UNIVERSITY OF SOUTHERN
QUEENSLAND
Faculty of Engineering and Surveying

In-Vehicle Display System

A dissertation submitted by

Michael Costa

in fulfilment of the requirements of

ENG4112 – Research Project

towards the degree of

Bachelor of Engineering (Electrical & Electronic)

Submitted: October 2005

Abstract

The design of an in-vehicle textual display system based on a matrix of light emitting diodes was achieved within this project. The concept of an illuminated display was researched, designed and developed into a working prototype.

The prototype of the display consists of eleven display sections, each consisting of a 5x7 LED matrix. The design allows these modular sections to be added or removed to vary the length of the display required.

The research into existing LED displays revealed a number of equally suited products was available, however, the goal of the project was the creation of an original design that could be manufactured into a new product in the future.

The display is controlled by a PIC microcontroller that outputs serial data to a series of cascaded serial-in parallel-out shift registers. The shift registers convert the stream of serial data to a parallel sequence used to scan each line of the display.

A handheld programmer is incorporated in the design and is used to create messages for the display. The programmer is based around a mobile phone layout using an alphanumeric keypad and multi-tap typing to enter each character.

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.

Prof G Baker
Dean
Faculty of Engineering and Surveying

Certification

I certify that the ideas, designs and experimental work, results, analyses 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.

Michael Costa

Student Number: 0011221440

____

Signature

____

Date

Acknowledgments

I would like to acknowledge the ideas and assistance provided by my fellow students and friends at USQ throughout this project. For their support with technical knowledge and assistance with the dissertation I'm very grateful.

I would to thank my supervisor, Mr Mark Phythian, for his guidance and advice, as well as Ross Leamon for proposing and supporting the project.

Thank you also to my family, for their continuing support throughout my degree.

Table of Contents

List of Figures

List of Tables

Glossary of Terms

ASCII – American Standard Code for Information Interchange: A character set and form of character encoding which represents text in computers and other communications equipment.

BCD – Binary Coded Decimal: A common form of encoding for decimal digits in computing and electronics equipment. Each digit is represented by four binary bits which have values from left to right of 8, 4, 2 and 1. The sum of these four bits equals the equivalent decimal value which ranges from 0-9.

CMOS - Complementary Metal Oxide Semiconductor: A widely used type of integrated circuit design which uses significantly less power that other forms of logic.

CPU – Central Processing Unit: The circuitry of a computer that executes calculations, also known as a processor. Two typical components of a CPU are the arithmetic logic unit (ALU), which performs arithmetic and logical operations, and the control unit, which extracts instructions from memory, decodes and executes them.

DIL – Dual-In-Line (package): A standard of electronic component layout where pins on the device are aligned in two parallel lines.

EMI – Electro-Magnetic Interference: Radiated or conducted energy that adversely affects circuit performance disrupting a device’s normal operation.

GaAsP- Gallium Arsenide Phosphide: A compound used in the production of red, orange and yellow light emitting diodes.

IC – Integrated Circuit: A set of electronic components and their interconnections that are etched or imprinted on a semiconductor substrate.

IDE – Integrated Development Environment: A set of programs run from a single user interface. Programming languages often include a text editor, compiler and debugger, which are all activated and function from a common menu.

I/O – Input and Output: Conductors used to transfer data between a central and a peripheral device.

IP – Ingress Protection: A classification of the level of protection that electrical appliances provide against the intrusion of solid objects, dust, accidental contact, and water.

LED – Light Emitting Diode: A semiconductor device that emits narrow-spectrum light when electrically biased in the forward direction.

PC - Personal Computers: A microcomputer intended to be used by one person at a time, and suitable for general purpose tasks such as word processing, programming, or game play.

PIC – Programmable Integrated Circuit: A family of microcontrollers made by Microchip Technology.

RISC – Reduced Instruction Set Computing: A microprocessor CPU design philosophy that favours a smaller and simpler set of instructions to streamline processing.

QWERTY – the most common modern-day layout of letters on most English language computer and typewriter keyboards. The name is taken from the first six letters shown on the keyboard's top row of letters.

USART – Universal Synchronous Asynchronous Receiver-Transmitter: A piece of computer hardware that translates between parallel bits of data and serial bits, in both synchronous and asynchronous modes.

Chapter 1 Introduction

1.1 Project Aim

The aim of this research project is the development of an in-vehicle textual display system based on a light emitting diode (LED) dot matrix. The project involves taking the device from an initial concept through a design phase, to constructing a prototype of the product. The system consists of the central display unit accompanied by a handheld interface to program messages into the display. The request for the design originated from Downey Engineering, a Toowoomba based electronics and control business that has sponsored the project. They are interested in creating the design as they have identified an opportunity in the market for such a product and may produce the device in the future.

The display system is to be designed initially for use in an automotive vehicle. It is intended that the unit be mounted on the rear parcel shelf of a vehicle, to convey information to motorists through the rear window. Applications for the vehicle mounted display include use in the event of a vehicle breakdown or for advertising. The design however remains universal in nature, allowing implementation in numerous other applications. The self-contained nature of the intended design will allow the display to be mounted almost anywhere it is needed and may provide information to passers-by or advertising for businesses.

1.2 Objectives

The main objectives accomplished in this project include:

1. Research information relating to the design of LED moving message displays and microcontroller based circuits.
2. Design a microcontroller driven, marquee type display to be mounted in a vehicle.
3. Design a handheld interface to the display system for calling up and programming messages.
4. Design software to provide pre-programmed messages, animations and user defined messages for the display system.
5. Develop a full set of working specifications of the display system to suit the technical / budget limitations of the project.
6. Construct a prototype of the display system mentioned in 2 and 3 above and evaluate its performance.

Research into existing LED display systems and microcontroller-based circuits was aimed at identifying features available in existing products that may be included in this design. The design parameters of the display and handheld programmer will adhere to the functional specifications that follow. Any deviations from the specification involved consultation with Downey Engineering. The set of working documents created for the design, forms the initial specification of this project, which includes documenting the development of the design and evaluating its effectiveness.

1.3 Display Systems

The project is derived from a proposal made by Downey Engineering to fund the design of an illuminated display; an initial step was the evaluation of the requirements that needed to be met. This was achieved through several consultations with Ross Leamon, the manager of engineering at Downey Engineering. In these meetings the technical features, physical operation and possible applications for the design were addressed. These discussions resulted in a functional specification that outlined the purpose and features that the design would ultimately fulfil. This document could then be used as a benchmark, against which created designs could be assessed for their merit.

The next objective in developing a design was to investigate existing products available. This was primarily done to assess the cost of available products as well as collect ideas for other features that could be incorporated in the design. Many examples where found that fulfilled the same criteria as the expected design. Numerous ideas for additional features that could be included in the display and programmer were found during the research, however due to the fact that the displays were commercially available products, any information on the inner workings of the devices was not freely available. The results of the research was an increase in features that could be incorporated in the project but a lack of the physical techniques used in creating moving message displays.

1.4 Development of the Functional Specifications

The starting point for the design aspect of the project was a flexible functional specification drafted by Downey Engineering. This document defined the need for an in-vehicle/industrial style of display that is both robust and economical. The technical features, physical operation and possible applications for the display were also discussed, however any limits placed on the design were negotiable. The only distinct specifications were:

- Voltage Supply Range: 10-30 volts DC
- Operating Temperature: 0-50 degrees Celsius
- Display Visibility: 20-30 metres
- Message Capacity: 90 messages
- Text Effects: Static, Scrolling, Flashing
- Accompanying handheld message programmer

A voltage supply range of 10-30 volts was chosen so the display was a versatile product that could operate in a number of applications. The 12V supply system of automotive vehicles is the primary intended voltage for the display, but systems that use 24 volts are not uncommon and were therefore incorporated into the specification. The operating temperature of 0-50 degrees Celsius was included as a requirement as these temperatures are considered to be the extreme temperatures that may be reached inside an average vehicle. As most integrated circuits and electronic parts that could be used in the design are rated for temperatures ranging well outside these limits, this requirement poses no great limit on the design.

The visibility of the display was an important aspect to consider when determining the display’s proposed characteristic. The screen needed to be large enough to be eye-catching and legible at a distance, yet sufficiently compact to be transportable and easily mounted. A display that was visible at 20-30 metres was thought to be a sufficient distance and yet still small enough to fit inside the rear window of a vehicle. As mentioned previously this specification was not set in stone and did vary within the design phase. The display size used in the initial prototyping is smaller than this specification for ease of handling. The text effects such as scrolling and flashing is implemented within the software of the display to enhance its ability to attract attention as well as displaying messages longer than the screen length. The inclusion of an accompanying handheld programmer for the display was decided upon to increase the flexibility of the system. The handheld programmer would be necessary to reprogram messages while the display was still in its operating location. This feature would allow messages to be modified easily without the need to either remove the display or carry a computer to the operating location for reprogramming.

1.5 Dissertation Overview

CHAPTER 2 introduces the background research and information relating to the design of the display, existing systems available and the design and performance requirements of the project.

In CHAPTER 3 the intended use and operation of the whole display is described. A description of the components used in the design and an explanation of how they are interfaced, is followed by a schematic of the display and handheld programmer.

CHAPTER 4 addresses the development of the prototypes of display and programmer focusing on the methods of construction.

CHAPTER 5 will detail the processes used for testing the display and handheld interface prototypes, while in construction and after completion.

CHAPTER 6 presents the software used in developing code for the display system as well as an explanation of additional the routines developed.

CHAPTER 7 discusses some of the broader issues that relate to the creation of the product including legal issues, aesthetics and electromagnetic compatibility.

CHAPTER 8 will conclude the dissertation by summarizing the achievements of the project, giving an analyses of the final design, comparing the performance with the specifications set out in Chapter 1, and providing an indication of any further work required.

Chapter 2 Background Research

2.1. Light Emitting Diode Technology

Illuminated textual displays based on light emitting diodes (LEDs) are quite a widely used medium, used to convey information in modern society. The use of light to display text and information is a well established technology, as it is both eye-catching during the day and easily visible at night. Neon lighting and incandescent bulbs have been utilized in this application for large advertising displays since the early 1920's. With the creation of semiconductor technology came the increased usage of LEDs in illuminated displays due to their high efficiency and long lifespan.

LEDs consist of a semiconductor material that has had impurity atoms introduced into it through a process called doping. The addition of different impurities changes the balance of electrons in the material, creating a N-type material that has extra free electrons or P-type material that has "holes" where electrons are scarce. Diodes consist of a section of N-type material bonded to a section of P-type material. When forward biased electrons move from the N-type region to the P-type region and holes move in the reverse direction. Electrons combine with holes of the P-type layer to release energy in the form of light. This process occurs in all diodes, but LEDs have been specifically designed to emit light in the visual or infrared range. The wavelength of the light emitted λ is given by:

$$\lambda \cong \frac{h\,c}{E_g} \tag{1.1}$$

where h is Planck's constant (6.626×10^{-34} J-s);

c is the velocity of light in vacuum; and

E_g is the bandgap energy for a particular semiconductor.

In regular diodes the semiconductor material absorbs most of the light energy created. LEDs however are specifically constructed to release a large number of photons outward from the semiconductor as shown in the following figure. The lens and the bottom reflecting surface increase the amount of light transmitted out of the top of the device.

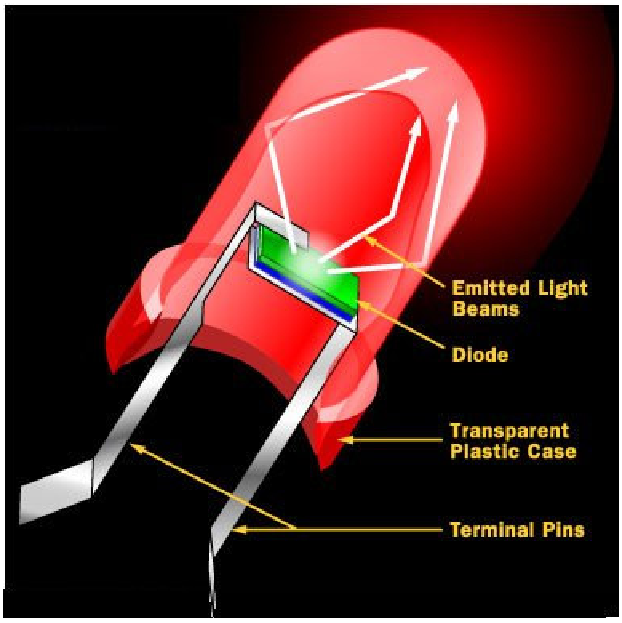
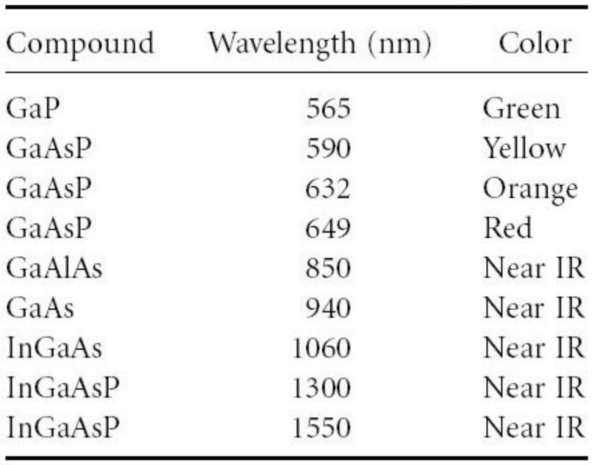


Figure 2.1 Light Emitting Diode

Source: How Stuff Works (2005)

LEDs of various wavelengths are commonly available and are obtained by combining ratios of semiconductor material that have different bandgap energies. By choosing different binary, ternary, and quaternary compositions of these semiconductors a light emitting diode of particular wavelength can be created as shown in the following table.

Table 2.1 Common LED Compounds and Wavelengths



Compound	Wavelength (nm)	Color
GaP	565	Green
GaAsP	590	Yellow
GaAsP	632	Orange
GaAsP	649	Red
GaAlAs	850	Near IR
GaAs	940	Near IR
InGaAs	1060	Near IR
InGaAsP	1300	Near IR
InGaAsP	1550	Near IR

2.1.1 Expansion of LED Usage

The use of a LEDs dot matrix for creating a text display system is quite common with its usage expanding greatly in recent years. Such displays can be found in airports, where they are used to display flight information, and in stock exchanges to display share prices. The wide usage of LED displays is a result of its ability to convey information to large audiences quickly and efficiently. As LED displays are often controlled by digital technology the information can swiftly and easily be updated. This feature of LED displays has led to a great flexibility of such products in countless applications. An additional benefit of using this form of display is that LEDs are a very efficient form of illumination. Unlike incandescent bulbs, LEDs do not generate a large amount of wasted energy in the form of heat.

2.1.2 Characteristics of LED Operation

The type of LEDs used for the project are the GaAsP high-efficiency red variety manufactured by Kingbright. High-efficiency red, GaAsP on a GaP substrate was the first non-low-current high-efficiency red LED. The term GaAsP represents the working chemistry of the LED, which is gallium arsenide phosphide, with an arsenic to phosphorus ratio around 40 to 60 on a gallium phosphide (GaP) substrate. The peak wavelength of the high-efficiency red GaAsP LED is usually around 630 nm. Typical drive current is 5 to 20 mA and maximum current is usually 30 mA. The typical voltage drop across the device at 20 mA is around 1.9 volts. (Klipstein 2004)

A primary goal for the display in this project is to achieve the maximum illumination from the light emitting diodes during operation. The LEDs of the display are used under pulsed conditions in order to increase the effective brightness of the illuminators. Pulsing higher currents than the continuous current rating through the LED achieves this maximum brightness. This is due to the fact that luminance is proportional to current in a LED. In order to achieve maximum brightness it is also necessary to drive the LED with a shorter on-time and a lower duty cycle for operation. If operated continuously, the increased current would cause the LED to over-heat, resulting in permanent damage to it in the process. In order to protect the LED from this damage the duty cycle of the pulses, determined as on-time divide by pulse period, should be kept below 10%. In addition to this, the on-time of the LED should be kept below 5 milliseconds. This process of reducing pulse length to offset the increased current is done in order to keep the average junction temperature of the LED below the rated value. If this temperature is exceeded for prolonged periods the lifetime of the LED can be reduced. In broad terms, the restrictions on a LED lifetime are primarily thermally induced. Given that the LEDs are strobed within the manufacturer's recommend guidelines for the specific type, then the normal lifetime of up to 100,000 hours of operation should be achieved. This lifetime relates to the operating temperature and the average dissipated current ratings. If low duty cycles are combined with short on-times, so that the junction temperature of the LED is kept close to ambient, then effective operating lifetimes can often be extended.

2.2 Display Scanning

A method commonly used to control illuminated displays is to turn the rows or columns of the display on and off in quick succession. Multiplexing, as it is termed, reduces the amount of input and output (I/O) lines required to control the individual elements of a large display. In multiplexing a common set of control lines is used to join each display to the control system. As a consequence the amount of conductors and ports required for controlling is significantly reduced when compared with connecting each display individually to the system. A sub-unit of the display, typically a row or column of LEDs, is connected to the common bus, which is controlled to output the required pattern for that sub-unit at each instant. The sub-units are then sequentially illuminated with the corresponding pattern. Repeatedly cycling through this process, called scanning is used to create the phenomena of visual persistence. An example of this process is shown in the following figures, where rapidly illuminating sections of the display causes an observer to see a single image.

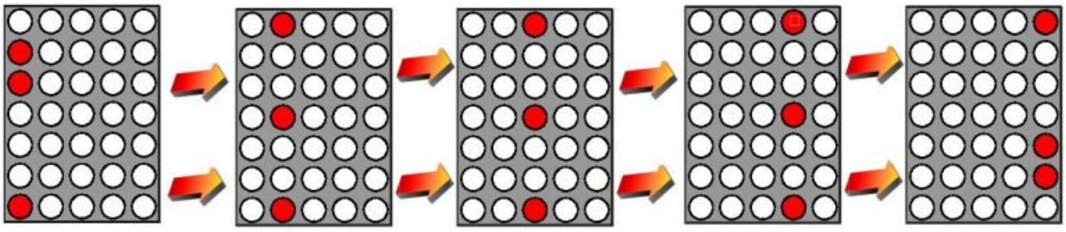


Figure 2.2 Rapid Scanning of the LED Display

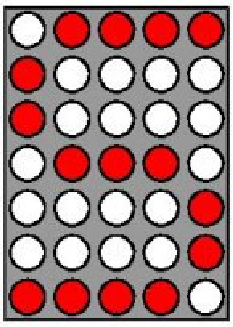


Figure 2.3 Observer’s Perception Caused by Visual Persistence

The theory of visual persistence is that the perceptual process of the brain and retina in the eye retain an image for a split second after viewing. David Cyganski further explains the physical process of visual persistence:

> *'Although an image on the retina decays gradually, rather than lasting a specific amount of time, there is a critical period during which the stimulus changes so little that the visual system cannot take in any new information even if the eyes are open. This period, on average, is about 50 milliseconds, or one-twentieth of a second. Thus, the average human visual system can only take in about 20 different images per second before they begin to blur together. If these images are sufficiently similar, then the blurring which takes place appears to the eye to resemble motion, in the same way we discern it when an object moves smoothly in the real world.'*(Cyganski, 1998).

The illumination of each sub-unit of a display takes place at a rate faster than the human eye and brain can register resulting in the appearance that all of the lines are operating at once. The rate at which the stimulus is flashed is an important factor in creating the illusion. The flicker fusion threshold is defined as the frequency at which all flicker of an intermittent light stimulus disappears. It is therefore important to have the frequency at which the individual elements of the display are cycled through higher than that of the flicker fusion threshold. If the higher rate is not then obtained the image presented to the viewer appears to flicker, which can be quite uncomfortable. A frame rate of less than 16 frames per second causes the mind to see these series of flashing images. The motion in the images can still be interpreted but the affects of the flashing become quite distracting.

Visual persistence can also be compared to the related phenomena of beta movement. In his 1912 study, Experimental Studies on the Seeing of Motion, Max Wertheimer described beta movement as a perceptual illusion whereby two or more still images are combined by the brain into surmised motion. The phenomena of beta movement can be utilised in illuminated displays to give the impression of movement of the image being displayed. Progressive frames of the image displayed will be laterally shifted along the screen, resulting in the illusion of movement to the observer.

2.3 Comparison of Market Products

There are numerous companies that produce textual displays similar to the design for this project. A few examples of different classes of display will be given to demonstrate the variety of products available. The simplest way to categorise LED displays is whether the information shown is purely alphanumeric, that is, consists of only characters and digits, or if the display is capable of presenting animation. Displays that are able to present animation are usually developed for large scale applications such as music concerts, sporting events or mass advertising. The common name for large scale displays of this type is “Jumbo Screens”. In a colour Jumbo Screen each pixel, the basic unit of the image, consists of a module containing at least one red, green and blue LED as shown below. These pixel modules are arranged in a rectangular grid to form the screen. The number and size of LEDs used to form one module varies with the dimensions of the full display as shown is Table 2.2.

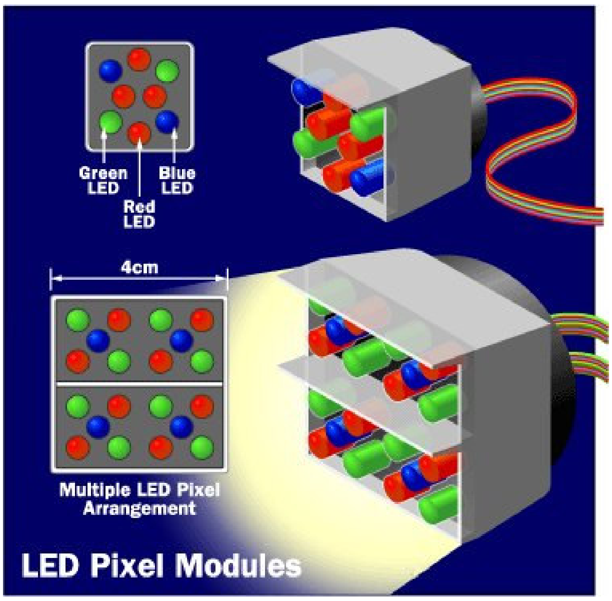


Figure 2.4 Jumbo Screen LED Modules

Source: How Stuff Works (2005)

Table 2.2 Jumbo Screen Dimensions

LED module size	**Screen size (metres)**
4 mm	2.56 x 1.92
25 mm	16 x 12
40 mm	25.6 x 19.2

In a Jumbo Screen a complicated computer system is required to control which of the LEDs are to be turn on, in addition to their levels of brightness. The computer samples the input signal, usually a television signal, for the intensity and colour information of the image. This is then translated into an intensity level for the three different LED colours that exist in each of the pixel modules. When these three colours of light combine, a matching representation of the pixel in the original input signal is produced. A typical 20-metre Jumbo Screen can consume up to 1.2 Watts per pixel, which is approximately 300,000 Watts for the full screen.

