

MULTICHANNEL TEMPERATURE LOGGER

LIM CHIANG WEI

**A project report submitted in partial fulfilment of the
requirements for the award of
Bachelor of Engineering (Hons.) Electronic Engineering**

**Faculty of Engineering and Science
Universiti Tunku Abdul Rahman**

May 2011

DECLARATION

I hereby declare that this project report is based on my original work except for citations and quotations which have been duly acknowledged. I also declare that it has not been previously and concurrently submitted for any other degree or award at UTAR or other institutions.

Signature : _____

Name : LIM CHIANG WEI

ID No. : 08 UEB 07845

Date : 6 MAY 2011

APPROVAL FOR SUBMISSION

I certify that this project report entitled “**MULTICHANNEL TEMPERATURE LOGGER**” was prepared by **LIM CHIANG WEI** has met the required standard for submission in partial fulfilment of the requirements for the award of Bachelor of Engineering (Hons.) Electronic Engineering at Universiti Tunku Abdul Rahman.

Approved by,

Signature : _____

Supervisor : Mr. Ng Choon Boon

Date : _____

The copyright of this report belongs to the author under the terms of the copyright Act 1987 as qualified by Intellectual Property Policy of University Tunku Abdul Rahman. Due acknowledgement shall always be made of the use of any material contained in, or derived from, this report.

© 2011, Lim Chiang Wei. All right reserved.

Specially dedicated to
my beloved family and friends

ACKNOWLEDGEMENTS

I would like to thank everyone who had contributed to the successful progress of this project. I would like to express my gratitude to my research supervisor, Mr. Ng Choon Boon for his invaluable advices, guidance and his enormous patience throughout the progress of the research.

In addition, I would also like to express my gratitude to my loving family and friends who had helped and given me encouragement in the research, my teammate, Stephen Lee Chin Wey for his help, ideas, and support.

MULTICHANNEL TEMPERATURE LOGGER

ABSTRACT

Digital data logger is an electronic device that records data from time to time with built in external instruments and sensors and replaces conventional chart recorders so data can be collected and analysed in more conveniently. A microcontroller-based multichannel temperature logger is developed to store data in a SD card and sends SMS to alert user the exceeded temperature reading set by user. Keypad is available to set the temperature limit, the channel used, and the mobile number. The logger measures temperature from four channels of 0 up to 400 °C and displays them on a LCD. The data is then stored in memory card at the intervals of 10 seconds, 30 seconds, 1 minutes, or 5 minutes at users' preference. A short message (SMS) from GSM modem is sent out to the preset mobile phone number if the temperature goes above the preset value. System uses 3 interconnected microcontrollers using SPI and UART serial communications. LM35DZ and K-type thermocouple are selected as temperature sensors, and SD card, 16×2 dot matrix LCD, and 4×4 matrix keypad, mobile phone for GSM modem and DS1307 for real time clock chip are interfaced with the 3 microcontrollers accordingly. SD Card interfaces via SPI and the LCD and keypad use 8 bit data lines to interface with microcontroller. The multichannel temperature logger is successfully developed and functions accordingly. The device can be implemented to become a low cost temperature logger in laboratory, where it assists during the experiment and to monitor production process as well as for meteorological research. Further development recommendations are to allow the user to customize file name, logging interval, as well as logging period.

TABLE OF CONTENTS

DECLARATION	ii
APPROVAL FOR SUBMISSION	iii
ACKNOWLEDGEMENTS	vi
ABSTRACT	vii
TABLE OF CONTENTS	viii
LIST OF TABLES	x
LIST OF FIGURES	xi
LIST OF SYMBOLS / ABBREVIATIONS	xv
LIST OF APPENDICES	xvi

CHAPTER

1	INTRODUCTION	1
	1.1 Background	1
	1.2 Aims and Objectives	2
	1.3 Scope of Works	3
	1.4 Motivation	3
	1.5 Organization of Chapters	4
2	LITERATURE REVIEW	6
	2.1 Data Logging	6
	2.1.1 Chart Recorders	6
	2.1.2 Data Logger	7
	2.2 Temperature Sensor	10
	2.3 Commercial Temperature Logger	12

3	METHODOLOGY	17
3.1	System Overview	17
3.2	Microcontroller	19
3.2.1	Main Microcontroller - PIC18F4620	23
3.2.2	Memory Card Microcontroller - PIC18F2620	26
3.2.3	RTC Microcontroller - PIC16F876A	28
3.3	Flash Memory Data Storage	34
3.3.1	Secure Digital Memory Card (SD Card)	34
3.3.2	SD Card Implementation	37
3.4	Liquid Crystal Display (LCD)	42
3.4.1	JHD162A LCD	44
3.4.2	Implementation of LCD	46
3.5	Keypad	49
3.5.1	Implementation of Keypad	52
4	RESULTS AND DISCUSSIONS	60
4.1	SD Card Testing	60
4.2	LCD and Keypad Implementation	62
5	CONCLUSION AND RECOMMENDATIONS	72
5.1	Summary	72
5.2	Recommendations	73
	REFERENCES	74
	APPENDICES	76

LIST OF TABLES

TABLE	TITLE	PAGE
2.1	General Features of Temperature Logger for Different Target User.	15
3.1	SD Card Specifications	36
3.2	SD Card Bus Protocol Vs SD Card SPI Bus Protocol	37
3.3	Hitachi 44780 Based LCD Pinout	45
3.4	Three control lines from the microcontroller.	45

LIST OF FIGURES

FIGURE	TITLE	PAGE
2.1	Strip Chart Recorders. (Wikipedia, 2010)	7
2.2	Circular Chart Recorders. (Wikipedia, 2010)	7
2.3	Thermocouple Measuring Circuit. (eFunda, Inc.,2011)	11
2.4	Extech Non-Contact Infrared Thermometer with IR Thermometer Probe.	12
2.5	Dickson SM320 Display Temperature Data Logger.	13
2.6	TandD Thermo Recorder Data Logger.	13
2.7	Fluke's Hart 1523/1524 Reference Thermometers.	14
2.8	Omega Engineering HH306A Datalogger Thermometer.	15
3.1	System Block Diagram.	18
3.2	Hardware Implementation between 3 Microcontrollers.	20
3.3	Main Microcontroller – PIC18F4620.	21
3.4	Memory Card Microcontroller – PIC18F2620.	21
3.5	RTC Microcontroller – PIC16F876A.	22
3.6	Hardware Implementation of PIC18F4620 Microcontroller	24
3.7	Hardware Implementation of Sensors, LCD and keypad with PIC18F4620 Microcontroller	25

3.8	Hardware Implementation of SD Card module with PIC18F2620 Microcontroller.	27
3.9	Hardware Implementation of RTC with PIC16F876A Microcontroller.	29
3.10	Programme Flow Diagram for PIC18F4620 Microcontroller.	31
3.11	Programme Flow Diagram for PIC18F2620 Microcontroller.	32
3.12	Programme Flow Diagram for PIC16F87A Microcontroller.	33
3.13	SD Card Pin Layout.	36
3.14	SPI Master-Slave Connection.	38
3.15	Programme Flow Diagram for SD Card Implementation.	39
3.16	SPI and SD Card initialization Code.	40
3.17	SD Card Data Writing Code.	41
3.18	Hardware Implementation of LCD with PIC18F4620 Microcontroller.	42
3.19	Hardware Implementation of LCD with PIC18F876A Microcontroller.	43
3.20	2×16 JHD162A LCD.	44
3.21	Programme Flow Diagram for Initialise the LCD.	46
3.22	LCD Initialization Code.	47
3.23	Subroutine for LCD Initialization Code.	47
3.24	LCD code for display character “TEMP”.	48
3.25	Subroutine for LCD to display characters.	48
3.26	4×4 Matrix Keypad.	49
3.27	Hardware Implementation of Keypad	50
3.28	Internal Structure of Matrix Keypad.	51

3.29	Programme Flow Diagram of Select Channel for Keypad.	52
3.30	Programme Flow Diagram of Temperature Limit for Keypad.	53
3.31	Code for Keypad in Selecting Channel.	55
3.32	Code for Keypad Check for Column 1.	56
3.33	Code for Keypad Check for Column 2.	57
3.34	Code for Keypad Check for Column 3.	58
3.35	Code for Keypad for Confirmation.	58
4.1	The Data Logged Inside the File.	61
4.2	The .txt Format File Opened in Microsoft Office Excel Spreadsheet.	61
4.3	The LCD Display for a SD Card Needed to Start the System.	62
4.4	The LCD Display Press ‘*’ to Start the System.	63
4.5	The LCD Displays The Channel Menu.	63
4.6	The LCD Shows The Entered Channel.	64
4.7	The LCD Display for Channel Confirmation.	64
4.8	The LCD Displays Logging Enabling Menu.	65
4.9	The LCD Displays the Logging Interval Menu 1.	65
4.10:	The LCD Displays The Logging Interval Menu 2.	65
4.11	The LCD Displays The Confirmation of Logging Interval.	66
4.12	The LCD Displays The Logging Period Menu 1.	66
4.13:	The LCD Displays The Logging Period Menu 2.	66
4.14	The LCD Displays The Confirmation of Logging Period.	67
4.15	The LCD Displays The Temperature a Preset Limit Menu.	67

4.16	The LCD Displays The Temperature a Preset Limit Menu with a Set of Keyed Limit.	68
4.17	The LCD Displays the Confirmation of Temperature Preset Limit.	68
4.18	The LCD Displays The Mobile Number Menu.	69
4.19	The LCD Displays The Mobile Number Menu with A Set of Number.	69
4.20	The LCD Displays the Confirmation of Mobile Number.	69
4.21	The LCD Display when the 2 Microcontrollers Exchange Data via SPI.	70
4.22	The LCD Display for 1st 2 Channels of Temperature Reading..	70
4.23	The LCD Display for 2nd 2 Channels of Temperature Reading.	71

LIST OF SYMBOLS / ABBREVIATIONS

LCD	liquid crystal display
RTD	resistor temperature detector
EEPROM	Electrically Erasable Programmable Read-Only Memory
GSM	Global System for Mobile Communications
SIM	Subscriber Identity Module
USB	Universal Serial Bus
GPRS	general packet radio service
UMTS	Universal Mobile Telecommunications System
HSDPA	High-Speed Downlink Packet Access
SMS	Short Message Service
MMS	Multimedia Messaging Service
ADC	analogue-to-digital converter
RTC	Real time clock

LIST OF APPENDICES

APPENDIX	TITLE	PAGE
A	Liquid Crystal Display (LCD)	76
B	Project Planning	79

CHAPTER 1

INTRODUCTION

1.1 Background

A temperature data logger is an instrument which can be programmed to record environmental temperature with a built in sensor or via external instruments and sensors such as thermistor, thermocouple, and resistance thermometer. The advantage of temperature loggers is that they can operate independently of a computer, unlike many other types of data acquisition devices and are available in various shapes and sizes. This reduces the workforce as the worker can perform other tasks before the specified period of time ends and also allows for a comprehensive, accurate picture of the environmental conditions being monitored. The range of the devices includes simple economical single channel fixed function loggers to more powerful programmable devices capable of handling hundreds of inputs. They are normally more economical than chart recorders and offer more flexibility with a greater variety of input types. Most loggers collect data which may be directly transferred to a computer.

It is needed to know the temperature of an object to understand its important properties and also to know the temperature in a certain device or process to know if things are working properly. For example one may need to go into the field to measure temperature in a device or process for certain reasons such as high or low temperatures may indicate a problem with a process, high temperatures may indicate excessive friction, wear or energy losses. Besides that, high temperatures may degrade materials. Temperature can be measured in a number of ways, and as with

all measurements there are uncertainties involved. Certain methods may be more appropriate for certain applications and conditions. For instance it is important to measure the temperature of food because microorganisms such as germs, bacteria, and viruses grow best on foods in the temperature danger zone and can cause foodborne illness. Therefore temperature data logger is developed as for the reason the conventional data logging method which uses human force to read and record the reading continuously without wasting the human resources.

1.2 Aims and Objectives

The project aim is to develop a microcontroller-based multichannel temperature logger which is able to store data in a SD Card and sends a SMS to alert the user the exceeded temperature reading set by the user. A keypad is available so the user can set the temperature limit, insert the channel which going to use and the mobile number to send the alert SMS. An LCD displays the temperature reading, as well as the number that pressed on keypad.

The multichannel temperature logger should be able to measure temperature from the four channels from 0°C up to 400 °C and display them on a LCD. All the temperature measurement data will be stored in a memory card at the intervals of 10 seconds, 30 second, 1 minute, and 5minutes in a format that is easy to be analyzed by the user. Besides that, it sends out a short message from GSM modem if the temperature exceeds a preset value which is set by user using keypad. All modules of the system are connected to a microcontroller. The microcontroller will receive the analogue signals from temperature sensor and analogue to digital conversion process is carried out. The ADC result will be stored in the memory card and also converted to display the current temperature on the LCD.

1.3 Scope of Works

The development of multichannel temperature logger is equally divided between 2 teammates, where the tasks of development of microcontrollers' communication, keypad and LCD interfacing with microcontroller as well as the data writing to memory card modules using microcontroller are assigned to author. Meanwhile, the author's teammate, Stephen Lee Chin Wey is assigned with developing the system to send short messages, counts the time using real time clock for data logging, and measuring and analysing the environment temperature modules. The author and his teammate then work together to integrate all the module developed, draw the printed circuit boards, assemble the hardware, and finally test the whole system for its' functionality.

1.4 Motivation

Multichannel Temperature Logger is a digital data logger that records data from time to time which is round the clock or in relation to location with a built in sensor. Upon activation, data loggers are typically deployed and left unattended to measure and record information for the duration of the monitoring period. This allows a comprehensive, accurate picture of the environmental conditions being monitored, such as air temperature. In most of the cases when a temperature control is required, the mean is needed to determine the parameters for the system under test. With an efficient logger we can determine parameters like response time, the slope of the temperature variation or long time stability. They are generally small, battery powered, portable, and equipped with a microprocessor, internal memory for data storage, and also sensors. The multichannel temperature logger has a local interface device such as keypad and LCD and can be used as a stand-alone device. Moreover, this device is built-in with SMS alert system to send out alert message to user whenever the reading exceeded the pre-set value.

Due to the conventional data logging method which uses human operator to read and record the reading continuously and this method is a waste of human

resource as they can be more productive. Moreover, the chart recording method collects the data in continuous form and not discrete which need to be keyed into the computer before it can be graphed for further analysis. This multichannel temperature data logger is used to solve above problem which automatically reads the temperature data at a preset time interval, logs into a memory device and the collected readings can be easily transferred to the computer for further analysis. Besides that, the temperature data logger built in with an alert system therefore it does not require human operator to monitor the device all the time.

This device can be applied in many ways, such as in food industry. Temperature data logger is often used to ensure that unprocessed food is stored at the correct temperature in cold stores. It is also used in ovens and production lines to ensure the food has been heated to the correct temperature for the required length of time. In the semiconductor industry clean rooms and production processes must be kept at a constant temperature to ensure quality control. Here the temperature data logger is indispensable. This device also can be applied to the environment of technology which does not need any manpower to monitor it and the built-in GSM technology can alert the user if the temperature is abnormal. Hospitals use many types of data logger for temperature recording. Fridges and freezers where drugs and samples are stored will have a temperature data logger permanently installed. Incubators are also generally monitored for temperature and here a data logger is often used for historical data.

1.5 Organization of Chapters

Chapter 1 gives a brief background of the temperature data logger and importance of measuring temperature. The aim and objectives are also stated here to provide direction for this project. Besides that, motivation and scope of works are also included in this chapter.

Chapter 2 involves the literature review. The review of data logger and its evolution as well as the types of temperature sensors and commercial temperature loggers are included here.

Chapter 3 discusses the methodology of this project. The hardware used is discussed in this chapter. The hardware and software implementations are also included in this chapter as well as discussion of flow diagram and programming detail.

Chapter 4 contains the result of the multichannel temperature logger module testing and the discussion of the testing conducted.

Chapter 5 indicates the recommendations on future work that can be done to improve the multichannel temperature logger. Besides that, the conclusion of this project is included as well.

CHAPTER 2

LITERATURE REVIEW

2.1 Data Logging

Data logging is a process of recording the data and monitoring the processing of the system so that the gathered data can be reviewed for analysis. Data logging is commonly used in monitoring a system where need human to collect plenty of data. Besides that, data logging helps in the situation where the information changes faster than a human can possibly collect. The earliest form of recording data involved manually taking measurements by human after that recording the data to a written log, and plotting the data on graph paper for analysing the system. This method has been replaced by chart recorders in late 19th century.

2.1.1 Chart Recorders

A chart recorder is a data logging system that use electrical or mechanical input by using different colour of pens draw onto a piece of paper. There are several types of chart recorders such as strip chart, circular chart and roll chart recorders. (Wikipedia, 2010) Strip chart recorders have a long strip of paper that is ejected out the side of the recorder as shown in Figure 2.1. Circular chart recorders have a rotating disc of paper that must be replaced more often, but are more compact and amenable to being enclosed behind glass as shown in Figure 2.2. Roll chart recorders are similar to strip

chart recorders except that the recorded data is stored on a round roll, and the unit is usually fully enclosed.

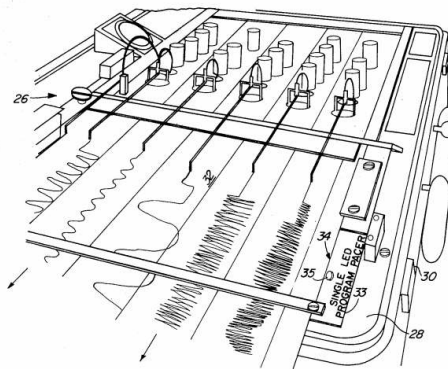


Figure 2.1: Strip Chart Recorders. (Wikipedia, 2010)

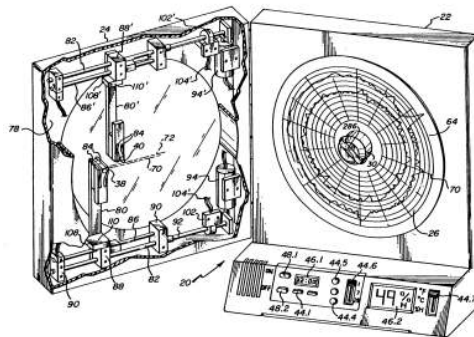


Figure 2.2: Circular Chart Recorders. (Wikipedia, 2010)

2.1.2 Data Logger

Today, as digital data loggers take a growing share of the market, it appears that mechanical chart recorder might be the next technology to drop out of sight. Until fairly recently, chart recorders were used to create a record of varying data over a specified period of time. Chart recorders are still the best devices for performing the data logging but chart recorders has been replaced with digital data logger due to the data logged is stored in electronic format and data analysis becomes more convenient. Besides, it offers more flexibility and lower cost because the data in the memory can

be erased and replaced with the new data, different from the conventional recorder where constant replacement of paper chart is needed.

