a LQI request to a remote device with a 64-bit address of 0x0013A200 40401234. The start index will be set to 0, and the transaction sequence number will be set to 0x76

API Frame:

7E 0016 11 01 0013A200 40401234 FFFE 00 00 0031 0000 00 00 76 00 CE

0x0016 - length

0x11 - Explicit transmit request

0x01 - frame ID (set to a non-zero value to enable the transmit status message, or set to 0 to disable)

0x0013A200 40401234 - 64-bit address of the remote

0xFFFE - 16-bit address of the remote (0xFFFE = unknown). Optionally, set to the 16-bit address of the destination if known.

0x00 - Source endpoint

0x00 - Destination endpoint

0x0031 - Cluster ID (LQI Request, or Neighbor table request)

0x0000 - Profile ID (ZigBee Device Profile)

0x00 - Broadcast radius

0x00 - Tx Options

0x76 - Transaction sequence number

0x00 - Required payload for LQI request command

0xCE - Checksum (0xFF - SUM(all bytes after length))

Description:

This API frame sends a ZDO LQI request (neighbor table request) to a remote device to obtain data from its neighbor table. Recall that the AO command must be set correctly on an API device to enable the explicit API receive frames in order to receive the ZDO response.

Example 2: Send a ZDO Network Address Request to discover the 16-bit address of a remote.

Looking at the ZigBee specification, the cluster ID for an Network Address Request is 0x0000, and the payload only requires the following:

[64-bit address] + [Request Type] + [Start Index]

This example will send a Network Address Request as a broadcast transmission to discover the 16-bit address of the device with a 64-bit address of 0x0013A200 40401234. The request type and start index will be set to 0, and the transaction sequence number will be set to 0x44

API Frame:

7E 001F 11 01 00000000 0000FFFF FFFE 00 00 0000 0000 00 00 44 34124040 00A21300 00 00 33

0x001F - length

0x11 - Explicit transmit request

0x01 - frame ID (set to a non-zero value to enable the transmit status message, or set to 0 to disable)

0x00000000 0000FFFF - 64-bit address for a broadcast transmission

0xFFFE - Set to this value for a broadcast transmission.

0x00 - Source endpoint

0x00 - Destination endpoint

0x0000 - Cluster ID (Network Address Request)

0x0000 - Profile ID (ZigBee Device Profile)

0x00 - Broadcast radius

0x00 - Tx Options

0x44 - Transaction sequence number

0x34124040 00A21300 00 00 - Required payload for Network Address Request command

0x33 - Checksum (0xFF - SUM(all bytes after length))

Description:

This API frame sends a broadcast ZDO Network Address Request to obtain the 16-bit address of a device with a 64-bit address of 0x0013A200 40401234. Note the bytes for the 64-bit address were inserted in little endian byte order. All multi-byte fields in the API payload of a ZDO command must have their data inserted in little endian byte order. Also recall that the AO command must be set correctly on an API device to enable the explicit API receive frames in order to receive the ZDO response.

Transmission Timeouts

The transmission timeout varies depending on the nature of the transmission. Timeouts are provided for the following transmission types:

- Transmitting to remote router or coordinator
- Transmitting to child end device
- Transmitting to remote end device.

Note: The timeouts in this section are theoretical timeouts and not precisely accurate. The application should pad the calculated maximum timeouts by a few hundred milliseconds.

Transmitting to Remote Router or Coordinator

The timeout when transmitting to a remote router or coordinator is settable with the NH command. The actual unicast timeout is computed as ((50 * NH) + 100). The default NH value is 30 which equates to a 1.6 second timeout.

The worst case transmission timeout to a remote router or coordinator includes 3 transmission attempts (1 attempt and 2 retries). The maximum total timeout to a remote router or coordinator is about:

3 * ((50 * NH) + 100).

For example, if NH=30 (0x1E), the total transmission to a remote router or coordinator is about

3 * ((50 * 30) + 100), or

3 * (1500 + 100), or

3 * (1600), or

4800 ms, or

4.8 seconds

Transmitting to Child End Device

When sending data to a child end device, a router or coordinator buffers data for a time based on the SP setting. The XBee ZB firmware buffers data for (1.2 * SP) to ensure data is buffered slightly longer than the SP time on an end device. (See "Managing End Devices" chapter for details.) The minimum packet buffer time is 1.2 seconds.

The worst case timeout would occur if the parent believes the end device is a child, but the end device is powered off or otherwise no longer within range of the parent. In this case, the worst case transmission timeout is about the same as when transmitting to a remote end device. (See section below.)

It is also worth noting that a number of provisions exist to help a parent router or coordinator remove stale children from its child table. In ZB firmware, the parent has a poll timeout that will expire an end device child from its child table if it has not received a poll request from the end device child within a certain timeout. Alternatively, if the end device joins a new parent, it will send a broadcast transmission to alert all devices that it has joined a new parent. This mechanism also allows the former parent to remove the end device from its child table.

Transmitting to Remote End Device

The worst case transmission to a remote end device occurs when a previously known end device has left or moved to a different parent, without the sender knowing about it. In this case, the transmission timeout would is defined below.

A transmission to a remote end device is based on the XBee SP parameter. (SP determines the sleep period on XBee end devices. See "Managing End Devices" chapter for details.) A device sending data to a remote end device waits for the unicast timeout, plus the sleep period, before failing the transmission attempt. This is calculated as (50 * NH) + SP. Similar to the remote router or coordinator case, two transmission retries are also expected. Taking retries into account, the worst case transmission timeout is about 3 * ((50 * NH) + (1.2 * SP)).

For example, suppose a router is configured with NH=30 (0x1E) and SP=1000 (0x3E8), and that it is trying to communicate with a remote end device that sleeps for 10 seconds. (Since SP is measured in 10ms units, a value of 1000 is 10 seconds.) The total transmission timeout to the remote end device would be:

3 * ((50 * NH) + (1.2 * SP)), or

3 * (1500 + (1.2 * SP)), or

3 * (1500 + 12,000)

3 * (13500), or

40,500ms, or

40.5 seconds.

Transmission Examples

Example 1: Send a unicast API data transmission to the coordinator using 64-bit address 0, with payload "TxData".

API Frame:

7E 0014 10 01 00000000 00000000 FFFE 00 00 54 78 44 61 74 61 AB

Field Composition:

0x0014 - length

0x10 - API ID (tx data)

0x01 - frame ID (set greater than 0 to enable the tx-status response)

0x00000000 00000000 - 64-bit address of coordinator (ZB definition)

0xFFFE - Required 16-bit address if sending data to 64-bit address of 0.

0x00 - Broadcast radius (0 = max hops)

0x00 - Tx options

0x54 78 44 61 74 61 - ASCII representation of "TxData" string

0xAB - Checksum (0xFF - SUM(all bytes after length))

Description:

This transmission sends the string "TxData" to the coordinator, without knowing the coordinator device's 64-bit address. A 64-bit address of 0 is defined as the coordinator in ZB firmware. If the coordinator's 64-bit address was known, the 64-bit address of 0 could be replaced with the coordinator's 64-bit address, and the 16-bit address could be set to 0.

Example 2 - Send a broadcast API data transmission that all devices can receive (including sleepy end devices), with payload "TxData".

API Frame:

7E 0014 10 01 00000000 0000FFFF FFFE 00 00 54 78 44 61 74 61 AD

Field Composition:

0x0014 - length

0x10 - API ID (tx data)

0x01 - frame ID (set to a non-zero value to enable the tx-status response)

0x00000000 0000FFFF - Broadcast definition (including sleeping end devices

0xFFFE - Required 16-bit address to send broadcast transmission.

0x00 - Broadcast radius (0 = max hops)

0x00 - Tx options

0x54 78 44 61 74 61 - ASCII representation of "TxData" string

0xAD - Checksum (0xFF - SUM(all bytes after length))

Description:

This transmission sends the string "TxData" as a broadcast transmission. Since the destination address is set to 0xFFFF, all devices, including sleeping end devices can receive this broadcast.

If receiver application addressing is enabled, the XBee will report all received data frames in the explicit format (0x91) to indicate the source and destination endpoints, cluster ID, and profile ID that each packet was received on. (Status messages like modem status and route record indicators are not affected.)

To enable receiver application addressing, set the AO command to 1 using the AT command frame (0x08).

API Frame:

7E 0005 08 01 414F 01 65

Field Composition:

0x0005 - length

0x08 - API ID (at command)

0x01 - frame ID (set to a non-zero value to enable AT command response frames)

0x414F - ASCII representation of 'A','O' (the command being issued)

0x01 - Parameter value

0x65 - Checksum (0xFF - SUM(all bytes after length))

Description:

Setting AO=1 is required for the XBee to use the explicit receive API frame (0x91) when RF data packets are received. This is required if the application needs indication of source or destination endpoint, cluster ID, and/or profile ID values used in received ZigBee data packets. ZDO messages can only be received if AO=1.

5. Security

ZigBee supports various levels of security that can be configured depending on the needs of the application. Security provisions include:

- 128-bit AES encryption
- Two security keys that can be preconfigured or obtained during joining
- Support for a trust center
- Provisions to ensure message integrity, confidentiality, and authentication.

The first half of this chapter describes various security features defined in the ZigBee-PRO specification, while the last half illustrates how the XBee and XBee-PRO modules can be configured to support these features

Security Modes

The ZigBee standard supports three security modes – residential, standard, and high security. Residential security was first supported in the ZigBee 2006 standard. This level of security requires a network key be shared among devices. Standard security adds a number of optional security enhancements over residential security, including an APS layer link key. High security adds entity authentication, and a number of other features not widely supported.

XBee ZB modules primarily support standard security, although end devices that support residential security can join and interoperate with standard security devices. The remainder of this chapter focuses on material that is relevant to standard security.

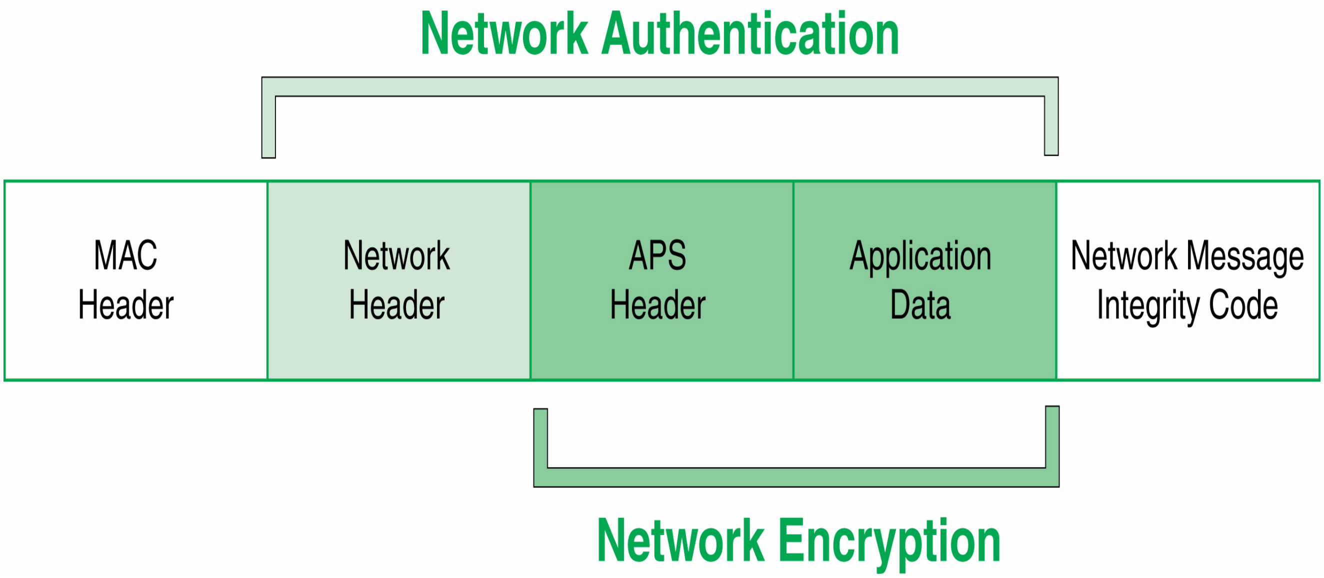
ZigBee Security Model

ZigBee security is applied to the Network and APS layers. Packets are encrypted with 128-bit AES encryption. A network key and optional link key can be used to encrypt data. Only devices with the same keys are able to communicate together in a network. Routers and end devices that will communicate on a secure network must obtain the correct security keys.

Network Layer Security

The network key is used to encrypt the APS layer and application data. In addition to encrypting application messages, network security is also applied to route request and reply messages, APS commands, and ZDO commands. Network encryption is not applied to MAC layer transmissions such as beacon transmissions, etc. If security is enabled in a network, all data packets will be encrypted with the network key.

Packets are encrypted and authenticated using 128-bit AES. This is shown in the figure below.



Frame Counter

The network header of encrypted packets includes a 32-bit frame counter. Each device in the network maintains a 32-bit frame counter that is incremented for every transmission. In addition, devices track the last known 32-bit frame counter for each of its neighbors. If a device receives a packet from a neighbor with a smaller frame counter than it has previously seen, the packet is discarded. The frame counter is used to protect against replay attacks.

If the frame counter reaches a maximum value of 0xFFFFFFFF, it does not wrap to 0 and no more transmissions can be sent. Due to the size of the frame counters, reaching the maximum value is a very unlikely event for most applications. The following table shows the required time under different conditions, for the frame counter to reach its maximum value.

Average Transmission Rate	Time until 32-bit frame counter expires
1 / second	136 years
10 / second	13.6 years

To clear the frame counters without compromising security, the network key can be changed in the network. When the network key is updated, the frame counters on all devices reset to 0. (See the Network Key Updates section for details.)

Message Integrity Code

The network header, APS header, and application data are all authenticated with AES-128. A hash is performed on these fields and is appended as a 4-byte message integrity code (MIC) to the end of the packet. The MIC allows receiving devices to ensure the message has not been changed. The MIC provides message integrity in the ZigBee security model. If a device receives a packet and the MIC does not match the device’s own hash of the data, the packet is dropped.

Network Layer Encryption and Decryption

Packets with network layer encryption are encrypted and decrypted by each hop in a route. When a device receives a packet with network encryption, it decrypts the packet and authenticates the packet. If the device is not the destination, it then encrypts and authenticates the packet, using its own frame counter and source address in the network header section.

Since network encryption is performed at each hop, packet latency is slightly longer in an encrypted network than in a non-encrypted network. Also, security requires 18 bytes of overhead to include a 32-bit frame counter, an 8-byte source address, 4-byte MIC, and 2 other bytes. This reduces the number of payload bytes that can be sent in a data packet.

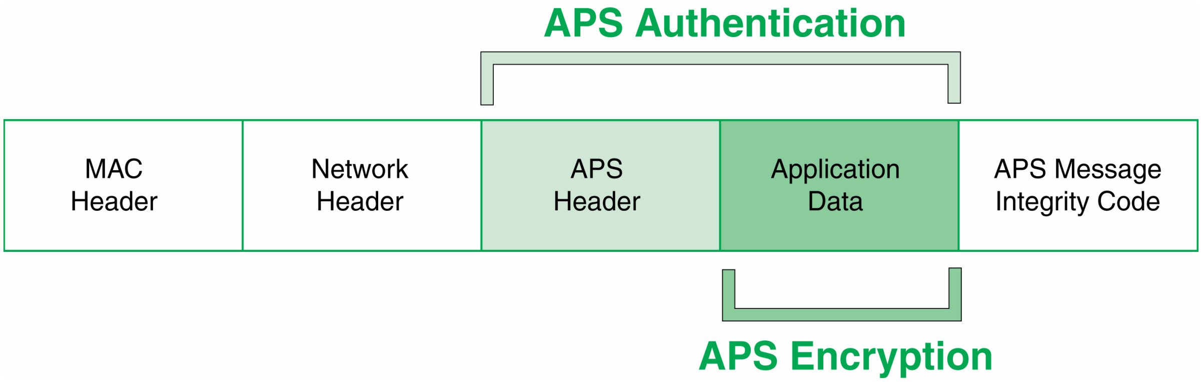
Network Key Updates

ZigBee supports a mechanism for changing the network key in a network. When the network key is changed, the frame counters in all devices reset to 0.

APS Layer Security

APS layer security can be used to encrypt application data using a key that is shared between source and destination devices. Where network layer security is applied to all data transmissions and is decrypted and re-encrypted on a hop-by-hop basis, APS security is optional and provides end-to-end security using an APS link key that only the source and destination device know. APS security can be applied on a packet-by-packet basis. APS security cannot be applied to broadcast transmissions.

If APS security is enabled, packets are encrypted and authenticated using 128-bit AES. This is shown in the figure below:



Message integrity Code

If APS security is enabled, the APS header and data payload are authenticated with AES-128. A hash is performed on these fields and appended as a 4-byte message integrity code (MIC) to the end of the packet. This MIC is different than the MIC appended by the network layer. The MIC allows the destination device to ensure the message has not been changed. If the destination device receives a packet and the MIC does not match the destination device's own hash of the data, the packet is dropped.

APS Link Keys

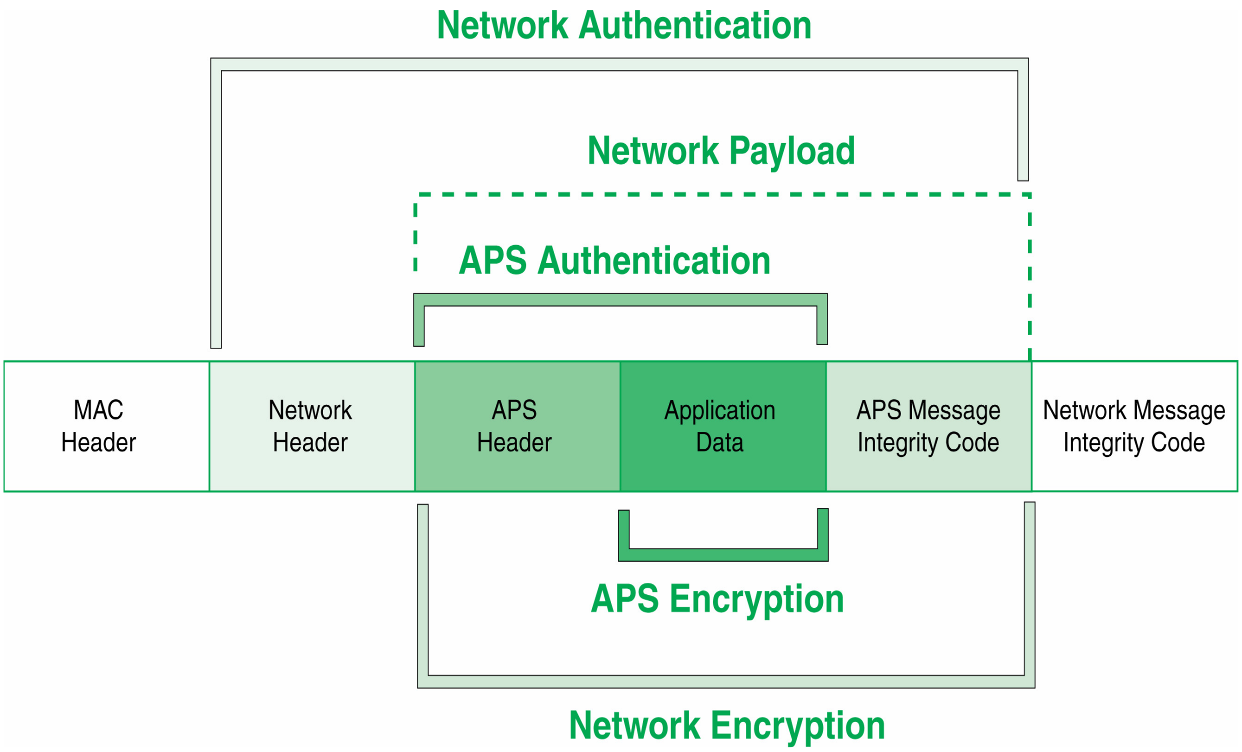
There are two kinds of APS link keys – trust center link keys and application link keys. A trust center link key is established between a device and the trust center, where an application link key is established between a device and another device in the network where neither device is the trust center.

APS Layer Encryption and Decryption

Packets with APS layer encryption are encrypted at the source and only decrypted by the destination. Since APS encryption appends a 4-byte MIC, the maximum data payload is reduced by 4 bytes when APS encryption is used

Network and APS Layer Encryption

Network and APS layer encryption can both be applied to data. The following figure demonstrates the authentication and encryption performed on the final ZigBee packet when both are applied.



Trust Center

ZigBee defines a trust center device that is responsible for authenticating devices that join the network. The trust center also manages link key distribution in the network.

Forming and Joining a Secure Network

The coordinator is responsible for selecting a network encryption key. This key can either be preconfigured or randomly selected. In addition, the coordinator generally operates as a trust center and must therefore select the trust center link key. The trust center link key can also be preconfigured or randomly selected.

Devices that join the network must obtain the network key when they join. When a device joins a secure network, the network and link keys can be sent to the joining device. If the joining device has a pre-configured trust center link key, the network key will be sent to the joining device encrypted by the link key. Otherwise, if the joining device is not pre-configured with the link key, the device could only join the network if the network key is sent unencrypted ("in the clear"). The trust center must decide whether or not to send the network key unencrypted to joining devices that are not pre-configured with the link key. Sending the network key unencrypted is not recommended as it can open a security hole in the network. To maximize security, devices should be pre-configured with the correct link key.

Implementing Security on the XBee

If security is enabled in the XBee ZB firmware, devices acquire the network key when they join a network. Data transmissions are always encrypted with the network key, and can optionally be end-to-end encrypted with the APS link key. The following sections discuss the security settings and options in the XBee ZB firmware.

Enabling Security

To enable security on a device, the EE command must be set to 1. If the EE command value is changed and changes are applied (i.e. AC command), the XBee module will leave the network (PAN ID and channel) it was operating on, and attempt to form or join a new network.

Note: The EE command must be set the same on all devices in a network. Changes to the EE command should be written to non-volatile memory (to be preserved through power cycle or reset events) using the WR command.

Setting the Network Security Key

The coordinator must select the network security key for the network. The NK command (write-only) is used to set the network key. If NK=0 (default), a random network key will be selected. (This should suffice for most applications.) Otherwise, if NK is set to a non-zero value, the network security key will use the value specified by NK. NK is only supported on the coordinator.

Routers and end devices with security enabled (ATEE=1)acquire the network key when they join a network. They will receive the network key encrypted with the link key if they share a pre-configured link key with the coordinator. See the following section for details.

Setting the APS Trust Center Link Key

The coordinator must also select the trust center link key, using the KY command. If KY=0 (default), the coordinator will select a random trust center link key (not recommended). Otherwise, if KY is set greater than 0, this value will be used as the pre-configured trust center link key. KY is write-only and cannot be read.

Note: Application link keys (sent between two devices where neither device is the coordinator) are not supported in ZB firmware at this time.

Random Trust Center Link Keys

If the coordinator selects a random trust center link key (KY=0, default), then it will allow devices to join the network without having a pre-configured link key. However, this will cause the network key to be sent unencrypted over-the-air to joining devices and is not recommended.

Pre-configured Trust Center Link Keys

If the coordinator uses a pre-configured link key (KY > 0), then the coordinator will not send the network key unencrypted to joining devices. Only devices with the correct pre-configured link key will be able to join and communicate on the network.

Using a Trust Center

The EO command can be used to define the coordinator as a trust center. If the coordinator is a trust center, it will be alerted to all new join attempts in the network. The trust center also has the ability to update or change the network key on the network.

In ZB firmware, a secure network can be established with or without a trust center. Network and APS layer encryption are supported if a trust center is used or not.

Updating the Network Key with a Trust Center

If the trust center has started a network and the NK value is changed, the coordinator will update the network key on all devices in the network. (Changes to NK will not force the device to leave the network.) The network will continue to operate on the same channel and PAN ID, but the devices in the network will update their network key, increment their network key sequence number, and restore their frame counters to 0.

Updating the Network Key without a Trust Center

If the coordinator is not running as a trust center, the network reset command (NR1) can be used to force all devices in the network to leave the current network and rejoin the network on another channel. When devices leave and reform then network, the frame counters are reset to 0. This approach will cause the coordinator to form a new network that the remaining devices should join. Resetting the network in this manner will bring the coordinator and routers in the network down for about 10 seconds, and will likely cause the 16-bit PAN ID and 16-bit addresses of the devices to change.

XBee Security Examples

This section covers some sample XBee configurations to support different security modes. Several AT commands are listed with suggested parameter values. The notation in this section includes an '=' sign to indicate what each command register should be set to - for example, EE=1. This is not the correct notation for setting command values in the XBee. In AT command mode, each command is issued with a leading 'AT' and no '=' sign - for example ATEE1. In the API, the two byte command is used in the command field, and parameters are populated as binary values in the parameter field.

Example 1: Forming a network with security (pre-configured link keys)

1. Start a coordinator with the following settings:

   a. ID=2234 (arbitrarily selected)

   b. EE=1

   c. NK=0

   d. KY=4455

   e. WR (save networking parameters to preserve them through power cycle)

2. Configure one or more routers or end devices with the following settings:

   a. ID=2234

   b. EE=1

   c. KY=4455

   d. WR (save networking parameters to preserve them through power cycle)

3. Read the AI setting on the coordinator and joining devices until they return 0 (formed or joined a network).

In this example, EE, ID, and KY are set the same on all devices. After successfully joining the secure network, all application data transmissions will be encrypted by the network key. Since NK was set to 0 on the coordinator, a random network key was selected. And since the link key (KY) was configured the same on all devices, to a non-zero value, the network key was sent encrypted by the pre-configured link key (KY) when the devices joined.

Example 2: Forming a network with security (obtaining keys during joining)

1. Start a coordinator with the following settings:

   a. ID=2235

   b. EE=1

   c. NK=0

d. KY=0

e. WR (save networking parameters to preserve them through power cycle)

2. Configure one or more routers or end devices with the following settings:

a. ID=2235

b. EE=1

c. KY=0

d. WR (save networking parameters to preserve them through power cycle)

3. Read the AI setting on the coordinator and joining devices until they return 0 (formed or joined a network).

In this example, EE, ID, and KY are set the same on all devices. Since NK was set to 0 on the coordinator, a random network key was selected. And since KY was set to 0 on all devices, the network key was sent unencrypted ("in the clear") when the devices joined. This approach introduces a security vulnerability into the network and is not recommended.

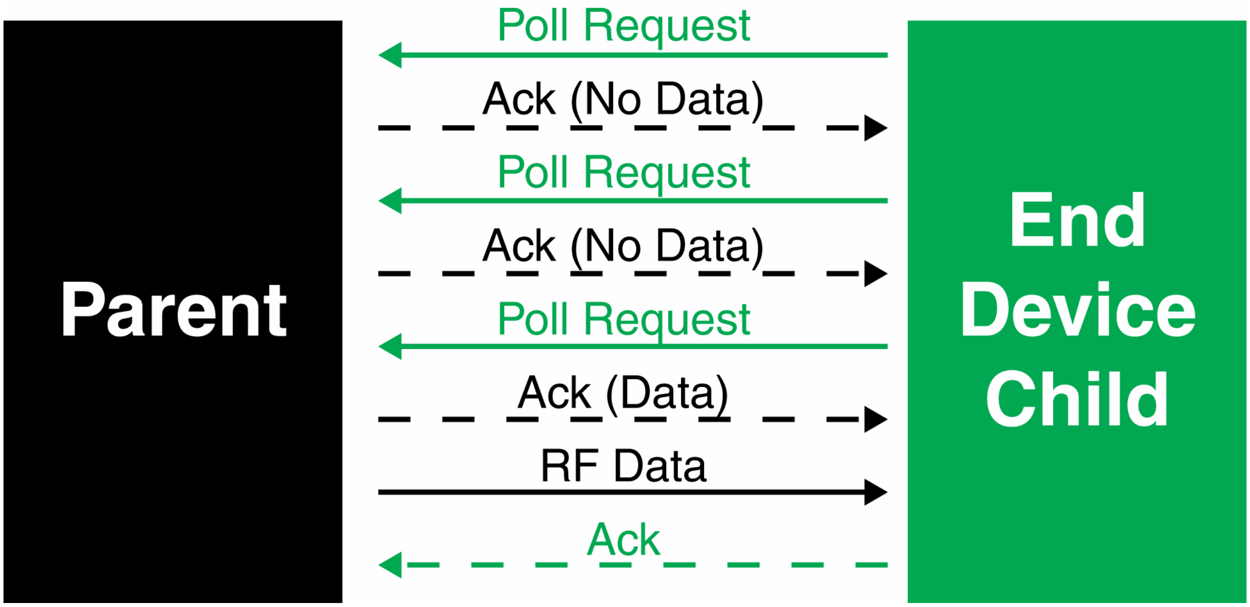
6. Managing End Devices

ZigBee end devices are intended to be battery-powered devices capable of sleeping for extended periods of time. Since end devices may not be awake to receive RF data at a given time, routers and coordinators are equipped with additional capabilities (including packet buffering and extended transmission timeouts) to ensure reliable data delivery to end devices.

End Device Operation

When an end device joins a ZigBee network, it must find a router or coordinator device that is allowing end devices to join. Once the end device joins a network, a parent-child relationship is formed between the end device and the router or coordinator that allowed it to join. See chapter 4 for details.

When the end device is awake, it sends poll request messages to its parent. When the parent receives a poll request, it checks a packet queue to see if it has any buffered messages for the end device. It then sends a MAC layer acknowledgment back to the end device that indicates if it has data to send to the end device or not.



If the end device receives the acknowledgment and finds that the parent has no data for it, the end device can return to idle mode or sleep. Otherwise, it will remain awake to receive the data. This polling mechanism allows the end device to enter idle mode and turn its receiver off when RF data is not expected in order to reduce current consumption and conserve battery life.

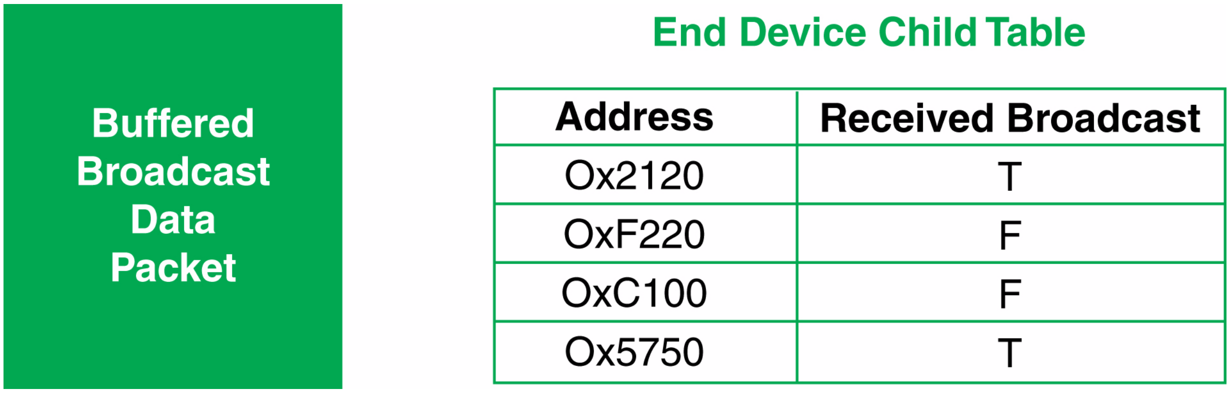
The end device can only send data directly to its parent. If an end device must send a broadcast or a unicast transmission to other devices in the network, it sends the message directly to its parent and the parent performs any necessary route or address discoveries to route the packet to the final destination.

Parent Operation

Each router or coordinator maintains a child table that contains the addresses of its end device children. A router or coordinator that has unused entries in its child table is said to have end device capacity, or the ability to allow new end devices to join. If the child table is completely filled (such that the number of its end device children matches the number of child table entries), the device cannot allow any more end devices to join to it.

Since the end device children are not guaranteed to be awake at a given time, the parent is responsible to manage incoming data packets in behalf of its end device children. If a parent receives an RF data transmission destined for one of its end device children, and if the parent has enough unused buffer space, it will buffer the packet. The data packet will remain buffered until a timeout expires, or until the end device sends a poll request to retrieve the data.

The parent can buffer one broadcast transmission for all of its end device children. When a broadcast transmission is received and buffered, the parent sets a flag in its child table when each child polls and retrieves the packet. Once all children have received the broadcast packet, the buffered broadcast packet is discarded. If all children have not received a buffered broadcast packet and a new broadcast is received, the old broadcast packet is discarded, the child table flags are cleared, and the new broadcast packet is buffered for the end device children. This is demonstrated in the figure below.



End Device Child Table

Address	Received Broadcast
Ox2120	T
OxF220	F
OxC100	F
Ox5750	T

When an end device sends data to its parent that is destined for a remote device in the network, the parent buffers the data packet until it can establish a route to the destination. The parent may perform a route or 16-bit address discovery in behalf of its end device children. Once a route is established, the parent sends the data transmission to the remote device.

End Device Poll Timeouts

To better support mobile end devices (end devices that can move around in a network), parent router and coordinator devices have a poll timeout for each end device child. If an end device does not send a poll request to its parent within the poll timeout, the parent will remove the end device from its child table. This allows the child table on a router or coordinator to better accommodate mobile end devices in the network.

Packet Buffer Usage

Packet buffer usage on a router or coordinator varies depending on the application. The following activities can require use of packet buffers for up to several seconds:

- Route and address discoveries
- Application broadcast transmissions
- Stack broadcasts (i.e. ZDO "Device Announce" messages when devices join a network)
- Unicast transmissions (buffered until acknowledgment is received from destination or retries exhausted)
- Unicast messages waiting for end device to wake.

Applications that use regular broadcasting or that require regular address or route discoveries will use up a significant number of buffers, reducing the buffer availability for managing packets for end device children. Applications should reduce the number of required application broadcasts, and consider implementing an external address table or many-to-one and source routing if necessary to improve routing efficiency.

Non-Parent Device Operation

Devices in the ZigBee network treat data transmissions to end devices differently than transmissions to other routers and coordinators. Recall that when a unicast transmission is sent, if a network acknowledgment is not received within a timeout, the device resends the transmission. When transmitting data to remote coordinator or router devices, the transmission timeout is relatively short since these devices are powered and responsive. However, since end devices may sleep for some time, unicast transmissions to end devices use an extended timeout mechanism in order to allow enough time for the end device to wake and receive the data transmission from its parent.

If a non-parent device does not know the destination is an end device, it will use the standard unicast timeout for the transmission. However, provisions exist in the Ember ZigBee stack for the parent to inform the message sender that the destination is an end device. Once the sender discovers the destination device is an end device, future transmissions will use the extended timeout. See the XBee Router / Coordinator Configuration section in this chapter for details.

XBee End Device Configuration

XBee end devices support two different sleep modes:

- Pin Sleep
- Cyclic Sleep.

Pin sleep allows an external microcontroller to determine when the XBee should sleep and when it should wake by controlling the Sleep_RQ pin. In contrast, cyclic sleep allows the sleep period and wake times to be configured through the use of AT commands. The sleep mode is configurable with the SM command.

In both pin and cyclic sleep modes, XBee end devices poll their parent every 100ms while they are awake to retrieve buffered data. When a poll request has been sent, the end device enables the receiver until an acknowledgment is received from the parent. (It generally takes about 10ms from the time the poll request is sent until the acknowledgment is received.) The acknowledgment indicates if the parent has buffered data for the end device child or not. If the acknowledgment indicates the parent has pending data, the end device will leave the receiver on to receive the data. Otherwise, the end device will turn off the receiver and enter idle mode (until the next poll request is sent) to reduce current consumption (and improve battery life).

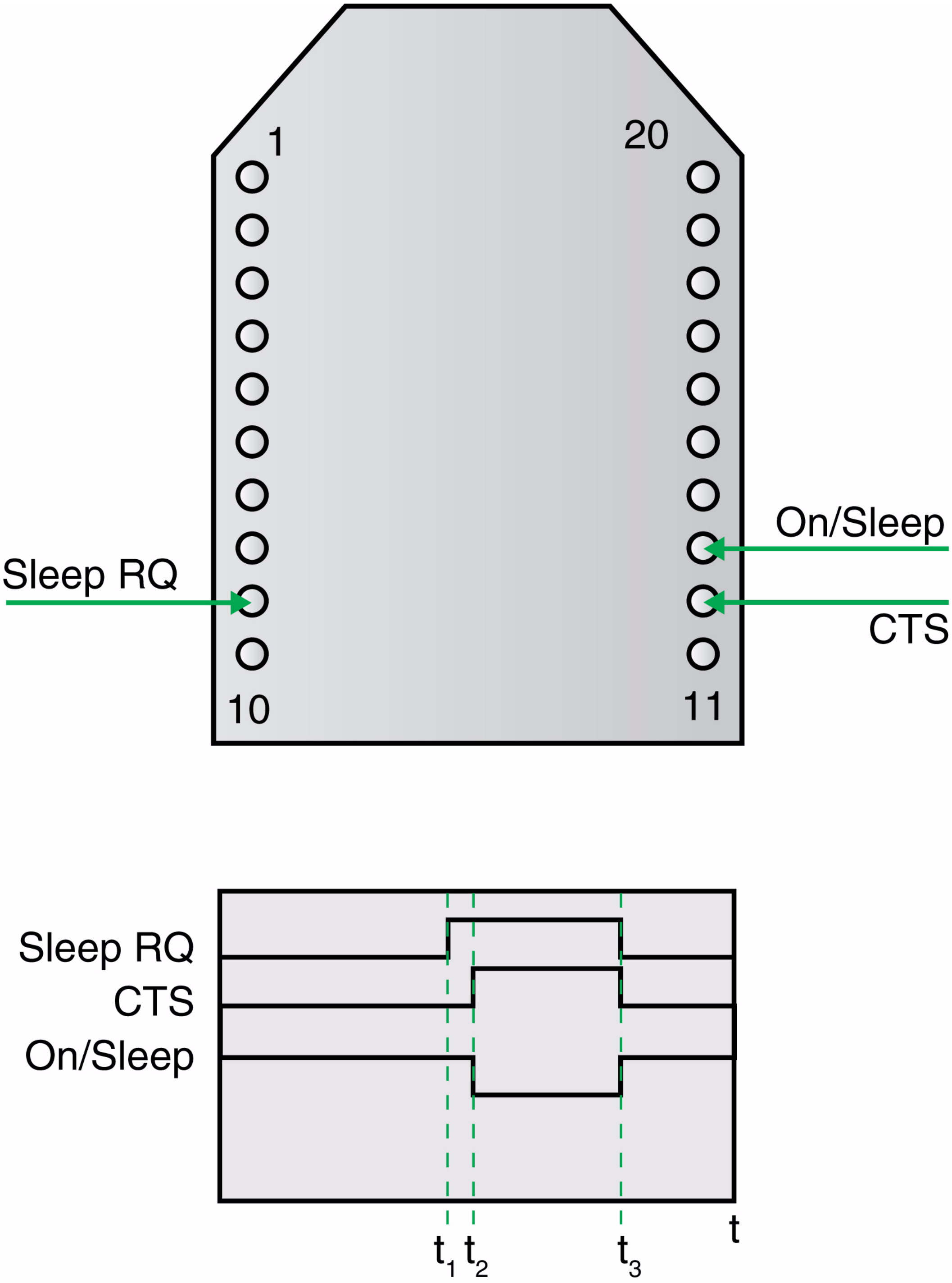
Once the module enters sleep mode, the On/Sleep pin (pin 13) is de-asserted (low) to indicate the module is entering sleep mode. If CTS hardware flow control is enabled (D7 command), the CTS pin (pin 12) is de-asserted (high) when entering sleep to indicate that serial data should not be sent to the module. The module will not respond to serial or RF data when it is sleeping. Applications that must communicate serially to sleeping end devices are encouraged to observe CTS flow control.

When the XBee wakes from sleep, the On/Sleep pin is asserted (high), and if flow control is enabled, the CTS pin is also asserted (low). If the module has not joined a network, it will scan all SC channels after waking to try and find a valid network to join.

Pin Sleep

Pin sleep allows the module to sleep and wake according to the state of the Sleep_RQ pin (pin 9). Pin sleep mode is enabled by setting the SM command to 1.

When Sleep_RQ is asserted (high), the module will finish any transmit or receive operations and enter a low power state. For example, if the module has not joined a network and Sleep_RQ is asserted (high), the module will sleep once the current join attempt completes (i.e. when scanning for a valid network completes). The module will wake from pin sleep when the Sleep_RQ pin is de-asserted (low).



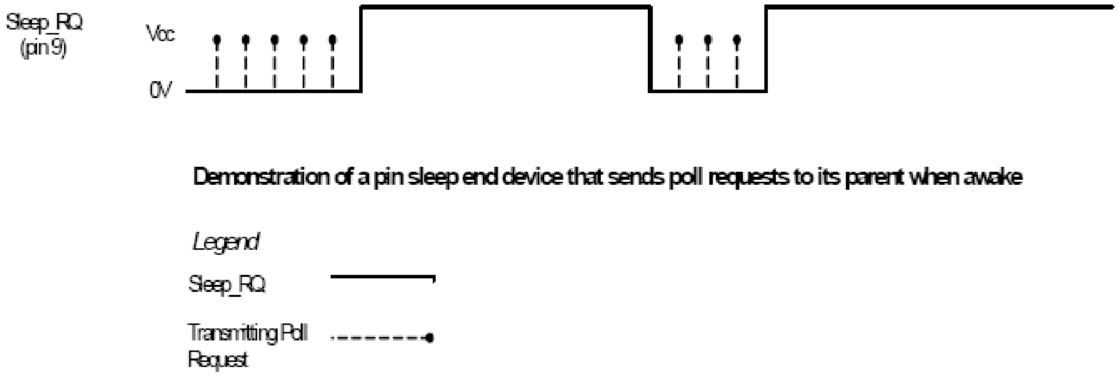
In the figure above, t1, t2, and t3 represent the following events:

- T1 - Time when Sleep_RQ is asserted (high)
- T2 - Time when the XBee enters sleep (CTS state change only if hardware flow control is enabled)
- T3 - Time when Sleep_RQ is de-asserted (low) and the module wakes.

The time between T1 and T2 varies depending on the state of the module. In the worst case scenario, if the end device is trying to join a network, or if it is waiting for an acknowledgment from a data transmission, the delay could be up to a few seconds.

When the XBee is awake and is joined to a network, it sends a poll request to its parent to see if the parent has any buffered data for it. The end device will continue to send poll requests every 100ms while it is awake.

Demonstration of Pin Sleep



Demonstration of a pin sleep end device that sends poll requests to its parent when awake

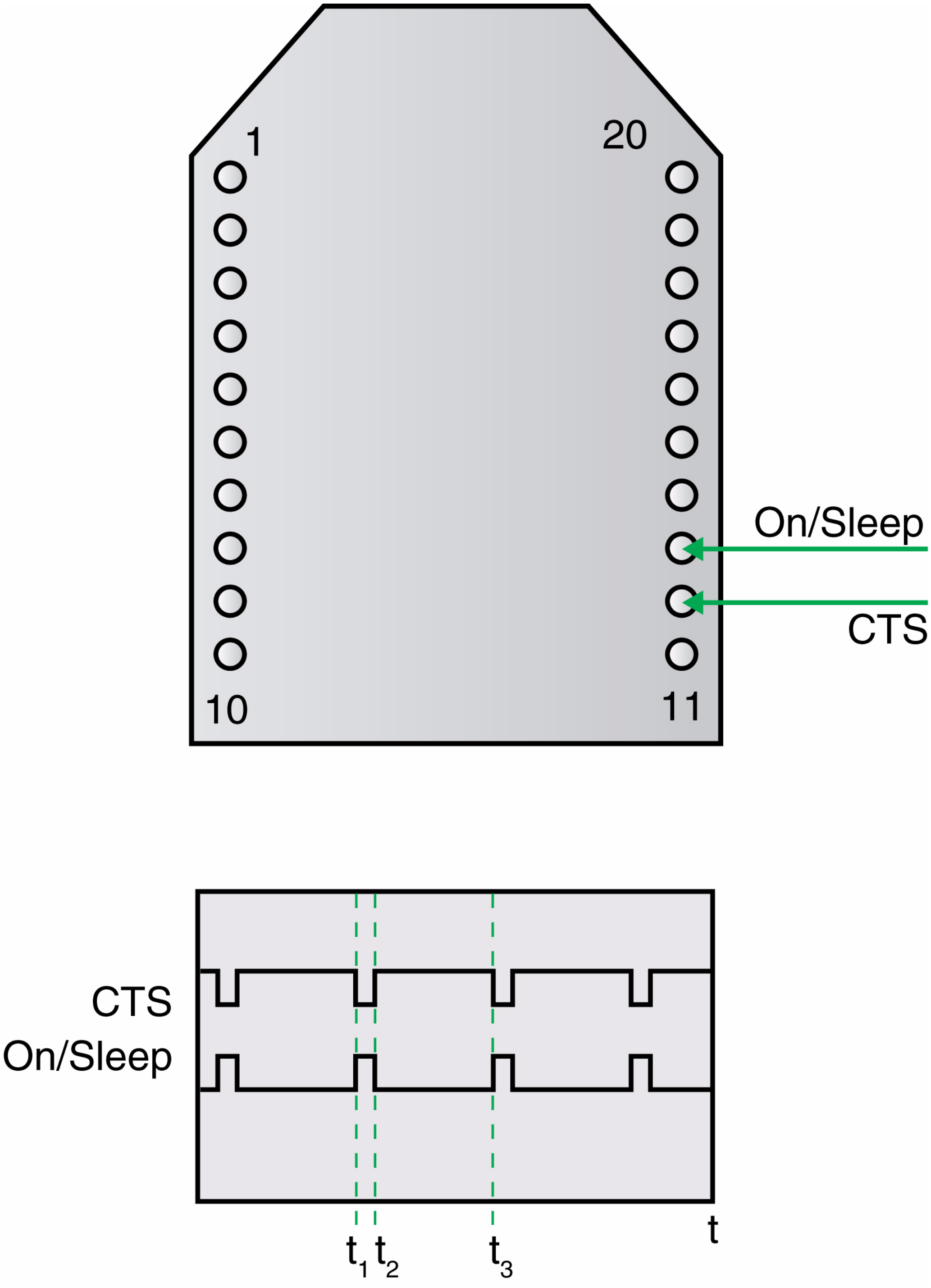
Parent and remote devices must be configured to buffer data correctly and to utilize adequate transmission timeouts. See the XBee Router / Coordinator Configuration section in this chapter for details.

Cyclic Sleep

Cyclic sleep allows the module to sleep for a specified time and wake for a short time to poll its parent for any buffered data messages before returning to sleep again. Cyclic sleep mode is enabled by setting the SM command to 4 or 5. SM5 is a slight variation of SM4 that allows the module to be woken prematurely by asserting the Sleep_RQ pin (pin 9). In SM5, the XBee can wake after the sleep period expires, or if a high-to-low transition occurs on the Sleep_RQ pin. Setting SM to 4 disables the pin wake option.

In cyclic sleep, the module sleeps for a specified time, and then wakes and sends a poll request to its parent to discover if the parent has any pending data for the end device. If the parent has buffered data for the end device, or if serial data is received, the XBee will remain awake for a time. Otherwise, it will enter sleep mode immediately.

The On/Sleep line is asserted (high) when the module wakes, and is de-asserted (low) when the module sleeps. If hardware flow control is enabled (D7 command), the CTS pin will assert (low) when the module wakes and can receive serial data, and de-assert (high) when the module sleeps.



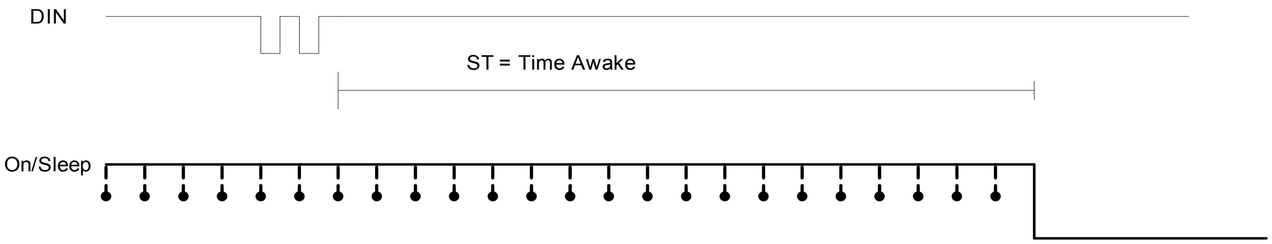
In the figure above, t1, t2, and t3 represent the following events:

- T1 - Time when the module wakes from cyclic sleep
- T2 - Time when the module returns to sleep
- T3 - Later time when the module wakes from cyclic sleep.

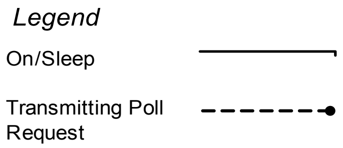
The wake time and sleep time are configurable with software commands as described in the sections below.

Wake Time (Until Sleep)

In cyclic sleep mode (SM=4 or 5), if serial or RF data is received, the module will start a sleep timer (time until sleep). Any data received serially or over the RF link will restart the timer. The sleep timer value is settable with the ST command. While the module is awake, it will send poll request transmissions every 100ms to check its parent for buffered data messages. The module returns to sleep when the sleep timer expires, or if the SI command is sent to it. The following image shows this behavior.



A cyclic sleep end device enters sleep mode when no serial or RF data is received for ST time .



Sleep Period

The sleep period is configured based on the SP, SN, and SO commands. The following table lists the behavior of these commands.

Com-mand	Range	Description
SP	0x20 - 0xAF0 (x 10 ms) (320 - 28,000 ms)	Configures the sleep period of the module.
SN	1 - 0xFFFF	Configures the number of sleep periods multiplier.
SO	0 - 0xFF	Defines options for sleep mode behavior. 0x02 - Always wake for full ST time 0x04 - Enable extended sleep (sleep for full (SP * SN) time)

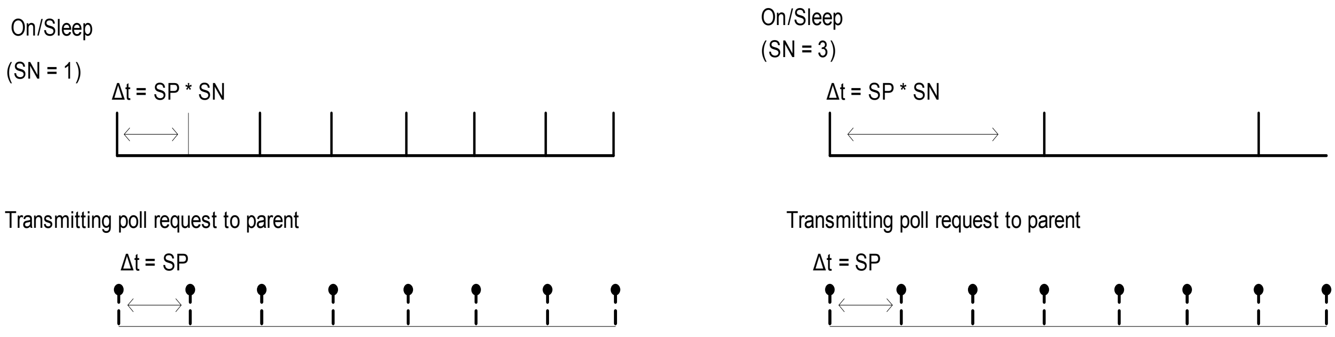
The XBee module supports both a short cyclic sleep and an extended cyclic sleep that make use of these commands.. These two modes allow the sleep period to be configured according to the application requirements.

