ARDUINO-BASED TEMPERATURE MONITORING AND CONTROL VIA CAN BUS

MOHAMMAD HUZAIFAH BIN CHE MANAF

UNIVERSITI MALAYSIA PAHANG

ARDUINO-BASED TEMPERATURE MONITORING AND CONTROL VIA CAN BUS

MOHAMMAD HUZAIFAH BIN CHE MANAF

This thesis is submitted as partial fulfilment of the requirements for the award of the Bachelor of Electrical Engineering (Hons.) (Electronics)

Faculty of Electrical & Electronics Engineering
Universiti Malaysia Pahang

DECEMBER 2016

UNIVERSITI MALAYSIA PAHANG

DECLARATION OF THESIS AND COPYRIGHT

Author's Full Name : MUHAMMAD HUZAIFAH BIN CHE MANAF

Date of Birth : 920504115621

Title : ARDUINO BASED TEMPERATURE MONITORING AND CONTROL VIA CAN BUS

Academic Session : 2016/2017

I declare that this thesis is classified as:

☐ CONFIDENTIAL	(Contains confidential information under the Official Secret Act 1997)*
☐ RESTRICTED	(Contains restricted information as specified by the organization where research was done)*
☑ OPEN ACCESS	I agree that my thesis to be published as online open access (Full Text)

I acknowledge that Universiti Malaysia Pahang reserve the right as follows:

1. The Thesis is the Property of Universiti Malaysia Pahang
2. The Library of Universiti Malaysia Pahang has the right to make copies for the purpose of research only.
3. The Library has the right to make copies of the thesis for academic exchange.

Certified by:

huzaifah

(Student's Signature)

920504115621

New IC/Passport Number
Date: 30th DECEMBER 2016

(Supervisor's Signature)

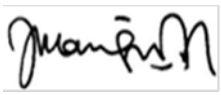
MAZIYAH BINTI MAT NOH

Name of Supervisor
Date: 30th DECEMBER 2016

NOTE : * If the thesis is CONFIDENTIAL or RESTRICTED, please attach with the letter page 2 from the organization with the period and reasons for confidentiality or restriction.

SUPERVISOR'S DECLARATION

I hereby declare that I have checked this thesis and in my opinion, this thesis is adequate in terms of scope and quality for the award of the degree of the Bachelor Degree of Electrical Engineering (Hons.) (Electronics).

Signature	:	
Name of Supervisor	:	MAZIYAH BINTI MAT NOH
Position	:	LECTURER OF ELECTRICAL & ELECTRONICS ENGINEERING
Date	:	30th DECEMBER 2016

STUDENT'S DECLARATION

I hereby declare that the work in this thesis is my own except for quotations and summaries which have been duly acknowledged. The thesis has not been accepted for any degree and is not concurrently submitted for award of other degree.

Signature	:	huzaifah
Name	:	MOHAMMAD HUZAIFAH BIN CHE MANAF
ID Number	:	EA11098
Date	:	30th DECEMBER 2016

ACKNOWLEDGMENTS

It is a pleasure to thank many people who made this thesis possible. I would like to take this opportunity to express my gratitude and sincere thanks to my supervisor Madam Maziyah Binti Mat Noh for her guidance, insight, and support he has provided throughout the course of this work. I learned about the great role of self-learning and the constant drive for understanding emerging technologies, and a passion for knowledge.

My special thanks go to research scholars, friends and juniors at Universiti Malaysia Pahang for their encouragement and help throughout the course. I would like to thank all faculty members and staff of the Falcuty of Electrical and Electronics Engineering, Universiti Malaysia Pahang for their extreme help throughout course.

Finally, I am forever indebted to my parents for their love, understanding, endless patience and encouragement when it was most required.

ABSTRACT

This project presents the Arduino-based Temperature Monitoring and Control via CAN Bus. Controller Area Network (CAN) bus that is designed to allow microcontrollers and other electronic devices to communicate with each other within a vehicle without a host computer. It was originally designed for multiplex electrical wiring within automobile, but also used in many other contexts. The desired temperature is obtained by controlling the ON and OFF states of the heater and fan. The temperature is controlled using Arduino Uno that is interfaced with the plant using CAN Bus that provides the communication between Arduino and LCD display.