A digital data logger is an electronic device that records data from time to time with a built in external instruments and sensors. The data loggers are mostly controller by digital processor such as computer or microcontroller. The data loggers are small, battery powered, portable, and equipped with a microprocessor, internal memory for data storage, and sensors. Besides that, data loggers interface with device such as keypad, liquid crystal display (LCD), and can be used as a stand-alone device. The importance interface device for data loggers are sensors that are commonly connected includes thermocouples, resistor temperature detectors (RTDs), thermistors, and etc. Data loggers automatically collect data on a 24-hour basis. Data loggers' measure and record data follow the logging interval and logging period to perform its job.

According to Petreus, et al (2006), result parameters like response time, temperature variation's slope or long time stability can be determined with an efficient logger. Three functioning modes are featured in a device implementation which are dead time, variation speed and also overshoots or stability. Classical logger can be applied if the temperature is to be monitored constantly at regular intervals and this mode is called rate monotonic logging. The logger mode which constantly monitors the temperature but only log at the moments when a significant variation occurs is called value sensitive logging. It is not efficient when the monitored temperature has multiple and high speed variations as the sensitive value logging mode is introduced to preserve the quantity of memory involved in the logging process. Another mode is where the logger takes records of the temperature at high speed but stop performs logging as it reaches the limit preset by the user.

Data loggers typically have slower sample rates compared to data acquisition devices. Small internal memory is introduced in early data loggers to store the data and is connected to the computer via RS-232 serial communication port to retrieve the data. The memory in more advanced loggers is very large to accommodate many days, or even months, of unattended recording. This memory may be battery-backed static random access memory, flash memory or EEPROM.

Temperature logger software typically is used to help the user download the data from the logger into an application that is used to analyze the logged data. It usually has the user setup interface at the initial use such as to select the inputs to measure, the sampling rate, the alert alarm settings, and others. Users also are allowed to define how the logging application period and select alarm conditions to control one or more outputs that can be used to notify of the alarm or even to perform an emergency shutdown of the system. In order to alert the users whenever the measured temperature exceeds the preset limit or there is abnormal trend detected, buzzer is added to one of the features of the temperature logger and it will be turned on until the user switch it off manually.

The size of the device getting smaller with more features and the design also becomes trendier as the technology of temperature logger improves. Logger becomes portable and user does not have to look for the nearby power point before using it as lesser power consumption happens due to the system size getting smaller and therefore battery can be attached to power up the device.

Improvements are also made on the way to alert the users whenever the reading gets abnormal or exceeds the limit. GSM modem or wireless network hardware is attached to it to send short message or call the user's mobile number, or email to report the user. They allows user to be more flexible as they do not have to be around the logger during operation and they can continue their other tasks anywhere. Besides that, data loggers range from simple single-channel input to complex multi-channel instruments. Simpler device has less programming flexibility. Some sophisticated instruments allow for cross-channel computations and alarms based on predetermined conditions. New data loggers can serve web pages, allowing numerous people to monitor a system remotely.

2.2 Temperature Sensor

Temperature sensor is a device that gathers data concerning the temperature from a source and converts it to a form that can be understood either by an observer or another device. Temperature sensors come in many different forms and are used for a wide variety of purposes, from simple home use to extremely accurate and precise scientific use. (Ladyada.net, 2011)

There are two types of sensors, the contact and non-contact sensors. Noncontact temperature sensors include many different types, however they are sharing one set of unique features, which is they are often involved with an optical property of materials called spectral emissivity or spectral emittance. Contact temperature sensors meanwhile measure their own temperature, where the object to which the sensor is in contact is assumed that the two are in thermal equilibrium, that is, there is no heat flow between them. Examples are thermocouple, resistance temperature detector (RTD) and thermistor.

A thermocouple is a junction between two different metals that produces a voltage related to a temperature difference. Thermocouples are among the easiest temperature sensors to use and obtain and are widely used in science and industry. (Varalakshmi, 2011) They are based on the Seebeck effect that occurs in electrical conductors that experience a temperature gradient along their length. They are simple, rugged, and inexpensive, need no batteries, and can measure over very wide temperature ranges. The main limitation is accuracy where system errors of less than one degree Celsius (C) can be difficult to achieve. In Figure 2.3, it shows working principle of thermocouple.

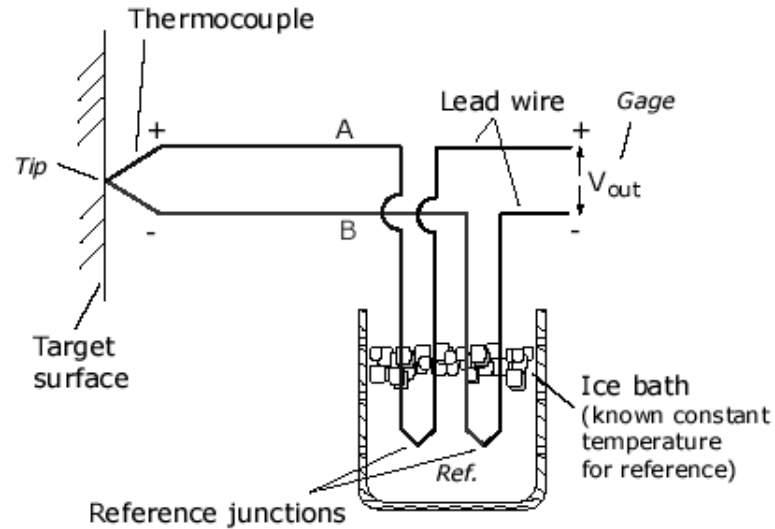


Figure 2.3: Thermocouple Measuring Circuit. (eFunda, Inc.,2011)

Resistance temperature detectors or resistive thermal devices (RTDs) are temperature sensors that exploit the predictable change in electrical resistance of some materials with changing temperature. It is a positive temperature coefficient device, which means that the resistance increases with temperature. The resistive property of the metal is called its resistivity. The resistance is proportional to length and inversely proportional to the cross sectional area. Due to higher accuracy and repeatability, they are slowly replacing the use of thermocouples in many industrial applications below 600 °C. The RTD is one of the most accurate temperature sensors as not only provides good accuracy; it also provides excellent stability and repeatability. They are popular because of their excellent stability, and exhibit the most linear signal with respect to temperature of any electronic temperature sensor. On the other hand, it also has disadvantage which is slow response time and low sensitivity. (Thermometrics, 2011).

Thermistor is a type of thermally sensitive resistor and have, according to type, a negative (NTC), or positive (PTC) resistance/temperature coefficient. Thermistors differ from resistance temperature detectors (RTD) in that the material used in a thermistor is generally a ceramic or polymer, while RTDs use pure metals. The temperature response is also different, where RTDs are useful over larger temperature ranges, while thermistors typically achieve a less precision compared to

RTDs within a limited temperature range, usually $-90\text{ }^{\circ}\text{C}$ to $130\text{ }^{\circ}\text{C}$. It is also able to detect minute changes in temperature, which could not be observed by RTD or thermocouple circuit. Thermistor is a non-linear device which is highly dependent upon process parameters. In order to overcome this problem, a linearization circuit should be used. (Temperatures.com, 2003)

2.3 Commercial Temperature Logger

Numerous temperature data loggers are available according to the users' specifications and one of the devices available is Extech Non-Contact Infrared Thermometer with IR Thermometer Probe as shown Figure 2.4. This device able to record and log temperature readings with 2 types of K thermocouple probes and an IR thermometer probe ranging from $-200\text{ }^{\circ}\text{C}$ to $1372\text{ }^{\circ}\text{C}$ with $0.1\text{ }^{\circ}\text{C}$ or $1.0\text{ }^{\circ}\text{C}$ resolution and accuracy of $\pm 0.15\%$ for K-type thermocouple and $\pm 2\%$ for infra-red probe respectively. The device has the logging capacity of 18,000 readings and 8:1 distance to spot ratio.



Figure 2.4: Extech Non-Contact Infrared Thermometer with IR Thermometer Probe.

Other than that, another commercial product of temperature logger is the Dickson SM320 Display Temperature Data Logger as shown in Figure 2.5. This device can measure the temperature from $-184\text{ }^{\circ}\text{C}$ to $1093\text{ }^{\circ}\text{C}$ using either thermistor or thermocouple with $0.1\text{ }^{\circ}\text{C}$ resolution and user selectable logging interval from 1

second to 24 hours. The device logging operation works at the temperature from -20 °C to 70 °C and has 6 months battery life. It can store log up to 32,000 reading and the logged data can be transferred out from the device using USB, serial connection, or Flash Card method.



Figure 2.5: Dickson SM320 Display Temperature Data Logger.

Meanwhile, Figure 2.6 shows the TandD Thermo Recorder Data Logger which is another type of temperature logger in the market temperatures within a broad range can be measured by connecting to any 3-wire Platinum Resistance Sensor. This device measures the 2 channels temperature ranging from -200°C to 600°C and does the logging in 15 different recording intervals. It able to store 8,000 readings for each channel and the data can be retrieved to the computer via the RS-232 serial communication. The battery attached is able to support the device from 5 to 8 months, depending on the recording intervals, environment, as well as the battery performance itself.



Figure 2.6: TandD Thermo Recorder Data Logger.

Apart from that, Fluke's Hart Scientific division produces 1523/1524 Reference Thermometers measures temperature from $-200\text{ }^{\circ}\text{C}$ to $1750\text{ }^{\circ}\text{C}$ with accuracy of $\pm 0.24\text{ }^{\circ}\text{C}$ for thermocouple and $\pm 0.002\text{ }^{\circ}\text{C}$ for thermistor which shown in Figure 2.7. The temperature is sampled at 0.3 seconds intervals and does 25 readings with statistics. The logged data can be transferred via RS-232 or USB and the battery power can lasts at least 20 hours on three AA batteries. The device has power saving features and it can be enabled or disabled for longer battery life or greater convenience.



Figure 2.7: Fluke's Hart 1523/1524 Reference Thermometers.

In Figure 2.8, Omega Engineering meanwhile produces HH306A Data Logger Thermometer with USB and RS232, where it can measure the temperature within the range of $-200\text{ }^{\circ}\text{C}$ to $1370\text{ }^{\circ}\text{C}$ and $\pm 0.2\%$ accuracy. The operating temperature of the device is within $0\text{ }^{\circ}\text{C}$ to $50\text{ }^{\circ}\text{C}$ and it can store up to 16,000 records at programmed intervals as fast as once per second. Besides, it also comes with USB and RS232 interface cables and Windows based software for the display and saving of data.



Figure 2.8: Omega Engineering HH306A Datalogger Thermometer.

In conclusion, there are several types of temperature loggers in the market that target different categorized of users based on their purpose such as educational purpose, industrial purpose and normal usage purpose. Table 2.1 shows the general features of temperature logger based on the user's purposes.

Table 2.1: General Features of Temperature Logger for Different Target User.

	Education	Consumer	Industrial
Cost	Low cost	Low cost	High cost
Data channel	1 to 10	2 to 4	5 to 20
Stand alone	Yes	Yes	Both
Battery backup	Yes	Yes	Yes
Power supply	No	No	Yes
Temperature range	-25°C to 110°C	-25°C to 180°C	-50°C to 800°C
Displays data	Yes	Yes	Yes
Display type	LCD	LCD	LCD (graphic)
Multiples display	Yes	No	Yes
Remote probes	Yes	Yes	Yes
Sample rate	10sec to 24 hours	1sec to 255min	1sec to 24hours
Data points stored	5000	2048	4500 to 64000
Probe cable length	1 to 2meters	2 to 4 meters	2 meters above
Alert relay	No	Yes (email or text)	Yes (email or text)
Maintenance	No	No	Yes

Every data logger on the shelves has its' specifications and special features to fit different users' requirement. However, there is no device that provides multichannel temperature measurement with data logging and alerts the user when the reading exceeded the limit using SMS feature. Devices that provide data logging do not provide SMS alert service to the user and storage is not removable and they have to connect the device to the computer in order to perform data collection. The project proposes to improve the features to measure high temperature of up to 400° C, data logging into removable memory storage, and SMS alert features. RTDs and thermocouples will be used as the temperature sensor due to both sensors can support high temperature which more than 400° C. Besides that, the linearity of both sensors consider easy for implementation. Lastly, the price for RTDs and thermocouples are inexpensive and their size is small as well.

CHAPTER 3

METHODOLOGY

3.1 System Overview

The system, as shown in Figure 3.1, consists of six peripheral devices connected to a control unit, which is the microcontroller, which are temperature sensors, liquid crystal display (LCD), keypad, Secure Digital (SD) memory card, GSM modem, and the Real Time Clock (RTC). Three microcontrollers are used in the system and interconnected to each other using serial peripheral interfaces.

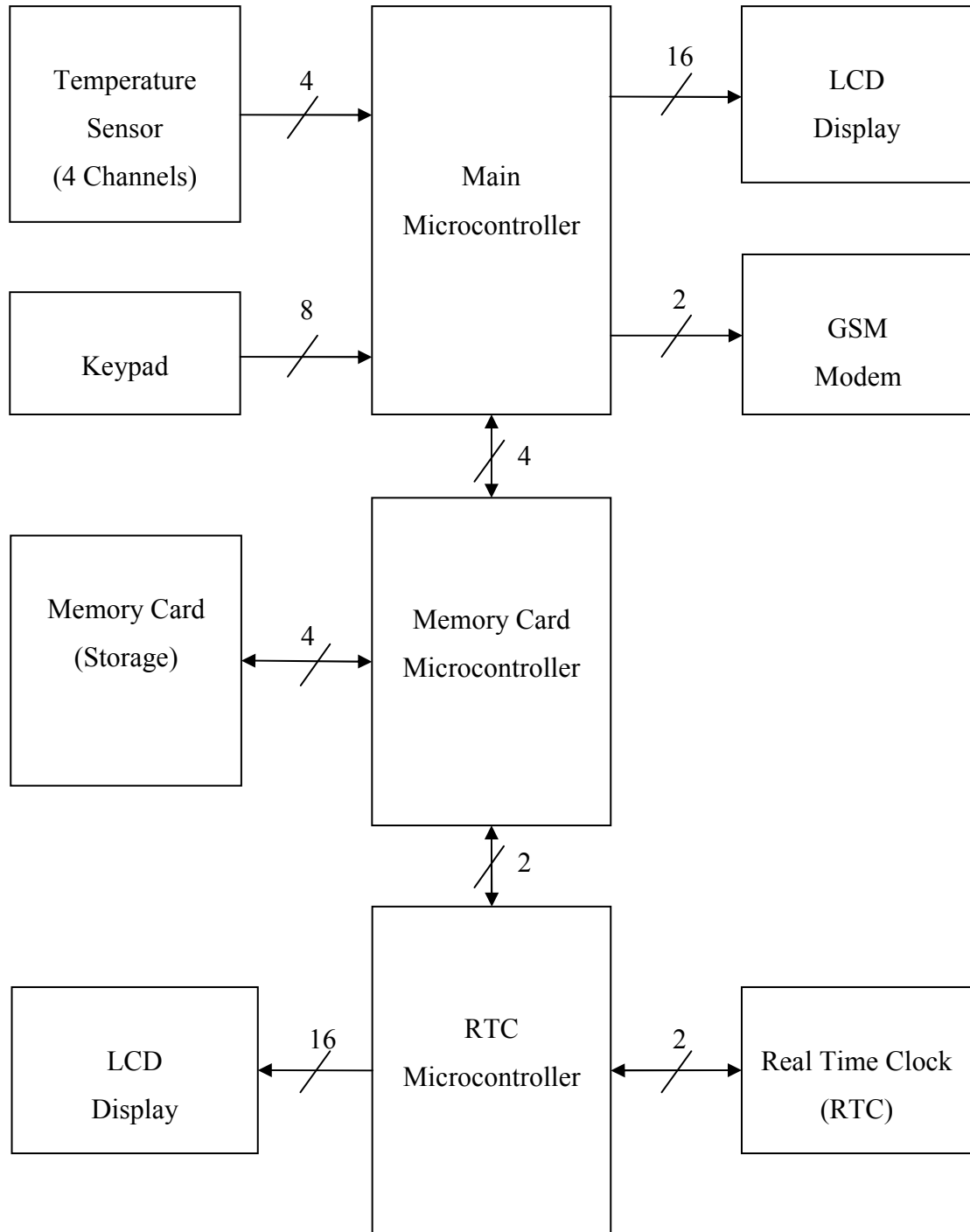


Figure 3.1: System Block Diagram.

Three microcontrollers are used in this system due to the devices connected use the same input output pins but different serial interface module of the microcontroller such as the SD Card and the RTC chip. Besides, the duration time for the microcontroller to write a row of data to the SD Card takes longer time as compare to the temperature measurement and this interfere the performance of the system. Therefore, one microcontroller is responsible of writing the data into the SD Card. The temperature sensors, keypad, LCD, and the GSM modem are connected to the main microcontroller, while the memory card microcontroller has only SD Card module connected to it. The RTC chip meanwhile is connected to the third microcontroller and this microcontroller only responsible to reads and sends the current date and time to the memory card microcontroller for logging purpose. Concerns on the system size, weight, as well as design are also taken into account instead of only system functions and efficiency to make the system portable and trendy.

3.2 Microcontroller

Microcontroller is a chip which consists of simple functions of a computer system, it is integrated with a processor core which is used to execute the instructions, memory which is used to store the instructions and data whether temporary or permanently and some programmable input and output peripherals which is capable to send or receive signal from the user. (Tooley, 2006)

The characteristics mentioned above makes the microcontroller different from microprocessor which has a processor core (CPU) for high speed data processing. Besides of data processing, microcontrollers also provides additional elements such as read and write memory for data storage, read only memory for programme storage, flash memory for permanent data storage, and input output interface for connection between the microcontroller and various peripherals.