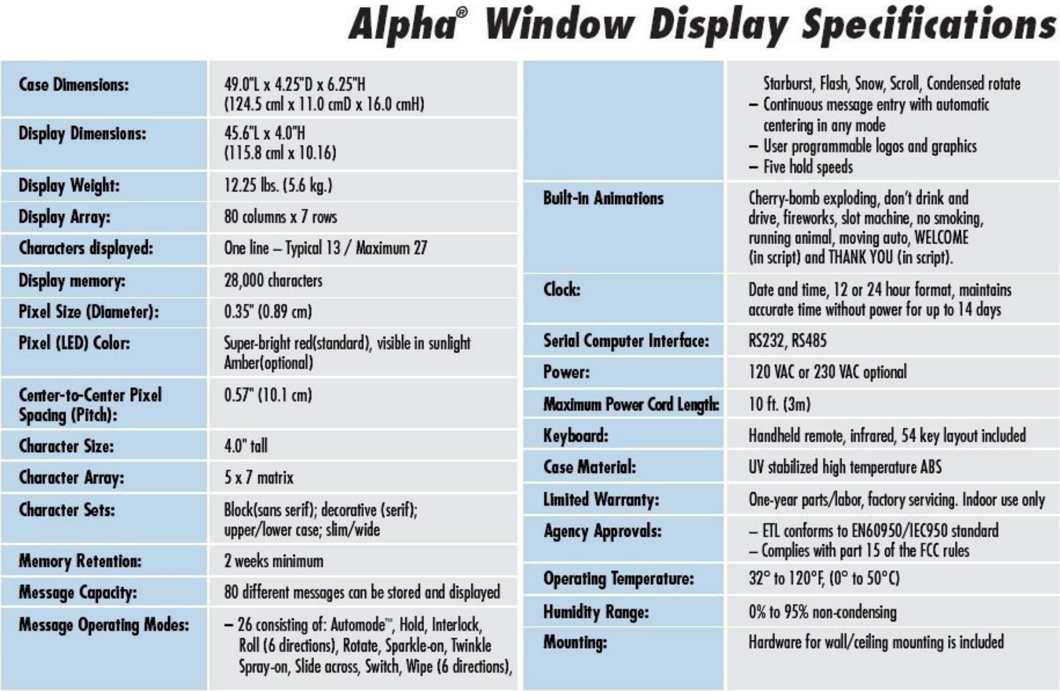
The use of a Jumbo Screen and other full animation displays would be an excessive design for this project as the screen sizes are much too large for the intended application and the features of such versatile displays would not be utilized enough to warrant such a complicated design. Consequently the focus is primarily on single line alphanumeric LED displays.

Alphanumeric type displays can be categorized by their ability to show different colours. Simple textual displays often utilise single colour, bi-colour and tri-colour illumination in their displays. These few colours are enough to represent simple text but the inclusion of other colours makes the display more eye-catching. Single colour LEDs are primarily used when the information shown is principally simple text where no emphasis or highlighting is required. Single colour LEDs were selected to be used in this project as incorporating bi-colour or tri-colour into the design added an additional level of complexity that was not needed for an initial prototype. Using the more complex multi-colour LEDs in future designs has been considered, however completing a fully functioning single colour display was the first step for this project.

The creation of the LED display for this project clearly is not pioneering a new technology or product line, as many examples of similar devices can be found in the market today. Two comparable units that are similar to the aim for this project are the GY2200 'Large Alphanumeric Display' produced by Vorne Visual Display & Productivity Tools and the 'Alpha Window Display' produced by Adaptive Micro Systems.

The Alpha display is the most comparable product available to the design created for this project. It offers a large 4-inch display typically of 13 characters. In addition to the standard features of a moving message display the Alpha contains a number of additional features that may be considered in future versions of the display. These features include, a clock to display the date and time in 12 and 24-hour format, an infrared link between the display and programmer and a variable character set to provide various fonts. An image of the Alpha display in use and a full list of its features are shown below.

Table 2.3 Alpha Window Display Specifications



Alpha® Window Display Specifications

Case Dimensions:	49.0"L x 4.25"D x 6.25"H (124.5 cml x 11.0 cmD x 16.0 cmH)
Display Dimensions:	45.6"L x 4.0"H (115.8 cml x 10.16)
Display Weight:	12.25 lbs. (5.6 kg.)
Display Array:	80 columns x 7 rows
Characters displayed:	One line – Typical 13 / Maximum 27
Display memory:	28,000 characters
Pixel Size (Diameter):	0.35" (0.89 cm)
Pixel (LED) Color:	Super-bright red(standard), visible in sunlight Amber(optional)
Center-to-Center Pixel Spacing (Pitch):	0.57" (10.1 cm)
Character Size:	4.0" tall
Character Array:	5 x 7 matrix
Character Sets:	Block(sans serif); decorative (serif); upper/lower case; slim/wide
Memory Retention:	2 weeks minimum
Message Capacity:	80 different messages can be stored and displayed
Message Operating Modes:	– 26 consisting of: Automode™, Hold, Interlock, Roll (6 directions), Rotate, Sparkle-on, Twinkle Spray-on, Slide across, Switch, Wipe (6 directions), Starburst, Flash, Snow, Scroll, Condensed rotate – Continuous message entry with automatic centering in any mode – User programmable logos and graphics – Five hold speeds
Built-in Animations	Cherry-bomb exploding, don't drink and drive, fireworks, slot machine, no smoking, running animal, moving auto, WELCOME (in script) and THANK YOU (in script).
Clock:	Date and time, 12 or 24 hour format, maintains accurate time without power for up to 14 days
Serial Computer Interface:	RS232, RS485
Power:	120 VAC or 230 VAC optional
Maximum Power Cord Length:	10 ft. (3m)
Keyboard:	Handheld remote, infrared, 54 key layout included
Case Material:	UV stabilized high temperature ABS
Limited Warranty:	One-year parts/labor, factory servicing. Indoor use only
Agency Approvals:	– ETL conforms to EN60950/IEC950 standard – Complies with part 15 of the FCC rules
Operating Temperature:	32° to 120°F, (0° to 50°C)
Humidity Range:	0% to 95% non-condensing
Mounting:	Hardware for wall/ceiling mounting is included

Source: Adaptive Micro Systems (2005)

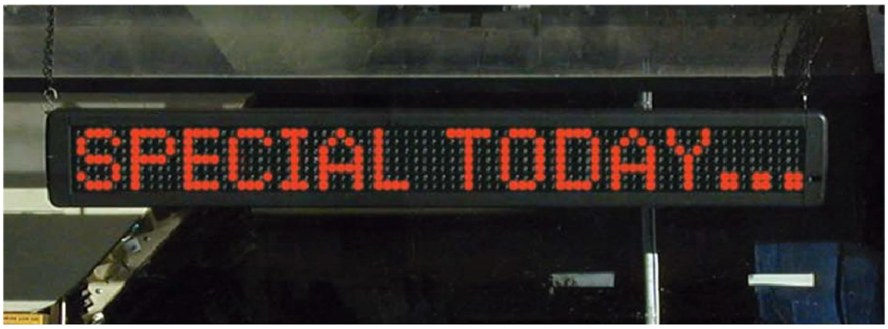


Figure 2.5 Alpha Window Display Produced by Adaptive Micro Systems

Source: Adaptive Micro Systems (2005)

The GY2200 alphanumeric display by Vorne, shown below, is also a comparative design although the display is to some extent smaller. Its features include:

- High-efficiency red LED display
- 2" font size with a full matrix display
- Smooth scrolling up to 60 characters
- Interface to almost any serial device
- Simple ASCII based protocol
- Five baud rates from 300 to 9600
- Viewing distances of 75' and beyond
- Full 128 ASCII character set
- Quick and easy DIP switch setup
- ASCII commands to scroll, flash

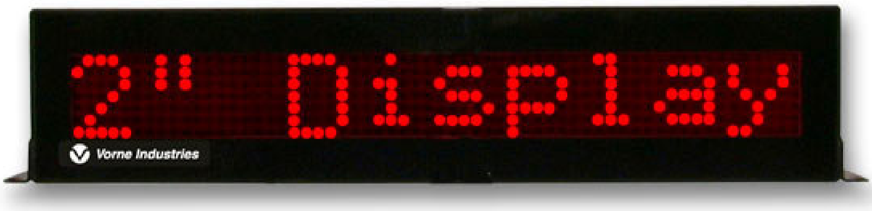


Figure 2.6 GY2200 Large Alphanumeric Display

Source: Vorne Visual Display & Productivity Tools (2005)

Features of both the GY2200 Large Alphanumeric Display and the Alpha Window Display have been used as a basis from which the design for this project's display has originated. Technical data regarding the internal circuitry of these devices was not available, however this research did provide useful knowledge of the existing displays available.

2.3 RS-232 Serial Communication

The RS-232 serial communication standard is used for data transfer between the handheld programmer and the display unit for this project. The RS-232 standard, stands for Recommended Standard number 232 of which the latest amendment is revision D. This format of communication was selected as it is widely supported by Personal Computers (PCs), is very simplistic in construction, and is compatible with the microcontroller used in both the display and programmer. Devices that use serial communications can be divided into two categories, Data Communications Equipment (DCE) and Data Terminal Equipment (DTE). The DTE is typically a PC and the DCE is the remote device which is communicated to over the serial connection. The full RS-232 standard specifies that a 25-pin "D" connector be used, however typical PC communications use a subset of the standard with a 9-pin "D" connector as shown below.

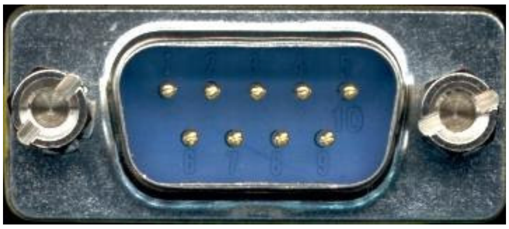


Figure 2.7 9-Pin D Connector

Source: Arcelect (2005)

The null modem configuration of the RS-232 standard is used in this project, which enables transmission between two DTEs. This configuration requires only 3 wires to be connected between the two devices. The transmit data line of one device is connected to the receive data line of the other device and vice versa. The signal ground lines of both devices are also joined to form a common ground voltage. In the null modem configuration the lines used for signal handshaking are not needed and are therefore looped back to each device. This technique, in effect, deceives both DTEs into believing that handshaking signals are functioning. For this to take place the data-terminal-ready line is looped back to the data-set-ready and the carrier-detect lines on both devices. When the data-terminal-ready line is made active, then the data-set-ready and the carrier detect immediately become active as they form the end of the loop back. A loop back is also made from request-to-send to clear-to-send on both devices. The entire configuration effectively bypass the handshaking protocols of the RS-232 standard to form a simplified communication link. The following figure shows the pin labelling of a 9-pin D connector and their usage in the RS-232 standard.

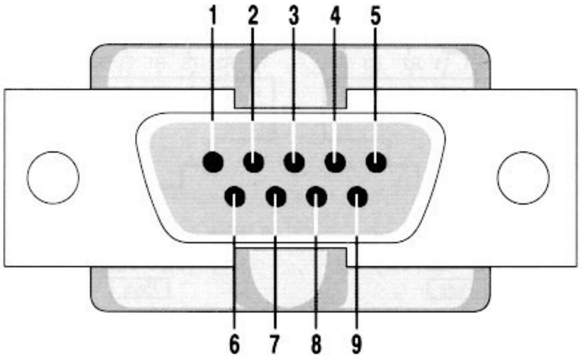


Figure 2.8 D Connector Pin Numbering

Source: Arcelect (2005)

Table 2.4 RS-232 Pin Description

D-Type-25 Pin No.	D-Type-9 Pin No.	Abbreviation	Full Name
Pin 2	Pin 3	TD	Transmit Data
Pin 3	Pin 2	RD	Receive Data
Pin 4	Pin 7	RTS	Request To Send
Pin 5	Pin 8	CTS	Clear To Send
Pin 6	Pin 6	DSR	Data Set Ready
Pin 7	Pin 5	SG	Signal Ground
Pin 8	Pin 1	CD	Carrier Detect
Pin 20	Pin 4	DTR	Data Terminal
Pin 22	Pin 9	RI	Ring Indicator

The electrical parameters of the logic level in the RS-232 standard state that:

- A logic 0 will be between +3 and +25 volts
- A logic 1 will be between –3 and –25 volts
- The region between +3 and –3 volts is undefined.

It is important to maintain the transmission voltage within the expected ranges so that the data received is decipherable.

Chapter 3 System Design

3.1 Description of Intended Use and Operation

The intended use of the display system developed in this project was initially specified for vehicle mounting, however the uses for the design remain universal. The adaptable characteristic of this type of unit means that it could be used in numerous other applications. The self-contained nature of the intended design will allow the display to be mounted almost anywhere it is required and could provide information to viewers at any of the following locations.

- Airports – Assisting passengers and pilots with flight information
- Banks – To increase sales and improve service.
- Factory Automation – To increase and enhance productivity.
- Drive-Through Restaurants – To advertise new products and improve service.
- Theatres – To advertise specials and guide movie patrons.
- Transportation – Assisting passengers in the event of a break down or accident.

When put into service, the display will be mounted according to its intended application with power either supplied via a 12V battery such as in vehicle mounting or through the use of a 240V to 12V plug pack for general power outlets. A pre-programmed message can then be called up for the unit to display using the message selector buttons. If the message required is not already stored within the display unit a new message can be created on the spot using the handheld programmer and uploaded to the unit.

3.2 Component Selection and Design Considerations

The primary task in developing an illuminated text display is the creation of the central control system that manages when each of the LEDs are to be on. This system needs to control a considerable amount of information in a short period of time. An illuminated display usually consists of smaller modules arranged together to from a larger screen, each module usually consisting of a 5 x 7 matrix of LEDs. The minimum number of bits that a control system has to address for a small 8 module display is 280, one bit for each LED. The goal of this project is to create a display containing approximately 12 to 16 modules. This means the control system is required to handle between 420 to 560 LEDs at any one time. As discussed previously, in section 2.2, the method of multiplexing modules reduces this figure, though the control of the entire display is still a significant feat.

Selection criteria for the control circuitry was developed out of the problem of controlling the large number of LEDs in a display. The control circuitry required:

- A large number of input/output ports
- Fast operating speed
- Affordability of development tools
- Easy programming
- High availability of parts for future manufacturing
- Large memory capacity
- Low power consumption

Based on these criteria a Programmable Integrated Circuit (PIC) microcontroller appeared to be the best option for the task. The microcontroller selected was a PIC16F877A 40 pin Enhanced Flash Microcontroller produced by Microchip.

3.2.1 PIC Microcontroller

A PIC microcontroller is a single integrated circuit chip that contains a processor, memory storage and input and output ports (I/O), all controlled by software stored within the chip. A microcontroller has the similar function of a Central Processing Unit (CPU) found in home PCs, in that information in the form of instructions are used to control the device. A microcontroller, however, is a trimmed down industrial version that is tailored to a specific application. Generally PIC microcontrollers have less throughput of data and a smaller memory capacity when compared the CPUs of a home PC. PIC microcontrollers are grouped by the size of their instruction word. The three current PIC microcontroller families are:

1. Base-Line: 12-bit Instruction Word length
2. Mid-Range: 14-bit Instruction Word length
3. High-End: 16-bit Instruction Word length

The following image is of the PIC16F877A microcontroller used in this project.

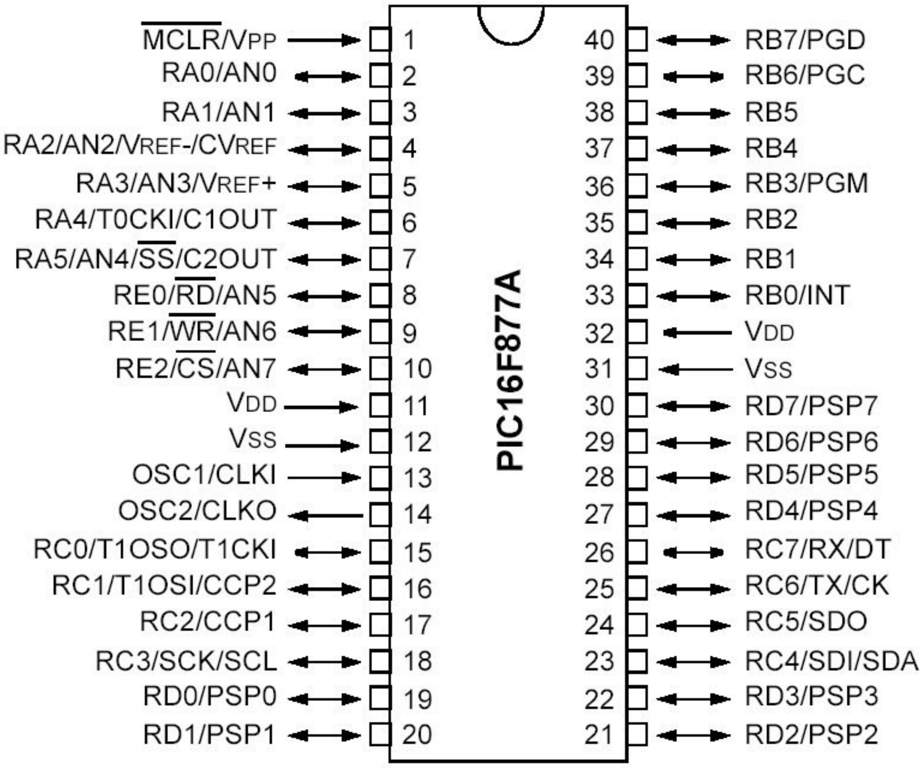


Figure 3.1 Microchip PIC16F877A pin configuration

Source: Microchip (2005)

The PIC16F877A is a Mid-Range 14-bit microcontroller that comes in a 40 pin dual-inline (DIL) package with 8K of flash memory and 256 bytes of EEPROM data memory. It is an extremely affordable device for the numerous features that it contains. The dual-inline-package format in which the IC is produced is easy to orientate on breadboard and stripboard projects, making connections to the pins straightforward. The flash memory it uses is a type of memory storage that has the ability to be electrically programmed and erased without the removal of the IC from the circuit. This in-circuit programming feature is a very significant advantage of the PIC16F877A, which aided in the creation of routines for the prototype. The ability to load, test and reprogram the chip while still within the circuit is an invaluable time saving feature. When selecting a microcontroller for the project it was important to choose a device with sufficient onboard memory, as the length of the final software code was unknown. The 8K of 14-bit word instructions was suitable for the estimated length of code that would be generated.

As controlling a large number of LEDs is an integral problem of the project, it necessitated that the microcontroller have the capability of handling a large number of input and output lines. The PIC16F877A has at maximum, 32 I/O lines for interfacing with other components. This number of control lines is sufficient for the intended task, but the increased number allowed more flexibility in the design phase. The I/O lines are grouped into five ports labelled A, B, C, D and E. Some additional features of these ports are shown below in Table 3.1.

Table 3.1 PIC16F877A Port Features

Name	No. of Bits	Pin Name	Functions
Port A	6	RA0-RA5	Digital input/output or Analogue inputs
Port B	8	RB0-RB7	Digital input/output or In-circuit programming pins
Port C	8	RC0-RC7	Digital input/output, PWM output or USART
Port D	8	RD0-RD7	Digital input/output or Parallel Slave Port
Port E	3	RE0-RE2	Digital input/output or Analogue inputs

The refresh rate of the display dictates that the control system have a fast processing speed so as to be able to process and deliver data at a rapid rate. The maximum operating frequency of the PIC16F877A is 20 MHz, which is sufficiently fast to process data for the display within the flicker fusion threshold time.

A simple programming feature was an important factor in selecting the PIC. The in-circuit programmable flash memory of the PIC16F877A is greatly beneficial as the limited budget for the project excluded the purchasing of expensive development tools. PIC microcontrollers produced by Microchip are fully supported by a range of development hardware including development boards and IC programmers. In addition to this, Microchip produces an Integrated Development Environment that contains assemblers, compilers, simulators, emulators and debugging software. The cost restraints on the project limited the purchasing of these support devices however. As a result freely available compilers and programming software was sourced from the internet and an inexpensive programmer was made on stripboard.

The supply availability of the PIC16F877A was an important deciding factor for choosing it from other Microchip PICs. This specific IC was commonly available from several electronics distributors, which is a necessity for the future manufacturing of the display.

As with most PICs the PIC16F877A contains a Reduced Instruction Set Computer (RISC) microprocessor. This type of construction increases the performance of the PIC due to the following features associated with RISC microprocessors:

- Harvard architecture
- Long word instructions
- Single word instructions
- Single cycle instructions
- Instruction pipelining

Harvard Architecture

The two most common forms of processor design are the von-Neumann and Harvard architectures. The Harvard design is characterised by having the program memory and data memory as separate memories that are accessed from separate buses. This arrangement has improved bandwidth over the von-Neumann model in which the program and data memory are retrieved from the same memory using the same bus. An illustration of both forms of processor architecture are shown below.

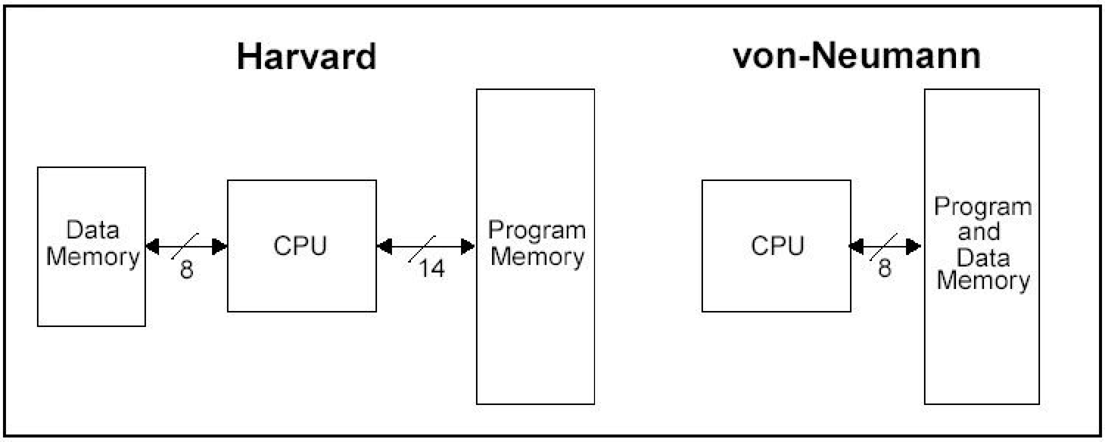


Figure 3.2 Harvard and von-Neumann Processor Architecture

Source: Microchip (2005)

Executing an instruction in a von-Neumann device often requires more than one access across the 8-bit bus to fetch an instruction. Next, the data required has to be fetched, operated on, then written to memory, all on the same bus. It is evident then, that the bus can become quite congested, reducing efficiency. Alternatively, on Harvard machines the entire 14-bit instruction used in the PIC16F877A is fetched in a single instruction cycle by means of the dedicated instruction bus. At the same time as the program memory is being accessed, the data memory can be read from or written to using the independent 8-bit data bus. These separate buses allow one instruction to be executing while the next instruction is being retrieved.

Long Word Instructions

As the Harvard architecture contains separate instruction and data buses, the instruction bit length can be made wider than the typical 8-bit data word. The large 14-bit instruction word increases the efficiency in the usage of the program memory, as the width of the program memory is optimised to suit architectural requirements of the instruction bus.

Single Word Instruction

As a result of the Harvard design, instruction opcodes are 14-bits wide, making it possible to have all single word instructions. A 14-bit wide program memory access retrieves one 14-bit instruction in a single cycle. With single word instructions, the number of program memory addresses is equal to the total number of instructions for the device. This implies that all locations are legitimate instructions. Alternatively the von Neumann architecture typically results in multi-byte instructions. A device with 4-kilobytes of program memory may therefore only contain 2K of instructions. Whilst this is a generalisation, it shows that there is no assurance that each location is a valid instruction.

Instruction Pipeline and Single Cycle Instructions

The instruction pipelining for the PIC16F877A is a two-stage pipeline which causes the fetch and execution of instructions to overlap. The retrieval of an instruction from memory takes a single machine cycle, while the execution of it takes another. Due to the overlapping of the fetching of the present instruction and the execution of the previous instruction, one instruction is retrieved and another is executed every single machine cycle. Due to the pipelining and 14-bit single instruction word of the PIC16F877A, each instruction contains all the information required and is effectively fetched and executed in a single cycle as shown in the following figure.