ABSTRAK

Projek ini membentangkan Pemantauan suhu berasaskan Arduino dan Kawalan melalui CAN Bus. Rangkaian kawasan pengawal (CAN) bas yang direka untuk membenarkan microcontrol-Lers dan peranti elektronik lain untuk berkomunikasi antara satu sama lain di dalam kenderaan dengan keluar komputer hos. Ia pada asalnya direka untuk pendawaian elektrik multipleks dalam kereta, tetapi juga digunakan dalam pelbagai konteks yang lain. suhu yang dikehendaki diperoleh dengan mengawal ON dan OFF negeri pemanas dan kipas. suhu adalah con-troll menggunakan Arduino Uno yang berantara muka dengan kilang menggunakan CAN Bus yang pro-vides komunikasi antara Arduino dan paparan LCD.

TABLE OF CONTENTS

		Page
SUPERVISOR'S DECLARATION		ii
STUDENT'S DECLARATION		iii
ACKNOWLEDGEMENT		iv
ABSTRACT		v
ABSTRAK		vi
TABLE OF CONTENTS		vii
LIST OF TABLES		ix
LIST OF FIGURES		x
LIST OF SYMBOLS		xii
LIST OF ABBREVIATIONS		xiii
CHAPTER 1	**INTRODUCTION**	
1.1	Background	1
1.2	Problem Statement	1
1.3	Project Objective	2
1.4	Scope of The Project	2
CHAPTER 2	**LITERATURE REVIEW**	
2.1	Introduction	3
2.2	CAN Bus	3
2.3	A Zigbee Based Temperature Monitoring System	4
2.4	Arduino Based Can Protocol Implementation In Vechicle Control System	4
2.5	Monitoring and Controlling of Temperature Using Hardware Description & Operation	5
CHAPTER 3	**METHODOLOGY**	
3.1	Introduction	6
3.2	Flow Chart	8
3.3	Block Diagram	9

3.4	System Design		8
	3.4.1	Microcontroller	10
	3.4.2	CAN Bus Shield	11
	3.4.3	Display	12
	3.4.4	DS18B20 Temperature Sensor	12
	3.4.5	LM35 Temperature Sensor	13
	3.4.6	Controller and Transceiver	14
	3.4.7	Relay	16
	3.4.8	Heater	16
	3.4.9	Potentiometer	17
	3.4.10	USB Fan	17
3.5	Prototype Circuit System Design		18
3.6	Software Development and Coding		18
3.7	Hardware Simulation		21
CHAPTER 4	**RESULTS AND ANALYSIS**		
4.1	Introduction		24
4.2	Fan		25
4.3	DC Water Pump		28
4.4	Analysis		31
CHAPTER 5	**CONCLUSION AND RECOMMENDATION**		
5.1	Conclusions		32
5.2	Recommendation		32
REFERENCES			33
APPENDICES			34
A	Appendix A		34
B	Appendix B		39
C	Appendix C		40

LIST OF TABLES

Table No.	Title	Page
4.1	The result by using Fan (distance 18cm)	26
4.2	The result by using Fan (distance 7cm)	26
4.3	The result by using Water Pump (5V)	29
4.4	The result by using Water Pump (9V)	29

LIST OF FIGURES

Figure No.	Title	Page
3.1	Flow chart of the study	8
3.2.	Block diagram of implemented system	9
3.3.1	Arduino UNO board	10
3.3.2	CAN-Bus Shield	11
3.3.3	LCD Display 2x16	12
3.3.4	DS18B20 Temperature sensor	13
3.3.5	LM35 Temperature sensor	14
3.3.6	MCP2515 Controller	15
3.3.7	MCP2551 Transceiver	15
3.3.8	Relay	16
3.3.9	Heating element	16
3.3.10	Potentiometer 10kΩ	17
3.3.11	Fan	17
3.4	Prototype hardware design	18
3.5	Hooked up between Arduino Uno and CAN Bus Shield	23
3.6	CAN Receiver Flow Chart	23
4.1	Example of Data Read by Sensor	24
4.2	The Fan hardware setup	25
4.3	Graph (distance 18cm)	27
4.4	Graph (distance 7cm)	27
4.5	The DC water pump hardware setup	28