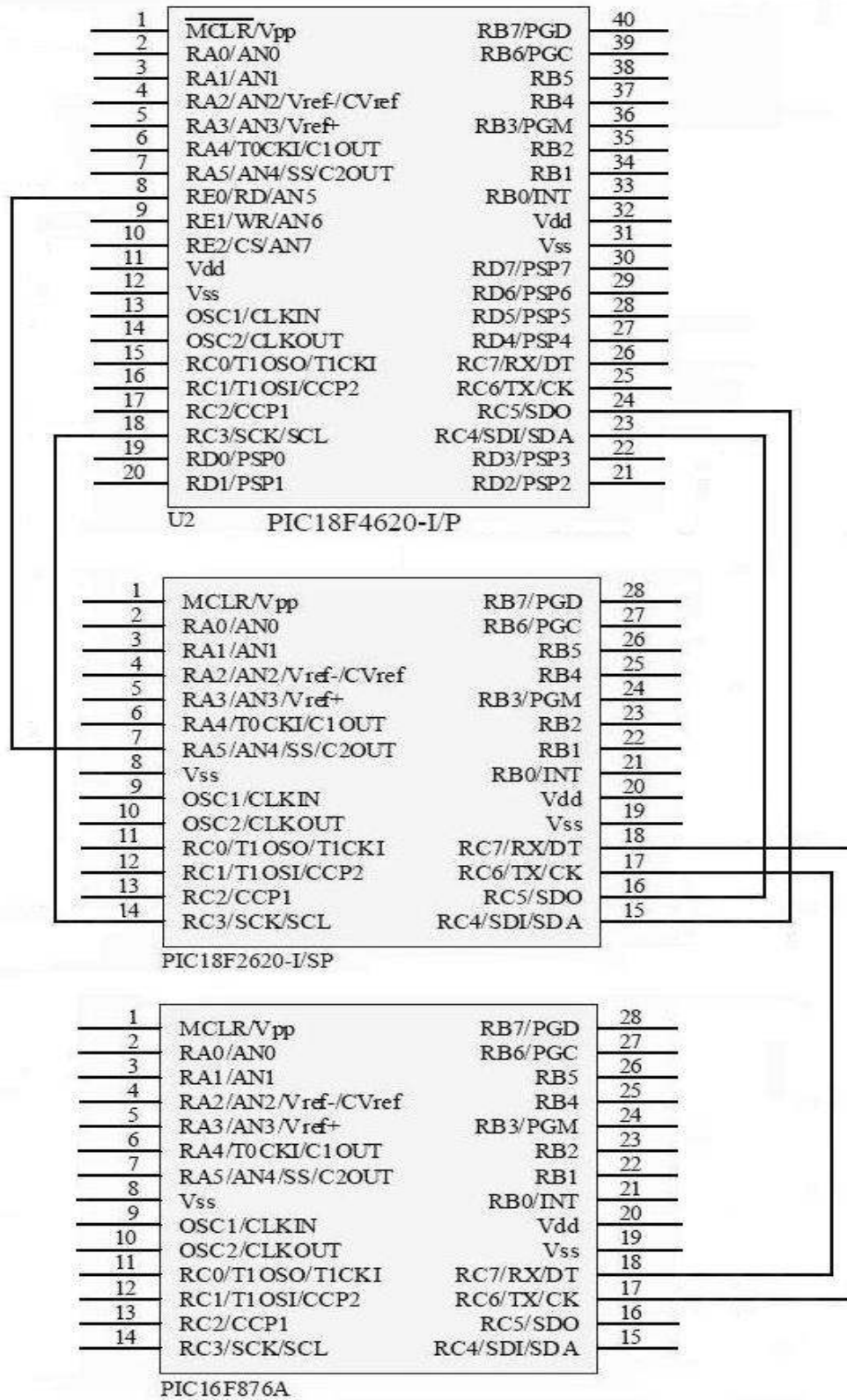


Figure 3.2: Hardware Implementation between 3 Microcontrollers.

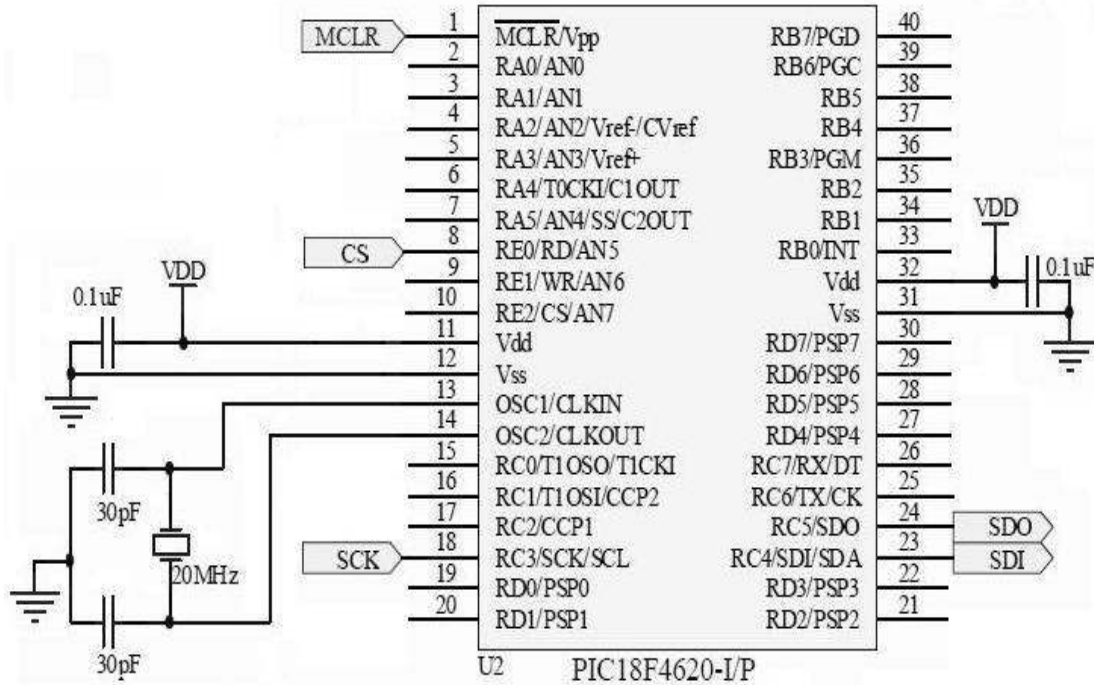


Figure 3.3: Main Microcontroller – PIC18F4620.

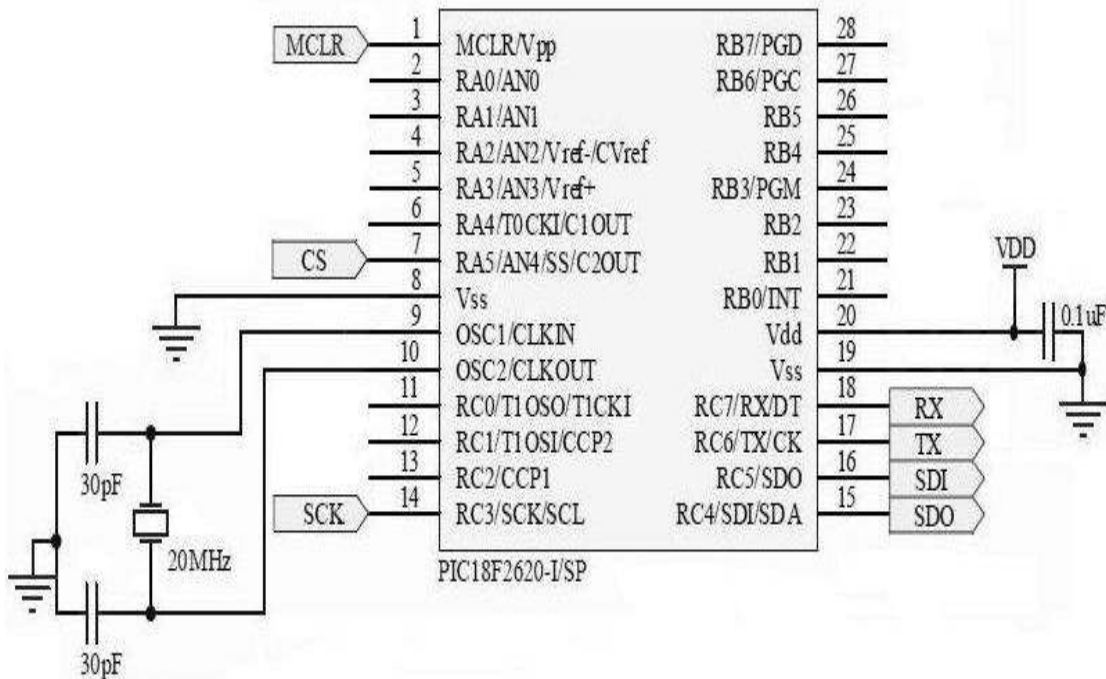


Figure 3.4: Memory Card Microcontroller – PIC18F2620.

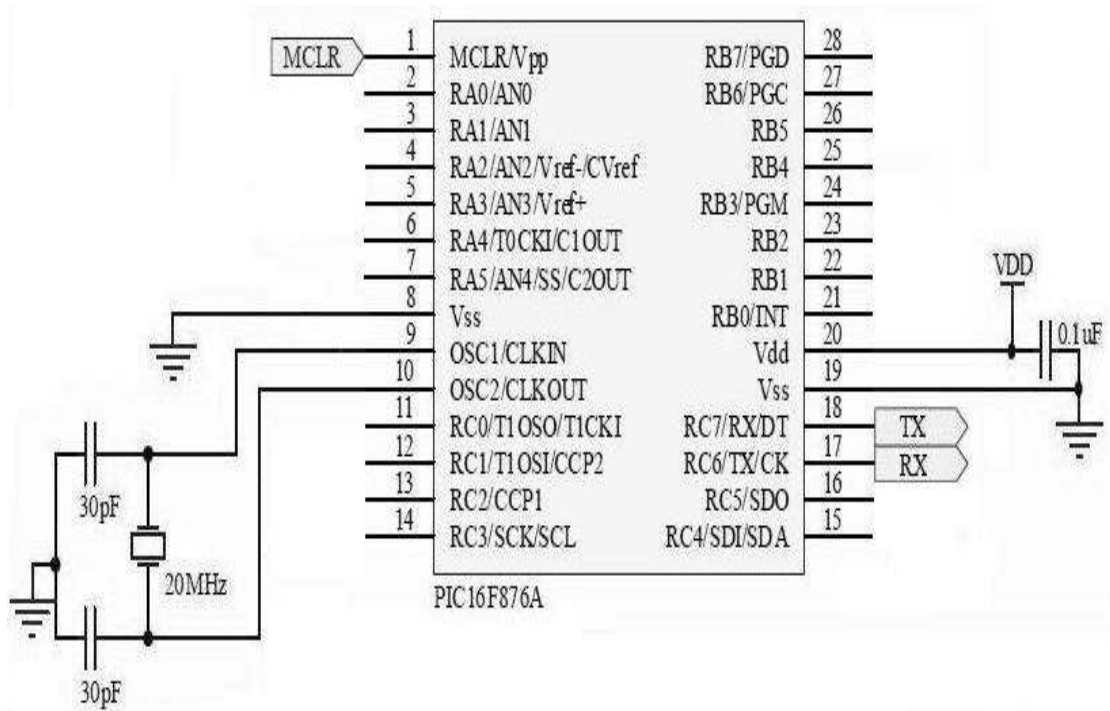


Figure 3.5: RTC Microcontroller – PIC16F876A.

The main microcontroller for this project is PIC18F4620 as shown in Figure 3.3 while in Figure 3.4 shows the PIC18F2620 microcontroller is used to interface with memory card slot and PIC16F876A microcontroller is used to interface with RTC module which shown in Figure 3.5. SPI module is used to interface between PIC18F4620 and PIC18F2620 microcontrollers. The PIC18F4620 microcontroller uses 4 pins to interface with the PIC18F2620 microcontroller where the slave select pin of PIC18F2620 microcontroller is connected to chip select pin of the PIC18F4620 microcontroller as shown in Figure 3.2. Furthermore, input data pin, SDI from the PIC18F4620 microcontroller is connected to the output data pin, SDO of the PIC16F2620 microcontroller and vice versa. The serial clock pin, SCK of both microcontrollers is connected together. It transfers the selected channels, the preset limit, mobile number, and current temperature reading from PIC18F4620 to PIC18F2620 microcontrollers to be logged into the memory card and the system restart once the logging period has reached.

In Figure 3.2, the UART module is used to transfer the data between PIC18F2620 and PIC16F876A microcontrollers. The transmit pin, TX of PIC18F2620 microcontroller is connected to the receive pin, RX of the PIC16F876A microcontroller and vice versa. The present date and time is sent from PIC16F876A to PIC18F2620 microcontrollers to be written into memory card every time the logging interval reached.

3.2.1 Main Microcontroller - PIC18F4620

An 8-bit microcontroller PIC18F4620 is chosen for the main microcontroller in this project due to the functions and input/output ports provided is adequate and the memory size of this microcontroller is large enough to store the programme. This microcontroller is chosen in this project as the core control unit because it has low cost, approximately RM30, small size, low power consumption, stable, fast response, and low error. The 40 pins multiple functions of the microcontroller consists of $32K \times 14$ words of Flash Program Memory, 3968×8 bytes of Data Memory (RAM), and 5 input output ports.

This microcontroller has peripheral features such as Master Synchronous Serial Port (MSSP) with Serial Peripheral Interface (SPI) to communicate with other peripheral devices such as serial EEPROM and shift register and Universal Synchronous Asynchronous Receiver Transmitter (USART/SCI) with 9-bit address detection to communicate with peripheral devices such as analogue to digital or digital to analogue integrated circuits and serial EEPROM. Besides, it also has 10-bit, up to 13 channels Analogue to Digital Converter (ADC) which used to convert analogue signal ranging from 0 V to 5 V to a set of binary signals 1 or 0. These features can be implemented into the project whereby MSSP can be used to connect with PIC18F2620 microcontroller to send measurement data as well as the temperature limit and the user's mobile number to be logged into the SD card.

Meanwhile, the USART module is used to interface with the mobile phone to use the short messaging service feature in the phone. As it serves as the core of the system, it connects most of the peripheral devices of the system such as temperature sensors, keypad, mobile phone, and liquid crystal display (LCD).

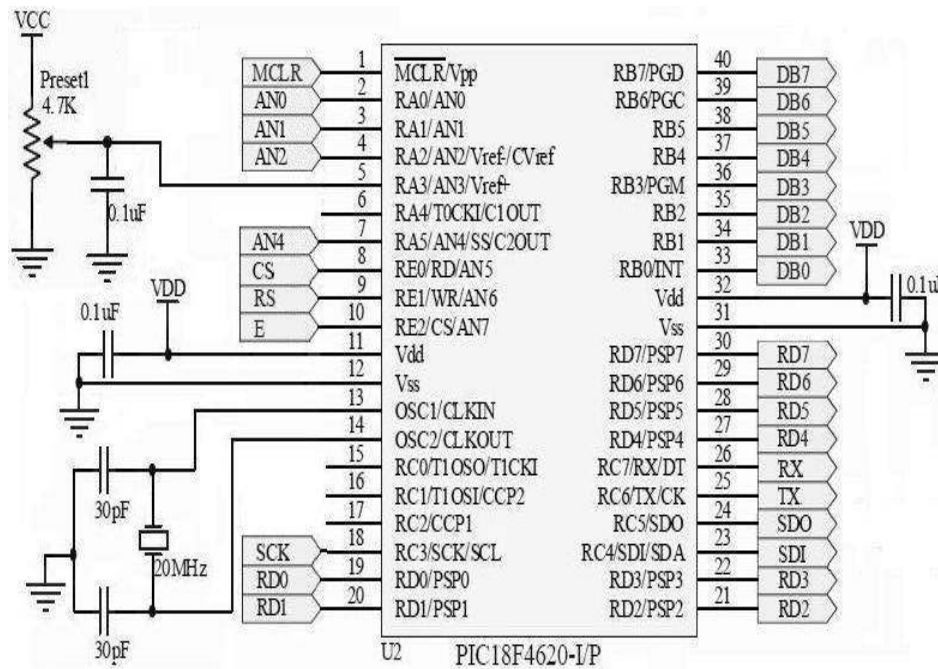


Figure 3.6: Hardware Implementation of PIC18F4620 Microcontroller

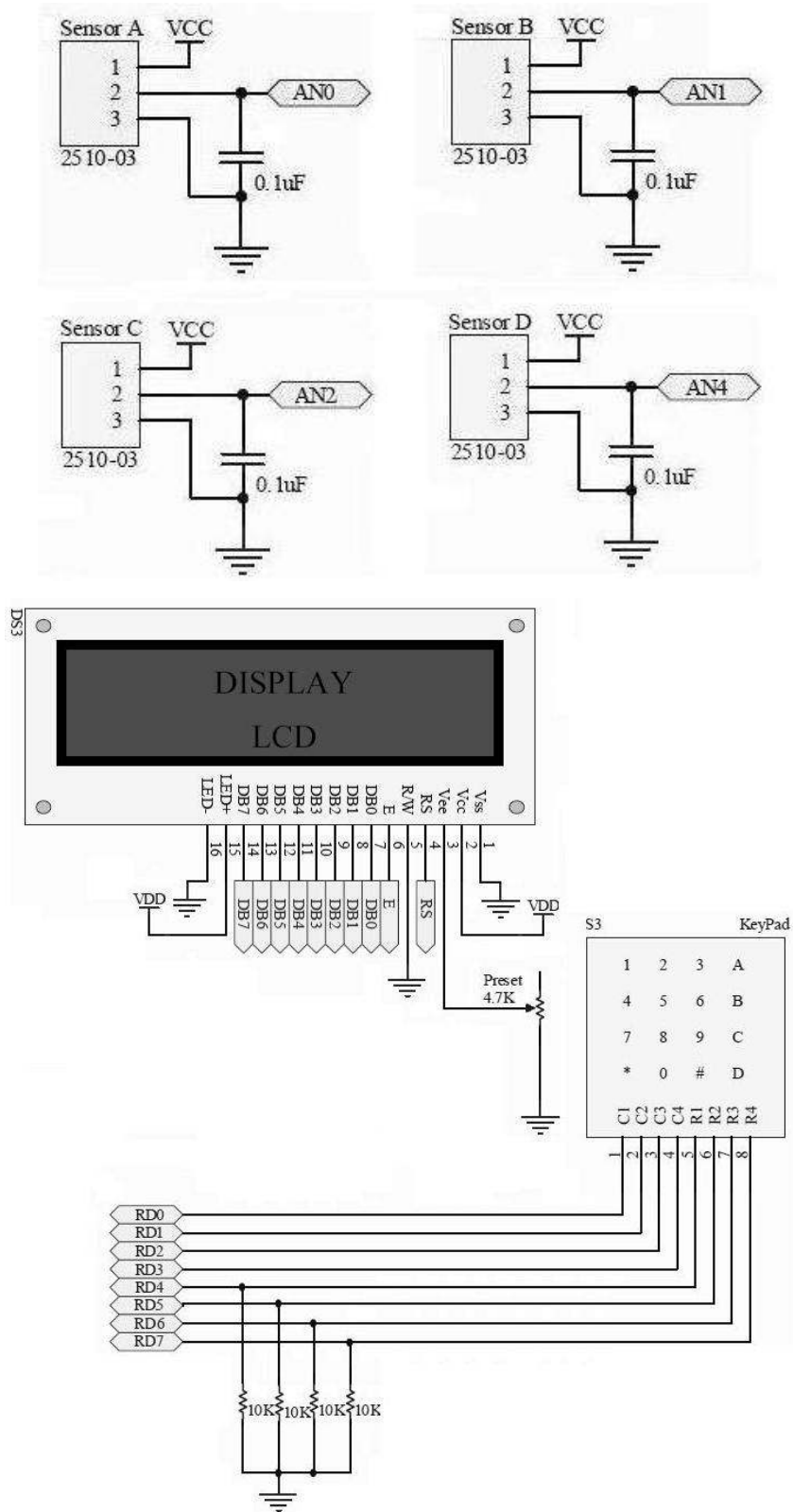


Figure 3.7: Hardware Implementation of Sensors, LCD and keypad with PIC18F4620 Microcontroller

In Figure 3.6 and Figure 3.7 show that temperature sensors are connected to pin AN0, AN1, AN2, and AN4 as the selected pins can be implemented as analogue input pins instead of digital input pins. Port B and Port E are used to interface with the LCD. As the microcontroller only sends instructions and data to the LCD without reading from it, all the pins connected are configured as the digital output pins. The keypad consists of 4 input and 4 output lines and all the lines are connected to Port D. Pin TX and RX at Port C are used to interface with mobile phone while pin SCK, SDI, SDO and RE0 are implemented to interface with the PIC18F2620 microcontroller.