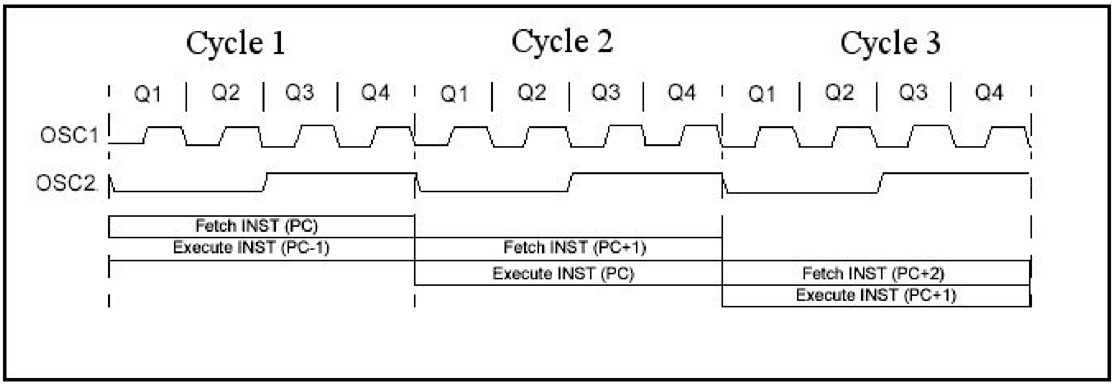


Figure 3.3 Single Cycle Instruction Operations

Source: Adapted from Microchip (2005)

The versatility and wide availability of the PIC16F877A is extremely suited to this type of design project. The 20MHz operating frequency is adequately fast to process data for the display. It has sufficient internal memory to store the controlling software code, and it is easily programmed. The greatest advantage of a PIC is the self-contained nature of the device. With the entire calculation, memory storage and input/output control system incorporated in one package, the circuitry required for the control system is greatly reduced. This one IC with minor additional parts to multiplex the I/O, can be used to control the entire array of LEDs in the display.

3.2.2 Light Emitting Diode Modules

In selecting the LEDs to use for the display the most freely available units were chosen for their ease of supply. These units needed to be successively side stackable to create a large rectangular matrix that forms the display screen. The most common form of modules of this type are the 5 x 7 dot matrix units which are widely available in a variety of sizes. For the first prototype a set of eleven 18mm Super Bright Red modules, produced by Kingbright, were purchased from RS Components. These displays were also preferred as they have pins that are aligned in a similar fashion to a DIL integrated circuit. The pin configuration made it easy to push in and orientate the LED modules on breadboard and stripboard, as well as making connections to the pins straightforward. The dot matrix of LEDs has a common anode connection for each of the 5 columns and a common cathode for each of the 7 rows. This feature of the LED modules is utilized for the multiplexing process. The following figure is a plan view of a single dot matrix module and its schematic.

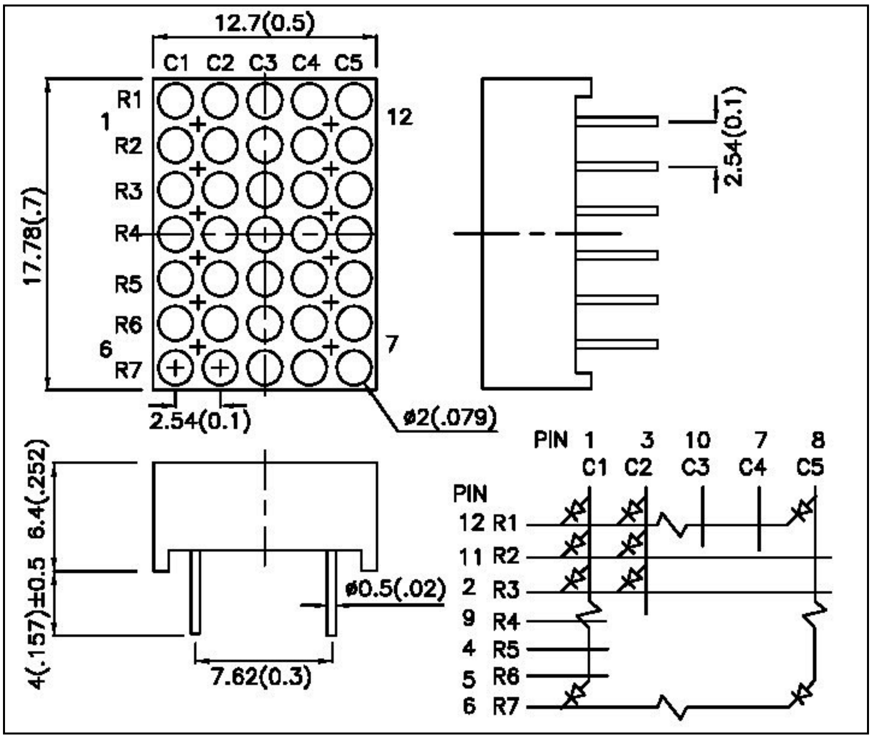


Figure 3.4 LED Package Dimensions & Circuit Diagram

Source: Kingbright (2005)

For the final display design larger versions of the Kingbright 5 x 7 dot matrix units are to be used to allow enhanced visibility of the display. These displays are approximately 40mm high and have a different pin layout to the smaller units. Using modules such as these allows for enhanced scalability in the final design. Each of the LED displays and associated circuitry becomes a discrete modular unit that can be added or removed depending on the display length required. By modularising the display units, production costs are also reduced when manufacturing the display.

3.2.3 Serial-In Parallel-Out Shift Registers

In order to obtain control over the 420 LEDs in an 11-module display, multiplexing of the I/O to the microprocessor is required. To do this, a series of serial-in to parallel-out shift registers are used to convert a stream of serial bits from the PIC to a parallel output port to the LEDs. This process effectively extends the I/O of the microcontroller by converting a serial port to a parallel one. To accomplish the large parallel output port required to address each LED, the shift registers are "daisy-chained" together. This means the serial output of one stage of the shift register sequence is cascaded to form the input to the next shift register as shown below.

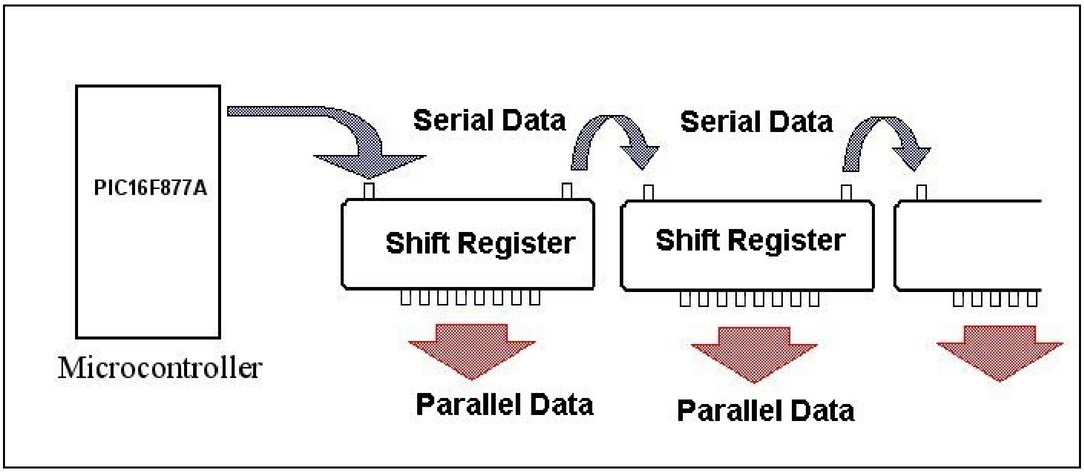


Figure 3.5 Conversion from Serial to Parallel Data

The shift registers used are a 74HC595, serial-in 8-bit parallel-out IC, which is shown below. The device has a high current 3-state output that goes into a high-impedance state when the output-enable pin is driven high. The devices have two separate registers within it, a shift register and a storage register; each register is provided with a separate clocking input. In operation, individual data bits are applied to the serial data input (SER) and clocked into the shift registers using the shift register clock pin (SRCLK). Once the entire 8 bits of serial data have been entered, the data can then be latched into the storage register where it becomes the parallel output. The output is latched by a positive-edge trigger on the storage register clock pin (RCLK).

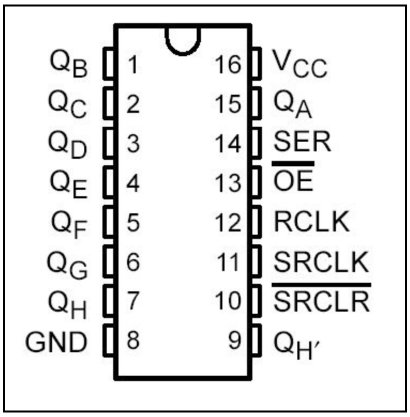


Figure 3.6 Pin configuration of the 74HC595 shift register

Source: Texas Instruments (2005)

When connected in this cascading manner, the shift register overflow pin (Q_H`) is used to transfer the excess data bit, caused by clocking more than 8 bits into the device, to the serial input pin of the next shift register in the chain. In the display the parallel outputs of the cascaded shift registers are connected to the current drivers to supply the LEDs.

3.2.4 Current Drivers

The parallel outputs of the shift registers are not rated to handle enough current to illuminate the LEDs that they control. To get full illumination of the LEDs in the prototype, each will be pulsed with 50mA, which is above the 35mA rating of the 74HC595. As a result, current drivers are used to supply each LED with enough current to fully illuminate. The devices used are ULN2003A that contain seven high current darlington arrays. All darlington pairs have an open collector which forms the input channels of the IC, and all pairs have a common emitter which is grounded. Each channel is capable of withstanding 500mA and the inputs are pinned opposite the outputs to simplify the layout on a board. The large parallel output created from the shift registers is the input to each of these arrays, which control the pattern of LEDs that are turned on. As the darlington arrays are configured as an open collector and a grounded emitter, they in fact control the LEDs by sinking current, effectively becoming an inverter which is represented in the image below. The current drivers are subsequently used to control the pattern of LEDs in a single column that are provided with a path to ground. Each column is then routinely illuminated by power supplied through a column selecting transistor for the multiplexing process.

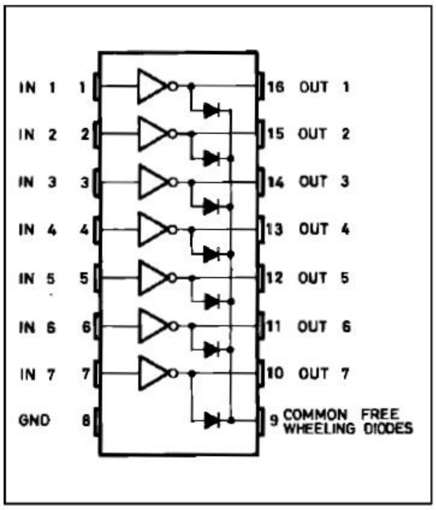


Figure 3.7 Pin Configuration of ULN2003A Darlington Array

Source Texas Instruments (2005)

3.2.5 Column Selecting Transistors

In the multiplexing process a method is needed to cross-reference every LED in the display matrix with a column and row position. The current drivers in conjunction with the shift registers combine to form a method of controlling which rows of LEDs are provided with a path to ground for sinking current. The column-selecting transistors are a simple sequential method of choosing which column of LEDs to source current to at a given time. The transistors used are a BD675 medium power transistor used for linear and switching applications. In the control circuit they are simply used for switching, when a high signal from the microcontroller is applied to the base, the transistor will conduct and current will flow to power the LEDs. The five transistors each supply the current to the five groups of 80 LEDs that are in one column, each of which is sequentially illuminated. Each LED is pulsed with 50mA every few milliseconds to sustain the illusion of visual persistence. With 80 LEDs on at any instant in time, each transistor needs to be able to carry a peak current of approximately 4A. The BD675 was selected over other transistors due to its high current carrying ability. Each device is able to conduct 4A of collector current for constant operation and 6A when pulsed.

3.2.6 Message Selector

The operation of the display is designed so that a series of short messages can be stored in the display using the programmer, and that each message can be called up using the message selector. A Binary Coded Decimal (BCD) pushbutton thumbwheel is used to select the required message for the display. The device, shown in the image below, contains up and down press buttons that are used to rotate the number wheel. The wheel displays the current number from 0 through to 9. The output pins of the device are designed to encode the current value on the wheel into the binary coded decimal format. The thumbwheels are made to snap together side by side to increase the values that can be selected. Two BCD thumbwheels are used in the design, which in combination can be used to select from messages 00 through to 99.

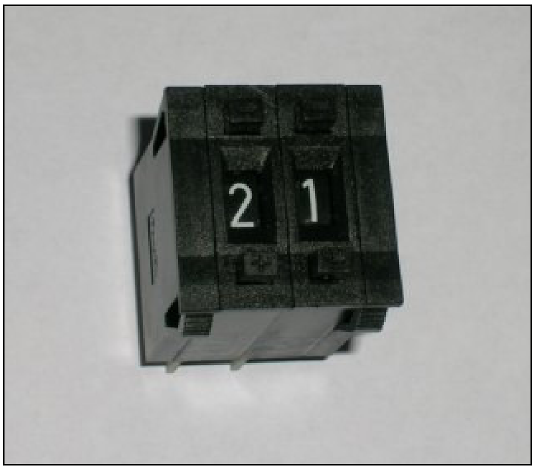


Figure 3.8 Binary Coded Decimal Thumbwheel

The BCD input format was used in the design so that in further models of the display the thumbwheel may be replaced with a connector to a programmable logic controller (PLC). The ability to adjust the design to include this feature was made so that display may be interfaced with PLCs to display warning and error messages at Downey Engineering.

3.2.7 Control Circuitry Power Supply

As the display is designed for use in a vehicle/industrial application it will be powered from a 12-30V DC supply. The display and programmer both use CMOS logic technology which typically operates at a voltage level of between 2V and 5.5V. To meet these needs a LM7805 series voltage regulator, shown in the image below, is used to step down the voltage to a fixed level of 5 volts. This is an optimum level for CMOS logic systems and is achieved with minimal external components to the regulator. Filter capacitors are also incorporated into the power supply circuit to filter out unwanted noise generated by the electronic systems of the cars engine.

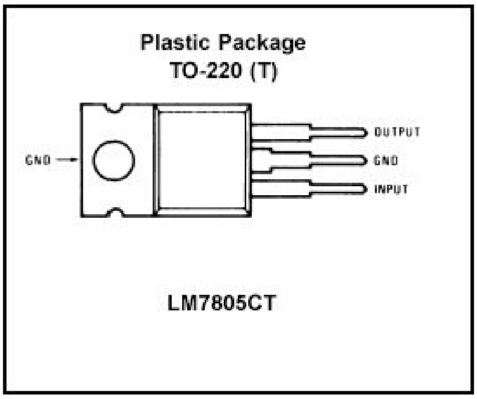


Figure 3.9 Pin Configuration of Voltage Regulator

Source: National Semiconductor (2005)

3.2.8 Switching Power Supply for the LED Displays

In providing power to the display it was realised that the LED matrix would consume a significant amount of power when compared to the control circuitry. It was estimated that the display at peak operation would draw 4A of current running at 5V. It was evident that a simple linear voltage regulator would not be sufficient for the power supply circuit. A linear regulator that reduces the voltage from 12V to 5V with an output of 4A would generate a significant amount of waste heat in the process. This design would be very inefficient; furthermore a linear voltage regulator capable of outputting 4A would require a bulky heat sink for dissipating waste heat. It was decided that two voltage regulated circuits be incorporated in the display; one linear voltage regulator to supply the low current to the control circuitry such as the PIC microcontroller and shift registers and another switching step-down voltage regulator for the higher current drawn by the display. The regulator used in the design is a National Semiconductor LM2576, which is shown in the image below. This integrated circuit is ideal for the application, as it requires a minimum number of external components. The adjustable version of this IC is used so that the voltage to the display LEDs can be varied to allow brightness control to be included in the design.

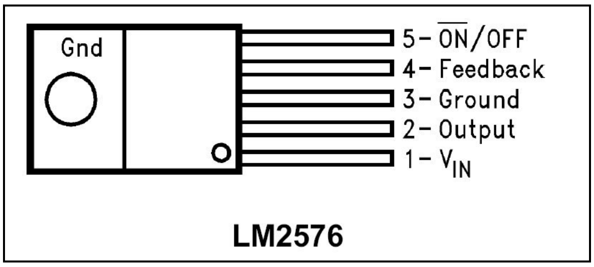


Figure 3.10 Pin Configuration of Switching Voltage Regulator

Source: National Semiconductor (2005)

The voltage regulating circuit used to provide power to the display is shown below. A potentiometer is used in the resistor divider to adjust the feedback voltage to the LM2576. This allows the output voltage from the regulator circuit to be adjusted. In the first prototype a voltage range of 2.5-8V was used to suit the smaller LED modules. The values of resistors R1 and R2 will need to be selected in further prototypes to suit the range of voltages required for the larger LEDs used.

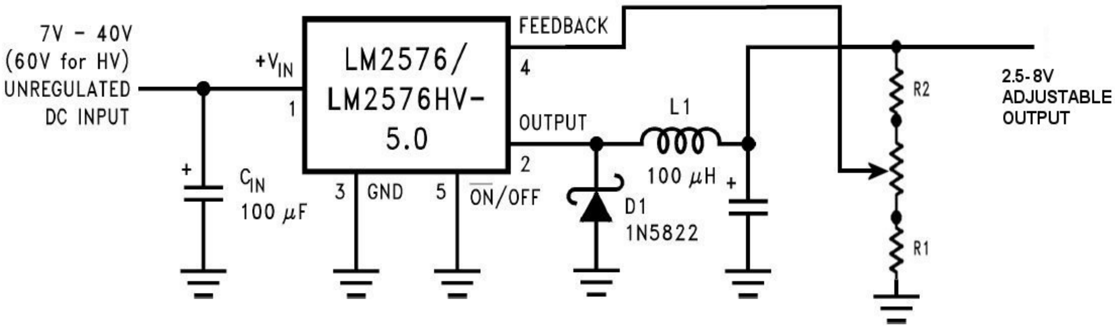


Figure 3.11 Switching Voltage Regulator Circuit

Source: National Semiconductor (2005)

3.2.9 RS-232 Transceivers

For the communication between the handheld programmer and the display unit, the RS-232 communication standard was used due to its ease of implementation and its wide usage in areas of communication. The serial communication was also chosen as the PIC microcontroller contains a Universal Synchronous Asynchronous Receiver Transmitter (USART), which is fully integrated in hardware. The USART transmits and receives data at the 5V levels of CMOS logic, while the standard for RS-232 communication requires logic levels of approximately ±10V. To overcome the difference in logic levels a Maxim MAX202 RS-232 transceiver IC, shown below, is used as an interface. This IC produces the required ±10V for RS-232 communication when supplied from a 5V source. When data is transmitted to the display using the serial connection, the MAX202 converts the signal to the CMOS levels used by the microcontroller.

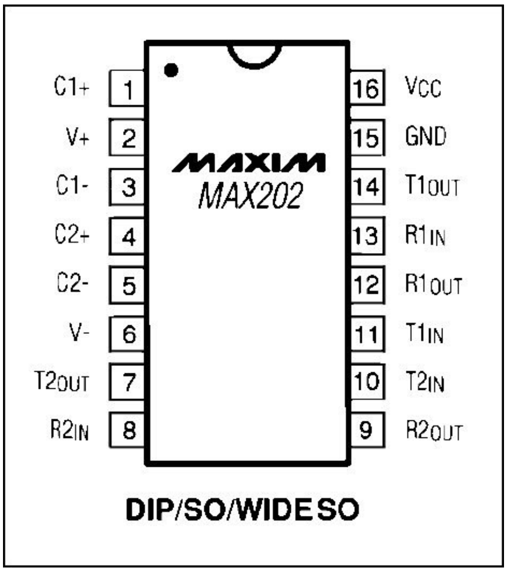


Figure 3.12 Pin configuration of Maxim RS-232 Transceiver

Source: Maxim (2003)

3.3 Handheld Programmer Components

The creation of the handheld programmer was done with special attention to a user-friendly approach for the design. The handheld programmer is based primarily with a mobile phone layout in mind. The familiarity with mobile phones in today's society has prompted this decision. An arrangement similar to creating and sending text messages on a mobile phone was used. The primary control circuity that is used in the programmer is the PIC16F877A microcontroller that is also used in the display. The choice to use the same microcontroller for both devices was clearly beneficial. By duplicating the control circuitry from the display to the programmer, time was saved in the prototyping and testing phase due to the ease of reproduction once an initial circuit is made. The additional devices the microcontroller interfaces with in the programmer includes a liquid crystal display (LCD), a 4x4 alphanumeric keypad and a Maxim RS-232 transceiver chip.

The programmer was made portable so that the display can be placed in the field of use and still be reprogrammed with new messages. By designing the system in this way, modifications can be made to messages without having to take the display out of service. It also eliminates the need for carrying a bulky laptop around if in-field programming is required. It is anticipated that the programmer unit will be contained in a small handheld case like a mobile phone when fully developed.

3.3.1 Liquid Crystal Display

A QP5515 two line 16 character LCD display is used in the project to display the massages created on the programmer unit before they are loaded into the display. The LCD can be interfaced with the microcontroller in either a 4-bit or 8-bit data bus mode. The 4-bit mode was chosen for the design so as to conserve the number of I/O lines used in the microcontroller. There are six other control lines used to operate the device. Positive and negative supply lines are used to power the device and an adjustable contrast line that varies the screen intensity as the input voltage is varied from 0 to +5 volts. Additionally, the enable signal, read/write and register select lines are other lines used for addressing and selecting the LCD. The LCD also contains a complete onboard ASCII set and can therefore display any of the characters that may be used on the LED display system.

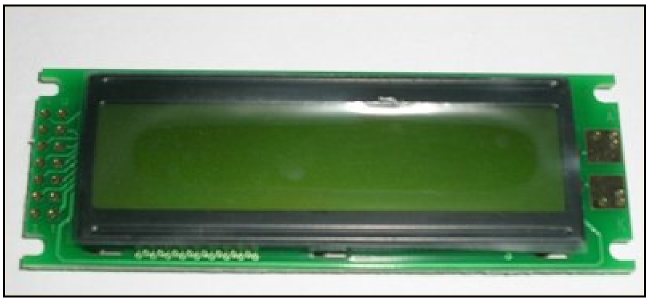


Figure 3.13 Liquid Crystal Display

3.3.2 Alphanumeric Keypad

An alphanumeric keypad was used in the programmer to more closely resemble the mobile phone layout. The familiarity of most people with creating text messages with such keypads prompted this decision. Another significant advantage of the alphanumeric keypad when compared to the typical QWERTY style keyboard, used for computers, is the considerable reduction is size. By using a small keypad of this size the programmer remains very compact while still providing the full amount of characters available from the normal QWERTY keyboard. To provide the type of user input typical of mobile phones, an input method termed multi-tap is used. In this method the user repeatedly presses a button of the keypad to scroll through the letters allocated to that button. For example, pressing the button labelled ‘5’ once would present the letter ‘J’, if press a second time the letter ‘K’, a third time the letter ‘L’ and finally the numeral ‘5’. An image of the keypad used is shown below along with letters associated with each button in multi-tap.