Short Cyclic Sleep

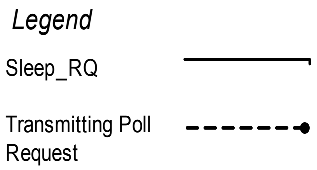
In short cyclic sleep mode, the sleep behavior of the module is defined by the SP and SN commands, and the SO command must be set to 0x00 (default) or 0x02. In short cyclic sleep mode, the SP command defines the sleep period and is settable up to 28 seconds. When the XBee enters short cyclic sleep, it remains in a low power state until the SP time has expired.

After the sleep period expires, the XBee sends a poll request transmission to its parent to determine if its parent has any buffered data waiting for the end device. Since router and coordinator devices can buffer data for end device children up to 30 seconds, the SP range (up to 28 seconds) allows the end device to poll regularly enough to receive buffered data. If the parent has data for the end device, the end device will start its sleep timer (ST) and continue polling every 100ms to receive data. If the end device wakes and finds that its parent has no data for it, the end device can return to sleep immediately.

The SN command can be used to control when the On/Sleep line is asserted (high). If SN is set to 1 (default), the On/Sleep line will be set high each time the XBee wakes from sleep. Otherwise, if SN is greater than 1, the On/Sleep line will only be set high if RF data is received, or after SN wake cycles occur. This allows an external device to remain powered off until RF data is received, or until a number of sleep periods have expired (SN sleep periods). This mechanism allows the XBee to wake at regular intervals to poll its parent for data without waking an external device for an extended time (SP * SN time). This is shown in the figure below.


On/Sleep
(SN = 1)
Δt = SP * SN
Transmitting poll request to parent
Δt = SP
On/Sleep
(SN = 3)
Δt = SP * SN
Transmitting poll request to parent
Δt = SP

Setting SN > 1 allows the XBee to silently poll for data without asserting On /Sleep. If RF data is received when polling, On/Sleep will immediately assert .



NOTE: SP controls the packet buffer time on routers and coordinators. SP should be set on all router and coordinator devices to match the longest end device SP time. See the XBee Router / Coordinator Configuration section for details.

Extended Cyclic Sleep

In extended cyclic sleep operation, an end device can sleep for a multiple of SP time which can extend the sleep time up to several days. The sleep period is configured using the SP and SN commands. The total sleep period is equal to (SP * SN) where SP is measured in 10ms units. The SO command must be set correctly to enable extended sleep.

Since routers and coordinators can only buffer incoming RF data for their end device children for up to 30 seconds, if an end device sleeps longer than 30 seconds, devices in the network need some indication when an end device is awake before they can send data to it. End devices that use extended cyclic sleep should send a transmission (such as an IO sample) when they wake to inform other devices that they are awake and can receive data. It is recommended that extended sleep end devices set SO to wake for the full ST time in order to provide other devices with enough time to send messages to the end device.

Similar to short cyclic sleep, end devices running in this mode will return to sleep when the sleep timer expires, or when the SI command is received.

Transmitting RF Data

An end device may transmit data when it wakes from sleep and has joined a network. End devices transmit directly to their parent and then wait for an acknowledgment to be received. The parent will perform any required address and route discoveries to help ensure the packet reaches the intended destination before reporting the transmission status to the end device.

Receiving RF Data

After waking from sleep, an end device sends a poll request to its parent to determine if the parent has any buffered data for it. In pin sleep mode, the end device polls every 100ms while the Sleep_RQ pin is de-asserted (low). In cyclic sleep mode, the end device will only poll once before returning to sleep unless the sleep timer (ST) is started (serial or RF data is received). If the sleep timer is started, the end device will continue to poll every 100ms until the sleep timer expires.

IO Sampling

End devices can be configured to send one or more IO samples when they wake from sleep. To enable IO sampling on an end device, the IR command must be set to a non-zero value, and at least one analog or digital IO pin must be enabled for sampling (D0 - D9, P0-P2 commands). If IO sampling is enabled, an end device sends an IO sample when it wakes and starts the ST timer. It will continue sampling at the IR rate until the sleep timer (ST) has expired. See chapter 8 for details.

Waking End Devices with the Commissioning Pushbutton

If the commissioning pushbutton functionality is enabled (D0 command), a high-to-low transition on the AD0/DIO0 pin (pin 20) will cause an end device to wake for 30 seconds. See the Commissioning Pushbutton section in chapter 7 for details.

Parent Verification

Since an end device relies on its parent to maintain connectivity with other devices in the network, XBee end devices include provisions to verify its connection with its parent. End devices monitor their link with their parent when sending poll messages and after a power cycle or reset event as described below.

When an end device wakes from sleep, it sends a poll request to its parent. In cyclic sleep, if RF or serial data is not received and the sleep timer is not started, the end device polls one time and returns to sleep for another sleep period. Otherwise, the end device continues polling every 100ms. If the parent does not send an acknowledgment response to three consecutive poll request transmissions, the end device assumes the parent is out of range, and attempts to find a new parent.

After a power-up or reset event, the end device does an orphan scan to locate its parent. If the parent does not send a response to the orphan scan, the end device attempts to find a new parent.

Rejoining

Once all devices have joined a ZigBee network, the permit-joining attribute should be disabled such that new devices are no longer allowed to join the network. Permit-joining can be enabled later as needed for short times. This provides some protection in preventing other devices from joining a live network.

If an end device cannot communicate with its parent, the end device must be able to join a new parent to maintain network connectivity. However, if permit-joining is disabled in the network, the end device will not find a device that is allowing new joins.

To overcome this problem, ZigBee supports rejoining, where an end device can obtain a new parent in the same network even if joining is not enabled. When an end device joins using rejoining, it performs a PAN ID scan to discover nearby networks. If a network is discovered that has the same 64-bit PAN ID as the end device, it will join the network by sending a rejoin request to one of the discovered devices. The device that receives the rejoin request will send a rejoin response if it can allow the device to join the network (i.e. child table not full). The rejoin mechanism can be used to allow a device to join the same network even if permit-joining is disabled.

To enable rejoining, NJ should be set less than 0xFF on the device that will join. If NJ < 0xFF, the device assumes the network is not allowing joining and first tries to join a network using rejoining. If multiple rejoining attempts fail, or if NJ=0xFF, the device will attempt to join using association.

XBee Router/Coordinator Configuration

XBee routers and coordinators may require some configuration to ensure the following are set correctly:

- RF packet buffering timeout
- Child poll timeout
- Transmission timeout.

The value of these timeouts depends on the sleep time used by the end devices. Each of these timeouts are discussed below.

RF Packet Buffering Timeout

When a router or coordinator receives an RF data packet intended for one of its end device children, it buffers the packet until the end device wakes and polls for the data, or until a packet buffering timeout occurs. This timeout is settable using the SP command. The actual timeout is (1.2 * SP), with a minimum timeout of 1.2 seconds and a maximum of 30 seconds. Since the packet buffering timeout is set slightly larger than the SP setting, SP should be set the same on routers and coordinators as it is on cyclic sleep end devices. For pin sleep devices, SP should be set as long as the pin sleep device can sleep, up to 30 seconds.

Note: In pin sleep and extended cyclic sleep, end devices can sleep longer than 30 seconds. If end devices sleep longer than 30 seconds, parent and non-parent devices must know when the end device is awake in order to reliably send data. For applications that require sleeping longer than 30 seconds, end devices should transmit an IO sample or other data when they wake to alert other devices that they can send data to the end device.

Child Poll Timeout

Router and coordinator devices maintain a timestamp for each end device child indicating when the end device sent its last poll request to check for buffered data packets. If an end device does not send a poll request to its parent for a certain period of time, the parent will assume the end device has moved out of range and will remove the end device from its child table. This allows routers and coordinators to be responsive to changing network conditions. The NC command can be issued at any time to read the number of remaining (unused) child table entries on a router or coordinator.

The child poll timeout is settable with the SP and SN commands. SP and SN should be set such that SP * SN matches the longest expected sleep time of any end devices in the network. The actual timeout is calculated as (3 * SP * SN), with a minimum of 5 seconds. For networks consisting of pin sleep end devices, the SP and SN values on the coordinator and routers should be set such that SP * SN matches the longest expected sleep period of any pin sleep device. The 3 multiplier ensures the end device will not be removed unless 3 sleep cycles pass without receiving a poll request. The poll timeout is settable up to a couple months.

Transmission Timeout

As mentioned in chapter 4, when sending RF data to a remote router, since routers are always on, the timeout is based on the number of hops the transmission may traverse. This timeout it settable using the NH command. (See chapter 4 for details.)

Since end devices may sleep for lengthy periods of time, the transmission timeout to end devices also includes some allowance for the sleep period of the end device. When sending data to a remote end device, the transmission timeout is calculated using the SP and NH commands. If the timeout occurs and an acknowledgment has not been received, the source device will resend the transmission until an acknowledgment is received, up to two more times.

The transmission timeout per attempt is:

3 * ((unicast router timeout) + (end device sleep time)), or

3 * ((50 * NH) + (1.2 * SP)), where SP is measured in 10ms units.

For best results, SP should be set on routers and coordinator devices to match the SP setting on the end devices.

Note: The NH command is used to determine the timeout when transmitting to routers.

Putting it all Together

Short Sleep Periods

Pin and cyclic sleep devices that sleep less than 30 seconds can receive data transmissions at any time since their parent device(s) will be able to buffer data long enough for the end devices to wake and poll to receive the data. SP should be set the same on all devices in the network. If end devices in a network have more than one SP setting, SP on the routers and coordinators should be set to match the largest SP setting of any end device. This will ensure the RF packet buffering, poll timeout, and transmission timeouts are set correctly.

Extended Sleep Periods

Pin and cyclic sleep devices that might sleep longer than 30 seconds cannot receive data transmissions reliably unless certain design approaches are taken. Specifically, the end devices should use IO sampling or another mechanism to transmit data when they wake to inform the network they can receive data. SP and SN should be set on routers and coordinators such that (SP * SN) matches the longest expected sleep time. This configures the poll timeout so end devices are not expired from the child table unless a poll request is not received for 3 consecutive sleep periods.

As a general rule of thumb, SP and SN should be set the same on all devices in almost all cases.

Sleep Examples

This section covers some sample XBee configurations to support different sleep modes. Several AT commands are listed with suggested parameter values. The notation in this section includes an '=' sign to indicate what each command register should be set to - for example, SM=4. This is not the correct notation for setting command values in the XBee. In AT command mode, each command is issued with a leading 'AT' and no '=' sign - for example ATSM4. In the API, the two byte command is used in the command field, and parameters are populated as binary values in the parameter field.

Example 1

Configure a device to sleep for 20 seconds, but set SN such that the On/Sleep line will remain de-asserted for up to 1 minute.

The following settings should be configured on the end device.

SM = 4 (cyclic sleep) or 5 (cyclic sleep, pin wake)

SP = 0x7D0 (2000 decimal). This causes the end device to sleep for 20 seconds since SP is measured in units of 10ms.

SN = 3. (With this setting, the On/Sleep pin will assert once every 3 sleep cycles, or when RF data is received)

SO = 0

All router and coordinator devices on the network should set SP to match SP on the end device. This ensures that RF packet buffering times and transmission timeouts will be set correctly.

Since the end device wakes after each sleep period (ATSP), the SN command can be set to 1 on all routers and the coordinator.

Example 2

Configure an end device to sleep for 20 seconds, send 4 IO samples in 2 seconds, and return to sleep.

Since SP is measured in 10ms units, and ST and IR are measured in 1ms units, configure an end device with the following settings:

SM = 4 (cyclic sleep) or 5 (cyclic sleep, pin wake)

SP = 0x7D0 (2000 decimal). This causes the end device to sleep for 20 seconds.

SN = 1

SO = 0

ST = 0x7D0 (2000 decimal). This sets the sleep timer to 2 seconds.

IR = 0x258 (600 decimal). Set IR to a value greater than (2 seconds / 4) to get 4 samples in 2 seconds. The end device sends an IO sample at the IR rate until the sleep timer has expired.

At least one analog or digital IO line must be enabled for IO sampling to work. To enable pin 19 (AD1/DIO1) as a digital input line, the following must be set:

D1 = 3

All router and coordinator devices on the network should set SP to match SP on the end device. This ensures that RF packet buffering times and transmission timeouts will be set correctly.

Example 3

Configure a device for extended sleep: to sleep for 4 minutes.

SP and SN must be set such that SP * SN = 4 minutes. Since SP is measured in 10ms units, the following settings can be used to obtain 4 minute sleep.

SM = 4 (cyclic sleep) or 5 (cyclic sleep, pin wake)

SP = 0x7D0 (2000 decimal, or 20 seconds)

SN = 0x0B (12 decimal)

SO = 0x04 (enable extended sleep)

With these settings, the module will sleep for SP * SN time, or (20 seconds * 12) = 240 seconds = 4 minutes.

For best results, the end device should send a transmission when it wakes to inform the coordinator (or network) when it wakes. It should also remain awake for a short time to allow devices to send data to it. The following are recommended settings.

ST = 0x7D0 (2 second wake time)

SO = 0x06 (enable extended sleep and wake for ST time)

IR = 0x800 (send 1 IO sample after waking). At least one analog or digital IO sample should be enabled for IO sampling.

With these settings, the end device will wake after 4 minutes and send 1 IO sample. It will then remain awake for 2 seconds before returning to sleep.

SP and SN should be set to the same values on all routers and coordinators that could allow the end device to join. This will ensure the parent does not timeout the end device from its child table too quickly.

The SI command can optionally be sent to the end device to cause it to sleep before the sleep timer expires.

7. Network Commissioning and Diagnostics

Network commissioning is the process whereby devices in a mesh network are discovered and configured for operation. The XBee modules include several features to support device discovery and configuration. In addition to configuring devices, a strategy must be developed to place devices to ensure reliable routes.

To accommodate these requirements, the XBee modules include various features to aid in device placement, configuration, and network diagnostics.

Device Configuration

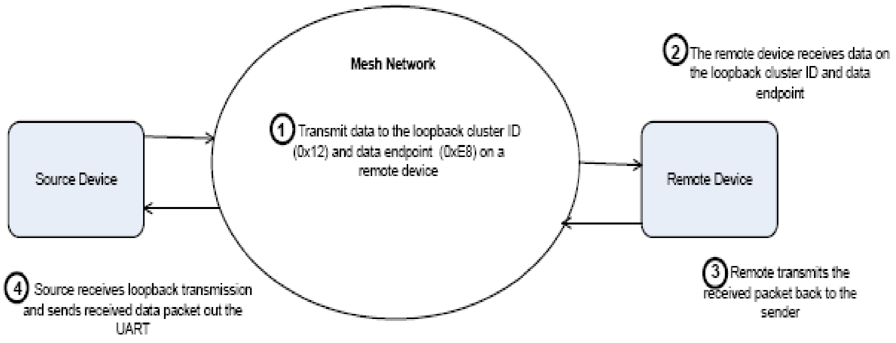
XBee/XBee-PRO ZB modules can be configured locally through serial commands (AT or API), or remotely through remote API commands. API devices can send configuration commands to set or read the configuration settings of any device in the network.

Device Placement

For a mesh network installation to be successful, the installer must be able to determine where to place individual XBee devices to establish reliable links throughout the mesh network.

Link Testing

A good way to measure the performance of a mesh network is to send unicast data through the network from one device to another to determine the success rate of many transmissions. To simplify link testing, the modules support a loopback cluster ID (0x12) on the data endpoint (0xE8). Any data sent to this cluster ID on the data endpoint will be transmitted back to the sender. This is shown in the figure below:


Mesh Network
1 Transmit data to the loopback cluster ID (0x12) and data endpoint (0xE8) on a remote device
2 The remote device receives data on the loopback cluster ID and data endpoint
Source Device
Remote Device
3 Remote transmits the received packet back to the sender
4 Source receives loopback transmission and sends received data packet out the UART

Demonstration of how the loopback cluster ID and data endpoint can be used to measure the link quality in a mesh network

The configuration steps to send data to the loopback cluster ID depend on the firmware type.

AT Firmware

To send data to the loopback cluster ID on the data endpoint of a remote device, set the CI command value to 0x12. The SE and DE commands should be set to 0xE8 (default value). The

DH and DL commands should be set to the address of the remote (0 for the coordinator, or the 64-bit address of the remote). After exiting command mode, any received serial characters will be transmitted to the remote device, and returned to the sender.

API Firmware

Send an Explicit Addressing ZigBee Command API frame (0x11) using 0x12 as the cluster ID and 0xE8 as the source and destination endpoint. Data packets received by the remote will be echoed back to the sender.

RSSI Indicators

It is possible to measure the received signal strength on a device using the DB command. DB returns the RSSI value (measured in –dBm) of the last received packet. However, this number can be misleading. The DB value only indicates the received signal strength of the last hop. If a transmission spans multiple hops, the DB value provides no indication of the overall transmission path, or the quality of the worst link – it only indicates the quality of the last link and should be used sparingly.

The DB value can be determined in hardware using the RSSI/PWM module pin (pin 6). If the RSSI PWM functionality is enabled (P0 command), when the module receives data, the RSSI PWM is set to a value based on the RSSI of the received packet. (Again, this value only indicates the quality of the last hop.) This pin could potentially be connected to an LED to indicate if the link is stable or not.

Device Discovery

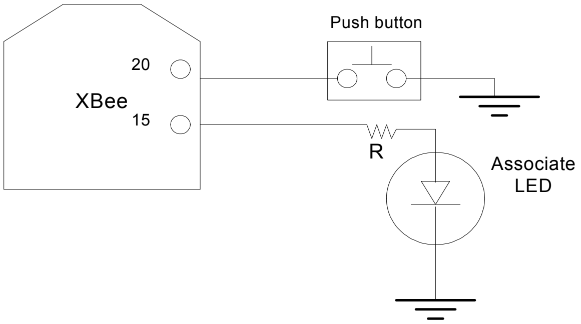
The node discovery command can be used to discover all modules that have joined a network. Issuing the ND command sends a broadcast node discovery command throughout the network. All devices that receive the command will send a response that includes the device's addressing information, node identifier string (see NI command), and other relevant information. This command is useful for generating a list of all module addresses in a network.

When a device receives the node discovery command, it waits a random time before sending its own response. The maximum time delay is set on the ND sender with the NT command. The ND originator includes its NT setting in the transmission to provide a delay window for all devices in the network. Large networks may need to increase NT to improve network discovery reliability. The default NT value is 0x3C (6 seconds).

Commissioning Pushbutton and Associate LED

The XBee modules support a set of commissioning and LED behaviors to aid in device deployment and commissioning. These include the commissioning push button definitions and associate LED behaviors. These features can be supported in hardware as shown below.

Figure 7-09.   Commissioning Pushbutton and Associate LED Functionalities



A pushbutton and an LED can be connected to module pins 20 and 15 respectively to support the commissioning pushbutton and associate LED functionalities.

Commissioning Pushbutton

The commissioning pushbutton definitions provide a variety of simple functions to aid in deploying devices in a network. The commissioning button functionality on pin 20 is enabled by setting the D0 command to 1 (enabled by default)..

Table 7-01.

Button Presses	If module is joined to a network	If module is not joined to a network
1	• Wakes an end device for 30 seconds • Sends a node identification broadcast transmission	• Wakes an end device for 30 seconds • Blinks a numeric error code on the Associate pin indicating the cause of join failure (see section 6.4.2).
2	• Sends a broadcast transmission to enable joining on the coordinator and all devices in the network for 1 minute. (If joining is permanently enabled on a device (NJ = 0xFF), this action has no effect on that device.)	• N/A
4	• Causes the device to leave the PAN. • Issues ATRE to restore module parameters to default values, including ID and SC. • The device attempts to join a network based on its ID and SC settings.	• Issues ATRE to restore module parameters to default values, including ID and SC. • The device attempts to join a network based on its ID and SC settings.

Button presses may be simulated in software using the ATCB command. ATCB should be issued with a parameter set to the number of button presses to execute. (i.e. sending ATCB1 will execute the action(s) associated with a single button press.)

The node identification frame is similar to the node discovery response frame – it contains the device's address, node identifier string (NI command), and other relevant data. All API devices that receive the node identification frame send it out their Uart as an API Node Identification Indicator frame (0x95).

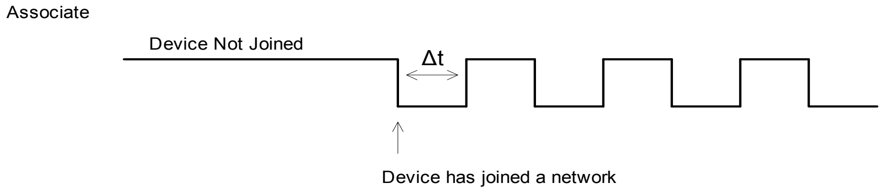
Associate LED

The Associate pin (pin 15) can provide indication of the device's network status and diagnostics information. To take advantage of these indications, an LED can be connected to the Associate pin as shown in the figure above. The Associate LED functionality is enabled by setting the D5 command to 1 (enabled by default). If enabled, the Associate pin is configured as an output and will behave as described in the following sections.

Joined Indication

The Associate pin indicates the network status of a device. If the module is not joined to a network, the Associate pin is set high. Once the module successfully joins a network, the Associate pin blinks at a regular time interval. This is shown in the following figure.

Figure 7-010. Joined Status of a Device



The associate pin can indicate the joined status of a device . Once the device has joined a network, the associate pin toggles state at a regular interval (Δt). The time can be set by using the LT command.

The LT command defines the blink time of the Associate pin. If set to 0, the device uses the default blink time (500ms for coordinator, 250ms for routers and end devices).

Diagnostics Support

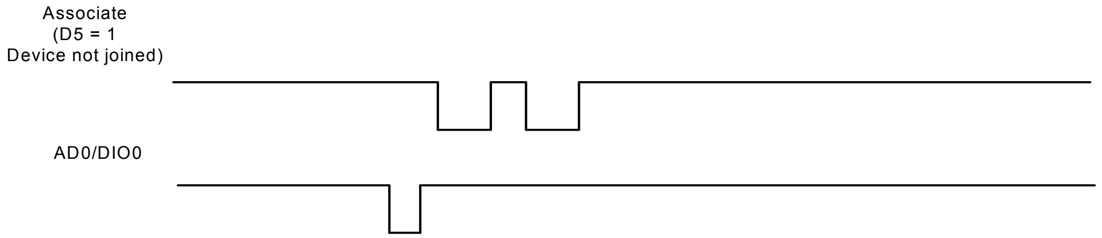
The Associate pin works with the commissioning pushbutton to provide additional diagnostics behaviors to aid in deploying and testing a network. If the commissioning push button is pressed once, and the device has not joined a network, the Associate pin blinks a numeric error code to indicate the cause of join failure. The number of blinks is equal to (AI value – 0x20). For example, if AI=0x22, 2 blinks occur.

If the commissioning push button is pressed once, and the device has joined a network, the device transmits a broadcast node identification packet. If the Associate LED functionality is enabled (D5 command), a device that receive this transmission will blink its Associate pin rapidly for 1 second.

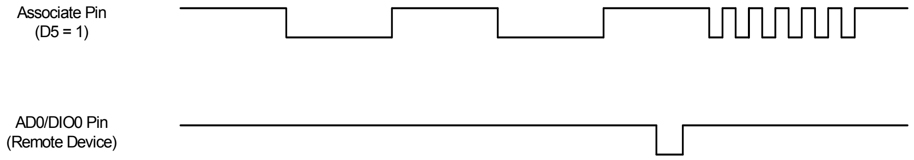
The following figures demonstrate these behaviors.

Figure 7-011. AI = 0x22

Figure 7-012.S Broadcast Node Identification Transmission



A single commissioning button press when the device has not joined a network tha causes the associate pin to blink to indicate the AI Code where: AI = # blinks + 0x2 In this example, AI = 0x22.



A single button press on a remote device causes a broadcast node identification transmission to be sent. All devices that receive this transmission blink their associate pin rapidly for one second if the associate LED functionality is enabled. (D5 = 1)

8. XBee Analog and Digital IO Lines

XBee ZB firmware supports a number of analog and digital IO pins that are configured through software commands. Analog and digital IO lines can be set or queried. The following table lists the configurable IO pins and the corresponding configuration commands.

Table 8-02.

Module Pin Names	Module Pin Numbers	Configuration Command
CD/DIO12	4	P2
PWM0/RSSIM/DIO10	6	P0
PWM/DIO11	7	P1
DIO4	11	D4
CTS/DIO7	12	D7
ASSOC/DIO5	15	D5
RTS/DIO6	16	D6
AD3/DIO3	17	D3
AD2/DIO2	18	D2
AD1/DIO1	19	DI
AD0/DIO0	20	D0

IO Configuration

To enable an analog or digital IO function on one or more XBee module pin(s), the appropriate configuration command must be issued with the correct parameter. After issuing the configuration command, changes must be applied on the module for the IO settings to take effect.

Table 8-03.

Pin Command Parameter	Description
0	Unmonitored digital input
1	Reserved for pin-specific alternate functionalities
2	Analog input, single ended (A/D pins only)
3	Digital input, monitored
4	Digital output, default low
5	Digital output, default high
6-9	Alternate functionalities, where applicable

Pull-up resistors can be set for each digital input line using the PR command. The PR value updates the state of all pull-up resistors.

IO Sampling

The XBee ZB modules have the ability to monitor and sample the analog and digital IO lines. IO samples can be read locally or transmitted to a remote device to provide indication of the current IO line states. (Only API firmware devices can send remote IO sample data out their UART.)

There are three ways to obtain IO samples, either locally or remotely:

- Queried Sampling
- Periodic Sampling
- Change Detection Sampling.

IO sample data is formatted as shown in the table below

Table 8-04.

Bytes	Name	Description
1	Sample Sets	Number of sample sets in the packet. (Always set to 1.)
2	Digital Channel Mask	Indicates which digital IO lines have sampling enabled. Each bit corresponds to one digital IO line on the module. • bit 0 = AD0/DIO0 • bit 1 = AD1/DIO1 • bit 2 = AD2/DIO2 • bit 3 = AD3/DIO3 • bit 4 = DIO4 • bit 5 = ASSOC/DIO5 • bit 6 = RTS/DIO6 • bit 7 = CTS/GPIO7 • bit 8 = N/A • bit 9 = N/A • bit 10 = RSSI/DIO10 • bit 11 = PWM/DIO11 • bit 12 = CD/DIO12 For example, a digital channel mask of 0x002F means DIO0,1,2,3, and 5 are enabled as digital IO.
1	Analog Channel Mask	Indicates which lines have analog inputs enabled for sampling. Each bit in the analog channel mask corresponds to one analog input channel. • bit 0 = AD0/DIO0 • bit 1 = AD1/DIO1 • bit 2 = AD2/DIO2 • bit 3 = AD3/DIO3 • bit 7 = Supply Voltage
Variable	Sampled Data Set	A sample set consisting of 1 sample for each enabled ADC and/or DIO channel, which has voltage inputs of 1143.75 and 342.1875mV. If any digital IO lines are enabled, the first two bytes of the data set indicate the state of all enabled digital IO. Only digital channels that are enabled in the Digital Channel Mask bytes have any meaning in the sample set. If no digital IO are enabled on the device, these 2 bytes will be omitted. Following the digital IO data (if any), each enabled analog channel will return 2 bytes. The data starts with AIN0 and continues sequentially for each enabled analog input channel up to AIN3, and the supply voltage (if enabled) at the end.

The sampled data set will include 2 bytes of digital IO data only if one or more IO lines on the device are configured as digital IO. If no pins are configured as digital IO, these 2 bytes will be omitted.

The digital IO data is only relevant if the same bit is enabled in the digital IO mask as shown in the following figure:

Analog samples are returned as 10-bit values. The analog reading is scaled such that 0x0000 represents 0V, and 0x3FF = 1.2V. (The analog inputs on the module cannot read more than 1.2V.) Analog samples are returned in order starting with AIN0 and finishing with AIN3, and the supply voltage. Only enabled analog input channels return data as shown in the figure below.

To convert the A/D reading to mV, do the following:

AD(mV) = (A/D reading * 1200mV) / 1024

TThe reading in the sample frame represents voltage inputs of 1143.75 and 342.1875mV for AD0 and AD1 respectively.

Queried Sampling

The IS command can be sent to a device locally, or to a remote device using the API remote command frame (see Chapter 8 for details). When the IS command is sent, the receiving device samples all enabled digital IO and analog input channels and returns an IO sample. If IS is sent locally, the IO sample is sent out the uart. If the IS command was received as a remote command, the IO sample is sent over-the-air to the device that sent the IS command.

If the IS command is issued in AT firmware, the module returns a carriage return-delimited list containing the above-listed fields. The API firmware returns an AT command response packet with the IO data included in the command data portion of the response frame.

The following table shows an example of the fields in an IS reponse.

.

Table 8-05.

Example	Sample AT Response
0x01	[1 sample set]
0x0C0C	[Digital Inputs: DIO 2, 3, 10, 11 low]
0x03	[Analog Inputs: A/D 0, 1]
0x0408	[Digital input states: DIO 3, 10 high, DIO 2, 11 low]
0x03D0	[Analog input ADIO 0= 0x3D0]
0x0124	[Analog input ADIO 1=0x120]

Periodic IO Sampling

Periodic sampling allows an XBee / XBee-PRO module to take an IO sample and transmit it to a remote device at a periodic rate. The periodic sample rate is set by the IR command. If IR is set to 0, periodic sampling is disabled. For all other values of IR, data will be sampled after IR milliseconds have elapsed and transmitted to a remote device. The DH and DL commands determine the destination address of the IO samples. DH and DL can be set to 0 to transmit to the coordinator, or to the 64-bit address of the remote device (SH and SL). Only devices running API firmware can send IO data samples out their Uart. Devices running AT firmware will discard received IO data samples.

A sleepy end device will transmit periodic IO samples at the IR rate until the ST timer expires and the device can resume sleeping.

Change Detection Sampling

Modules can be configured to transmit a data sample immediately whenever a monitored digital IO pin changes state. The IC command is a bitmask that can be used to set which digital IO lines should be monitored for a state change. If one or more bits in IC is set, an IO sample will be transmitted as soon as a state change is observed in one of the monitored digital IO lines. Change detection samples are transmitted to the 64-bit address specified by DH and DL.

RSSI PWM

The XBee module features an RSSI/PWM pin (pin 6) that, if enabled, will adjust the PWM output to indicate the signal strength of the last received packet. The P0 (P-zero) command is used to enable the RSSI pulse width modulation (PWM) output on the pin. If P0 is set to 1, the RSSI/PWM pin will output a pulse width modulated signal where the frequency is adjusted based on the received signal strength of the last packet. Otherwise, for all other P0 settings, the pin can be used for general purpose IO.

When a data packet is received, if P0 is set to enable the RSSI/PWM feature, the RSSI PWM output is adjusted based on the RSSI of the last packet. The RSSI/PWM output will be enabled for a time based on the RP command. Each time an RF packet is received, the RSSI/PWM output is adjusted based on the RSSI of the new packet, and the RSSI timer is reset. If the RSSI timer expires, the RSSI/PWM pin is driven low. RP is measured in 100ms units and defaults to a value of 40 (4 seconds).

The RSSI PWM runs at 12MHz and has 2400 total counts (200us period).

RSSI (in dBm) is converted to PWM counts using the following equation:

PWM counts = (41 * RSSI_Unsigned) - 5928

IO Examples

Example 1: Configure the following IO settings on the XBee.

Configure AD1/DIO1 as a digital input with pullup resistor enabled

Configure AD2/DIO2 as an analog input

Configure DIO4 as a digital output, driving high.

To configure AD1/DIO1 as an input, issue the ATD1 command with a parameter of 3 ("ATD13"). To enable pull-up resistors on the same pin, the PR command should be issued with bit 3 set (i.e. ATPR8, ATPR1FFF, etc).

The ATD2 command should be issued with a parameter of 2 to enable the analog input ("ATD22"). Finally, DIO4 can be set as an output, driving high by issuing the ATD4 command with a parameter value of 5 ("ATD45").

After issuing these commands, changes must be applied before the module IO pins will be updated to the new states. The AC or CN commands can be issued to apply changes (i.e. ATAC).

Example 2: Calculate the PWM counts for a packet received with an RSSI of -84dBm.

RSSI = -84 = 0xAC = 172 decimal (unsigned)

PWM counts = (41 * 172) - 5928

PWM counts = 1124

With a total of 2400 counts, this yields an ON time of (1124 / 2400) = 46.8%

Example 3: Configure the RSSI/PWM pin to operate for 2 seconds after each received RF packet.

First, ensure the RSSI/PWM functionality is enabled by reading the P0 (P-zero) command. It should be set to 1 (default).

To configure the duration of the RSSI/PWM output, set the RP command. To achieve a 2 second PWM output, set RP to 0x14 (20 decimal, or 2 seconds) and apply changes (AC command).

After applying changes, all received RF data packets should set the RSSI timer for 2 seconds.

9. API Operation

As an alternative to Transparent Operation, API (Application Programming Interface) Operations are available. API operation requires that communication with the module be done through a structured interface (data is communicated in frames in a defined order). The API specifies how commands, command responses and module status messages are sent and received from the module using a UART Data Frame.

Please note that Digi may add new API frames to future versions of firmware, so please build into your software interface the ability to filter out additional API frames with unknown Frame Types.

API Frame Specifications

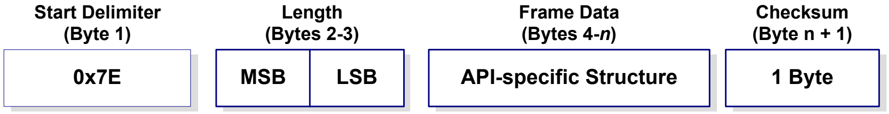
Two API modes are supported and both can be enabled using the AP (API Enable) command. Use the following AP parameter values to configure the module to operate in a particular mode:

- AP = 1: API Operation
- AP = 2: API Operation (with escaped characters)

API Operation (AP parameter = 1)

When this API mode is enabled (AP = 1), the UART data frame structure is defined as follows:

Figure 9-01. UART Data Frame Structure:



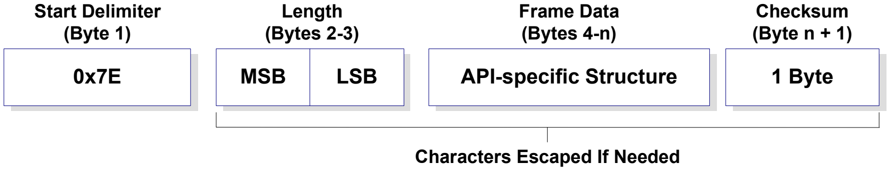
MSB = Most Significant Byte, LSB = Least Significant Byte

Any data received prior to the start delimiter is silently discarded. If the frame is not received correctly or if the checksum fails, the module will reply with a module status frame indicating the nature of the failure.

API Operation - with Escape Characters (AP parameter = 2)

When this API mode is enabled (AP = 2), the UART data frame structure is defined as follows:

Figure 9-02. UART Data Frame Structure - with escape control characters:



MSB = Most Significant Byte, LSB = Least Significant Byte

Escape characters. When sending or receiving a UART data frame, specific data values must be escaped (flagged) so they do not interfere with the data frame sequencing. To escape an interfering data byte, insert 0x7D and follow it with the byte to be escaped XOR'd with 0x20.

Data bytes that need to be escaped:

- 0x7E – Frame Delimiter
- 0x7D – Escape
- 0x11 – XON
- 0x13 – XOFF

Example - Raw UART Data Frame (before escaping interfering bytes):
0x7E 0x00 0x02 0x23 0x11 0xCB

0x11 needs to be escaped which results in the following frame:
0x7E 0x00 0x02 0x23 0x7D 0x31 0xCB

Note: In the above example, the length of the raw data (excluding the checksum) is 0x0002 and the checksum of the non-escaped data (excluding frame delimiter and length) is calculated as: 0xFF - (0x23 + 0x11) = (0xFF - 0x34) = 0xCB.

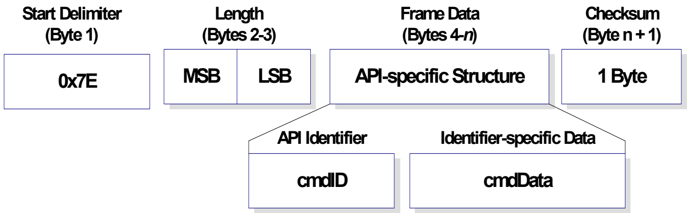
Length

The length field has two-byte value that specifies the number of bytes that will be contained in the frame data field. It does not include the checksum field.

Frame Data

Frame data of the UART data frame forms an API-specific structure as follows:

Figure 9-03. UART Data Frame & API-specific Structure:



The cmdID frame (API-identifier) indicates which API messages will be contained in the cmdData frame (Identifier-specific data). Note that multi-byte values are sent big endian.The XBee modules support the following API frames:

Table 9-06. API Frame Names and Values

API Frame Names	API ID
AT Command	0x08
AT Command - Queue Parameter Value	0x09
ZigBee Transmit Request	0x10
Explicit Addressing ZigBee Command Frame	0x11
Remote Command Request	0x17
Create Source Route	0x21
AT Command Response	0x88
Modem Status	0x8A
ZigBee Transmit Status	0x8B
ZigBee Receive Packet (AO=0)	0x90
ZigBee Explicit Rx Indicator (AO=1)	0x91
ZigBee IO Data Sample Rx Indicator	0x92
XBee Sensor Read Indicator (AO=0)	0x94
Node Identification Indicator (AO=0)	0x95
Remote Command Response	0x97
Over-the-Air Firmware Update Status	0xA0
Route Record Indicator	0xA1

Checksum

To test data integrity, a checksum is calculated and verified on non-escaped data.

To calculate: Not including frame delimiters and length, add all bytes keeping only the lowest 8 bits of the result and subtract the result from 0xFF.

To verify: Add all bytes (include checksum, but not the delimiter and length). If the checksum is correct, the sum will equal 0xFF.

API Examples

Example: Create an API AT command frame to configure an XBee to allow joining (set NJ to 0xFF). The frame should look like:

0x7E 0x00 0x05 0x08 0x01 0x4E 0x4A 0xFF 5F

Where 0x0005 = length

0x08 = AT Command API frame type

0x01 = Frame ID (set to non-zero value)

0x4E4A = AT Command ('NJ')

0xFF = value to set command to

0x5F = Checksum

The checksum is calculated as [0xFF - (0x08 + 0x01 + 0x4E + 0x4A + 0xFF)]

Example: Send an ND command to discover the devices in the PAN. The frame should look like:

0x7E 0x00 0x04 0x08 0x01 0x4E 0x44 0x64

Where 0x0004 = length

0x08 = AT Command API frame type

0x01 = Frame ID (set to non-zero value)

0x4E44 = AT command ('ND')

0x64 = Checksum

The checksum is calculated as [0xFF - (0x08 + 0x01 + 0x4E + 0x44)]

Example: Send a remote command to the coordinator to set AD1/DIO1 as a digital input (D1=3) and apply changes to force the IO update. The API remote command frame should look like:

0x7E 0x00 0x10 0x17 0x01 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0xFF 0xFE 0x02 0x44 0x31 0x03 0x70

Where

0x10 = length (16 bytes excluding checksum)

0x17 = Remote Command API frame type

0x01 = Frame ID

0x0000000000000000 = Coordinator's address (can be replaced with coordinator's actual 64-bit address if known)

0xFFFE = 16- bit Destination Address

0x02 = Apply Changes (Remote Command Options)

0x4431 = AT command ('D1')

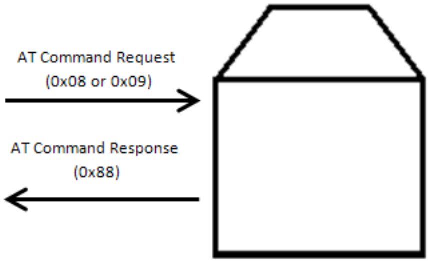
0x03 = Command Parameter (the parameter could also be sent as 0x0003 or 0x00000003)

0x70 = Checksum

API UART Exchanges

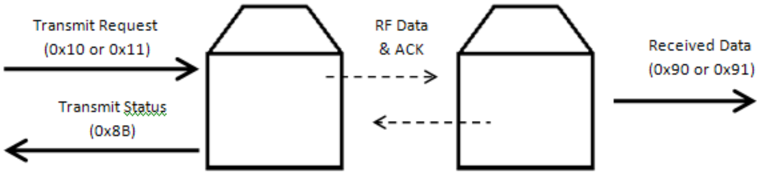
AT Commands

The following image shows the API frame exchange that takes place at the UART when sending an AT command request to read or set a module parameter. The response can be disabled by setting the frame ID to 0 in the request.



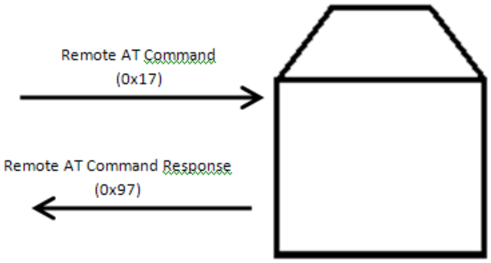
Transmitting and Receiving RF Data

The following image shows the API exchanges that take place at the UART when sending RF data to another device. The transmit status frame is always sent at the end of a data transmission unless the frame ID is set to 0 in the transmit request. If the packet cannot be delivered to the destination, the transmit status frame will indicate the cause of failure. The received data frame (0x90 or 0x91) is set by the AP command.



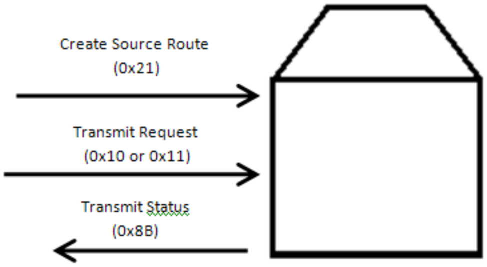
Remote AT Commands

The following image shows the API frame exchanges that take place at the UART when sending a remote AT command. A remote command response frame is not sent out the UART if the remote device does not receive the remote command.



Source Routing

The following image shows the API frame exchanges that take place at the UART when sending a source routed transmission.



Supporting the API

Applications that support the API should make provisions to deal with new API frames that may be introduced in future releases. For example, a section of code on a host microprocessor that handles received serial API frames (sent out the module's DOUT pin) might look like this:

```
void XBee_HandleRxAPIFrame(_apiFrameUnion *papiFrame){
   switch(papiFrame->api_id){
      case RX_RF_DATA_FRAME:
         //process received RF data frame
         break;

      case RX_IO_SAMPLE_FRAME:
         //process IO sample frame
         break;

      case NODE_IDENTIFICATION_FRAME:
         //process node identification frame
         break;

      default:
         //Discard any other API frame types that are not being used
         break;
   }
}
```

API Frames

The following sections illustrate the types of frames encountered while using the API.

AT Command

Frame Type: 0x08
Used to query or set module parameters on the local device. This API command applies changes after executing the command. (Changes made to module parameters take effect once changes are applied.) The API example below illustrates an API frame when modifying the NJ parameter value of the module

	Frame Fields		Offset	Example	Description
API Packet	Start Delimiter		0	0x7E	
	Length		MSB 1	0x00	Number of bytes between the length and the checksum
			LSB 2	0x04	
	Frame-specific Data	Frame Type	3	0x08	
		Frame ID	4	0x52 (R)	Identifies the UART data frame for the host to correlate with a subsequent ACK (acknowledgement). If set to 0, no response is sent.
		AT Command	5	0x4E (N)	Command Name - Two ASCII characters that identify the AT Command.
			6	0x4A (J)	
		Parameter Value (optional)			If present, indicates the requested parameter value to set the given register. If no characters present, register is queried.
	Checksum		9	0x0D	0xFF - the 8 bit sum of bytes from offset 3 to this byte.

The above example illustrates an AT command when querying an NJ value.

AT Command - Queue Parameter Value

Frame Type: 0x09
This API type allows module parameters to be queried or set. In contrast to the "AT Command" API type, new parameter values are queued and not applied until either the "AT Command" (0x08) API type or the AC (Apply Changes) command is issued. Register queries (reading parameter values) are returned immediately.

Example: Send a command to change the baud rate (BD) to 115200 baud, but don't apply changes yet. (Module will continue to operate at the previous baud rate until changes are applied.)

	Frame Fields		Offset	Example	Description
API Packet	Start Delimiter		0	0x7E	
	Length		MSB 1	0x00	Number of bytes between the length and the checksum
			LSB 2	0x05	
	Frame-specific Data	Frame Type	3	0x09	
		Frame ID	4	0x01	Identifies the UART data frame for the host to correlate with a subsequent ACK (acknowledgement). If set to 0, no response is sent.
		AT Command	5	0x42 (B)	Command Name - Two ASCII characters that identify the AT Command.
			6	0x44 (D)	
		Parameter Value (ATBD7 = 115200 baud)		0x07	If present, indicates the requested parameter value to set the given register. If no characters present, register is queried.
	Checksum		9	0x68	0xFF - the 8 bit sum of bytes from offset 3 to this byte.

Note: In this example, the parameter could have been sent as a zero-padded 2-byte or 4-byte value.

ZigBee Transmit Request

Frame Type: 0x10

A Transmit Request API frame causes the module to send data as an RF packet to the specified destination.

The 64-bit destination address should be set to 0x000000000000FFFF for a broadcast transmission (to all devices). The coordinator can be addressed by either setting the 64-bit address to all 0x00s and the 16-bit address to 0xFFFE, OR by setting the 64-bit address to the coordinator's 64-bit address and the 16-bit address to 0x0000. For all other transmissions, setting the 16-bit address to the correct 16-bit address can help improve performance when transmitting to multiple destinations. If a 16-bit address is not known, this field should be set to 0xFFFE (unknown). The Transmit Status frame (0x8B) will indicate the discovered 16-bit address, if successful.

The broadcast radius can be set from 0 up to NH. If set to 0, the value of NH specifies the broadcast radius (recommended). This parameter is only used for broadcast transmissions.

The maximum number of payload bytes can be read with the NP command.

Note: if source routing is used, the RF payload will be reduced by two bytes per intermediate hop in the source route. This exampe shows if escaping is disabled (AP=1).

	Frame Fields		Offset	Example	Description
API Packet	Start Delimiter		0	0x7E	
	Length		MSB 1	0x00	Number of bytes between the length and the checksum
			LSB 2	0x16	
	Frame-specific Data	Frame Type	3	0x10	
		Frame ID	4	0x01	Identifies the UART data frame for the host to correlate with a subsequent ACK (acknowledgement). If set to 0, no response is sent.
		64-bit Destination Address	MSB 5	0x00	Set to the 64-bit address of the destination device. The following addresses are also supported: 0x0000000000000000 - Reserved 64-bit address for the coordinator 0x000000000000FFFF - Broadcast address
			6	0x13	
			7	0xA2	
			8	0x00	
			9	0x40	
			10	0x0A	
			11	0x01	
			LSB 12	0x27	
		16-bit Destination Network Address	MSB 13	0xFF	Set to the 16-bit address of the destination device, if known. Set to 0xFFFE if the address is unknown, or if sending a broadcast.
			LSB 14	0xFE	
		Broadcast Radius	15	0x00	Sets maximum number of hops a broadcast transmission can occur. If set to 0, the broadcast radius will be set to the maximum hops value.
		Options	16	0x00	All other bits must be set to 0.
		RF Data	17	0x54	Data that is sent to the destination device
			18	0x78	
			19	0x44	
			20	0x61	
			21	0x74	
			22	0x61	
			23	0x30	
			24	0x41	
	Checksum		25	0x13	0xFF - the 8 bit sum of bytes from offset 3 to this byte.

Example: The example above shows how to send a transmission to a module where escaping is disabled (AP=1) with destination address 0x0013A200 40014011, payload "TxData1B". If escaping is enabled (AP=2), the frame should look like:

0x7E 0x00 0x16 0x10 0x01 0x00 0x7D 0x33 0xA2 0x00 0x40 0x0A 0x01 0x27

0xFF 0xFE 0x00 0x00 0x54 0x78 0x44 0x61 0x74 0x61 0x30 0x41 0x7D 0x33

The checksum is calculated (on all non-escaped bytes) as [0xFF - (sum of all bytes from API frame type through data payload)].

Example: Send a transmission to the coordinator without specifying the coordinator's 64-bit address. The API transmit request frame should look like:

0x7E 0x00 0x16 0x10 0x01 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0xFF 0xFE 0x00 0x00 0x54 0x78 032 0x43 0x6F 0x6F 0x72 0x64 0xFC

Where 0x16 = length (22 bytes excluding checksum)

0x10 = ZigBee Transmit Request API frame type

0x01 = Frame ID (set to non-zero value)

0x0000000000000000 = Coordinator's address (can be replaced with coordinator's actual 64-bit address if known

0xFFFE = 16-bit Destination Address

0x00 = Broadcast radius

0x00 = Options

0x547832436F6F7264 = Data payload ("Tx2Coord")

0xFC = Checksum

Explicit Addressing ZigBee Command Frame

Frame Type: 0x11

Allows ZigBee application layer fields (endpoint and cluster ID) to be specified for a data transmission.
Similar to the ZigBee Transmit Request, but also requires ZigBee application layer addressing fields to be specified (endpoints, cluster ID, profile ID). An Explicit Addressing Request API frame causes the module to send data as an RF packet to the specified destination, using the specified source and destination endpoints, cluster ID, and profile ID.

The 64-bit destination address should be set to 0x000000000000FFFF for a broadcast transmission (to all devices). The coordinator can be addressed by either setting the 64-bit address to all 0x00s and the 16-bit address to 0xFFFE, OR by setting the 64-bit address to the coordinator's 64-bit address and the 16-bit address to 0x0000. For all other transmissions, setting the 16-bit address to the correct 16-bit address can help improve performance when transmitting to multiple destinations. If a 16-bit address is not known, this field should be set to 0xFFFE (unknown). The Transmit Status frame (0x8B) will indicate the discovered 16-bit address, if successful.

The broadcast radius can be set from 0 up to NH. If set to 0, the value of NH specifies the broadcast radius (recommended). This parameter is only used for broadcast transmissions.

The maximum number of payload bytes can be read with the NP command. Note: if source routing is used, the RF payload will be reduced by two bytes per intermediate hop in the source route.

	Frame Fields		Offset	Example	Description
API Packet	Start Delimiter		0	0x7E	
	Length		MSB 1	0x00	Number of bytes between the length and the checksum
			LSB 2	0x1A	
	Frame-specific Data	Frame Type	3	0x11	
		Frame ID	4	0x01	Identifies the UART data frame for the host to correlate with a subsequent ACK (acknowledgement). If set to 0, no response is sent.
		64-bit Destination Address	MSB 5	0x00	Set to the 64-bit address of the destination device. The following addresses are also supported: 0x0000000000000000 - Reserved 64-bit address for the coordinator 0x000000000000FFFF - Broadcast address
			6	0x00	
			7	0x00	
			8	0x00	
			9	0x00	
			10	0x00	
			11	0x00	
			12	0x00	
		16-bit Destination Network Address	MSB 13	0xFF	Set to the 16-bit address of the destination device, if known. Set to 0xFFFE if the address is unknown, or if sending a broadcast.
			LSB 14	0xFE	
		Source Endpoint	15	0xA0	Source endpoint for the transmission.
		Destination Endpoint	16	0xA1	Destination endpoint for the transmission.
		Cluster ID	17	0x15	Cluster ID used in the transmission
			18	0x54	
		Profile ID	19	0xC1	Profile ID used in the transmission
			20	0x05	
		Broadcast Radius	21	0x00	Sets the maximum number of hops a broadcast transmission can traverse. If set to 0, the transmission radius will be set to the network maximum hops value.
		Transmit Options	22	0x00	All bits must be set to 0.
		Data Payload	23	0x54	
			24	0x78	
			25	0x44	
			26	0x61	
			27	0x74	
			28	0x61	
	Checksum		29	0x3A	0xFF - the 8 bit sum of bytes from offset 3 to this byte.