The programme is developed using C programming in MPLAB environment and compiled using MPLAB C 18 C Compiler into .hex file before is written into the microcontroller. MPLAB C18 C Compiler is a cross-compiler that runs on a PC and produces code that can be executed by the Microchip PIC18 family of microcontrollers. Like an assembler, the MPLAB C18 compiler translates human-understandable statements into ones and zeros for the microcontroller to execute. Unlike an assembler, the compiler does not do a one-to-one translation of machine mnemonics into machine code.

3.2.2 Memory Card Microcontroller - PIC18F2620

PIC18F2620 microcontroller is selected as the microcontroller which responsible to write data into the SD card. The 28 pins multiple functions of the microcontroller consists of 32K× 14 words of Flash Program Memory, 3968 × 8 bytes of Data Memory (RAM), and 3 input output ports. PIC18F2620 belongs to the same family with PIC18F4620. This type of microcontroller is selected due to it has only 28 pins, which is sufficient to interface with two microcontrollers and the SD Card. Selecting microcontroller with lesser pins, small package size, and lower features reduces the implementation cost and the size of the hardware.

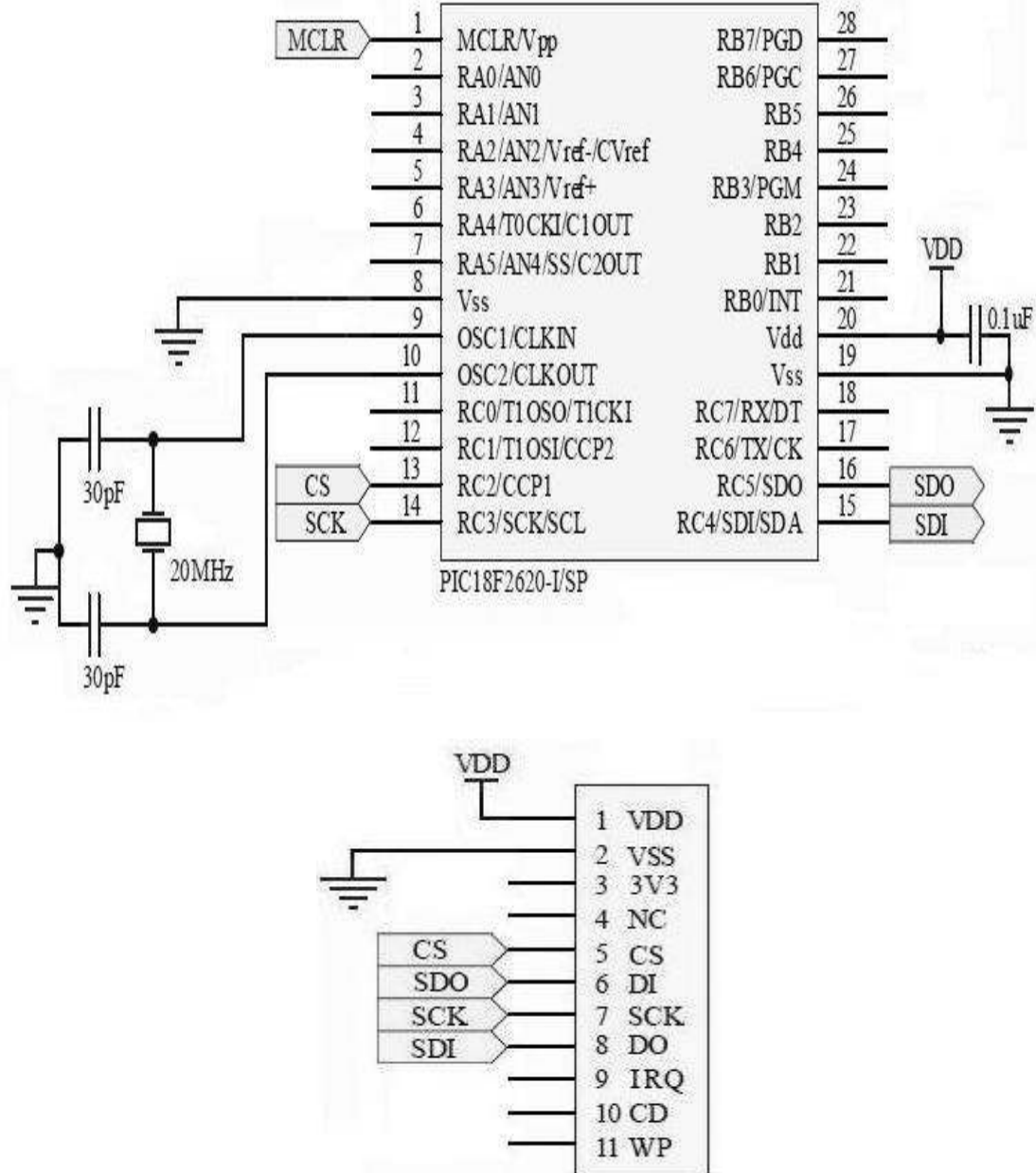


Figure 3.8: Hardware Implementation of SD Card Module with PIC18F2620 Microcontroller.

The microcontroller is connected with the SD Card using MSSP module where it functions in Master mode. Four pins are used for the interfacing, which are SCK, SDI, SDO, and RC2 pins which shown in Figure 3.8. Besides, the

microcontroller also connected with PIC18F4620 using MSSP module where it functions in Slave mode. In Slave mode, the microcontroller is connected using SCK, SDI, SDO, and (SS) pins. Data such as temperature limit, mobile number, and current temperature reading are transferred from the main microcontroller to be written into SD card. Apart from that, to write current measurement date and time, it obtains the date and time from PIC16F876A microcontroller, where the data is transferred using USART module in Receive mode.

PIC18F2620 microcontroller is programmed using mikroC Compiler in mikroC IDE environment into .hex format before is written into it. mikroC is a powerful, feature rich development tool for PICmicros and is designed to provide the programmer with the easy solution for developing applications for embedded systems, without compromising performance or control.

3.2.3 RTC Microcontroller - PIC16F876A

PIC16F876A is selected to interface with the DS1307 RTC chip and display the date and time using LCD screen. It has the feature of $8K \times 14$ words of Flash Program Memory, 368×8 bytes of Data Memory (RAM), and 3 input output ports. This 28 pins microcontroller is chosen to interface with RTC chip because of its small size and cost, as well as the functions and the input output ports provided is adequate and the memory size of this microcontroller is large enough to store the programme.

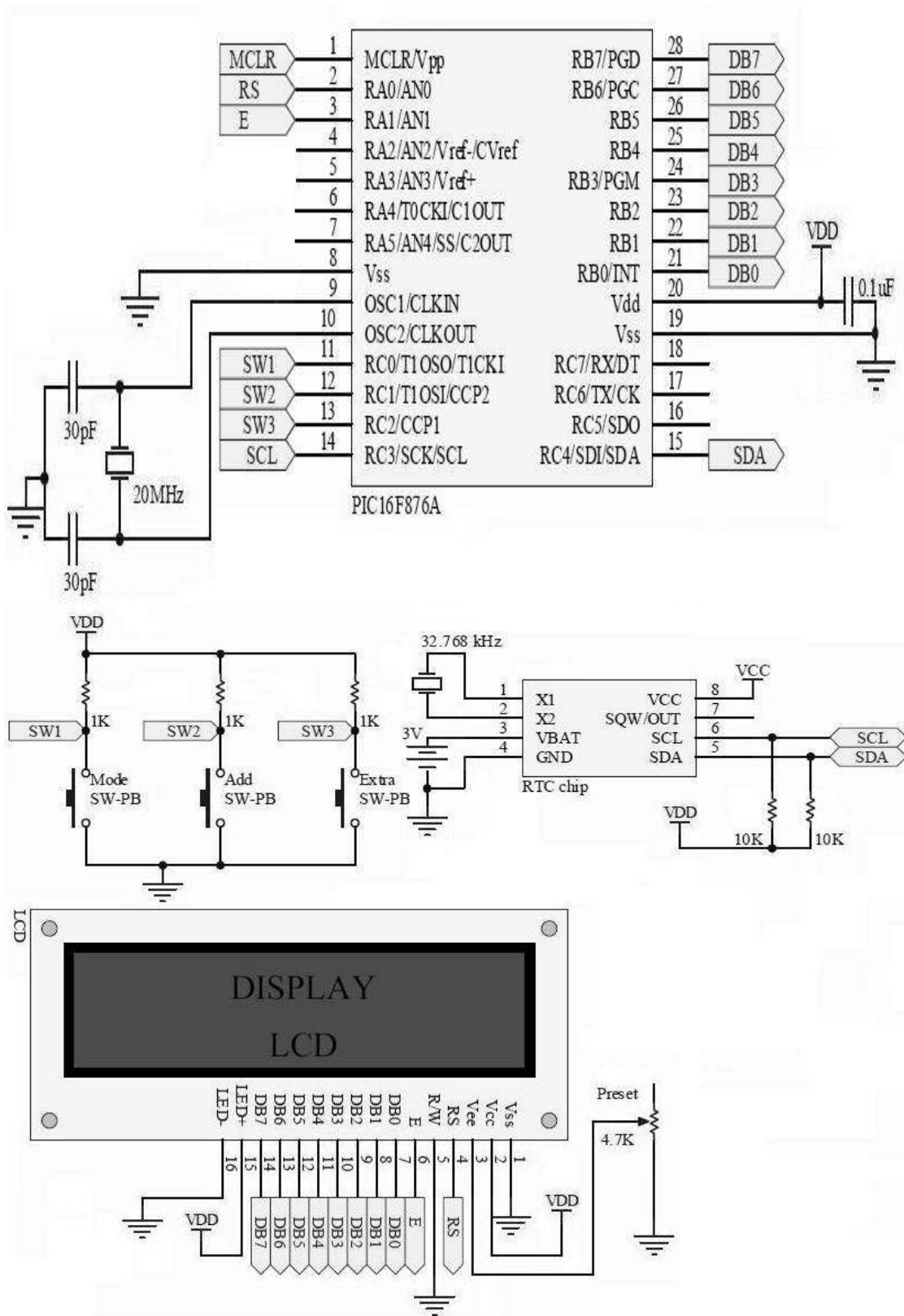


Figure 3.9: Hardware Implementation of RTC with PIC16F876A Microcontroller.

The RTC chip is connected using I2C Master mode in the MSSP module where SDA and SCL pins are used as shown in Figure 3.9. Information such as current date and time are transferred from the chip into the microcontroller. The microcontroller is then connected to the PIC18F2620 to send the current date and time using USART module in Transmit mode as shown in Figure 3.2. RTC chip is unable to be connected directly to the PIC18F2620 as the microcontroller only works in either I2C or SPI mode. PIC16F876A and PIC18F2620 microcontrollers are connected with each other using TX and RX pins, where date and time are transferred to be written into the SD card.

PIC16F876A microcontroller is programmed using MPLAB IDE environment and compiled using HI-TECH C Compiler into .hex file before it is written into the microcontroller.

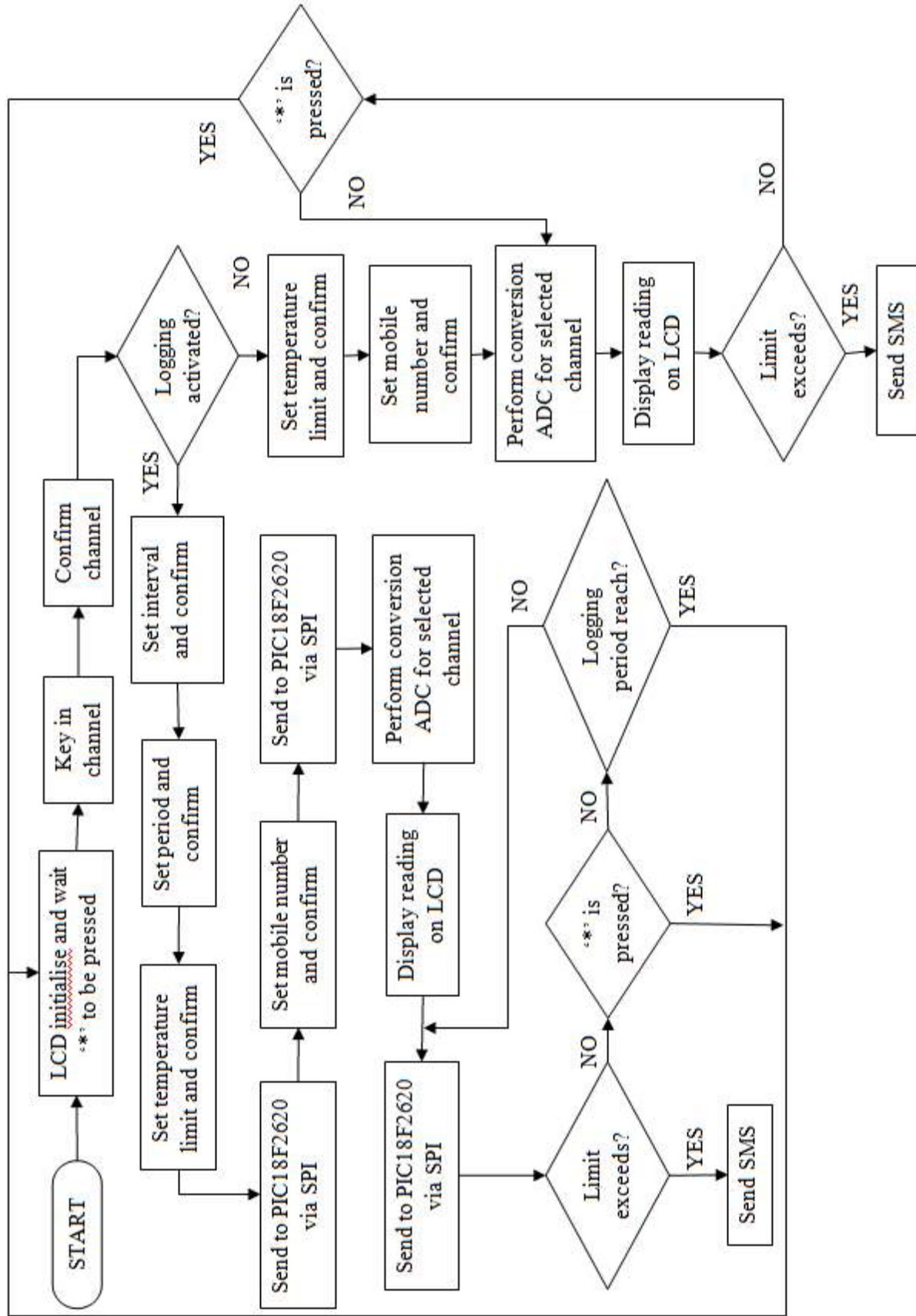


Figure 3.10: Programme Flow Diagram for PIC18F4620 Microcontroller.

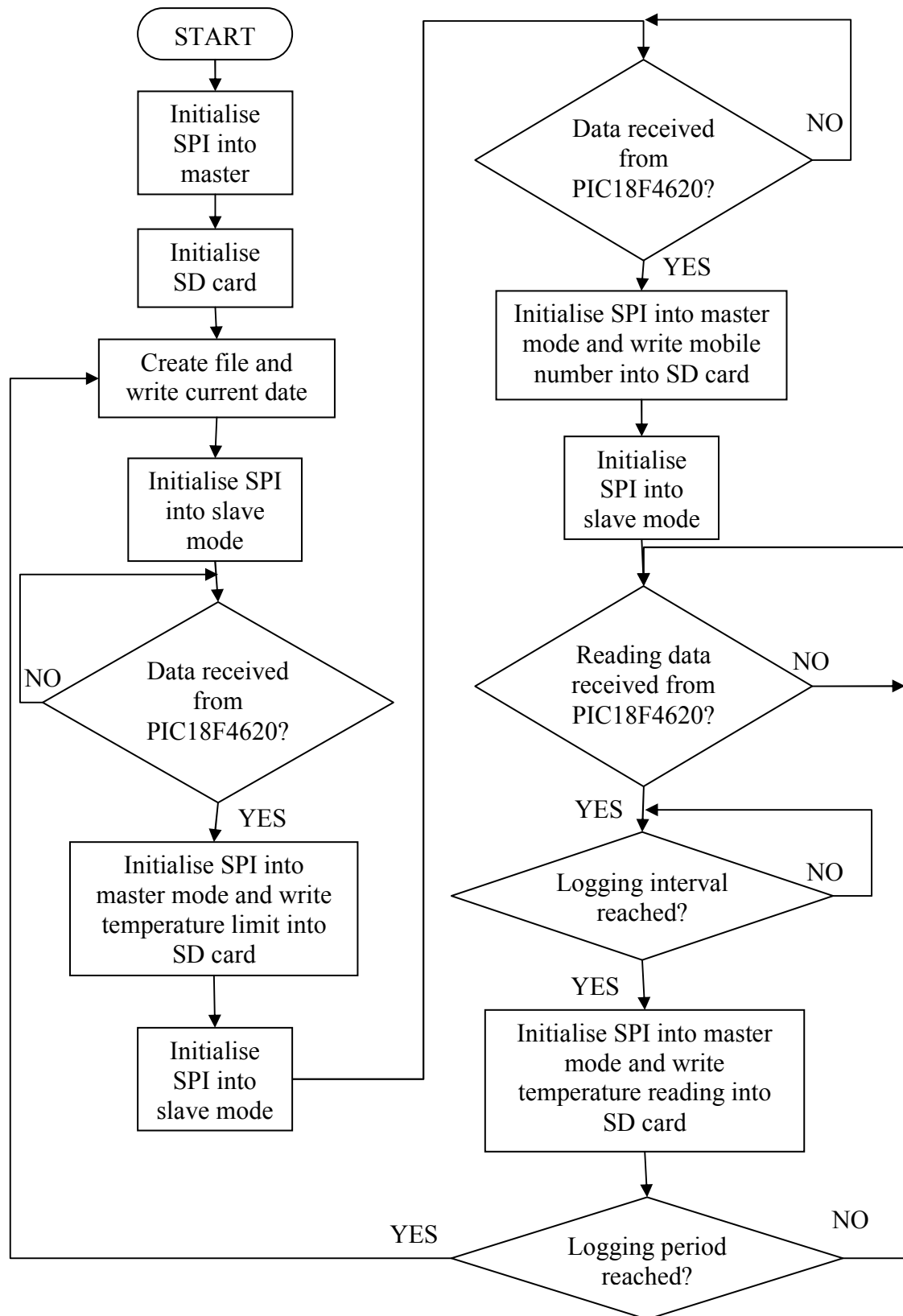


Figure 3.11: Programme Flow Diagram for PIC18F2620 Microcontroller.