1	2 ABC	3 DEF
4 GHI	5 JKL	6 MNO
7 PQRS	8 TUV	9 WXYZ
*	0	#

Figure 3.14 Multi-tap Keypad Layout

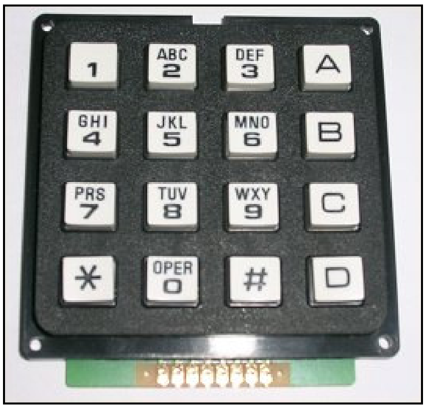
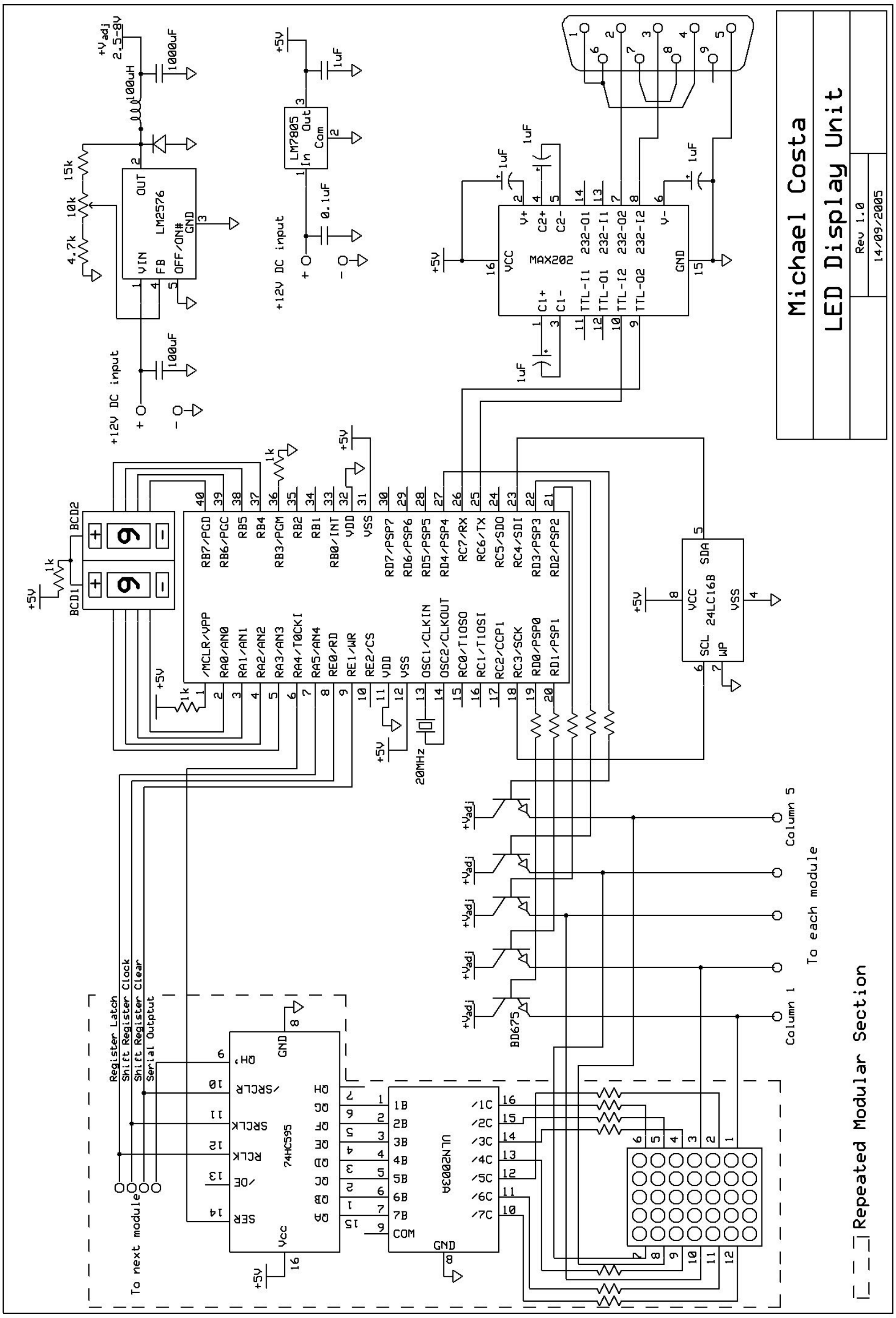
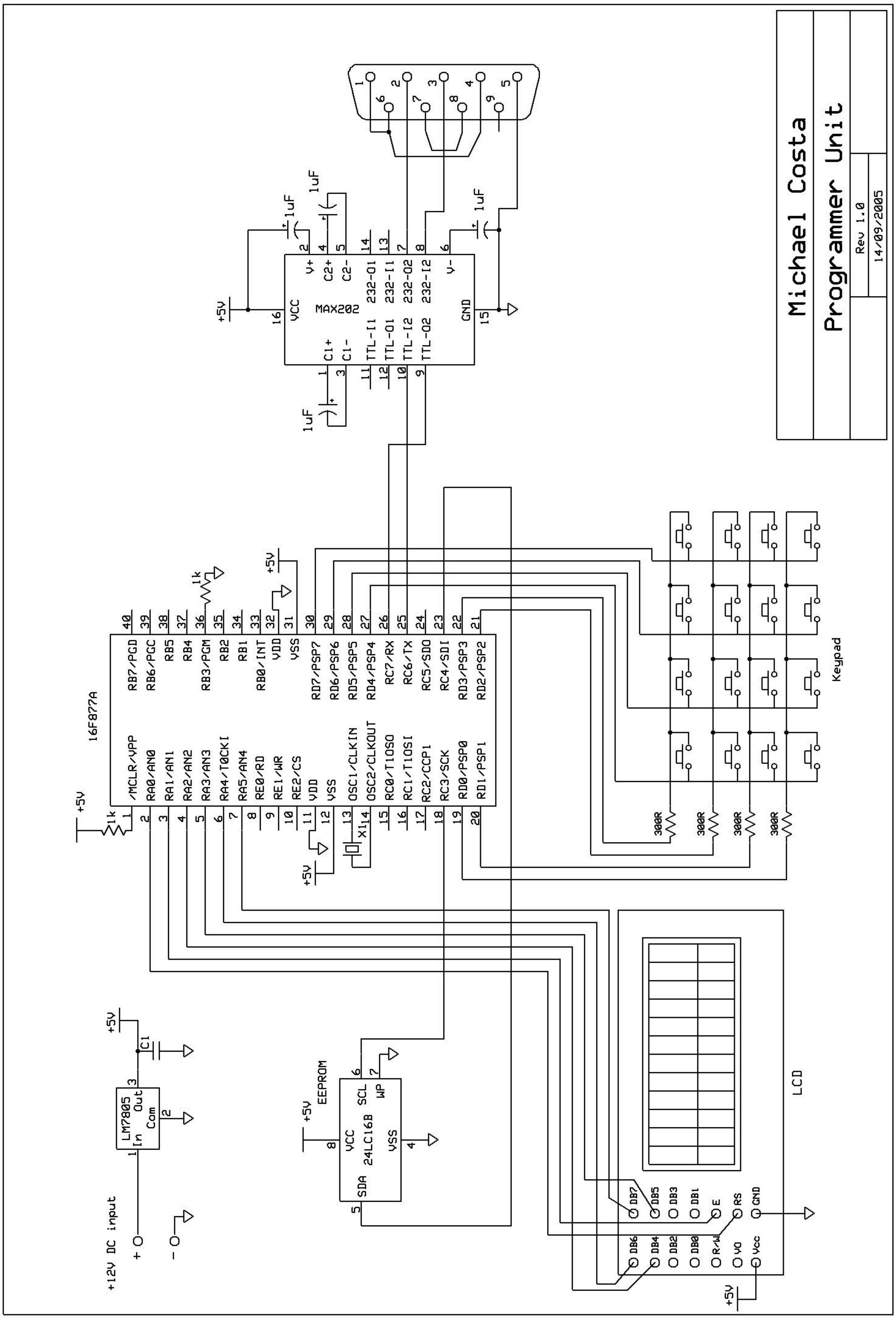


Figure 3.15 Alphanumeric Keypad

3.4 Display Schematic



3.5 Programmer Schematic



Chapter 4 Implementation

4.1 Prototype Construction

In developing the first prototype for the project, time and expense were the most important factors under consideration. The prototype was constructed in order to test the physical viability of the design, ensuring that the circuitry operated as expected. The initial construction needed to be completed promptly to meet the deadlines of the project and to allow time for correcting any faults. The prototype was developed extremely economically using several breadboards and stripboard on which the components were mounted. Through this type of construction, modifications could be easily made before entering into the development of larger scale, and more costly prototypes.

To begin construction the low voltage programmer was created. The microcontroller circuit that provides the main control for both the display unit and programmer was then constructed on a breadboard. The reason for mounting the main control circuitry on a breadboard was that the ports used for various tasks would ultimately change as the project developed and until the layout was finalised the pin connections could easily be altered as required. The microcontroller requires very few external parts for typical operation. Power was supplied to the device from the linear regulator via pins Vss and Vdd, and a filter capacitor was placed as near as possible across the supply lines to reduce external noise. A 20 MHz external crystal was connected to pins OSC1 and OSC2 of the device to provide timing oscillations. An operating frequency of 20 MHz was selected as it is the highest operating frequency for the 16F877A microcontroller. This high frequency crystal allowed freedom from any timing constraints in processing data for the display. By selecting the highest possible operating frequency the correct illumination pattern for every row of the display can be retrieved and serially streamed to the shift registers within the time constraints of the flicker fusion threshold mentioned in section 2.2.

The master clear pin MCLR is an active low reset to the PIC16F877A and was therefore connected to Vcc using a pull-up resistor. This prevents the microcontroller from resetting during operation. In the final design a pushbutton may be incorporated to pull the MCLR pin low as a master reset to the display. The microcontroller was programmed in low voltage mode, such that a voltage source of only +5V was needed to download code in the PIC's memory. In the low-voltage programming mode a 1kΩ resistor is required on pin RB3 to ground to ensure continued operation. In this mode RB3 is dedicated to the programming function and ceases to be used as a general purpose I/O pin. Although the loss of one of the pins on the 8-bit port was troublesome it was an acceptable loss for saving money on buying a specialised PIC programmer.

These few external components are all that was needed to start using the PIC microcontroller. Once the control of ports on the 16F877A was established, through simple test programs, the construction of the serial to parallel shift register chain was attempted. Firstly a single shift register was wired to the microcontroller and tested, then two shift registers were cascaded on the breadboard to trial the cascading process. Once satisfactory results were achieved, a large-scale chain of shift registers and associated current drivers and LED modules were completed. The significant number of IC's used in this circuit meant that it could not be constructed on a single breadboard due to size limitations, instead stripboard was used. To decrease the expense in developing the prototypes for the project it was decided that as many parts as practical would be reused in each successive model. To facilitate this, IC sockets were soldered on to the stripboard into which the various components could easily be placed and removed. This process is shown in the image below. Instead of soldering the integrated circuits directly onto the board, using the IC socket method, components could easily be exchanged and reused in further prototypes.

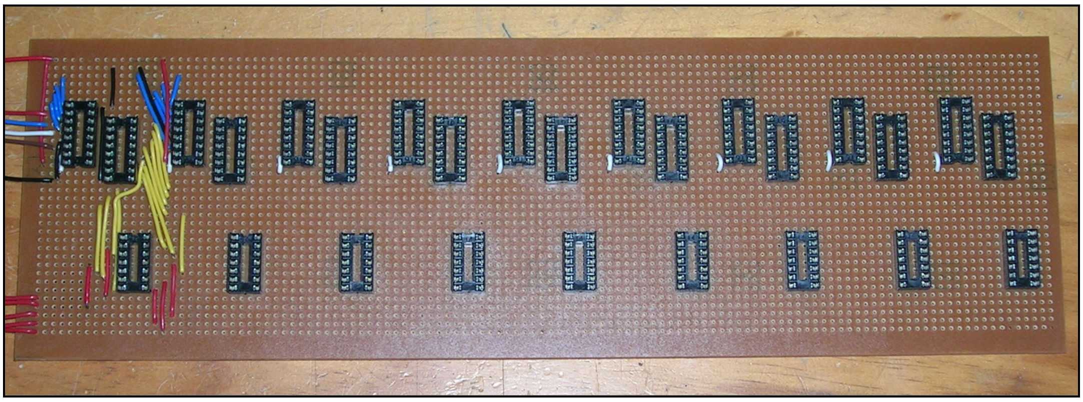


Figure 4.1 Stripboard Mounted with IC Sockets

Using stripboard and IC sockets, although very inexpensive, it was also extremely time consuming. Each connection on the board needed to be individually soldered, including all the wiring connections. The copper strips on the reverse side of the board also required cutting in specific places to isolate components, which again was extremely time consuming. An image of the finished prototype display is shown below.

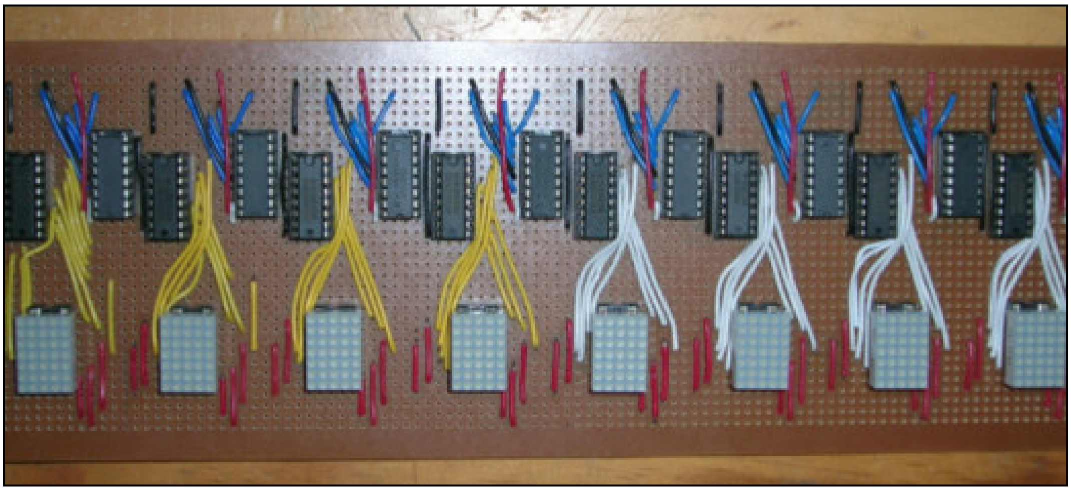


Figure 4.2 Initial Display Prototype

After completing the display circuit the switching power supply circuit was developed. While this was being constructed, the linear regulator was used to provide power for the testing of each of the displays individually. As stated previously the current drawn from the entire display is too great for the linear regulator to handle alone. Due to this, the linear regulator was only used to power one display at a time for testing purposes while the switching regulator was being built.

After constructing the display of the first prototype, the circuitry for the serial connection between the handheld programmer and the display unit was developed. This circuit consists of a Maxim RS-232 transceiver IC and additional external capacitors. The purpose of this circuit is to modify the voltage levels of the inputs from the CMOS logic of the microcontroller to the RS-232 serial communication voltage levels typically ±10 volts. Once the serial link was built, communication to a PC was tested by running a communication terminal program, which confirmed the serial data transfer. As the handheld programmer was not yet complete, the PC was an effective substitute as it also can be used to program the display. The serial link to the display was verified using a routine that received a character typed in the terminal program on a PC. The routine immediately returned the same character that was received via the serial link. The routine effectively received data from the PC and bounced it straight back. This established that the circuit and software created worked correctly and could be used if implemented in both units to send data between the handheld programmer and the display.

A prototype of the handheld programmer was then constructed to test the circuit's feasibility. An image of the prototype developed on a breadboard is shown below.

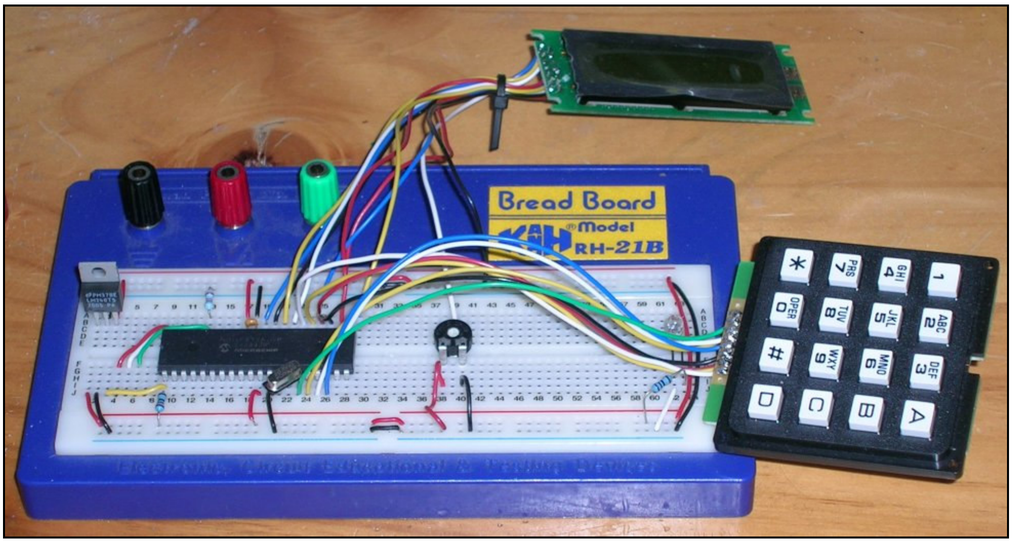


Figure 4.3 Programmer Prototype

This circuit was subsequently used to create messages using the keypad; each was displayed via the LCD. The serial transceiver was then implemented with the programmer allowing the prototype to transmit the created messages via the serial link to a PC. The addition of the serial link and the message transmitted is shown in the images below.

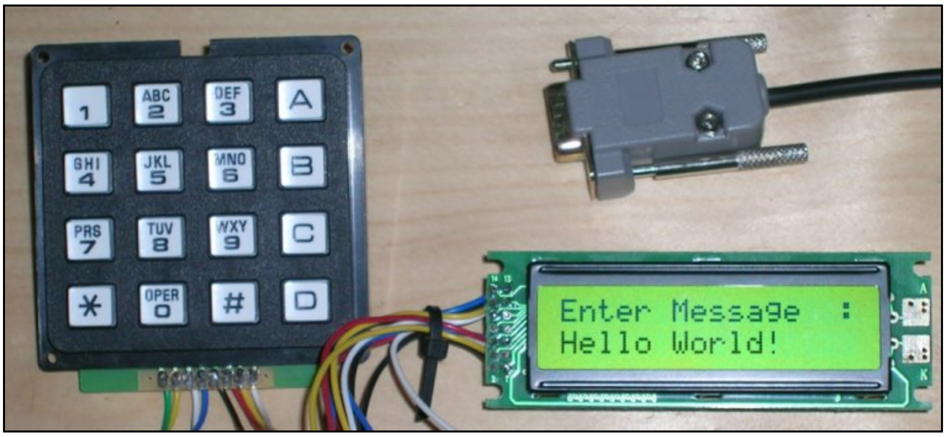


Figure 4.4 Serial Link Message

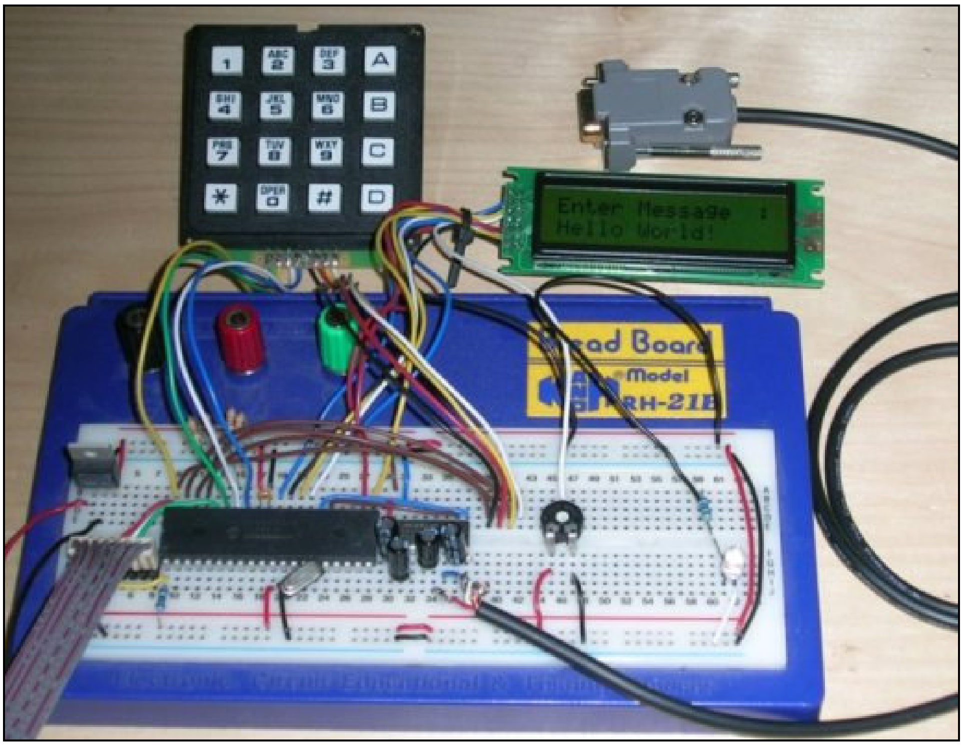
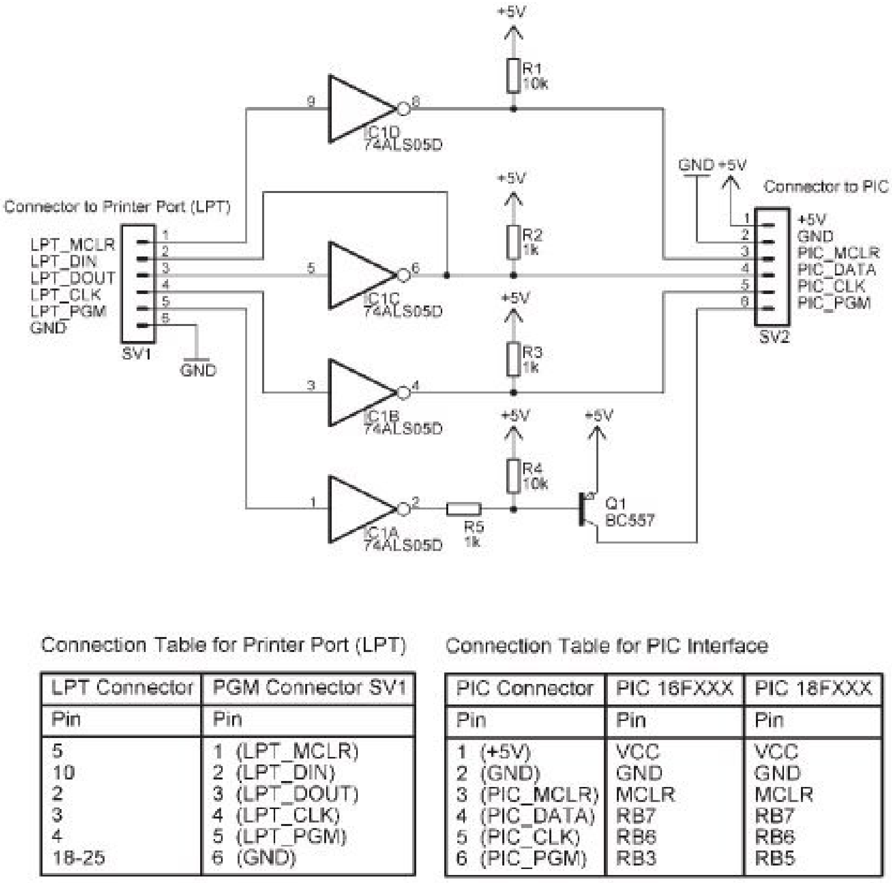


Figure 4.5 Programmer Prototype

4.2 Low-Voltage Programmer Construction

The low-voltage programmer is used to transfer the code developed on a PC to the memory of the microcontroller. As the programming devices and software for the PIC16F877A would have been an additional cost for the project, it was decided that a simple programmer could be built from a design created by Christian Stadler (2004). The programmer circuit consist of a single hex invertor IC, a BC557 transistor and several resistors. The low-voltage programmer is supplied with power from the 5 volt supply for the microcontroller and uses the printer port of a PC to communicate with the 16F877A. The programmer interfaces with 4 pins of the microcontroller, the programming voltage input (MCLR/Vpp), the low-voltage programming enable pin (RB3/PGM), the in circuit programming clock (RB6/PGC) and the in-circuit programming data pin (RB7/PGD). With these lines of the microcontroller interfaced with the programmer, the alternate end is plugged into a PC using male D25 parallel connector. Free software developed by Christian Stadler (2004) was also used to load the programs developed into the PIC16F877A. The schematic of the low-voltage programmer and an image of the completed circuit are shown below.