Example: Send a data transmission to the coordinator (64-bit address of 0x00s) using a source endpoint of 0xA0, destination endpoint 0xA1, cluster ID =0x1554, and profile ID 0xC105. Payload will be "TxData".

Remote AT Command Request

Frame Type: 0x17

Used to query or set module parameters on a remote device. For parameter changes on the remote device to take effect, changes must be applied, either by setting the apply changes options bit, or by sending an AC command to the remote.

	Frame Fields		Offset	Example	Description
API Packet	**Start Delimiter**		0	0x7E	
	Length		MSB 1	0x00	Number of bytes between the length and the checksum
			LSB 2	0x10	
	Frame-specific Data	**Frame Type**	3	0x17	
		Frame ID	4	0x01	Identifies the UART data frame for the host to correlate with a subsequent ACK (acknowledgement). If set to 0, no response is sent.
		64-bit Destination Address	MSB 5	0x00	Set to the 64-bit address of the destination device. The following addresses are also supported: 0x0000000000000000 - Reserved 64-bit address for the coordinator 0x000000000000FFFF - Broadcast address
			6	0x13	
			7	0xA2	
			8	0x00	
			9	0x40	
			10	0x40	
			11	0x11	
			LSB 12	0x22	
		16-bit Destination Network Address	MSB 13	0xFF	Set to the 16-bit address of the destination device, if known. Set to 0xFFFE if the address is unknown, or if sending a broadcast.
			LSB 14	0xFE	
		Remote Command Options	15	0x02 (apply changes)	0x02 - Apply changes on remote. (If not set, AC command must be sent before changes will take effect.) All other bits must be set to 0.
		AT Command	16	0x42 (B)	Name of the command
			17	0x48 (H)	
		Command Parameter	18	0x01	If present, indicates the requested parameter value to set the given register. If no characters present, the register is queried.
	Checksum		18	0xF5	0xFF - the 8 bit sum of bytes from offset 3 to this byte.

Example: Send a remote command to change the broadcast hops register on a remote device to 1 (broadcasts go to 1-hop neighbors only), and apply changes so the new configuration value immediately takes effect. In this example, the 64-bit address of the remote is 0x0013A200 40401122, and the destination 16-bit address is unknown.

Create Source Route

Frame Type: 0x21

This frame creates a source route in the module. A source route specifies the complete route a packet should traverse to get from source to destination. Source routing should be used with many-to-one routing for best results.

Note: Both the 64-bit and 16-bit destination addresses are required when creating a source route. These are obtained when a Route Record Indicator (0xA1) frame is received.

Frame Fields		Offset	Example	Description
Start Delimiter		0	0x7E	
Length		MSB 1	0x00	Number of bytes between the length and the checksum
		LSB 2	0x14	
Frame-specific Data	Frame Type	3	0x21	
	Frame ID	4	0x00	The Frame ID should always be set to 0.
	64-bit Destination Address	MSB 5	0x00	Set to the 64-bit address of the destination device. The following addresses are also supported: 0x0000000000000000 - Reserved 64-bit address for the coordinator 0x000000000000FFFF - Broadcast address
		6	0x13	
		7	0xA2	
		8	0x00	
		9	0x40	
		10	0x40	
		11	0x11	
		LSB 12	0x22	
	16-bit Destination Network Address	MSB 13	0x33	Set to the 16-bit address of the destination device, if known. Set to 0xFFFE if the address is unknown, or if sending a broadcast.
		LSB 14	0x44	
	Route Command Options	15	0x00	Set to 0.
	Number of Addresses	16	0x03	The number of addresses in the source route (excluding source and destination).
	Address 1	17	0xEE	(neighbor of destination)
		18	0xFF	
	Address 2 (closer hop	19	0xCC	Address of intermediate hop
		20	0xDD	
	Address 3	21	0xAA	(neighbor of source)
		22	0xBB	
Checksum		23	0x01	0xFF - the 8 bit sum of bytes from offset 3 to this byte.

Example: Intermediate hop addresses must be ordered starting with the neighbor of the destination, and working closer to the source. For example, suppose a route is found between A and E as shown below.

A ' B ' C ' D ' E

If device E has the 64-bit and 16-bit addresses of 0x0013A200 40401122 and 0x3344, and if devices B, C, and D have the following 16-bit addresses:

B = 0xAABB

C = 0xCCDD

D = 0xEEFF

The example above shows how to send the Create Source Route frame to establish a source route between A and E.

AT Command Response

Frame Type: 0x88
In response to an AT Command message, the module will send an AT Command Response message. Some commands will send back multiple frames (for example, the ND (Node Discover) command).

	Frame Fields		Offset	Example	Description
API Packet	Start Delimiter		0	0x7E	
	Length		MSB 1	0x00	Number of bytes between the length and the checksum
			LSB 2	0x05	
	Frame-specific Data	Frame Type	3	0x88	
		Frame ID	4	0x01	Identifies the UART data frame being reported. Note: If Frame ID = 0 in AT Command Mode, no AT Command Response will be given.
		AT Command	5	'B' = 0x42	Command Name - Two ASCII characters that identify the AT Command.
			6	'D' = 0x44	
		Command Status	7	0x00	0 = OK 1 = ERROR 2 = Invalid Command 3 = Invalid Parameter
		Command Data			Register data in binary format. If the register was set, then this field is not returned, as in this example.
	Checksum		8	0xF0	0xFF - the 8 bit sum of bytes from offset 3 to this byte.

Example: Suppose the BD parameter is changed on the local device with a frame ID of 0x01. If successful (parameter was valid), the following response would be received.

Modem Status

Frame Type: (0x8A)
RF module status messages are sent from the module in response to specific conditions.

Example: The following API frame is returned when an API coordinator forms a network.

	Frame Fields		Offset	Example	Description
API Packet	Start Delimiter		0	0x7E	
	Length		MSB 1	0x00	Number of bytes between the length and the checksum
			LSB 2	0x02	
	Frame-specific Data	Frame Type	3	0x8A	
		Status	4	0x06	0 = Hardware reset 1 = Watchdog timer reset 2 =Joined network (routers and end devices) 3 =Disassociated 6 =Coordinator started
	Checksum		5	0x6F	0xFF - the 8 bit sum of bytes from offset 3 to this byte.

ZigBee Transmit Status

Frame Type: 0x8B
When a TX Request is completed, the module sends a TX Status message. This message will indicate if the packet was transmitted successfully or if there was a failure.

	Frame Fields		Offset	Example	Description
API Packet	Start Delimiter		0	0x7E	
	Length		MSB 1	0x00	Number of bytes between the length and the checksum
			LSB 2	0x07	
	Frame-specific Data	Frame Type	3	0x8B	
		Frame ID	4	0x01	Identifies the UART data frame being reported. Note: If Frame ID = 0 in AT Command Mode, no AT Command Response will be given.
		16-bit address of destination0x7D	5	0x7D	16-bit Network Address the packet was delivered to (if success). If not success, this address matches the Destination Network Address that was provided in the Transmit Request Frame.
			6	0x84	
		Transmit Retry Count	7	0x00	The number of application transmission retries that took place.
		Delivery Status	8	0x00	0x00 = Success 0x02 = CCA Failure 0x15 = Invalid destination endpoint 0x21 = Network ACK Failure 0x22 = Not Joined to Network 0x23 = Self-addressed 0x24 = Address Not Found 0x25 = Route Not Found 0x74 = Data payload too large
		Discovery Status	9	0x01	0x00 = No Discovery Overhead 0x01 = Address Discovery 0x02 = Route Discovery 0x03 = Address and Route Discovery
	Checksum		10	0x71	0xFF - the 8 bit sum of bytes from offset 3 to this byte.

Example: Suppose a unicast data transmission was sent to a destination device with a 16-bit address of 0x7D84. (The transmission could have been sent with the 16-bit address set to 0x7D84 or 0xFFFE.)

ZigBee Receive Packet

Frame Type: (0x90)

When the module receives an RF packet, it is sent out the UART using this message type.

	Frame Fields		Offset	Example	Description
API Packet	Start Delimiter		0	0x7E	
	Length		MSB 1	0x00	Number of bytes between the length and the checksum
	Frame-specific Data		LSB 2	0x12	
		Frame Type	3	0x90	
		Frame ID	4	0x00	Identifies the UART data frame for the host to correlate with a subsequent ACK (acknowledgement). If set to 0, no response is sent.
		64-bit Source Address	MSB 5	0x13	64-bit address of sender
			6	0xA2	
			7	0x00	
			8	0x40	
			9	0x52	
			10	0x2B	
			LSB 11	0xAA	
		16-bit Source Network Address	MSB 12	0x7D	16-bit address of sender
			LSB 13	0x84	
		Receive Options	14	0x01	0x01 - Packet Acknowledged 0x02 - Packet was a broadcast packet
		Received Data	15	0x52	Received RF data
			16	0x78	
			17	0x44	
			18	0x61	
			19	0x74	
			20	0x61	
	Checksum		21	0x0D	0xFF - the 8 bit sum of bytes from offset 3 to this byte.

Example: Suppose a device with a 64-bit address of 0x0013A200 40522BAA, and 16-bit address 0x7D84 sends a unicast data transmission to a remote device with payload "RxData". If AO=0 on the receiving device, it would send the following frame out its UART.

ZigBee Explicit Rx Indicator

Frame Type:0x91

When the modem receives a ZigBee RF packet it is sent out the UART using this message type (when AO=1).

	Frame Fields		Offset	Example	Description
API Packet	Start Delimiter		0	0x7E	
	Length		MSB 1	0x00	Number of bytes between the length and the checksum
	Frame-specific Data		LSB 2	0x18	
		Frame Type	3	0x91	
		64-bit Source Address	MSB 4	0x00	64-bit address of sender
			5	0x13	
			6	0xA2	
			7	0x00	
			8	0x40	
			9	0x52	
			10	0x2B	
			LSB 11	0xAA	
		16-bit Source Network Address	MSB 12	0x7D	16-bit address of sender.
			LSB 13	0x84	
		Source Endpoint	14	0xE0	Endpoint of the source that initiated the transmission
		Destination Endpoint	15	0xE0	Endpoint of the destination the message is addressed to.
		Cluster ID	16	0x22	Cluster ID the packet was addressed to.
			17	0x11	
		Profile ID	18	0xC1	Profile ID the packet was addressed to.
			19	0x05	
		Receive Options	20	0x02	0x01 – Packet Acknowledged 0x02 – Packet was a broadcast packet
		Received Data	21	0x52	Received RF data
			22	0x78	
			23	0x44	
			24	0x61	
			25	0x74	
			26	0x61	
	Checksum		27	0x52	0xFF - the 8 bit sum of bytes from offset 3 to this byte.

Example: Suppose a device with a 64-bit address of 0x0013A200 40522BAA, and 16-bit address 0x7D84 sends a broadcast data transmission to a remote device with payload "RxData". Suppose the transmission was sent with source and destination endpoints of 0xE0, cluster ID=0x2211, and profile ID=0xC105. If AO=1 on the receiving device, it would send the following frame out its UART.

ZigBee IO Data Sample Rx Indicator

Frame Type: 0x92

When the module receives an IO sample frame from a remote device, it sends the sample out the UART using this frame type (when AO=0). Only modules running API firmware will send IO samples out the UART.

	Frame Fields		Offset	Example	Description
API Packet	Start Delimiter		0	0x7E	
	Length		MSB 1	0x00	Number of bytes between the length and the checksum
	Frame-specific Data		LSB 2	0x14	
		Frame Type	3	0x92	
		64-bit Source Address	MSB 4	0x00	64-bit address of sender
			5	0x13	
			6	0xA2	
			7	0x00	
			8	0x40	
			9	0x52	
			10	0x2B	
			LSB 11	0xAA	
		16-bit Source Network Address	MSB 12	0x7D	16-bit address of sender.
			LSB 13	0x84	
		Receive Options	14	0x01	0x01 - Packet Acknowledged 0x02 - Packet was abroadcast packet
		Number of Samples	15	0x01	Number of sample sets included in the payload. (Always set to 1)
		Digital Channel Mask*	16	0x00	Bitmask field that indicates which digital IO lines on the remote have sampling enabled (if any).
			17	0x1C	
		Analog Channel Mask**	18	0x02	Bitmask field that indicates which analog IO lines on the remote have sampling enabled (if any).
		Digital Samples (if included)	19	0x00	If the sample set includes any digital IO lines (Digital Channel Mask > 0), these two bytes contain samples for all enabled digital IO lines. DIO lines that do not have sampling enabled return 0. Bits in these 2 bytes map the same as they do in the Digital Channels Mask field.
			20	0x14	
		Analog Sample	21	0x02	If the sample set includes any analog input lines (Analog Channel Mask > 0), each enabled analog input returns a 2-byte value indicating the A/D measurement of that input. Analog samples are ordered sequentially from AD0/DIO0 to AD3/DIO3, to the supply voltage.
			22	0x25	
	Checksum		23	0xF5	0xFF - the 8 bit sum of bytes from offset 3 to this byte.

*	N/A	N/A	N/A	CD/DIO12	PWM/DIO11	RSSI/DIO10	N/A	N/A
	CTS/DIO7	RTS/DIO6	ASSOC/DIO5	DIO4	AD3/DIO3	AD2/DIO2	AD1/DIO1	AD0/DIO0
**	Supply Voltage	N/A	N/A	N/A	AD3	AD2	AD1	AD0

Example: Suppose an IO sample is received with analog and digital IO, from a remote with a 64-bit address of 0x0013A200 40522BAA and a 16-bit address of 0x7D84. If pin AD1/DIO1 is enabled as an analog input, AD2/DIO2 and DIO4 are enabled as a digital inputs (currently high), and AD3/DIO3 is enabled as a digital output (low) the IO sample would look like:

XBee Sensor Read Indicator

Frame Type: 0x94

When the module receives a sensor sample (from a Digi 1-wire sensor adapter), it is sent out the UART using this message type (when AO=0).

	Frame Fields		Offset	Example	Description
API Packet	Start Delimiter		0	0x7E	
	Length		MSB 1	0x00	Number of bytes between the length and the checksum
	Frame-specific Data		LSB 2	0x17	
		Frame Type	3	0x94	
		64-bit Source Address	MSB 4	0x00	64-bit address of sender
			5	0x13	
			6	0xA2	
			7	0x00	
			8	0x40	
			9	0x52	
			10	0x2B	
			LSB 11	0xAA	
		16-bit Source Network Address	MSB 12	0xDD	16-bit address of sender.
			LSB 13	0x6C	
		Receive Options	14	0x01	0x01 - Packet Acknowledged 0x02 - Packet was abroadcast packet
		1-Wire Sensors	15	0x03	0x01 = A/D Sensor Read 0x02 = Temperature Sensor Read 0x60 = Water present (module CD pin low)
		A/D Values	16	0x00	Indicates a two-byte value for each of four A/D sensors (A, B, C, D) Set to 0xFFFFFFFFFFFFFFFF if no A/Ds are found.
			17	0x02	
			18	0x00	
			19	0xCE	
			20	0x00	
			21	0xEA	
			22	0x00	
			23	0x52	
		Temperature Read	24	0x01	Indicates the two-byte value read from a digital thermometer if present. Set to 0xFFFF if not found.
			25	0x6A	
	Checksum		26	0x8B	0xFF - the 0x8 bit sum of bytes from offset 3 to this byte.

Example: Suppose a 1-wire sensor sample is received from a device with a 64-bit address of 0x0013A200 40522BAA and a 16-bit address of 0xDD6C. If the sensor sample was taken from a 1-wire humidity sensor, the API frame could look like this (if AO=0):

For convenience, let's label the A/D and temperature readings as AD0, AD1, AD2, AD3, and T. Using the data in this example:

AD0 = 0x0002

AD1 = 0x00CE

AD2 = 0x00EA

AD3 = 0x0052

T = 0x016A

To convert these to temperature and humidity values, the following equations should be used.

Temperature (°C) = (T / 16), for T < 2048
= - (T & 0x7FF) / 16, for T >= 2048

Vsupply = (AD2 * 5.1) / 255

Voutput = (AD3 * 5.1) / 255

Relative Humidity = ((Voutput / Vsupply) - 0.16) / (0.0062)

True Humidity = Relative Humidity / (1.0546 - (0.00216 * Temperature (°C)))

Looking at the sample data, we have:

Vsupply = (234 * 5.1 / 255) = 4.68

Voutput = (82 * 5.1 / 255) = 1.64

Temperature = (362 / 16) = 22.625°C

Relative H = (161.2903 * ((1.64/4.68) - 0.16)) = 161.2903 * (0.19043) = 30.71%

True H = (30.71 / (1.0546 - (0.00216 * 22.625))) = (30.71 / 1.00573) = 30.54%

Node Identification Indicator

Frame Type: 0x95

This frame is received when a module transmits a node identification message to identify itself (when AO=0). The data portion of this frame is similar to a network discovery response frame (see ND command).

	Frame Fields		Offset	Example	Description
API Packet	Start Delimiter		0	0x7E	
	Length		MSB 1	0x00	Number of bytes between the length and the checksum
			LSB 2	0x20	
	Frame-specific Data	Frame Type	3	0x95	
		64-bit Source Address	MSB 4	0x00	64-bit address of sender
			5	0x13	
			6	0xA2	
			7	0x00	
			8	0x40	
			9	0x52	
			10	0x2B	
			LSB 11	0xAA	
		16-bit Source Network Address	MSB 12	0x7D	16-bit address of sender.
			LSB 13	0x84	
		Receive Options	14	0x02	0x01 - Packet Acknowledged 0x02 - Packet was abroadcast packet
		Source 16-bit address	15	0x7D	Set to the 16-bit network address of the remote. Set to 0xFFFE if unknown.
			16	0x84	
		64-bit Network address	17	0x00	Indicates the 64-bit address of the remote module that transmitted the node identification frame.
			18	0x13	
			19	0xA2	
			20	0x00	
			21	0x40	
			22	0x52	
			23	0x2B	
			24	0xAA	
		NI String	25	0x20	Node identifier string on the remote device. The NI-String is terminated with a NULL byte (0x00).
			26	0x00	
		Parent 16-bit address	27	0xFF	Indicates the 16-bit address of the remote's parent or 0xFFFE if the remote has no parent.
			28	0xFE	
		Device Type	29	0x01	0 = Coordinator 1 = Router 2 = End Device
		Source Event	30	0x01	1 = Frame sent by node identification pushbutton event (see D0 command) 2 = Frame sent after joining event occurred (see JN command). 3 = Frame sent after power cycle event occurred (see JN command).
		Digi Profile ID	31	0xC1	Set to Digi's application profile ID.
			32	0x05	
		Manufacturer ID	33	0x10	Set to Digi's Manufacturer ID.
			34	0x1E	
	Checksum		35	0x1B	0xFF - the 8 bit sum of bytes from offset 3 to this byte.

Example: If the commissioning push button is pressed on a remote router device with 64-bit address 0x0013A200 40522BAA, 16-bit address 0x7D84, and default NI string, the following node identification indicator would be received.

Remote Command Response

Frame Type: 0x97

If a module receives a remote command response RF data frame in response to a Remote AT Command Request, the module will send a Remote AT Command Response message out the UART. Some commands may send back multiple frames--for example, Node Discover (ND) command.

	Frame Fields		Offset	Example	Description
API Packet	Start Delimiter		0	0x7E	
	Length		MSB 1	0x00	Number of bytes between the length and the checksum
			LSB 2	0x13	
	Frame-specific Data	Frame Type	3	0x97	
		Frame ID	4	0x55	This is the same value passed in to the request. .
		64-bit Source (remote) Address	MSB 5	0x00	The address of the remote radio returning this response.
			6	0x13	
			7	0xA2	
			8	0x00	
			9	0x40	
			10	0x52	
			11	0x2B	
			LSB 12	0xAA	
		16-bit Source (remote) Address	MSB 13	0x7D	Set to the 16-bit network address of the remote. Set to 0xFFFE if unknown.
			LSB 14	0x84	
		AT Commands	15	0x53	Name of the command
			16	0x4C	
		Command Status	17	0x00	0 = OK 1 = ERROR 2 = Invalid Command 3 = Invalid Parameter
		Command Data	18	0x40	Register data in binary format. If the register was set, then this field is not returned.
			19	0x52	
			20	0x2B	
			21	0xAA	
	Checksum		22	0xF0	0xFF - the 8 bit sum of bytes from offset 3 to this byte.

Example: If a remote command is sent to a remote device with 64-bit address 0x0013A200 40522BAA and 16-bit address 0x7D84 to query the SL command, and if the frame ID=0x55, the response would look like:

Over-the-Air Firmware Update Status

Frame Type: 0xA0

The Over-the-Air Firmware Update Status frame provides a status indication of a firmware update transmission attempt. If a query command (0x01 0x51) is sent to a target with a 64-bit address of 0x0013A200 40522BAA through an updater with 64-bit address 0x0013A200403E0750 and 16-bit address 0x0000, the following is the expected response.

API Packet	Start Delimiter		0	0x7E	
	Length		MSB 1	0x00	Number of bytes between the length and the checksum
			LSB 2	0x16	
	Frame-specific Data	Frame Type	3	0xA0	
		64-bit Source (remote) Address	MSB 4	0x00	The address of the remote radio returning this response.
			5	0x13	
			6	0xA2	
			7	0x00	
			8	0x40	
			9	0x3E	
			10	0x07	
			LSB 11	0x50	
		16-bit Destination Address	12	0x00	16-bit address of the updater device
			13	0x00	
		Receive Options	14	0x01	0x01 - Packet Acknowledged. 0x02 - Packet was a broadcast.
		Bootloader Message Type	15	0x52	0x06 - ACK 0x15 - NACK 0x40 - No Mac ACK 0x51 - Query (received if the bootloader is not active on the target) 0x52 - Query Response
		Block Number	16	0x00	Block number used in the update request. Set to 0 if not applicable.
		64-bit Target Address	17	0x00	64-bit Address of remote device that is being updated (target).
			18	0x13	
			19	0xA2	
			20	0x00	
			21	0x40	
			22	0x52	
			23	0x2B	
			24	0xAA	
	Checksum		25	0x66	0xFF - the 8 bit sum of bytes from offset 3 to this byte.

If a query request returns a 0x15 (NACK) status, the target is likely waiting for a firmware update image. If no messages are sent to it for about 75 seconds, the target will timeout and accept new query messages.

If a query returns a 0x51 (QUERY) status, then the target's bootloader is not active and will not respond to query messages.

Route Record Indicator

Frame Type: 0xA1

The route record indicator is received whenever a device sends a ZigBee route record command. This is used with many-to-one routing to create source routes for devices in a network.

	Frame Fields		Offset	Example	Description
API Packet	Start Delimiter		0	0x7E	
	Length		MSB 1	0x00	Number of bytes between the length and the checksum
			LSB 2	0x13	
	Frame-specific Data	Frame Type	3	0xA1	
		64-bit Source Address	MSB 4	0x00	64-bit address of the device that initiated the route record.
			5	0x13	
			6	0xA2	
			7	0x00	
			8	0x40	
			9	0x40	
			10	0x11	
			LSB 11	0x22	
		Source (updater) 16-bit Address	12	0x33	16-bit address of the device that initiated the route record.
			13	0x44	
		Receive Options	14	0x01	0x01 - Packet Acknowledged. 0x02 - Packet was a broadcast.
		Number of Addresses	15	0x03	The number of addresses in the source route (excluding source and destination).
		Address 1	16	0xEE	(neighbor of destination)
			17	0xFF	
		Address 2 (closer hop	18	0xCC	Address of intermediate hop
			19	0xDD	
		Address n (neighbor of source)	20	0xAA	Two bytes per 16-bit address.
			21	0xBB	
	Checksum		22	0x80	0xFF - the 8 bit sum of bytes from offset 3 to this byte.

Example: Suppose device E sends a route record that traverses multiple hops en route to data collector device A as shown below.

A B C D E

If device E has the 64-bit and 16-bit addresses of 0x0013A200 40401122 and 0x3344, and if devices B, C, and D have the following 16-bit addresses:

B = 0xAABB

C = 0xCCDD

D = 0xEEFF

The data collector will send the following API frame out its UART.

Sending ZigBee Device Objects (ZDO) Commands with the API

ZigBee Device Objects (ZDOs) are defined in the ZigBee Specification as part of the ZigBee Device Profile. These objects provide functionality to manage and map out the ZigBee network and to discover services on ZigBee devices. ZDOs are typically required when developing a ZigBee product that will interoperate in a public profile such as home automation or smart energy, or when communicating with ZigBee devices from other vendors. The ZDO can also be used to perform several management functions such as frequency agility (energy detect and channel changes - Mgmt Network Update Request), discovering routes (Mgmt Routing Request) and neighbors (Mgmt LQI Request), and managing device connectivity (Mgmt Leave and Permit Join Request).

The following table shows some of the more prominent ZDOs with their respective cluster identifier. Each ZDO command has a defined payload. See the "ZigBee Device Profile" section of the ZigBee Specification for details

ZDO Command	Cluster ID
Network Address Request	0x0000
IEEE Address Request	0x0001
Node Descriptor Request	0x0002
Simple Descriptor Request	0x0004
Active Endpoints Request	0x0005
Match Descriptor Request	0x0006
Mgmt LQI Request	0x0031
Mgmt Routing Request	0x0032
Mgmt Leave Request	0x0034
Mgmt Permit Joining Request	0x0036
Mgmt Network Update Request	0x0038

The Explicit Transmit API frame (0x11) is used to send ZigBee Device Objects commands to devices in the network. Sending ZDO commands with the Explicit Transmit API frame requires some formatting of the data payload field.

When sending a ZDO command with the API, all multiple byte values in the ZDO command (API payload) (i.e. u16, u32, 64-bit addresses) must be sent in little endian byte order for the command to be executed correctly on a remote device.

For an API XBee to receive ZDO responses, the AO command must be set to 1 to enable the explicit receive API frame.

The following table shows how the Explicit API frame can be used to send an "Active Endpoints" request to discover the active endpoints on a device with a 16-bit address of 0x1234.

	Frame Fields			Offset	Example	Description
API Packet	Start Delimiter			0	0x7E	
	Length			MSB 1	0x00	Number of bytes between the length and the checksum
	Frame-specific Data			LSB 2	0x17	
		Frame Type		3	0x11	
		Frame ID		4	0x01	Identifies the UART data frame for the host to correlate with a subsequent transmit status. If set to 0, no transmit status frame will be sent out the UART.
		64-bit Destination Address		MSB 5	0x00	64-bit address of the destination device (big endian byte order). For unicast transmissions, set to the 64-bit address of the destination device, or to 0x0000000000000000 to send a unicast to the coordinator. Set to 0x000000000000FFFF for broadcast.
				6	0x00	
				7	0x00	
				8	0x00	
				9	0x00	
				10	0x00	
				11	0x00	
				12	0x00	
		16-bit Destination Network Address		MSB 13	0xFF	16-bit address of the destination device (big endian byte order). Set to 0xFFFE for broadcast, or if the 16-bit address is unknown.
				LSB 14	0xFE	
		Source Endpoint		15	0x00	Set to 0x00 for ZDO transmissions (endpoint 0 is the ZDO endpoint).
		Destination Endpoint		16	0x00	Set to 0x00 for ZDO transmissions (endpoint 0 is the ZDO endpoint).
		Cluster ID		MSB 17	0x00	Set to the cluster ID that corresponds to the ZDO command being sent. 0x0005 = Active Endpoints Request
				LSB 18	0x00	
		Profile ID		MSB 19	0x05	Set to 0x0000 for ZDO transmissions (Profile ID 0x0000 is the ZigBee Device Profile that supports ZDOs).
				LSB 20	0x00	
		Broadcast Radius		21	0x00	Sets the maximum number of hops a broadcast transmission can traverse. If set to 0, the transmission radius will be set to the network maximum hops value..
		Transmit Options		22	0x00	All bits must be set to 0.
		Data Payload	Transaction Sequence Number	23	0x01	The required payload for a ZDO command. All multi-byte ZDO parameter values (u16, u32, 64-bit address) must be sent in little endian byte order. The Active Endpoints Request includes the following payload: [16-bit NwkAddrOfInterest] Note the 16-bit address in the API example (0x1234) is sent in little endian byte order (0x3412).
			ZDO Payload	24	0x34	
				25	0x12	
	Checksum			26	0xA6	0xFF minus the 8 bit sum of bytes from offset 3 to this byte.

Sending ZigBee Cluster Library (ZCL) Commands with the API

The ZigBee Cluster Library defines a set of attributes and commands (clusters) that can be supported in multiple ZigBee profiles. The ZCL commands are typically required when developing a ZigBee product that will interoperate in a public profile such as home automation or smart energy, or when communicating with ZigBee devices from other vendors. Applications that are not designed for a public profile or for interoperability applications can skip this section.

The following table shows some prominent clusters with their respective attributes and commands.

Cluster (Cluster ID)	Attributes (Attribute ID)	Cluster ID
Basic (0x0000)	Application Version (0x0001) Hardware Version (0x0003) Model Identifier (0x0005)	-Reset to defaults (0x00)
Identify (0x0003)	Identify Time (0x0000)	Identify (0x00) Identify Query (0x01)
Time (0x000A)	Time (0x0000) Time Status (0x0001) Time Zone (0x0002)	
Thermostat (0x0201)	Local Temperature (0x0000) Occupancy (0x0002)	-Setpoint raise / lower (0x00)

The ZCL defines a number of profile-wide commands that can be supported on any profile, also known as general commands. These commands include the following.

Command (Command ID)	Description
Read Attributes (0x00)	Used to read one or more attributes on a remote device.
Read Attributes Response (0x01)	Generated in response to a read attributes command.
Write Attributes (0x02)	Used to change one or more attributes on a remote device.
Write Attributes Response (0x04)	Sent in response to a write attributes command.
Configure Reporting (0x06)	Used to configure a device to automatically report on the values of one or more of its attributes.
Report Attributes (0x0A)	Used to report attributes when report conditions have been satisfied.
Discover Attributes (0x0C)	Used to discover the attribute identifiers on a remote device.
Discover Attributes Response (0x0D)	Sent in response to a discover attributes command.

The Explicit Transmit API frame (0x11) is used to send ZCL commands to devices in the network. Sending ZCL commands with the Explicit Transmit API frame requires some formatting of the data payload field.

When sending a ZCL command with the API, all multiple byte values in the ZCL command (API Payload) (i.e. u16, u32, 64-bit addresses) must be sent in little endian byte order for the command to be executed correctly on a remote device.

Note: When sending ZCL commands, the AO command should be set to 1 to enable the explicit receive API frame. This will provide indication of the source 64- and 16-bit addresses, cluster ID, profile ID, and endpoint information for each received packet. This information is required to properly decode received data.

The following table shows how the Explicit API frame can be used to read the hardware version attribute from a device with a 64-bit address of 0x0013A200 40401234 (unknown 16-bit address). This example uses arbitrary source and destination endpoints. Recall the hardware version attribute (attribute ID 0x0003) is part of the basic cluster (cluster ID 0x0000). The Read Attribute general command ID is 0x00.

	Frame Fields				Offset	Example	Description
API Packet	Start Delimiter				0	0x7E	
	Length				MSB 1	0x00	Number of bytes between the length and the checksum
					LSB 2	0x19	
	Frame-specific Data	Frame Type			3	0x11	
		Frame ID			4	0x01	Identifies the UART data frame for the host to correlate with a subsequent transmit status. If set to 0, no transmit status frame will be sent out the UART.
		64-bit Destination Address			MSB 5	0x00	64-bit address of the destination device (big endian byte order). For unicast transmissions, set to the 64-bit address of the destination device, or to 0x0000000000000000 to send a unicast to the coordinator. Set to 0x000000000000FFFF for broadcast.
					6	0x13	
					7	0xA2	
					8	0x00	
					9	0x40	
					10	0x40	
					11	0x12	
					12	0x34	
		16-bit Destination Network Address			MSB 13	0xFF	16-bit address of the destination device (big endian byte order). Set to 0xFFFE for broadcast, or if the 16-bit address is unknown.
					LSB 14	0xFE	
		Source Endpoint			15	0x41	Set to the source endpoint on the sending device. (0x41 arbitrarily selected).
		Destination Endpoint			16	0x42	Set to the destination endpoint on the remote device. (0x42 arbitrarily selected)
		Cluster ID			MSB 17	0x00	Set to the cluster ID that corresponds to the ZCL command being sent. 0x0000 = Basic Cluster
					LSB 18	0x00	
		Profile ID			MSB 19	0x00	Set to the profile ID supported on the device. (0xD123 arbitrarily selected).
					LSB 20	0xD1	
		Broadcast Radius			21	0x00	Sets the maximum number of hops a broadcast transmission can traverse. If set to 0, the transmission radius will be set to the network maximum hops value..
		Transmit Options			22	0x00	All bits must be set to 0.
		Data Payload	ZCL Frame Header	Frame Control	23	0x00	Bitfield that defines the command type and other relevant information in the ZCL command. See the ZCL specification for details.
				Transaction Sequence Number	24	0x01	A sequence number used to correlate a ZCL command with a ZCL response. (The hardware version response will include this byte as a sequence number in the response.) The value 0x01 was arbitrarily selected.
				Command ID	25	0x00	Since the frame control "frame type" bits are 00, this byte specifies a general command. Command ID 0x00 is a Read Attributes command.
			ZCL Payload	Attribute ID	26	0x03	The payload for a "Read Attributes" command is a list of Attribute Identifiers that are being read. Note the 16-bit Attribute ID (0x0003) is sent in little endian byte order (0x0300). All multi-byte ZCL header and payload values must be sent in little endian byte order.
					27	0x00	
	Checksum				28	0xFA	0xFF minus the 8 bit sum of bytes from offset 3 to this byte.

In the above example, the Frame Control was constructed as follows:

Name	Bits	Example Value Description
Frame Type	0-1	00 - Command acts across the entire profile
Manufacturer Specific	2	0 - The manufacturer code field is omitted from the ZCL Frame Header.
Direction	3	0 - The command is being sent from the client side to the server side.
Disable Default Response	4	0 - Default response not disabled
Resvered	5-7	Set to 0.

See the ZigBee Cluster Library specification for details.

Sending Public Profile Commands with the API

Commands in public profiles such as Smart Energy and Home Automation can be sent with the XBee API using the Explicit Transmit API frame (0x11). Sending public profile commands with the Explicit Transmit API frame requires some formatting of the data payload field. Most of the public profile commands fit into the ZigBee Cluster Library (ZCL) architecture as described in the previous section.

The following table shows how the Explicit API frame can be used to send a demand response and load control message (cluster ID 0x701) in the smart energy profile (profile ID 0x0109) in the revision 14 Smart Energy specification. The message will be a "Load Control Event" (command ID 0x00) and will be sent to a device with 64-bit address of 0x0013A200 40401234 with a 16-bit address of 0x5678. The event will start a load control event for water heaters and smart appliances, for a duration of 1 minute, starting immediately.

Note: When sending public profile commands, the AO command should be set to 1 to enable the explicit receive API frame. This will provide indication of the source 64- and 16-bit addresses, cluster ID, profile ID, and endpoint information for each received packet. This information is required to properly decode received data.

	Frame Fields				Offset	Example	Description
API Packet	Start Delimiter				0	0x7E	
	Length				MSB 1	0x00	Number of bytes between the length and the checksum
					LSB 2	0x19	
	Frame-specific Data	Frame Type			3	0x11	
		Frame ID			4	0x01	Identifies the UART data frame for the host to correlate with a subsequent transmit status. If set to 0, no transmit status frame will be sent out the UART.
		64-bit Destination Address			MSB 5	0x00	64-bit address of the destination device (big endian byte order). For unicast transmissions, set to the 64-bit address of the destination device, or to 0x0000000000000000 to send a unicast to the coordinator. Set to 0x000000000000FFFF for broadcast.
					6	0x13	
					7	0xA2	
					8	0x00	
					9	0x40	
					10	0x40	
					11	0x12	
					12	0x34	
		16-bit Destination Network Address			MSB 13	0x56	16-bit address of the destination device (big endian byte order). Set to 0xFFFE for broadcast, or if the 16-bit address is unknown.
					LSB 14	0x78	
		Source Endpoint			15	0x41	Set to the source endpoint on the sending device. (0x41 arbitrarily selected).
		Destination Endpoint			16	0x42	Set to the destination endpoint on the remote device. (0x42 arbitrarily selected)
		Cluster ID			MSB 17	0x07	Set to the cluster ID that corresponds to the ZCL command being sent. 0x0701 = Demand response and load control cluster ID
					LSB 18	0x01	
		Profile ID			MSB 19	0x01	Set to the profile ID supported on the device. 0x0109 = Smart Energy profile ID.
					LSB 20	0x09	
		Broadcast Radius			21	0x00	Sets the maximum number of hops a broadcast transmission can traverse. If set to 0, the transmission radius will be set to the network maximum hops value..
		Transmit Options			22	0x00	All bits must be set to 0.
		Data Payload	ZCL Frame Header	Frame Control	23	0x09	Bitfield that defines the command type and other relevant information in the ZCL command. See the ZCL specification for details.
				Transaction Sequence Number	24	0x01	A sequence number used to correlate a ZCL command with a ZCL response. (The hardware version response will include this byte as a sequence number in the response.) The value 0x01 was arbitrarily selected.
					25	0x00	Since the frame control "frame type" bits are 01, this byte specifies a cluster-specific command. Command ID 0x00 in the Demand Response and Load Control cluster is a Load Control Event command. (See Smart Energy specification.)

Frame Fields				Offset	Example	Description
		ZCL Payload - Load Control Event Data	Issuer Event ID	26	0x78	4-byte unique identifier. Note the 4-byte ID is sent in little endian byte order (0x78563412). The event ID in this example (0x12345678) was arbitrarily selected.
				27	0x56	
				28	0x34	
				29	0x12	
			Device Class	30	0x14	to apply the load control event. A bit value of 0x0014 enables smart appliances and water heaters. Note the 2-byte bit field value is sent in little endian byte order.
				31	0x00	
			Utility Enrollment Group	32	0x00	Used to identify sub-groups of devices in the device-class. 0x00 addresses all groups.
			Start Time	33	0x00	UTC timestamp representing when the event should start. A value of 0x00000000 indicates "now".
				34	0x00	
				35	0x00	
				36	0x00	
			Duration in Minutes	37	0x01	This 2-byte value must be sent in little endian byte order.
				38	0x00	
			Criticality Level	39	0x04	Indicates the criticality level of the event. In this example, the level is "voluntary".
			Cooling Temperature	40	0xFF	Requested offset to apply to the normal cooling set point. A value of 0xFF indicates the temperature offset value is not used.
			Heating Temperature Offset	41	0xFF	Requested offset to apply to the normal heating set point. A value of 0xFF indicates the temperature offset value is not used.
			Cooling Temperature Set Point	42	0x00	Requested cooling set point in 0.01 degrees Celsius. A value of 0x8000 means the set point field is not used in this event. Note the 0x80000 is sent in little endian byte order.
				43	0x80	
			Heating Temperature Set Point	44	0x00	Requested heating set point in 0.01 degrees Celsius. A value of 0x8000 means the set point field is not used in this event. Note the 0x80000 is sent in little endian byte order.
				45	0x80	
			Average Load Adjustment Percentage	46	0x80	Maximum energy usage limit. A value of 0x80 indicates the field is not used.
			Duty Cycle	47	0xFF	Defines the maximum "On" duty cycle. A value of 0xFF indicates the duty cycle is not used in this event.
			Duty Cycle Event Control	48	0x00	A bitmap describing event options.
Checksum				49	0x5B	0xFF minus the 8 bit sum of bytes from offset 3 to this byte.

In the above example, the Frame Control was constructed as follows:

Name	Bits	Example Value Description
Frame Type	0-1	01- Command acts across the entire profile
Manufacturer Specific	2	0 - The manufacturer code field is omitted from the ZCL Frame Header.

Name	Bits	Example Value Description
Direction	3	0 - The command is being sent from the client side to the server side.
Disable Default Response	4	0 - Default response not disabled
Resvered	5-7	Set to 0.

10. XBee Command Reference Tables

Addressing

Table 10-07. Addressing Commands)

AT Command	Name and Description	Node Type[1]	Parameter Range	Default
DH	**Destination Address High**.Set/Get the upper 32 bits of the 64-bit destination address. When combined with DL, it defines the 64-bit destination address for data transmission. Special definitions for DH and DL include 0x000000000000FFFF (broadcast) and 0x0000000000000000 (coordinator).	CRE	0 - 0xFFFFFFFF	0
DL	**Destination Address Low**. Set/Get the lower 32 bits of the 64-bit destination address. When combined with DH, it defines the 64-bit destination address for data transmissions. Special definitions for DH and DL include 0x000000000000FFFF (broadcast) and 0x0000000000000000 (coordinator).	CRE	0 - 0xFFFFFFFF	0xFFFF(Coordinator) 0 (Router/End Device)
MY	**16-bit Network Address**. Read the 16-bit network address of the module. A value of 0xFFFE means the module has not joined a ZigBee network	CRE	0 - 0xFFFE [read-only]	0xFFFE
MP	1**6-bit Parent Network Address**. Read the 16-bit network address of the module's parent. A value of 0xFFFE means the module does not have a parent.	E	0 - 0xFFFE [read-only]	0xFFFE
NC	**Number of Remaining Children**. Read the number of end device children that can join the device. If NC returns 0, then the device cannot allow any more end device children to join.	CR	0 - MAX_CHILDREN (maximum varies)	read-only
SH	**Serial Number High**. Read the high 32 bits of the module's unique 64-bit address.	CRE	0 - 0xFFFFFFFF [read-only]	factory-set
SL	**Serial Number Low**. Read the low 32 bits of the module's unique 64-bit address.	CRE	0 - 0xFFFFFFFF [read-only]	factory-set
NI	**Node Identifier.** Stores a string identifier. The register only accepts printable ASCII data. In AT Command Mode, a string can not start with a space. A carriage return ends the command. Command will automatically end when maximum bytes for the string have been entered. This string is returned as part of the ND (Node Discover) command. This identifier is also used with the DN (Destination Node) command.	CRE	20-Byte printable ASCII string	ASCII space character (0x20)
DD	**Device Type Identifier**. Stores a device type value. This value can be used to differentiate different XBee-based devices. Digi reserves the range 0 - 0xFFFFFF.	CRE	0 - 0xFFFFFFFF	0x30000
SE	**Source Endpoint**. Set/read the ZigBee application layer source endpoint value. This value will be used as the source endpoint for all data transmissions. SE is only supported in AT firmware.The default value 0xE8 (Data endpoint) is the Digi data endpoint	CRE	0 - 0xFF	0xE8
DE	**Destination Endpoint**. Set/read Zigbee application layer destination ID value. This value will be used as the destination endpoint all data transmissions. DE is only supported in AT firmware.The default value (0xE8) is the Digi data endpoint.	CRE	0 - 0xFF	0xE8
CI	**Cluster Identifier**. Set/read Zigbee application layer cluster ID value. This value will be used as the cluster ID for all data transmissions. CI is only supported in AT firmware.The default value0x11 (Transparent data cluster ID).	CRE	0 - 0xFFFF	0x11
NP	**Maximum RF Payload Bytes**. This value returns the maximum number of RF payload bytes that can be sent in a unicast transmission. If many-to-one and source routing are used (AR < 0xFF), or if APS security is used on a transmission, the maximum payload size is reduced further. Note: NP returns a hexadecimal value. (i.e. if NP returns 0x54, this is equivalent to 84 bytes)	CRE	0 - 0xFFFF	[read-only]

Node types that support the command: C=Coordinator, R=Router, E=End Device

Networking

Table 10-08. Networking Commands

AT Command	Name and Description	Node Type[1]	Parameter Range	Default
CH	**Operating Channel**. Read the channel number used for transmitting and receiving between RF modules. Uses 802.15.4 channel numbers. A value of 0 means the device has not joined a PAN and is not operating on any channel.	CRE	0, 0x0B - 0x1A (XBee) 0, 0x0B - 0x18 (XBee-PRO)	[read-only]
ID	**Extended PAN ID**. Set/read the 64-bit extended PAN ID. If set to 0, the coordinator will select a random extended PAN ID, and the router / end device will join any extended PAN ID. Changes to ID should be written to non-volatile memory using the WR command to preserve the ID setting if a power cycle occurs.	CRE	0 - 0xFFFFFFFFFFFFFFFF	0

Table 10-08. Networking Commands

AT Command	Name and Description	Node Type[1]	Parameter Range	Default
OP	**Operating Extended PAN ID**. Read the 64-bit extended PAN ID. The OP value reflects the operating extended PAN ID that the module is running on. If ID > 0, OP will equal ID.	CRE	0x01 - 0xFFFFFFFFFFFFFFFF	[read-only]
NH	**Maximum Unicast Hops**. Set / read the maximum hops limit. This limit sets the maximum broadcast hops value (BH) and determines the unicast timeout. The timeout is computed as (50 * NH) + 100 ms. The default unicast timeout of 1.6 seconds (NH=0x1E) is enough time for data and the acknowledgment to traverse about 8 hops.	CRE	0 - 0xFF	0x1E
BH	**Broadcast Hops.** Set/Read the maximum number of hops for each broadcast data transmission. Setting this to 0 will use the maximum number of hops.	CRE	0 - 0x20	0
OI	**Operating 16-bit PAN ID**. Read the 16-bit PAN ID. The OI value reflects the actual 16-bit PAN ID the module is running on. .	CRE	0 - 0xFFFF	[read-only]
NT	**Node Discovery Timeout**. Set/Read the node discovery timeout. When the network discovery (ND) command is issued, the NT value is included in the transmission to provide all remote devices with a response timeout. Remote devices wait a random time, less than NT, before sending their response.	CRE	0x20 - 0xFF [x 100 msec]	0x3C (60d)
NO	**Network Discovery options.** Set/Read the options value for the network discovery command. The options bitfield value can change the behavior of the ND (network discovery) command and/or change what optional values are returned in any received ND responses or API node identification frames. Options include: 0x01 = Append DD value (to ND responses or API node identification frames) 002 = Local device sends ND response frame when ND is issued.	CRE	0 - 0x03 [bitfield]	0
SC	**Scan Channels**. Set/Read the list of channels to scan. ***Coordinator*** - Bit field list of channels to choose from prior to starting network. ***Router/End Device*** - Bit field list of channels that will be scanned to find a Coordinator/ Router to join. Changes to SC should be written using WR command to preserve the SC setting if a power cycle occurs. Bit (Channel): 0 (0x0B) 4 (0x0F) 8 (0x13) 12 (0x17) 1 (0x0C) 5 (0x10) 9 (0x14) 13 (0x18) 2 (0x0D) 6 (0x11) 10 (0x15) 14 (0x19) 3 (0x0E) 7 (0x12) 11 (0x16) 15 (0x1A)	CRE	**XBee** 1 - 0xFFFF [bitfield] **XBee-PRO** 1 - 0x1FFE [bitfield] (bits 14, 15 not allowed)	0x3FFF.
SD	**Scan Duration**. Set/Read the scan duration exponent. Changes to SD should be written using WR command. ***Coordinator*** - Duration of the Active and Energy Scans (on each channel) that are used to determine an acceptable channel and Pan ID for the Coordinator to startup on. ***Router / End Device*** - Duration of Active Scan (on each channel) used to locate an available Coordinator / Router to join during Association. Scan Time is measured as:(# Channels to Scan) * (2 ^ SD) * 15.36ms - The number of channels to scan is determined by the SC parameter. The XBee can scan up to 16 channels (SC = 0xFFFF). Sample Scan Duration times (13 channel scan): If SD = 0, time = 0.200 sec SD = 2, time = 0.799 sec SD = 4, time = 3.190 sec SD = 6, time = 12.780 sec **Note**: SD influences the time the MAC listens for beacons or runs an energy scan on a given channel. The SD time is not a good estimate of the router/end device joining time requirements. ZigBee joining adds additional overhead including beacon processing on each channel, sending a join request, etc. that extend the actual joining time.	CRE	0 - 7 [exponent]	3
ZS	**ZigBee Stack Profile**. Set / read the ZigBee stack profile value. This must be set the same on all devices that should join the same network.	CRE	0 - 2	0
NJ	**Node Join Time**. Set/Read the time that a Coordinator/Router allows nodes to join. This value can be changed at run time without requiring a Coordinator or Router to restart. The time starts once the Coordinator or Router has started. The timer is reset on power-cycle or when NJ changes.	CR	0 - 0xFF [x 1 sec]	0xFF (always allows joining)
JV	**Channel Verification**. Set/Read the channel verification parameter. If JV=1, a router will verify the coordinator is on its operating channel when joining or coming up from a power cycle. If a coordinator is not detected, the router will leave its current channel and attempt to join a new PAN. If JV=0, the router will continue operating on its current channel even if a coordinator is not detected.	R	0 - Channel verification disabled 1 - Channel verification enabled	0
JN	**Join Notification**. Set / read the join notification setting. If enabled, the module will transmit a broadcast node identification packet on power up and when joining. This action blinks the Associate LED rapidly on all devices that receive the transmission, and sends an API frame out the UART of API devices. This feature should be disabled for large networks to prevent excessive broadcasts.	RE	0 - 1	0
AR	**Aggregate Routing Notification**. Set/read time between consecutive aggregate route broadcast messages. If used, AR should be set on only one device to enable many-to-one routing to the device. Setting AR to 0 only sends one broadcast	CR	0 - 0xFF	0xFF

Table 10-08. Networking Commands

AT Command	Name and Description	Node Type[1]	Parameter Range	Default
AI	**Association Indication**. Read information regarding last node join request: 0x00 - Successful completion - Coordinator started or Router/End Device found and joined with a parent. 0xAB - Attempted to join a device that did not respond. 0xAC - Secure join error - network security key received unsecured 0xAD - Secure join error - network security key not received 0xAF - Secure join error - joining device does not have the right preconfigured link key 0x21 - Scan found no PANs 0x22 - Scan found no valid PANs based on current SC and ID settings 0x23 - Valid Coordinator or Routers found, but they are not allowing joining (NJ expired) 0x27 - Node Joining attempt failed (typically due to incompatible security settings) 0x2A - Coordinator Start attempt failed‘ 0xFF - Scanning for a Parent 0x2B - Checking for an existing coordinator	CRE	0 - 0xFF [read-only]	--

Security

Table 10-09. Security Commands

AT Command	Name and Description	Node Type[1]	Parameter Range	Default
EE	**Encryption Enable**. Set/Read the encryption enable setting.	CRE	0 - Encryption disabled 1 - Encryption enabled	0
EO	**Encryption Options.** Configure options for encryption. Unused option bits should be set to 0. Options include: 0x01 - Send the security key unsecured over-the-air during joins 0x02 - Use trust center (coordinator only)r	CRE	0 - 0xFF	
NK	**Network Encryption Key**. Set the 128-bit AES network encryption key. This command is write-only; NK cannot be read. If set to 0 (default), the module will select a random network key.	C	128-bit value	0
KY	**Link Key**. Set the 128-bit AES link key. This command is write only; KY cannot be read. Setting KY to 0 will cause the coordinator to transmit the network key in the clear to joining devices, and will cause joining devices to acquire the network key in the clear when joining.	CRE	128-bit value	0