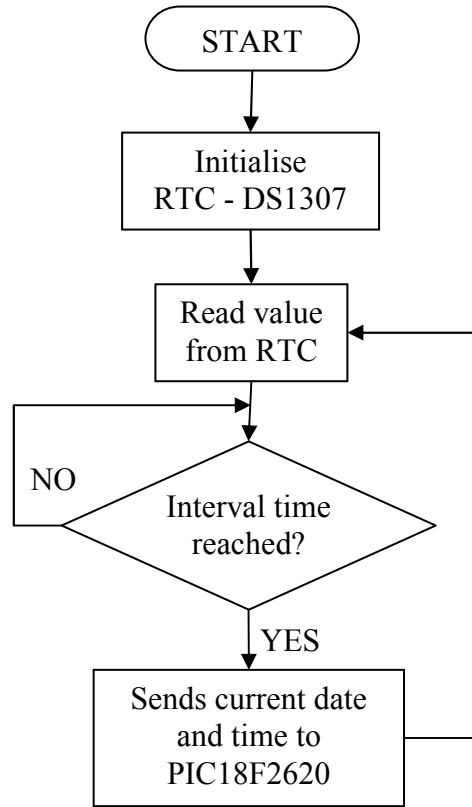


Figure 3.12: Programme Flow Diagram for PIC16F87A Microcontroller.

The overall flow diagram is the system start with all the initialization such as LCD initialization and SPI initialization as shows in the Figure 3.10, Figure 3.11 and Figure 3.12. Then, the system checks for the presence of SD Card. When a SD Card detected, the system will prompt the information such as logging interval, logging period, temperature preset value and mobile number to be transferred from PIC18F4620 to PIC18F2620 microcontrollers as shown in Figure 3.10 and written into the SD Card as shown in Figure 3.11. Meanwhile, the date and time will be transferred from PIC16F876A to PIC18F2620 microcontrollers as shown in Figure 3.12.

3.3 Flash Memory Data Storage

Flash memory is a non-volatile computer storage technology that can be electrically erased and reprogrammed in units of memory called blocks. Non-volatile is computer memory that can retain the stored information even when not powered and is typically used for the task of secondary storage, or long-term persistent storage. (Wikipedia, 2010)

Flash memory is primarily used in solid-state drives, USB flash drives, and memory cards for general storage and transfer of data between computers and other digital products. It is a variation of electrically erasable programmable read-only memory (EEPROM) unlike flash memory is erased and rewritten at the byte level, which is slower than flash memory updating. Flash memory costs far less than byte-programmable EEPROM and therefore has become the dominant technology wherever a significant amount of non-volatile, solid state storage is needed.

Memory card are commonly used for to store digital information in electronic devices such as digital cameras, mobile phones, laptop computers and also MP3 players, due to its small size, re-recordable, and can retain data without power. The most common type of memory card in use today is the SD card compared to other types such as MMC, CF card, XD card or SM card. SD card comes in capacities of up to 64 Gigabytes.

3.3.1 Secure Digital Memory Card (SD Card)

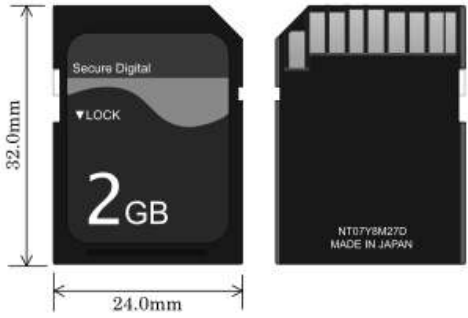
Secure Digital (SD) Card is a non-volatile memory card format developed by Panasonic, SanDisk, and Toshiba for portable devices use. The SD Card is a flash memory storage device designed to provide high-capacity, non-volatile, and rewritable storage in a small size. These devices are frequently used in many

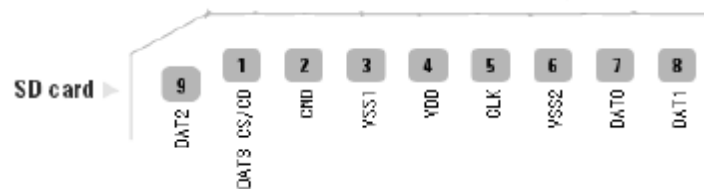
electronic consumer goods such as cameras, computers, Global Positioning (GPS) devices, mobile phones, and Personal Digital Assistants (PDA). Currently they are available at capacities from 256MB to 8GB. The SD Cards come in three sizes: standard, mini, and micro. SD Card specifications are maintained by the SD Card Association. MiniSD and microSD Cards are electrically compatible with the standard SD Cards and can be inserted in special adapters and used as standard SD Cards in standard card slots.

The SD Memory Card is based on the Multimedia Card and memory card standard and has almost the same form factor of a postage stamp. The only difference is the SD memory card is slightly thicker and has a write protection switch. MMC cards can be used in all SD Memory Card slots but not vice versa as they are thinner than SD Memory Cards. Therefore, the Secure Digital (SD) memory card is used as storage in this project due to the MMC is obsolescing.

SD Card has low power consumption, smaller physical size and reduced mechanical complexity. The pins layout of SD Card is shown in Figure 3.13. The Table 3.1 shows that SD Card specification.

Table 3.1: SD Card Specifications

	SD Card
Size	 <p>32.0mm</p> <p>24.0mm</p> <p>32×24×2.1 (mm)</p>
Area	768mm ² (100)
Card Volume	1.613mm ³ (100)
Thickness	2.1mm
Weight	Approximate 2g
Number of pins	9 pins
Operating Voltage	2.7V – 3.6V
Write - protect Switch	YES
Copyright Protection	CPRM
Capability	-
Capacity	Up to 2 GB

**Figure 3.13: SD Card Pin Layout.**

3.3.2 SD Card Implementation

SD Card interfaces with microcontroller using two different protocols, which are SD Card protocol and the SPI (Serial Peripheral Interface) protocol.

Table 3.2: SD Card Bus Protocol Vs SD Card SPI Bus Protocol

SD Card Using SD Bus	SD Card Using SPI Bus
Six-wire communication channel (clock, command, 4 data lines)	Three-wire serial data bus (Clock, dataIn, dataOut) and card specific CS signal(hardwired card selection)
Error-protected data transfer	Optional non protected data transfer mode available
Single or multiple block oriented data transfer	Single or multiple block oriented data transfer

From Table 3.2, the implementation of SD Card SPI bus protocol is chosen as 4 pins is used instead of 6 pins in used in SD Card bus protocol. Furthermore, SPI bus protocol has optional for non protected data transfer mode

The SPI protocol is being more widely used compared to SD Card protocol because SPI has simpler subset of the SD protocol for microcontrollers use. Instead of implementing the SD Card protocol which used 6 wires to interface with microcontroller, the SPI bus uses only 4-wire serial communications interface. The Serial Peripheral Interface (SPI) circuit is a synchronous serial data link that is standard across many microprocessors and other peripheral chips. For SD protocol, the data lines are reserved for data blocks with a 16 bits CRC protection which is not available in 8-bit microcontroller as in compare to SPI protocol. SPI bus is a synchronous clock shifts serial data into and out of the microcontrollers in blocks of 8 bits. Besides, both protocol using the same block oriented to transfer the data.

Besides, SPI bus is a master/slave interface which the master drives the serial clock. When using SPI, data is simultaneously transmitted and received, making it a full-duplexed protocol which allows 2 devices to communicate in both directions.

The SPI bus specifies four logic signals.

- SCK, CLK — Serial Clock (output from master)
- SDI, DI, SI — Serial Data In, Data In, Serial In
- SDO, DO, SO — Serial Data Out, Data Out, Serial Out
- nCS, CS, CSB, CSN, nSS, STE — Chip Select, Slave Transmit Enable
(active low; output from master)

The SDI/SDO (DI/DO, SI/SO) convention requires that SDO on the master be connected to SDI on the slave, and vice-versa. Chip select polarity is rarely active high, although some notations suggest otherwise which shown in Figure 3.14.

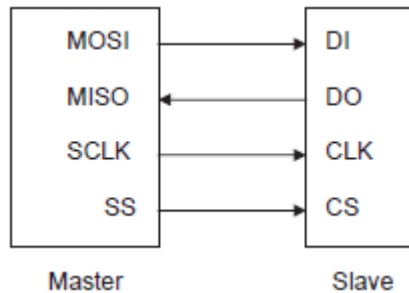


Figure 3.14: SPI Master-Slave Connection.

Two main criteria to be concerned when selecting a microcontroller for applications to write data to a media, which are its' sufficiency of RAM to support the access method and the microcontroller supports an SPI bus. SD Cards must be written in 512 byte blocks which also the sector size of the SD card. Therefore, to append a byte to a file, the sector must be read into a buffer, the location modified,

and the sector written back to disk. Hence, PIC18F2620 microcontroller was chosen to interface with the SD Card.

As shown previously in Figure 3.8, interfacing between an SD/MMC Card module and the microcontroller uses four input output lines include the three SPI bus lines (SCK, SDO and SDI), and chip select line and level translation is required between the 5 Volt input output of the PIC and the 3.3 volt input output of the SD Card. Level translation for the microcontroller SDO, SCK and CS outputs is implemented with simple resistor voltage dividers in the SD Card module. The signal from DO of the SD Card to the SDI input of the microcontroller is not straight forward and requires a TTL buffer. This is because the microcontroller, in SPI mode, has the SDI input and is configured as a Schmidt trigger input and the guaranteed logic high out of the SD Card is less than the guaranteed logic high level of the microcontroller.

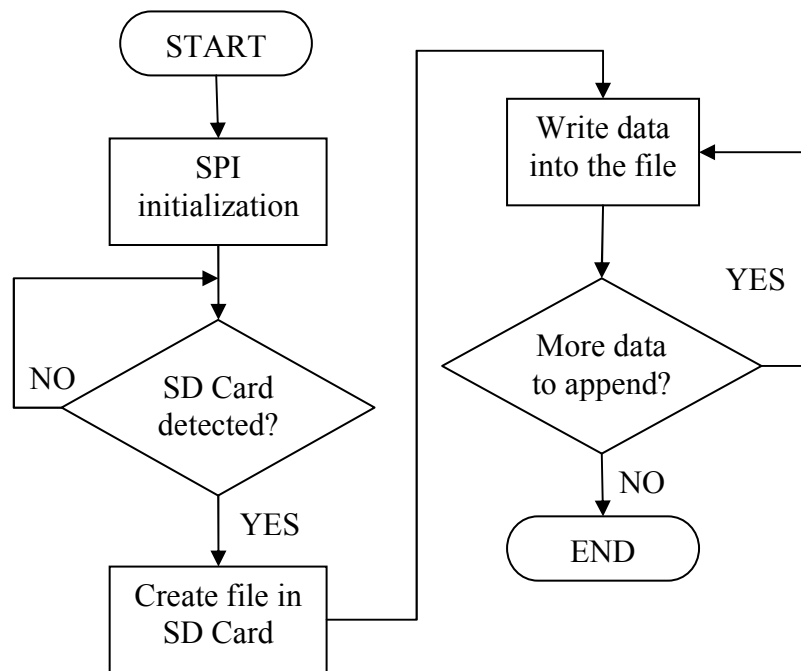


Figure 3.15: Programme Flow Diagram for SD Card Implementation.

From the flow diagram shown in Figure 3.15, the SPI is initialized before the presence of the SD Card is checked. Then, it creates a file inside the memory card with predefined format file name. The data is then written into the file. Furthermore, it continues to append the same file if more data is to be written according to the defined logging interval.

```

Spi_Init_Advanced
(MASTER_OSC_DIV16, DATA_SAMPLE_MIDDLE, CLK_IDLE_HIGH, LOW_2_HIGH);
// Loop until SD Card is detected initialized
while (Mmc_Init(&PORTC, 2));
    PORTB.f5=~PORTB.f5;

```

Figure 3.16: SPI and SD Card initialization Code.

Figure 3.16 shows the initialization subroutine for SPI module to be connected with SD Card. The MMC library in mikroC compiler provides data access on MMC via SPI serial communication and it supports SD Card standard as well. The library works only with PIC18 family, and it creates and read files from the root directory only. The SPI module must be initialized using `Spi_Init_Advanced()` subroutine before initializing SD Card using `Mmc_Init()` subroutine. SCK and SDI pins are set as input while SDO and CS, RC2 pins set as output in order to interface with the SD Card slot. The SPI module as well as the configuration code for SD Card is then initialized. The SPI module is set to master mode with the oscillation clock is 1/16 of the microcontroller oscillation frequency, the input data is chosen sampled at middle of data output time, the polarity is set to high when IDLE mode, and the data transmit occurs on transition from LOW to HIGH edge. Then the programme waits for the SD Card detection and initialisation. A LED is set at PORTB bit5 to show that SPI and SD Card initialisations are completed.

Figure 3.17 shows the file created with the `Mmc_Fat_Assign()` subroutine and `Mmc_Fat_Append()` subroutine writes the data into the file that created. The

preset temperature value is written into the file according to the channel selected to SD Card once it received the data from PIC18F4620 to PIC18F2620 microcontrollers via SPI. `rec_lim[i]` subroutine is an array to store the data from PIC18F4620 microcontroller while `Mmc_Fat_Write()` subroutine writes the data into SD Card.

```

        Mmc_Fat_Assign(&name, 0x80);
        Mmc_Fat_Append();
        PORTB.f5=~PORTB.f5;
    for (i=0;i<loop_ch;i++)
    {
        if (ch_lim[i]=='A')
        {
            Mmc_Fat_Write(file_write_tempA,sizeof(file_write_tempA));
            Mmc_Fat_Write(file_write_lim,sizeof(file_write_lim));
            array_num (rec_lim[i]);
            Mmc_Fat_Write(data_read,sizeof(data_read));
            Mmc_Fat_Write(file_write_symbol,sizeof(file_write_symbol));
            enter_write();
        }

        if ( ch_lim[i]=='B')
        {
            Mmc_Fat_Write(file_write_tempB,sizeof(file_write_tempB));
            Mmc_Fat_Write(file_write_lim,sizeof(file_write_lim));
            array_num (rec_lim[i]);
            Mmc_Fat_Write(data_read,sizeof(data_read));
            Mmc_Fat_Write(file_write_symbol,sizeof(file_write_symbol));
            enter_write();
        }

        if (ch_lim[i]=='C')
        {
            Mmc_Fat_Write(file_write_tempC,sizeof(file_write_tempC));
            Mmc_Fat_Write(file_write_lim,sizeof(file_write_lim));
            array_num (rec_lim[i]);
            Mmc_Fat_Write(data_read,sizeof(data_read));
            Mmc_Fat_Write(file_write_symbol,sizeof(file_write_symbol));
            enter_write();
        }

        if ( ch_lim[i]=='D')
        {
            Mmc_Fat_Write(file_write_tempD,sizeof(file_write_tempD));
            Mmc_Fat_Write(file_write_lim,sizeof(file_write_lim));
            array_num (rec_lim[i]);
            Mmc_Fat_Write(data_read,sizeof(data_read));
            Mmc_Fat_Write(file_write_symbol,sizeof(file_write_symbol));
            enter_write();
        }
    }

```

Figure 3.17: SD Card Data Writing Code.

3.4 Liquid Crystal Display (LCD)

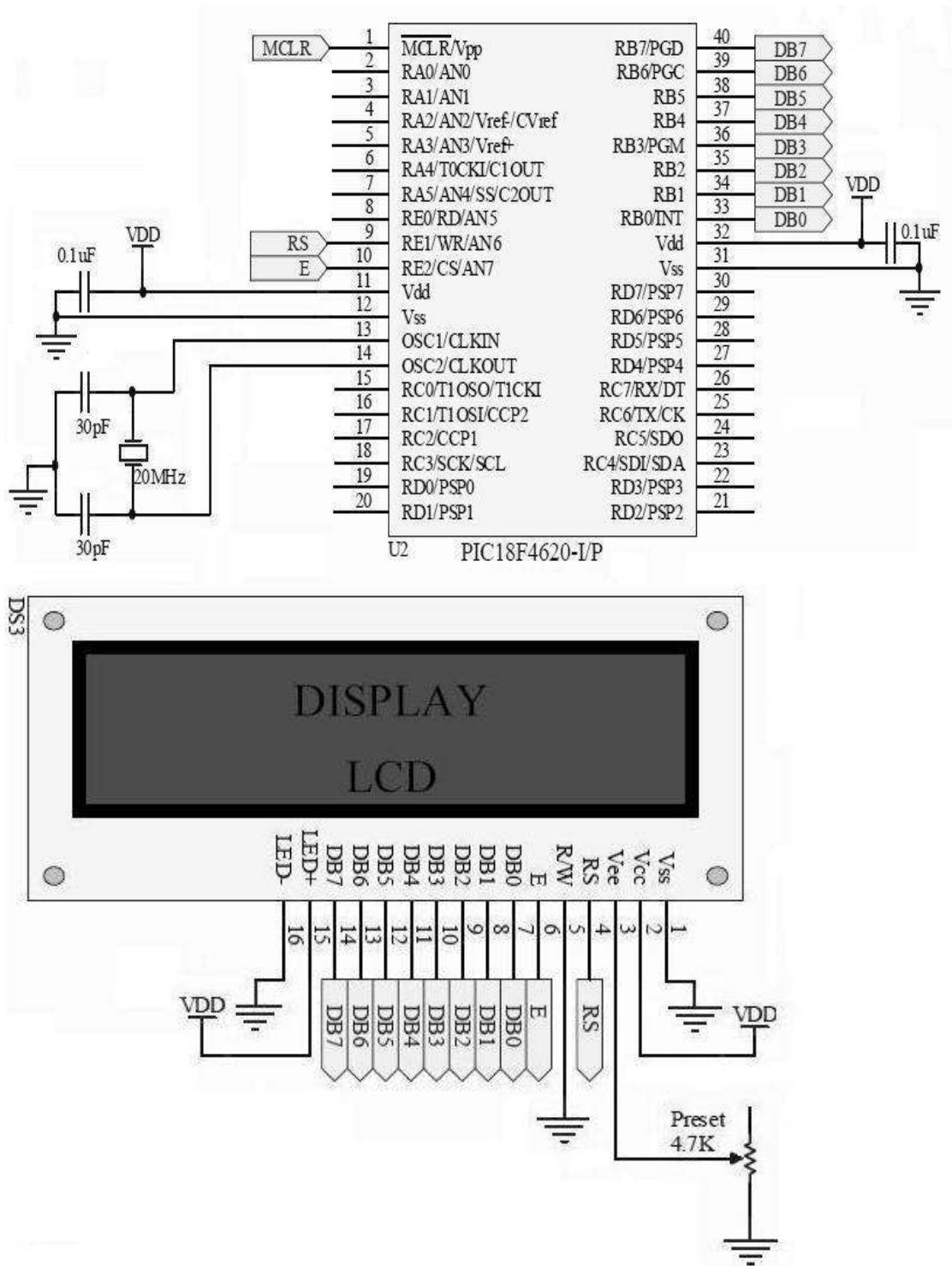


Figure 3.18: Hardware Implementation of LCD with PIC18F4620 Microcontroller.