Connection Table for Printer Port (LPT)

LPT Connector	PGM Connector SV1
Pin	Pin
5 10 2 3 4 18-25	1 (LPT_MCLR) 2 (LPT_DIN) 3 (LPT_DOUT) 4 (LPT_CLK) 5 (LPT_PGM) 6 (GND)

Connection Table for PIC Interface

PIC Connector	PIC 16FXXX	PIC 18FXXX
Pin	Pin	Pin
1 (+5V) 2 (GND) 3 (PIC_MCLR) 4 (PIC_DATA) 5 (PIC_CLK) 6 (PIC_PGM)	VCC GND MCLR RB7 RB6 RB3	VCC GND MCLR RB7 RB6 RB5

Figure 4.6 Low-Voltage Programmer Schematic

Source: Christian Stadler (2004)

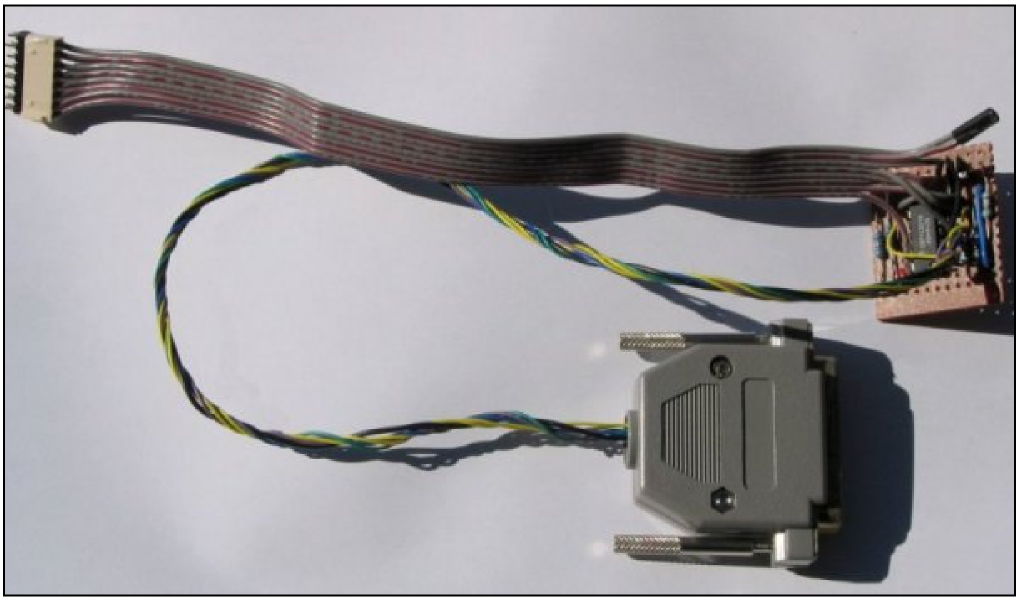


Figure 4.7 Low-Voltage Programmer

Chapter 5 Testing and Problems

5.1 Testing Procedure

The procedure of continually testing the circuity as it was progressively built upon, was an essential part in maintaining the integrity of the design. As each new section of the system was developed, it was tested to see that it functioned properly, and then integrated with the existing circuitry. Finally the combined circuit was tested to verify that jointly both systems operated as expected. This process of testing in a modular fashion ensured that once a new module was constructed it was checked for faults before integrating it with the existing system.

The first form of testing done was on the low-voltage PIC programmer and microcontroller circuit. It was important to establish the functionality of this circuit, as it forms the central control for all other parts of the system. The microcontroller and low-voltage programmer were constructed and a series of simple experimental programs were developed to test each of them. These programs were designed to ensure that the programmer performed its task adequately and also to help better understand the control of the ports on the PIC16F877A. As this was the first time for using this particular device and associated software, it was important to be familiar with the devices strengths and limitations.

A simple program designed to flash LEDs was the first test used to develop awareness of the microcontroller's port operations and to calculate appropriate timing code for future applications. A puzzling problem that manifested during this simple test was that toggling bits in a specific port appeared to clear other bits within the same port. After much experimentation, the source of the problem was found to be the functionality of the bit addressing command in the C compiler. The command PORTX.FY within the compiler was designed so that all bits in port X are set to 0 and the bit number Y in that port is set to 1. It was expected that this command maintained the bit pattern on the port and only affected bit number Y. As this was not the case, the addressing of individual bits on a port was achieved by composing 8-bit data words that exhibited the appropriate bit pattern, then applying this entire word to the port.

The next modular section constructed consisted of a single shift register used to convert a serial data flow from the microcontroller into a parallel output that controls the LEDs. A significant amount of time was spent on perfecting this process as timing and the control sequence of this stage was extremely important. The program developed for this process cycles through the data required and selects one bit at a time from the data word, then applies it in conjunction with a clock pulse to the shift register until all the data bits are transmitted. Once this is completed, a latching pulse signals the shift register to apply the received data to its parallel output port. Individual LEDs were connected to these output pins of the shift register to watch the conversion from serial input to parallel as it occurred. By significantly slowing the transmitting rate and latching the output after each bit was transferred, the change of data on the output of the shift register could easily be observed. Once satisfied that the conversion was taking place correctly a second shift register was added to form a second stage in the cascaded chain. The same clocking and latching lines were used for each of the registers however the overflow bit from one stage formed the input into the next, thus providing an unbroken chain of serial data. When both register where completed, tested and worked properly then the next step of creating two complete LED displays was attempted.

To expand the shift register chain to form two displays, the current driver ICs and column selecting transistor circuitry was developed. The pin connections for the current driver ICs are manufactured so that all inputs pins are on one side of the package and the corresponding outputs are placed opposite. This handy arrangement allowed the current drivers to directly interface with the parallel output ports of shift registers by placing them parallel to one another.

The row transistors where implemented with the collectors of each connected to the positive rail of the switching power supply via a current limiting resistor. The five base pins of each of the transistor were connected to the microcontroller to be sequentially turned on. As each transistor is selected the corresponding column of LEDs in the display is supplied with power. To do this the emitters of these transistors were connected to the five common anodes on all of the LED dot matrix modules. The seven outputs of each of the current drivers where then connected to the seven cathodes of each row of the displays to provide a path to ground. This completed circuitry provided two fully working displays after creating the software to operate the display.

The creation of the display control algorithms was quite an involved process requiring many days of testing and modifying. The task this software performs involves retrieving the bit pattern for the first column in each letter of the display and serially outputting this to the shift registers. Once the data for the first column is transmitted, a latching signal is sent to the shift registers, followed by a pulse to the base of the column number 1 transistor, which illuminates the first column of LEDs. The entire process is then repeated for columns two to five. As expected, developing the timing and sequence of control for this program was quite difficult. With the completion of this routine the two display units were then able to display the first characters for the project. The first two display modules that were completed to test the cascading process are shown in the following image.

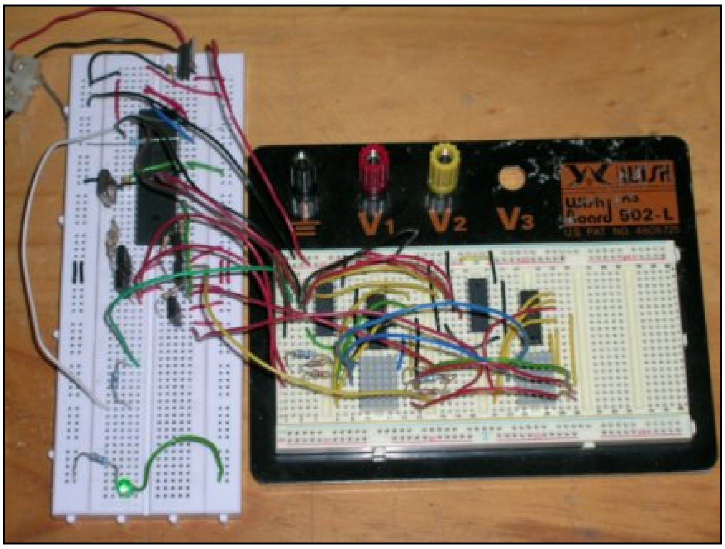


Figure 5.1 Initial Displays Tested

The next step in implementation involved the construction of another 9 modules for the display. The testing process required for this stage was quite repetitive. The construction was broken down into modular sections each containing a LED display, a current driver and a shift register. As each of these discreet display modules was built on stripboard, all connections were checked for continuity to ensure the integrity of the design. This was done because each successive module was dependent on the ones before it functioning correctly. If one shift register in the cascaded chain failed to work all the modules following it would also malfunction. To ensure a fault did not occur a rigorous routine of testing each module upon completion was applied.

After the completion of all displays extensive testing and modifications were done on the pulse timing and current supplied to the LEDs of the displays. Various combinations of pulse lengths and current values were experimented with to achieve a high display brightness that did not flicker. The first problem encountered was that characters of the display flickered for the initial refresh rate chosen. After experimentation with higher refresh rates a simple error was discovered. It was assumed that an on time of approximately 20 milliseconds for a column of LEDs would be sufficient to prevent most of the flickering. What was overlooked however, was that each of the columns are only on for one fifth of the time as each of the other four columns are pulsed in succession. This effectively requires the refresh rate to be five times faster then what was originally thought. Fortunately the 20MHz crystal selected for the operating frequency meant that the on time could easy be reduced. The adverse effect of reducing the on time is that the luminance of the LEDs is reduced making the display appear dim. To counter this, the current driven through the LEDs was increased making the display brighter. As the displays used for the prototype are different from the ones to be used in the final product a degree of fine-tuning will need to be done to attain the best ratio of LED current and on-time to achieve maximum brightness without damaging the display.

The second problem found with the initial design was a mistake made with the placement of the current limiting resistors. To save space and minimise the number of parts required, it was that decided that instead of placing a resistor in series with each LED cathode, a resistor would be placed on the 5 anodes common to all displays in the unit. The unforseen drawback was that a resistor placed here limited the current to the entire column, effectively dividing up the set current between each illuminated LED. The visible effect of this was that columns with all LEDs on at a time appeared dim and columns with not as many LEDs illuminated were relatively bright. This oversight was resolved in the first prototype by reducing the value of the current limiting resistor effectively increasing the brightness of all the LEDs so that the mistake was scarcely noticeable. A constructive result of the mistake was that the error was identified in the prototype and will be resolved in the next design by placing a LED at the cathode of each LED module. This is a clear example of the benefits in creating prototypes for testing purposes.

The switching voltage regulator that powers the LEDs of the display was constructed next. It was simply tested by measuring the output voltage of the circuit once it was constructed. As a potentiometer was incorporated in the adjustable power supply, it was varied and the output was measured using a voltmeter to observe the response. The range of voltage output from this circuit was from 2.5V to 8V. Depending on the voltage required for the final display the feedback resistors that set the limits to the output range can be replaced accordingly. An image of the complete display segments after the switching voltage regulator was interfaced is shown below.

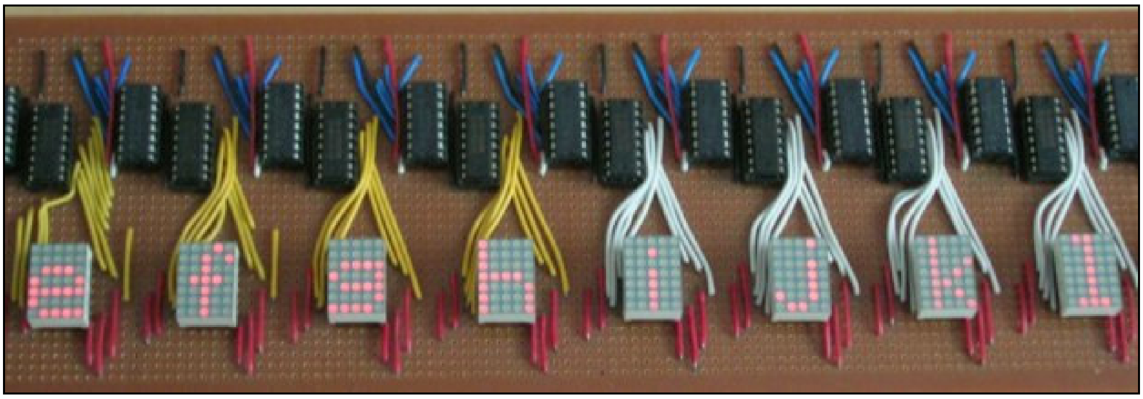


Figure 5.2 Initial Display Test

The next modular section of the display that was constructed and tested was the RS-232 transceiver circuits, used for the communication between the handheld programmer and the display. The purpose of these devices is to convert the voltage from the +5V supply to the inverted logic levels of ±10V compatible with the RS-232 standard. The testing carried out on this section of the design was a simple check of the output voltages for both logic levels. The voltage tests proved that the circuitry was functioning correctly outputting +10V for a logic input of level 0 and –10V for an input of level 1.

A routine was then developed to test the serial connection. This routine received an ASCII character from a PC running a terminal program and immediately returned this same character via the serial link. This confirmed that the serial data transfer routine operated, correctly receiving and transmitting the data sent.

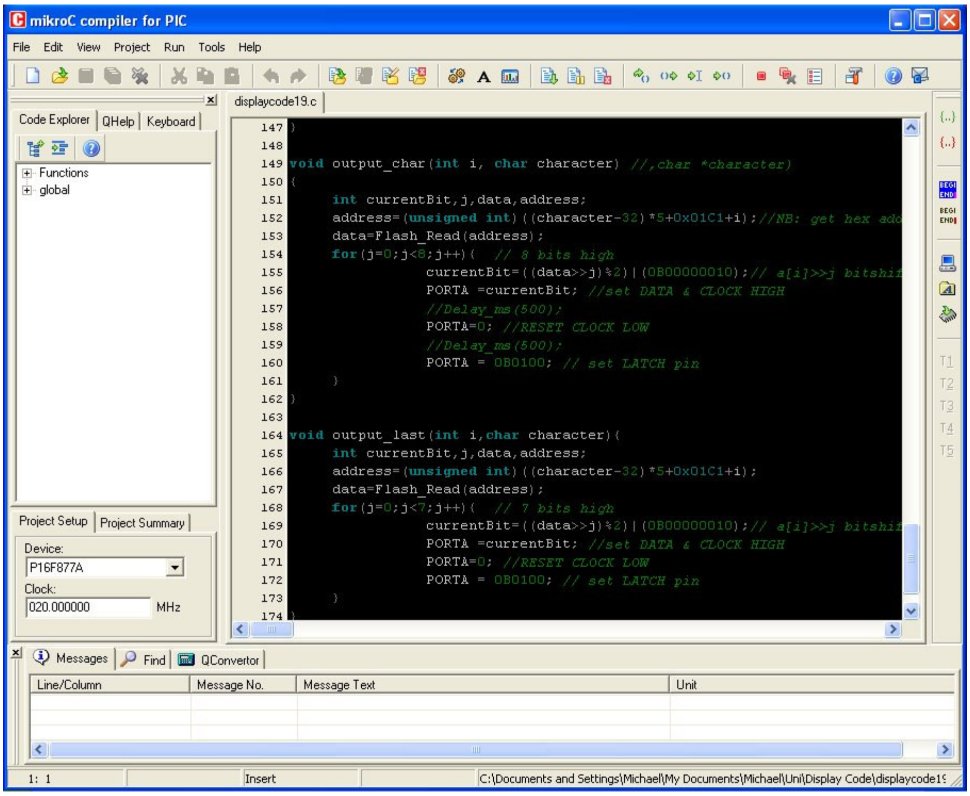
By creating messages on the keypad and viewing them on the LCD the prototype for the handheld programmer was then tested. These messages were then sent across the serial connection, concluding the testing phase of the project.

Chapter 6 Software Development

6.1 MikroC Development Environment

In creating code to control the display and programmer, it was important to choose compatible development software that was also freely available. A decision was made to develop code in a higher level language in order to make the programming process easier. A disadvantage of using a higher level language is that it is less economical; generating a larger, less efficient code then lower level languages. This can become a problem if the memory capacity of the target device is limited. The benefit in higher level languages however is that the developer can more easily and quickly comprehend the path of logic created. The program is more clear as the code comprises of commands based on structured English words such as if, else and while. The higher-level language compilers also have the benefit of containing complimentary libraries that contain useful code. The language used to develop code for the project was C, because an evaluation version of a C compiler for the PIC16F877A called MikroC was available for free download.

The MikroC compiler software is a specifically designed integrated development environment (IDE), used to generate code for Microchip's PIC microcontrollers. The latest version 2.1 of the development software supports a large number of chipsets including microcontrollers in the PIC12, PIC16 and PIC18 families. Apart from the code editor the MikroC environment also contains other useful development tools including a source-level debugger to simulate the operations of a Microchip microcontroller. The debugger was found to be very helpful in finding and resolving problems in C code written for the microcontrollers. This tool gives the ability to monitor program items while the program is running. It displays variables and registers of the microcontroller, their addresses and values. Each of these are updated as the simulation progresses. In addition to these features MikroC also contains the USART communication terminal for RS232 communication, which was helpful in developing and testing the serial communications code for the project. The image below is of the MikroC integrated development environment.



```
}

void output_char(int i, char character) //,char *character)
{
      int currentBit,j,data,address;
      address=(unsigned int)((character-32)*5+0x01C1+i);//NB: get hex ado
      data=Flash_Read(address);
      for(j=0;j<8;j++){   // 8 bits high
                  currentBit=((data>>j)%2)|(0B00000010);// a[i]>>j bitshif
                  PORTA =currentBit; //set DATA & CLOCK HIGH
                  //Delay_ms(500);
                  PORTA=0; //RESET CLOCK LOW
                  //Delay_ms(500);
                  PORTA = 0B0100; // set LATCH pin
      }
}

void output_last(int i,char character){
      int currentBit,j,data,address;
      address=(unsigned int)((character-32)*5+0x01C1+i);
      data=Flash_Read(address);
      for(j=0;j<7;j++){   // 7 bits high
                  currentBit=((data>>j)%2)|(0B00000010);// a[i]>>j bitshif
                  PORTA =currentBit; //set DATA & CLOCK HIGH
                  PORTA=0; //RESET CLOCK LOW
                  PORTA = 0B0100; // set LATCH pin
      }
}
```

Device: P16F877A

Clock: 020.000000 MHz

1: 1

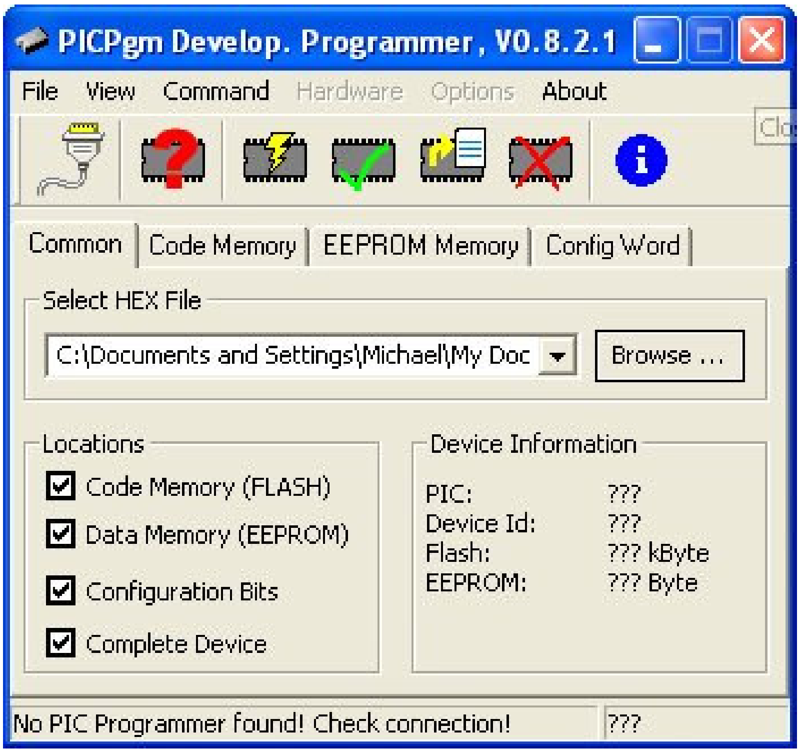
C:\Documents and Settings\Michael\My Documents\Michael\Uni\Display Code\displaycode19

Figure 6.1 MikroC Integrated Development Environment

Source: MikroElektronika (2005)

6.2 PICpgm Development Programmer

The PICpgm Development Programmer software was used to load the hexadecimal file generated by the MikroC compiler into the program memory of the PIC16F877A microcontroller. This effective software is freely available from the internet and provides an easy method of loading and reading code to a microcontroller when used in conjunction with the low-voltage programmer circuit. The program communicates with the target microcontroller using the printer port on a PC, via the low-voltage programmer. The operation of this software is very uncomplicated, with features including an auto-detect hardware button for identifying the specific programmer connected to the printer port and an auto-detect PIC feature for recognising the target device for the hexadecimal file. A screenshot of the program is shown below.



Select HEX File

C:\Documents and Settings\Michael\My Doc

Locations

☑ Code Memory (FLASH)

☑ Data Memory (EEPROM)

☑ Configuration Bits

☑ Complete Device

Device Information

PIC:	???
Device Id:	???
Flash:	??? kByte
EEPROM:	??? Byte

No PIC Programmer found! Check connection! ???

Figure 6.2 PICpgm Development Programmer

6.3 Code Developed For The Display

The process by which the C code was written for the microcontroller was through a bottom up implementation using modular programming. The advantage of this form of implementation is that the lower level modules of the program are completed first so that the dependency on the configuration of the hardware is completed in the early stages of the development. The early software developed for the project was simple routines designed to help better understand the functions and libraries included in the MikroC IDE, as well as to learn the control, port features and associated commands of the microcontroller. These programs included the initialisation of the data direction registers in the microcontroller such as TRISA for the direction of bits in port A. By setting the bits in each register, the various lines within the port could be set for either a general purpose input or output. Once mastering the initialisation for the microcontroller, timing routines were developed to accurately control outputs to the shift registers and column selecting transistors.

The routine that addresses the shift registers retrieves the bit pattern for the next column to be displayed from a character map in the microcontrollers memory. Next, a series of loops bit shifts the outgoing data in conjunction with a clock pulse to form a serial output. This routine is nested within a further loop that cycles through the 5 columns of the display, latching each in succession and selecting the appropriate transistor to provide power to each column. There are two copies of the routine that serially delivers data to the shift registers. One is only used to output data to the first register in the chain and the other is used for all the other shift registers. This is due to the fact that data delivered into registers 2 to 11 must be previously passed from other registers in the chain. This process requires multiples of 8 bit shifts to completely traverse one register. The register 1 however only requires 7 shifts of data because it is the earliest in the chain and receives its input directly from the microcontroller.

Once the routine was developed to show characters on the prototype, a character map of all the ASCII characters to be use for display had to be created. This table consists of an array of five binary words for each character. An image of the entire character map is shown below. An example of how this data is arranged for a letter is as follows. The letter is broken down into a matrix of pixels that is 5 columns wide and 7 pixels high, each pixel corresponding to an individual LED in the final display. These patterns are then used to create five 8-bit words that form a character. Eight bits are used to form each word due to the 8-bit nature of the memory. By using an 8-bit word the data can be arranged concurrently within the memory, however only 7 of the bits are truly used. The most significant bit in each word is redundant, used to complete the 8-bit word and each was set to a logic zero.