RF Interfacing

Table 10-010.RF Interfacing Commands

AT Command	Name and Description	Node Type[1]	Parameter Range	Default
PL	**Power Level**. Select/Read the power level at which the RF module transmits conducted power.	CRE	**XBee** (boost mode disabled) 0 = -8 dBm 1 = -4 dBm 2 = -2 dBm 3 = 0 dBm 4 = +2 dBm **XBee-PRO** 4 = 17 dBm **XBee-PRO (International Variant)** 4 = 10dBm	4
PM	**Power Mode**. Set/read the power mode of the device. Enabling boost mode will improve the receive sensitivity by 1dB and increase the transmit power by 2dB Note: Enabling boost mode on the XBee-PRO will not affect the output power. Boost mode imposes a slight increase in current draw. See section 1.2 for details.	CRE	0-1, 0= -Boost mode disabled, 1= Boost mode enabled.	1
DB	**Received Signal Strength**. This command reports the received signal strength of the last received RF data packet. The DB command only indicates the signal strength of the last hop. It does not provide an accurate quality measurement for a multihop link. DB can be set to 0 to clear it. The DB command value is measured in -dBm. For example if DB returns 0x50, then the RSSI of the last packet received was -80dBm.	CRE	0 - 0xFF Observed range for XBee-PRO: 0x1A - 0x58 For XBee: 0x 1A - 0x5C	

1. Node types that support the command: C = Coordinator, R = Router, E = End Device

Serial Interfacing (I/O)

Table 10-011.Serial Interfacing Commands

AT Command	Name and Description	Node Type[1]	Parameter Range	Default
AP	API Enable. Enable API Mode. The AP command is only supported when using API firmware: 21xx (API coordinator), 23xx (API router), 29xx (API end device).	CRE	1 - 2 1 = API-enabled 2 = API-enabled (w/escaped control characters)	1
AO	**API Options**. Configure options for API. Current options select the type of receive API frame to send out the Uart for received RF data packets.	CRE	0 - Default receive API indicators enabled 1 - Explicit Rx data indicator API frame enabled (0x91)	0
BD	**Interface Data Rate**. Set/Read the serial interface data rate for communication between the module serial port and host. Any value above 0x07 will be interpreted as an actual baud rate. When a value above 0x07 is sent, the closest interface data rate represented by the number is stored in the BD register.	CRE	0x80 - 0xE1000 (non-standard rates up to 921kbps)	3
NB	**Serial Parity**. Set/Read the serial parity setting on the module.	CRE	0 = No parity 1 = Even parity 2 = Odd parity 3 = Mark parity	0
RO	**Packetization Timeout**. Set/Read number of character times of inter-character silence required before packetization. Set (RO=0) to transmit characters as they arrive instead of buffering them into one RF packet The RO command is only supported when using AT firmware: 20xx (AT coordinator), 22xx (AT router), 28xx (AT end device).	CRE	0 - 0xFF [x character times]	3
D7	**DIO7 Configuration**. Select/Read options for the DIO7 line of the RF module.	CRE	0 = Disabled 1 = CTS Flow Control 3 = Digital input 4 = Digital output, low 5 = Digital output, high 6 = RS-485 transmit enable (low enable) 7 = RS-485 transmit enable (high enable)	1
D6	**DIO6 Configuration.** Configure options for the DIO6 line of the RF module.	CRE	0 = Disabled 1 = RTS flow control 3 = Digital input 4 = Digital output, low 5 = Digital output, high	0

[1] Node types that support the command: C = Coordinator, R = Router, E = End Device

I/O Commands

Table 10-012.I/O Commands

AT Command	Name and Description	Node Type[1]	Parameter Range	Default
IR	**IO Sample Rate**. Set/Read the IO sample rate to enable periodic sampling. For periodic sampling to be enabled, IR must be set to a non-zero value, and at least one module pin must have analog or digital IO functionality enabled (see D0-D8, P0-P2 commands). The sample rate is measured in milliseconds.	CRE	0 - 0xFFFF (ms)	0
IC	I**O Digital Change Detection**. Set/Read the digital IO pins to monitor for changes in the IO state. IC works with the individual pin configuration commands (D0-D8, P0-P2). If a pin is enabled as a digital input/output, the IC command can be used to force an immediate IO sample transmission when the DIO state changes. IC is a bitmask that can be used to enable or disable edge detection on individual channels. Unused bits should be set to 0. Bit (IO pin): 0 (DIO0)4 (DIO4)8 (DIO8) 1 (DIO1) 5 (DIO5) 9 (DIO9) 2 (DIO2) 6 (DIO6) 10 (DIO10) 3 (DIO3) 7 (DIO7) 11 (DIO11)	CRE	: 0 - 0xFFFF	0
P0	**PWM0 Configuration**. Select/Read function for PWM0.	CRE	0 = Disabled 1 = RSSI PWM 3 - Digital input, monitored 4 - Digital output, default low 5 - Digital output, default high	1

Table 10-012.I/O Commands

AT Command	Name and Description	Node Type[1]	Parameter Range	Default
P1	**DIO11 Configuration**. Configure options for the DIO11 line of the RF module.	CRE	0 - Unmonitored digital input 3- Digital input, monitored 4- Digital output, default low 5- Digital output, default high	0
P2	**DIO12 Configuration**. Configure options for the DIO12 line of the RF module.	CRE	0 - Unmonitored digital input 3- Digital input, monitored 4- Digital output, default low 5- Digital output, default high	0
P3	**DIO13 Configuration**. Set/Read function for DIO13. This command is not yet supported.	CRE	0, 3-5 0 – Disabled 3 – Digital input 4 – Digital output, low 5 – Digital output, high	
D0	**AD0/DIO0 Configuration**. Select/Read function for AD0/DIO0.	CRE	1 - Commissioning button enabled 2 - Analog input, single ended 3 - Digital input 4 - Digital output, low 5 - Digital output, high	1
D1	**AD1/DIO1 Configuration**. Select/Read function for AD1/DIO1.	CRE	0, 2-5 0 – Disabled 2 - Analog input, single ended 3 – Digital input 4 – Digital output, low 5 – Digital output, high	0
D2	AD2/DIO2 Configuration. Select/Read function for AD2/DIO2.	CRE	0, 2-5 0 – Disabled 2 - Analog input, single ended 3 – Digital input 4 – Digital output, low 5 – Digital output, high	0
D3	**AD3/DIO3 Configuration**. Select/Read function for AD3/DIO3.	CRE	0, 2-5 0 – Disabled 2 - Analog input, single ended 3 – Digital input 4 – Digital output, low 5 – Digital output, high	0
D4	**DIO4 Configuration**. Select/Read function for DIO4.	CRE	0, 3-5 0 – Disabled 3 – Digital input 4 – Digital output, low 5 – Digital output, high	0
D5	**DIO5 Configuration**. Configure options for the DIO5 line of the RF module.	CRE	0 = Disabled 1 = Associated indication LED 3 = Digital input 4 = Digital output, default low 5 = Digital output, default high	1

Table 10-012.I/O Commands

AT Command	Name and Description	Node Type[1]	Parameter Range	Default
D8	**DIO8 Configuration**. Set/Read function for DIO8. This command is not yet supported.	CRE	0, 3-5 0 – Disabled 3 – Digital input 4 – Digital output, low 5 – Digital output, high	
LT	**Assoc LED Blink Time**. Set/Read the Associate LED blink time. If the Associate LED functionality is enabled (D5 command), this value determines the on and off blink times for the LED when the module has joined a network. If LT=0, the default blink rate will be used (500ms coordinator, 250ms router/end device). For all other LT values, LT is measured in 10ms.	CRE	0x14 - 0xFF (200 - 2550 ms)	0
PR	Set/read the bit field that configures the internal pull-up resistor status for the I/O lines. "1" specifies the pull-up resistor is enabled. "0" specifies no pullup.(30k pull-up resistors) Bits:" 0 - DIO4 (Pin 11) 1 - AD3 / DIO3 (Pin 17) 2 - AD2 / DIO2 (Pin 18) 3 - AD1 / DIO1 (Pin 19) 4 - AD0 / DIO0 (Pin 20) 5 - RTS / DIO6 (Pin 16) 6 - DTR / Sleep Request / DIO8 (Pin 9) 7 - DIN / Config (Pin 3) 8 - Associate / DIO5 (Pin 15) 9 - On/Sleep / DIO9 (Pin 13) 10 - DIO12 (Pin 4) 11 - PWM0 / RSSI / DIO10 (Pin 6) 12 - PWM1 / DIO11 (Pin 7)	CRE	0 - 0x1FFF	0 - 0x1FFF
RP	**RSSI PWM Timer.** Time RSSI signal will be output after last transmission. When RP = 0xFF, output will always be on.	CRE	0 - 0xFF [x 100 ms]	0x28 (40d)
CB	**Commissioning Pushbutton**. This command can be used to simulate commissioning button presses in software. The parameter value should be set to the number of button presses to be simulated. For example, sending the ATCB1 command will execute the action associated with 1 commissioning button press.	CRE		
%V	**Supply Voltage.** Reads the voltage on the Vcc pin. Divide the read value by 1024 A %V reading of 0x900 (2304 decimal) represents 2700mV or 2.70V.	R	-	-

Diagnostics

Table 10-013.Diagnostics Commands

AT Command	Name and Description	Node Type[1]	Parameter Range	Default
VR	**Firmware Version**. Read firmware version of the module. The firmware version returns 4 hexadecimal values (2 bytes) "ABCD". Digits ABC are the main release number and D is the revision number from the main release. "B" is a variant designator. XBee and XBee-PRO ZB modules return: 0x2xxx versions. XBee and XBee-PRO ZNet modules return: 0x1xxx versions. ZNet firmware is not compatible with ZB firmware.	CRE	0 - 0xFFFF [read-only]	Factory-set
HV	**Hardware Version**. Read the hardware version of the module.version of the module. This command can be used to distinguish among different hardware platforms. The upper byte returns a value that is unique to each module type. The lower byte indicates the hardware revision. XBee ZB and XBee ZNet modules return the following (hexadecimal) values: 0x19xx - XBee module 0x1Axx - XBee-PRO module	CRE	0 - 0xFFFF [read-only]	Factory-set

[1] Node types that support the command:C = Coordinator, R = Router, E = End Device

AT Command Options

Table 10-014.AT Command Options Commands

AT Command	Name and Description	Node Type[1]	Parameter Range	Default
CT	**Command Mode Timeout.** Set/Read the period of inactivity (no valid commands received) after which the RF module automatically exits AT Command Mode and returns to Idle Mode.	CRE	2 - 0x028F [x 100 ms]	0x64 (100d)
CN	**Exit Command Mode.** Explicitly exit the module from AT Command Mode.	CRE	--	--
GT	**Guard Times**. Set required period of silence before and after the Command Sequence Characters of the AT Command Mode Sequence (GT + CC + GT). The period of silence is used to prevent inadvertent entrance into AT Command Mode.	CRE	1 - 0x0CE4 [x 1 ms] (max of 3.3 decimal sec)	0x3E8 (1000d)
CC	**Command Sequence Character**. Set/Read the ASCII character value to be used between Guard Times of the AT Command Mode Sequence (GT + CC + GT). The AT Command Mode Sequence enters the RF module into AT Command Mode. The CC command is only supported when using AT firmware: 20xx (AT coordinator), 22xx (AT router), 28xx (AT end device).	CRE	0 - 0xFF	0x2B ('+' ASCII)

1. Node types that support the command: C = Coordinator, R = Router, E = End Device

Sleep Commands

Table 10-015.Sleep Commands

AT Command	Name and Description	Node Type[1]	Parameter Range	Default
SM	**Sleep Mode** Sets the sleep mode on the RF module	E	0-Sleep disabled 1-Pin sleep enabled 4-Cyclic sleep enabled 5 - Cyclic sleep, pin wake	0
SN	**Number of Sleep Periods.** Sets the number of sleep periods to not assert the On/Sleep pin on wakeup if no RF data is waiting for the end device. This command allows a host application to sleep for an extended time if no RF data is present	CRE	1 - 0xFFFF	1
SP	**Sleep Period.** This value determines how long the end device will sleep at a time, up to 28 seconds. (The sleep time can effectively be extended past 28 seconds using the SN command.) On the parent, this value determines how long the parent will buffer a message for the sleeping end device. It should be set at least equal to the longest SP time of any child end device.	CRE	0x20 - 0xAF0 x 10ms (Quarter second resolution)	0x20
ST	**Time Before Sleep** Sets the time before sleep timer on an end device.The timer is reset each time serial or RF data is received. Once the timer expires, an end device may enter low power operation. Applicable for cyclic sleep end devices only.	E	1 - 0xFFFE (x 1ms)	0x1388 (5 seconds)
SO Command	**Sleep Options**. Configure options for sleep. Unused option bits should be set to 0. Sleep options include: 0x02 - Always wake for ST time 0x04 - Sleep entire SN * SP time Sleep options should not be used for most applications. See Sleep Mode chapter for more information.	E	0 - 0xFF	0
WH	**Wake Host**. Set/Read the wake host timer value. If the wake host timer is set to a non-zero value, this timer specifies a time (in millisecond units) that the device should allow after waking from sleep before sending data out the UART or transmitting an IO sample. If serial characters are received, the WH timer is stopped immediately.	E	0 - 0xFFFF (x 1ms)	

Execution Commands

Where most AT commands set or query register values, execution commands cause an action to be executed on the module. Execution commands are executed immediately and do not require changes to be applied.l

Table 10-01. Execution Commands

AT Command	Name and Description	Node Type[1]	Parameter Range	Default
AC	**Apply Changes**. Applies changes to all command registers causing queued command register values to be applied. For example, changing the serial interface rate with the BD command will not change the UART interface rate until changes are applied with the AC command. The CN command and 0x08 API command frame also apply changes.	CRE	-	

Table 10-01. Execution Commands

AT Command	Name and Description	Node Type[1]	Parameter Range	Default
WR	**Write**. Write parameter values to non-volatile memory so that parameter modifications persist through subsequent resets. Note: Once WR is issued, no additional characters should be sent to the module until after the "OK\r" response is received. The WR command should be used sparingly. The EM250 supports a limited number of write cycles.“	CRE	--	--
RE	**Restore Defaults**. Restore module parameters to factory defaults.	CRE	--	--
FR	**Software Reset.** Reset module. Responds immediately with an OK status, and then performs a software reset about 2 seconds later.	CRE	--	--
NR	**Network Reset**. Reset network layer parameters on one or more modules within a PAN. Responds immediately with an “OK” then causes a network restart. All network configuration and routing information is consequently lost. *If NR = 0*: Resets network layer parameters on the node issuing the command. *If NR = 1*: Sends broadcast transmission to reset network layer parameters on all nodes in the PAN.	CRE	0 - 1	--
SI	**Sleep Immediately**. Cause a cyclic sleep module to sleep immediately rather than wait for the ST timer to expire.	E	-	-
ND	**Node Discover.** Discovers and reports all RF modules found. The following information is reported for each module discovered. MY<CR> SH<CR> SL<CR> NI<CR> (Variable length) PARENT_NETWORK ADDRESS (2 Bytes)<CR> DEVICE_TYPE<CR> (1 Byte: 0=Coord, 1=Router, 2=End Device) STATUS<CR> (1 Byte: Reserved) PROFILE_ID<CR> (2 Bytes) MANUFACTURER_ID<CR> (2 Bytes) <CR> After (NT * 100) milliseconds, the command ends by returning a <CR>. ND also accepts a Node Identifier (NI) as a parameter (optional). In this case, only a module that matches the supplied identifier will respond. If ND is sent through the API, each response is returned as a separate AT_CMD_Response packet. The data consists of the above listed bytes without the carriage return delimiters. The NI string will end in a "0x00" null character. The radius of the ND command is set by the BH command.	CRE	optional 20-Byte NI or MY value	--
DN	**Destination Node.** Resolves an NI (Node Identifier) string to a physical address (case-sensitive). The following events occur after the destination node is discovered: <AT Firmware> 1. DL & DH are set to the extended (64-bit) address of the module with the matching NI (Node Identifier) string. 2. OK (or ERROR)\r is returned. 3. Command Mode is exited to allow immediate communication <API Firmware> 1. The 16-bit network and 64-bit extended addresses are returned in an API Command Response frame. If there is no response from a module within (NT * 100) milliseconds or a parameter is not specified (left blank), the command is terminated and an “ERROR” message is returned. In the case of an ERROR, Command Mode is not exited. The radius of the DN command is set by the BH command.	CRE	up to 20-Byte printable ASCII string	--
IS	**Force Sample** Forces a read of all enabled digital and analog input lines.	CRE	--	--
1S	**XBee Sensor Sample**. Forces a sample to be taken on an XBee Sensor device. This command can only be issued to an XBee sensor device using an API remote command.	RE	-	-

Node types that support the command: C = Coordinator, R = Router, E = End Device

11. OEM Support

This chapter provides customization information for the XBee/XBee-PRO ZB modules. In addition to providing an extremely flexible and powerful API, the XBee and XBee-PRO ZB modules are a robust development platform that have passed FCC and ETSI testing. Developers can customize default parameters, or even write or load custom firmware for Ember's EM250 chip.

X-CTU Configuration Tool

Digi provides a Windows X-CTU configuration tool for configuring module parameters and updating firmware. The XCTU has the capability to do the following:

- Discover all XBee devices in the network
- Update firmware on a local module (requires USB or serial connection)
- Read or write module configuration parameters on a local or remote device
- Save and load configuration profiles containing customized settings.

Contact Digi support for more information about the X-CTU.

Customizing XBee ZB Firmware

Once module parameters are tested in an application and finalized, Digi can manufacture modules with specific, customer-defined configurations for a nominal fee. These custom configurations can lock in a firmware version or set command values when the modules are manufactured, eliminating the need for customers to adjust module parameters on arrival. Alternatively, Digi can program custom firmware, including Ember's EZSP UART image, into the modules during manufacturing. Contact Digi to create a custom configuration.

Design Considerations for Digi Drop-In Networking

XBee/XBee-PRO embedded RF modules contain a variety of features that allow for interoperabilitywith Digi's full line of Drop-in Networking products. Interoperability with other "DIN" products can offer these advantages:

- Add IP-connectivity to your network via Cellular, Ethernet or WiFi with a ConnectPort X Gateway.
- Extend the range of your network with the XBee Wall Router.
- Make deployment easy by enabling the Commissioning Pushbutton (pin 20) and AssociateLED (pin 15) to operate with the Network Commissioning Tool software.
- Interface with standard RS-232, USB, Analog & Digital I/O, RS-485, and other industrial devices using XBee Adapters.
- Monitor and manage your network securely from remote locations with Connectware Manager software.

We encourage you to contact our technical representatives for consideration, implementation, or design review of your product for interoperability with Digi's Drop-in Networking solutions.

XBee Bootloader

XBee modules use a modified version of Ember’s boot loader. This bootloader version supports a custom entry mechanism that uses module pins DIN (pin 3), $\overline{\text{DTR}}$ / SLEEP_RQ (pin 9), and $\overline{\text{RTS}}$ (pin 16). To invoke the boot loader, do the following:

1. Set $\overline{\text{DTR}}$ / SLEEP_RQ low (TTL 0V) and RTS high.

2. Send a serial break to the DIN pin and power cycle or reset the module.

3. When the module powers up, $\overline{\text{DTR}}$ / SLEEP_RQ and DIN should be low (TTL 0V) and RTS should be high.

4. Terminate the serial break and send a carriage return at 115200bps to the module.

5. If successful, the module will send the Ember boot loader menu out the DOUT pin at 115200bps.

6. Commands can be sent to the boot loader at 115200bps.

Note: Hardware flow control should be disabled when entering and communicating with the EM250 bootloader.

Programming XBee Modules

Firmware on the XBee and XBee-PRO ZB modules can be updated through one of two means:

- Serially
- SIF header.

Each method is described below.

Serial Firmware Updates

Serial firmware updates make use of the XBee custom bootloader which ships in all units. This modified bootloader is based on Ember's standalone bootloader, but with a modified entry mechanism. The modified entry mechanism uses module pins 3, 9, and 16 (DIN, DTR, and RTS respectively).

The X-CTU program can update firmware serially on the XBee and XBee-PRO ZB modules. Contact Digi support for details.

If an application requires custom firmware to update the XBee firmware serially, the following steps are required.

Invoke XBee Bootloader

See the "XBee Bootloader" section above for steps to invoke the bootloader

Send Firmware Image

After invoking the bootloader, the Ember bootloader will send the bootloader menu characters out the UART at 115200 bps. The application should do the following to upload a firmware image.

1. Look for the bootloader prompt "BL >" to ensure the bootloader is active

2. Send an ASCII "1" character to initiate a firmware update

3. After sending a "1", the EM250 waits for an XModem CRC upload of an .ebl image over the serial line at 115200 bps. The .ebl file must be sent to the EM250 in order.

If no serial transaction is initiated within a 60 second timeout period, the bootloader times out and returns to the menu. If the upload is interrupted with a power cycle or reset event, the EM250 will If no transaction is initiated within 60 seconds, the bootloader times out and returns to the menu. If the upload is interrupted with a power cycle or reset event, the EM250 will detect an invalid application image and enter bootloader mode. The entire ebl image should be uploaded again to recover. If an error occurs while uploading, the EM250 bootloader returns an error code from the following table:

Hex Error Code	Description
0x21	The bootloader encountered an error while trying to parse the Start of Header (SOH) character in the XModem frame.
0x22	The bootloader detected an invalid checksum in the XModem frame.
0x23	The bootloader encountered an error while trying to parse the high byte of the CRC in the XModem frame.
0x24	The bootloader encountered an error while trying to parse the low byte of the CRC in the XModem frame.

Hex Error Code	Description
0x25	The bootloader encountered an error in the sequence number of the current XModem frame.
0x26	The frame that the bootloader was trying to parse was deemed incomplete (some bytes missing or lost).
0x27	The bootloader encountered a duplicate of the previous XModem frame.
0x41	No .ebl header was received when expected.
0x42	Header failed CRC.
0x43	File failed CRC.
0x44	Unknown tag detected in .ebl image.
0x45	Invalid .ebl header signature.
0x46	Trying to flash odd number of bytes.
0x47	Indexed past end of block buffer.
0x48	Attempt to overwrite bootloader flash.
0x49	Attempt to overwrite SIMEE flash.
0x4A	Flash erase failed.
0x4B	Flash write failed.
0x4C	End tag CRC wrong length.
0x4D	Received data before query request/response

SIF Firmware Updates

The XBee/XBee-PRO modules have a 2x5 SIF header that can be used with Ember's InSight tools to upload firmware onto the modules. These tools include a USB device (USBLink) and Ethernet-enabled InSight Adapters. Contact Ember for details.

Warning: If programming firmware through the SIF interface, be aware that uploading firmware through the SIF header can potentially erase the XBee bootloader. If this happens, serial firmware updates will not work.

(The pinout for the SIF headers are shown in chapter 1.)

Writing Custom Firmware

The XBee/XBee-PRO module can be used as a hardware development platform for the EM250. Custom firmware images can be developed around the EmberZNet 2.5.x and 3.x mesh stacks (for the EM250) and uploaded to the XBee.

Warning: If programming firmware through the SIF interface, be aware that uploading firmware through the SIF header can potentially erase the XBee bootloader. If this happens, serial firmware updates will not work.

Regulatory Compliance

XBee modules are FCC and ETSI certified for operation on all 16 channels. The EM250 output power can be configured up to 3dBm with boost mode enabled.

XBee-PRO modules are certified for operation on 14 of the 16 band channels (channels 11 - 24). The scan channels mask of XBee-PRO devices must be set in the application to disable the upper two channels (i.e. 0x01FFF800). The XBee-PRO contains power compensation circuitry to adjust the output power near 18dBm or 10dBm depending on the part number. For best results, the EM250 should be configured with an output power level of 0dBm (or -2dBm if boost mode is enabled). The end product is responsible to adhere to these requirements.

Enabling GPIO 1 and 2

Most of the remaining sections in this chapter describe how to configure GPIO 1 and 2 to function correctly in custom applications that run on the XBee and XBee-PRO modules. In order for GPIO pins 1 and 2 to be configurable, the application must set the GPIO_CFG register to enable GPIO 1 and 2. Bits 4 - 7 in the GPIO_CFG register control the functionality of various GPIO lines. The following table lists values for these bits that enable GPIO 1 and 2. Other functionality is affected by these settings. See the EM250 datasheet from Ember for a complete listing of functionality.

GPIO_CFG[7:4]Enabled Functionality	Enabled Functionality
0000	GPIO 0, 1, 2, 3, 9, 10, 11, 12
0111	0111GPIO 0, 1, 2, 3, 12
1010	GPIO 0, 1, 2, 3
1101	GPIO 0, 1, 2, 3, 11, 12

Example 1

The following code enables GPIO 0, 1, 2, 3, 9, 10, 11, and 12 and maintains all other GPIO_CFG bits.

```
int16u x;
x = GPIO_CFG;
x &= (0xFF0F);// Clear bits 4 - 7
GPIO_CFG = x;
```

Example 2

The following code enables GPIO 0, 1, 2, 3, and 12 and maintains all other GPIO_CFG bits.

```
int16u x;
x = GPIO_CFG;
x &= (0xFF0F);// Clear bits 4 - 7
x |= 0x0070;// Set bits 4 - 7 to 0111 as shown in the table above.
GPIO_CFG = x;
```

Detecting XBee vs XBee-PRO

For some applications, it may be necessary to determine if the code is running on an XBee or an XBee-PRO device. The GPIO1 pin on the EM250 is used to identify the module type (see table 1-03 in chapter 1). GPIO1 is connected to ground on the XBee module. The following code could be used to determine if a module is an XBee or XBee-PRO:

```
GPIO_DIRCLRL = GPIO(1);// Set GPIO1 as an input
GPIO_PUL |= GPIO(1);// Enable GPIO1 pullup resistor
ModuleIsXBeePro = (GPIO_INL & GPIO(1));//ModuleIsXBeePro > 0 if XBee-PRO, =0 if non-PRO.
```

Ensuring Optimal Output Power

XBee modules manufactured before February 2008 had an incorrect configuration setting that caused the default output power mode to be set incorrectly. Digi's ZB and ZNet firmware compensate for this by setting the output power mode in the application firmware.

Custom applications should call the emberSetTxPowerMode() function to set the output power mode as shown below:

XBee Applications

```
emberSetTxPowerMode(EMBER_TX_POWER_MODE_DEFAULT);  or
emberSetTxPowerMode(EMBER_TX_POWER_MODE_BOOST);
```

XBee-PRO Applications:

```
emberSetTxPowerMode(EMBER_TX_POWER_MODE_ALTERNATE);  or
emberSetTxPowerMode(EMBER_TX_POWER_MODE_BOOST_AND_ALTERNATE);
```

XBee-PRO modules must also set a couple of IO lines to enable output power compensation. This is shown below. Once the IO lines are initialized (after powerup), the XBee will enable the power amplifier and LNA as needed.

On Powerup:

```
  /* GPIO 2 should be set low for at least 10 milliseconds when coming up from power cycle. */
  GPIO_DIRSETL = GPIO(2);// Set GPIO 2 as an output
  GPIO_CLRL = GPIO(2);// Drive GPIO 2 low


  /* After at least 10ms, GPIO 2 should be set high to power the output power compensation
circuitry.
  At the same time GPIO 1 should be configured as an output and set low to enable the output
power
  compensation circuitry. */
  GPIO_DIRSETL = GPIO(1) | GPIO(2);// Set GPIO 1,2 as outputs
  GPIO_CLRL = GPIO(1);// Drive GPIO 1 low
  GPIO_SETL = GPIO(2);// Drive GPIO 2 high
```

Improving Low Power Current Consumption

To improve low power current consumption, the XBee should set a couple of unused IO lines as output low. This can be done during application initialization as shown below.

XBee (non-PRO) Initialization:

```
 /* GPIO 1 and 2 are not used in the XBee (non-PRO) and should be set as outputs and driven low
to
 reduce current draw. */
 GPIO_DIRSETL = GPIO(1) | GPIO(2);// Set GPIO 1,2 as outputs
 GPIO_CLRL = GPIO(1) | GPIO(2);// Set GPIO 1,2 low
```

XBee-PRO modules should disable the power compensation circuitry when sleeping to reduce current draw. This is shown below.

When sleeping (end devices):

```
 /* The power compensation shutdown line on XBee-PRO modules (GPIO 1) should be set high
when
 entering sleep to reduce current consumption.  */
 GPIO_SETL = GPIO(1);
```

When waking from sleep (end devices):

```
 /* The power compensation shutdown line on XBee-PRO (GPIO 1) should be set low to enable the
 power compensation circuitry and LNA. */
 GPIO_CLRL = GPIO(1);
```

Appendix A: Definitions

Definitions

Table A-01. Terms and Definitions

ZigBee Node Types	
Coordinator	A node that has the unique function of forming a network. The coordinator is responsible for establishing the operating channel and PAN ID for an entire network. Once established, the coordinator can form a network by allowing routers and end devices to join to it. Once the network is formed, the coordinator functions like a router (it can participate in routing packets and be a source or destination for data packets). -- One coordinator per PAN -- Establishes/Organizes PAN -- Can route data packets to/from other nodes -- Can be a data packet source and destination -- Mains-powered Refer to the XBee coordinator section for more information.
Router	A node that creates/maintains network information and uses this information to determine the best route for a data packet. A router must join a network before it can allow other routers and end devices to join to it. A router can participate in routing packets and is intended to be a mains-powered node. -- Several routers can operate in one PAN -- Can route data packets to/from other nodes -- Can be a data packet source and destination -- Mains-powered Refer to the XBee router section for more information.
End device	End devices must always interact with their parent to receive or transmit data. (See ‘joining definition.) They are intended to sleep periodically and therefore have no routing capacity. An end device can be a source or destination for data packets but cannot route packets. End devices can be battery-powered and offer low-power operation. -- Several end devices can operate in one PAN -- Can be a data packet source and destination -- All messages are relayed through a coordinator or router -- Lower power modes
ZigBee Protocol	
PAN	Personal Area Network - A data communication network that includes a coordinator and one or more routers/end devices.

Table A-01. Terms and Definitions

Joining	The process of a node becoming part of a ZigBee PAN. A node becomes part of a network by joining to a coordinator or a router (that has previously joined to the network). During the process of joining, the node that allowed joining (the parent) assigns a 16-bit address to the joining node (the child).
Network Address	The 16-bit address assigned to a node after it has joined to another node. The coordinator always has a network address of 0.
Operating Channel	The frequency selected for data communications between nodes. The operating channel is selected by the coordinator on power-up.
Energy Scan	A scan of RF channels that detects the amount of energy present on the selected channels. The coordinator uses the energy scan to determine the operating channel.
Route Request	Broadcast transmission sent by a coordinator or router throughout the network in attempt to establish a route to a destination node.
Route Reply	Unicast transmission sent back to the originator of the route request. It is initiated by a node when it receives a route request packet and its address matches the Destination Address in the route request packet.
Route Discovery	The process of establishing a route to a destination node when one does not exist in the Routing Table. It is based on the AODV (Ad-hoc On-demand Distance Vector routing) protocol.
ZigBee Stack	ZigBee is a published specification set of high-level communication protocols for use with small, low-power modules. The ZigBee stack provides a layer of network functionality on top of the 802.15.4 specification. For example, the mesh and routing capabilities available to ZigBee solutions are absent in the 802.15.4 protocol.

Appendix B: Agency Certifications

United States FCC

The XBee RF Module complies with Part 15 of the FCC rules and regulations. Compliance with the labeling requirements, FCC notices and antenna usage guidelines is required.

To fufill FCC Certification, the OEM must comply with the following regulations:

1.The system integrator must ensure that the text on the external label provided with this device is placed on the outside of the final product. [Figure A-01]

2.XBee RF Module may only be used with antennas that have been tested and approved for use with this module [refer to the antenna tables in this section].

OEM Labeling Requirements

WARNING: The Original Equipment Manufacturer (OEM) must ensure that FCC labeling requirements are met. This includes a clearly visible label on the outside of the final product enclosure that displays the contents shown in the figure below.

Required FCC Label for OEM products containing the XBee RF Module

> Contains FCC ID: OUR-XBEE2*
>
> The enclosed device complies with Part 15 of the FCC Rules. Operation is subject to the following two conditions: (*i.*) this device may not cause harmful interference and (*ii.*) this device must accept any interference received, including interference that may cause undesired operation.

Required FCC Label for OEM products containing the XBee PRO RF Module

> Contains FCC ID:MCQ-XBEEPRO2*
>
> The enclosed device complies with Part 15 of the FCC Rules. Operation is subject to the following two conditions: (*i.*) this device may not cause harmful interference and (*ii.*) this device must accept any interference received, including interference that may cause undesired operation.

FCC Notices

IMPORTANT: The XBee and XBee PRO RF Module have been certified by the FCC for use with other products without any further certification (as per FCC section 2.1091). Modifications not expressly approved by Digi could void the user's authority to operate the equipment.

IMPORTANT: OEMs must test final product to comply with unintentional radiators (FCC section 15.107 & 15.109) before declaring compliance of their final product to Part 15 of the FCC Rules.

IMPORTANT: The RF module has been certified for remote and base radio applications. If the module will be used for portable applications, the device must undergo SAR testing.

This equipment has been tested and found to comply with the limits for a Class B digital device, pursuant to Part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation. This equipment generates, uses and can radiate radio frequency energy and, if not installed and used in accordance with the instructions, may cause harmful interference to radio communications. However, there is no guarantee that interference will not occur in a particular installation.

If this equipment does cause harmful interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures: Re-orient or relocate the receiving antenna, Increase the separation between the equipment and receiver, Connect equipment and receiver to outlets on different circuits, or Consult the dealer or an experienced radio/TV technician for help.

FCC-Approved Antennas (2.4 GHz)

The XBee and XBee-PRO RF Module can be installed utilizing antennas and cables constructed with standard connectors (Type-N, SMA, TNC, etc.) if the installation is performed professionally and according to FCC guidelines. For installations not performed by a professional, non-standard connectors (RPSMA, RPTNC, etc.) must be used.

The modules are FCC approved for fixed base station and mobile applications on channels 0x0B-0x1A for Xbee Series2 and on channels 0x0B - 0x18 for Xbee ZNet-PRO 2.5 . If the antenna is mounted at least 20cm (8 in.) from nearby persons, the application is considered a mobile application. Antennas not listed in the table must be tested to comply with FCC Section 15.203 (Unique Antenna Connectors) and Section 15.247 (Emissions).

XBee RF Modules: XBee RF Modules have been tested and approved for use with all the antennas listed in the tables below. (Cable-loss IS required when using gain antennas as shown below.)

Table A-01. antennas approved for use with the XBee RF Modules

YAGI CLASS ANTENNAS					
Part Number	**Type (Description)**	**Gain**	**Application***	**Min. Separation Required**	**Cable-loss**
A24-Y6NF	Yagi (6-element)	8.8 dBi	Fixed	2 m	N/A
A24-Y7NF	Yagi (7-element)	9.0 dBi	Fixed	2 m	N/A
A24-Y9NF	Yagi (9-element)	10.0 dBi	Fixed	2 m	N/A
A24-Y10NF	Yagi (10-element)	11.0 dBi	Fixed	2 m	N/A
A24-Y12NF	Yagi (12-element)	12.0 dBi	Fixed	2 m	N/A
A24-Y13NF	Yagi (13-element)	12.0 dBi	Fixed	2 m	N/A
A24-Y15NF	Yagi (15-element)	12.5 dBi	Fixed	2 m	N/A
A24-Y16NF	Yagi (16-element)	13.5 dBi	Fixed	2 m	N/A
A24-Y16RM	Yagi (16-element, RPSMA connector)	13.5 dBi	Fixed	2 m	N/A
A24-Y18NF	Yagi (18-element)	15.0 dBi	Fixed	2 m	N/A
OMNI-DIRECTIONAL ANTENNAS					
Part Number	**Type (Description)**	**Gain**	**Application***	**Min. Separation Required**	**Cable-loss**
A24-C1	Surface Mount integral chip	-1.5 dBi	Fixed/Mobile	20 cm	N/A
A24-F2NF	Omni-directional (Fiberglass base station)	2.1 dBi	Fixed/Mobile	20 cm	N/A
A24-F3NF	Omni-directional (Fiberglass base station)	3.0 dBi	Fixed/Mobile	20 cm	N/A
A24-F5NF	Omni-directional (Fiberglass base station)	5.0 dBi	Fixed/Mobile	20 cm	N/A
A24-F8NF	Omni-directional (Fiberglass base station)	8.0 dBi	Fixed	2 m	N/A
A24-F9NF	Omni-directional (Fiberglass base station)	9.5 dBi	Fixed	2 m	N/A
A24-F10NF	Omni-directional (Fiberglass base station)	10.0 dBi	Fixed	2 m	N/A
A24-F12NF	Omni-directional (Fiberglass base station)	12.0 dBi	Fixed	2 m	N/A
A24-F15NF	Omni-directional (Fiberglass base station)	15.0 dBi	Fixed	2 m	N/A
A24-W7NF	Omni-directional (Base station)	7.2 dBi	Fixed	2 m	N/A
A24-M7NF	Omni-directional (Mag-mount base station)	7.2 dBi	Fixed	2 m	N/A
PANEL CLASS ANTENNAS					
Part Number	**Type (Description)**	**Gain**	**Application***	**Min. Separation Required**	**Cable-loss**
A24-P8SF	Flat Panel	8.5 dBi	Fixed	2 m	N/A
A24-P8NF	Flat Panel	8.5 dBi	Fixed	2 m	N/A
A24-P13NF	Flat Panel	13.0 dBi	Fixed	2 m	N/A
A24-P14NF	Flat Panel	14.0 dBi	Fixed	2 m	N/A
A24-P15NF	Flat Panel	15.0 dBi	Fixed	2 m	N/A
A24-P16NF	Flat Panel	16.0 dBi	Fixed	2 m	N/A
A24-P19NF	Flat Panel	19.0 dBi	Fixed	2m	1.5 dB

Table A-02. antennas approved for use with the XBee-PRO RF Modules

YAGI CLASS ANTENNAS					
Part Number	**Type (Description)**	**Gain**	**Application***	**Min. Separation Required**	**Cable-loss**
A24-Y6NF	Yagi (6-element)	8.8 dBi	Fixed	2 m	7.8dB
A24-Y7NF	Yagi (7-element)	9.0 dBi	Fixed	2 m	8 dB
A24-Y9NF	Yagi (9-element)	10.0 dBi	Fixed	2 m	9 dB
A24-Y10NF	Yagi (10-element)	11.0 dBi	Fixed	2 m	10 dB
A24-Y12NF	Yagi (12-element)	12.0 dBi	Fixed	2 m	11 dB
A24-Y13NF	Yagi (13-element)	12.0 dBi	Fixed	2 m	11 dB
A24-Y15NF	Yagi (15-element)	12.5 dBi	Fixed	2 m	11.5 dB
A24-Y16NF	Yagi (16-element)	13.5 dBi	Fixed	2 m	12.5 dB
A24-Y16RM	Yagi (16-element, RPSMA connector)	13.5 dBi	Fixed	2 m	12.5 dB
A24-Y18NF	Yagi (18-element)	15.0 dBi	Fixed	2 m	14 dB
OMNI-DIRECTIONAL ANTENNAS					
Part Number	**Type (Description)**	**Gain**	**Application***	**Min. Separation Required**	**Cable-loss**
A24-C1	Surface Mount integral chip	-1.5dBi	Fixed/Mobile	20 cm	-
A24-F2NF	Omni-directional (Fiberglass base station)	2.1 dBi	Fixed/Mobile	20 cm	-
A24-F3NF	Omni-directional (Fiberglass base station)	3.0 dBi	Fixed/Mobile	20 cm	.3 dB
A24-F5NF	Omni-directional (Fiberglass base station)	5.0 dBi	Fixed/Mobile	20 cm	2.3 dB
A24-F8NF	Omni-directional (Fiberglass base station)	8.0 dBi	Fixed	2 m	5.3 dB
A24-F9NF	Omni-directional (Fiberglass base station)	9.5 dBi	Fixed	2 m	6.8 dB
A24-F10NF	Omni-directional (Fiberglass base station)	10.0 dBi	Fixed	2 m	7.3 dB
A24-F12NF	Omni-directional (Fiberglass base station)	12.0 dBi	Fixed	2 m	9.3dB
A24-F15NF	Omni-directional (Fiberglass base station)	15.0 dBi	Fixed	2 m	12.3dB
A24-W7NF	Omni-directional (Base station)	7.2 dBi	Fixed	2 m	4.5 dB
A24-M7NF	Omni-directional (Mag-mount base station)	7.2 dBi	Fixed	2 m	4.5 dB
PANEL CLASS ANTENNAS					
Part Number	**Type (Description)**	**Gain**	**Application***	**Min. Separation Required**	**Cable-loss**
A24-P8SF	Flat Panel	8.5 dBi	Fixed	2 m	8.2 dB
A24-P8NF	Flat Panel	8.5 dBi	Fixed	2 m	82 dB
A24-P13NF	Flat Panel	13.0 dBi	Fixed	2 m	12.7 dB
A24-P14NF	Flat Panel	14.0 dBi	Fixed	2 m	13.7 dB
A24-P15NF	Flat Panel	15.0 dBi	Fixed	2 m	14.7 dB
A24-P16NF	Flat Panel	16.0 dBi	Fixed	2 m	15.7 dB
A24-P19NF	Flat Panel	19.0 dBi	Fixed	2m	18.7 dB

*** If using the RF module in a portable application** (For example - If the module is used in a handheld device and the antenna is less than 20cm from the human body when the device is in operation): The integrator is responsible for passing additional SAR (Specific Absorption Rate) testing based on FCC rules 2.1091 and FCC Guidelines for Human Exposure to Radio Frequency Electromagnetic Fields, OET Bulletin and Supplement C. The testing results will be submitted to the FCC for approval prior to selling the integrated unit. The required SAR testing measures emissions from the module and how they affect the person.

RF Exposure

WARNING: To satisfy FCC RF exposure requirements for mobile transmitting devices, a separation distance of 20 cm or more should be maintained between the antenna of this device and persons during device operation. To ensure compliance, operations at closer than this distance are not recommended. The antenna used for this transmitter must not be co-located in conjunction with any other antenna or transmitter.

The preceding statement must be included as a CAUTION statement in OEM product manuals in order to alert users of FCC RF Exposure compliance.

Europe (ETSI)

The XBee RF Module has been certified for use in several European countries. For a complete list, refer to www.digi.com

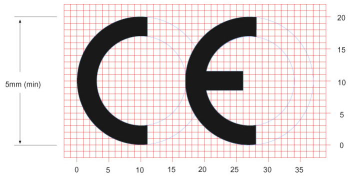
If the XBee RF Modules are incorporated into a product, the manufacturer must ensure compliance of the final product to the European harmonized EMC and low-voltage/safety standards. A Declaration of Conformity must be issued for each of these standards and kept on file as described in Annex II of the R&TTE Directive.

Furthermore, the manufacturer must maintain a copy of the XBee user manual documentation and ensure the final product does not exceed the specified power ratings, antenna specifications, and/ or installation requirements as specified in the user manual. If any of these specifications are exceeded in the final product, a submission must be made to a notified body for compliance testing to all required standards.

OEM Labeling Requirements

The 'CE' marking must be affixed to a visible location on the OEM product.

Figure B-01. CE Labeling Requirements



The CE mark shall consist of the initials "CE" taking the following form:

- If the CE marking is reduced or enlarged, the proportions given in the above graduated drawing must be respected.
- The CE marking must have a height of at least 5mm except where this is not possible on account of the nature of the apparatus.
- The CE marking must be affixed visibly, legibly, and indelibly.

Restrictions

Power Output: The power output of the XBee RF Module must not exceed 10 dBm. The power level is set using the PL command and the PL parameter must equal “0” (10 dBm).

France: France imposes restrictions on the 2.4 GHz band. Go to www.art-telecom.Fr or contact MaxStream for more information.

Norway: Norway prohibits operation near Ny-Alesund in Svalbard. More information can be found at the Norway Posts and Telecommunications site (www.npt.no).

Declarations of Conformity

Digi has issued Declarations of Conformity for the XBee RF Modules concerning emissions, EMC and safety. Files are located in the 'documentation' folder of the Digi CD.

Important Note

Digi does not list the entire set of standards that must be met for each country. Digi customers assume full responsibility for learning and meeting the required guidelines for each country in their distribution market. For more information relating to European compliance of an OEM product incorporating the XBee RF Module, contact Digi, or refer to the following web sites:

CEPT ERC 70-03E - Technical Requirements, European restrictions and general requirements: Available at www.ero.dk/.

R&TTE Directive - Equipment requirements, placement on market: Available at www.ero.dk/.

Approved Antennas

When integrating high-gain antennas, European regulations stipulate EIRP power maximums. Use the following guidelines to determine which antennas to design into an application.

XBee OEM Module
The following antennas types have been tested and approved for use with the XBee Module:

Antenna Type: Yagi
RF module was tested and approved with 15 dBi antenna gain with 1 dB cable-loss (EIRP Maimum of 14 dBm). Any Yagi type antenna with 14 dBi gain or less can be used with no cable-loss.

Antenna Type: Omni-Directional

RF module was tested and approved with 15 dBi antenna gain with 1 dB cable-loss (EIRP Maimum of 14 dBm). Any Omni-Directional type antenna with 14 dBi gain or less can be used with no cable-loss.

Antenna Type: Flat Panel

RF module was tested and approved with 19 dBi antenna gain with 4.8 dB cable-loss (EIRP Maimum of 14.2 dBm). Any Flat Panel type antenna with 14.2 dBi gain or less can be used with no cable-loss.

XBee RF Module

The following antennas have been tested and approved for use with the embedded XBee RF Module:

- Dipole (2.1 dBi, Omni-directional, Articulated RPSMA, Digi part number A24-HABSM)
- Chip Antenna (-1.5 dBi)
- Attached Monopole Whip (1.5 dBi)

XBee-PRO RF Module

The following antennas have been tested and approved for use with the embedded XBee-PRO RF Module:

- Dipole (2.1 dBi, Omni-directional, Articulated RPSMA, Digi part number A24-HABSM)
- Chip Antenna (-1.5 dBi)
- Attached Monopole Whip (1.5 dBi).

Canada (IC)

Labeling Requirements

Labeling requirements for Industry Canada are similar to those of the FCC. A clearly visible label on the outside of the final product enclosure must display the following text:

Contains Model XBee Radio, IC: 4214A-XBEE2

The integrator is responsible for its product to comply with IC ICES-003 & FCC Part 15, Sub. B - Unintentional Radiators. ICES-003 is the same as FCC Part 15 Sub. B and Industry Canada accepts FCC test report or CISPR 22 test report for compliance with ICES-003.

If it contains an XBee-PRO OEM Module, the clearly visible label on the outside of the final product enclosure must display the following text:

Contains Model XBee PRO Radio, IC: 1846A-XBEEPRO2

The integrator is responsible for its product to comply with IC ICES-003 & FCC Part 15, Sub. B - Unintentional Radiators. ICES-003 is the same as FCC Part 15 Sub. B and Industry Canada accepts FCC test report or CISPR 22 test report for compliance with ICES-003.

Transmitters for Detachable Antennas

This device has been designed to operate with the antennas listed in table A-3 and having a maximum of 17.5 dB. Antennas not included in this list or having a gain greater than 17.5 dB are strictly prohibited for use with this device. The required antenna impedance is 50 Ω

Detachable Antenna

To reduce potential radio interference to other users, the antenna type and gained shuold be so chosen that the equivaleny istropically radiated power (e.i.r.p.) is not more than permitted for successful communication

Appendix C: Migrating from ZNet 2.5 to XBee ZB

The following paragraph contains the significant differences in XBee ZB compared to its predecessor, ZNet 2.5.

- Command Set
- Firmware Versions
- New Features.

Command Set

The following ZNet 2.5 commands have changed for XBee ZB:

- ZA - Set / read the ZigBee Addressing enable command. This command was required in ZNet 2.5 to enable application level addressing commands SE, DE, CI. XBee ZB does not support ZA. The SE, DE, and CI values always determine the application level addressing values.
- AI - Read the association status. AI now includes several new values.

Firmware Versions

ZNet 2.5 supported coordinator and router/end device targets. Due to flash constraints, XBee ZB split the router/end device target into 2 different targets - router, and end device. There is not a router/end device target.

New Features

ZB offers many new and improved features over ZNet 2.5, including:

- Data transmissions are directly resolved to APS unicasts. This provides the ability to send and receive ZDO commands.
- NH command configures the unicast transmission timeout. This command can extend the number of unicast hops dramatically over the 6-8 hop limit that existed in ZNet 2.5.
- ZS command allows ZigBee stack profile to be set as required.

Appendix D: XBee ZB Firmware Matrix

Overview of Features

The XBee ZB firmware supports the following versions:

- 204x - AT Coordinator
- 214x - API Coordinator
- 224x - AT Router
- 234x - API Router
- 284x - AT End Device
- 294x - API End Device.

The supported features of each firmware version are listed in the table below:

Feature	204x (AT Crd)	214x (API Crd)	224x (AT Rtr)	234x (API Rtr)	284x (AT End-Dev)	294x (API End-Dev)
Aggregator Capable	X	X	X	X		
Allows Joining	X	X	X	X		
AT Cmd Mode	X		X		X	
API		X		X		X
Channel Verification on Join (JV)			X	X		
Commissioning Button	X	X	X	X	X	X
IO Sampling (IS)			X		X	X
Receives unicast traffic from devices in the network	X	X	X	X	X	X
Receives broadcast traffic from devices in the network	X	X	X	X	X	X
Responds to Remote Commands	X	X	X	X	X	X
Responds to Network Discovery	X	X	X	X	X	X
RF data transmissions / receptions are always routed through the parent					X	X
Routes Data	X	X	X	X		

Sends Network Discovery (ND)	X	X	X	X	X	
Sends Network Reset (NR1)	X	X	X	X	X	
Sends Remote Commands		X		X		X
Sleep Modes					X	X

Appendix E: Additional Information

1-Year Warranty

XBee RF Modules from Digi, Inc. (the "Product") are warranted against defects in materials and workmanship under normal use, for a period of 1-year from the date of purchase. In the event of a product failure due to materials or workmanship, Digi will repair or replace the defective product. For warranty service, return the defective product to MaxStream, shipping prepaid, for prompt repair or replacement.

The foregoing sets forth the full extent of MaxStream's warranties regarding the Product. Repair or replacement at MaxStream's option is the exclusive remedy. THIS WARRANTY IS GIVEN IN LIEU OF ALL OTHER WARRANTIES, EXPRESS OR IMPLIED, AND DIGI SPECIFICALLY DISCLAIMS ALL WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT SHALL DIGI, ITS SUPPLIERS OR LICENSORS BE LIABLE FOR DAMAGES IN EXCESS OF THE PURCHASE PRICE OF THE PRODUCT, FOR ANY LOSS OF USE, LOSS OF TIME, INCONVENIENCE, COMMERCIAL LOSS, LOST PROFITS OR SAVINGS, OR OTHER INCIDENTAL, SPECIAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PRODUCT, TO THE FULL EXTENT SUCH MAY BE DISCLAIMED BY LAW. SOME STATES DO NOT ALLOW THE EXCLUSION OR LIMITATION OF INCIDENTAL OR CONSEQUENTIAL DAMAGES. THEREFORE, THE FOREGOING EXCLUSIONS MAY NOT APPLY IN ALL CASES. This warranty provides specific legal rights. Other rights which vary from state to state may also apply.