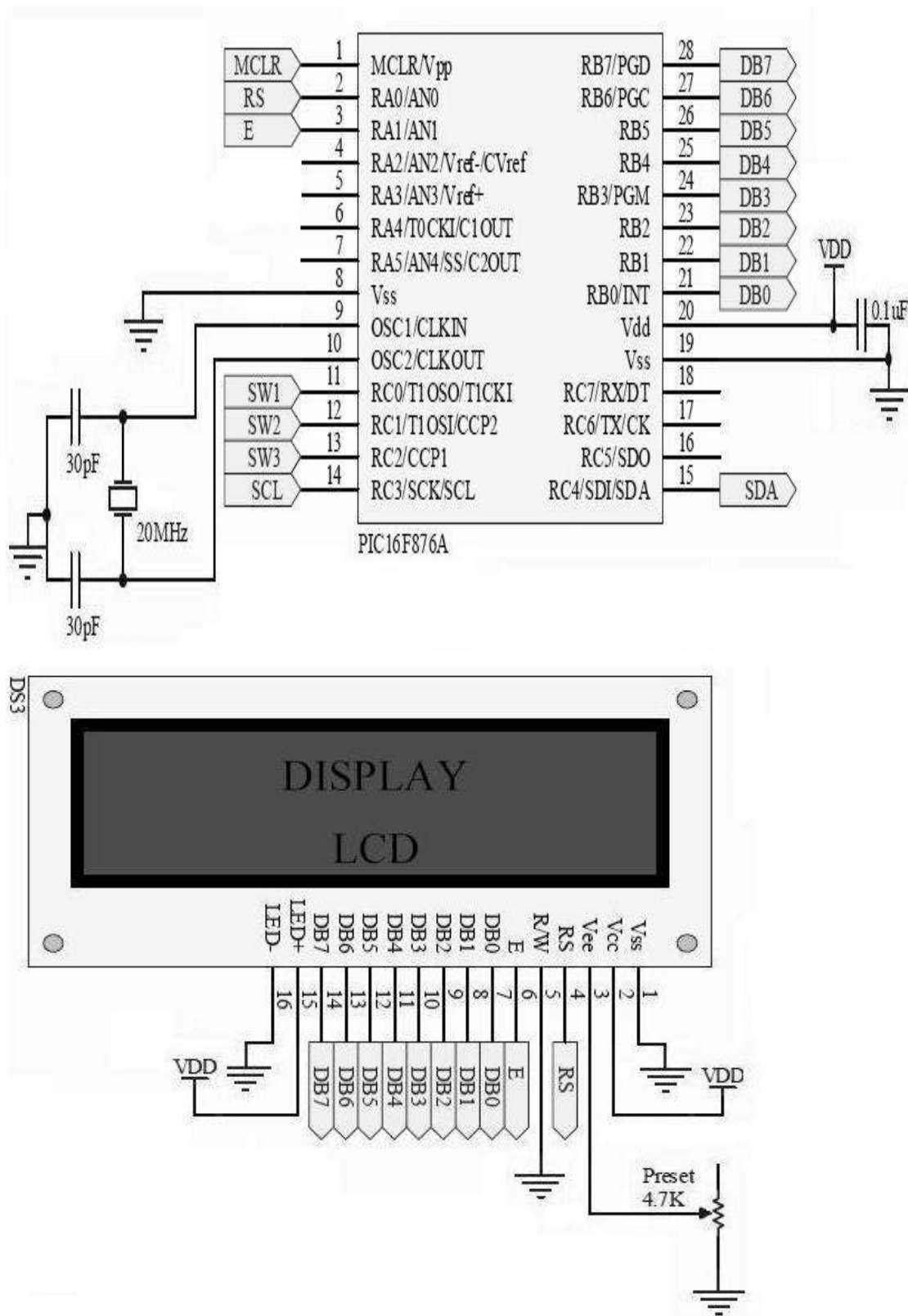


Figure 3.19: Hardware Implementation of LCD with PIC18F876A Microcontroller.

PIC18F4620 and PIC16f876A microcontrollers are connected with JHD162A LCD. For PIC18F4620 microcontroller as shown in Figure 3.18, the PORTB bits 0 to bit 7 are connected to the LCD data bit 0 to 7 with high nibble of LCD while PORTE bit 1 is connected to the LCD RS input which is register select. Besides that, PORTE bit 2 is connected to the LCD EN bit which is enable bit.

On the other hand, the LCD data bit 0 to bit 7 for RTC display are connected to PORTB bit 0 to bit 7 of PIC16F876A microcontroller, RS pin is connected to RA0, and the E pin is connected to RA1 which shows in Figure 3.19.

3.4.1 JHD162A LCD

A Hitachi 44780-Based LCD with model JHD162A is chosen for this project due to its' high quality 16 character by 2 line intelligent display module, with back lighting and works with almost any microcontroller. 2 lines are used instead of 1 line display due to the project is minimum 4 channels as shown in Figure 3.20. Therefore, the LCD will display 2 channels at one time. Besides that, the JHD162A LCD is built with 5×7 dot matrix character with cursor.



Figure 3.20: 2×16 JHD162A LCD.

Table 3.3: Hitachi 44780 Based LCD Pinout

Pin	Description
1 Vss	Ground
2 Vcc	+5V supply
3 Vee	Contrast Voltage
4 R/S	Instruction / Data Mode select
5 R/W	Read / write
6 E	Enable
7-14 D0-D7	Data lines
15	Backlit LED +V Vdd (Optional signal)
16	Backlit LED -V Vss (Optional signal)

Table 3.4: Three control lines from the microcontroller.

Control Line	Description
Enable (E)	This line allows access to the display through R/W and RS lines. When this line is low, the LCD is disabled and ignores signals from R/W and RS. When (E) line is high, the LCD checks the state of the two control lines and responds accordingly.
Read/Write(R/W)	This line determines the direction of data between the LCD and microcontroller. When it is low, data is written to the LCD. When it is high, data is read from the LCD
Register select (RS)	With the help of this line, the LCD interprets the type of data on data lines. When it is low, an instruction is being written to the LCD. When it is high, a character is being written to the LCD

3.4.2 Implementation of LCD

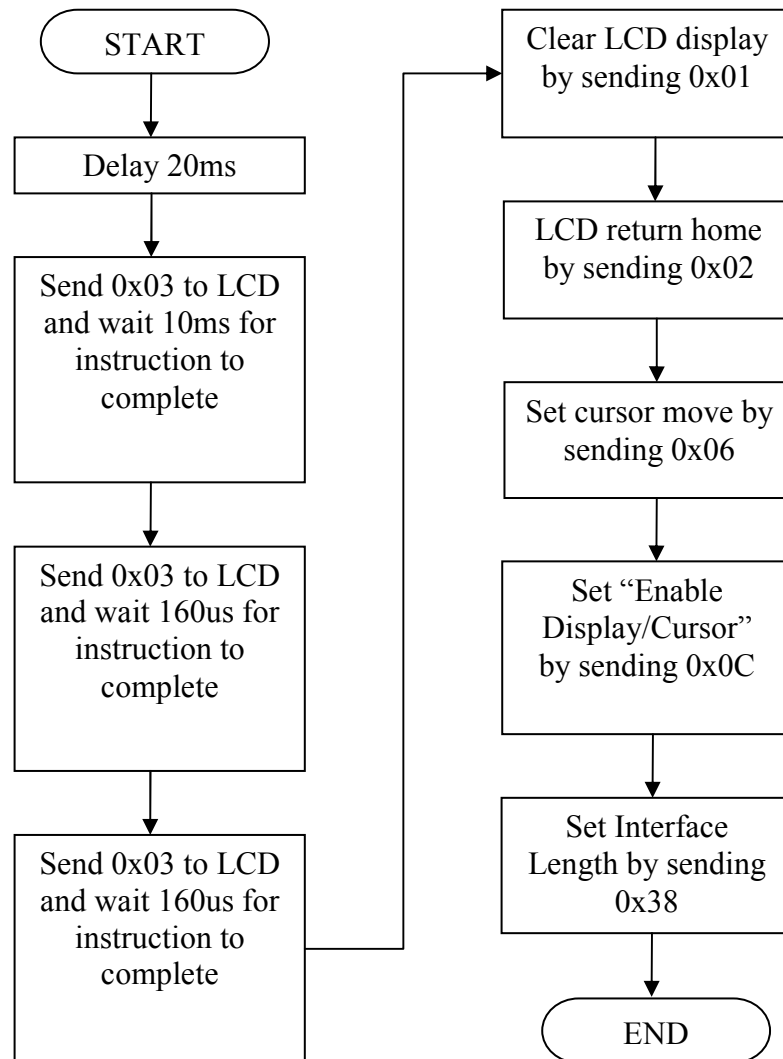


Figure 3.21: Programme Flow Diagram for Initialise the LCD.

From the flow diagram shows in Figure 3.21, the initialization of LCD starts with a 20ms delay to activate the LCD microcontroller itself. Then, it sends 0x03 thrice to ensure the LCD is reset with the cursor move to home position and ready to accept new configuration. Next, the LCD display is cleared continued with LCD cursor returns to home position. After that, the cursor move direction bit is set to move the cursor to next position after each byte of character is written to LCD, the

display is set on, the cursor is off, and the cursor blink is off. Lastly, in the interface length mode is set to 8 bits data length instead of 4 bits data length with 2 display lines. All the LCD initialization code is shown in Figure 3.22.

```

        delay_msec(2);           // delay 20ms
        send_config(0x03);       // send 0x03 to LCD
        delay_msec(1);           // delay 20ms
        send_config(0x03);       // send 0x03 to LCD
        delay_usec(8);           // delay 20ms
        send_config(0x03);       // send 0x03 to LCD
        delay_usec(8);           // delay 20ms
        send_config(0b00000001); // clear display at lcd
        send_config(0b00000010); // Lcd Return to home
        send_config(0b00000110); // entry mode-cursor increase 1
//display on, cursor off and cursor blink off
        send_config(0b00001100);
        send_config(0b00111000); //function set

```

Figure 3.22: LCD Initialization Code.

```

void send_config(unsigned char data)
{
    RS=0;
    lcd=data;
    delay(500);
    e_pulse();
}

void e_pulse(void)
{
    E=1;
    delay(500);
    E=0;
    delay(500);
}

```

Figure 3.23: Subroutine for LCD Initialization Code.

```

void disp_temp(void)
{
    send_char('T');
    send_char('e');
    send_char('m');
    send_char('p');
    send_char(' ');
}

```

Figure 3.24: LCD code for display character “TEMP”.

```

Void send_char(unsigned char data)
{
    RS=1;
    lcd=data;
    delay(500);
    e_pulse();
}

```

Figure 3.25: Subroutine for LCD to display characters.

Character on HD44780 LCD is displayed by sending its respective ASCII code. Hence to display ‘T’ on LCD microcontroller has to send 54h as data as shown in Figure 3.24. When RS pin = 0, the instruction register is selected and information on data bus is treated as commands as shown in Figure 3.23. The instruction register (IR) stores instruction codes, such as display clear and cursor shift, and address information for display data RAM (DDRAM) and character generator RAM (CGRAM). The IR can only be written from the microcontrollers.

From Figure 3.25, when RS pin=1 data register is selected and information on data bus is taken as ASCII value of respective character to be displayed on HD44780 LCD. The data register (DR) temporarily stores data to be written into DDRAM or CGRAM and temporarily stores data to be read from DDRAM or CGRAM. When

address information is written into the IR, data is written and then stored into the DR from DDRAM or CGRAM by an internal operation.

RW pin is used to either read from LCD when RW=1 or write to LCD when RW=0. As it only writes into the LCD without reading from it, therefore RW is connected to ground. When a High to Low pulse is applied on the Enable pin the information present on the data bus is latched into the LCD register as shown in Figure 3.23 and Figure 3.25

3.5 Keypad

A 4×4 matrix keypad is chosen for this project instead of 4×3 matrix keypad due to the project need ask user to choose which channel are going to use such as channel A, B, C and D as shown in Figure 3.26. Besides that 4×4 matrix keypad only consider 8 data lines to microcontroller port compare to the switching type keypad which need 16 data lines for 16 buttons. Moreover, 4x4 matrix keypad is more cheaper compare to I2C type keypad which only use 2 data line where they are SCK pin and SDI pin in microcontroller.



Figure 3.26: 4×4 Matrix Keypad.

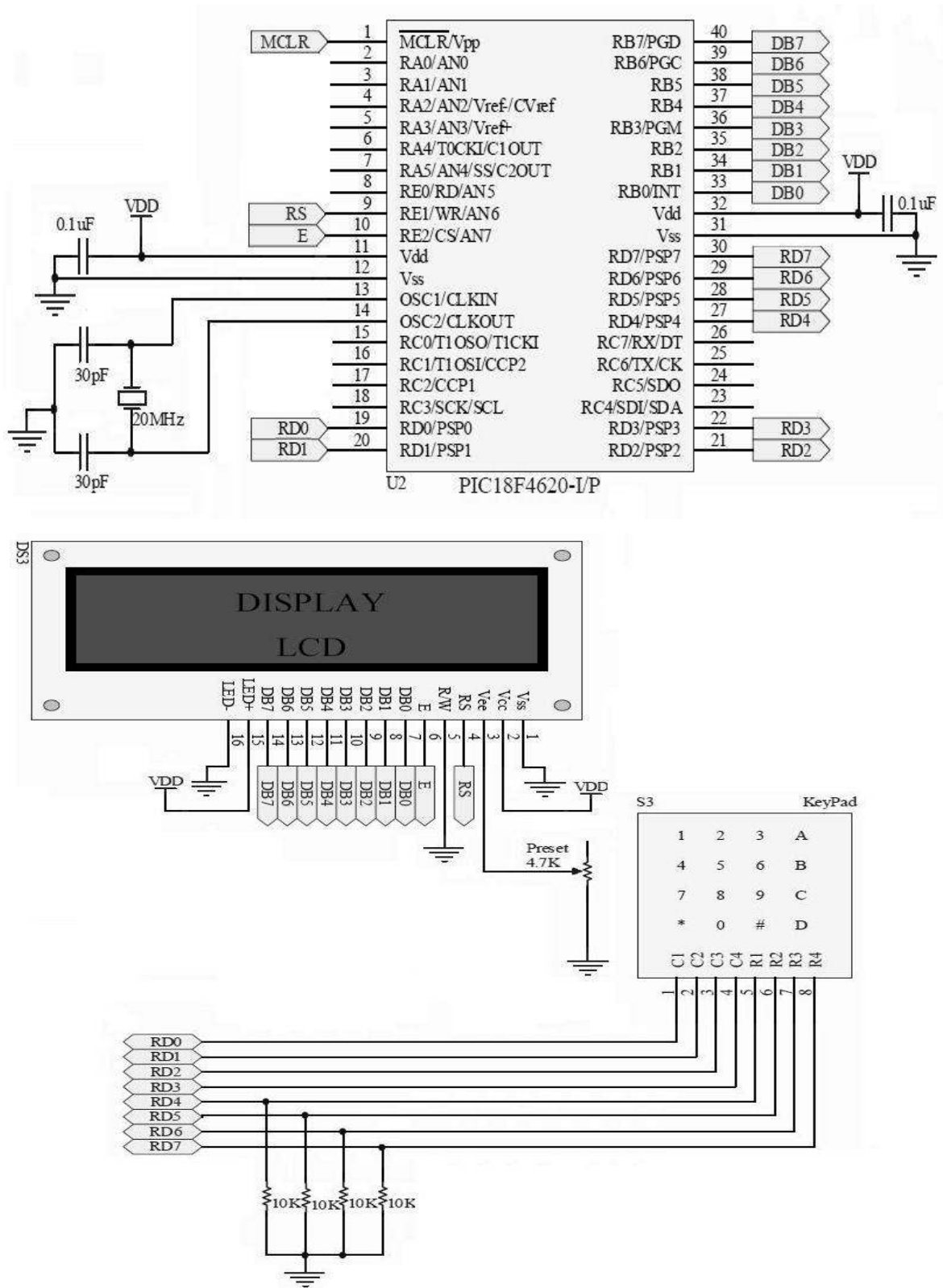


Figure 3.27: Hardware Implementation of Keypad

As shown in Figure 3.28, the keypad has four rows and four columns. In between each overlapping row and column line there is a key.

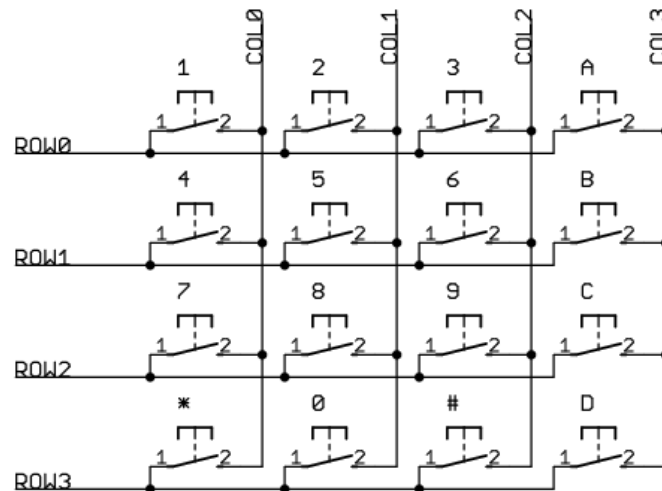


Figure 3.28: Internal Structure of Matrix Keypad.

This whole procedure of reading the keypad is called scanning. As shown in the Figure 3.26, the keypad rows are connected to PORTD bit 4 to bit 7 as input to the microcontroller while the columns are connected to PORTD bit 0 to 3 as output from microcontroller. 10 k Ω Pull-down resistors are connected at rows of keypad for the interfacing of keypad with microcontroller as shown in Figure 3.26 to allow the PIC microcontroller to read a definite value as input and the corresponding hexadecimal value of the pressed key is sent on LCD.