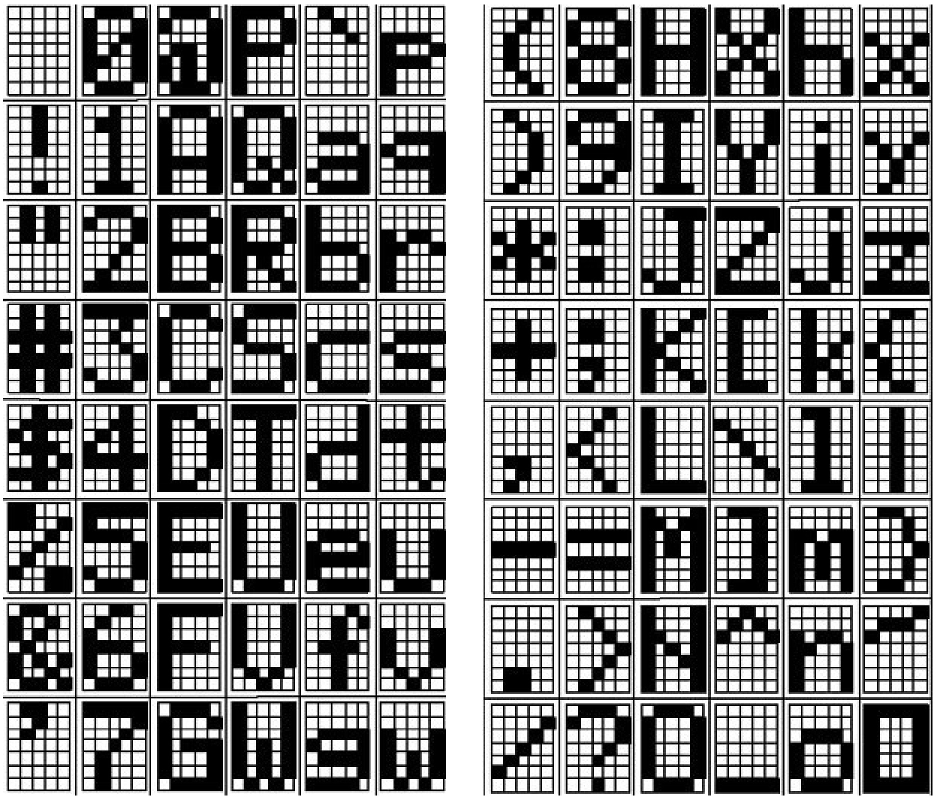


Figure 6.3 Character Map of ASCII Characters

Source: Maxim (2005)

After coding the character map, routines were also created to control the LCD, keypad and serial data link of the programmer prototype. The code used to operate these devices was derived from the libraries functions available in the MikroC compiler. The pre-made functions saved a considerable amount of time in developing routines for the project.

The code developed for this project has been used to test the prototype created and will provide a basis for the future models of the display and programmer. At present the routines developed have not been fully interfaced into a single software program for either device. The routines have however provided the ability to test all features of the design. In the future the routines will be combined to create one program for the display and another for the handheld programmer. A listing of all the code created for this project can be found in the appendix.

Chapter 7 Production Related Issues

As the ultimate goal from the design is for the display to be made into a viable product, some considerations have been made regarding the associated manufacturing issues and industry standards. These include:

- Legal issues in regards to mounting lights on vehicles
- UV stabilised plastic for the display casing
- Ingress protection ratings of the casing
- Electromagnetic compatibility of the design
- Modularity and easy of assembly and instillation
- Aesthetics and consumer demand.

7.1 Ultraviolet Protection for the Display Case

As the display may be mounted in locations that receive direct sunlight for the duration of its lifetime, the casing for the display will need to possess some form of resistance to the solar radiation. The expected material used for the casing of the display is plastic, which if exposed to sunlight can cause cracking, chalking, and fading. The ultraviolet (UV) radiation in sunlight causes the chemical bonds within the plastic to breakdown in a process called photodegradation. To counteract the damaging effects of UV light, substances called UV stabilisers can be added to form UV stabilised plastics. These stabilised plastics are the type of material that will be considered by Downey Engineering for use in the display casing. UV stabilisers can be classified into two categories, ultraviolet light absorbers (UVA) and hindered amine light stabilizers (HALS). HALS are an extremely efficient method of resisting light-induced degradation, achieving stabilisation of plastics at relatively low concentrations. This is due to a cyclic process whereby more HALS are regenerated in the stabilisation process rather then being consumed. Ultraviolet absorbers alternatively, function by absorbing the destructive ultraviolet radiation and dissipating it as thermal energy. The effectiveness of UV absorbers is a function of both sample thickness and stabilizer concentration. In operation, high concentrations of UV absorbers and a sufficient thickness of the polymer are required before enough absorption occurs to effectively hinder photodegradation. The choice of which type of UV stabilised plastic to use for the display ultimately rests with Downey Engineering. The need to incorporate this protective material in the design is significant and suppliers of the plastic will be investigated. Factors that will need to be considered include the purchasing costs, availability of supply and the process for moulding of the plastic.

7.2 Ingress Protection for Electronic Enclosures

The ingress protection (IP) rating of enclosures is a standard measure of how well an enclosure prevents the entry of foreign material such as dirt or water. The degrees of protection are commonly expressed as "IP" followed by two digits, for example IP66, where the digits define the degree of protection. The first digit, which represents the Foreign Bodies Protection rating, indicates the extent to which the equipment is protected against particles entering or escaping the enclosure. The second digit indicates the level of protection against water incursion. A third digit can also occur which indicates protection against mechanical impacts, however it is often omitted when there is no protection from impacts associated with the item. The complete standard is specified in Australian Standards AS1939 and International Standard EN60529. The following tables show the parameters of the ingress material at each level and the conditions for which each may occur.

Table 7.1 Foreign Bodies Protection Index for IP Ratings

First Index - Foreign Bodies Protection, Solids		
Index	Protection against Human/Tool Contact	Protection against solid objects (foreign bodies)
0	No special protection	
1	Back of hand, Fist	Large foreign bodies, diam. >50mm
2	Finger	Medium-sized foreign bodies, diam. >12
3	Tools and wires etc with a thickness >2.5mm	Small foreign bodies, diam. >2.5mm
4	Tools and wires etc with a thickness >1mm	Granular foreign bodies, diam. >1mm
5	Complete protection, (limited ingress permitted)	Dust protected; dust deposits permitted, but must not affect the function of the unit.
6	Complete protection	Dust-proof

Table 7.2 Water Protection Index for IP Ratings

Second Index - Water Protection, Liquids		
Index	Protection against water	Protection condition
0	No special protection	
1	Water dripping/falling vertically	Condensation/Light rain
2	Water sprayed at an angle (up to 15° degrees from the vertical)	Light rain with wind
3	Spray water (any direction up to 60° degrees from the vertical)	Heavy rainstorm
4	Spray water from all directions, (limited ingress)	Splashing
5	Low pressure water jets from all directions, (limited ingress)	Hose down, residential
6	High pressure jets from all directions, (limited ingress permitted)	Hose down, commercial.
7	Temporary immersion, 15 cm to 1m	Immersion in tank
8	Permanent Immersion, under pressure	Full immersion

Table 7.3 Impact Protection Index for IP Ratings

Third Index - Impact Protection, Impact		
Index	Protection against impact	Equivalent mass impact
0	No special protection	
1	Protected against 0.225J impact	eg. 150g weight falling from 15cm height
2	Protected against 0.375J impact	eg. 250g weight falling from 15cm height
3	Protected against 0.5J impact	eg. 250g weight falling from 20cm height
4	Protected against 2.0J impact	eg. 500g weight falling from 40cm height
5	Protected against 6.0J impact	eg. 0.61183kg weight falling from 1m height
6	Protected against 20.0J impact	eg. 2.0394kg weight falling from 1m height

There are two IP ratings that will be considered for the display enclosure when manufactured. The first model would only be used in indoor applications with no opportunity for the display to get wet. This model's casing would be designed to meet the IP40 rating with protection from material greater than 1mm but no protection against water entering the device. The second model would be rated at the higher IP66 level, providing complete protection from dust entering the device and resistance to high-pressure jets of water such as a hose down. This model would be suited to industrial applications where the display may meet more severe conditions.

7.3 Electro-Magnetic Compatibility

When designing the circuitry for the display an important contributing factor was the protection of the display from electromagnetic interference generated in its working environment and also the prevention of producing interference. Electromagnetic interference, EMI, is radiated or conducted energy that adversely affects circuit performance disrupting a device's normal operation. When installed in an automotive or industrial setting, considerations must be made to prevent EMI from corrupting the digital processes within the display. In addition to stopping external interference from entering the device care was also taken to prevent the display from emitting interference. The methods used to prevent the acceptance and emission of electromagnetic interference included:

- Filter capacitors on the power rail inputs to integrated circuits to prevent voltage spikes corrupting logic
- Filters on both the linear and switching power supplies to prevent incoming or outgoing interference from occurring on the supply lines.
- Shielding on the serial connection cable to the display
- Short track lengths on the printed circuit board to in further designs reduce electromagnetic coupling.

When the next prototype is created, which is not covered in the scope of this project, testing will need to be done to determine the severity of interference that automotive vehicles produce from such devices as the electronic fuel injection system. From these tests a decision can be made as to whether the EMC protection in the current design is adequate or if more protection is needed. The problem of radiated interference is unlikely however, as the display is to be mounted in the rear window, which is sufficiently far from sources of interference located around the vehicles engine. Any conducted interference that enters the circuit from the power supply line should also be minimal due to the power supply filters.

7.4 Modularity and Ease of Assembly

The design layout for the display needs be carefully thought through to ensure that the assembly is uncomplicated and provides easy access to adjustable parts. Consciously placing components to provide easy access for installation and replacement of components if necessary, will improve the efficiency of the manufacturing and allow adjustments to the prototype to be made. A specific example of this is that a range of colour LEDs may be used in different models of the display. As the different coloured LEDs require different currents, it is important to place the current limiting resistors of each display in an accessible position on the printed circuit board (PCB) so they can be easily exchanged.

Modularity of the PCB design is also an important factor to incorporate in the design layout to facilitate more efficient and economical construction. A PCB design of the display system is currently being produced for the second prototype. The primary circuitry that has been modularised in the display is the combination of a shift register, current driver and one LED module. By grouping one or more of these component subunits together, a PCB design can be created and easily reproduced for manufacturing. These PCB modules will then be linked via ribbon cable in a cascaded manner allowing the display to be extended or truncated where necessary. This feature makes for a very flexible design that can be adapted to suit specific display lengths.

7.5 Aesthetics and Consumer Demand

In addition to a product that operates effectively, an important quality of the display is that it must be aesthetically pleasing to consumers. To become a commercially viable product the display must be eye-catching to both consumers of the display and their target audience. Following this project a larger second prototype will be installed in publicly viewable locations in an attempt to test the marketability of the display. A website or phone number will be advertised on the display in an attempt to obtain the public’s response and gauge how well the product is received.

7.6 Legal Issues

While considering characteristics of the display, issues arose regarding the legal implications of mounting an illuminated display in an automotive vehicle. Extensive research was undertaken to find the Australian laws regarding mounting lights in vehicles specifically the Transport Operations Regulation 1999 (Road Use Management – Vehicle Standards and Safety). A section of interest Part 7, Lights and Reflectors, Division 19 Other Lights Reflectors or Signals subsection 5(a) states:

> *"A vehicle other than an exempt vehicle, must not be fitted with a light or reflector that shows red light to the front."*

This clause clearly states that the red display cannot be mounted facing out the front of the car but can be mount facing the back. Division 8, Unauthorised Lights, subsection 1 declares:

> *"A person must not fit a light or reflector unless the light or reflector is required to be fitted to the vehicle or is optional equipment for the vehicle…".*

Section 2 of the same passage adds:

> *"However a person does not contravene subsection 1 if the person reasonably believes the vehicle is not to be used on a road."*

From these two sections it can be concluded that the display can only be fitted in a vehicle which is not being used on the road. This implies that the display must not be used while the vehicle is driven on the road. It can be used however, if the vehicle is stationary or parked away from the road. These laws still allow the display to be used in applications such as a breakdown warning sign or advertising, however the device should not be used whilst the vehicle is being used on a road. This law is restrictive to some extent, preventing users from using the display while in transit however it can still be used if the vehicle is not currently being driven.

Chapter 8 Conclusion

8.1 Evaluation of the Design and Prototype

Tests performed on the prototype created verified that the circuit design functioned correctly. The prototype unit was used to display pre-programmed messages on the smaller LED modules and the programmer was capable of creating and sending user defined messages across the serial connection. Some simple textual animations such as scrolling and flashing where also created for the display and can be found in the appendix. The set goals within this project have been achieved, proving the feasibility of the display thus far. In the near future, further testing will need to be done on the additional prototypes that are to be produced. Routines created in this project will also be combined into independent programs for the display and programmer prototypes. These prototypes will more closely resemble the final product, as they will be made on printed circuit boards.

8.2 Future Developments

In the future a second full-scale prototype will be developed using larger LED modules. The printed circuit board for this second trial is currently being designed using Protel, a printed circuit board design environment. This second prototype will be used to assess the marketability of the device by demonstrating the display in a public location. The PCB design, a full listing of components used and a copy of this document will then be provided to Downey Engineering where a decision on the subsequent development of the display will be made. Future work that may be required includes the design of a housing for the display system via 3D modelling and the design of a software package for editing and downloading pre-programmed messages from a PC to the display system. These tasks were part of the "if time permits" section of the project specification and could not completed due to time restrictions. Concepts that may be included in further models of the display include:

- An infrared link between the handheld programmer and the display unit to allow messages to be created at a distance
- Incorporating multicolour LEDs in the display
- In built time and date features for a digital clock function
- Increased animation design for the text display modes

8.3 Achievement of Objectives

The research relating to the design of LED displays and microcontroller-based circuits was investigated throughout this project. This knowledge provided better methods for approaching the task while helping to gain an understanding of microcontroller systems that will be useful for future work. In addition to this, an evaluation of existing displays available was undertaken and extra features that could be implemented in further designs were found.

The design for a display that can be mounted in a vehicle and the design of the accompanying handheld controller was completed. This design was then implemented in the initial prototype for the system, to verify its physical viability and to investigate any possible improvements. This document details the process used in creating the design, detailing how it functions and suggests future improvements. The document forms the working specification for the initial prototype and a basis from which a viable product can develop in the future.

References

Adaptive Micro Systems [Online] Available: http://www.adaptivedisplays.com/ [Accessed 10th August 2005]

Cyganski, D. (1998) *Human Visual Persistence* [Online] Available: http://www.ece.wpi.edu/infoeng/index.html. [Accessed 26 April 2005]

How Stuff Works [Online image] Available: http://science.howstuffworks.com/led.htm [Accessed 24th August 2005]

Kingbright 2005, *Single Color Dot Matrix Display Datasheet,* [Online] Available: http://www.us.kingbright.com/default1.asp [Accessed 15th August]

Klipstein, D. *LED types by Color, Brightness, and Chemistry* [Online] Available: http://members.misty.com/don/light.html. [Accessed 15th August 2005]

Maxim 2003, *RS-232 Transceivers Datasheet*, [Online] Available: http://www.maxim-ic.com/quick_view2.cfm/qv_pk/1369 [Accessed 20 March 2005]

Microchip Technology. *PIC16F87XA Data Sheet: 28/40/44-Pin Enhanced Flash Microcontrollers* [Online] Available: http://www.microchip.com [Accessed 25th April 2005]

MikroC [Homepage of Mikroelektronika] [Online] Available: http://www.mikroelektronika.co.yu/ [Accessed 31 May 2005]

National Semiconductor 2000, *LM78XX Series Voltage Regulators Datasheet,* [Online] Available: http://www.national.com/pf/LM/LM7805C.html [Accessed 28 March 2005]

National Semiconductor 1999, *LM2576 3A Step-Down Voltage Regulator Datasheet,* [Online] Available: http://www.national.com/pf/LM/LM2576.html [Accessed 21 August 2005]

RS232 Data Interface [Online] Available: http://www.arcelect.com [Accessed 15th August 2005]

ST Microelectronics, *ULN2001A-ULN2004A Datasheet,* [Online] Available: http://focus.ti.com/docs/prod/folders/print/ulq2003a-q1.html [Accessed 28 March 2005]

Tai, D. *PICpgm Development Programmer* [Online] Available: http://members.aon.at/electronics/pic/picpgm/index.html [Accessed 21 April 2005]

Vorne Visual Display & Productivity Tools [Online] Available: http://www.vorne.com/ [Accessed 10th August 2005]

Appendix A

Project Specification

University of Southern Queensland
Faulty of Engineering and Surveying

ENG 4111/4112 Research Project

PROJECT SPECIFICATION

FOR:	**Michael Costa**
TOPIC:	In-Vehicle Display System
SUPERVISORS:	Mark Phythian Ross Leamon, Downey Engineering
SPONSORSHIP:	Downey Engineering, 3 Progress Court Toowoomba Ph: 46394266
PROJECT AIM:	The aim of the project involves the development of an in-vehicle textual display system based on an LED dot matrix. This project involves taking the device from concept through design, to a near market ready product.
PROGRAMME:	Issue A, 15 March 2005

1. Research information relating to the design of LED moving message displays and microcontroller based circuits.
2. Design a microcontroller driven, marquee type display to be mounted in a vehicle.
3. Design a handheld interface to the display system for calling up and programming messages.
4. Design software to provide numerous preprogrammed messages, animations and user defined messages for the display system.
5. Develop a full set of working specifications of the display system to suit the technical / budget limitations of the project.
6. Construct a prototype of the display system mentioned in 2 and 3 above and evaluate.

As time permits

7. Design of housing for the display system via 3D modeling
8. Design a software package for PC designed for editing / downloading preprogrammed messages to the display system

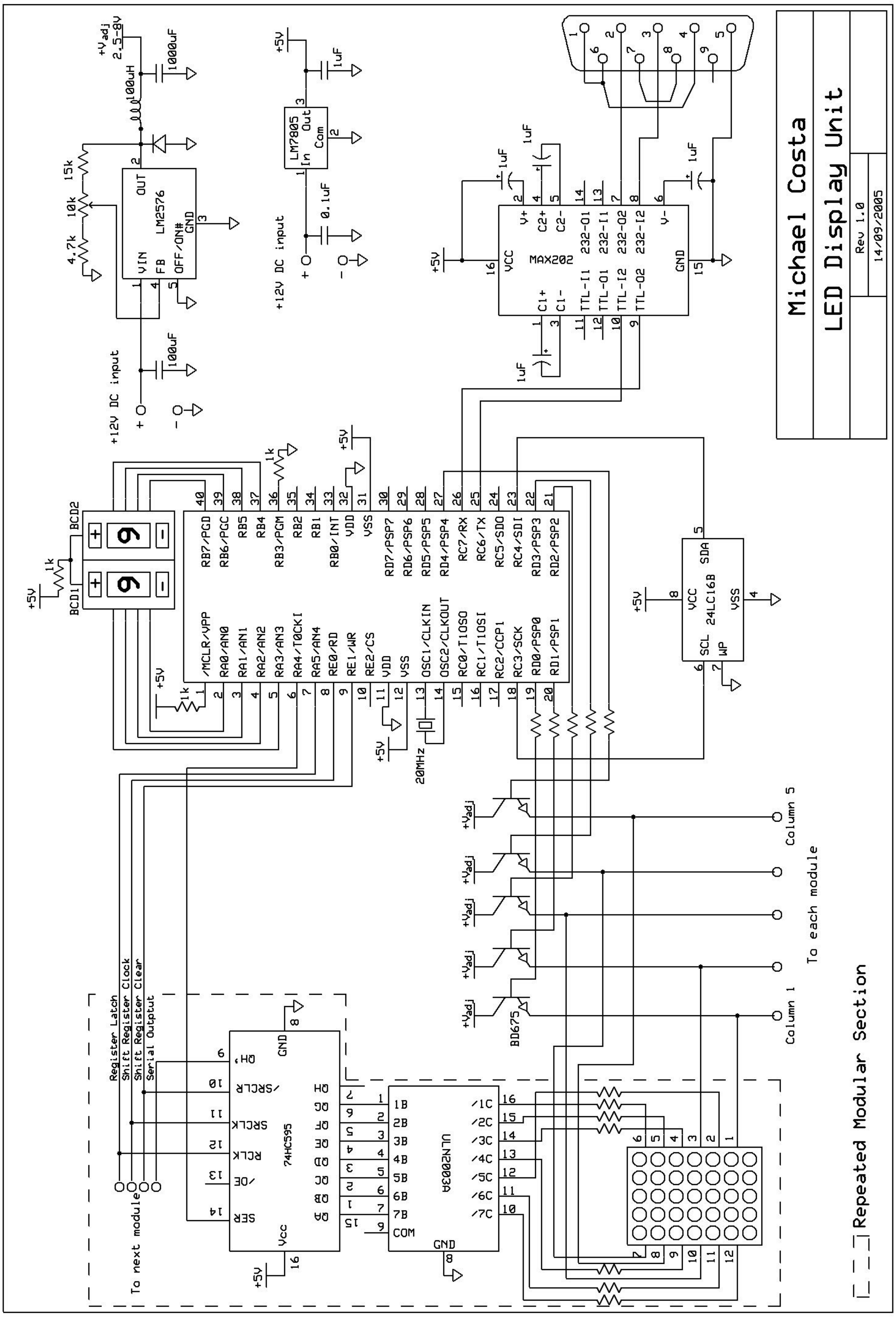
AGREED:

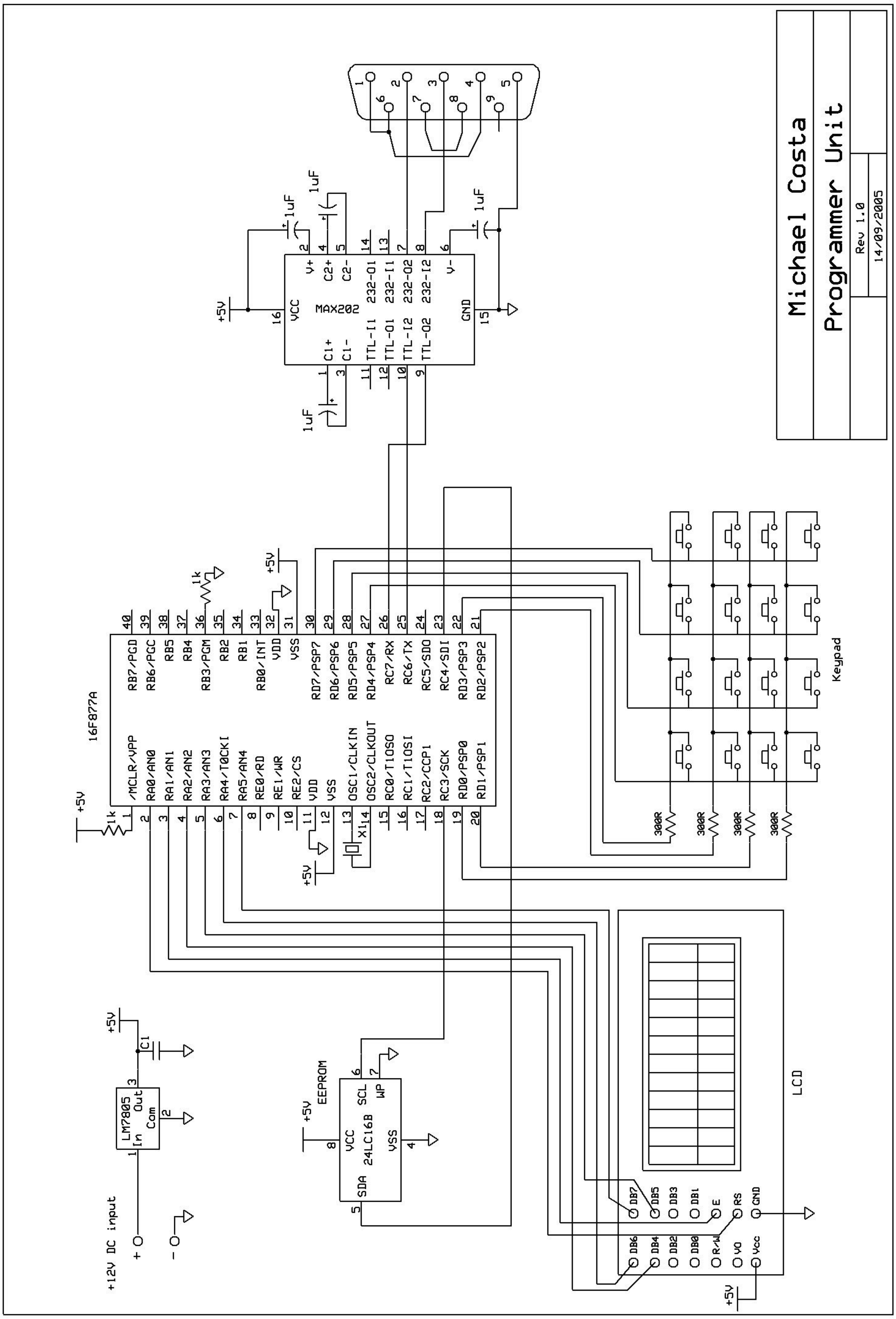
___________________ (Student)   ___________________ (Supervisor)