Contact Digi

Free and unlimited technical support is included with every Digi Radio Modem sold. For the best in wireless data solutions and support, please use the following resources:

Technical Support:	Phone.	(866) 765-9885 toll-free U.S.A. & Canada (801) 765-9885 Worldwide
	Live Chat.	www.digi.com
	Online Support.	http://www.digi.com/support/eservice/login.jsp

Digi’s office hours are 8:00 am - 5:00 pm [U.S. Mountain Time]

XBee™ / XBee-PRO™ OEM RF Modules

XBee/XBee-PRO OEM RF Modules

RF Module Operation

RF Module Configuration

Appendices

Product Manual v1.xAx - 802.15.4 Protocol

For OEM RF Module Part Numbers: XB24-...-001, XBP24-...-001

IEEE® 802.15.4 OEM RF Modules by MaxStream

355 South 520 West, Suite 180
Lindon, UT 84042
Phone: (801) 765-9885
Fax: (801) 765-9895

rf-xperts@maxstream.net
www.MaxStream.net (live chat support)

M100232
2007.05.031

Technical Support:	Phone: (801) 765-9885
	Live Chat: www.maxstream.net
	E-mail: rf-xperts@maxstream.net

Contents

1. XBee/XBee-PRO OEM RF Modules 4

1.1. Key Features 4

1.1.1. Worldwide Acceptance 4

1.2. Specifications 5

1.3. Mechanical Drawings 6

1.4. Mounting Considerations 6

1.5. Pin Signals 7

1.6. Electrical Characteristics 8

2. RF Module Operation 9

2.1. Serial Communications 9

2.1.1. UART Data Flow 9

2.1.2. Transparent Operation 10

2.1.3. API Operation 10

2.1.4. Flow Control 11

2.2. ADC and Digital I/O Line Support 12

2.2.1. I/O Data Format 12

2.2.2. API Support 13

2.2.3. Sleep Support 13

2.2.4. DIO Pin Change Detect 13

2.2.5. Sample Rate (Interval) 13

2.2.6. I/O Line Passing 14

2.2.7. Configuration Example 14

2.3. XBee/XBee-PRO Networks 15

2.3.1. NonBeacon 15

2.3.2. NonBeacon (w/ Coordinator) 15

2.3.3. Association 16

2.4. XBee/XBee-PRO Addressing 19

2.4.1. Unicast Mode 19

2.4.2. Broadcast Mode 19

2.5. Modes of Operation 20

2.5.1. Idle Mode 20

2.5.2. Transmit/Receive Modes 20

2.5.3. Sleep Mode 22

2.5.4. Command Mode 24

3. RF Module Configuration 25

3.1. Programming the RF Module 25

3.1.1. Programming Examples 25

3.2. Command Reference Tables 26

3.3. Command Descriptions 34

3.4. API Operation 54

3.4.1. API Frame Specifications 54

3.4.2. API Types 55

Appendix A: Agency Certifications 59

United States (FCC) 59

OEM Labeling Requirements 59

FCC Notices 59

FCC-Approved Antennas (2.4 GHz) 60

Europe (ETSI) 61

OEM Labeling Requirements 61

Restrictions 61

Declarations of Conformity 61

Approved Antennas 62

Canada (IC) 62

Labeling Requirements 62

Japan 62

Labeling Requirements 62

Appendix B: Development Guide 63

Development Kit Contents 63

Interfacing Options 63

RS-232 Development Board 64

External Interface 64

RS-232 Pin Signals 65

Wiring Diagrams 66

Adapters 67

USB Development Board 68

External Interface 68

USB Pin Signals 68

X-CTU Software 69

Installation 69

Serial Communications Software 69

Appendix C: Additional Information 70

1-Year Warranty 70

Ordering Information 70

Contact MaxStream 71

1. XBee/XBee-PRO OEM RF Modules

The XBee and XBee-PRO OEM RF Modules were engineered to meet IEEE 802.15.4 standards and support the unique needs of low-cost, low-power wireless sensor networks. The modules require minimal power and provide reliable delivery of data between devices.

The modules operate within the ISM 2.4 GHz frequency band and are pin-for-pin compatible with each other.

1.1. Key Features

Long Range Data Integrity

XBee

- Indoor/Urban: up to 100' (30 m)
- Outdoor line-of-sight: up to 300' (100 m)
- Transmit Power: 1 mW (0 dBm)
- Receiver Sensitivity: -92 dBm

XBee-PRO

- Indoor/Urban: up to 300' (100 m)
- Outdoor line-of-sight: up to 1 mile (1500 m)
- Transmit Power: 100 mW (20 dBm) EIRP
- Receiver Sensitivity: -100 dBm

RF Data Rate: 250,000 bps

Advanced Networking & Security

Retries and Acknowledgements

DSSS (Direct Sequence Spread Spectrum)

Each direct sequence channels has over 65,000 unique network addresses available

Source/Destination Addressing

Unicast & Broadcast Communications

Point-to-point, point-to-multipoint and peer-to-peer topologies supported

Coordinator/End Device operations

Low Power

XBee

- TX Current: 45 mA (@3.3 V)
- RX Current: 50 mA (@3.3 V)
- Power-down Current: < 10 µA

XBee-PRO

- TX Current: 215 mA (@3.3 V)
- RX Current: 55 mA (@3.3 V)
- Power-down Current: < 10 µA

ADC and I/O line support

Analog-to-digital conversion, Digital I/O

I/O Line Passing

Easy-to-Use

No configuration necessary for out-of box RF communications

Free X-CTU Software
(Testing and configuration software)

AT and API Command Modes for configuring module parameters

Extensive command set

Small form factor

Free & Unlimited RF-XPert Support

1.1.1. Worldwide Acceptance

FCC Approval (USA) Refer to Appendix A [p59] for FCC Requirements.
Systems that contain XBee/XBee-PRO RF Modules inherit MaxStream Certifications.

ISM (Industrial, Scientific & Medical) **2.4 GHz frequency band**

Manufactured under **ISO 9001:2000** registered standards

XBee/XBee-PRO RF Modules are optimized for use in the **United States**, **Canada**, **Australia, Israel and Europe**. Contact MaxStream for complete list of government agency approvals.

1.2. Specifications

Table 1-01. Specifications of the XBee/XBee-PRO OEM RF Modules

Specification	XBee	XBee-PRO
Performance		
Indoor/Urban Range	up to 100 ft. (30 m)	Up to 300’ (100 m)
Outdoor RF line-of-sight Range	up to 300 ft. (100 m)	Up to 1 mile (1500 m)
Transmit Power Output (software selectable)	1mW (0 dBm)	60 mW (18 dBm) conducted, 100 mW (20 dBm) EIRP*
RF Data Rate	250,000 bps	250,000 bps
Serial Interface Data Rate (software selectable)	1200 - 115200 bps (non-standard baud rates also supported)	1200 - 115200 bps (non-standard baud rates also supported)
Receiver Sensitivity	-92 dBm (1% packet error rate)	-100 dBm (1% packet error rate)
Power Requirements		
Supply Voltage	2.8 – 3.4 V	2.8 – 3.4 V

Transmit Current (typical)	45mA (@ 3.3 V)	If PL=0 (10dBm): 137mA(@3.3V), 139mA(@3.0V)
		PL=1 (12dBm): 155mA (@3.3V), 153mA(@3.0V)
		PL=2 (14dBm): 170mA (@3.3V), 171mA(@3.0V)
		PL=3 (16dBm): 188mA (@3.3V), 195mA(@3.0V)
		PL=4 (18dBm): 215mA (@3.3V), 227mA(@3.0V)

Idle / Receive Current (typical)	50mA (@ 3.3 V)	55mA (@ 3.3 V)
Power-down Current	< 10 µA	< 10 µA
General		
Operating Frequency	ISM 2.4 GHz	ISM 2.4 GHz
Dimensions	0.960” x 1.087” (2.438cm x 2.761cm)	0.960” x 1.297” (2.438cm x 3.294cm)
Operating Temperature	-40 to 85º C (industrial)	-40 to 85º C (industrial)
Antenna Options	Integrated Whip, Chip or U.FL Connector	Integrated Whip, Chip or U.FL Connector
Networking & Security		
Supported Network Topologies	Point-to-point, Point-to-multipoint & Peer-to-peer	
Number of Channels (software selectable)	16 Direct Sequence Channels	12 Direct Sequence Channels
Addressing Options	PAN ID, Channel and Addresses	PAN ID, Channel and Addresses
Agency Approvals		
United States (FCC Part 15.247)	OUR-XBEE	OUR-XBEEPRO
Industry Canada (IC)	4214A XBEE	4214A XBEEPRO
Europe (CE)	ETSI	ETSI (Max. 10 dBm transmit power output)*
Japan	n/a	005NYCA0378 (Max. 10 dBm transmit power output)**

* When operating in Europe: XBee-PRO RF Modules must be configured to operate at a maximum transmit power output level of 10 dBm. The power output level is set using the PL command. The PL parameter must equal “0” (10 dBm).

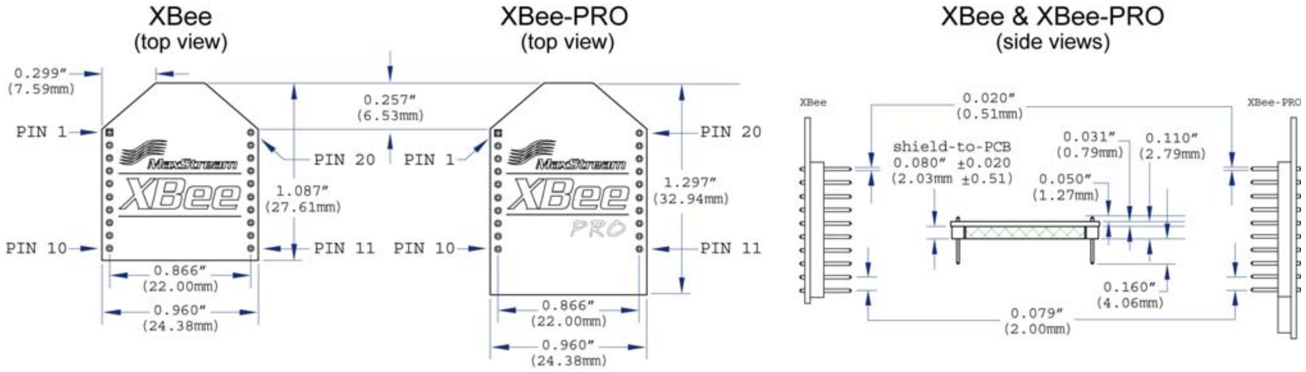
Additionally, European regulations stipulate an EIRP power maximum of 12.86 dBm (19 mW) for the XBee-PRO and 12.11 dBm for the XBee when integrating high-gain antennas.

** When operating in Japan: Transmit power output is limited to 10 dBm. A special part number is required when ordering modules approved for use in Japan. Contact MaxStream for more information [call 1-801-765-9885 or send e-mails to sales@maxstream.net].

Antenna Options: The ranges specified are typical when using the integrated Whip (1.5 dBi) and Dipole (2.1 dBi) antennas. The Chip antenna option provides advantages in its form factor; however, it typically yields shorter range than the Whip and Dipole antenna options when transmitting outdoors. For more information, refer to the “XBee Antenna” application note located on MaxStream’s web site (http://www.maxstream.net/support/knowledgebase/article.php?kb=153).

1.3. Mechanical Drawings

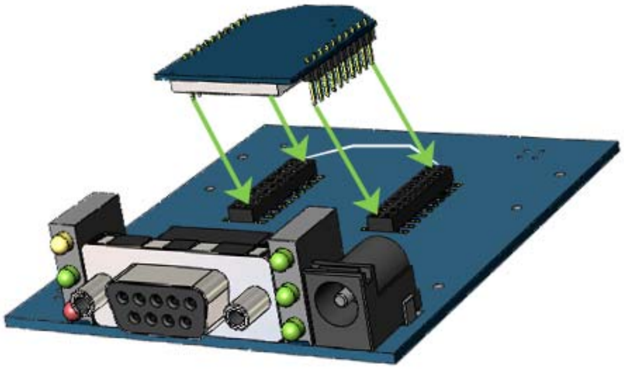
Figure 1-01. Mechanical drawings of the XBee/XBee-PRO OEM RF Modules (antenna options not shown)
The XBee and XBee-PRO RF Modules are pin-for-pin compatible.



1.4. Mounting Considerations

The XBee/XBee-PRO RF Module was designed to mount into a receptacle (socket) and therefore does not require any soldering when mounting it to a board. The XBee Development Kits contain RS-232 and USB interface boards which use two 20-pin receptacles to receive modules.

Figure 1-02. XBee Module Mounting to an RS-232 Interface Board.



The receptacles used on MaxStream development boards are manufactured by Century Interconnect. Several other manufacturers provide comparable mounting solutions; however, MaxStream currently uses the following receptacles:

- Through-hole single-row receptacles -
  Samtec P/N: MMS-110-01-L-SV (or equivalent)
- Surface-mount double-row receptacles -
  Century Interconnect P/N: CPRMSL20-D-0-1 (or equivalent)
- Surface-mount single-row receptacles -
  Samtec P/N: SMM-110-02-SM-S

MaxStream also recommends printing an outline of the module on the board to indicate the orientation the module should be mounted.

1.5. Pin Signals

Figure 1-03. XBee/XBee-PRO RF Module Pin Numbers

(top sides shown - shields on bottom)

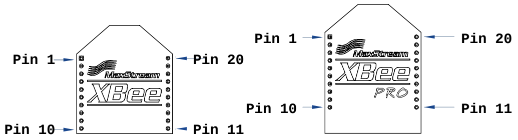


Table 1-02. Pin Assignments for the XBee and XBee-PRO Modules
(Low-asserted signals are distinguished with a horizontal line above signal name.)

Pin #	Name	Direction	Description
1	VCC	-	Power supply
2	DOUT	Output	UART Data Out
3	DIN / $\overline{\text{CONFIG}}$	Input	UART Data In
4	DO8*	Output	Digital Output 8
5	$\overline{\text{RESET}}$	Input	Module Reset (reset pulse must be at least 200 ns)
6	PWM0 / RSSI	Output	PWM Output 0 / RX Signal Strength Indicator
7	PWM1	Output	PWM Output 1
8	[reserved]	-	Do not connect
9	$\overline{\text{DTR}}$ / SLEEP_RQ / DI8	Input	Pin Sleep Control Line or Digital Input 8
10	GND	-	Ground
11	AD4 / DIO4	Either	Analog Input 4 or Digital I/O 4
12	$\overline{\text{CTS}}$ / DIO7	Either	Clear-to-Send Flow Control or Digital I/O 7
13	ON / $\overline{\text{SLEEP}}$	Output	Module Status Indicator
14	VREF	Input	Voltage Reference for A/D Inputs
15	Associate / AD5 / DIO5	Either	Associated Indicator, Analog Input 5 or Digital I/O 5
16	$\overline{\text{RTS}}$ / AD6 / DIO6	Either	Request-to-Send Flow Control, Analog Input 6 or Digital I/O 6
17	AD3 / DIO3	Either	Analog Input 3 or Digital I/O 3
18	AD2 / DIO2	Either	Analog Input 2 or Digital I/O 2
19	AD1 / DIO1	Either	Analog Input 1 or Digital I/O 1
20	AD0 / DIO0	Either	Analog Input 0 or Digital I/O 0

* Function is not supported at the time of this release

Design Notes:

- Minimum connections: VCC, GND, DOUT & DIN
- Minimum connections for updating firmware: VCC, GND, DIN, DOUT, RTS & DTR
- Signal Direction is specified with respect to the module
- Module includes a 50k Ω pull-up resistor attached to $\overline{\text{RESET}}$
- Several of the input pull-ups can be configured using the PR command
- Unused pins should be left disconnected

1.6. Electrical Characteristics

Table 1-03. DC Characteristics (VCC = 2.8 - 3.4 VDC)

Symbol	Characteristic	Condition	Min	Typical	Max	Unit
V_{IL}	Input Low Voltage	All Digital Inputs	-	-	0.35 * VCC	V
V_{IH}	Input High Voltage	All Digital Inputs	0.7 * VCC	-	-	V
V_{OL}	Output Low Voltage	I_{OL} = 2 mA, VCC >= 2.7 V	-	-	0.5	V
V_{OH}	Output High Voltage	I_{OH} = -2 mA, VCC >= 2.7 V	VCC - 0.5	-	-	V
II_{IN}	Input Leakage Current	V_{IN} = VCC or GND, all inputs, per pin	-	0.025	1	µA
II_{OZ}	High Impedance Leakage Current	V_{IN} = VCC or GND, all I/O High-Z, per pin	-	0.025	1	µA
TX	Transmit Current	VCC = 3.3 V	-	45 (XBee) 215 (PRO)	-	mA
RX	Receive Current	VCC = 3.3 V	-	50 (XBee) 55 (PRO)	-	mA
PWR-DWN	Power-down Current	SM parameter = 1	-	< 10	-	µA

Table 1-04. ADC Characteristics (Operating)

Symbol	Characteristic	Condition	Min	Typical	Max	Unit
V_{REFH}	VREF - Analog-to-Digital converter reference range		2.08	-	V_{DDAD}	V
I_{REF}	VREF - Reference Supply Current	Enabled	-	200	-	µA
		Disabled or Sleep Mode	-	< 0.01	0.02	µA
V_{INDC}	Analog Input Voltage[1]		V_{SSAD} - 0.3	-	V_{DDAD} + 0.3	V

[1] Maximum electrical operating range, not valid conversion range.

Table 1-05. ADC Timing/Performance Characteristics[1]

Symbol	Characteristic	Condition	Min	Typical	Max	Unit
R_{AS}	Source Impedance at Input[2]	2.08V ≤ V_{DDAD} ≤ 3.6V	-	-	10	kΩ
V_{AIN}	Analog Input Voltage[3]		V_{REFL}		V_{REFH}	V
RES	Ideal Resolution (1 LSB)[4]		2.031	-	3.516	mV
DNL	Differential Non-linearity[5]		-	±0.5	±1.0	LSB
INL	Integral Non-linearity[6]		-	±0.5	±1.0	LSB
E_{ZS}	Zero-scale Error[7]		-	±0.4	±1.0	LSB
F_{FS}	Full-scale Error[8]		-	±0.4	±1.0	LSB
E_{IL}	Input Leakage Error[9]		-	±0.05	±5.0	LSB
E_{TU}	Total Unadjusted Error[10]		-	±1.1	±2.5	LSB

[1] All ACCURACY numbers are based on processor and system being in WAIT state (very little activity and no IO switching) and that adequate low-pass filtering is present on analog input pins (filter with 0.01 µF to 0.1 µF capacitor between analog input and VREFL). Failure to observe these guidelines may result in system or microcontroller noise causing accuracy errors which will vary based on board layout and the type and magnitude of the activity.

Data transmission and reception during data conversion may cause some degradation of these specifications, depending on the number and timing of packets. It is advisable to test the ADCs in your installation if best accuracy is required.

[2] R_{AS} is the real portion of the impedance of the network driving the analog input pin. Values greater than this amount may not fully charge the input circuitry of the ATD resulting in accuracy error.

[3] Analog input must be between V_{REFL} and V_{REFH} for valid conversion. Values greater than V_{REFH} will convert to $3FF.

[4] The resolution is the ideal step size or 1LSB = $(V_{REFH}–V_{REFL})/1024$

[5] Differential non-linearity is the difference between the current code width and the ideal code width (1LSB). The current code width is the difference in the transition voltages to and from the current code.

[6] Integral non-linearity is the difference between the transition voltage to the current code and the adjusted ideal transition voltage for the current code. The adjusted ideal transition voltage is $(\text{Current Code}–1/2)*(1/((V_{REFH}+E_{FS})–(V_{REFL}+E_{ZS})))$.

[7] Zero-scale error is the difference between the transition to the first valid code and the ideal transition to that code. The Ideal transition voltage to a given code is $(\text{Code}–1/2)*(1/(V_{REFH}–V_{REFL}))$.

[8] Full-scale error is the difference between the transition to the last valid code and the ideal transition to that code. The ideal transition voltage to a given code is $(\text{Code}–1/2)*(1/(V_{REFH}–V_{REFL}))$.

[9] Input leakage error is error due to input leakage across the real portion of the impedance of the network driving the analog pin. Reducing the impedance of the network reduces this error.

[10] Total unadjusted error is the difference between the transition voltage to the current code and the ideal straight-line transfer function. This measure of error includes inherent quantization error (1/2LSB) and circuit error (differential, integral, zero-scale, and full-scale) error. The specified value of E_{TU} assumes zero E_{IL} (no leakage or zero real source impedance).

2. RF Module Operation

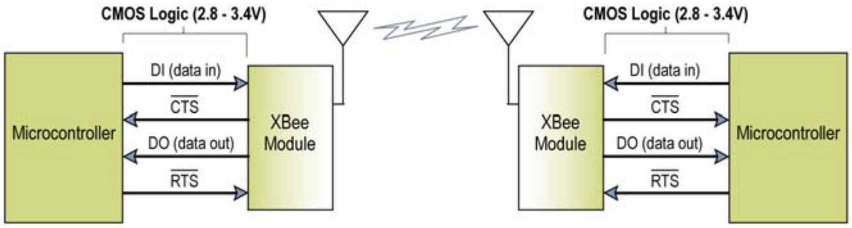
2.1. Serial Communications

The XBee/XBee-PRO OEM RF Modules interface to a host device through a logic-level asynchronous serial port. Through its serial port, the module can communicate with any logic and voltage compatible UART; or through a level translator to any serial device (For example: Through a MaxStream proprietary RS-232 or USB interface board).

2.1.1. UART Data Flow

Devices that have a UART interface can connect directly to the pins of the RF module as shown in the figure below.

Figure 2-01. System Data Flow Diagram in a UART-interfaced environment
(Low-asserted signals distinguished with horizontal line over signal name.)

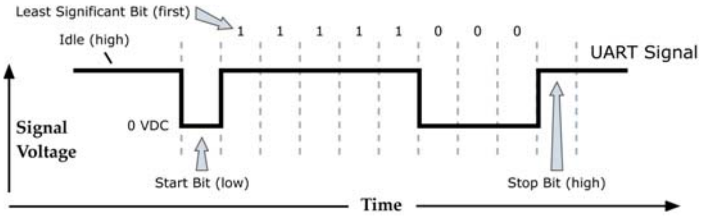


Serial Data

Data enters the module UART through the DI pin (pin 3) as an asynchronous serial signal. The signal should idle high when no data is being transmitted.

Each data byte consists of a start bit (low), 8 data bits (least significant bit first) and a stop bit (high). The following figure illustrates the serial bit pattern of data passing through the module.

Figure 2-02. UART data packet 0x1F (decimal number "31") as transmitted through the RF module
Example Data Format is 8-N-1 (bits - parity - # of stop bits)



The module UART performs tasks, such as timing and parity checking, that are needed for data communications. Serial communications depend on the two UARTs to be configured with compatible settings (baud rate, parity, start bits, stop bits, data bits).

2.1.2. Transparent Operation

By default, XBee/XBee-PRO RF Modules operate in Transparent Mode. When operating in this mode, the modules act as a serial line replacement - all UART data received through the DI pin is queued up for RF transmission. When RF data is received, the data is sent out the DO pin.

Serial-to-RF Packetization

Data is buffered in the DI buffer until one of the following causes the data to be packetized and transmitted:

1. No serial characters are received for the amount of time determined by the RO (Packetization Timeout) parameter. If RO = 0, packetization begins when a character is received.
2. The maximum number of characters that will fit in an RF packet (100) is received.
3. The Command Mode Sequence (GT + CC + GT) is received. Any character buffered in the DI buffer before the sequence is transmitted.

If the module cannot immediately transmit (for instance, if it is already receiving RF data), the serial data is stored in the DI Buffer. The data is packetized and sent at any RO timeout or when 100 bytes (maximum packet size) are received.

If the DI buffer becomes full, hardware or software flow control must be implemented in order to prevent overflow (loss of data between the host and module).

2.1.3. API Operation

API (Application Programming Interface) Operation is an alternative to the default Transparent Operation. The frame-based API extends the level to which a host application can interact with the networking capabilities of the module.

When in API mode, all data entering and leaving the module is contained in frames that define operations or events within the module.

Transmit Data Frames (received through the DI pin (pin 3)) include:

- RF Transmit Data Frame
- Command Frame (equivalent to AT commands)

Receive Data Frames (sent out the DO pin (pin 2)) include:

- RF-received data frame
- Command response
- Event notifications such as reset, associate, disassociate, etc.

The API provides alternative means of configuring modules and routing data at the host application layer. A host application can send data frames to the module that contain address and payload information instead of using command mode to modify addresses. The module will send data frames to the application containing status packets; as well as source, RSSI and payload information from received data packets.

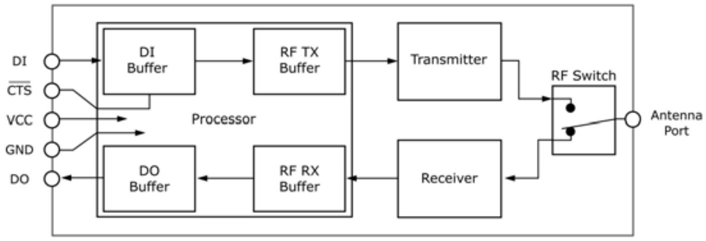
The API operation option facilitates many operations such as the examples cited below:

-> Transmitting data to multiple destinations without entering Command Mode
-> Receive success/failure status of each transmitted RF packet
-> Identify the source address of each received packet

To implement API operations, refer to API sections [p54].

2.1.4. Flow Control

Figure 2-03. Internal Data Flow Diagram



DI (Data In) Buffer

When serial data enters the RF module through the DI pin (pin 3), the data is stored in the DI Buffer until it can be processed.

Hardware Flow Control ($\overline{\text{CTS}}$). When the DI buffer is 17 bytes away from being full; by default, the module de-asserts $\overline{\text{CTS}}$ (high) to signal to the host device to stop sending data [refer to D7 (DIO7 Configuration) parameter]. $\overline{\text{CTS}}$ is re-asserted after the DI Buffer has 34 bytes of memory available.

How to eliminate the need for flow control:

1. Send messages that are smaller than the DI buffer size.
2. Interface at a lower baud rate [BD (Interface Data Rate) parameter] than the throughput data rate.

Case in which the DI Buffer may become full and possibly overflow:

If the module is receiving a continuous stream of RF data, any serial data that arrives on the DI pin is placed in the DI Buffer. The data in the DI buffer will be transmitted over-the-air when the module is no longer receiving RF data in the network.

Refer to the RO (Packetization Timeout), BD (Interface Data Rate) and D7 (DIO7 Configuration) command descriptions for more information.

DO (Data Out) Buffer

When RF data is received, the data enters the DO buffer and is sent out the serial port to a host device. Once the DO Buffer reaches capacity, any additional incoming RF data is lost.

Hardware Flow Control ($\overline{\text{RTS}}$). If $\overline{\text{RTS}}$ is enabled for flow control (D6 (DIO6 Configuration) Parameter = 1), data will not be sent out the DO Buffer as long as $\overline{\text{RTS}}$ (pin 16) is de-asserted.

Two cases in which the DO Buffer may become full and possibly overflow:

1. If the RF data rate is set higher than the interface data rate of the module, the module will receive data from the transmitting module faster than it can send the data to the host.
2. If the host does not allow the module to transmit data out from the DO buffer because of being held off by hardware or software flow control.

Refer to the D6 (DIO6 Configuration) command description for more information.

2.2. ADC and Digital I/O Line Support

The XBee/XBee-PRO RF Modules support ADC (Analog-to-digital conversion) and digital I/O line passing. The following pins support multiple functions:

Table 2-01. Pin functions and their associated pin numbers and commands
AD = Analog-to-Digital Converter, DIO = Digital Input/Output
Pin functions not applicable to this section are denoted within (parenthesis).

Pin Function	Pin#	AT Command
AD0 / DIO0	20	D0
AD1 / DIO1	19	D1
AD2 / DIO2	18	D2
AD3 / DIO3 / (COORD_SEL)	17	D3
AD4 / DIO4	11	D4
AD5 / DIO5 / (ASSOCIATE)	15	D5
DIO6 / (RTS)	16	D6
DIO7 / (CTS)	12	D7
DI8 / (DTR) / (Sleep_RQ)	9	D8

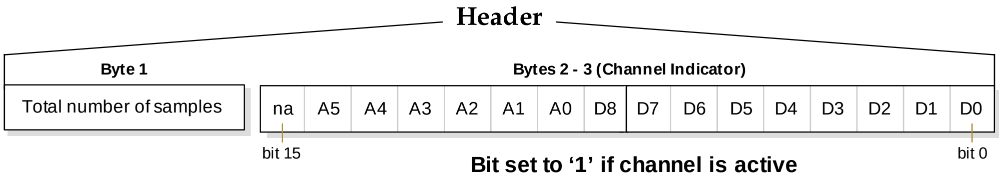
To enable ADC and DIO pin functions:

For ADC Support:	Set ATDn = 2
For Digital Input support:	Set ATDn = 3
For Digital Output Low support:	Set ATDn = 4
For Digital Output High support:	Set ATDn = 5

2.2.1. I/O Data Format

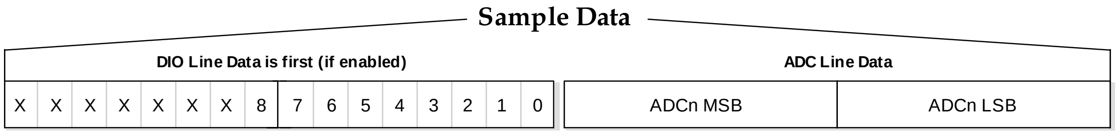
I/O data begins with a header. The first byte of the header defines the number of samples forthcoming. A sample is comprised of input data and the inputs can contain either DIO or ADC. The last 2 bytes of the header (Channel Indicator) define which inputs are active. Each bit represents either a DIO line or ADC channel.

Figure 2-04. Header



Sample data follows the header and the channel indicator frame is used to determine how to read the sample data. If any of the DIO lines are enabled, the first 2 bytes are the DIO data and the ADC data follows. ADC channel data is stored as an unsigned 10-bit value right-justified on a 16-bit boundary.

Figure 2-05. Sample Data



2.2.2. API Support

I/O data is sent out the UART using an API frame. All other data can be sent and received using Transparent Operation [refer to p10] or API framing if API mode is enabled (AP > 0).

API Operations support two RX (Receive) frame identifiers for I/O data:

- 0x82 for RX (Receive) Packet: 64-bit address I/O
- 0x83 for RX (Receive) Packet: 16-bit address I/O

The API command header is the same as shown in the “RX (Receive) Packet: 64-bit Address” and “RX (Receive) Packet: 64-bit Address” API types [refer to p58]. RX data follows the format described in the I/O Data Format section [p12].

Applicable Commands: AP (API Enable)

2.2.3. Sleep Support

When an RF module wakes, it will always do a sample based on any active ADC or DIO lines. This allows sampling based on the sleep cycle whether it be Cyclic Sleep (SM parameter = 4 or 5) or Pin Sleep (SM = 1 or 2). To gather more samples when awake, set the IR (Sample Rate) parameter.

For Cyclic Sleep modes: If the IR parameter is set, the module will stay awake until the IT (Samples before TX) parameter is met. The module will stay awake for ST (Time before Sleep) time.

Applicable Commands: IR (Sample Rate), IT (Samples before TX), SM (Sleep Mode), IC (DIO Change Detect)

2.2.4. DIO Pin Change Detect

When “DIO Change Detect” is enabled (using the IC command), DIO lines 0-7 are monitored. When a change is detected on a DIO line, the following will occur:

1. An RF packet is sent with the updated DIO pin levels. This packet will not contain any ADC samples.
2. Any queued samples are transmitted before the change detect data. This may result in receiving a packet with less than IT (Samples before TX) samples.

Note: Change detect will not affect Pin Sleep wake-up. The D8 pin (DTR/Sleep_RQ/DI8) is the only line that will wake a module from Pin Sleep. If not all samples are collected, the module will still enter Sleep Mode after a change detect packet is sent.

Applicable Commands: IC (DIO Change Detect), IT (Samples before TX)

NOTE: Change detect is only supported when the Dx (DIOx Configuration) parameter equals 3,4 or 5.

2.2.5. Sample Rate (Interval)

The Sample Rate (Interval) feature allows enabled ADC and DIO pins to be read periodically on modules that are not configured to operate in Sleep Mode. When one of the Sleep Modes is enabled and the IR (Sample Rate) parameter set, the module will stay awake until IT (Samples before TX) samples have been collected.

Once a particular pin is enabled, the appropriate sample rate must be chosen. The maximum sample rate that can be achieved while using one A/D line is 1 sample/ms or 1 KHz (Note that the modem will not be able to keep up with transmission when IR & IT are equal to “1”).

Applicable Commands: IR (Sample Rate), IT (Samples before TX), SM (Sleep Mode)

2.2.6. I/O Line Passing

Virtual wires can be set up between XBee/XBee-PRO Modules. When an RF data packet is received that contains I/O data, the receiving module can be setup to update any enabled outputs (PWM and DIO) based on the data it receives.

Note that I/O lines are mapped in pairs. For example: AD0 can only update PWM0 and DI5 can only update DO5). The default setup is for outputs not to be updated, which results in the I/O data being sent out the UART (refer to the IU (Enable I/O Output) command). To enable the outputs to be updated, the IA (I/O Input Address) parameter must be setup with the address of the module that has the appropriate inputs enabled. This effectively binds the outputs to a particular module's input. This does not affect the ability of the module to receive I/O line data from other modules - only its ability to update enabled outputs. The IA parameter can also be setup to accept I/O data for output changes from any module by setting the IA parameter to 0xFFFF.

When outputs are changed from their non-active state, the module can be setup to return the output level to it non-active state. The timers are set using the Tn (Dn Output Timer) and PT (PWM Output Timeout) commands. The timers are reset every time a valid I/O packet (passed IA check) is received. The IC (Change Detect) and IR (Sample Rate) parameters can be setup to keep the output set to their active output if the system needs more time than the timers can handle.

Note: DI8 can not be used for I/O line passing.

Applicable Commands: IA (I/O Input Address), Tn (Dn Output Timeout), P0 (PWM0 Configuration), P1 (PWM1 Configuration), M0 (PWM0 Output Level), M1 (PWM1 Output Level), PT (PWM Output Timeout), RP (RSSSI PWM Timer)

2.2.7. Configuration Example

As an example for a simple A/D link, a pair of RF modules could be set as follows:

Remote Configuration	Base Configuration

DL = 0x1234
MY = 0x5678
D0 = 2
D1 = 2
IR = 0x14
IT = 5
	DL = 0x5678
	MY = 0x1234
	P0 = 2
	P1 = 2
	IU = 1
	IA = 0x5678 (or 0xFFFF)

These settings configure the remote module to sample AD0 and AD1 once each every 20 ms. It then buffers 5 samples each before sending them back to the base module. The base should then receive a 32-Byte transmission (20 Bytes data and 12 Bytes framing) every 100 ms.

2.3. XBee/XBee-PRO Networks

The following IEEE 802.15.4 network types are supported by the XBee/XBee-PRO RF modules:

- NonBeacon
- NonBeacon (w/ Coordinator)

The following terms will be used to explicate the network operations:

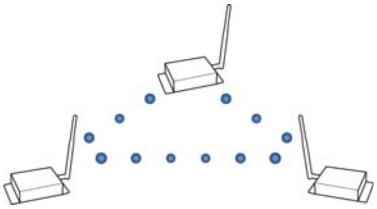
Table 2-02. Terms and definitions

Term	Definition
PAN	Personal Area Network - A data communication network that includes one or more End Devices and optionally a Coordinator.
Coordinator	A Full-function device (FFD) that provides network synchronization by polling nodes [NonBeacon (w/ Coordinator) networks only]
End Device	*When in the same network as a Coordinator* - RF modules that rely on a Coordinator for synchronization and can be put into states of sleep for low-power applications.
Association	The establishment of membership between End Devices and a Coordinator. Association is only applicable in NonBeacon (w/Coordinator) networks.

2.3.1. NonBeacon

By default, XBee/XBee-PRO RF Modules are configured to support NonBeacon communications. NonBeacon systems operate within a Peer-to-Peer network topology and therefore are not dependent upon Master/Slave relationships. This means that modules remain synchronized without use of master/server configurations and each module in the network shares both roles of master and slave. MaxStream's peer-to-peer architecture features fast synchronization times and fast cold start times. This default configuration accommodates a wide range of RF data applications.

Figure 2-06. NonBeacon Peer-to-Peer Architecture



A peer-to-peer network can be established by configuring each module to operate as an End Device (CE = 0), disabling End Device Association on all modules (A1 = 0) and setting ID and CH parameters to be identical across the network.

2.3.2. NonBeacon (w/ Coordinator)

A device is configured as a Coordinator by setting the CE (Coordinator Enable) parameter to “1”. Coordinator power-up is governed by the A2 (Coordinator Association) parameter.

In a NonBeacon (w/ Coordinator) system, the Coordinator can be configured to use direct or indirect transmissions. If the SP (Cyclic Sleep Period) parameter is set to “0”, the Coordinator will send data immediately. Otherwise, the SP parameter determines the length of time the Coordinator will retain the data before discarding it. Generally, SP (Cyclic Sleep Period) and ST (Time before Sleep) parameters should be set to match the SP and ST settings of the End Devices.

Association plays a critical role in the implementation of a NonBeacon (w/ Coordinator) system. Refer to the Association section [next page] for more information.

2.3.3. Association

Association is the establishment of membership between End Devices and a Coordinator and is only applicable in NonBeacon (w/ Coordinator) networks. The establishment of membership is useful in scenarios that require a central unit (Coordinator) to relay messages to or gather data from several remote units (End Devices), assign channels or assign PAN IDs.

An RF data network that consists of one Coordinator and one or more End Devices forms a PAN (Personal Area Network). Each device in a PAN has a PAN Identifier [ID (PAN ID) parameter]. PAN IDs must be unique to prevent miscommunication between PANs. The Coordinator PAN ID is set using the ID (PAN ID) and A2 (Coordinator Association) commands.

An End Device can associate to a Coordinator without knowing the address, PAN ID or channel of the Coordinator. The A1 (End Device Association) parameter bit fields determine the flexibility of an End Device during association. The A1 parameter can be used for an End Device to dynamically set its destination address, PAN ID and/or channel.

> For example: If the PAN ID of a Coordinator is known, but the operating channel is not; the A1 command on the End Device should be set to enable the 'Auto_Associate' and 'Reassign_Channel' bits. Additionally, the ID parameter should be set to match the PAN ID of the associated Coordinator.

Coordinator / End Device Setup and Operation

To configure a module to operate as a Coordinator, set the CE (Coordinator Enable) parameter to '1'. Set the CE parameter of End Devices to '0' (default). Coordinator and End Devices should contain matching firmware versions.

NonBeacon (w/ Coordinator) Systems

In a NonBeacon (w/ Coordinator) system, the Coordinator can be configured to use direct or indirect transmissions. If the SP (Cyclic Sleep Period) parameter is set to '0', the Coordinator will send data immediately. Otherwise, the SP parameter determines the length of time the Coordinator will retain the data before discarding it. Generally, SP (Cyclic Sleep Period) and ST (Time before Sleep) parameters should be set to match the SP and ST settings of the End Devices.

Coordinator Power-up

Coordinator power-up is governed by the A2 (Coordinator Association) command. On power-up, the Coordinator undergoes the following sequence of events:

1. Check A2 parameter- Reassign_PANID Flag

Set (bit 0 = 1) - The Coordinator issues an Active Scan. The Active Scan selects one channel and transmits a BeaconRequest command to the broadcast address (0xFFFF) and broadcast PAN ID (0xFFFF). It then listens on that channel for beacons from any Coordinator operating on that channel. The listen time on each channel is determined by the SD (Scan Duration) parameter value.

Once the time expires on that channel, the Active Scan selects another channel and again transmits the BeaconRequest as before. This process continues until all channels have been scanned, or until 5 PANs have been discovered. When the Active Scan is complete, the results include a list of PAN IDs and Channels that are being used by other PANs. This list is used to assign an unique PAN ID to the new Coordinator. The ID parameter will be retained if it is not found in the Active Scan results. Otherwise, the ID (PAN ID) parameter setting will be updated to a PAN ID that was not detected.

Not Set (bit 0 = 0) - The Coordinator retains its ID setting. No Active Scan is performed.

2. Check A2 parameter - Reassign_Channel Flag (bit 1)

Set (bit 1 = 1) - The Coordinator issues an Energy Scan. The Energy Scan selects one channel and scans for energy on that channel. The duration of the scan is specified by the SD (Scan Duration) parameter. Once the scan is completed on a channel, the Energy Scan selects the next channel and begins a new scan on that channel. This process continues until all channels have been scanned.

When the Energy Scan is complete, the results include the maximal energy values detected on each channel. This list is used to determine a channel where the least energy was detected. If an Active Scan was performed (Reassign_PANID Flag set), the channels used by the detected PANs are eliminated as possible channels. Thus, the results of the Energy Scan and the Active Scan (if performed) are used to find the best channel (channel with the least energy that is not used by any detected PAN). Once the best channel has been selected, the CH (Channel) parameter value is updated to that channel.

Not Set (bit 1 = 0) - The Coordinator retains its CH setting. An Energy Scan is not performed.

3. Start Coordinator

The Coordinator starts on the specified channel (CH parameter) and PAN ID (ID parameter). Note, these may be selected in steps 1 and/or 2 above. The Coordinator will only allow End Devices to associate to it if the A2 parameter "AllowAssociation" flag is set. Once the Coordinator has successfully started, the Associate LED will blink 1 time per second. (The LED is solid if the Coordinator has not started.)

4. Coordinator Modifications

Once a Coordinator has started:
Modifying the A2 (Reassign_Channel or Reassign_PANID bits), ID, CH or MY parameters will cause the Coordinator's MAC to reset (The Coordinator RF module (including volatile RAM) is not reset). Changing the A2 AllowAssociation bit will not reset the Coordinator's MAC. In a non-beaconing system, End Devices that associated to the Coordinator prior to a MAC reset will have knowledge of the new settings on the Coordinator. Thus, if the Coordinator were to change its ID, CH or MY settings, the End Devices would no longer be able to communicate with the non-beacon Coordinator. Once a Coordinator has started, the ID, CH, MY or A2 (Reassign_Channel or Reassign_PANID bits) should not be changed.

End Device Power-up

End Device power-up is governed by the A1 (End Device Association) command. On power-up, the End Device undergoes the following sequence of events:

1. Check A1 parameter - AutoAssociate Bit

Set (bit 2 = 1) - End Device will attempt to associate to a Coordinator. (refer to steps 2-3).

Not Set (bit 2 = 0) - End Device will not attempt to associate to a Coordinator. The End Device will operate as specified by its ID, CH and MY parameters. Association is considered complete and the Associate LED will blink quickly (5 times per second). When the AutoAssociate bit is not set, the remaining steps (2-3) do not apply.

2. Discover Coordinator (if Auto-Associate Bit Set)

The End Device issues an Active Scan. The Active Scan selects one channel and transmits a BeaconRequest command to the broadcast address (0xFFFF) and broadcast PAN ID (0xFFFF). It then listens on that channel for beacons from any Coordinator operating on that channel. The listen time on each channel is determined by the SD parameter.

Once the time expires on that channel, the Active Scan selects another channel and again transmits the BeaconRequest command as before. This process continues until all channels have been scanned, or until 5 PANs have been discovered. When the Active Scan is complete, the results include a list of PAN IDs and Channels that are being used by detected PANs.

The End Device selects a Coordinator to associate with according to the A1 parameter "Reassign_PANID" and "Reassign_Channel" flags:

Reassign_PANID Bit Set (bit 0 = 1)- End Device can associate with a PAN with any ID value.

Reassign_PANID Bit Not Set (bit 0 = 0) - End Device will only associate with a PAN whose ID setting matches the ID setting of the End Device.

Reassign_Channel Bit Set (bit 1 = 1) - End Device can associate with a PAN with any CH value.

Reassign_Channel Bit Not Set (bit 1 = 0)- End Device will only associate with a PAN whose CH setting matches the CH setting of the End Device.

After applying these filters to the discovered Coordinators, if multiple candidate PANs exist, the End Device will select the PAN whose transmission link quality is the strongest. If no valid Coordinator is found, the End Device will either go to sleep (as dictated by its SM (Sleep Mode) parameter) or retry Association.

Note - An End Device will also disqualify Coordinators if they are not allowing association (A2 - AllowAssociation bit); or, if the Coordinator is not using the same NonBeacon scheme as the End Device. (They must both be programmed with NonBeacon code.)

3. Associate to Valid Coordinator

Once a valid Coordinator is found (step 2), the End Device sends an AssociationRequest message to the Coordinator. It then waits for an AssociationConfirmation to be sent from the Coordinator. Once the Confirmation is received, the End Device is Associated and the Associate LED will blink rapidly (2 times per second). The LED is solid if the End Device has not associated.

4. End Device Changes once an End Device has associated

Changing A1, ID or CH parameters will cause the End Device to disassociate and restart the Association procedure.

If the End Device fails to associate, the AI command can give some indication of the failure.

2.4. XBee / XBee-PRO Addressing

Every RF data packet sent over-the-air contains a Source Address and Destination Address field in its header. The RF module conforms to the 802.15.4 specification and supports both short 16-bit addresses and long 64-bit addresses. A unique 64-bit IEEE source address is assigned at the factory and can be read with the SL (Serial Number Low) and SH (Serial Number High) commands. Short addressing must be configured manually. A module will use its unique 64-bit address as its Source Address if its MY (16-bit Source Address) value is “0xFFFF” or “0xFFFE”.

To send a packet to a specific module using 64-bit addressing: Set Destination Address (DL + DH) to match the Source Address (SL + SH) of the intended destination module.

To send a packet to a specific module using 16-bit addressing: Set DL (Destination Address Low) parameter to equal the MY parameter and set the DH (Destination Address High) parameter to ‘0’.

2.4.1. Unicast Mode

By default, the RF module operates in Unicast Mode. Unicast Mode is the only mode that supports retries. While in this mode, receiving modules send an ACK (acknowledgement) of RF packet reception to the transmitter. If the transmitting module does not receive the ACK, it will re-send the packet up to three times or until the ACK is received.

Short 16-bit addresses. The module can be configured to use short 16-bit addresses as the Source Address by setting (MY < 0xFFFE). Setting the DH parameter (DH = 0) will configure the Destination Address to be a short 16-bit address (if DL < 0xFFFE). For two modules to communicate using short addressing, the Destination Address of the transmitter module must match the MY parameter of the receiver.

The following table shows a sample network configuration that would enable Unicast Mode communications using short 16-bit addresses.

Table 2-03. Sample Unicast Network Configuration (using 16-bit addressing)

Parameter	RF Module 1	RF Module 2
MY (Source Address)	0x01	0x02
DH (Destination Address High)	0	0
DL (Destination Address Low)	0x02	0x01

Long 64-bit addresses. The RF module’s serial number (SL parameter concatenated to the SH parameter) can be used as a 64-bit source address when the MY (16-bit Source Address) parameter is disabled. When the MY parameter is disabled (set MY = 0xFFFF or 0xFFFE), the module’s source address is set to the 64-bit IEEE address stored in the SH and SL parameters.

When an End Device associates to a Coordinator, its MY parameter is set to 0xFFFE to enable 64-bit addressing. The 64-bit address of the module is stored as SH and SL parameters. To send a packet to a specific module, the Destination Address (DL + DH) on one module must match the Source Address (SL + SH) of the other.

2.4.2. Broadcast Mode

Any RF module within range will accept a packet that contains a broadcast address. When configured to operate in Broadcast Mode, receiving modules do not send ACKs (Acknowledgements) and transmitting modules do not automatically re-send packets as is the case in Unicast Mode.

To send a broadcast packet to all modules regardless of 16-bit or 64-bit addressing, set the destination addresses of all the modules as shown below.

Sample Network Configuration (All modules in the network):

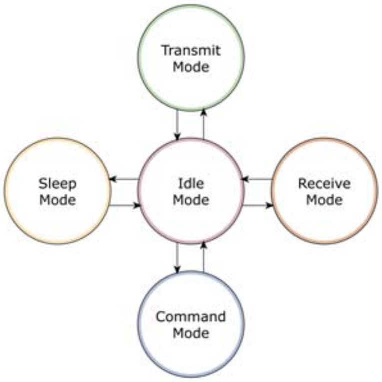
- DL (Destination Low Address) = 0x0000FFFF
- DH (Destination High Address) = 0x00000000 (default value)

NOTE: When programming the module, parameters are entered in hexadecimal notation (without the “0x” prefix). Leading zeros may be omitted.

2.5. Modes of Operation

XBee/XBee-PRO RF Modules operate in five modes.

Figure 2-07. Modes of Operation



2.5.1. Idle Mode

When not receiving or transmitting data, the RF module is in Idle Mode. The module shifts into the other modes of operation under the following conditions:

- Transmit Mode (Serial data is received in the DI Buffer)
- Receive Mode (Valid RF data is received through the antenna)
- Sleep Mode (Sleep Mode condition is met)
- Command Mode (Command Mode Sequence is issued)

2.5.2. Transmit/Receive Modes

RF Data Packets

Each transmitted data packet contains a Source Address and Destination Address field. The Source Address matches the address of the transmitting module as specified by the MY (Source Address) parameter (if MY >= 0xFFFE), the SH (Serial Number High) parameter or the SL (Serial Number Low) parameter. The <Destination Address> field is created from the DH (Destination Address High) and DL (Destination Address Low) parameter values. The Source Address and/or Destination Address fields will either contain a 16-bit short or long 64-bit long address.

The RF data packet structure follows the 802.15.4 specification.

[Refer to the XBee/XBee-PRO Addressing section for more information]

Direct and Indirect Transmission

There are two methods to transmit data:

- Direct Transmission - data is transmitted immediately to the Destination Address
- Indirect Transmission - A packet is retained for a period of time and is only transmitted after the destination module (Source Address = Destination Address) requests the data.

Indirect Transmissions can only occur on a Coordinator. Thus, if all nodes in a network are End Devices, only Direct Transmissions will occur. Indirect Transmissions are useful to ensure packet delivery to a sleeping node. The Coordinator currently is able to retain up to 2 indirect messages.

Direct Transmission

A NonBeaconing Coordinator can be configured to use only Direct Transmission by setting the SP (Cyclic Sleep Period) parameter to “0”. Also, a NonBeaconing Coordinator using indirect transmissions will revert to direct transmission if it knows the destination module is awake.

To enable this behavior, the ST (Time before Sleep) value of the Coordinator must be set to match the ST value of the End Device. Once the End Device either transmits data to the Coordinator or polls the Coordinator for data, the Coordinator will use direct transmission for all subsequent data transmissions to that module address until ST time (or number of beacons) occurs with no activity (at which point it will revert to using indirect transmissions for that module address). “No activity” means no transmission or reception of messages with a specific address. Global messages will not reset the ST timer.

Indirect Transmission

To configure Indirect Transmissions in a PAN (Personal Area Network), the SP (Cyclic Sleep Period) parameter value on the Coordinator must be set to match the longest sleep value of any End Device. The SP parameter represents time in NonBeacon systems and beacons in Beacon-enabled systems. The sleep period value on the Coordinator determines how long (time or number of beacons) the Coordinator will retain an indirect message before discarding it.

In NonBeacon networks, an End Device must poll the Coordinator once it wakes from Sleep to determine if the Coordinator has an indirect message for it. For Cyclic Sleep Modes, this is done automatically every time the module wakes (after SP time). For Pin Sleep Modes, the A1 (End Device Association) parameter value must be set to enable Coordinator polling on pin wake-up. Alternatively, an End Device can use the FP (Force Poll) command to poll the Coordinator as needed.