3.5.1 Implementation of Keypad

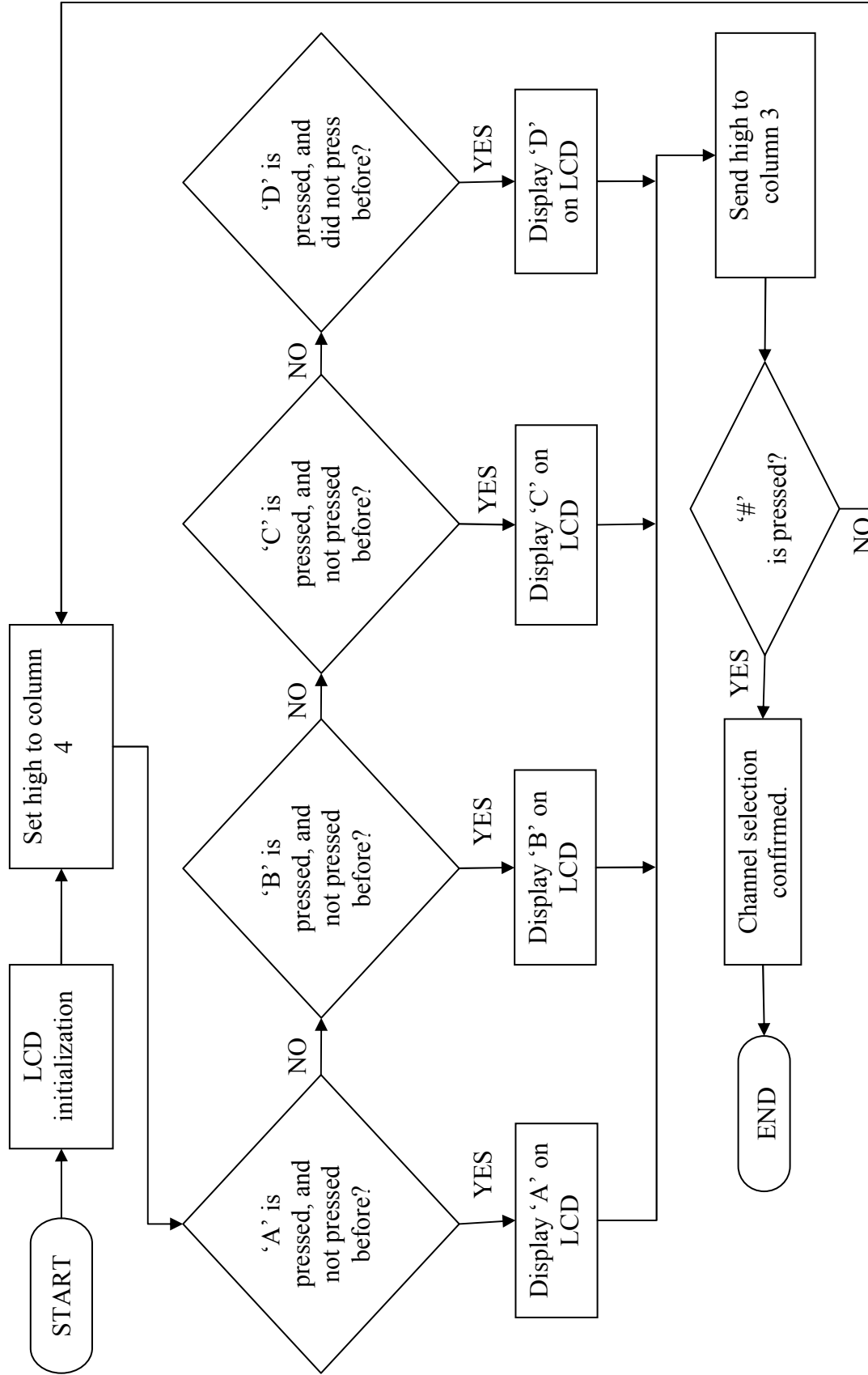


Figure 3.29: Programme Flow Diagram of Select Channel for Keypad.

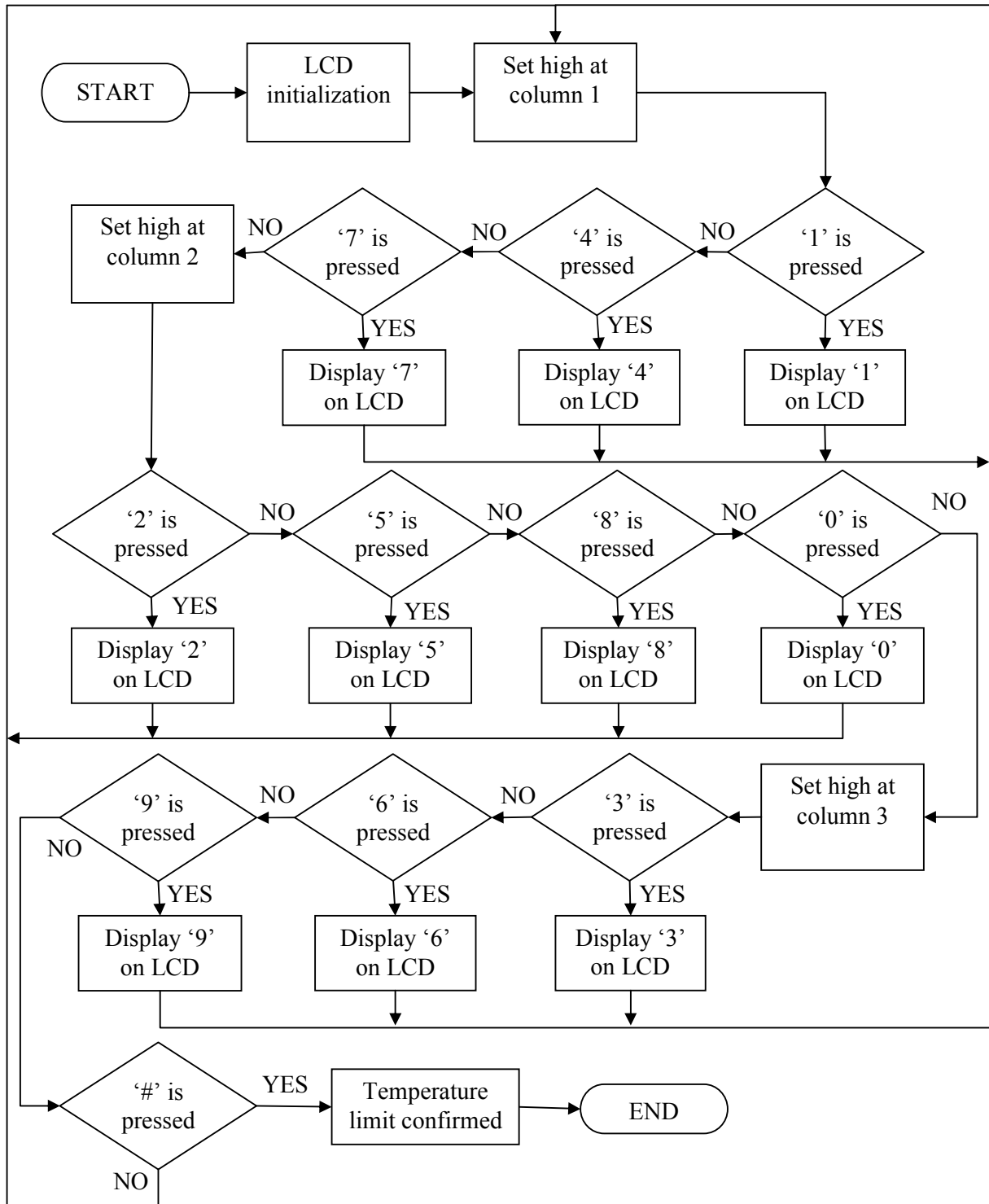


Figure 3.30: Programme Flow Diagram of Temperature Limit for Keypad.

There are two different ways to write the microcontroller subroutines for keypad input, where users are allowed to key in a non-repeatable channel numbers as shown in the flow diagram of Figure 3.29. On the other hand, the flow diagram of the keypad numbers can be repeatedly pressed to enter the temperature limit as shown in Figure 3.30.

From the flow diagram shown in Figure 3.29 and Figure 3.30, the LCD is initialized to show the character from keypad. In order to detect a key press, the microcontroller will continue scans all columns in HIGH state and activate each one by one. When a column is activated, the microcontroller scans the row is “activated”. When no key is pressed, all the row input pins remain in LOW state by the pull-down resistor. However when a key is pressed, the column pin is connected to the row pin and the microcontroller detects the particular row pin in HIGH state and display the appropriate character on the LCD.

```

while(1)
{
    PORTD=0X08;

    if(PORTDbits.RD4)
    {
        delay_msec(10);           // debouncing delay
        if(PORTDbits.RD4&&!sel_A)
        {
            send_char('A');
            sel_A=1;               // Set flat for A
            delay_msec(5);
        }
    }
    else if(PORTDbits.RD5)
    {
        delay_msec(10);           // debouncing delay
        if(PORTDbits.RD5&&!sel_B)
        {
            send_char('B');
            sel_B=1;               // Set flat for B
            delay_msec(5);
        }
    }
    else if(PORTDbits.RD6)
    {
        delay_msec(10);           // debouncing delay
        if(PORTDbits.RD6&&!sel_C)
        {
            send_char('C');
            sel_C=1;               // Set flat for C
            delay_msec(5);
        }
    }
    else if(PORTDbits.RD7)
    {
        delay_msec(10);           // debouncing delay
        if(PORTDbits.RD7&&!sel_D)
        {
            send_char('D');
            sel_D=1;               // Set flat for D
            delay_msec(5);
        }
    }
    PORTD=0X04;
    if(PORTDbits.RD7)
    {
        delay_msec(10);           // debouncing delay
    }

    // check any channel is selected

    if(PORTDbits.RD7&&(sel_A||sel_B||sel_C||sel_D))
    {
        break;
    }
}
}

```

Figure 3.31: Code for Keypad in Selecting Channel.

```
PORTD=0x01;           // check column 1
if(PORTDbits.RD4)     // check row 1
{
  delay_msec(10);
  if(PORTDbits.RD4)
  {
    send_char('1');
    input[j++]=1;
    while(PORTDbits.RD4);
  }
}
else if(PORTDbits.RD5) // check row 2
{
  delay_msec(10);
  if(PORTDbits.RD5)
  {
    send_char('4');
    input[j++]=4;
    while(PORTDbits.RD5);
  }
}
else if(PORTDbits.RD6) // check row 3
{
  delay_msec(10);
  if(PORTDbits.RD6)
  {
    send_char('7');
    input[j++]=7;
    while(PORTDbits.RD6);
  }
}
```

Figure 3.32: Code for Keypad Check for Column 1.


```
PORTD=0x02;           // check column 2
if(PORTDbits.RD4)     // check row 1
{
    delay_msec(10);
    if(PORTDbits.RD4)
    {
        send_char('2');
        input[j++]=2;
        while(PORTDbits.RD4);
    }
}
else if(PORTDbits.RD5) // check row 2
{
    delay_msec(10);
    if(PORTDbits.RD5)
    {
        send_char('5');
        input[j++]=5;
        while(PORTDbits.RD5);
    }
}
else if(PORTDbits.RD6) // check row 3
{
    delay_msec(10);
    if(PORTDbits.RD6)
    {
        send_char('8');
        input[j++]=8;
        while(PORTDbits.RD6);
    }
}
else if(PORTDbits.RD7) // check row 4
{
    delay_msec(10);
    if(PORTDbits.RD7)
    {
        send_char('0');
        input[j++]=0;
        while(PORTDbits.RD7);
    }
}
```

Figure 3.33: Code for Keypad Check for Column 2.

```

PORTD=0x04;           // check column 2
if(PORTDbits.RD4)     // check row 1
{
    delay_msec(10);
    if(PORTDbits.RD4)
    {
        send_char('3');
        input[j++]=3;
        while(PORTDbits.RD4);
    }
}
else if(PORTDbits.RD5) // check row 2
{
    delay_msec(10);
    if(PORTDbits.RD5)
    {
        send_char('6');
        input[j++]=6;
        while(PORTDbits.RD5);
    }
}
else if(PORTDbits.RD6) // check row 3
{
    delay_msec(10);
    if(PORTDbits.RD6)
    {
        send_char('9');
        input[j++]=9;
        while(PORTDbits.RD6);
    }
}
}

```

Figure 3.34: Code for Keypad Check for Column 3.

```

if(PORTDbits.RD7)
{
    delay_msec(10);
    if(PORTDbits.RD7&&j!=0)
    {
        break;
        while(PORTDbits.RD7);
    }
}
}

```

Figure 3.35: Code for Keypad for Confirmation.

Figure 3.31 shows the channel selection subroutine, where only Column 3 and Column 4 will be set to HIGH state where Column 4 is set for button A, B, C and D detection while column 3 is set to activate button '#' to confirm the channel selected.

Meanwhile in the Figure 3.32, Figure 3.33, and Figure 3.34 are the subroutine for entering the temperature limit. First, the programme will set HIGH state to Column 1 to check the rows for button 1, 4 and 7 as shown in Figure 3.32. Next, the Column 2 set to HIGH state to detect button 2, 5, 8 and 0 as shown in Figure 3.33. In order to detect button 3, 6 and 9, the Column 3 set to HIGH state which shown in Figure 3.34. At the same time, the programme checks for button '#' for temperature confirmation purpose as shown in Figure 3.35.

A debouncing short delay of 100ms is required to capture the key pressed correctly to allow the microcontroller to obtain a stable and consistent reading. The microcontroller should be able to handle multiple key inputs and one of methods is by verifying a push and pull event, in which case only the initial key pressed during that event would be accepted.

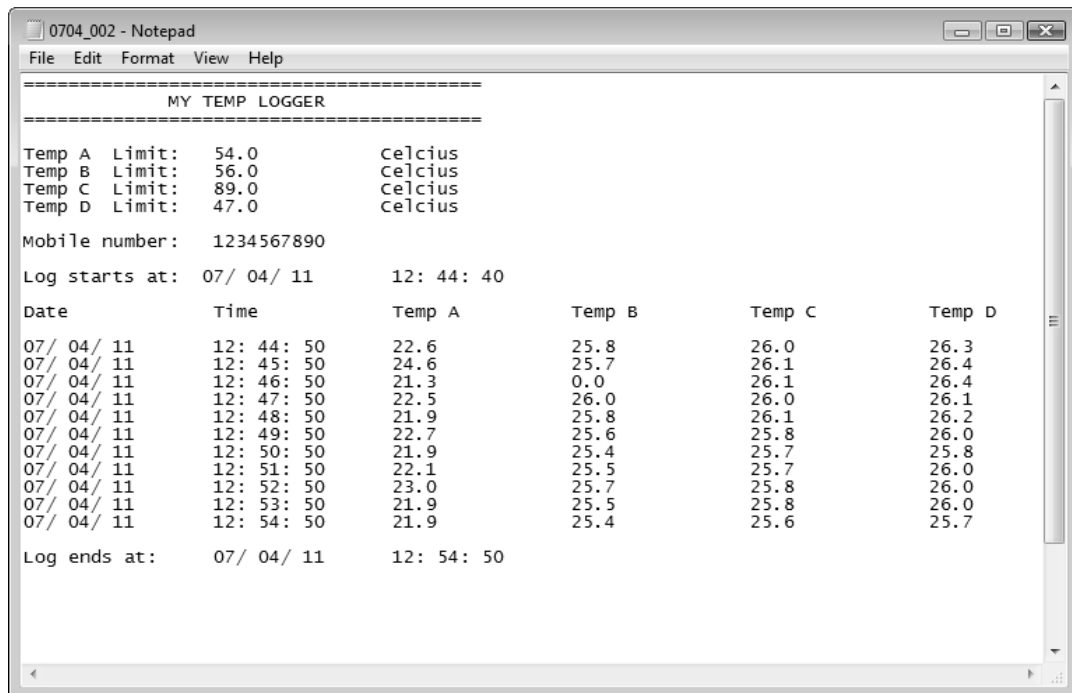
CHAPTER 4

RESULTS AND DISCUSSIONS

4.1 SD Card Testing

Testing on SPI connection between SD Card slot and microcontroller is carried out. When the SD Card detected, the 18F2620 starts to initialise the SPI and the SD Card. Then, a file name is created by using date and month which are transfer from PIC16F876A via UART module with the date and time properties of the file are defined. When the '*' key is pressed during temperature reading operation, observation shows that system is restarted and channel selection menu is displayed. A new file will be created with the file name is incremented.

After the file is created, a "MY TEMP LOGGER" file header is written into it. The preset limit and mobile number is retrieved from PIC18F4620 to PIC18F2620 microcontroller to be logged into the memory card. The log starting time also will be written into the file for timing purpose. The PIC18F2620 microcontroller will then checks the logging interval from time to time to receive the data from PIC18F4620 and PIC16F876A microcontrollers and subsequently written into the memory card. Besides that, the log ending time will also be written into the file to indicate that the data logging process is successfully performed in the logging period. The information and data are stored in .txt format as shown in Figure 4.1 and this file can also be opened in Microsoft Office Excel spreadsheet as shown in Figure 4.2.



```

0704_002 - Notepad
File Edit Format View Help
=====
MY TEMP LOGGER
=====
Temp A Limit: 54.0 Celcius
Temp B Limit: 56.0 Celcius
Temp C Limit: 89.0 Celcius
Temp D Limit: 47.0 Celcius

Mobile number: 1234567890

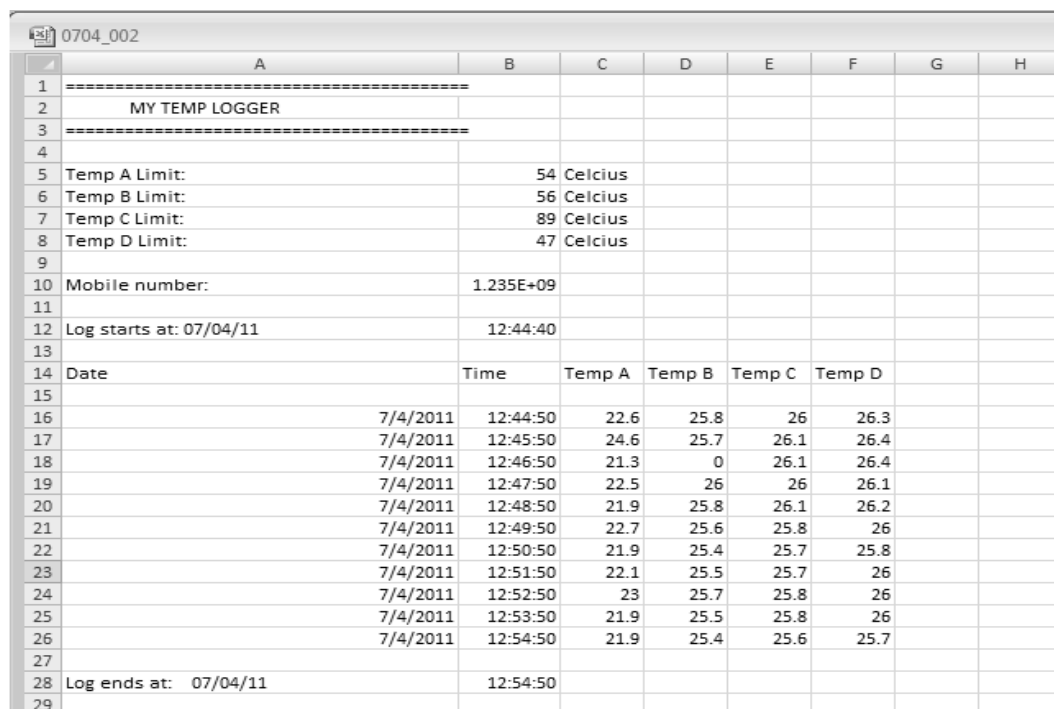
Log starts at: 07/ 04/ 11 12: 44: 40

Date          Time          Temp A      Temp B      Temp C      Temp D
07/ 04/ 11    12: 44: 50    22.6        25.8        26.0        26.3
07/ 04/ 11    12: 45: 50    24.6        25.7        26.1        26.4
07/ 04/ 11    12: 46: 50    21.3        0.0         26.1        26.4
07/ 04/ 11    12: 47: 50    22.5        26.0        26.0        26.1
07/ 04/ 11    12: 48: 50    21.9        25.8        26.1        26.2
07/ 04/ 11    12: 49: 50    22.7        25.6        25.8        26.0
07/ 04/ 11    12: 50: 50    21.9        25.4        25.7        25.8
07/ 04/ 11    12: 51: 50    22.1        25.5        25.7        26.0
07/ 04/ 11    12: 52: 50    23.0        25.7        25.8        26.0
07/ 04/ 11    12: 53: 50    21.9        25.5        25.8        26.0
07/ 04/ 11    12: 54: 50    21.9        25.4        25.6        25.7

Log ends at: 07/ 04/ 11 12: 54: 50

```

Figure 4.1: The Data Logged Inside the File.