(dated) _____/_____/_____

Appendix B

Circuit Schematics


Michael Costa
LED Display Unit
Rev 1.0
14/09/2005
+12V DC input
+Vadj 2.5-8V
100uH
1000uF
100uF
15k
10k
4.7k
LM2576
VIN
FB
OFF/ON#
OUT
GND
+5V
LM7805
In
Out
Com
1uF
0.1uF
MAX202
VCC
V+
V-
C1+
C1-
C2+
C2-
TTL-I1
TTL-O1
TTL-I2
TTL-O2
232-O1
232-I1
232-O2
232-I2
GND
BCD1
BCD2
1k
/MCLR/VPP
RA0/AN0
RA1/AN1
RA2/AN2
RA3/AN3
RA4/T0CKI
RA5/AN4
RE0/RD
RE1/WR
RE2/CS
VDD
VSS
OSC1/CLKIN
OSC2/CLKOUT
RC0/T1OSO
RC1/T1OSI
RC2/CCP1
RC3/SCK
RD0/PSP0
RD1/PSP1
RB7/PGD
RB6/PGC
RB5
RB4
RB3/PGM
RB2
RB1
RB0/INT
RD7/PSP7
RD6/PSP6
RD5/PSP5
RD4/PSP4
RC7/RX
RC6/TX
RC5/SDO
RC4/SDI
RD3/PSP3
RD2/PSP2
20MHz
24LC16B
SDA
SCL
WP
+Vadj
BD675
Column 1
Column 5
To each module
Register Latch
Shift Register Clock
Shift Register Clear
Serial Output
To next module
74HC595
QH'
/SRCLR
SRCLK
RCLK
/OE
SER
Vcc
QA
QB
QC
QD
QE
QF
QG
QH
ULN2003A
1B
2B
3B
4B
5B
6B
7B
COM
/1C
/2C
/3C
/4C
/5C
/6C
/7C
Repeated Modular Section


Michael Costa
Programmer Unit
Rev 1.0
14/09/2005
16F877A
MAX202
EEPROM
24LC16B
LM7805
In
Out
Com
C1
LCD
Keypad
+12V DC input
+5V
1k
300R
1uF
X1
VCC
GND
SCL
SDA
WP
VSS
/MCLR/VPP
RA0/AN0
RA1/AN1
RA2/AN2
RA3/AN3
RA4/T0CKI
RA5/AN4
RE0/RD
RE1/WR
RE2/CS
VDD
OSC1/CLKIN
OSC2/CLKOUT
RC0/T1OSO
RC1/T1OSI
RC2/CCP1
RC3/SCK
RD0/PSP0
RD1/PSP1
RB7/PGD
RB6/PGC
RB5
RB4
RB3/PGM
RB2
RB1
RB0/INT
RD7/PSP7
RD6/PSP6
RD5/PSP5
RD4/PSP4
RC7/RX
RC6/TX
RC5/SDO
RC4/SDI
RD3/PSP3
RD2/PSP2
C1+
C1-
C2+
C2-
V+
V-
TTL-I1
TTL-O1
TTL-I2
TTL-O2
232-O1
232-I1
232-O2
232-I2
DB7
DB5
DB3
DB1
E
RS
DB6
DB4
DB2
DB0
R/W
V0
Vcc

Appendix C

Code Listing

This appendix contains the C code developed for the PIC16F877A that was used to test sections of the design. The code was developed using the mikroC, a C compiler by mikroElektronika, and incorporates some of the inbuilt functions of the software.

```
/*
Name: characterdisplay.c
Purpose: Scrolls through the entire character map outputting 11 characters at a time
to the display
Aurthor: Michael Costa 2005*/


//CHARACTER MAP
const char charTable[]={0B00000000, 0B00000000, 0B00000000, 0B00000000,
0B00000000,//space
0B00000000, 0B00000000, 0B01111001, 0B00000000, 0B00000000,     //exclamation_mark
0B00000000, 0B01110000, 0B00000000, 0B01110000, 0B00000000,     //double_quotes
0B00010100, 0B01111111, 0B00010100, 0B01111111, 0B00010100,     //hash
0B00010010, 0B00101010, 0B01111111, 0B00101010, 0B00100100,     //dollar_sign
0B01100010, 0B01100100, 0B00001000, 0B00010011, 0B00100011,     //percent
0B00110110, 0B01001001, 0B01010101, 0B00100010, 0B00000101,     //ampersand
0B00000000, 0B01010000, 0B01100000, 0B00000000, 0B00000000,     //apostrophe
0B00000000, 0B00011100, 0B00100010, 0B01000001, 0B00000000,     //left_parenthesis
0B00000000, 0B01000001, 0B00100010, 0B00011100, 0B00000000,     //right_parenthesis
0B00010100, 0B00001000, 0B00111110, 0B00001000, 0B00010100,     //asterisk
0B00001000, 0B00001000, 0B00111110, 0B00001000, 0B00001000,     //plus
0B00000000, 0B00000101, 0B00000110, 0B00000000, 0B00000000,     //comma
0B00001000, 0B00001000, 0B00001000, 0B00001000, 0B00001000,     //minus
0B00000000, 0B00000011, 0B00000011, 0B00000000, 0B00000000,     //full_stop
0B00000010, 0B00000100, 0B00001000, 0B00010000, 0B00100000,     //forward_slash
0B00111110, 0B01000101, 0B01001001, 0B01010001, 0B00111110,     //zero
0B00000000, 0B00100001, 0B01111111, 0B00000001, 0B00000000,     //one
0B00100001, 0B01000011, 0B01000101, 0B01001001, 0B00110001,     //two
0B01000010, 0B01000001, 0B01010001, 0B01101001, 0B01000110,     //three
0B00001100, 0B00010100, 0B00100100, 0B01111111, 0B00000100,     //four
0B01110010, 0B01010001, 0B01010001, 0B01010001, 0B01001110,     //five
0B00011110, 0B00101001, 0B01001001, 0B01001001, 0B00000110,     /six
0B01000000, 0B01000111, 0B01001000, 0B01010000, 0B01100000,     //seven
0B00110110, 0B01001001, 0B01001001, 0B01001001, 0B00110110,     //eight
0B00110000, 0B01001001, 0B01001001, 0B01001010, 0B00111100,     //nine
0B00000000, 0B00110110, 0B00110110, 0B00000000, 0B00000000,     //colon
0B00000000, 0B00110101, 0B00110110, 0B00000000, 0B00000000,     //semicolon
0B00001000, 0B00010100, 0B00100010, 0B01000001, 0B00000000,     //less_than
0B00010100, 0B00010100, 0B00010100, 0B00010100, 0B00010100,     //equals
0B00000000, 0B01000001, 0B00100010, 0B00010100, 0B00001000,     //greater_than
0B00100000, 0B01000000, 0B01000101, 0B01001000, 0B00110000,     //question_mark
0B00100110, 0B01001001, 0B01001111, 0B01000001, 0B00111110,     //at
0B00111111, 0B01000100, 0B01000100, 0B01000100, 0B00111111,     //A
0B01111111, 0B01001001, 0B01001001, 0B01001001, 0B00110110,     //B
0B00111110, 0B01000001, 0B01000001, 0B01000001, 0B00100010,     //C
0B01111111, 0B01000001, 0B01000001, 0B00100010, 0B00011100,     //D
0B01111111, 0B01001001, 0B01001001, 0B01001001, 0B01000001,     //E
0B01111111, 0B01001000, 0B01001000, 0B01001000, 0B01000000,     //F
0B00111110, 0B01000001, 0B01001001, 0B01001001, 0B00101111,     //G
```

```
0B01111111, 0B00001000, 0B00001000, 0B00001000, 0B01111111,     //H
0B00000000, 0B01000001, 0B01111111, 0B01000001, 0B00000000,     //I
0B00000010, 0B00000001, 0B01000001, 0B01111110, 0B01000000,     //J
0B01111111, 0B00001000, 0B00010100, 0B00100010, 0B01000001,     //K
0B01111111, 0B00000001, 0B00000001, 0B00000001, 0B00000001,     //L
0B01111111, 0B00100000, 0B00011000, 0B00100000, 0B01111111,     //M
0B01111111, 0B00010000, 0B00001000, 0B00000100, 0B01111111,     //N
0B00111110, 0B01000001, 0B01000001, 0B01000001, 0B00111110,     //O
0B01111111, 0B01001000, 0B01001000, 0B01001000, 0B00110000,     //P
0B00111110, 0B01000001, 0B01000101, 0B01000010, 0B00111101,     //Q
0B01111111, 0B01001000, 0B01001100, 0B01001010, 0B00110001,     //R
0B00110001, 0B01001001, 0B01001001, 0B01001001, 0B01000110,     //S
0B01000000, 0B01000000, 0B01111111, 0B01000000, 0B01000000,     //T
0B01111110, 0B00000001, 0B00000001, 0B00000001, 0B01111110,     //U
0B01111100, 0B00000010, 0B00000001, 0B00000010, 0B01111100,     //V
0B01111110, 0B00000001, 0B00001110, 0B00000001, 0B01111110,     //W
0B01100011, 0B00010100, 0B00001000, 0B00010100, 0B01100011,     //X
0B01110000, 0B00001000, 0B00000111, 0B00001000, 0B01110000,     //Y
0B01000011, 0B01000101, 0B01001001, 0B01010001, 0B01100001,     //Z
0B00000000, 0B01111111, 0B01000001, 0B01000001, 0B00000000,     //left_square
0B00100000, 0B00010000, 0B00001000, 0B00000100, 0B00000010,     //back_slash
0B00000000, 0B01000001, 0B01000001, 0B01111111, 0B00000000,     //right_square
0B00010000, 0B00100000, 0B01000000, 0B00100000, 0B00010000,     //circumflex
0B00000001, 0B00000001, 0B00000001, 0B00000001, 0B00000001,     //underscore
0B00000000, 0B01000000, 0B00100000, 0B00010000, 0B00000000,     //grave_accent
0B00000010, 0B00010101, 0B00010101, 0B00010101, 0B00001111,     //a
0B01111111, 0B00001001, 0B00001001, 0B00001001, 0B00000110,     //b
0B00001110, 0B00010001, 0B00010001, 0B00010001, 0B00010001,     //c
0B00000110, 0B00001001, 0B00001001, 0B00001001, 0B01111111,     //d
0B00001110, 0B00010101, 0B00010101, 0B00010101, 0B00001101,     //e
0B00000000, 0B00001000, 0B00111111, 0B01001000, 0B00100000,     //f
0B00001001, 0B00010101, 0B00010101, 0B00010101, 0B00011110,     //g
0B01111111, 0B00001000, 0B00001000, 0B00001000, 0B00000111,     //h
0B00000000, 0B00000000, 0B00101111, 0B00000000, 0B00000000,     //i
0B00000010, 0B00000001, 0B00000001, 0B01011110, 0B00000000,     //j
0B00000000, 0B01111111, 0B00000100, 0B00001010, 0B00010001,     //k
0B00000000, 0B01000001, 0B01111111, 0B00000001, 0B00000000,     //l
0B00011111, 0B00010000, 0B00001110, 0B00010000, 0B00011111,     //m
0B00011111, 0B00001000, 0B00010000, 0B00010000, 0B00001111,     //n
0B00001110, 0B00010001, 0B00010001, 0B00010001, 0B00001110,     //o
0B00011111, 0B00010100, 0B00010100, 0B00010100, 0B00001000,     //p
0B00001000, 0B00010100, 0B00010100, 0B00010100, 0B00011111,     //q
0B00011111, 0B00001000, 0B00010000, 0B00010000, 0B00001000,     //r
0B00001001, 0B00010101, 0B00010101, 0B00010101, 0B00010010,     //s
0B00010000, 0B00010000, 0B01111110, 0B00010001, 0B00010010,     //t
0B00011110, 0B00000001, 0B00000001, 0B00000001, 0B00011110,     //u
0B00011100, 0B00000010, 0B00000001, 0B00000010, 0B00011100,     //v
0B00011110, 0B00000001, 0B00000110, 0B00000001, 0B00011110,     //w
0B00010001, 0B00001010, 0B00000100, 0B00001010, 0B00010001,     //x
```

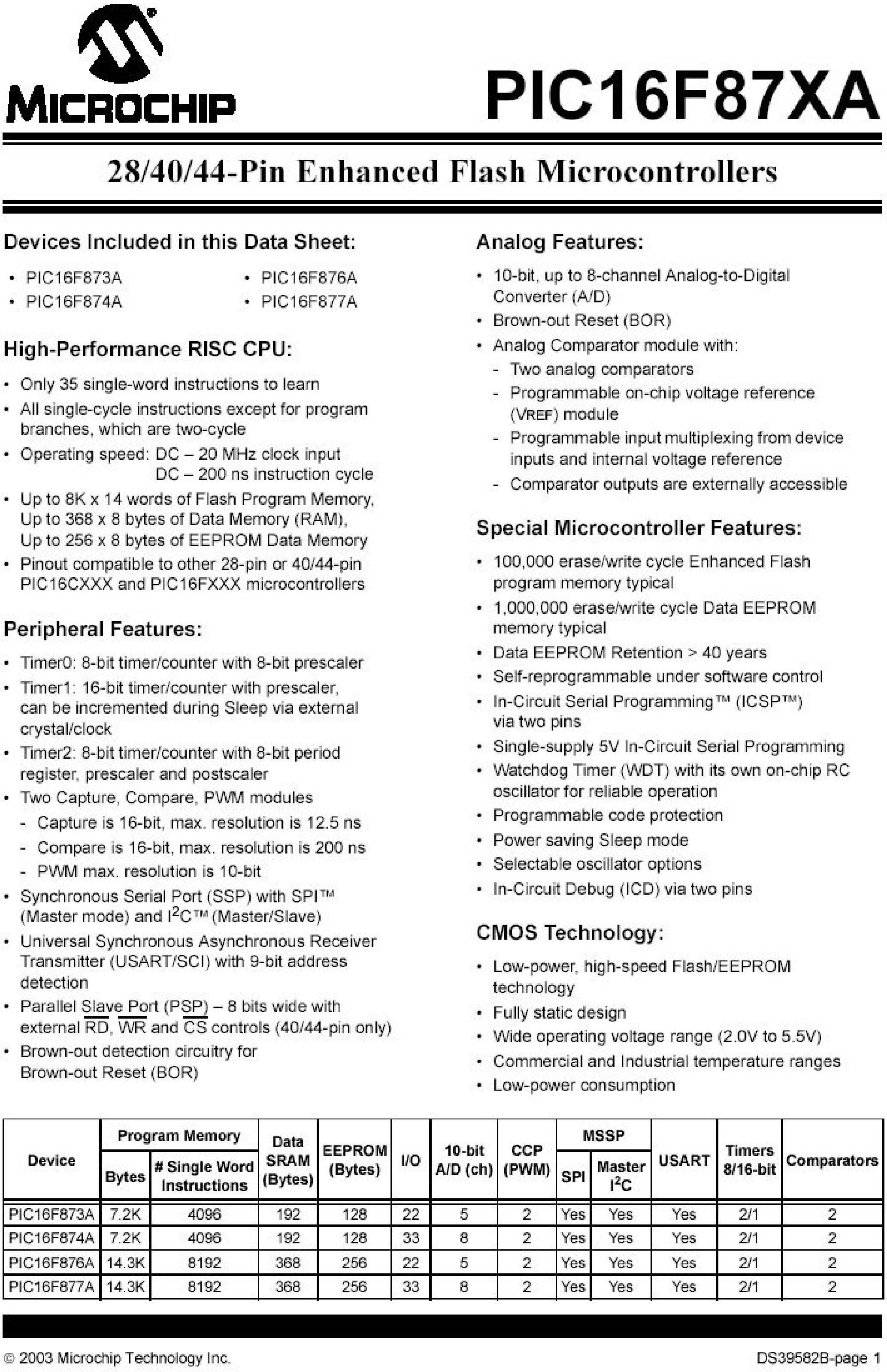
```
0B00010000, 0B00001001, 0B00000110, 0B00001000, 0B00010000,     //y
0B00010001, 0B00010011, 0B00010101, 0B00011001, 0B00010001,     //z
0B00001000, 0B00110110, 0B01000001, 0B01000001, 0B00000000,     //leftcurly
0B00000000, 0B00000000, 0B01111111, 0B00000000, 0B00000000,     //vertline
0B00000000, 0B01000001, 0B01000001, 0B00110110, 0B00001000,     //rightcurly
0B00000100, 0B00001000, 0B00001000, 0B00001000, 0B00010000,     //tilde
0B01111111, 0B01000001, 0B01000001, 0B01000001, 0B01111111};    //del

void output_char(int i,char character); // character output function
void output_last(int i,char character); // last character output function


int main(){
     int i=0, j=0,k=0, currentBit=0; // loop counters
     char character;
    TRISA = 0; //PORTA is output control lines for shift registers
    TRISD = 0; // PORTD is output lines & pin 5 is serial reset
    PORTA = 0; // order of PORTA bits: 0=data 1=latch 2=clock,
    PORTD=0;  //order of PORTD bits: 0=LED column1,1=LED column2 etc. 5=shift reg.
reset

 while(1){
      for(j=0;j<85;j++){ //loop through all characters
       for(k=0;k<50;k++){ // loop each character 50 times
        for(i=0;i<5;i++){   //5 bits bytes per character

               PORTD=0B100000; //reset off
               // 11 characters displayed on prototype
                character=(42+j); output_char(i,character);
                character=(41+j); output_char(i,character);
                character=(40+j); output_char(i,character);
                character=(39+j); output_char(i,character);
                character=(38+j); output_char(i,character);
                character=(37+j); output_char(i,character);
                character=(36+j); output_char(i,character);
                character=(35+j); output_char(i,character);
                character=(34+j); output_char(i,character);
                character=(33+j); output_char(i,character);
                character=(32+j); output_last(i,character);

                PORTD =(1<<i)|(0B100000) ;// Turn ON diodes on PORTD ith column
                Delay_ms(1);// LED on time
                PORTD = 0;   //Turn OFF diodes on PORTD & SERIAL RESET

                }
            }
        }
    }
```

```
}

void output_char(int i, char character) //,char *character){
     int currentBit,j,data,address;
     address=(unsigned int)((character-32)*5+0x01C1+i);
//NB: 01C1 is a hex addr of the start of charTable gotten from the .asm file
     data=Flash_Read(address);
     for(j=0;j<8;j++){  // data is 8 bits high
                currentBit=((data>>j)%2)|(0B00000010);
               // a[i]>>j bitshifts a right j th digit
               //%2 finds isolates last bit of binary digit
               // OR to set clock bit
                PORTA =currentBit; //set DATA & CLOCK HIGH
                PORTA=0; //RESET CLOCK LOW
                PORTA = 0B0100; // set LATCH pin
     }
}
void output_last(int i,char character){
     int currentBit,j,data,address;
     address=(unsigned int)((character-32)*5+0x01C1+i);
     data=Flash_Read(address);
     for(j=0;j<7;j++){  // 7 bits high
                currentBit=((data>>j)%2)|(0B00000010);
               // a[i]>>j bitshifts a right nth digit
               //%2 finds isolates last bit of binary digit
               // OR to set clock bit
                PORTA =currentBit; //set DATA & CLOCK HIGH
                PORTA=0; //RESET CLOCK LOW
                PORTA = 0B0100; // set LATCH pin
     }
}
/////////////////////////////////////////////EOF/////////////////////////////////////

/*
Name: EEPROM_read
Purpose: A routine developed to access messages stored in an external EEPROM
connected to the PIC16F877A
Aurthor: Michael Costa 2005*/

char EEPROM_read(int addr){
 char character;
 PORTC = 0;
 TRISC = 0;
 I2C_Init(100000);
 I2C_Start();              // issue I2C start signal
 I2C_Wr(0xA2);             // send byte via I2C  (device address + W)
 I2C_Wr(addr);                // send byte (data address)
 I2C_Repeated_Start();     // issue I2C signal repeated start
 I2C_Wr(0xA3);             // send byte (device address + R)
 PORTB = I2C_Rd(0u);      // Read the data (NO acknowledge)
 I2C_Stop();               // issue I2C stop signal
 return (character)
}
/////////////////////////////////////////////EOF////////////////////////////////////

Name: Programmer.c
Purpose: A routine to create messages on the handheld programmer using multi-tap
typing and display the result on the LCD.
Aurthor: Michael Costa 2005*/

char message[16];
int maxLength=16, charCount=0, pressCount=0, kp=0, oldkp=-1;
void main() {

  Keypad_Init(&PORTC);     //initialise keypad
  Lcd_Init(&PORTD);        //initialise LCD

  Lcd_Cmd(Lcd_CLEAR);       // Clear display
  Lcd_Cmd(Lcd_CURSOR_OFF);  // Turn cursor off
  Lcd_Out(1, 1, "Enter message :"); //First line of LCD
  Lcd_Cmd(LCD_SECOND_ROW);    // Move cursor to second line
  Lcd_Cmd(LCD_BLINK_CURSOR_ON); // Turn cursor on

 do{
     do{             //loop while waiting for input
     Delay_ms(250); // keypad debounce
     kp = Keypad_Released();
     }while (!kp);

     if(oldkp==kp){        //test for repeated press
                   pressCount++;  // flag repeated press
                   charCount--;
                   Lcd_Cmd(LCD_MOVE_CURSOR_LEFT); //write over previous character
                   }
                   else{  pressCount=0;}

       switch(kp){
     case 1:Lcd_Cmd(LCD_MOVE_CURSOR_LEFT); break;      // 1 = backspace
     case 2:Lcd_Chr_Cp('A'+pressCount);                // 2 = A,B,C
          message[charCount]=('A'+pressCount);
          charCount++; break;
     case 3:Lcd_Chr_Cp('D'+pressCount);                // 3 = D,E,F
          message[charCount]=('D'+pressCount);
          charCount++; break;
     case 4:Lcd_Chr_Cp('G'+pressCount);                // 4 = G,H,I
          message[charCount]=('G'+pressCount);
          charCount++; break;
     case 5:Lcd_Chr_Cp('J'+pressCount);                // 5 = J,K,L
```

```
          message[charCount]=('J'+pressCount);
          charCount++; break;
     case 6:Lcd_Chr_Cp('M'+pressCount);                // 6 = M,N,O
          message[charCount]=('M'+pressCount);
          charCount++; break;
     case 7:Lcd_Chr_Cp('P'+pressCount);                // 7 = P,Q,R,S
          message[charCount]=('P'+pressCount);
          charCount++; break;
     case 8:Lcd_Chr_Cp('T'+pressCount);                // 8 = T,U,V
          message[charCount]=('T'+pressCount);
          charCount++; break;
     case 9:Lcd_Chr_Cp('W'+pressCount);                // 9 W,X,Y,Z
          message[charCount]=('W'+pressCount);
          charCount++; break;
     case 10:charCount=maxLength; break;            // * = OK/END button
     case 11:Lcd_Chr_Cp('0'+pressCount);            // 0 = 0-9
          message[charCount]=('0'+pressCount);
          charCount++; break;
     case 12:Lcd_Chr_Cp('!'+pressCount);               //   #   =   other   punctuation
characters
          message[charCount]=('!'+pressCount)
          ;charCount++; break;
     default :Lcd_Out_Cp("Error");break;
     }

     oldkp=kp;
  }while(!(maxLength==charCount));

  Lcd_Out(1, 1, "                "};//clear first row
  Lcd_Out(1, 1, "DONE");
  Lcd_Cmd(LCD_CURSOR_OFF);
}
/////////////////////////////////////////////EOF/////////////////////////////////////

Name: USART.c
Purpose: To test a serial connection from a PC to the PIC16F877A
Aurthor: mikroElektronika
//A routine used to test the serial connection to a PC by reflecting a received
character

void main() {

  // Initialize USART module (8 bit, 2400 baud rate, no parity bit..)
  Usart_Init(2400);

  do {
    if (Usart_Data_Ready()) {   // If data is received
      i = Usart_Read();         // Read the received data
      Usart_Write(i);           // Send data via USART
    }
  } while (1);
}
/////////////////////////////////////////////EOF/////////////////////////////////////
////
```