CCA (Clear Channel Assessment)

Prior to transmitting a packet, a CCA (Clear Channel Assessment) is performed on the channel to determine if the channel is available for transmission. The detected energy on the channel is compared with the CA (Clear Channel Assessment) parameter value. If the detected energy exceeds the CA parameter value, the packet is not transmitted.

Also, a delay is inserted before a transmission takes place. This delay is settable using the RN (Backoff Exponent) parameter. If RN is set to “0”, then there is no delay before the first CCA is performed. The RN parameter value is the equivalent of the “minBE” parameter in the 802.15.4 specification. The transmit sequence follows the 802.15.4 specification.

By default, the MM (MAC Mode) parameter = 0. On a CCA failure, the module will attempt to re-send the packet up to two additional times.

When in Unicast packets with RR (Retries) = 0, the module will execute two CCA retries. Broadcast packets always get two CCA retries.

Acknowledgement

If the transmission is not a broadcast message, the module will expect to receive an acknowledgement from the destination node. If an acknowledgement is not received, the packet will be resent up to 3 more times. If the acknowledgement is not received after all transmissions, an ACK failure is recorded.

2.5.3. Sleep Mode

Sleep Modes enable the RF module to enter states of low-power consumption when not in use. In order to enter Sleep Mode, one of the following conditions must be met (in addition to the module having a non-zero SM parameter value):

- Sleep_RQ (pin 9) is asserted.
- The module is idle (no data transmission or reception) for the amount of time defined by the ST (Time before Sleep) parameter. [NOTE: ST is only active when SM = 4-5.]

Table 2-04. Sleep Mode Configurations

Sleep Mode Setting	Transition into Sleep Mode	Transition out of Sleep Mode (wake)	Characteristics	Related Commands	Power Consumption
Pin Hibernate (SM = 1)	Assert (high) Sleep_RQ (pin 9)	De-assert (low) Sleep_RQ	Pin/Host-controlled / NonBeacon systems only / Lowest Power	(SM)	< 10 µA (@3.0 VCC)
Pin Doze (SM = 2)	Assert (high) Sleep_RQ (pin 9)	De-assert (low) Sleep_RQ	Pin/Host-controlled / NonBeacon systems only / Fastest wake-up	(SM)	< 50 µA
Cyclic Sleep (SM = 4 - 5)	Automatic transition to Sleep Mode as defined by the SM (Sleep Mode) and ST (Time before Sleep) parameters.	Transition occurs after the cyclic sleep time interval elapses. The time interval is defined by the SP (Cyclic Sleep Period) parameter.	RF module wakes in pre-determined time intervals to detect if RF data is present / When SM = 5, NonBeacon systems only	(SM), SP, ST	< 50 µA when sleeping

The SM command is central to setting Sleep Mode configurations. By default, Sleep Modes are disabled (SM = 0) and the module remains in Idle/Receive Mode. When in this state, the module is constantly ready to respond to serial or RF activity.

Higher Voltages. Sleep Mode current consumption is highly sensitive to voltage. Voltages above 3.0V will cause much higher current consumption.

Table 2-05. Sample Sleep Mode Currents

	XBee			XBee-PRO		
Vcc (V)	SM=1	SM=2	SM=4,5	SM=1	SM=2	SM=4,5
2.8–3.0	<3 µA	<35uA	<34uA	<4uA	<34uA	<34uA
3.1	8uA	37mA	36uA	12uA	39uA	37uA
3.2	32uA	48uA	49uA	45uA	60uA	55uA
3.3	101uA	83uA	100uA	130uA	115uA	120uA
3.4	255uA	170uA	240uA	310uA	260uA	290uA

Pin / Host-controlled Sleep Modes

The transient current when waking from pin sleep (SM = 1 or 2) does not exceed the idle current of the module. The current ramps up exponentially to its idle current.

Pin Hibernate (SM = 1)

- Pin/Host-controlled
- Typical power-down current: < 10 µA (@3.0 VCC)
- Wake-up time: 13.2 msec

Pin Hibernate Mode minimizes quiescent power (power consumed when in a state of rest or inactivity). This mode is voltage level-activated; when Sleep_RQ is asserted, the module will finish any transmit, receive or association activities, enter Idle Mode and then enter a state of sleep. The module will not respond to either serial or RF activity while in pin sleep.

To wake a sleeping module operating in Pin Hibernate Mode, de-assert Sleep_RQ (pin 9). The module will wake when Sleep_RQ is de-asserted and is ready to transmit or receive when the $\overline{\text{CTS}}$ line is low. When waking the module, the pin must be de-asserted at least two 'byte times' after $\overline{\text{CTS}}$ goes low. This assures that there is time for the data to enter the DI buffer.

Pin Doze (SM = 2)

- Pin/Host-controlled
- Typical power-down current: < 50 µA
- Wake-up time: 2 msec

Pin Doze Mode functions as does Pin Hibernate Mode; however, Pin Doze features faster wake-up time and higher power consumption.

To wake a sleeping module operating in Pin Doze Mode, de-assert Sleep_RQ (pin 9). The module will wake when Sleep_RQ is de-asserted and is ready to transmit or receive when the $\overline{\text{CTS}}$ line is low. When waking the module, the pin must be de-asserted at least two 'byte times' after $\overline{\text{CTS}}$ goes low. This assures that there is time for the data to enter the DI buffer.

Cyclic Sleep Modes

Cyclic Sleep Remote (SM = 4)

- Typical Power-down Current: < 50 µA (when asleep)
- Wake-up time: 2 msec

The Cyclic Sleep Modes allow modules to periodically check for RF data. When the SM parameter is set to ‘4’, the module is configured to sleep, then wakes once a cycle to check for data from a module configured as a Cyclic Sleep Coordinator (SM = 0, CE = 1). The Cyclic Sleep Remote sends a poll request to the coordinator at a specific interval set by the SP (Cyclic Sleep Period) parameter. The coordinator will transmit any queued data addressed to that specific remote upon receiving the poll request.

If no data is queued for the remote, the coordinator will not transmit and the remote will return to sleep for another cycle. If queued data is transmitted back to the remote, it will stay awake to allow for back and forth communication until the ST (Time before Sleep) timer expires.

Also note that $\overline{\text{CTS}}$ will go low each time the remote wakes, allowing for communication initiated by the remote host if desired.

Cyclic Sleep Remote with Pin Wake-up (SM = 5)

Use this mode to wake a sleeping remote module through either the RF interface or by the de-assertion of Sleep_RQ for event-driven communications. The cyclic sleep mode works as described above (Cyclic Sleep Remote) with the addition of a pin-controlled wake-up at the remote module. The Sleep_RQ pin is edge-triggered, not level-triggered. The module will wake when a low is detected then set $\overline{\text{CTS}}$ low as soon as it is ready to transmit or receive.

Any activity will reset the ST (Time before Sleep) timer so the module will go back to sleep only after there is no activity for the duration of the timer. Once the module wakes (pin-controlled), further pin activity is ignored. The module transitions back into sleep according to the ST time regardless of the state of the pin.

[Cyclic Sleep Coordinator (SM = 6)]

- Typical current = Receive current
- Always awake

NOTE: The SM=6 parameter value exists solely for backwards compatibility with firmware version 1.x60. If backwards compatibility with the older firmware version is not required, always use the CE (Coordinator Enable) command to configure a module as a Coordinator.

This mode configures a module to wake cyclic sleeping remotes through RF interfacing. The Coordinator will accept a message addressed to a specific remote 16 or 64-bit address and hold it in a buffer until the remote wakes and sends a poll request. Messages not sent directly (buffered and requested) are called "Indirect messages". The Coordinator only queues one indirect message at a time. The Coordinator will hold the indirect message for a period 2.5 times the sleeping period indicated by the SP (Cyclic Sleep Period) parameter. The Coordinator's SP parameter should be set to match the value used by the remotes.

2.5.4. Command Mode

To modify or read RF Module parameters, the module must first enter into Command Mode - a state in which incoming characters are interpreted as commands. Two Command Mode options are supported: AT Command Mode [refer to section below] and API Command Mode [p54].

AT Command Mode

To Enter AT Command Mode:

Send the 3-character command sequence "+++" and observe guard times before and after the command characters. [Refer to the "Default AT Command Mode Sequence" below.]

Default AT Command Mode Sequence (for transition to Command Mode):

- No characters sent for one second [GT (Guard Times) parameter = 0x3E8]
- Input three plus characters ("+++") within one second [CC (Command Sequence Character) parameter = 0x2B.]
- No characters sent for one second [GT (Guard Times) parameter = 0x3E8]

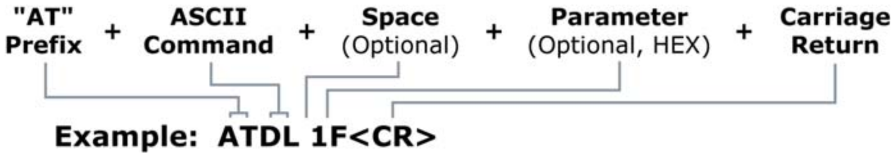
All of the parameter values in the sequence can be modified to reflect user preferences.

NOTE: Failure to enter AT Command Mode is most commonly due to baud rate mismatch. Ensure the 'Baud' setting on the "PC Settings" tab matches the interface data rate of the RF module. By default, the BD parameter = 3 (9600 bps).

To Send AT Commands:

Send AT commands and parameters using the syntax shown below.

Figure 2-08. Syntax for sending AT Commands



To read a parameter value stored in the RF module's register, omit the parameter field.

The preceding example would change the RF module Destination Address (Low) to "0x1F". To store the new value to non-volatile (long term) memory, subsequently send the WR (Write) command.

For modified parameter values to persist in the module's registry after a reset, changes must be saved to non-volatile memory using the WR (Write) Command. Otherwise, parameters are restored to previously saved values after the module is reset.

System Response. When a command is sent to the module, the module will parse and execute the command. Upon successful execution of a command, the module returns an "OK" message. If execution of a command results in an error, the module returns an "ERROR" message.

To Exit AT Command Mode:

1. Send the ATCN (Exit Command Mode) command (followed by a carriage return).

   [OR]

2. If no valid AT Commands are received within the time specified by CT (Command Mode Timeout) Command, the RF module automatically returns to Idle Mode.

For an example of programming the RF module using AT Commands and descriptions of each configurable parameter, refer to the RF Module Configuration chapter [p25].

3. RF Module Configuration

3.1. Programming the RF Module

Refer to the Command Mode section [p24] for more information about entering Command Mode, sending AT commands and exiting Command Mode. For information regarding module programming using API Mode, refer to the API Operation sections [p54].

3.1.1. Programming Examples

Refer to the 'X-CTU' section of the Development Guide [Appendix B] for more information regarding the X-CTU configuration software.

Setup

The programming examples in this section require the installation of MaxStream's X-CTU Software and a serial connection to a PC. (MaxStream stocks RS-232 and USB boards to facilitate interfacing with a PC.)

1. Install MaxStream's X-CTU Software to a PC by double-clicking the "setup_X-CTU.exe" file. (The file is located on the MaxStream CD and under the 'Software' section of the following web page: www.maxstream.net/support/downloads.php)
2. Mount the RF module to an interface board, then connect the module assembly to a PC.
3. Launch the X-CTU Software and select the 'PC Settings' tab. Verify the baud and parity settings of the Com Port match those of the RF module.

NOTE: Failure to enter AT Command Mode is most commonly due to baud rate mismatch. Ensure the 'Baud' setting on the 'PC Settings' tab matches the interface data rate of the RF module. By default, the BD parameter = 3 (which corresponds to 9600 bps).

Sample Configuration: Modify RF Module Destination Address

Example: Utilize the X-CTU "Terminal" tab to change the RF module's DL (Destination Address Low) parameter and save the new address to non-volatile memory.

After establishing a serial connection between the RF module and a PC [refer to the 'Setup' section above], select the "Terminal" tab of the X-CTU Software and enter the following command lines ('CR' stands for carriage return):

Method 1 (One line per command)

Send AT Command	**System Response**
+++	OK <CR> (Enter into Command Mode)
ATDL <Enter>	{current value} <CR> (Read Destination Address Low)
ATDL1A0D <Enter>	OK <CR> (Modify Destination Address Low)
ATWR <Enter>	OK <CR> (Write to non-volatile memory)
ATCN <Enter>	OK <CR> (Exit Command Mode)

Method 2 (Multiple commands on one line)

Send AT Command	**System Response**
+++	OK <CR> (Enter into Command Mode)
ATDL <Enter>	{current value} <CR> (Read Destination Address Low)
ATDL1A0D,WR,CN <Enter>	OK<CR> OK<CR> OK<CR>

Sample Configuration: Restore RF Module Defaults

Example: Utilize the X-CTU "Modem Configuration" tab to restore default parameter values.

After establishing a connection between the module and a PC [refer to the 'Setup' section above], select the "Modem Configuration" tab of the X-CTU Software.

1. Select the 'Read' button.
2. Select the 'Restore' button.

3.2. Command Reference Tables

XBee/XBee-PRO RF Modules expect numerical values in hexadecimal. Hexadecimal values are designated by a “0x” prefix. Decimal equivalents are designated by a “d” suffix. Commands are contained within the following command categories (listed in the order that their tables appear):

- Special
- Networking & Security
- RF Interfacing
- Sleep (Low Power)
- Serial Interfacing
- I/O Settings
- Diagnostics
- AT Command Options

All modules within a PAN should operate using the same firmware version.

Special

Table 3-01. XBee-PRO Commands - Special

AT Command	Command Category	Name and Description	Parameter Range	Default
WR	Special	Write. Write parameter values to non-volatile memory so that parameter modifications persist through subsequent power-up or reset. Note: Once WR is issued, no additional characters should be sent to the module until after the response "OK\r" is received.	-	-
RE	Special	Restore Defaults. Restore module parameters to factory defaults.	-	-
FR (v1.x80*)	Special	Software Reset. Responds immediately with an OK then performs a hard reset ~100ms later.	-	-

* Firmware version in which the command was first introduced (firmware versions are numbered in hexadecimal notation.)

Networking & Security

Table 3-02. XBee/XBee-PRO Commands - Networking & Security (Sub-categories designated within {brackets})

AT Command	Command Category	Name and Description	Parameter Range	Default
CH	Networking {Addressing}	Channel. Set/Read the channel number used for transmitting and receiving data between RF modules (uses 802.15.4 protocol channel numbers).	0x0B - 0x1A (XBee) 0x0C - 0x17 (XBee-PRO)	0x0C (12d)
ID	Networking {Addressing}	PAN ID. Set/Read the PAN (Personal Area Network) ID. Use 0xFFFF to broadcast messages to all PANs.	0 - 0xFFFF	0x3332 (13106d)
DH	Networking {Addressing}	Destination Address High. Set/Read the upper 32 bits of the 64-bit destination address. When combined with DL, it defines the destination address used for transmission. To transmit using a 16-bit address, set DH parameter to zero and DL less than 0xFFFF. 0x000000000000FFFF is the broadcast address for the PAN.	0 - 0xFFFFFFFF	0
DL	Networking {Addressing}	Destination Address Low. Set/Read the lower 32 bits of the 64-bit destination address. When combined with DH, DL defines the destination address used for transmission. To transmit using a 16-bit address, set DH parameter to zero and DL less than 0xFFFF. 0x000000000000FFFF is the broadcast address for the PAN.	0 - 0xFFFFFFFF	0
MY	Networking {Addressing}	16-bit Source Address. Set/Read the RF module 16-bit source address. Set MY = 0xFFFF to disable reception of packets with 16-bit addresses. 64-bit source address (serial number) and broadcast address (0x000000000000FFFF) is always enabled.	0 - 0xFFFF	0
SH	Networking {Addressing}	Serial Number High. Read high 32 bits of the RF module's unique IEEE 64-bit address. 64-bit source address is always enabled.	0 - 0xFFFFFFFF [read-only]	Factory-set
SL	Networking {Addressing}	Serial Number Low. Read low 32 bits of the RF module's unique IEEE 64-bit address. 64-bit source address is always enabled.	0 - 0xFFFFFFFF [read-only]	Factory-set
RR (v1.xA0*)	Networking {Addressing}	XBee Retries. Set/Read the maximum number of retries the module will execute in addition to the 3 retries provided by the 802.15.4 MAC. For each XBee retry, the 802.15.4 MAC can execute up to 3 retries.	0 - 6	0
RN	Networking {Addressing}	Random Delay Slots. Set/Read the minimum value of the back-off exponent in the CSMA-CA algorithm that is used for collision avoidance. If RN = 0, collision avoidance is disabled during the first iteration of the algorithm (802.15.4 - macMinBE).	0 - 3 [exponent]	0
MM (v1.x80*)	Networking {Addressing}	MAC Mode. Set/Read MAC Mode value. MAC Mode enables/disables the use of a MaxStream header in the 802.15.4 RF packet. When Mode 0 is enabled (MM=0), duplicate packet detection is enabled as well as certain AT commands. Modes 1 and 2 are strict 802.15.4 modes.	0 - 2 0 = MaxStream Mode 1 = 802.15.4 (no ACKs) 2 = 802.15.4 (with ACKs)	0

Table 3-02. XBee/XBee-PRO Commands - Networking & Security (Sub-categories designated within {brackets})

AT Command	Command Category	Name and Description	Parameter Range	Default
NI (v1.x80*)	Networking {Identification}	Node Identifier. Stores a string identifier. The register only accepts printable ASCII data. A string can not start with a space. Carriage return ends command. Command will automatically end when maximum bytes for the string have been entered. This string is returned as part of the ND (Node Discover) command. This identifier is also used with the DN (Destination Node) command.	20-character ASCII string	-
ND (v1.x80*)	Networking {Identification}	Node Discover. Discovers and reports all RF modules found. The following information is reported for each module discovered (the example cites use of Transparent operation (AT command format) - refer to the long ND command description regarding differences between Transparent and API operation). MY<CR> SH<CR> SL<CR> DB<CR> NI<CR><CR> The amount of time the module allows for responses is determined by the NT parameter. In Transparent operation, command completion is designated by a <CR> (carriage return). ND also accepts a Node Identifier as a parameter. In this case, only a module matching the supplied identifier will respond.	optional 20-character NI value	
NT (v1.xA0*)	Networking {Identification}	Node Discover Time. Set/Read the amount of time a node will wait for responses from other nodes when using the ND (Node Discover) command.	0x01 - 0xFC	0x19
DN (v1.x80*)	Networking {Identification}	Destination Node. Resolves an NI (Node Identifier) string to a physical address. The following events occur upon successful command execution: 1. DL and DH are set to the address of the module with the matching Node Identifier. 2. “OK” is returned. 3. RF module automatically exits AT Command Mode If there is no response from a module within 200 msec or a parameter is not specified (left blank), the command is terminated and an “ERROR” message is returned.	20-character ASCII string	-
CE (v1.x80*)	Networking {Association}	Coordinator Enable. Set/Read the coordinator setting.	0 - 1 0 = End Device 1 = Coordinator	0
SC (v1.x80*)	Networking {Association}	Scan Channels. Set/Read list of channels to scan for all Active and Energy Scans as a bitfield. This affects scans initiated in command mode (AS, ED) and during End Device Association and Coordinator startup: bit 0 - 0x0B bit 4 - 0x0F bit 8 - 0x13 bit12 - 0x17 bit 1 - 0x0C bit 5 - 0x10 bit 9 - 0x14 bit13 - 0x18 bit 2 - 0x0D bit 6 - 0x11 bit 10 - 0x15 bit14 - 0x19 bit 3 - 0x0E bit 7 - 0x12 bit 11 - 0x16 bit 15 - 0x1A	0 - 0xFFFF [bitfield] (bits 0, 14, 15 not allowed on the XBee-PRO)	0x1FFE (all XBee-PRO Channels)
SD (v1.x80*)	Networking {Association}	Scan Duration. Set/Read the scan duration exponent. *End Device* - Duration of Active Scan during Association. On beacon system, set SD = BE of coordinator. SD must be set at least to the highest BE parameter of any Beaconing Coordinator with which an End Device or Coordinator wish to discover. *Coordinator* - If ‘ReassignPANID’ option is set on Coordinator [refer to A2 parameter], SD determines the length of time the Coordinator will scan channels to locate existing PANs. If ‘ReassignChannel’ option is set, SD determines how long the Coordinator will perform an Energy Scan to determine which channel it will operate on. ‘Scan Time’ is measured as (# of channels to scan] * (2 ^ SD) * 15.36ms). The number of channels to scan is set by the SC command. The XBee can scan up to 16 channels (SC = 0xFFFF). The XBee PRO can scan up to 13 channels (SC = 0x3FFE). Example: The values below show results for a 13 channel scan: If SD = 0, time = 0.18 sec SD = 8, time = 47.19 sec SD = 2, time = 0.74 sec SD = 10, time = 3.15 min SD = 4, time = 2.95 sec SD = 12, time = 12.58 min SD = 6, time = 11.80 sec SD = 14, time = 50.33 min	0-0x0F [exponent]	4
A1 (v1.x80*)	Networking {Association}	End Device Association. Set/Read End Device association options. bit 0 - ReassignPanID 0 - Will only associate with Coordinator operating on PAN ID that matches module ID 1 - May associate with Coordinator operating on any PAN ID bit 1 - ReassignChannel 0 - Will only associate with Coordinator operating on matching CH Channel setting 1 - May associate with Coordinator operating on any Channel bit 2 - AutoAssociate 0 - Device will not attempt Association 1 - Device attempts Association until success Note: This bit is used only for Non-Beacon systems. End Devices in Beacon-enabled system must always associate to a Coordinator bit 3 - PollCoordOnPinWake 0 - Pin Wake will not poll the Coordinator for indirect (pending) data 1 - Pin Wake will send Poll Request to Coordinator to extract any pending data bits 4 - 7 are reserved	0 - 0x0F [bitfield]	0

Table 3-02. XBee/XBee-PRO Commands - Networking & Security (Sub-categories designated within {brackets})

AT Command	Command Category	Name and Description	Parameter Range	Default
A2 (v1.x80*)	Networking {Association}	Coordinator Association. Set/Read Coordinator association options. bit 0 - ReassignPanID 0 - Coordinator will not perform Active Scan to locate available PAN ID. It will operate on ID (PAN ID). 1 - Coordinator will perform Active Scan to determine an available ID (PAN ID). If a PAN ID conflict is found, the ID parameter will change. bit 1 - ReassignChannel - 0 - Coordinator will not perform Energy Scan to determine free channel. It will operate on the channel determined by the CH parameter. 1 - Coordinator will perform Energy Scan to find a free channel, then operate on that channel. bit 2 - AllowAssociation - 0 - Coordinator will not allow any devices to associate to it. 1 - Coordinator will allow devices to associate to it. bits 3 - 7 are reserved	0 - 7 [bitfield]	0
AI (v1.x80*)	Networking {Association}	Association Indication. Read errors with the last association request: 0x00 - Successful Completion - Coordinator successfully started or End Device association complete 0x01 - Active Scan Timeout 0x02 - Active Scan found no PANs 0x03 - Active Scan found PAN, but the CoordinatorAllowAssociation bit is not set 0x04 - Active Scan found PAN, but Coordinator and End Device are not configured to support beacons 0x05 - Active Scan found PAN, but the Coordinator ID parameter does not match the ID parameter of the End Device 0x06 - Active Scan found PAN, but the Coordinator CH parameter does not match the CH parameter of the End Device 0x07 - Energy Scan Timeout 0x08 - Coordinator start request failed 0x09 - Coordinator could not start due to invalid parameter 0x0A - Coordinator Realignment is in progress 0x0B - Association Request not sent 0x0C - Association Request timed out - no reply was received 0x0D - Association Request had an Invalid Parameter 0x0E - Association Request Channel Access Failure. Request was not transmitted - CCA failure 0x0F - Remote Coordinator did not send an ACK after Association Request was sent 0x10 - Remote Coordinator did not reply to the Association Request, but an ACK was received after sending the request 0x11 - [reserved] 0x12 - Sync-Loss - Lost synchronization with a Beaconing Coordinator 0x13 - Disassociated - No longer associated to Coordinator	0 - 0x13 [read-only]	-
DA (v1.x80*)	Networking {Association}	Force Disassociation. End Device will immediately disassociate from a Coordinator (if associated) and reattempt to associate.	-	-
FP (v1.x80*)	Networking {Association}	Force Poll. Request indirect messages being held by a coordinator.	-	-

Table 3-02. XBee/XBee-PRO Commands - Networking & Security (Sub-categories designated within {brackets})

AT Command	Command Category	Name and Description	Parameter Range	Default
AS (v1.x80*)	Networking {Association}	Active Scan. Send Beacon Request to Broadcast Address (0xFFFF) and Broadcast PAN (0xFFFF) on every channel. The parameter determines the time the radio will listen for Beacons on each channel. A PanDescriptor is created and returned for every Beacon received from the scan. Each PanDescriptor contains the following information: CoordAddress (SH, SL)<CR> CoordPanID (ID)<CR> CoordAddrMode <CR> 0x02 = 16-bit Short Address 0x03 = 64-bit Long Address Channel (CH parameter) <CR> SecurityUse<CR> ACLEntry<CR> SecurityFailure<CR> SuperFrameSpec<CR> (2 bytes): bit 15 - Association Permitted (MSB) bit 14 - PAN Coordinator bit 13 - Reserved bit 12 - Battery Life Extension bits 8-11 - Final CAP Slot bits 4-7 - Superframe Order bits 0-3 - Beacon Order GtsPermit<CR> RSSI<CR> (RSSI is returned as -dBm) TimeStamp<CR> (3 bytes) <CR> A carriage return <CR> is sent at the end of the AS command. The Active Scan is capable of returning up to 5 PanDescriptors in a scan. The actual scan time on each channel is measured as Time = [(2 ^SD PARAM) * 15.36] ms. Note the total scan time is this time multiplied by the number of channels to be scanned (16 for the XBee and 13 for the XBee-PRO). Also refer to SD command description.	0 - 6	-
ED (v1.x80*)	Networking {Association}	Energy Scan. Send an Energy Detect Scan. This parameter determines the length of scan on each channel. The maximal energy on each channel is returned & each value is followed by a carriage return. An additional carriage return is sent at the end of the command. The values returned represent the detected energy level in units of -dBm. The actual scan time on each channel is measured as Time = [(2 ^ED) * 15.36] ms. Note the total scan time is this time multiplied by the number of channels to be scanned (refer to SD parameter).	0 - 6	-
EE (v1.xA0*)	Networking {Security}	AES Encryption Enable. Disable/Enable 128-bit AES encryption support. Use in conjunction with the KY command.	0 - 1	0 (disabled)
KY (v1.xA0*)	Networking {Security}	AES Encryption Key. Set the 128-bit AES (Advanced Encryption Standard) key for encrypting/decrypting data. The KY register cannot be read.	0 - (any 16-Byte value)	-

* Firmware version in which the command was first introduced (firmware versions are numbered in hexadecimal notation.)

RF Interfacing

Table 3-03. XBee/XBee-PRO Commands - RF Interfacing

AT Command	Command Category	Name and Description	Parameter Range	Default
PL	RF Interfacing	Power Level. Select/Read the power level at which the RF module transmits conducted power. *NOTE: XBee-PRO RF Modules optimized for use in Japan contain firmware that limits transmit power output to 10 dBm. If PL=4 (default), the maximum power output level is fixed at 10 dBm.*	0 - 4 (XBee / XBee-PRO) 0 = -10 / 10 dBm 1 = -6 / 12 dBm 2 = -4 / 14 dBm 3 = -2 / 16 dBm 4 = 0 / 18 dBm	4
CA (v1.x80*)	RF Interfacing	CCA Threshold. Set/read the CCA (Clear Channel Assessment) threshold. Prior to transmitting a packet, a CCA is performed to detect energy on the channel. If the detected energy is above the CCA Threshold, the module will not transmit the packet.	0 - 0x50 [-dBm]	0x2C (-44d dBm)

* Firmware version in which the command was first introduced (firmware versions are numbered in hexadecimal notation.)

Sleep (Low Power)

Table 3-04. XBee/XBee-PRO Commands - Sleep (Low Power)

AT Command	Command Category	Name and Description	Parameter Range	Default
SM	Sleep (Low Power)	Sleep Mode. <NonBeacon firmware> Set/Read Sleep Mode configurations.	0 - 5 0 = No Sleep 1 = Pin Hibernate 2 = Pin Doze 3 = Reserved 4 = Cyclic sleep remote 5 = Cyclic sleep remote w/ pin wake-up 6 = [Sleep Coordinator] for backwards compatibility w/ v1.x6 only; otherwise, use CE command.	0
ST	Sleep (Low Power)	Time before Sleep. <NonBeacon firmware> Set/Read time period of inactivity (no serial or RF data is sent or received) before activating Sleep Mode. ST parameter is only valid with Cyclic Sleep settings (SM = 4 - 5). Coordinator and End Device ST values must be equal. Also note, the GT parameter value must always be less than the ST value. (If GT > ST, the configuration will render the module unable to enter into command mode.) If the ST parameter is modified, also modify the GT parameter accordingly.	1 - 0xFFFF [x 1 ms]	0x1388 (5000d)
SP	Sleep (Low Power)	Cyclic Sleep Period. <NonBeacon firmware> Set/Read sleep period for cyclic sleeping remotes. Coordinator and End Device SP values should always be equal. To send Direct Messages, set SP = 0. *End Device* - SP determines the sleep period for cyclic sleeping remotes. Maximum sleep period is 268 seconds (0x68B0). *Coordinator* - If non-zero, SP determines the time to hold an indirect message before discarding it. A Coordinator will discard indirect messages after a period of (2.5 * SP).	0 - 0x68B0 [x 10 ms]	0
DP (1.x80*)	Sleep (Low Power)	Disassociated Cyclic Sleep Period. <NonBeacon firmware> *End Device* - Set/Read time period of sleep for cyclic sleeping remotes that are configured for Association but are not associated to a Coordinator. (i.e. If a device is configured to associate, configured as a Cyclic Sleep remote, but does not find a Coordinator, it will sleep for DP time before reattempting association.) Maximum sleep period is 268 seconds (0x68B0). DP should be > 0 for NonBeacon systems.	1 - 0x68B0 [x 10 ms]	0x3E8 (1000d)

* Firmware version in which the command was first introduced (firmware versions are numbered in hexadecimal notation.)

Serial Interfacing

Table 3-05. XBee-PRO Commands - Serial Interfacing

AT Command	Command Category	Name and Description	Parameter Range	Default
BD	Serial Interfacing	Interface Data Rate. Set/Read the serial interface data rate for communications between the RF module serial port and host. Request non-standard baud rates with values above 0x80 using a terminal window. Read the BD register to find actual baud rate achieved.	0 - 7 (standard baud rates) 0 = 1200 bps 1 = 2400 2 = 4800 3 = 9600 4 = 19200 5 = 38400 6 = 57600 7 = 115200 0x80 - 0x1C200 (non-standard baud rates)	3
RO	Serial Interfacing	Packetization Timeout. Set/Read number of character times of inter-character delay required before transmission. Set to zero to transmit characters as they arrive instead of buffering them into one RF packet.	0 - 0xFF [x character times]	3
AP (v1.x80*)	Serial Interfacing	API Enable. Disable/Enable API Mode.	0 - 2 0 = Disabled 1 = API enabled 2 = API enabled (w/escaped control characters)	0
NB	Serial Interfacing	Parity. Set/Read parity settings.	0 - 4 0 = 8-bit (no parity or 7-bit (any parity) 1 = 8-bit even 2 = 8-bit odd 3 = 8-bit mark 4 = 8-bit space	0

Table 3-05. XBee-PRO Commands - Serial Interfacing

AT Command	Command Category	Name and Description	Parameter Range	Default
PR (v1.x80*)	Serial Interfacing	Pull-up Resistor Enable. Set/Read bitfield to configure internal pull-up resistor status for I/O lines Bitfield Map: bit 0 - AD4/DIO4 (pin11) bit 1 - AD3 / DIO3 (pin17) bit 2 - AD2/DIO2 (pin18) bit 3 - AD1/DIO1 (pin19) bit 4 - AD0 / DIO0 (pin20) bit 5 - RTS / AD6 / DIO6 (pin16) bit 6 - DTR / SLEEP_RQ / DI8 (pin9) bit 7 - DIN/CONFIG (pin3) Bit set to “1” specifies pull-up enabled; “0” specifies no pull-up	0 - 0xFF	0xFF

* Firmware version in which the command was first introduced (firmware versions are numbered in hexadecimal notation.)

I/O Settings

Table 3-06. XBee-PRO Commands - I/O Settings (sub-category designated within {brackets})

AT Command	Command Category	Name and Description	Parameter Range	Default
D8	I/O Settings	DI8 Configuration. Select/Read options for the DI8 line (pin 9) of the RF module.	0 - 1 0 = Disabled 3 = DI (1,2,4 & 5 n/a)	0
D7 (v1.x80*)	I/O Settings	DIO7 Configuration. Select/Read settings for the DIO7 line (pin 12) of the RF module. Options include CTS flow control and I/O line settings.	0 - 1 0 = Disabled 1 = CTS Flow Control 2 = (n/a) 3 = DI 4 = DO low 5 = DO high	1
D6 (v1.x80*)	I/O Settings	DIO6 Configuration. Select/Read settings for the DIO6 line (pin 16) of the RF module. Options include RTS flow control and I/O line settings.	0 - 1 0 = Disabled 1 = RTS flow control 2 = (n/a) 3 = DI 4 = DO low 5 = DO high	0
D5 (v1.x80*)	I/O Settings	DIO5 Configuration. Configure settings for the DIO5 line (pin 15) of the RF module. Options include Associated LED indicator (blinks when associated) and I/O line settings.	0 - 1 0 = Disabled 1 = Associated indicator 2 = ADC 3 = DI 4 = DO low 5 = DO high	1
D0 - D4 (v1.xA0*)	I/O Settings	(DIO4 -DIO4) Configuration. Select/Read settings for the following lines: AD0/DIO0 (pin 20), AD1/DIO1 (pin 19), AD2/DIO2 (pin 18), AD3/DIO3 (pin 17), AD4/DIO4 (pin 11). Options include: Analog-to-digital converter, Digital Input and Digital Output.	0 - 1 0 = Disabled 1 = (n/a) 2 = ADC 3 = DI 4 = DO low 5 = DO high	0
IU (v1.xA0*)	I/O Settings	I/O Output Enable. Disables/Enables I/O data received to be sent out UART. The data is sent using an API frame regardless of the current AP parameter value.	0 - 1 0 = Disabled 1 = Enabled	1
IT (v1.xA0*)	I/O Settings	Samples before TX. Set/Read the number of samples to collect before transmitting data. Maximum number of samples is dependent upon the number of enabled inputs.	1 - 0xFF	1
IS (v1.xA0*)	I/O Settings	Force Sample. Force a read of all enabled inputs (DI or ADC). Data is returned through the UART. If no inputs are defined (DI or ADC), this command will return error.	8-bit bitmap (each bit represents the level of an I/O line setup as an output)	-
IO (v1.xA0*)	I/O Settings	Digital Output Level. Set digital output level to allow DIO lines that are setup as outputs to be changed through Command Mode.	-	-
IC (v1.xA0*)	I/O Settings	DIO Change Detect. Set/Read bitfield values for change detect monitoring. Each bit enables monitoring of DIO0 - DIO7 for changes. If detected, data is transmitted with DIO data only. Any samples queued waiting for transmission will be sent first.	0 - 0xFF [bitfield]	0 (disabled)
IR (v1.xA0*)	I/O Settings	Sample Rate. Set/Read sample rate. When set, this parameter causes the module to sample all enabled inputs at a specified interval.	0 - 0xFFFF [x 1 msec]	0
AV (v1.xA0*)	I/O Settings	ADC Voltage Reference. <XBee-PRO only> Set/Read ADC reference voltage switch.	0 - 1 0 = VREF pin 1 = Internal	0

Table 3-06. **XBee-PRO Commands - I/O Settings** (sub-category designated within {brackets})

AT Command	Command Category	Name and Description	Parameter Range	Default
IA (v1.xA0*)	I/O Settings {I/O Line Passing}	I/O Input Address. Set/Read addresses of module to which outputs are bound. Setting all bytes to 0xFF will not allow any received I/O packet to change outputs. Setting address to 0xFFFF will allow any received I/O packet to change outputs.	0 - 0xFFFFFFFFFFFFFFFF	0xFFFFFFFF FFFFFFFF
T0 - T7 (v1.xA0*)	I/O Settings {I/O Line Passing}	(D0 - D7) Output Timeout. Set/Read Output timeout values for lines that correspond with the D0 - D7 parameters. When output is set (due to I/O line passing) to a non-default level, a timer is started which when expired will set the output to it default level. The timer is reset when a valid I/O packet is received.	0 - 0xFF [x 100 ms]	0xFF
P0	I/O Settings {I/O Line Passing}	PWM0 Configuration. Select/Read function for PWM0 pin.	0 - 2 0 = Disabled 1 = RSSI 2 = PWM Output	1
P1 (v1.xA0*)	I/O Settings {I/O Line Passing}	PWM1 Configuration. Select/Read function for PWM1 pin.	0 - 2 0 = Disabled 1 = RSSI 2 = PWM Output	0
M0 (v1.xA0*)	I/O Settings {I/O Line Passing}	PWM0 Output Level. Set/Read the PWM0 output level.	0 - 0x03FF	-
M1 (v1.xA0*)	I/O Settings {I/O Line Passing}	PWM1 Output Level. Set/Read the PWM0 output level.	0 - 0x03FF	-
PT (v1.xA0*)	I/O Settings {I/O Line Passing}	PWM Output Timeout. Set/Read output timeout value for both PWM outputs. When PWM is set to a non-zero value: Due to I/O line passing, a time is started which when expired will set the PWM output to zero. The timer is reset when a valid I/O packet is received.]	0 - 0xFF [x 100 ms]	0xFF
RP	I/O Settings {I/O Line Passing}	RSSI PWM Timer. Set/Read PWM timer register. Set the duration of PWM (pulse width modulation) signal output on the RSSI pin. The signal duty cycle is updated with each received packet and is shut off when the timer expires.]	0 - 0xFF [x 100 ms]	0x28 (40d)

* Firmware version in which the command was first introduced (firmware versions are numbered in hexadecimal notation.)

Diagnostics

Table 3-07. **XBee/XBee-PRO Commands - Diagnostics**

AT Command	Command Category	Name and Description	Parameter Range	Default
VR	Diagnostics	Firmware Version. Read firmware version of the RF module.	0 - 0xFFFF [read-only]	Factory-set
VL (v1.x80*)	Diagnostics	Firmware Version - Verbose. Read detailed version information (including application build date, MAC, PHY and bootloader versions).	-	-
HV (v1.x80*)	Diagnostics	Hardware Version. Read hardware version of the RF module.	0 - 0xFFFF [read-only]	Factory-set
DB	Diagnostics	Received Signal Strength. Read signal level [in dB] of last good packet received (RSSI). Absolute value is reported. (For example: 0x58 = -88 dBm) Reported value is accurate between -40 dBm and RX sensitivity.	0x17-0x5C (XBee) 0x24-0x64 (XBee-PRO) [read-only]	-
EC (v1.x80*)	Diagnostics	CCA Failures. Reset/Read count of CCA (Clear Channel Assessment) failures. This parameter value increments when the module does not transmit a packet because it detected energy above the CCA threshold level set with CA command. This count saturates at its maximum value. Set count to "0" to reset count.	0 - 0xFFFF	-
EA (v1.x80*)	Diagnostics	ACK Failures. Reset/Read count of acknowledgment failures. This parameter value increments when the module expires its transmission retries without receiving an ACK on a packet transmission. This count saturates at its maximum value. Set the parameter to "0" to reset count.	0 - 0xFFFF	-
ED (v1.x80*)	Diagnostics	Energy Scan. Send 'Energy Detect Scan'. ED parameter determines the length of scan on each channel. The maximal energy on each channel is returned and each value is followed by a carriage return. Values returned represent detected energy levels in units of -dBm. Actual scan time on each channel is measured as Time = [(2 ^ SD) * 15.36] ms. Total scan time is this time multiplied by the number of channels to be scanned.	0 - 6	-

* Firmware version in which the command was first introduced (firmware versions are numbered in hexadecimal notation.)

AT Command Options

Table 3-08. XBee/XBee-PRO Commands - AT Command Options

AT Command	Command Category	Name and Description	Parameter Range	Default
CT	AT Command Mode Options	Command Mode Timeout. Set/Read the period of inactivity (no valid commands received) after which the RF module automatically exits AT Command Mode and returns to Idle Mode.	2 - 0xFFFF [x 100 ms]	0x64 (100d)
CN	AT Command Mode Options	Exit Command Mode. Explicitly exit the module from AT Command Mode.	--	--
AC (v1.xA0*)	AT Command Mode Options	Apply Changes. Explicitly apply changes to queued parameter value(s) and re-initialize module.	--	--
GT	AT Command Mode Options	Guard Times. Set required period of silence before and after the Command Sequence Characters of the AT Command Mode Sequence (GT+ CC + GT). The period of silence is used to prevent inadvertent entrance into AT Command Mode.	2 - 0x0CE4 [x 1 ms]	0x3E8 (1000d)
CC	AT Command Mode Options	Command Sequence Character. Set/Read the ASCII character value to be used between Guard Times of the AT Command Mode Sequence (GT+CC+GT). The AT Command Mode Sequence enters the RF module into AT Command Mode.	0 - 0xFF	0x2B ('+' ASCII)

* Firmware version in which the command was first introduced (firmware versions are numbered in hexadecimal notation.)

3.3. Command Descriptions

Command descriptions in this section are listed alphabetically. Command categories are designated within "< >" symbols that follow each command title. XBee/XBee-PRO RF Modules expect parameter values in hexadecimal (designated by the "0x" prefix).

All modules operating within the same network should contain the same firmware version.

A1 (End Device Association) Command

<Networking {Association}> The A1 command is used to set and read association options for an End Device.

Use the table below to determine End Device behavior in relation to the A1 parameter.

AT Command: ATA1

Parameter Range: 0 – 0x0F [bitfield]

Default Parameter Value: 0

Related Commands: ID (PAN ID), NI (Node Identifier), CH (Channel), CE (Coordinator Enable), A2 (Coordinator Association)

Minimum Firmware Version Required: v1.x80

Bit number	End Device Association Option
0 - ReassignPanID	0 - Will only associate with Coordinator operating on PAN ID that matches Node Identifier
	1 - May associate with Coordinator operating on any PAN ID
1 - ReassignChannel	0 - Will only associate with Coordinator operating on Channel that matches CH setting
	1 - May associate with Coordinator operating on any Channel
2 - AutoAssociate	0 - Device will not attempt Association
	1 - Device attempts Association until success Note: This bit is used only for Non-Beacon systems. End Devices in a Beaconing system must always associate to a Coordinator
3 - PollCoordOnPinWake	0 - Pin Wake will not poll the Coordinator for pending (indirect) Data
	1 - Pin Wake will send Poll Request to Coordinator to extract any pending data
4 - 7	[reserved]

A2 (Coordinator Association) Command

<Networking {Association}> The A2 command is used to set and read association options of the Coordinator.

Use the table below to determine Coordinator behavior in relation to the A2 parameter.

AT Command: ATA2

Parameter Range: 0 – 7 [bitfield]

Default Parameter Value: 0

Related Commands: ID (PAN ID), NI (Node Identifier), CH (Channel), CE (Coordinator Enable), A1 (End Device Association), AS Active Scan), ED (Energy Scan)

Minimum Firmware Version Required: v1.x80

Bit number	End Device Association Option
0 - ReassignPanID	0 - Coordinator will not perform Active Scan to locate available PAN ID. It will operate on ID (PAN ID).
	1 - Coordinator will perform Active Scan to determine an available ID (PAN ID). If a PAN ID conflict is found, the ID parameter will change.
1 - ReassignChannel	0 - Coordinator will not perform Energy Scan to determine free channel. It will operate on the channel determined by the CH parameter.
	1 - Coordinator will perform Energy Scan to find a free channel, then operate on that channel.
2 - AllowAssociate	0 - Coordinator will not allow any devices to associate to it.
	1 - Coordinator will allow devices to associate to it.
3 - 7	[reserved]

The binary equivalent of the default value (0x06) is 00000110. 'Bit 0' is the last digit of the sequence.

AC (Apply Changes) Command

AT Command: ATAC

Minimum Firmware Version Required: v1.xA0

<AT Command Mode Options> The AC command is used to explicitly apply changes to module parameter values. 'Applying changes' means that the module is re-initialized based on changes made to its parameter values. Once changes are applied, the module immediately operates according to the new parameter values.

This behavior is in contrast to issuing the WR (Write) command. The WR command saves parameter values to non-volatile memory, but the module still operates according to previously saved values until the module is re-booted or the CN (Exit AT Command Mode) command is issued.

Refer to the "AT Command - Queue Parameter Value" API type for more information.

AI (Association Indication) Command

AT Command: ATAI

Parameter Range: 0 - 0x13 [read-only]

Related Commands: AS (Active Scan), ID (PAN ID), CH (Channel), ED (Energy Scan), A1 (End Device Association), A2 (Coordinator Association), CE (Coordinator Enable)

Minimum Firmware Version Required: v1.x80

<Networking {Association}> The AI command is used to indicate occurrences of errors during the last association request.

Use the table below to determine meaning of the returned values.

Returned Value (Hex)	Association Indication
0x00	Successful Completion - Coordinator successfully started or End Device association complete
0x01	Active Scan Timeout
0x02	Active Scan found no PANs
0x03	Active Scan found PAN, but the Coordinator Allow Association bit is not set
0x04	Active Scan found PAN, but Coordinator and End Device are not configured to support beacons
0x05	Active Scan found PAN, but Coordinator ID (PAN ID) value does not match the ID of the End Device
0x06	Active Scan found PAN, but Coordinator CH (Channel) value does not match the CH of the End Device
0x07	Energy Scan Timeout
0x08	Coordinator start request failed
0x09	Coordinator could not start due to Invalid Parameter
0x0A	Coordinator Realignment is in progress
0x0B	Association Request not sent
0x0C	Association Request timed out - no reply was received
0x0D	Association Request had an Invalid Parameter
0x0E	Association Request Channel Access Failure - Request was not transmitted - CCA failure
0x0F	Remote Coordinator did not send an ACK after Association Request was sent
0x10	Remote Coordinator did not reply to the Association Request, but an ACK was received after sending the request
0x11	[reserved]
0x12	Sync-Loss - Lost synchronization with a Beaconing Coordinator
0x13	Disassociated - No longer associated to Coordinator
0xFF	RF Module is attempting to associate

AP (API Enable) Command

<Serial Interfacing> The AP command is used to enable the RF module to operate using a frame-based API instead of using the default Transparent (UART) mode.

AT Command: ATAP	
Parameter Range:0 – 2	
Parameter	Configuration
0	Disabled (Transparent operation)
1	API enabled
2	API enabled (with escaped characters)
Default Parameter Value:0	
Minimum Firmware Version Required: v1.x80	

Refer to the API Operation section when API operation is enabled (AP = 1 or 2).

AS (Active Scan) Command

<AT Command Mode Options> The AS command is used to send a Beacon Request to a Broadcast (0xFFFF) and Broadcast PAN (0xFFFF) on every channel. The parameter determines the amount of time the RF module will listen for Beacons on each channel. A ‘PanDescriptor’ is created and returned for every Beacon received from the scan. Each PanDescriptor contains the following information:

AT Command: ATAS
Parameter Range: 0 – 6
Related Command: SD (Scan Duration), DL (Destination Low Address), DH (Destination High Address), ID (PAN ID), CH (Channel)
Minimum Firmware Version Required: v1.x80

CoordAddress (SH + SL parameters)<CR> (NOTE: If MY on the coordinator is set less than 0xFFFF, the MY value is displayed)
CoordPanID (ID parameter)<CR>
CoordAddrMode <CR>
  0x02 = 16-bit Short Address
  0x03 = 64-bit Long Address
Channel (CH parameter) <CR>
SecurityUse<CR>
ACLEntry<CR>
SecurityFailure<CR>
SuperFrameSpec<CR> (2 bytes):
  bit 15 - Association Permitted (MSB)
  bit 14 - PAN Coordinator
  bit 13 - Reserved
  bit 12 - Battery Life Extension
  bits 8-11 - Final CAP Slot
  bits 4-7 - Superframe Order
  bits 0-3 - Beacon Order
GtsPermit<CR>
RSSI<CR> (- RSSI is returned as -dBm)
TimeStamp<CR> (3 bytes)
<CR> (A carriage return <CR> is sent at the end of the AS command.

The Active Scan is capable of returning up to 5 PanDescriptors in a scan. The actual scan time on each channel is measured as Time = [(2 ^ (SD Parameter)) * 15.36] ms. Total scan time is this time multiplied by the number of channels to be scanned (16 for the XBee, 12 for the XBee-PRO).

NOTE: Refer the scan table in the SD description to determine scan times. If using API Mode, no <CR>’s are returned in the response. Refer to the API Mode Operation section.

AV (ADC Voltage Reference) Command

<Serial Interfacing> The AV command is used to set/read the ADC reference voltage switch. The XBee-PRO has an ADC voltage reference switch which allows the module to select between an on-board voltage reference or to use the VREF pin on the connector.

This command only applies to XBee-PRO RF Modules and will return error on an XBee RF Module.

AT Command: ATAV	
Parameter Range:0 – 1	
Parameter	Configuration
0	VREF Pin
1	Internal (on-board reference – VCC)
Default Parameter Value:0	
Minimum Firmware Version Required: v1.xA0	

BD (Interface Data Rate) Command

<Serial Interfacing> The BD command is used to set and read the serial interface data rate used between the RF module and host. This parameter determines the rate at which serial data is sent to the module from the host. Modified interface data rates do not take effect until the CN (Exit AT Command Mode) command is issued and the system returns the 'OK' response.

When parameters 0-7 are sent to the module, the respective interface data rates are used (as shown in the table on the right).

The RF data rate is not affected by the BD parameter. If the interface data rate is set higher than the RF data rate, a flow control configuration may need to be implemented.

AT Command: ATBD	
Parameter Range:0 – 7 (standard rates) 0x80-0x1C200 (non-stndard rates)	
Parameter	Configuration (bps)
0	1200
1	2400
2	4800
3	9600
4	19200
5	38400
6	57600
7	115200
Default Parameter Value:3	

Non-standard Interface Data Rates:
Any value above 0x07 will be interpreted as an actual baud rate. When a value above 0x07 is sent, the closest interface data rate represented by the number is stored in the BD register. For example, a rate of 19200 bps can be set by sending the following command line "ATBD4B00". NOTE: When using MaxStream's X-CTU Software, non-standard interface data rates can only be set and read using the X-CTU 'Terminal' tab. Non-standard rates are not accessible through the 'Modem Configuration' tab.

When the BD command is sent with a non-standard interface data rate, the UART will adjust to accommodate the requested interface rate. In most cases, the clock resolution will cause the stored BD parameter to vary from the parameter that was sent (refer to the table below). Reading the BD command (send "ATBD" command without an associated parameter value) will return the value actually stored in the module's BD register.

Parameters Sent Versus Parameters Stored

BD Parameter Sent (HEX)	Interface Data Rate (bps)	BD Parameter Stored (HEX)
0	1200	0
4	19,200	4
7	115,200	7
12C	300	12B
1C200	115,200	1B207

CA (CCA Threshold) Command

<RF Interfacing> CA command is used to set and read CCA (Clear Channel Assessment) thresholds.