	A	B	C	D	E	F	G	H
1	=====							
2	MY TEMP LOGGER							
3	=====							
4								
5	Temp A Limit:	54	Celcius					
6	Temp B Limit:	56	Celcius					
7	Temp C Limit:	89	Celcius					
8	Temp D Limit:	47	Celcius					
9								
10	Mobile number:	1.235E+09						
11								
12	Log starts at: 07/04/11	12:44:40						
13								
14	Date	Time	Temp A	Temp B	Temp C	Temp D		
15								
16	7/4/2011	12:44:50	22.6	25.8	26	26.3		
17	7/4/2011	12:45:50	24.6	25.7	26.1	26.4		
18	7/4/2011	12:46:50	21.3	0	26.1	26.4		
19	7/4/2011	12:47:50	22.5	26	26	26.1		
20	7/4/2011	12:48:50	21.9	25.8	26.1	26.2		
21	7/4/2011	12:49:50	22.7	25.6	25.8	26		
22	7/4/2011	12:50:50	21.9	25.4	25.7	25.8		
23	7/4/2011	12:51:50	22.1	25.5	25.7	26		
24	7/4/2011	12:52:50	23	25.7	25.8	26		
25	7/4/2011	12:53:50	21.9	25.5	25.8	26		
26	7/4/2011	12:54:50	21.9	25.4	25.6	25.7		
27								
28	Log ends at: 07/04/11	12:54:50						
29								

Figure 4.2: The .txt Format File Opened in Microsoft Office Excel Spreadsheet.

Testing is also conducted where the power supply is removed when data logging process takes place. The observation shows that the log ending time is not written into the file. The same problem occurs when the SD Card is removed abruptly from the SD Card during logging operation.

Logging process is not interfered although the GSM is activated when the temperature reading exceed the preset value. When the SD card is formatted to FAT 32, data logging process does not take in place. This proves that system only supports FAT 16 format SD card. The system is also tested with 4 GB SD card and it fails to write a single file in it. Therefore, the system only works with the SD Card of capacity less than or equal to 2 GB. Besides, the logging period will be affected when the RTC mode is set and the system will not stop logging until the '*' button is pressed to restart the system.

4.2 LCD and Keypad Implementation

The LCD starts with the message to notify user to insert SD Card to the system as shown in Figure 4.3. The system only proceeds to next screen when the SD Card is inserted and the keypad is enabled. The system then prompt the user to press '*' from the keypad to start the system as shown in Figure 4.4.

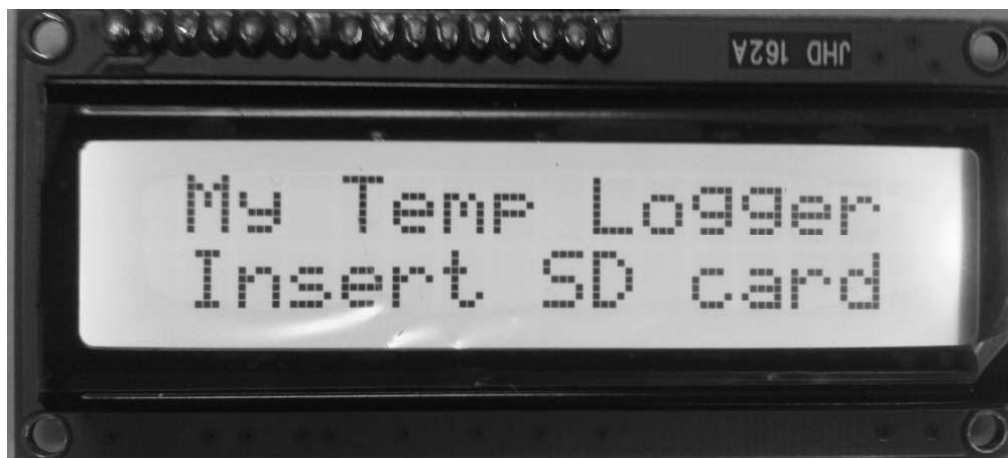


Figure 4.3: The LCD Display for a SD Card Needed to Start the System.



Figure 4.4: The LCD Display Press '*' to Start the System.

In Figure 4.5, the system prompts the user to select the channels such as A, B, C or D. In this situation the user is not allowed to press twice for the same channel. This is proved by pressing the same channel button repeatedly and the consequence is the display only shows one time on the LCD. The '#' and '*' buttons are enabled to confirm the selected channel and cancel the channel selection respectively as shown in Figure 4.7.



Figure 4.5: The LCD Displays the Channel Menu.



Figure 4.6: The LCD Shows the Entered Channel.



Figure 4.7: The LCD Display for Channel Confirmation.

The system then will prompt whether the user wants to perform data logging and again the '#' and '*' are enabled to confirm the mode which shown in Figure 4.8. When '#' is pressed, the system then prompts the user to select the logging interval which shown from Figure 4.9 to Figure 4.11 and the logging period as shown from Figure 4.12 to Figure 4.14. The possibilities combinations of logging intervals and logging periods are tested and it works well during the operation.



Figure 4.8: The LCD Displays Logging Enabling Menu.



Figure 4.9: The LCD Displays the Logging Interval Menu 1.



Figure 4.10: The LCD Displays the Logging Interval Menu 2.



Figure 4.11: The LCD Displays the Confirmation of Logging Interval.



Figure 4.12: The LCD Displays the Logging Period Menu 1.



Figure 4.13: The LCD Displays the Logging Period Menu 2.



Figure 4.14: The LCD Displays the Confirmation of Logging Period.

Next, the LCD displays the temperature preset limit if channel is selected in the beginning and same to the other three channels as shown in Figure 4.15. Then, '#' key is pressed for confirmation which shown in Figure 4.16 and Figure 4.17. After the temperature preset limit has confirmed, the PIC18F4620 microcontroller transfers the selected channel and temperature preset limit to PIC18F2620 microcontroller using SPI module for data logging into the SD Card. Meanwhile, 'Please wait' is displayed when the data is exchanged between 2 microcontrollers which shown in Figure 4.21.

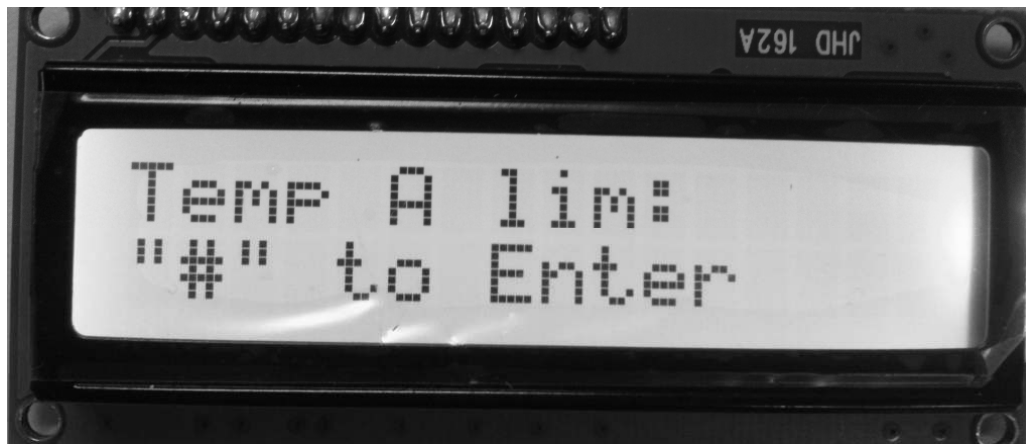


Figure 4.15: The LCD Displays the Temperature a Preset Limit Menu.



Figure 4.16: The LCD Displays the Temperature Preset Limit Menu with a Set of Keyed Limit.



Figure 4.17: The LCD Displays the Confirmation of Temperature Preset Limit.

In Figure 4.18, the system prompts the user to key in the mobile number for sending SMS to alert the user when the temperature reading is exceeded the preset limit. After the '#' key is pressed for confirmation, the mobile number is sent from the PIC18F4620 to PIC18F2620 microcontrollers using SPI module for data logging into the SD Card. Meanwhile, the 'Please wait' message is displayed when the data is exchanged between 2 microcontrollers which is shown in Figure 4.21.



Figure 4.18: The LCD Displays the Mobile Number Menu.



Figure 4.19: The LCD Displays the Mobile Number Menu with a Set of Number.



Figure 4.20: The LCD Displays the Confirmation of Mobile Number.



Figure 4.21: The LCD Display when the 2 Microcontrollers Exchange Data via SPI.

Finally, the LCD is updated repeatedly with the ADC value converted from the output voltage of the temperature sensor. The LCD displays 2 channels temperature reading at one time and switches to the other 2 channels temperature reading at the interval or 500 ms as shown in Figure 4.22 and Figure 4.23.



Figure 4.22: The LCD Display for 1st 2 Channels of Temperature Reading..



Figure 4.23: The LCD Display for 2nd 2 Channels of Temperature Reading.

The LCD and display function successfully implemented according to flow diagram in the methodology. The LCD displays the desired data accordingly without any error appear as well as the characters are cleared. Besides that, the LCD backlight can be switched off for power saving purpose.

CHAPTER 5

CONCLUSION AND RECOMMENDATIONS

5.1 Summary

In conclusion, the aim and objectives of this project have been achieved. The Multichannel Temperature Logger is successfully constructed and the whole system functions successfully. The device is able to measure the temperature range from 0 to 400 °C using K-type thermocouple and from 0 to 150 °C for LM35DZ temperature sensor. Both sensors convert the output voltage from temperature sensor to digital signal via ADC.

For SD Card, temperature limits and mobile number are entered and transferred to PIC18F2620 from PIC18F4620 microcontrollers and saved into SD card via SPI serial communication. Besides that, the logging method becomes flexible which allows user to choose the logging interval and the logging period. A short message (SMS) is sent out via GSM modem when the preset temperature limit is exceeded. RTC is also successfully implemented and the data and time sent to PIC18F2620 from 16F876A microcontrollers via UART. Lastly, all the information and data are successfully logged into SD Card in text file which can be opened in Microsoft Excel. Overall, the system was implemented successfully.

However, there are some limitations as the system itself is still imperfect with certain shortcomings which are the logging intervals and the logging period are limited to only 4 selections and not user-defined. Besides that, temperature sensor is

also one of the limitation for the Multichannel Temperature Logger due to it only support LM35DZ and K-type thermocouple.

The logger has potential to be developed further to become a low cost temperature logger in laboratory use for education purpose, where it assists the students during the experiment. Besides that, the temperature logger has the potential to monitor the environment temperature such as measuring temperature in production processes, as well as for meteorological research.

5.2 Recommendations

Further system improvement of the Multichannel Temperature Logger should be carried out to enhance its reliability, accuracy and functions. It is recommended that the analogue signal from temperature sensor can be transmitted to microcontroller via wireless instead of using a long wire. This is due the long wire has high resistance which may affect the analogue signal input to the microcontroller. The system can be operated with battery power, therefore, can be operated at least for some period when the power supply is disconnected. In order to make the system more size compact, the LCD display and keypad can be integrated into touch screen LCD. Using a single microcontroller for the whole system also reduces the size as well and increases the efficiency.

The recommended improvement for data logging is the SD Card slot can be replaced with USB data storage as the SD Card may obsolete in future. Furthermore, improvement can be made to allow user to customize file name and allow users to set own logging period and logging interval as well. System can also be improved with various type of alert system instead of only sending SMS to the user, such as calling and email with built in Wi-Fi in the system.

REFERENCES

- Data Logger. (2010). *Introduction to Data Loggers*. Retrieved July 10, 2010, from <http://www.omega.com/prodinfo/dataloggers.html>
- Dickson (2011). *Display Temperature Data Logger*. Retrieved March 25, 2011, from <http://www.dicksondata.com/products/SM320>
- Fluke (2011). *1523/1524 Reference Thermometers*. Retrieved March 25, 2011, from <http://www.hartscientific.com/products/1523-1524.htm>
- Ibrahim, D. (2008). *Advanced PIC Microcontroller Projects in C: From USB to RTOS with the PIC18F Series*. Burlington: Elsevier Ltd.
- Ibrahim, D.(2010). *SD Card Projects Using the PIC Microcontroller*. Burlington: Elsevier Ltd.
- Ladyada.net. (2011). *Temperature*. Retrieved March 2, 2011, from <http://www.ladyada.net/learn/sensors/tmp36.html>
- Microchip. (2010). *HI-TECH C for the PIC10/12/16 MCU Family*. Retrieved July 25, 2010, from http://www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE &nodeId=1406&dDocName=en542849
- Microchip. (2011). *MPLAB C Compiler for PIC18 MCUs*. Retrieved January 25, 2011, from http://www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=1406&dDocName=en010014
- Microchip. (2010). *MPLAB Integrated Development Environment* . Retrieved July 25, 2010, from http://www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=1406&dDocName=en019469&part=SW007002
- Moore, G. (1986). Data Loggers, Modern Recorders. *Electronics and Power*, 32(2),132-136, February 1986.
- Petreus, D., Juhos, Z., Pitica, D. (2006). System in Package for Temperature Logging. *Electronics Technology, 2006. ISSE '06. 29th International Spring Seminar on*, vol., no., pp.309-312, 10-14 May 2006.

- Thermometrics. (2011). *Platinum Resistance Thermometer*. Retrieved March 2, 2011, from <http://www.thermometricscorp.com/platresther.html>
- Tooley, M. (2006). *Electronic Circuits Fundamentals and Applications*. Burlington: Elsevier Ltd.
- MicroDAQ.com. (2011). *Differential Temperature Data Logger*. Retrieved March 10, 2011, from <http://www.microdaq.com/extech/temperature/differential-temp-logger.php>
- MicroDAQ.com. (2011). *TandD Thermo Recorder Data Logger*. Retrieved March 10, 2011, from http://www.microdaq.com/tandd/wide_range_data_logger.php
- Omega. (2011). *Introduction to Data Logger*. Retrieved March 25, 2011, from <http://www.omega.com/prodinfo/dataloggers.html>
- Omega. (2011). *Datalogger Thermometer with USB and RS232*. Retrieved March 25, 2011, from <http://www.omega.com/pptst/HH306A.html>
- SD Association. (2010). *SD Technology Overview*. Retrieved July 20, 2010, from <http://www.sdcard.org/developers/tech/>
- SD Association. (2010) *SD Card*. Retrieved July 21, 2010, from <http://www.sdcard.org/developers/tech/sdcard/>
- Two Dimensional Instruments. (2011). *Therma Viewer. The best solution for monitoring temperature*. Retrieved March 25, 2011, from <http://e2di.com/PDFs/White-paper-on-paperless-chart-recorders.htm>
- United Electronic Industries, Inc. (2010). *Data Logger [definition and description]*. Retrieved July 10, 2010, from <http://www.ueidaq.com/data-logger.html>
- Varalakshmi. (2011). *Thermocouples*. Retrieved March 2, 2011, from <http://www.articlesnatch.com/Article/Thermocouples/1724971>
- Wikipedia. (2010) *Chart Recorder*. Retrieved July 21, 2010, from http://en.wikipedia.org/wiki/Data_logging

APPENDICES

APPENDIX A: Liquid Crystal Display (LCD)

R/S	R/W	D7	D6	D5	D4	D3	D2	D1	D0	Instruction/Description
4	5	14	13	12	11	10	9	8	7	Pins
0	0	0	0	0	0	0	0	0	1	Clear Display
0	0	0	0	0	0	0	0	1	*	Return Cursor and LCD to Home Position
0	0	0	0	0	0	0	1	ID	S	Set Cursor Move Direction
0	0	0	0	0	0	1	D	C	B	Enable Display/Cursor
0	0	0	0	0	1	SC	RL	*	*	Move Cursor/Shift Display
0	0	0	0	1	DL	N	F	*	*	Set Interface Length
0	0	0	1	A	A	A	A	A	A	Move Cursor into CGRAM
0	0	1	A	A	A	A	A	A	A	Move Cursor to Display
0	1	BF	*	*	*	*	*	*	*	Poll the "Busy Flag"
1	0	D	D	D	D	D	D	D	D	Write a Character to the Display at the Current Cursor Position
1	1	D	D	D	D	D	D	D	D	Read the Character on the Display at the Current Cursor Position

Instruction for HD44780 Hitachi LCD.

The bit descriptions for the different commands are:

"*" - Not Used/Ignored. This bit can be either "1" or "0"

Set Cursor Move Direction:

ID - Increment the Cursor After Each Byte Written to Display if Set
S - Shift Display when Byte Written to Display

Enable Display/Cursor

D - Turn Display On(1)/Off(0)
C - Turn Cursor On(1)/Off(0)
B - Cursor Blink On(1)/Off(0)

Move Cursor/Shift Display

SC - Display Shift On(1)/Off(0)
RL - Direction of Shift Right(1)/Left(0)

Set Interface Length

DL - Set Data Interface Length 8(1)/4(0)
N - Number of Display Lines 1(0)/2(1)
F - Character Font 5x10(1)/5x7(0)

Poll the "Busy Flag"

BF - This bit is set while the LCD is processing

Move Cursor to CGRAM/Display

A - Address

Read/Write ASCII to the Display

D - Data

Instruction Description for HD44780 Hitachi LCD.

Upper 4 bits Lower 4 bits	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
xxxx0000	CG RAM (1)		0	@P`P							-	タ	ニ	α	p	
xxxx0001	(2)		!	1AQa9							。	ア	チ	△	ä	q
xxxx0010	(3)		"	2BRbr							「	イ	ツ	×	β	θ
xxxx0011	(4)		#	3CScs							」	ウ	テ	ε	ε	ω
xxxx0100	(5)		\$	4DTdt							、	エ	ト	φ	μ	Ω
xxxx0101	(6)		%	5EUeu							・	オ	ナ	1	ε	Ü
xxxx0110	(7)		&	6FVfv							ヲ	カ	ニ	ヨ	ρ	Σ
xxxx0111	(8)		'	7GWgw							フ	キ	ヌ	ラ	g	π
xxxx1000	(1)		<	8HXhx							イ	ク	ネ	リ	√	×
xxxx1001	(2)		>	9IYiy							ウ	ケ	ノ	ル	'	∪
xxxx1010	(3)		*	:JZjz							エ	コ	ノ	レ	j	≠
xxxx1011	(4)		+	;K[k<							オ	サ	ヒ	ロ	*	≠
xxxx1100	(5)		,	<L[l							カ	シ	フ	ワ	φ	≠
xxxx1101	(6)		-	=M]m}							ユ	ズ	ヘ	ン	≠	÷
xxxx1110	(7)		.	>N^n+							ヨ	セ	ホ	°	ñ	
xxxx1111	(8)		/	?O_oe							ッ	ソ	マ	°	ö	■

Correspondences between Character Codes and Character patterns.

line	80	81	82	83	84	85	86	87	88	89	8A	8B	8C	8D	8E	8F
1	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H
line	C0	C1	C2	C3	C4	C5	C6	C7	C8	C9	CA	CB	CC	CD	CE	CF
2	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H

Addresses of Cursor Position for 16x2 JHD162A LCD

APPENDIX B: Project Planning