Appendix D

Extracts from the PIC16F877A Microcontroller Data Sheet



MICROCHIP

PIC16F87XA

28/40/44-Pin Enhanced Flash Microcontrollers

Devices Included in this Data Sheet:

- PIC16F873A
- PIC16F874A
- PIC16F876A
- PIC16F877A

High-Performance RISC CPU:

- Only 35 single-word instructions to learn
- All single-cycle instructions except for program branches, which are two-cycle
- Operating speed: DC – 20 MHz clock input
DC – 200 ns instruction cycle
- Up to 8K x 14 words of Flash Program Memory, Up to 368 x 8 bytes of Data Memory (RAM), Up to 256 x 8 bytes of EEPROM Data Memory
- Pinout compatible to other 28-pin or 40/44-pin PIC16CXXX and PIC16FXXX microcontrollers

Peripheral Features:

- Timer0: 8-bit timer/counter with 8-bit prescaler
- Timer1: 16-bit timer/counter with prescaler, can be incremented during Sleep via external crystal/clock
- Timer2: 8-bit timer/counter with 8-bit period register, prescaler and postscaler
- Two Capture, Compare, PWM modules
  - Capture is 16-bit, max. resolution is 12.5 ns
  - Compare is 16-bit, max. resolution is 200 ns
  - PWM max. resolution is 10-bit
- Synchronous Serial Port (SSP) with SPI™ (Master mode) and I^{2}C™ (Master/Slave)
- Universal Synchronous Asynchronous Receiver Transmitter (USART/SCI) with 9-bit address detection
- Parallel Slave Port (PSP) – 8 bits wide with external $\overline{\text{RD}}$, $\overline{\text{WR}}$ and $\overline{\text{CS}}$ controls (40/44-pin only)
- Brown-out detection circuitry for Brown-out Reset (BOR)

Analog Features:

- 10-bit, up to 8-channel Analog-to-Digital Converter (A/D)
- Brown-out Reset (BOR)
- Analog Comparator module with:
  - Two analog comparators
  - Programmable on-chip voltage reference (VREF) module
  - Programmable input multiplexing from device inputs and internal voltage reference
  - Comparator outputs are externally accessible

Special Microcontroller Features:

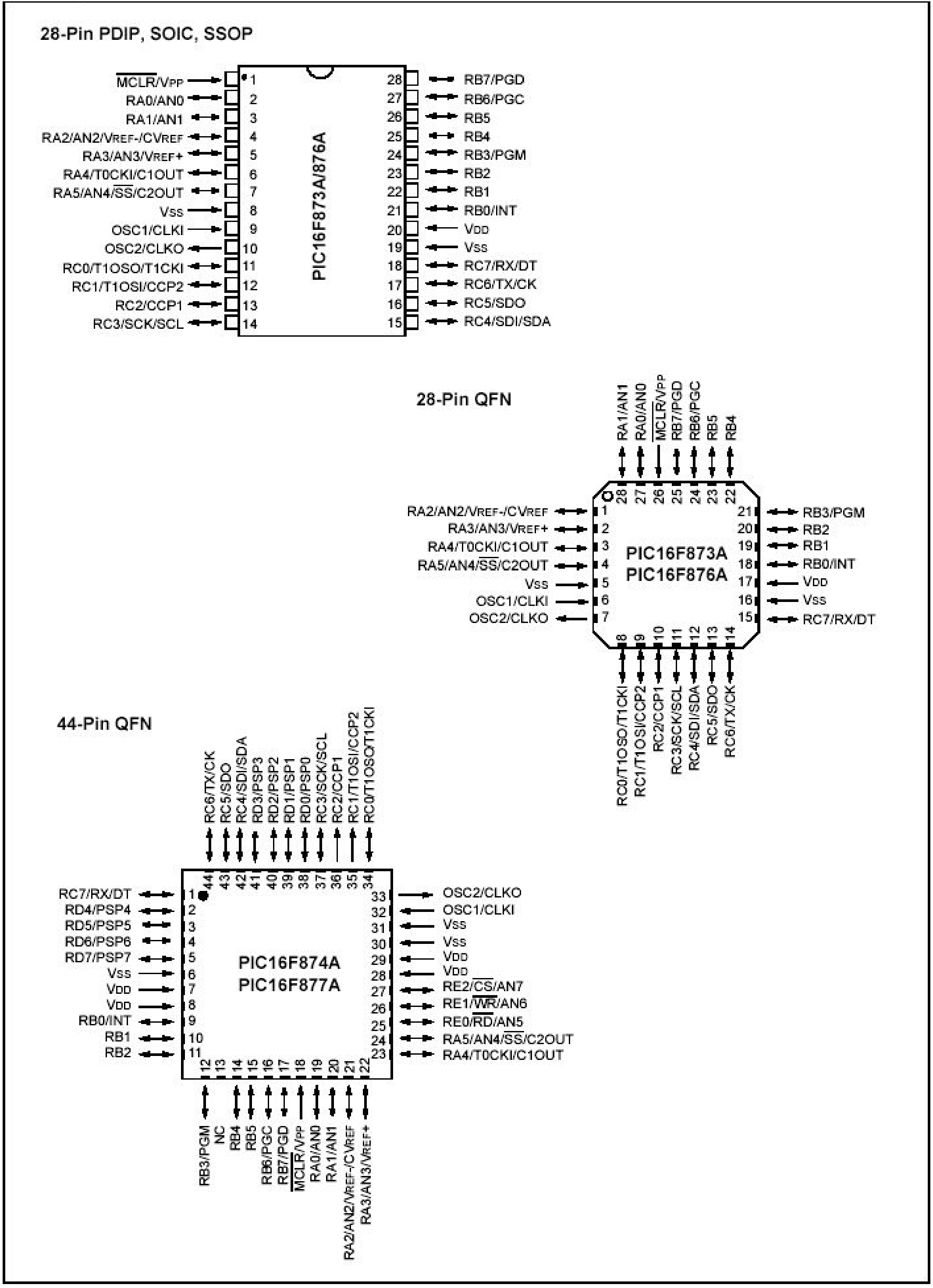
- 100,000 erase/write cycle Enhanced Flash program memory typical
- 1,000,000 erase/write cycle Data EEPROM memory typical
- Data EEPROM Retention > 40 years
- Self-reprogrammable under software control
- In-Circuit Serial Programming™ (ICSP™) via two pins
- Single-supply 5V In-Circuit Serial Programming
- Watchdog Timer (WDT) with its own on-chip RC oscillator for reliable operation
- Programmable code protection
- Power saving Sleep mode
- Selectable oscillator options
- In-Circuit Debug (ICD) via two pins

CMOS Technology:

- Low-power, high-speed Flash/EEPROM technology
- Fully static design
- Wide operating voltage range (2.0V to 5.5V)
- Commercial and Industrial temperature ranges
- Low-power consumption

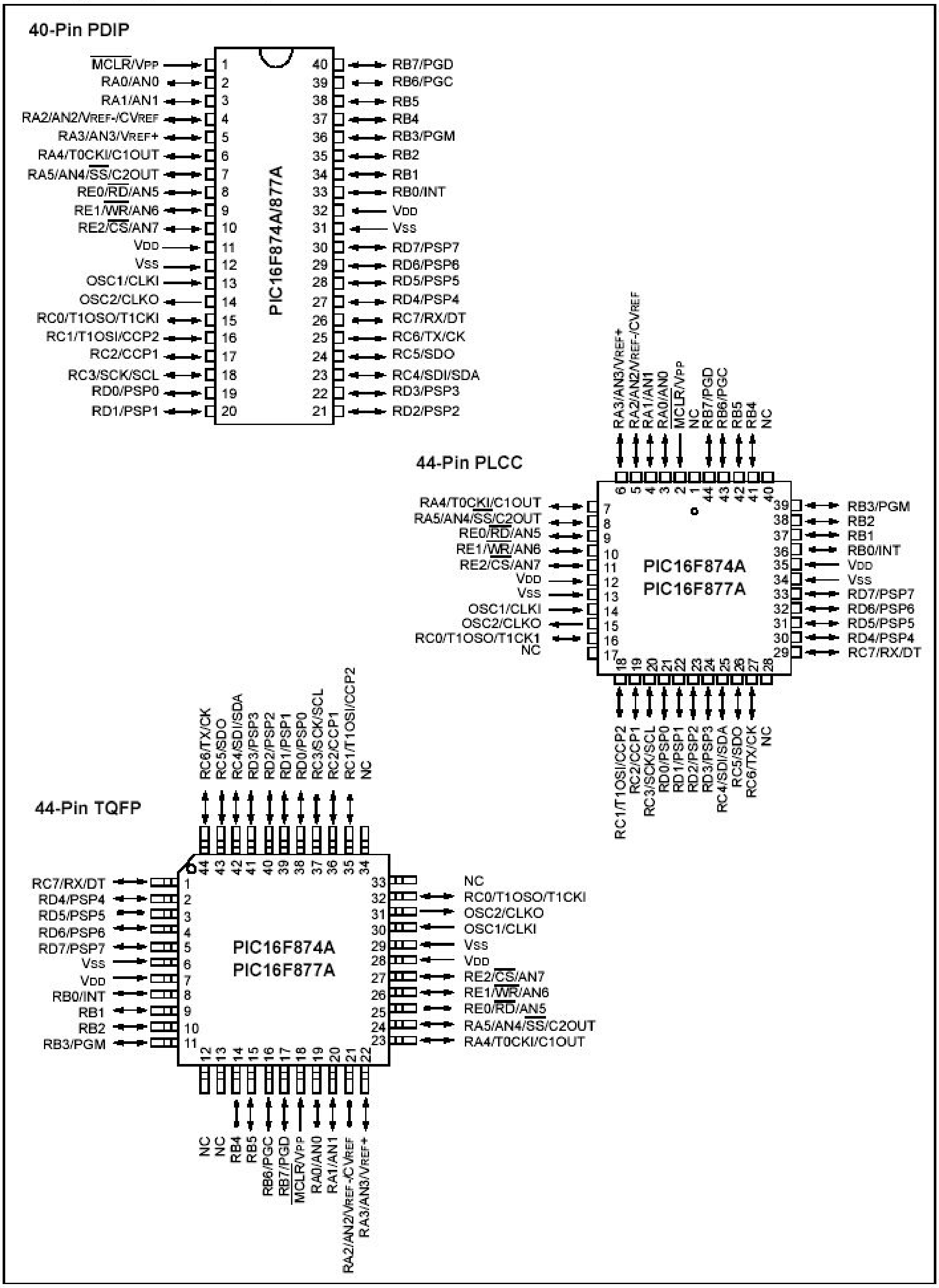
Device	Program Memory		Data SRAM (Bytes)	EEPROM (Bytes)	I/O	10-bit A/D (ch)	CCP (PWM)	MSSP		USART	Timers 8/16-bit	Comparators
	Bytes	# Single Word Instructions						SPI	Master I^{2}C			
PIC16F873A	7.2K	4096	192	128	22	5	2	Yes	Yes	Yes	2/1	2
PIC16F874A	7.2K	4096	192	128	33	8	2	Yes	Yes	Yes	2/1	2
PIC16F876A	14.3K	8192	368	256	22	5	2	Yes	Yes	Yes	2/1	2
PIC16F877A	14.3K	8192	368	256	33	8	2	Yes	Yes	Yes	2/1	2

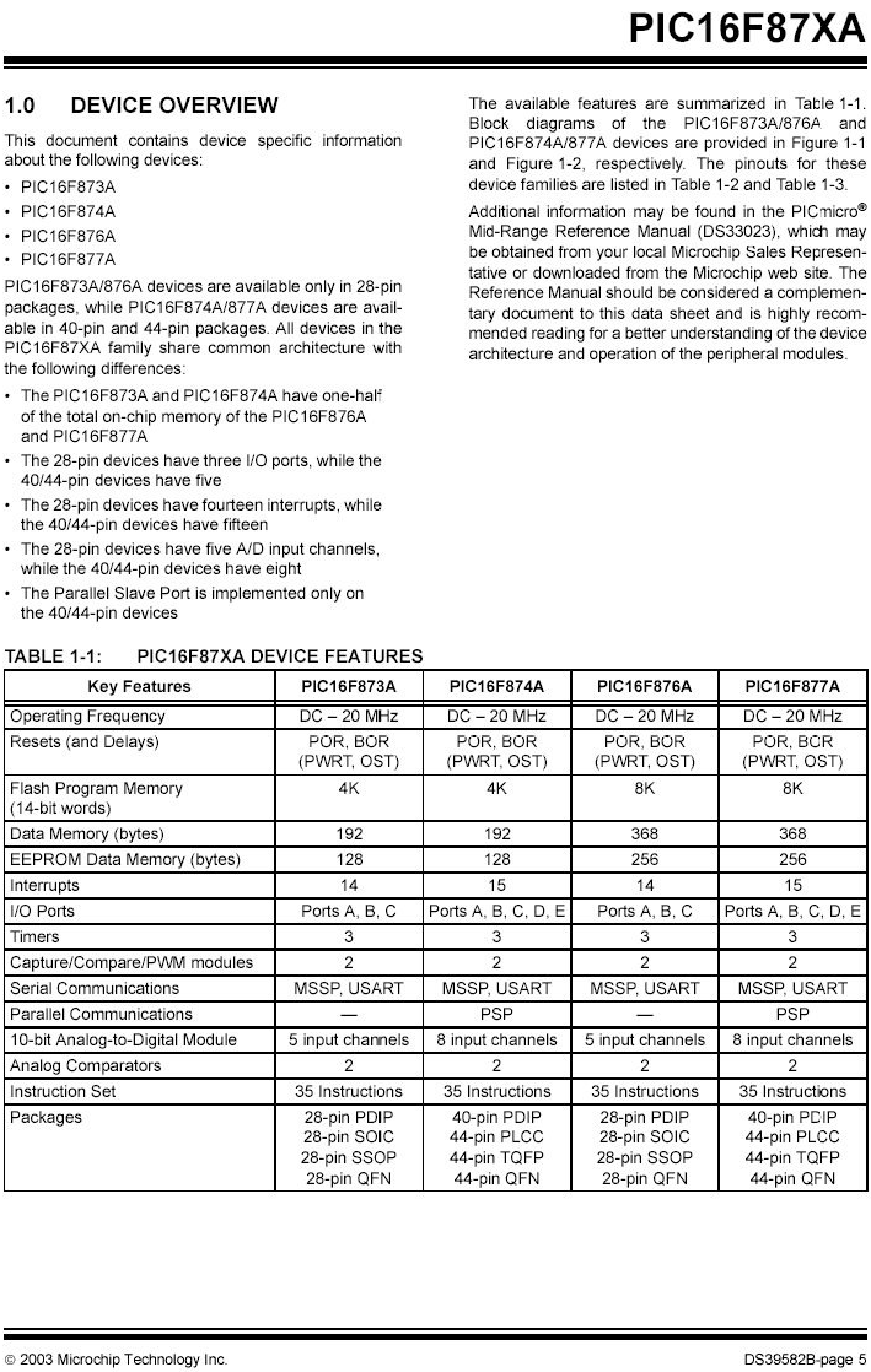
Pin Diagrams



PIC16F87XA

Pin Diagrams (Continued)





1.0 DEVICE OVERVIEW

This document contains device specific information about the following devices:

- PIC16F873A
- PIC16F874A
- PIC16F876A
- PIC16F877A

PIC16F873A/876A devices are available only in 28-pin packages, while PIC16F874A/877A devices are available in 40-pin and 44-pin packages. All devices in the PIC16F87XA family share common architecture with the following differences:

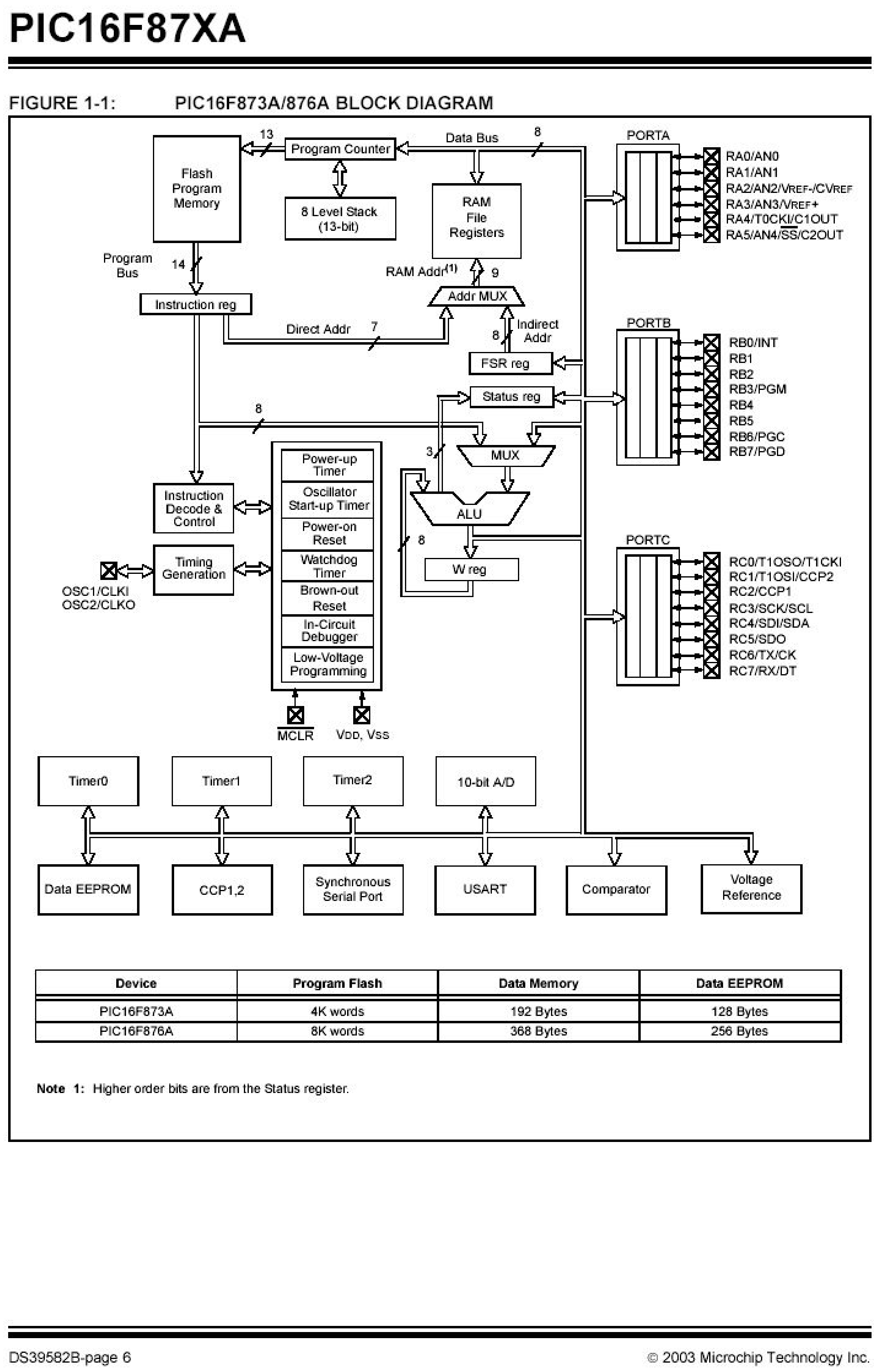
- The PIC16F873A and PIC16F874A have one-half of the total on-chip memory of the PIC16F876A and PIC16F877A
- The 28-pin devices have three I/O ports, while the 40/44-pin devices have five
- The 28-pin devices have fourteen interrupts, while the 40/44-pin devices have fifteen
- The 28-pin devices have five A/D input channels, while the 40/44-pin devices have eight
- The Parallel Slave Port is implemented only on the 40/44-pin devices

The available features are summarized in Table 1-1. Block diagrams of the PIC16F873A/876A and PIC16F874A/877A devices are provided in Figure 1-1 and Figure 1-2, respectively. The pinouts for these device families are listed in Table 1-2 and Table 1-3.

Additional information may be found in the PICmicro® Mid-Range Reference Manual (DS33023), which may be obtained from your local Microchip Sales Representative or downloaded from the Microchip web site. The Reference Manual should be considered a complementary document to this data sheet and is highly recommended reading for a better understanding of the device architecture and operation of the peripheral modules.

TABLE 1-1: PIC16F87XA DEVICE FEATURES

Key Features	PIC16F873A	PIC16F874A	PIC16F876A	PIC16F877A
Operating Frequency	DC – 20 MHz	DC – 20 MHz	DC – 20 MHz	DC – 20 MHz
Resets (and Delays)	POR, BOR (PWRT, OST)	POR, BOR (PWRT, OST)	POR, BOR (PWRT, OST)	POR, BOR (PWRT, OST)
Flash Program Memory (14-bit words)	4K	4K	8K	8K
Data Memory (bytes)	192	192	368	368
EEPROM Data Memory (bytes)	128	128	256	256
Interrupts	14	15	14	15
I/O Ports	Ports A, B, C	Ports A, B, C, D, E	Ports A, B, C	Ports A, B, C, D, E
Timers	3	3	3	3
Capture/Compare/PWM modules	2	2	2	2
Serial Communications	MSSP, USART	MSSP, USART	MSSP, USART	MSSP, USART
Parallel Communications	—	PSP	—	PSP
10-bit Analog-to-Digital Module	5 input channels	8 input channels	5 input channels	8 input channels
Analog Comparators	2	2	2	2
Instruction Set	35 Instructions	35 Instructions	35 Instructions	35 Instructions
Packages	28-pin PDIP 28-pin SOIC 28-pin SSOP 28-pin QFN	40-pin PDIP 44-pin PLCC 44-pin TQFP 44-pin QFN	28-pin PDIP 28-pin SOIC 28-pin SSOP 28-pin QFN	40-pin PDIP 44-pin PLCC 44-pin TQFP 44-pin QFN



PIC16F87XA

FIGURE 1-1: PIC16F873A/876A BLOCK DIAGRAM

Device	Program Flash	Data Memory	Data EEPROM
PIC16F873A	4K words	192 Bytes	128 Bytes
PIC16F876A	8K words	368 Bytes	256 Bytes

Note 1: Higher order bits are from the Status register.