Prior to transmitting a packet, a CCA is performed to detect energy on the transmit channel. If the detected energy is above the CCA Threshold, the RF module will not transmit the packet.

AT Command: ATCA
Parameter Range: 0 – 0x50 [-dBm]
Default Parameter Value: 0x2C (-44 decimal dBm)
Minimum Firmware Version Required: v1.x80

CC (Command Sequence Character) Command

<AT Command Mode Options> The CC command is used to set and read the ASCII character used between guard times of the AT Command Mode Sequence (GT + CC + GT). This sequence enters the RF module into AT Command Mode so that data entering the module from the host is recognized as commands instead of payload.

AT Command: ATCC
Parameter Range: 0 – 0xFF
Default Parameter Value: 0x2B (ASCII “+”)
Related Command: GT (Guard Times)

The AT Command Sequence is explained further in the AT Command Mode section.

CE (Coordinator Enable) Command

<Serial Interfacing> The CE command is used to set and read the behavior (End Device vs. Coordinator) of the RF module.

AT Command: ATCE

Parameter Range:0 – 1

Parameter	Configuration
0	End Device
1	Coordinator

Default Parameter Value:0

Minimum Firmware Version Required: v1.x80

CH (Channel) Command

<Networking {Addressing}> The CH command is used to set/read the operating channel on which RF connections are made between RF modules. The channel is one of three addressing options available to the module. The other options are the PAN ID (ID command) and destination addresses (DL & DH commands).

AT Command: ATCH
Parameter Range: 0x0B – 0x1A (XBee) 0x0C – 0x17 (XBee-PRO)
Default Parameter Value: 0x0C (12 decimal)
Related Commands: ID (PAN ID), DL (Destination Address Low, DH (Destination Address High)

In order for modules to communicate with each other, the modules must share the same channel number. Different channels can be used to prevent modules in one network from listening to transmissions of another. Adjacent channel rejection is 23 dB.

The module uses channel numbers of the 802.15.4 standard.
Center Frequency = 2.405 + (CH - 11d) * 5 MHz (d = decimal)

Refer to the XBee/XBee-PRO Addressing section for more information.

CN (Exit Command Mode) Command

<AT Command Mode Options> The CN command is used to explicitly exit the RF module from AT Command Mode.

AT Command: ATCN

CT (Command Mode Timeout) Command

<AT Command Mode Options> The CT command is used to set and read the amount of inactive time that elapses before the RF module automatically exits from AT Command Mode and returns to Idle Mode.

Use the CN (Exit Command Mode) command to exit AT Command Mode manually.

AT Command: ATCT
Parameter Range:2 – 0xFFFF [x 100 milliseconds]
Default Parameter Value: 0x64 (100 decimal (which equals 10 decimal seconds))
Number of bytes returned: 2
Related Command: CN (Exit Command Mode)

D0 - D4 (DIOn Configuration) Commands

<I/O Settings> The D0, D1, D2, D3 and D4 commands are used to select/read the behavior of their respective AD/DIO lines (pins 20, 19, 18, 17 and 11 respectively).

Options include:

- Analog-to-digital converter
- Digital input
- Digital output

AT Commands:
ATD0, ATD1, ATD2, ATD3, ATD4

Parameter Range:0 - 5

Parameter	Configuration
0	Disabled
1	n/a
2	ADC
3	DI
4	DO low
5	DO high

Default Parameter Value:0

Minimum Firmware Version Required: 1.x.A0

D5 (DIO5 Configuration) Command

<I/O Settings> The D5 command is used to select/read the behavior of the DIO5 line (pin 15).

Options include:

- Associated Indicator (LED blinks when the module is associated)
- Analog-to-digital converter
- Digital input
- Digital output

AT Command: ATD5

Parameter Range:0 - 5

Parameter	Configuration
0	Disabled
1	Associated Indicator
2	ADC
3	DI
4	DO low
5	DO high

Default Parameter Value:1

Parameters 2-5 supported as of firmware version 1.xA0

D6 (DIO6 Configuration) Command

<I/O Settings> The D6 command is used to select/read the behavior of the DIO6 line (pin 16). Options include:

- RTS flow control
- Analog-to-digital converter
- Digital input
- Digital output

AT Command: ATD6

Parameter Range:0 - 5

Parameter	Configuration
0	Disabled
1	RTS Flow Control
2	n/a
3	DI
4	DO low
5	DO high

Default Parameter Value:0

Parameters 3-5 supported as of firmware version 1.xA0

D7 (DIO7 Configuration) Command

<I/O Settings> The D7 command is used to select/read the behavior of the DIO7 line (pin 12). Options include:

- CTS flow control
- Analog-to-digital converter
- Digital input
- Digital output

AT Command: ATD7

Parameter Range:0 – 5

Parameter	Configuration
0	Disabled
1	CTS Flow Control
2	n/a
3	DI
4	DO low
5	DO high

Default Parameter Value:1

Parameters 3-5 supported as of firmware version 1.x.A0

D8 (DI8 Configuration) Command

<I/O Settings> The D8 command is used to select/read the behavior of the DI8 line (pin 9). This command enables configuring the pin to function as a digital input. This line is also used with Pin Sleep.

AT Command: ATD8

Parameter Range:0 – 5
(1, 2, 4 & 5 n/a)

Parameter	Configuration
0	Disabled
3	DI

Default Parameter Value:0

Minimum Firmware Version Required: 1.xA0

DA (Force Disassociation) Command

<(Special)> The DA command is used to immediately disassociate an End Device from a Coordinator and reattempt to associate.

AT Command: ATDA

Minimum Firmware Version Required: v1.x80

DB (Received Signal Strength) Command

<Diagnostics> DB parameter is used to read the received signal strength (in dBm) of the last RF packet received. Reported values are accurate between -40 dBm and the RF module's receiver sensitivity.

AT Command: ATDB

Parameter Range [read-only]:
0x17-0x5C (XBee), 0x24-0x64 (XBee-PRO)

Absolute values are reported. For example: 0x58 = -88 dBm (decimal). If no packets have been received (since last reset, power cycle or sleep event), “0” will be reported.

DH (Destination Address High) Command

<Networking {Addressing}> The DH command is used to set and read the upper 32 bits of the RF module's 64-bit destination address. When combined with the DL (Destination Address Low) parameter, it defines the destination address used for transmission.

AT Command: ATDH

Parameter Range: 0 – 0xFFFFFFFF

Default Parameter Value: 0

Related Commands: DL (Destination Address Low), CH (Channel), ID (PAN VID), MY (Source Address)

An module will only communicate with other modules having the same channel (CH parameter), PAN ID (ID parameter) and destination address (DH + DL parameters).

To transmit using a 16-bit address, set the DH parameter to zero and the DL parameter less than 0xFFFF. 0x000000000000FFFF (DL concatenated to DH) is the broadcast address for the PAN.

Refer to the XBee/XBee-PRO Addressing section for more information.

DL (Destination Address Low) Command

<Networking {Addressing}> The DL command is used to set and read the lower 32 bits of the RF module's 64-bit destination address. When combined with the DH (Destination Address High) parameter, it defines the destination address used for transmission.

A module will only communicate with other modules having the same channel (CH parameter), PAN ID (ID parameter) and destination address (DH + DL parameters).

AT Command: ATDL
Parameter Range: 0 - 0xFFFFFFFF
Default Parameter Value: 0
Related Commands: DH (Destination Address High), CH (Channel), ID (PAN VID), MY (Source Address)

To transmit using a 16-bit address, set the DH parameter to zero and the DL parameter less than 0xFFFF. 0x000000000000FFFF (DL concatenated to DH) is the broadcast address for the PAN.

Refer to the XBee/XBee-PRO Addressing section for more information.

DN (Destination Node) Command

<Networking {Identification}> The DN command is used to resolve a NI (Node Identifier) string to a physical address. The following events occur upon successful command execution:

AT Command: ATDN
Parameter Range: 20-character ASCII String
Minimum Firmware Version Required: v1.x80

1. DL and DH are set to the address of the module with the matching NI (Node Identifier).
2. 'OK' is returned.
3. RF module automatically exits AT Command Mode.

If there is no response from a modem within 200 msec or a parameter is not specified (left blank), the command is terminated and an 'ERROR' message is returned.

DP (Disassociation Cyclic Sleep Period) Command

<Sleep Mode (Low Power)>

NonBeacon Firmware

End Device - The DP command is used to set and read the time period of sleep for cyclic sleeping remotes that are configured for Association but are not associated to a Coordinator. (i.e. If a device is configured to associate, configured as a Cyclic Sleep remote, but does not find a Coordinator; it will sleep for DP time before reattempting association.) Maximum sleep period is 268 seconds (0x68B0). DP should be > 0 for NonBeacon systems.

AT Command: ATDP
Parameter Range: 1 - 0x68B0 [x 10 milliseconds]
Default Parameter Value:0x3E8 (1000 decimal)
Related Commands: SM (Sleep Mode), SP (Cyclic Sleep Period), ST (Time before Sleep)
Minimum Firmware Version Required: v1.x80

EA (ACK Failures) Command

<Diagnostics> The EA command is used to reset and read the count of ACK (acknowledgement) failures. This parameter value increments when the module expires its transmission retries without receiving an ACK on a packet transmission. This count saturates at its maximum value.

Set the parameter to "0" to reset count.

AT Command: ATEA
Parameter Range:0 - 0xFFFF
Minimum Firmware Version Required: v1.x80

EC (CCA Failures) Command

<Diagnostics> The EC command is used to read and reset the count of CCA (Clear Channel Assessment) failures. This parameter value increments when the RF module does not transmit a packet due to the detection of energy that is above the CCA threshold level (set with CA command). This count saturates at its maximum value.

Set the EC parameter to "0" to reset count.

AT Command: ATEC
Parameter Range:0 – 0xFFFF
Related Command: CA (CCA Threshold)
Minimum Firmware Version Required: v1.x80

ED (Energy Scan) Command

<Networking {Association}> The ED command is used to send an "Energy Detect Scan". This parameter determines the length of scan on each channel. The maximal energy on each channel is returned and each value is followed by a carriage return. An additional carriage return is sent at the end of the command.

AT Command: ATED
Parameter Range:0 – 6
Related Command: SD (Scan Duration), SC (Scan Channel)
Minimum Firmware Version Required: v1.x80

The values returned represent the detected energy level in units of -dBm. The actual scan time on each channel is measured as Time = [(2 ^ ED PARAM) * 15.36] ms.

Note: Total scan time is this time multiplied by the number of channels to be scanned. Also refer to the SD (Scan Duration) table. Use the SC (Scan Channel) command to choose which channels to scan.

EE (AES Encryption Enable) Command

<Networking {Security}> The EE command is used to set/read the parameter that disables/enables 128-bit AES encryption.

The XBee/XBee-PRO firmware uses the 802.15.4 Default Security protocol and uses AES encryption with a 128-bit key. AES encryption dictates that all modules in the network use the same key and the maximum RF packet size is 95 Bytes.

When encryption is enabled, the module will always use its 64-bit long address as the source address for RF packets. This does not affect how the MY (Source Address), DH (Destination Address High) and DL (Destination Address Low) parameters work

AT Command: ATEE	
Parameter Range:0 – 1	
Parameter	Configuration
0	Disabled
1	Enabled
Default Parameter Value:0	
Related Commands: KY (Encryption Key), AP (API Enable), MM (MAC Mode)	
Minimum Firmware Version Required: v1.xA0	

If MM (MAC Mode) > 0 and AP (API Enable) parameter > 0:
With encryption enabled and a 16-bit short address set, receiving modules will only be able to issue RX (Receive) 64-bit indicators. This is not an issue when MM = 0.

If a module with a non-matching key detects RF data, but has an incorrect key: When encryption is enabled, non-encrypted RF packets received will be rejected and will not be sent out the UART.

Transparent Operation --> All RF packets are sent encrypted if the key is set.

API Operation --> Receive frames use an option bit to indicate that the packet was encrypted.

FP (Force Poll) Command

<Networking (Association)> The FP command is used to request indirect messages being held by a Coordinator.

AT Command: ATFP
Minimum Firmware Version Required: v1.x80

FR (Software Reset) Command

<Special> The FR command is used to force a software reset on the RF module. The reset simulates powering off and then on again the module.

AT Command: ATFR
Minimum Firmware Version Required: v1.x80

GT (Guard Times) Command

<AT Command Mode Options> GT Command is used to set the DI (data in from host) time-of-silence that surrounds the AT command sequence character (CC Command) of the AT Command Mode sequence (GT + CC + GT).

The DI time-of-silence is used to prevent inadvertent entrance into AT Command Mode.

Refer to the Command Mode section for more information regarding the AT Command Mode Sequence.

AT Command: ATGT
Parameter Range:2 – 0x0CE4 [x 1 millisecond]
Default Parameter Value:0x3E8 (1000 decimal)
Related Command: CC (Command Sequence Character)

HV (Hardware Version) Command

<Diagnostics> The HV command is used to read the hardware version of the RF module.

AT Command: ATHV
Parameter Range:0 – 0xFFFF [Read-only]
Minimum Firmware Version Required: v1.x80

IA (I/O Input Address) Command

<I/O Settings {I/O Line Passing}> The IA command is used to bind a module output to a specific address. Outputs will only change if received from this address. The IA command can be used to set/read both 16 and 64-bit addresses.

Setting all bytes to 0xFF will not allow the reception of any I/O packet to change outputs. Setting the IA address to 0xFFFF will cause the module to accept all I/O packets.

AT Command: ATIA
Parameter Range:0 – 0xFFFFFFFFFFFFFFFF
Default Parameter Value:0xFFFFFFFFFFFFFFFF (will not allow any received I/O packet to change outputs)
Minimum Firmware Version Required: v1.xA0

IC (DIO Change Detect) Command

<I/O Settings> Set/Read bitfield values for change detect monitoring. Each bit enables monitoring of DIO0 - DIO7 for changes.

If detected, data is transmitted with DIO data only. Any samples queued waiting for transmission will be sent first.

AT Command: ATIC
Parameter Range:0 – 0xFF [bitfield]
Default Parameter Value:0 (disabled)
Minimum Firmware Version Required: 1.xA0

Refer to the “ADC and Digital I/O Line Support” sections of the “RF Module Operations” chapter for more information.

ID (Pan ID) Command

<Networking {Addressing}> The ID command is used to set and read the PAN (Personal Area Network) ID of the RF module. Only modules with matching PAN IDs can communicate with each other. Unique PAN IDs enable control of which RF packets are received by a module.

AT Command: ATID
Parameter Range: 0 – 0xFFFF
Default Parameter Value:0x3332 (13106 decimal)

Setting the ID parameter to 0xFFFF indicates a global transmission for all PANs. It does not indicate a global receive.

IO (Digital Output Level) Command

<I/O Settings> The IO command is used to set digital output levels. This allows DIO lines setup as outputs to be changed through Command Mode.

AT Command: ATIO
Parameter Range: 8-bit bitmap (where each bit represents the level of an I/O line that is setup as an output.)
Minimum Firmware Version Required: v1.xA0

IR (Sample Rate) Command

<I/O Settings> The IR command is used to set/read the sample rate. When set, the module will sample all enabled DIO/ADC lines at a specified interval. This command allows periodic reads of the ADC and DIO lines in a non-Sleep Mode setup.

Example: When IR = 0x0A, the sample rate is 10 ms (or 100 Hz).

AT Command: ATIR
Parameter Range: 0 - 0xFFFF [x 1 msec] (cannot guarantee 1 ms timing when IT=1)
Default Parameter Value:0
Related Command: IT (Samples before TX)
Minimum Firmware Version Required: v1.xA0

IS (Force Sample) Command

<I/O Settings> The IS command is used to force a read of all enabled DIO/ADC lines. The data is returned through the UART.

When operating in Transparent Mode (AP=0), the data is retuned in the following format:

AT Command: ATIS
Parameter Range: 1 - 0xFF
Default Parameter Value:1
Minimum Firmware Version Required: v1.xA0

```
All bytes are converted to ASCII:
    number of samples<CR>
    channel mask<CR>
    DIO data<CR> (If DIO lines are enabled<CR>
    ADC channel Data<cr> <-This will repeat for every enabled ADC channel<CR>
    <CR>  (end of data noted by extra <CR>)
```

When operating in API mode (AP > 0), the command will immediately return an 'OK' response. The data will follow in the normal API format for DIO data.

IT (Samples before TX) Command

<I/O Settings> The IT command is used to set/read the number of DIO and ADC samples to collect before transmitting data.

One ADC sample is considered complete when all enabled ADC channels have been read. The module can buffer up to 93 Bytes of sample data. Since the module uses a 10-bit A/D converter, each sample uses two Bytes. This leads to a maximum buffer size of 46 samples or IT=0x2E.

AT Command: ATIT
Parameter Range: 1 - 0xFF
Default Parameter Value:1
Minimum Firmware Version Required: v1.xA0

When Sleep Modes are enabled and IR (Sample Rate) is set, the module will remain awake until IT samples have been collected.

IU (I/O Output Enable) Command

<I/O Settings> The IU command is used to disable/enable I/O UART output. When enabled (IU = 1), received I/O line data packets are sent out the UART. The data is sent using an API frame regardless of the current AP parameter value.

AT Command: ATIU	
Parameter Range:0 – 1	
Parameter	Configuration
0	Disabled – Received I/O line data packets will NOT sent out UART.
1	Enabled – Received I/O line data will be sent out UART
Default Parameter Value:1	
Minimum Firmware Version Required: 1.xA0	

KY (AES Encryption Key) Command

AT Command: ATKY
Parameter Range:0 – (any 16-Byte value)
Default Parameter Value:0
Related Command: EE (Encryption Enable)
Minimum Firmware Version Required: v1.xA0

<Networking {Security}> The KY command is used to set the 128-bit AES (Advanced Encryption Standard) key for encrypting/decrypting data. Once set, the key cannot be read out of the module by any means.

The entire payload of the packet is encrypted using the key and the CRC is computed across the ciphertext. When encryption is enabled, each packet carries an additional 16 Bytes to convey the random CBC Initialization Vector (IV) to the receiver(s). The KY value may be “0” or any 128-bit value. Any other value, including entering KY by itself with no parameters, is invalid. All ATKY entries (valid or not) are received with a returned 'OK'.

A module with the wrong key (or no key) will receive encrypted data, but the data driven out the serial port will be meaningless. A module with a key and encryption enabled will receive data sent from a module without a key and the correct unencrypted data output will be sent out the serial port. Because CBC mode is utilized, repetitive data appears differently in different transmissions due to the randomly-generated IV.

When queried, the system will return an ‘OK’ message and the value of the key will not be returned.

M0 (PWM0 Output Level) Command

<I/O Settings> The M0 command is used to set/read the output level of the PWM0 line (pin 6).

Before setting the line as an output:

1. Enable PWM0 output (P0 = 2)
2. Apply settings (use CN or AC)

The PWM period is 64 µsec and there are 0x03FF (1023 decimal) steps within this period. When M0 = 0 (0% PWM), 0x01FF (50% PWM), 0x03FF (100% PWM), etc.

AT Command: ATM0
Parameter Range:0 – 0x03FF [steps]
Default Parameter Value:0
Related Commands: P0 (PWM0 Enable), AC (Apply Changes), CN (Exit Command Mode)
Minimum Firmware Version Required: v1.xA0

M1 (PWM1 Output Level) Command

<I/O Settings> The M1 command is used to set/read the output level of the PWM1 line (pin 7).

Before setting the line as an output:

1. Enable PWM1 output (P1 = 2)
2. Apply settings (use CN or AC)

AT Command: ATM1
Parameter Range:0 – 0x03FF
Default Parameter Value:0
Related Commands: P1 (PWM1 Enable), AC (Apply Changes), CN (Exit Command Mode)
Minimum Firmware Version Required: v1.xA0

MM (MAC Mode) Command

<Networking {Addressing}> The MM command is used to set and read the MAC Mode value. The MM command disables/enables the use of a MaxStream header contained in the 802.15.4 RF packet. By default (MM = 0), MaxStream Mode is enabled and the module adds an extra header to the data portion of the 802.15.4 packet. This enables the following features:

- ND and DN command support
- Duplicate packet detection when using ACKs

The MM command allows users to turn off the use of the extra header. Modes 1 and 2 are strict 802.15.4 modes. If the MaxStream header is disabled, ND and DN parameters are also disabled.

Note: When MM > 0, application and CCA failure retries are not supported.

AT Command: ATMM

Parameter Range:0 – 2

Parameter	Configuration
0	MaxStream Mode (802.15.4 + MaxStream header)
1	802.15.4 (no ACKs)
2	802.15.4 (with ACKs)

Default Parameter Value:0

Related Commands: ND (Node Discover), DN (Destination Node)

Minimum Firmware Version Required: v1.x80

MY (16-bit Source Address) Command

<Networking {Addressing}> The MY command is used to set and read the 16-bit source address of the RF module.

By setting MY to 0xFFFF, the reception of RF packets having a 16-bit address is disabled. The 64-bit address is the module's serial number and is always enabled.

AT Command: ATMY

Parameter Range: 0 – 0xFFFF

Default Parameter Value: 0

Related Commands: DH (Destination Address High), DL (Destination Address Low), CH (Channel), ID (PAN ID)

NB (Parity) Command

<Serial Interfacing> The NB command is used to select/read the parity settings of the RF module for UART communications.

AT Command: ATNB

Parameter Range: 0 – 4

Parameter	Configuration
0	8-bit (no parity or 7-bit (any parity)
1	8-bit even
2	8-bit odd
3	8-bit mark
4	8-bit space

Default Parameter Value: 0

Number of bytes returned: 1

ND (Node Discover) Command

<Networking {Identification}> The ND command is used to discover and report all modules on its current operating channel (CH parameter) and PAN ID (ID parameter). ND also accepts an NI (Node Identifier) value as a parameter. In this case, only a module matching the supplied identifier will respond.

AT Command: ATND
Range: optional 20-character NI value
Related Commands: CH (Channel), ID (Pan ID), MY (Source Address), SH (Serial Number High), SL (Serial Number Low), NI (Node Identifier), NT (Node Discover Time)
Minimum Firmware Version Required: v1.x80

ND uses a 64-bit long address when sending and responding to an ND request. The ND command causes a module to transmit a globally addressed ND command packet. The amount of time allowed for responses is determined by the NT (Node Discover Time) parameter.

In AT Command mode, command completion is designated by a carriage return (0x0D). Since two carriage returns end a command response, the application will receive three carriage returns at the end of the command. If no responses are received, the application should only receive one carriage return. When in API mode, the application should receive a frame (with no data) and status (set to 'OK') at the end of the command. When the ND command packet is received, the remote sets up a random time delay (up to 2.2 sec) before replying as follows:

Node Discover Response (AT command mode format - Transparent operation):
MY (Source Address) value<CR>
SH (Serial Number High) value<CR>
SL (Serial Number Low) value<CR>
DB (Received Signal Strength) value<CR>
NI (Node Identifier) value<CR>
<CR> (This is part of the response and not the end of command indicator.)

Node Discover Response (API format - data is binary (except for NI)):
2 bytes for MY (Source Address) value
4 bytes for SH (Serial Number High) value
4 bytes for SL (Serial Number Low) value
1 byte for DB (Received Signal Strength) value
NULL-terminated string for NI (Node Identifier) value (max 20 bytes w/out NULL terminator)

NI (Node Identifier) Command

<Networking {Identification}> The NI command is used to set and read a string for identifying a particular node.

AT Command: ATNI
Parameter Range: 20-character ASCII string
Related Commands: ND (Node Discover), DN (Destination Node)
Minimum Firmware Version Required: v1.x80

Rules:

- Register only accepts printable ASCII data.
- A string can not start with a space.
- A carriage return ends command
- Command will automatically end when maximum bytes for the string have been entered.

This string is returned as part of the ND (Node Discover) command. This identifier is also used with the DN (Destination Node) command.

NT (Node Discover Time) Command

<Networking {Identification}> The NT command is used to set the amount of time a base node will wait for responses from other nodes when using the ND (Node Discover) command. The NT value is transmitted with the ND command.

AT Command: ATNT
Parameter Range: 0x01 - 0xFC [x 100 msec]
Default: 0x19 (2.5 decimal seconds)
Related Commands: ND (Node Discover)
Minimum Firmware Version Required: 1.xA0

Remote nodes will set up a random hold-off time based on this time. The remotes will adjust this time down by 250 ms to give each node the ability to respond before the base ends the command. Once the ND command has ended, any response received on the base would be discarded.

P0 (PWM0 Configuration) Command

<I/O Setting {I/O Line Passing}> The P0 command is used to select/read the function for PWM0 (Pulse Width Modulation output 0). This command enables the option of translating incoming data to a PWM so that the output can be translated back into analog form.

With the IA (I/O Input Address) parameter correctly set, AD0 values can automatically be passed to PWM0.

AT Command: ATP0
The second character in the command is the number zero (“0”), not the letter “O”.

Parameter Range: 0 – 2

Parameter	Configuration
0	Disabled
1	RSSI
2	PWM0 Output

Default Parameter Value: 1

P1 (PWM1 Configuration) Command

<I/O Setting {I/O Line Passing}> The P1 command is used to select/read the function for PWM1 (Pulse Width Modulation output 1). This command enables the option of translating incoming data to a PWM so that the output can be translated back into analog form.

With the IA (I/O Input Address) parameter correctly set, AD1 values can automatically be passed to PWM1.

AT Command: ATP1

Parameter Range: 0 – 2

Parameter	Configuration
0	Disabled
1	RSSI
2	PWM1 Output

Default Parameter Value: 0

Minimum Firmware Version Required: v1.xA0

PL (Power Level) Command

<RF Interfacing> The PL command is used to select and read the power level at which the RF module transmits conducted power.

WHEN OPERATING IN EUROPE:
XBee-PRO RF Modules must be configured to operate at a maximum transmit power output level of 10 dBm. The PL parameter must equal “0” (10 dBm).

Additionally, European regulations stipulate an EIRP power maximum of 12.86 dBm (19 mW) for the XBee-PRO and 12.11 dBm for the XBee when integrating high-gain antennas.

AT Command: ATPL

Parameter Range: 0 – 4

Parameter	XBee	XBee-PRO
0	-10 dBm	10 dBm
1	-6 dBm	12 dBm
2	-4 dBm	14 dBm
3	-2 dBm	16 dBm
4	0 dBm	18 dBm

Default Parameter Value: 4

WHEN OPERATING IN JAPAN:
XBee-PRO RF Modules optimized for use in Japan contain firmware that limits transmit power output to 10 dBm. If PL=4 (default), the maximum power output level is 10 dBm. For a list of module part numbers approved for use in Japan, contact MaxStream [call 1-801-765-9885 or send e-mail to sales@maxstream.net].

PR (Pull-up Resistor Enable) Command

<Serial Interfacing> The PR command is used to set and read the bit field that is used to configure internal the pull-up resistor status for I/O lines. “1” specifies the pull-up resistor is enabled. “0” specifies no pull up.

bit 0 - AD4/DIO4 (pin 11)
bit 1 - AD3/DIO3 (pin 17)
bit 2 - AD2/DIO2 (pin 18)
bit 3 - AD1/DIO1 (pin 19)
bit 4 - AD0/DIO0 (pin 20)
bit 5 - AD6/DIO6 (pin 16)
bit 6 - DI8 (pin 9)
bit 7 - DIN/CONFIG (pin 3)

AT Command: ATPR
Parameter Range: 0 – 0xFF
Default Parameter Value: 0xFF (all pull–up resistors are enabled)
Minimum Firmware Version Required: v1.x80

For example: Sending the command “ATPR 6F” will turn bits 0, 1, 2, 3, 5 and 6 ON; and bits 4 & 7 will be turned OFF. (The binary equivalent of “0x6F” is “01101111”. Note that ‘bit 0’ is the last digit in the bitfield.

PT (PWM Output Timeout) Command

<I/O Settings {I/O Line Passing}> The PT command is used to set/read the output timeout value for both PWM outputs.

When PWM is set to a non-zero value: Due to I/O line passing, a time is started which when expired will set the PWM output to zero. The timer is reset when a valid I/O packet is received.

AT Command: ATPT
Parameter Range: 0 – 0xFF [x 100 msec]
Default Parameter Value: 0xFF
Minimum Firmware Version Required: 1.xA0

RE (Restore Defaults) Command

<(Special)> The RE command is used to restore all configurable parameters to their factory default settings. The RE command does not write restored values to non-volatile (persistent) memory. Issue the WR (Write) command subsequent to issuing the RE command to save restored parameter values to non-volatile memory.

AT Command: ATRE

RN (Random Delay Slots) Command

<Networking & Security> The RN command is used to set and read the minimum value of the back-off exponent in the CSMA-CA algorithm. The CSMA-CA algorithm was engineered for collision avoidance (random delays are inserted to prevent data loss caused by data collisions).

AT Command: ATRN
Parameter Range: 0 – 3 [exponent]
Default Parameter Value: 0

If RN = 0, collision avoidance is disabled during the first iteration of the algorithm (802.15.4 - macMinBE).

CSMA-CA stands for "Carrier Sense Multiple Access - Collision Avoidance". Unlike CSMA-CD (reacts to network transmissions after collisions have been detected), CSMA-CA acts to prevent data collisions before they occur. As soon as a module receives a packet that is to be transmitted, it checks if the channel is clear (no other module is transmitting). If the channel is clear, the packet is sent over-the-air. If the channel is not clear, the module waits for a randomly selected period of time, then checks again to see if the channel is clear. After a time, the process ends and the data is lost.

RO (Packetization Timeout) Command

AT Command: ATRO
Parameter Range:0 - 0xFF [x character times]
Default Parameter Value: 3

<Serial Interfacing> RO command is used to set and read the number of character times of inter-character delay required before transmission.

RF transmission commences when data is detected in the DI (data in from host) buffer and RO character times of silence are detected on the UART receive lines (after receiving at least 1 byte).

RF transmission will also commence after 100 Bytes (maximum packet size) are received in the DI buffer.

Set the RO parameter to '0' to transmit characters as they arrive instead of buffering them into one RF packet.

RP (RSSI PWM Timer) Command

AT Command: ATRP
Parameter Range:0 - 0xFF [x 100 msec]
Default Parameter Value: 0x28 (40 decimal)

<I/O Settings {I/O Line Passing}> The RP command is used to enable PWM (Pulse Width Modulation) output on the RF module. The output is calibrated to show the level a received RF signal is above the sensitivity level of the module. The PWM pulses vary from 24 to 100%. Zero percent means PWM output is inactive. One to 24% percent means the received RF signal is at or below the published sensitivity level of the module. The following table shows levels above sensitivity and PWM values.

The total period of the PWM output is 64 µs. Because there are 445 steps in the PWM output, the minimum step size is 144 ns.

PWM Percentages

dB above Sensitivity	PWM percentage (high period / total period)
10	41%
20	58%
30	75%

A non-zero value defines the time that the PWM output will be active with the RSSI value of the last received RF packet. After the set time when no RF packets are received, the PWM output will be set low (0 percent PWM) until another RF packet is received. The PWM output will also be set low at power-up until the first RF packet is received. A parameter value of 0xFF permanently enables the PWM output and it will always reflect the value of the last received RF packet.

RR (XBee Retries) Command

AT Command: ATRR
Parameter Range: 0 - 6
Default: 0
Minimum Firmware Version Required: 1.xA0

<Networking {Addressing}> The RR command is used set/read the maximum number of retries the module will execute in addition to the 3 retries provided by the 802.15.4 MAC. For each XBee retry, the 802.15.4 MAC can execute up to 3 retries.

This values does not need to be set on all modules for retries to work. If retries are enabled, the transmitting module will set a bit in the Maxstream RF Packet header which requests the receiving module to send an ACK (acknowledgement). If the transmitting module does not receive an ACK within 200 msec, it will re-send the packet within a random period up to 48 msec. Each XBee retry can potentially result in the MAC sending the packet 4 times (1 try plus 3 retries). Note that retries are not attempted for packets that are purged when transmitting with a Cyclic Sleep Coordinator.

SC (Scan Channels) Command

<Networking {Association}> The SC command is used to set and read the list of channels to scan for all Active and Energy Scans as a bit field.

This affects scans initiated in command mode [AS (Active Scan) and ED (Energy Scan) commands] and during End Device Association and Coordinator startup.

AT Command: ATSC
Parameter Range: 0 – 0xFFFF [Bitfield] (bits 0, 14, 15 are not allowed when using the XBee-PRO)
Default Parameter Value: 0x1FFE (all XBee-PRO channels)
Related Commands: ED (Energy Scan), SD (Scan Duration)
Minimum Firmware Version Required: v1.x80

bit 0 - 0x0B	bit 4 - 0x0F	bit 8 - 0x13	bit 12 - 0x17
bit 1 - 0x0C	bit 5 - 0x10	bit 9 - 0x14	bit 13 - 0x18
bit 2 - 0x0D	bit 6 - 0x11	bit 10 - 0x15	bit 14 - 0x19
bit 3 - 0x0E	bit 7 - 0x12	bit 11 - 0x16	bit 15 - 0x1A

SD (Scan Duration) Command

<Networking {Association}> The SD command is used to set and read the exponent value that determines the duration (in time) of a scan.

End Device (Duration of Active Scan during Association) - In a Beacon system, set SD = BE of the Coordinator. SD must be set at least to the highest BE parameter of any Beaconing Coordinator with which an End Device or Coordinator wish to discover.

AT Command: ATSD
Parameter Range: 0 – 0x0F
Default Parameter Value: 4
Related Commands: ED (Energy Scan), SC (Scan Channel)
Minimum Firmware Version Required: v1.x80

Coordinator - If the 'ReassignPANID' option is set on the Coordinator [refer to A2 parameter], the SD parameter determines the length of time the Coordinator will scan channels to locate existing PANs. If the 'ReassignChannel' option is set, SD determines how long the Coordinator will perform an Energy Scan to determine which channel it will operate on.

Scan Time is measured as ((# of Channels to Scan) * (2 ^ SD) * 15.36ms). The number of channels to scan is set by the SC command. The XBee RF Module can scan up to 16 channels (SC = 0xFFFF). The XBee PRO RF Module can scan up to 12 channels (SC = 0x1FFE).

Examples: Values below show results for a 12-channel scan

If SD = 0, time = 0.18 sec	SD = 8, time = 47.19 sec
SD = 2, time = 0.74 sec	SD = 10, time = 3.15 min
SD = 4, time = 2.95 sec	SD = 12, time = 12.58 min
SD = 6, time = 11.80 sec	SD = 14, time = 50.33 min

SH (Serial Number High) Command

<Diagnostics> The SH command is used to read the high 32 bits of the RF module's unique IEEE 64-bit address.

The module serial number is set at the factory and is read-only.

AT Command: ATSH
Parameter Range: 0 – 0xFFFFFFFF [read-only]
Related Commands: SL (Serial Number Low), MY (Source Address)

SL (Serial Number Low) Command

<Diagnostics> The SL command is used to read the low 32 bits of the RF module's unique IEEE 64-bit address.

The module serial number is set at the factory and is read-only.

AT Command: ATSL
Parameter Range: 0 – 0xFFFFFFFF [read-only]
Related Commands: SH (Serial Number High), MY (Source Address)

SM (Sleep Mode) Command

<Sleep Mode (Low Power)> The SM command is used to set and read Sleep Mode settings. By default, Sleep Modes are disabled (SM = 0) and the RF module remains in Idle/Receive Mode. When in this state, the module is constantly ready to respond to either serial or RF activity.

SM command options vary according to the networking system type. By default, the module is configured to operate in a NonBeacon system.

* The Sleep Coordinator option (SM=6) only exists for backwards compatibility with firmware version 1.x06 only. In all other cases, use the CE command to enable a Coordinator.

AT Command: ATSM

Parameter Range: 0 – 6

Parameter	Configuration
0	Disabled
1	Pin Hibernate
2	Pin Doze
3	(reserved)
4	Cyclic Sleep Remote
5	Cyclic Sleep Remote (with Pin Wake-up)
6	Sleep Coordinator*

Default Parameter Value: 0

Related Commands: SP (Cyclic Sleep Period), ST (Time before Sleep)

SP (Cyclic Sleep Period) Command

<Sleep Mode (Low Power)> The SP command is used to set and read the duration of time in which a remote RF module sleeps. After the cyclic sleep period is over, the module wakes and checks for data. If data is not present, the module goes back to sleep. The maximum sleep period is 268 seconds (SP = 0x68B0).

The SP parameter is only valid if the module is configured to operate in Cyclic Sleep (SM = 4-6). Coordinator and End Device SP values should always be equal.

To send Direct Messages, set SP = 0.

AT Command: ATSP	
Parameter Range:	NonBeacon Firmware: 1 – 0x68B0 [x 10 milliseconds]
Default Parameter Value:	NonBeacon Firmware: 0
Related Commands: SM (Sleep Mode), ST (Time before Sleep), DP (Disassociation Cyclic Sleep Period, BE (Beacon Order)	

NonBeacon Firmware

End Device - SP determines the sleep period for cyclic sleeping remotes. Maximum sleep period is 268 seconds (0x68B0).

Coordinator - If non-zero, SP determines the time to hold an indirect message before discarding it. A Coordinator will discard indirect messages after a period of (2.5 * SP).

ST (Time before Sleep) Command

<Sleep Mode (Low Power)> The ST command is used to set and read the period of inactivity (no serial or RF data is sent or received) before activating Sleep Mode.

NonBeacon Firmware

Set/Read time period of inactivity (no serial or RF data is sent or received) before activating Sleep Mode. ST parameter is only valid with Cyclic Sleep settings (SM = 4 - 5).

Coordinator and End Device ST values must be equal.

AT Command: ATST	
Parameter Range:	NonBeacon Firmware: 1 – 0xFFFF [x 1 millisecond]
Default Parameter Value:	NonBeacon Firmware: 0x1388 (5000 decimal)
Related Commands: SM (Sleep Mode), ST (Time before Sleep)	

T0 - T7 ((D0-D7) Output Timeout) Command

AT Commands: ATT0 – ATT7
Parameter Range:0 – 0xFF [x 100 msec]
Default Parameter Value:0xFF
Minimum Firmware Version Required: v1.xA0

<I/O Settings {I/O Line Passing}> The T0, T1, T2, T3, T4, T5, T6 and T7 commands are used to set/read output timeout values for the lines that correspond with the D0 - D7 parameters. When output is set (due to I/O line passing) to a non-default level, a timer is started which when expired, will set the output to its default level. The timer is reset when a valid I/O packet is received. The Tn parameter defines the permissible amount of time to stay in a non-default (active) state. If Tn = 0, Output Timeout is disabled (output levels are held indefinitely).

VL (Firmware Version - Verbose)

AT Command: ATVL
Parameter Range:0 – 0xFF [x 100 milliseconds]
Default Parameter Value: 0x28 (40 decimal)
Minimum Firmware Version Required: v1.x80

<Diagnostics> The VL command is used to read detailed version information about the RF module. The information includes:
application build date; MAC, PHY and bootloader versions; and build dates.

VR (Firmware Version) Command

AT Command: ATVR
Parameter Range: 0 – 0xFFFF [read only]

<Diagnostics> The VR command is used to read which firmware version is stored in the module.

XBee version numbers will have four significant digits. The reported number will show three or four numbers and is stated in hexadecimal notation. A version can be reported as "ABC" or "ABCD". Digits ABC are the main release number and D is the revision number from the main release. "D" is not required and if it is not present, a zero is assumed for D. "B" is a variant designator. The following variants exist:

- "0" = Non-Beacon Enabled 802.15.4 Code
- "1" = Beacon Enabled 802.15.4 Code

WR (Write) Command

AT Command: ATWR

<(Special)> The WR command is used to write configurable parameters to the RF module's non-volatile memory. Parameter values remain in the module's memory until overwritten by subsequent use of the WR Command.

If changes are made without writing them to non-volatile memory, the module reverts back to previously saved parameters the next time the module is powered-on.

NOTE: Once the WR command is sent to the module, no additional characters should be sent until after the “OK/r” response is received.

3.4. API Operation

By default, XBee/XBee-PRO RF Modules act as a serial line replacement (Transparent Operation) - all UART data received through the DI pin is queued up for RF transmission. When the module receives an RF packet, the data is sent out the DO pin with no additional information.

Inherent to Transparent Operation are the following behaviors:

- If module parameter registers are to be set or queried, a special operation is required for transitioning the module into Command Mode.
- In point-to-multipoint systems, the application must send extra information so that the receiving module(s) can distinguish between data coming from different remotes.

As an alternative to the default Transparent Operation, API (Application Programming Interface) Operations are available. API operation requires that communication with the module be done through a structured interface (data is communicated in frames in a defined order). The API specifies how commands, command responses and module status messages are sent and received from the module using a UART Data Frame.

3.4.1. API Frame Specifications

Two API modes are supported and both can be enabled using the AP (API Enable) command. Use the following AP parameter values to configure the module to operate in a particular mode:

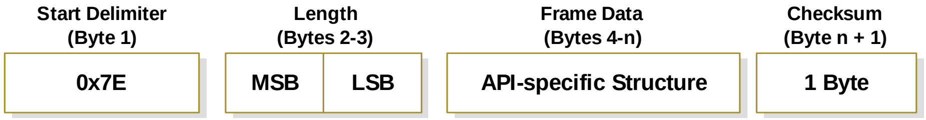
- AP = 0 (default): Transparent Operation (UART Serial line replacement)
  API modes are disabled.
- AP = 1: API Operation
- AP = 2: API Operation (with escaped characters)

Any data received prior to the start delimiter is silently discarded. If the frame is not received correctly or if the checksum fails, the data is silently discarded.

API Operation (AP parameter = 1)

When this API mode is enabled (AP = 1), the UART data frame structure is defined as follows:

Figure 3-01. UART Data Frame Structure:

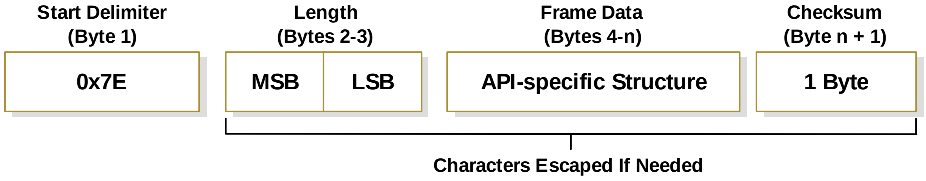


MSB = Most Significant Byte, LSB = Least Significant Byte

API Operation - with Escape Characters (AP parameter = 2)

When this API mode is enabled (AP = 2), the UART data frame structure is defined as follows:

Figure 3-02. UART Data Frame Structure - with escape control characters:



MSB = Most Significant Byte, LSB = Least Significant Byte

Escape characters. When sending or receiving a UART data frame, specific data values must be escaped (flagged) so they do not interfere with the UART or UART data frame operation. To escape an interfering data byte, insert 0x7D and follow it with the byte to be escaped XOR'd with 0x20.

Data bytes that need to be escaped:

- 0x7E – Frame Delimiter
- 0x7D – Escape
- 0x11 – XON
- 0x13 – XOFF

Example - Raw UART Data Frame (before escaping interfering bytes):
0x7E 0x00 0x02 0x23 0x11 0xCB

0x11 needs to be escaped which results in the following frame:
0x7E 0x00 0x02 0x23 0x7D 0x31 0xCB

Note: In the above example, the length of the raw data (excluding the checksum) is 0x0002 and the checksum of the non-escaped data (excluding frame delimiter and length) is calculated as: 0xFF - (0x23 + 0x11) = (0xFF - 0x34) = 0xCB.

Checksum

To test data integrity, a checksum is calculated and verified on non-escaped data.

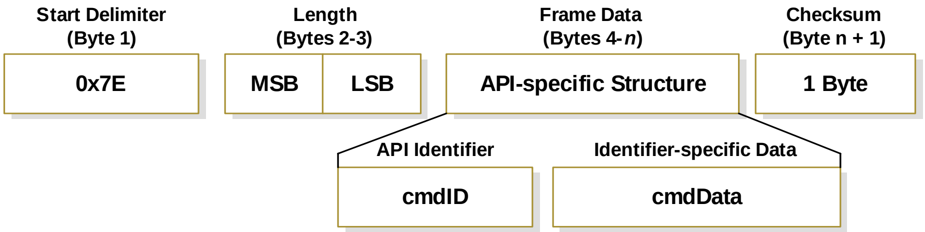
To calculate: Not including frame delimiters and length, add all bytes keeping only the lowest 8 bits of the result and subtract from 0xFF.

To verify: Add all bytes (include checksum, but not the delimiter and length). If the checksum is correct, the sum will equal 0xFF.

3.4.2. API Types

Frame data of the UART data frame forms an API-specific structure as follows:

Figure 3-03. UART Data Frame & API-specific Structure:

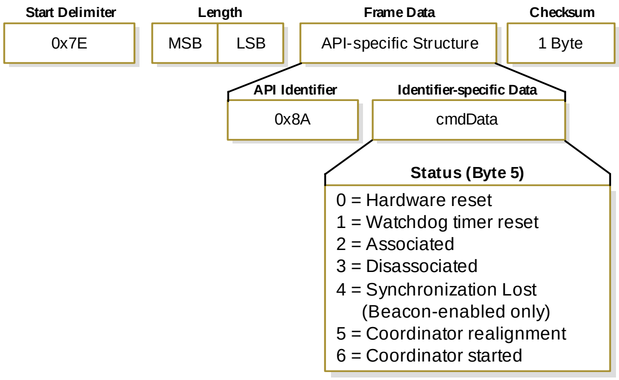


The cmdID frame (API-identifier) indicates which API messages will be contained in the cmdData frame (Identifier-specific data). Refer to the sections that follow for more information regarding the supported API types. Note that multi-byte values are sent big endian.

Modem Status

API Identifier: 0x8A
RF module status messages are sent from the module in response to specific conditions.

Figure 3-04. Modem Status Frames



AT Command

API Identifier Value: 0x08

The “AT Command” API type allows for module parameters to be queried or set. When using this command ID, new parameter values are applied immediately. This includes any register set with the “AT Command - Queue Parameter Value” (0x09) API type.

Figure 3-05. AT Command Frames

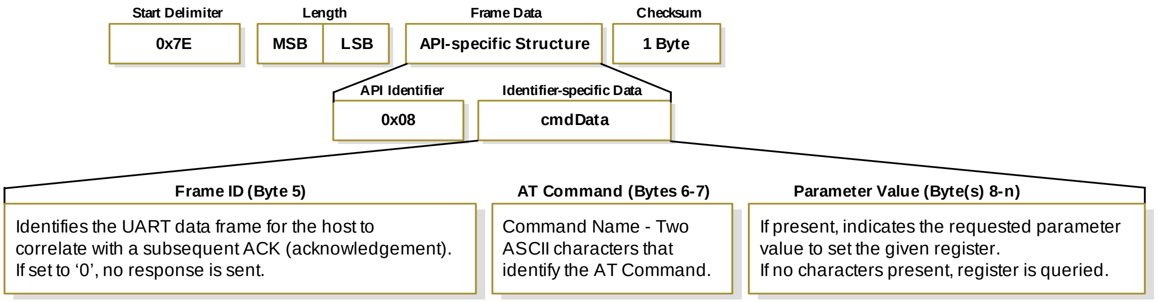
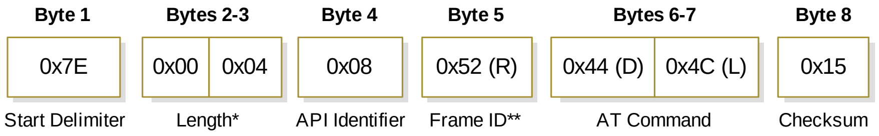

Start Delimiter
0x7E
Length
MSB
LSB
Frame Data
API-specific Structure
Checksum
1 Byte
API Identifier
0x08
Identifier-specific Data
cmdData
Frame ID (Byte 5)
Identifies the UART data frame for the host to correlate with a subsequent ACK (acknowledgement). If set to ‘0’, no response is sent.
AT Command (Bytes 6-7)
Command Name - Two ASCII characters that identify the AT Command.
Parameter Value (Byte(s) 8-n)
If present, indicates the requested parameter value to set the given register. If no characters present, register is queried.

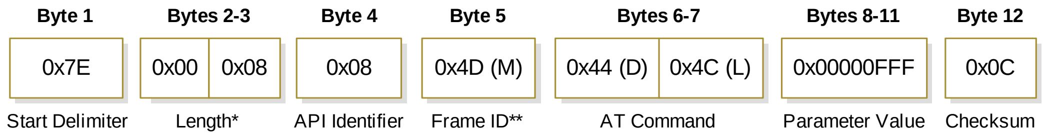
Figure 3-06. Example: API frames when reading the DL parameter value of the module.



** Length [Bytes] = API Identifier + Frame ID + AT Command*

*** “R” value was arbitrarily selected.*

Figure 3-07. Example: API frames when modifying the DL parameter value of the module.



** Length [Bytes] = API Identifier + Frame ID + AT Command + Parameter Value*

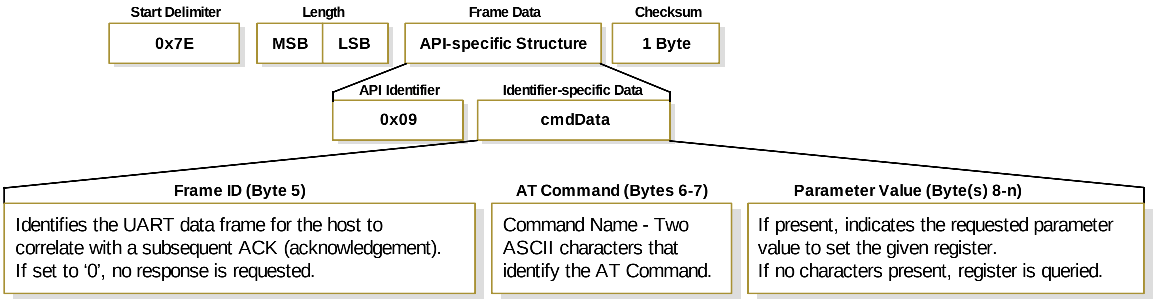
*** “M” value was arbitrarily selected.*

AT Command - Queue Parameter Value

API Identifier Value: 0x09

This API type allows module parameters to be queried or set. In contrast to the “AT Command” API type, new parameter values are queued and not applied until either the “AT Command” (0x08) API type or the AC (Apply Changes) command is issued. Register queries (reading parameter values) are returned immediately.

Figure 3-08. AT Command Frames
(Note that frames are identical to the “AT Command” API type except for the API identifier.)

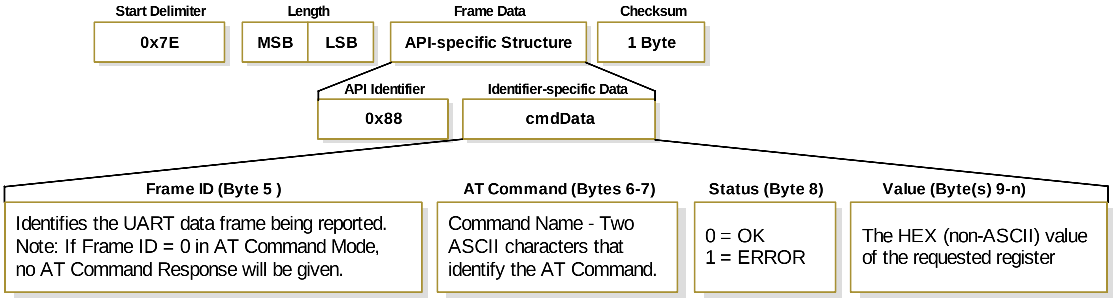

Start Delimiter
0x7E
Length
MSB
LSB
Frame Data
API-specific Structure
Checksum
1 Byte
API Identifier
0x09
Identifier-specific Data
cmdData
Frame ID (Byte 5)
Identifies the UART data frame for the host to correlate with a subsequent ACK (acknowledgement). If set to ‘0’, no response is requested.
AT Command (Bytes 6-7)
Command Name - Two ASCII characters that identify the AT Command.
Parameter Value (Byte(s) 8-n)
If present, indicates the requested parameter value to set the given register. If no characters present, register is queried.

AT Command Response

API Identifier Value: 0x88
Response to previous command.

In response to an AT Command message, the module will send an AT Command Response message. Some commands will send back multiple frames (for example, the ND (Node Discover) and AS (Active Scan) commands). These commands will end by sending a frame with a status of ATCMD_OK and no cmdData.

Figure 3-09. AT Command Response Frames.



Identifies the UART data frame being reported. Note: If Frame ID = 0 in AT Command Mode, no AT Command Response will be given.

Command Name - Two ASCII characters that identify the AT Command.

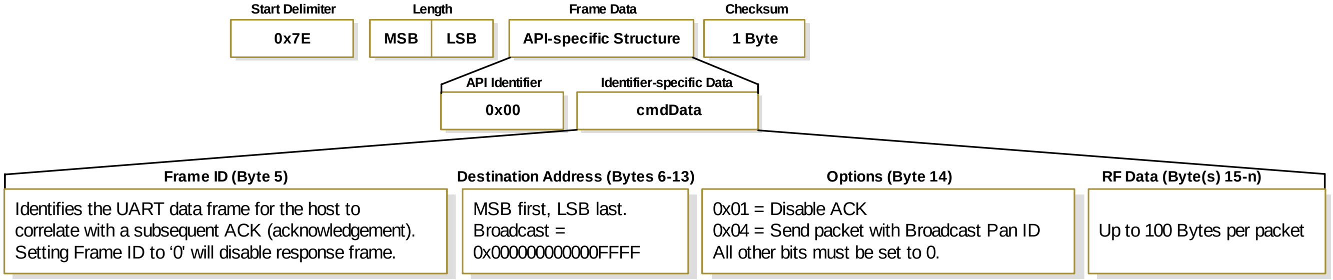
0 = OK
1 = ERROR

The HEX (non-ASCII) value of the requested register

TX (Transmit) Request: 64-bit address

API Identifier Value: 0x00
A TX Request message will cause the module to send RF Data as an RF Packet.

Figure 3-10. TX Packet (64-bit address) Frames



Identifies the UART data frame for the host to correlate with a subsequent ACK (acknowledgement). Setting Frame ID to '0' will disable response frame.

MSB first, LSB last.
Broadcast =
0x000000000000FFFF

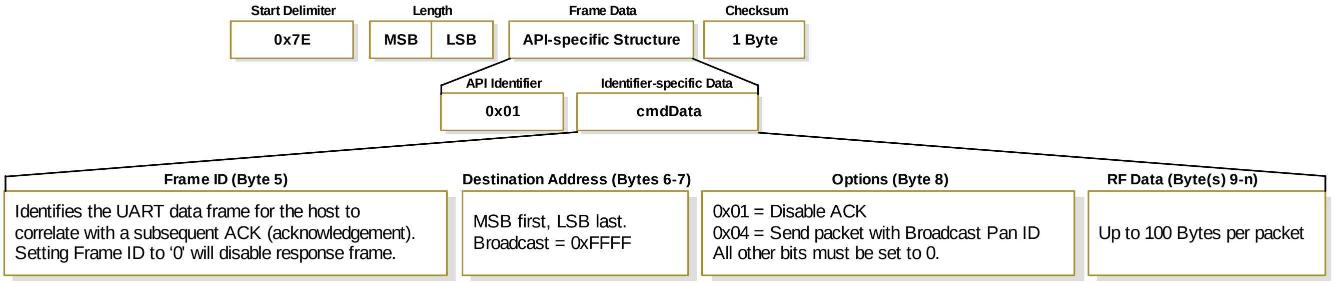
0x01 = Disable ACK
0x04 = Send packet with Broadcast Pan ID
All other bits must be set to 0.

Up to 100 Bytes per packet

TX (Transmit) Request: 16-bit address

API Identifier Value: 0x01
A TX Request message will cause the module to send RF Data as an RF Packet.

Figure 3-11. TX Packet (16-bit address) Frames



Identifies the UART data frame for the host to correlate with a subsequent ACK (acknowledgement). Setting Frame ID to '0' will disable response frame.

MSB first, LSB last.
Broadcast = 0xFFFF

0x01 = Disable ACK
0x04 = Send packet with Broadcast Pan ID
All other bits must be set to 0.

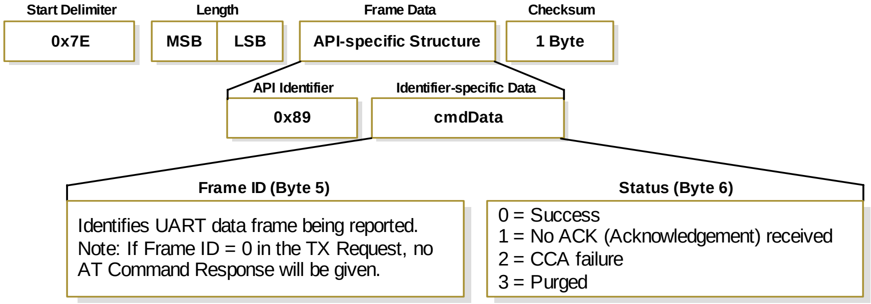
Up to 100 Bytes per packet

TX (Transmit) Status

API Identifier Value: 0x89

When a TX Request is completed, the module sends a TX Status message. This message will indicate if the packet was transmitted successfully or if there was a failure.

Figure 3-12. TX Status Frames



Identifies UART data frame being reported. Note: If Frame ID = 0 in the TX Request, no AT Command Response will be given.

0 = Success
1 = No ACK (Acknowledgement) received
2 = CCA failure
3 = Purged

NOTES:

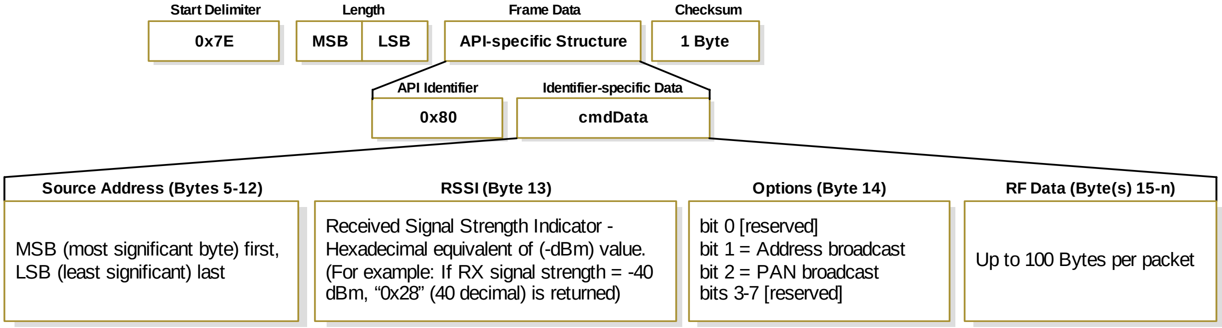
- "STATUS = 1" occurs when all retries are expired and no ACK is received.
- If transmitter broadcasts (destination address = 0x000000000000FFFF), only "STATUS = 0 or 2" will be returned.
- "STATUS = 3" occurs when Coordinator times out of an indirect transmission. Timeout is defined as (2.5 x SP (Cyclic Sleep Period) parameter value).

RX (Receive) Packet: 64-bit Address

API Identifier Value: 0x80

When the module receives an RF packet, it is sent out the UART using this message type.

Figure 3-13. RX Packet (64-bit address) Frames



MSB (most significant byte) first, LSB (least significant) last

Received Signal Strength Indicator - Hexadecimal equivalent of (-dBm) value. (For example: If RX signal strength = -40 dBm, "0x28" (40 decimal) is returned)

bit 0 [reserved]
bit 1 = Address broadcast
bit 2 = PAN broadcast
bits 3-7 [reserved]

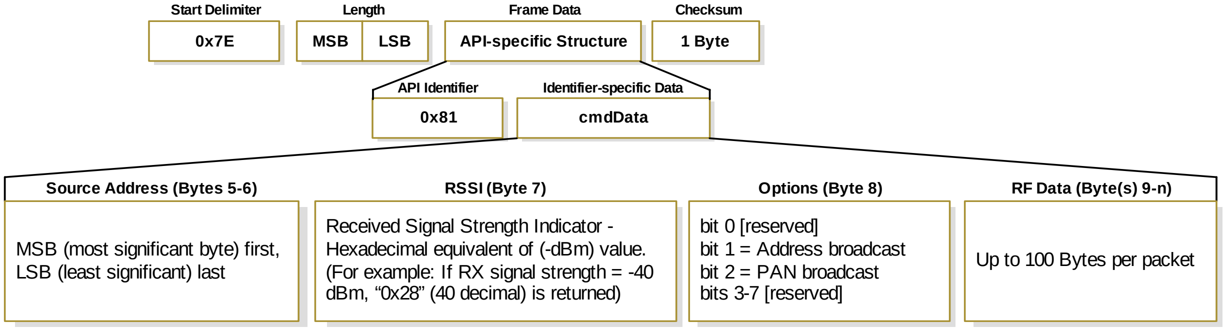
Up to 100 Bytes per packet

RX (Receive) Packet: 16-bit Address

API Identifier Value: 0x81

When the module receives an RF packet, it is sent out the UART using this message type.

Figure 3-14. RX Packet (16-bit address) Frames



MSB (most significant byte) first, LSB (least significant) last

Received Signal Strength Indicator - Hexadecimal equivalent of (-dBm) value. (For example: If RX signal strength = -40 dBm, "0x28" (40 decimal) is returned)

bit 0 [reserved]
bit 1 = Address broadcast
bit 2 = PAN broadcast
bits 3-7 [reserved]

Up to 100 Bytes per packet

Appendix A: Agency Certifications

United States (FCC)

XBee/XBee-PRO RF Modules comply with Part 15 of the FCC rules and regulations. Compliance with the labeling requirements, FCC notices and antenna usage guidelines is required.

To fulfill FCC Certification requirements, the OEM must comply with the following regulations:

1. The system integrator must ensure that the text on the external label provided with this device is placed on the outside of the final product [Figure A-01].
2. XBee/XBee-PRO RF Modules may only be used with antennas that have been tested and approved for use with this module [refer to the antenna tables in this section].

OEM Labeling Requirements

WARNING: The Original Equipment Manufacturer (OEM) must ensure that FCC labeling requirements are met. This includes a clearly visible label on the outside of the final product enclosure that displays the contents shown in the figure below.

Figure A-01. Required FCC Label for OEM products containing the XBee/XBee-PRO RF Module

Contains FCC ID: OUR-XBEE/OUR-XBEEPRO**

The enclosed device complies with Part 15 of the FCC Rules. Operation is subject to the following two conditions: (*i.*) this device may not cause harmful interference and (*ii.*) this device must accept any interference received, including interference that may cause undesired operation.

* The FCC ID for the XBee is “OUR-XBEE”. The FCC ID for the XBee-PRO is “OUR-XBEEPRO”.

FCC Notices

IMPORTANT: The XBee/XBee-PRO OEM RF Module has been certified by the FCC for use with other products without any further certification (as per FCC section 2.1091). Modifications not expressly approved by MaxStream could void the user's authority to operate the equipment.

IMPORTANT: OEMs must test final product to comply with unintentional radiators (FCC section 15.107 & 15.109) before declaring compliance of their final product to Part 15 of the FCC Rules.

IMPORTANT: The RF module has been certified for remote and base radio applications. If the module will be used for portable applications, the device must undergo SAR testing.

This equipment has been tested and found to comply with the limits for a Class B digital device, pursuant to Part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation. This equipment generates, uses and can radiate radio frequency energy and, if not installed and used in accordance with the instructions, may cause harmful interference to radio communications. However, there is no guarantee that interference will not occur in a particular installation.

If this equipment does cause harmful interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures: Re-orient or relocate the receiving antenna, Increase the separation between the equipment and receiver, Connect equipment and receiver to outlets on different circuits, or Consult the dealer or an experienced radio/TV technician for help.

FCC-Approved Antennas (2.4 GHz)

XBee/XBee-PRO RF Modules can be installed using antennas and cables constructed with standard connectors (Type-N, SMA, TNC, etc.) if the installation is performed professionally and according to FCC guidelines. For installations not performed by a professional, non-standard connectors (RPSMA, RPTNC, etc) must be used.

The modules are FCC-approved for fixed base station and mobile applications on channels 0x0B - 0x1A (XBee) and 0x0C - 0x17 (XBee-PRO). If the antenna is mounted at least 20cm (8 in.) from nearby persons, the application is considered a mobile application. Antennas not listed in the table must be tested to comply with FCC Section 15.203 (Unique Antenna Connectors) and Section 15.247 (Emissions).

XBee OEM RF Modules (1 mW): XBee Modules have been tested and approved for use with all of the antennas listed in the tables below (Cable-loss IS NOT required).

XBee-PRO OEM RF Modules (60 mW): XBee-PRO Modules have been tested and approved for use with the antennas listed in the tables below (Cable-loss IS required when using antennas listed in Table A-02).

Table A-01. Antennas approved for use with the XBee/XBee-PRO RF Modules (Cable-loss is not required.)

Part Number	Type (Description)	Gain	Application*	Min. Separation
A24-HSM-450	Dipole (Half-wave articulated RPSMA - 4.5”)	2.1 dBi	Fixed/Mobile	20 cm
A24-HABSM	Dipole (Articulated RPSMA)	2.1 dBi	Fixed	20 cm
A24-HABUF-P5I	Dipole (Half-wave articulated bulkhead mount U.FL. w/ 5” pigtail)	2.1 dBi	Fixed	20 cm
A24-QI	Monopole (Integrated whip)	1.5 dBi	Fixed	20 cm

Table A-02. Antennas approved for use with the XBee RF Modules (Cable-loss is required)

Part Number	Type (Description)	Gain	Application*	Min. Separation	Required Cable-loss
Omni-Directional Class Antennas					
A24-Y6NF	Yagi (6-element)	8.8 dBi	Fixed	2 m	1.7 dB
A24-Y7NF	Yagi (7-element)	9.0 dBi	Fixed	2 m	1.9 dB
A24-Y9NF	Yagi (9-element)	10.0 dBi	Fixed	2 m	2.9 dB
A24-Y10NF	Yagi (10-element)	11.0 dBi	Fixed	2 m	3.9 dB
A24-Y12NF	Yagi (12-element)	12.0 dBi	Fixed	2 m	4.9 dB
A24-Y13NF	Yagi (13-element)	12.0 dBi	Fixed	2 m	4.9 dB
A24-Y15NF	Yagi (15-element)	12.5 dBi	Fixed	2 m	5.4 dB
A24-Y16NF	Yagi (16-element)	13.5 dBi	Fixed	2 m	6.4 dB
A24-Y16RM	Yagi (16-element, RPSMA connector)	13.5 dBi	Fixed	2 m	6.4 dB
A24-Y18NF	Yagi (18-element)	15.0 dBi	Fixed	2 m	7.9 dB
Omni-Directional Class Antennas					
A24-C1	Surface Mount	-1.5 dBi	Fixed/Mobile	20 cm	-
A24-F2NF	Omni-directional (Fiberglass base station)	2.1 dBi	Fixed/Mobile	20 cm	
A24-F3NF	Omni-directional (Fiberglass base station)	3.0 dBi	Fixed/Mobile	20 cm	
A24-F5NF	Omni-directional (Fiberglass base station)	5.0 dBi	Fixed/Mobile	20 cm	
A24-F8NF	Omni-directional (Fiberglass base station)	8.0 dBi	Fixed	2 m	
A24-F9NF	Omni-directional (Fiberglass base station)	9.5 dBi	Fixed	2 m	0.2 dB
A24-F10NF	Omni-directional (Fiberglass base station)	10.0 dBi	Fixed	2 m	0.7 dB
A24-F12NF	Omni-directional (Fiberglass base station)	12.0 dBi	Fixed	2 m	2.7 dB
A24-F15NF	Omni-directional (Fiberglass base station)	15.0 dBi	Fixed	2 m	5.7 dB
A24-W7NF	Omni-directional (Base station)	7.2 dBi	Fixed	2 m	
A24-M7NF	Omni-directional (Mag-mount base station)	7.2 dBi	Fixed	2 m	
Panel Class Antennas					
A24-P8SF	Flat Panel	8.5 dBi	Fixed	2 m	1.5 dB
A24-P8NF	Flat Panel	8.5 dBi	Fixed	2 m	1.5 dB
A24-P13NF	Flat Panel	13.0 dBi	Fixed	2 m	6 dB
A24-P14NF	Flat Panel	14.0 dBi	Fixed	2 m	7 dB
A24-P15NF	Flat Panel	15.0 dBi	Fixed	2 m	8 dB
A24-P16NF	Flat Panel	16.0 dBi	Fixed	2 m	9 dB

Table A-03. Antennas approved for use with the XBee/XBee-PRO RF Modules (Cable-loss is required)

Part Number	Type (Description)	Gain	Application*	Min. Separation	Required Cable-loss
A24-C1	Surface Mount	-1.5 dBi	Fixed/Mobile	20 cm	-
A24-Y4NF	Yagi (4-element)	6.0 dBi	Fixed	2 m	8.1 dB
A24-Y6NF	Yagi (6-element)	8.8 dBi	Fixed	2 m	10.9 dB
A24-Y7NF	Yagi (7-element)	9.0 dBi	Fixed	2 m	11.1 dB
A24-Y9NF	Yagi (9-element)	10.0 dBi	Fixed	2 m	12.1 dB
A24-Y10NF	Yagi (10-element)	11.0 dBi	Fixed	2 m	13.1 dB
A24-Y12NF	Yagi (12-element)	12.0 dBi	Fixed	2 m	14.1 dB
A24-Y13NF	Yagi (13-element)	12.0 dBi	Fixed	2 m	14.1 dB
A24-Y15NF	Yagi (15-element)	12.5 dBi	Fixed	2 m	14.6 dB
A24-Y16NF	Yagi (16-element)	13.5 dBi	Fixed	2 m	15.6 dB
A24-Y16RM	Yagi (16-element, RPSMA connector)	13.5 dBi	Fixed	2 m	15.6 dB
A24-Y18NF	Yagi (18-element)	15.0 dBi	Fixed	2 m	17.1 dB
A24-F2NF	Omni-directional (Fiberglass base station)	2.1 dBi	Fixed/Mobile	20 cm	4.2 dB
A24-F3NF	Omni-directional (Fiberglass base station)	3.0 dBi	Fixed/Mobile	20 cm	5.1 dB
A24-F5NF	Omni-directional (Fiberglass base station)	5.0 dBi	Fixed/Mobile	20 cm	7.1 dB
A24-F8NF	Omni-directional (Fiberglass base station)	8.0 dBi	Fixed	2 m	10.1 dB
A24-F9NF	Omni-directional (Fiberglass base station)	9.5 dBi	Fixed	2 m	11.6 dB
A24-F10NF	Omni-directional (Fiberglass base station)	10.0 dBi	Fixed	2 m	12.1 dB
A24-F12NF	Omni-directional (Fiberglass base station)	12.0 dBi	Fixed	2 m	14.1 dB
A24-F15NF	Omni-directional (Fiberglass base station)	15.0 dBi	Fixed	2 m	17.1 dB
A24-W7NF	Omni-directional (Base station)	7.2 dBi	Fixed	2 m	9.3 dB
A24-M7NF	Omni-directional (Mag-mount base station)	7.2 dBi	Fixed	2 m	9.3 dB
A24-P8SF	Flat Panel	8.5 dBi	Fixed	2 m	8.6 dB
A24-P8NF	Flat Panel	8.5 dBi	Fixed	2 m	8.6 dB
A24-P13NF	Flat Panel	13.0 dBi	Fixed	2 m	13.1 dB
A24-P14NF	Flat Panel	14.0 dBi	Fixed	2 m	14.1 dB
A24-P15NF	Flat Panel	15.0 dBi	Fixed	2 m	15.1 dB
A24-P16NF	Flat Panel	16.0 dBi	Fixed	2 m	16.1 dB
A24-P19NF	Flat Panel	19.0 dBi	Fixed	2 m	19.1 dB

*** If using the RF module in a portable application** (For example - If the module is used in a handheld device and the antenna is less than 20cm from the human body when the device is operation): The integrator is responsible for passing additional SAR (Specific Absorption Rate) testing based on FCC rules 2.1091 and FCC Guidelines for Human Exposure to Radio Frequency Electromagnetic Fields, OET Bulletin and Supplement C. The testing results will be submitted to the FCC for approval prior to selling the integrated unit. The required SAR testing measures emissions from the module and how they affect the person.

RF Exposure

⚠ WARNING: To satisfy FCC RF exposure requirements for mobile transmitting devices, a separation distance of 20 cm or more should be maintained between the antenna of this device and persons during device operation. To ensure compliance, operations at closer than this distance is not recommended. The antenna used for this transmitter must not be co-located in conjunction with any other antenna or transmitter.

The preceding statement must be included as a CAUTION statement in OEM product manuals in order to alert users of FCC RF Exposure compliance.

Europe (ETSI)

The XBee/XBee-PRO RF Module has been certified for use in several European countries. For a complete list, refer to www.maxstream.net.

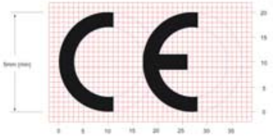
If the XBee/XBee-PRO RF Modules are incorporated into a product, the manufacturer must ensure compliance of the final product to the European harmonized EMC and low-voltage/safety standards. A Declaration of Conformity must be issued for each of these standards and kept on file as described in Annex II of the R&TTE Directive.

Furthermore, the manufacturer must maintain a copy of the XBee/XBee-PRO user manual documentation and ensure the final product does not exceed the specified power ratings, antenna specifications, and/or installation requirements as specified in the user manual. If any of these specifications are exceeded in the final product, a submission must be made to a notified body for compliance testing to all required standards.

OEM Labeling Requirements

The 'CE' marking must be affixed to a visible location on the OEM product.

Figure A-02. CE Labeling Requirements



The CE mark shall consist of the initials "CE" taking the following form:

- If the CE marking is reduced or enlarged, the proportions given in the above graduated drawing must be respected.
- The CE marking must have a height of at least 5mm except where this is not possible on account of the nature of the apparatus.
- The CE marking must be affixed visibly, legibly, and indelibly.

Restrictions

Power Output: The power output of the XBee-PRO RF Modules must not exceed 10 dBm. The power level is set using the PL command and the PL parameter must equal “0” (10 dBm).

France: France imposes restrictions on the 2.4 GHz band. Go to www.art-telecom.Fr or contact MaxStream for more information.

Norway: Norway prohibits operation near Ny-Alesund in Svalbard. More information can be found at the Norway Posts and Telecommunications site (www.npt.no).

Declarations of Conformity

MaxStream has issued Declarations of Conformity for the XBee/XBee-PRO RF Modules concerning emissions, EMC and safety. Files are located in the 'documentation' folder of the MaxStream CD.

Important Note

MaxStream does not list the entire set of standards that must be met for each country. MaxStream customers assume full responsibility for learning and meeting the required guidelines for each country in their distribution market. For more information relating to European compliance of an OEM product incorporating the XBee/XBee-PRO RF Module, contact MaxStream, or refer to the following web sites:

CEPT ERC 70-03E - Technical Requirements, European restrictions and general requirements: Available at www.ero.dk/.

R&TTE Directive - Equipment requirements, placement on market: Available at www.ero.dk/.

Approved Antennas

When integrating high-gain antennas, European regulations stipulate EIRP power maximums. Use the following guidelines to determine which antennas to design into an application.

XBee OEM RF Module

The following antenna types have been tested and approved for use with the XBee Module:

Antenna Type: Yagi
RF module was tested and approved with 15 dBi antenna gain with 1 dB cable-loss (EIRP Maximum of 14 dBm). Any Yagi type antenna with 14 dBi gain or less can be used with no cable-loss.

Antenna Type: Omni-directional
RF module was tested and approved with 15 dBi antenna gain with 1 dB cable-loss (EIRP Maximum of 14 dBm). Any Omni-directional type antenna with 14 dBi gain or less can be used with no cable-loss.

Antenna Type: Flat Panel
RF module was tested and approved with 19 dBi antenna gain with 4.8 dB cable-loss (EIRP Maximum of 14.2 dBm). Any Flat Panel type antenna with 14.2 dBi gain or less can be used with no cable-loss.

XBee-PRO OEM RF Module (@ 10 dBm Transmit Power, PL parameter value must equal 0)

The following antennas have been tested and approved for use with the embedded XBee-PRO RF Module:

- Dipole (2.1 dBi, Omni-directional, Articulated RPSMA, MaxStream part number A24-HABSM)
- Chip Antenna (-1.5 dBi)
- Attached Monopole Whip (1.5 dBi)

The RF modem encasement was designed to accommodate the RPSMA antenna option.

Canada (IC)

Labeling Requirements

Labeling requirements for Industry Canada are similar to those of the FCC. A clearly visible label on the outside of the final product enclosure must display the following text:

Contains Model XBee Radio, IC: 4214A-XBEE
Contains Model XBee-PRO Radio, IC: 4214A-XBEEPRO

The integrator is responsible for its product to comply with IC ICES-003 & FCC Part 15, Sub. B - Unintentional Radiators. ICES-003 is the same as FCC Part 15 Sub. B and Industry Canada accepts FCC test report or CISPR 22 test report for compliance with ICES-003.

Japan

In order to gain approval for use in Japan, the XBee-PRO RF Module must contain firmware that limits its transmit power output to 10 dBm.

For a list of module part numbers approved for use in Japan, contact MaxStream [call 1-801-765-9885 or send e-mail to sales@maxstream.net].

Labeling Requirements

A clearly visible label on the outside of the final product enclosure must display the following text:

ID: 005NYCA0378

Appendix B: Development Guide

Development Kit Contents

The XBee Professional Development Kit includes the hardware and software needed to rapidly create long range wireless data links between devices (XBee and XBee-PRO Starter Kits, that contain fewer modules and accessories, are also available).

Table B-01. Items Included in the Development Kit (Professional)

Item	Qty.	Description	Part #
XBee-PRO Module	2	(1) OEM RF Module w/ U.FL antenna connector (1) OEM RF Module w/ attached wire antenna	XBP24-AUI-001 XBP24-AWI-001
XBee Module	3	(1) OEM RF Module w/ U.FL antenna connector (1) OEM RF Module w/ attached whip antenna (1) OEM RF Module w/ chip antenna	XB24-AUI-001 XB24-AWI-001 XB24-ACI-001
RS-232 Development Board	4	Board for interfacing between modules and RS-232 devices (Converts signal levels, displays diagnostic info, & more)	XBIB-R
USB Development Board	1	Board for interfacing between modules & USB devices (Converts signal levels, displays diagnostic info, & more)	XBIB-U
RS-232 Cable (6', straight-through)	1	Cable for connecting RS-232 interface board with DTE devices (devices that have a male serial DB-9 port - such as most PCs)	JD2D3-CDS-6F
USB Cable (6')	1	Cable for connecting USB interface board to USB devices	JU1U2-CSB-6F
Serial Loopback Adapter	1	[Red] Adapter for configuring the module assembly (module + RS-232 interface board) to function as a repeater for range testing	JD2D3-CDL-A
NULL Modem Adapter (male-to-male)	1	[Black] Adapter for connecting the module assembly (module + RS-232 interface board) to other DCE (female DB-9) devices	JD2D2-CDN-A
NULL Modem Adapter (female-to-female)	1	[Gray] Adapter for connecting serial devices. It allows users to bypass the radios to verify serial cabling is functioning properly.	JD3D3-CDN-A
Power Adapter (9VDC, 1 A)	1	Adapter for powering the RS-232 development board	JP5P2-9V11-6F
Battery Clip (9V)	1	Clip for remotely powering the RS-232 board w/ a 9V battery	JP2P3-C2C-4I
RPSMA Antenna	2	RPSMA half-wave dipole antenna (2.4 GHz, 2.1 dB)	A24-HASM-450
RF Cable Assembly	2	Adapter for connecting RPSMA antenna to U.FL connector	JF1R6-CR3-4I
CD	1	Documentation and Software	MD0030
Quick Start Guide	1	Step-by-step instruction on how to create wireless links & test range capabilities of the modules	MD0026

Interfacing Options

The development kit includes an RS-232 and a USB interface board. Both boards provide a direct connection to many serial devices and therefore provide access to the RF module registries. Parameters stored in the registry allow OEMs and integrators to customize the modules to suite the needs of their data radio systems.

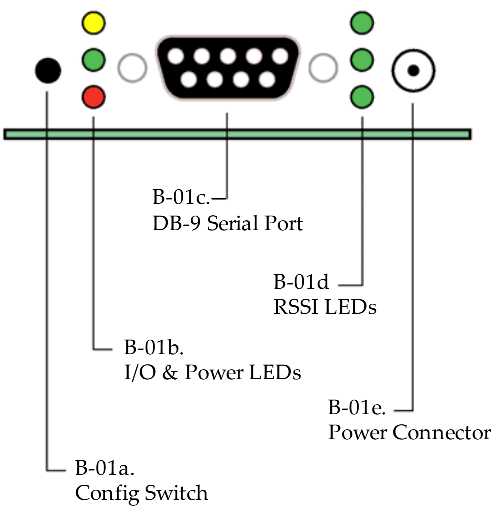
The following sections illustrate how to use the interface boards for development purposes. The MaxStream Interface board provides means for connecting the module to any node that has an available RS-232 or USB connector. Since the module requires signals to enter at TTL voltages, one of the main functions of the interface board is to convert signals between TTL levels and RS-232 and USB levels.

Note: In the following sections, an OEM RF Module mounted to an interface board will be referred to as a "Module Assembly".

RS-232 Development Board

External Interface

Figure B-01. Front View



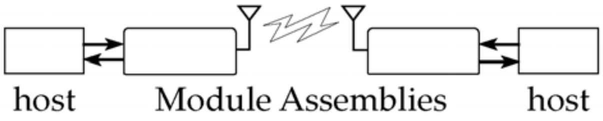
B-01a. Reset Switch

The Reset Switch is used to reset (re-boot) the RF module. This switch only applies when using the configuration tabs of MaxStream's X-CTU Software.

B-01b. I/O & Power LEDs

LEDs indicate RF module activity as follows:

Yellow (top LED)	=	Serial Data Out (to host)
Green (middle)	=	Serial Data In (from host)
Red (bottom)	=	Power/Association Indicator (Refer to the D5 (DIO5 Configuration) parameter)



B-01c. Serial Port

Standard female DB-9 (RS-232) connector.

B-01d. RSSI LEDs

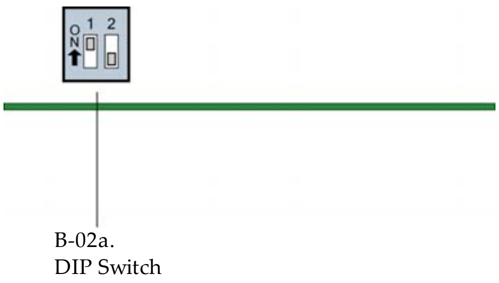
RSSI LEDs indicate the amount of fade margin present in an active wireless link. Fade margin is defined as the difference between the incoming signal strength and the module's receiver sensitivity.

3 LEDs ON	=	Very Strong Signal (> 30 dB fade margin)
2 LEDs ON	=	Strong Signal (> 20 dB fade margin)
1 LED ON	=	Moderate Signal (> 10 dB fade margin)
0 LED ON	=	Weak Signal (< 10 dB fade margin)

B-01e. Power Connector

5-14 VDC power connector

Figure B-02. Back View



B-02a. DIP Switch

DIP Switch functions are not supported in this release. Future downloadable firmware versions will support DIP Switch configurations.

RS-232 Pin Signals

Figure B-03. Pins used on the female RS-232 (DB-9) Serial Connector

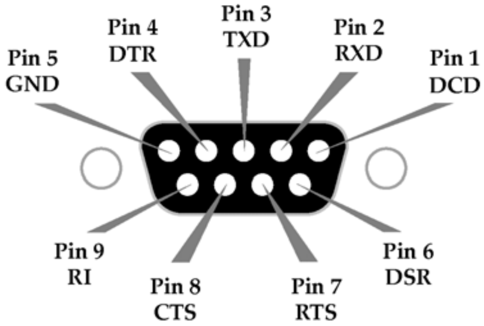


Table B-02. Pin Assignments and Implementations

DB-9 Pin	RS-232 Name	Description	Implementation
1	DCD	Data-Carrier-Detect	Connected to DSR (pin6)
2	RXD	Receive Data	Serial data exiting the module assembly (to host)
3	TXD	Transmit Data	Serial data entering into the module assembly (from host)
4	DTR	Data-Terminal-Ready	Can enable Power-down on the module assembly
5	GND	Ground Signal	Ground
6	DSR	Data-Set-Ready	Connected to DCD (pin1)
7	$\overline{\text{RTS}}$ / CMD	Request-to-Send / Command Mode	Enables $\overline{\text{RTS}}$ flow control or Command Mode
8	$\overline{\text{CTS}}$	Clear-to-Send	Provides $\overline{\text{CTS}}$ flow control
9	RI	Ring Indicator	Optional power input that is connected internally to the positive lead of the front power connector

* Functions listed in the implementation column may not be available at the time of release.

Wiring Diagrams

Figure B-04. DTE Device (RS-232, male DB-9 connector) wired to a DCE Module Assembly (female DB-9)

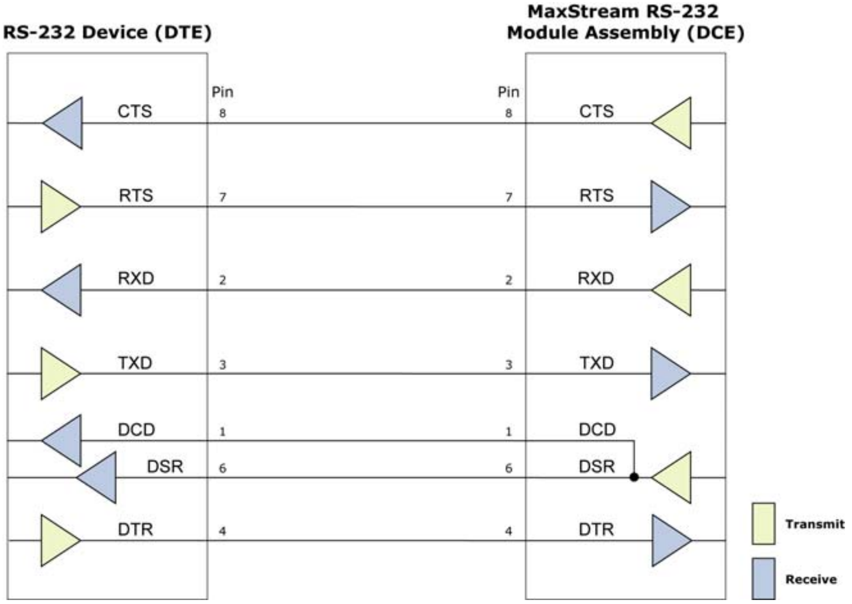
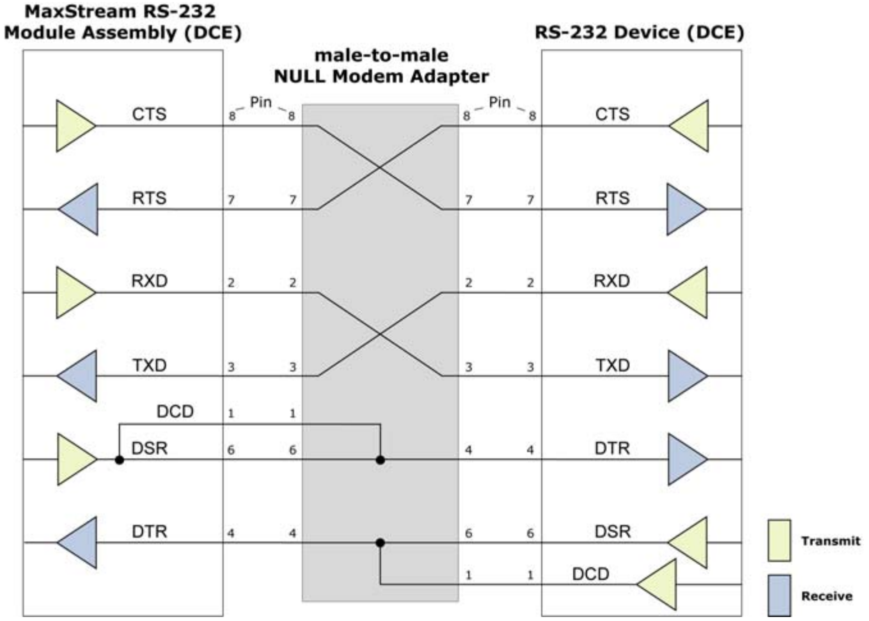
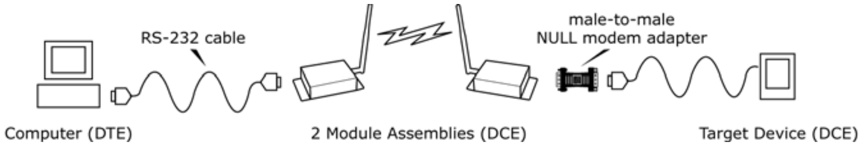


Figure B-05. DCE Module Assembly (female DB-9 connector) wired to a DCE Device (RS-232, male DB-9)



Sample Wireless Connection: DTE <--> DCE <--> DCE <--> DCE

Figure B-06. Typical wireless link between DTE and DCE devices



Adapters

The development kit includes several adapters that support the following functions:

- Performing Range Tests
- Testing Cables
- Connecting to other RS-232 DCE and DTE devices
- Connecting to terminal blocks or RJ-45 (for RS-485/422 devices)

NULL Modem Adapter (male-to-male)

Part Number: JD2D2-CDN-A (Black, DB-9 M-M) The male-to-male NULL modem adapter is used to connect two DCE devices. A DCE device connects with a straight-through cable to the male serial port of a computer (DTE).

Figure B-07. Male NULL modem adapter and pinouts

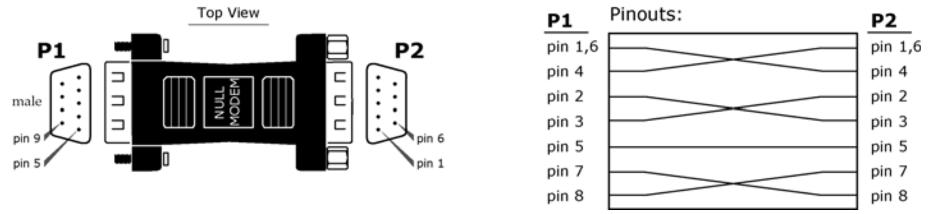
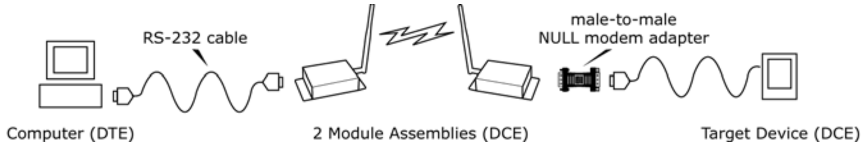


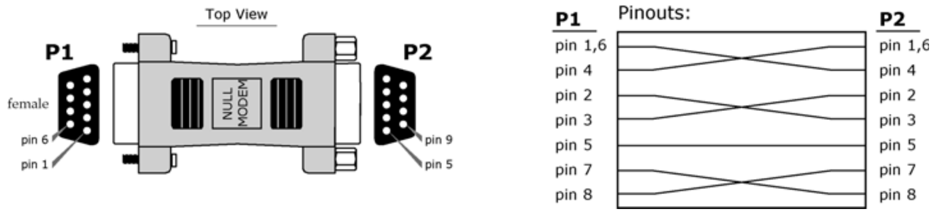
Figure B-08. Example of a MaxStream Radio Modem (DCE Device) connecting to another DCE device)



NULL Modem Adapter (female-to-female)

Part Number: JD3D3-CDN-A (Gray, DB-9 F-F) The female-to-female NULL modem adapter is used to verify serial cabling is functioning properly. To test cables, insert the female-to-female NULL modem adapter in place of a pair of module assemblies (RS-232 interface board + XTend Module) and test the connection without the modules in the connection.

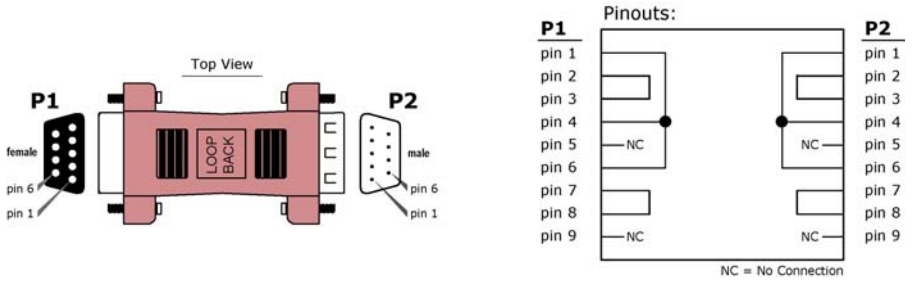
Figure B-09. Female NULL modem adapter and pinouts



Serial Loopback Adapter

Part Number: JD2D3-CDL-A (Red, DB-9 M-F) The serial loopback adapter is used for range testing. During a range test, the serial loopback adapter configures the module to function as a repeater by looping serial data back into the radio for retransmission.

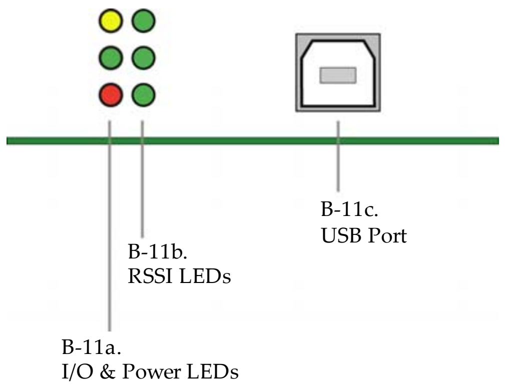
Figure B-10. Serial loopback adapter and pinouts



USB Development Board

External Interface

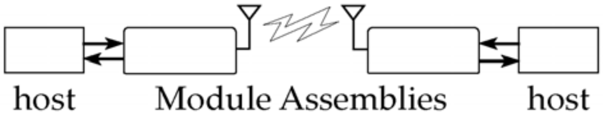
Figure B-11. Front View



B-11a. I/O & Power LEDs

LEDs indicate RF module activity as follows:

Yellow (top LED) = Serial Data Out (to host)
Green (middle) = Serial Data In (from host)
Red (bottom) = Power/Association Indicator (Refer to the D5 (DIO5 Configuration) parameter)



B-11b. RSSI LEDs

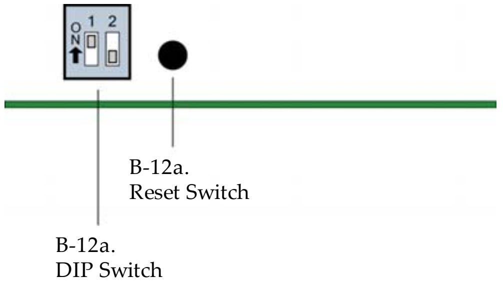
RSSI LEDs indicate the amount of fade margin present in an active wireless link. Fade margin is defined as the difference between the incoming signal strength and the module's receiver sensitivity.

3 LEDs ON	=	Very Strong Signal (> 30 dB fade margin)
2 LEDs ON	=	Strong Signal (> 20 dB fade margin)
1 LED ON	=	Moderate Signal (> 10 dB fade margin)
0 LED ON	=	Weak Signal (< 10 dB fade margin)

B-11c. USB Port

Standard Type-B OEM connector is used to communicate with OEM host and power the RF module.

Figure B-12. Back View



B-12a. DIP Switch

DIP Switch functions are not supported in this release. Future downloadable firmware versions will support the DIP Switch configurations.

B-12b. Reset Switch

The Reset Switch is used to reset (re-boot) the RF module.

USB Pin Signals

Table B-03. USB signals and their implantations on the XBee/XBee-PRO RF Module

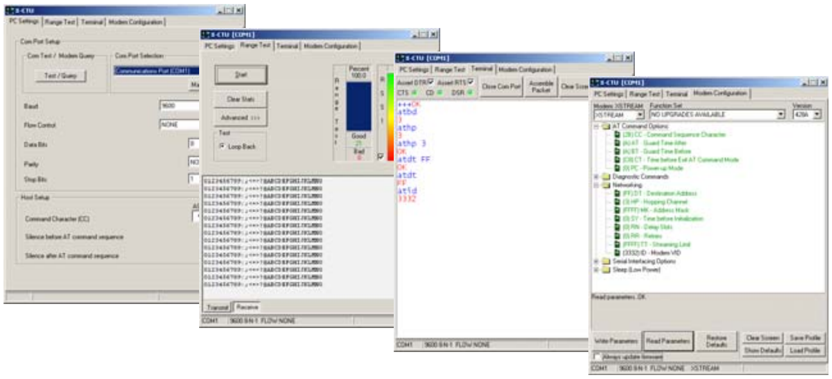
Pin	Name	Description	Implementation
1	VBUS	Power	Power the RF module
2	D-	Transmitted & Received Data	Transmit data to and from the RF module
3	D+	Transmitted & Received Data	Transmit data to and from the RF module
4	GND	Ground Signal	Ground

X-CTU Software

X-CTU is a MaxStream-provided software program used to interface with and configure MaxStream RF Modules. The software application is organized into the following four tabs:

- PC Settings tab - Setup PC serial ports for interfacing with an RF module
- Range Test tab - Test the RF module's range and monitor packets sent and received
- Terminal tab - Set and read RF module parameters using AT Commands
- Modem Configuration tab - Set and read RF module parameters

Figure B-13. X-CTU User Interface (PC Settings, Range Test, Terminal and Modem Configuration tabs)



NOTE: PC Setting values are visible at the bottom of the Range Test, Terminal and Modem Configuration tabs. A shortcut for editing PC Setting values is available by clicking on any of the values.

Installation

Double-click the "setup_X-CTU.exe" file and follow prompts of the installation screens. This file is located in the 'software' folder of the MaxStream CD and also under the 'Downloads' section of the following web page: www.maxstream.net/support/downloads.php

Setup

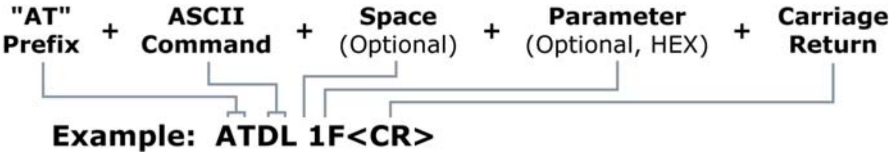
To use the X-CTU software, a module assembly (An RF module mounted to an interface Board) must be connected to a serial port of a PC.

NOTE: Failure to enter AT Command Mode is most commonly due to baud rate mismatch. The interface data rate and parity settings of the serial port ("PC Settings" tab) must match those of the module (BD (Baud Rate) and NB (Parity) parameters respectively).

Serial Communications Software

A terminal program is built into the X-CTU Software. Other terminal programs such as "HyperTerminal" can also be used to configure modules and monitor communications. When issuing AT Commands through a terminal program interface, use the following syntax:

Figure B-14. Syntax for sending AT Commands



NOTE: To read a parameter value stored in a register, leave the parameter field blank.

The example above issues the DL (Destination Address Low) command to change destination address of the module to "0x1F". To save the new value to the module’s non-volatile memory, issue WR (Write) command after modifying parameters.

Appendix C: Additional Information

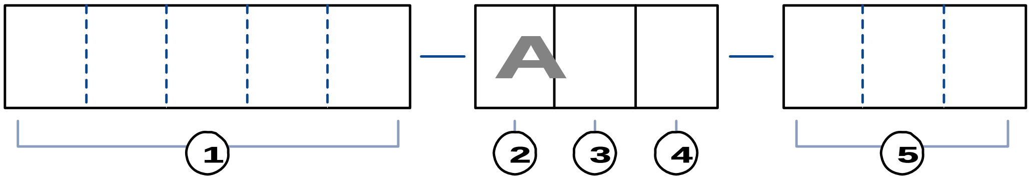
1-Year Warranty

XBee/XBee-PRO RF Modules from MaxStream, Inc. (the "Product") are warranted against defects in materials and workmanship under normal use, for a period of 1-year from the date of purchase. In the event of a product failure due to materials or workmanship, MaxStream will repair or replace the defective product. For warranty service, return the defective product to MaxStream, shipping prepaid, for prompt repair or replacement.

The foregoing sets forth the full extent of MaxStream's warranties regarding the Product. Repair or replacement at MaxStream's option is the exclusive remedy. THIS WARRANTY IS GIVEN IN LIEU OF ALL OTHER WARRANTIES, EXPRESS OR IMPLIED, AND MAXSTREAM SPECIFICALLY DISCLAIMS ALL WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT SHALL MAXSTREAM, ITS SUPPLIERS OR LICENSORS BE LIABLE FOR DAMAGES IN EXCESS OF THE PURCHASE PRICE OF THE PRODUCT, FOR ANY LOSS OF USE, LOSS OF TIME, INCONVENIENCE, COMMERCIAL LOSS, LOST PROFITS OR SAVINGS, OR OTHER INCIDENTAL, SPECIAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PRODUCT, TO THE FULL EXTENT SUCH MAY BE DISCLAIMED BY LAW. SOME STATES DO NOT ALLOW THE EXCLUSION OR LIMITATION OF INCIDENTAL OR CONSEQUENTIAL DAMAGES. THEREFORE, THE FOREGOING EXCLUSIONS MAY NOT APPLY IN ALL CASES. This warranty provides specific legal rights. Other rights which vary from state to state may also apply.

Ordering Information

Figure C-01. Divisions of the XBee/XBee-PRO RF Module Part Numbers



1 **MaxStream Product Family**
XB24 = XBee 2.4GHz
XBP24 = XBee-PRO 2.4GHz

2 **Reserved for internal use**
Insert the letter 'A'

3 **Antenna Option**
C = Chip Antenna
U = U.FL RF Connector
W = Integrated Whip Antenna

4 **Rating**
I = Industrial (-40 to 85° C)

5 **Protocol**
001 = 802.15.4
002 = ZigBee

For example:

XBP24-AWI-001 = XBee-PRO OEM RF Module, 2.4 GHz, attached whip antenna, Industrial temperature rating, IEEE 802.15.4 standard

If operating in Japan, XBee-PRO RF Modules must contain firmware that limits transmit power output to 10 dBm. For a list of module part numbers approved for use in Japan, contact MaxStream [call 1-801-765-9885 or send e-mail to sales@maxstream.net].

Contact MaxStream

Free and unlimited technical support is included with every MaxStream Radio Modem sold. For the best in wireless data solutions and support, please use the following resources:

Documentation:	www.maxstream.net/support/downloads.php	
Technical Support:	Phone.	(866) 765-9885 toll-free U.S.A. & Canada (801) 765-9885 Worldwide
	Live Chat.	www.maxstream.net
	E-Mail.	rf-xperts@maxstream.net

MaxStream office hours are 8:00 am - 5:00 pm [U.S. Mountain Standard Time]