4.6	Graph (5V)	30
4.7	Graph (9V)	30
5.1	Integrate Software and Hardware	39

LIST OF SYMBOLS

Ω	Ohm
$℃$	Celsius
mV	Millivolt
V	Volt

LIST OF ABBREVIATIONS

CAN	Controller Area Network
MSCAN	Motorola Scalable Controller Area Network
LM35	Linear Monolithic 35
ADC	Analog to Digital Converter
LCD	Liquid Crystal Display
PWM	Pulse Width Modulation
IDE	Integrated Development Environment
RFI	Radio Frequency Interference
GSM	Global System for Mobile
DC	Direct Current

CHAPTER 1

INTRODUCTION

1.1 BACKGROUND

Controller Area Network (CAN) was initially created by German automotive system supplier Robert Bosch in the mid-1980s for automotive applications as a method for enabling robust serial communication. Thereafter, CAN was standardized as ISO-11898 and ISO-11519, establishing itself as the standard protocol for in-vehicle networking in the auto industry. By networking the electronics in vehicles with CAN, however, they could be controlled from a central point. By this it could increase the functionality, add modularity, and makes diagnostic process more efficient. CAN bus can transfer the serial data one by one. CAN bus subsystems are accessible via the control unit on the standard termination, split termination, biased split termination. This project involves the implementation of Arduino board and sensor on CAN protocol. The sensor and hardware must follow or compatible with CAN protocol after integrating both software and hardware.

1.2 PROBLEM STATEMENT

Temperature is one of the very important parameters that need be monitored before unnecessary event occurs. Therefore there is a need to design temperature control system to avoid unnecessary event to occur.

1.3 PROJECT OBJECTIVES

The main objective is to design and built the temperature control system that will:

(i) Display the current temperature.
(ii) Control the temperature to be at the desired temperature.

1.4 SCOPE OF THE PROJECT

This project is focused on the design of the water temperature monitoring and control system. The scope of the project are:

(i) To design and fabricate the temperature monitoring and control system using Arduino microcontroller as main controller.
(ii) Software development on Arduino microcontroller and sensor using CAN Bus.

CHAPTER 2

LITERATURE REVIEW

2.1 INTRODUCTION

In this chapter the past working researches based on CAN Bus system are reviewed. The reviews are concentrated on the design techniques for implementation of CAN Bus for data monitoring and taking an appropriate decision based on the data in the control system. Based on the reviewed then was used to achieve the objective of this project works.

2.2 THE CAN-BUS

The Controller Area Network (CAN) is a serial, asynchronous, multi-master communication protocol for connecting electronic control modules in automotive and industrial applications[1]. CAN was designed for automotive applications needing high levels of data integrity and data rates of up to 1 Mbit/s. In this project Controller Area Network protocol is implemented using on chip Motorola Scalable Controller Area Network (MSCAN) of MC9S12DP256B 16-bit Microcontroller. This paper presents "Monitoring of Temperature using LM35 based on Controller Area Network architecture" .The system is constituted of two CAN nodes, each CAN node is formed by a transceiver MCP2551 and MCP2515 controller.

2.3 A ZIGBEE BASED TEMPERATURE MONITORING SYSTEM

In this paper, the design and development of a Zigbee based smart non-invasive temperature monitoring device is developed and reported in this paper. The system can be used to monitor temperature parameters[2]. The system consists of a sensor device, a temperature sensor which is integrated with a Zigbee module to transmit the data measured by the sensor. The data is received by a Zigbee module at the receiver end which monitors the values and if the value falls below a threshold the computer sends an alarm through Zigbee module. This sets off an alarm, allowing help to be provided to the user. The device is powered by battery for outdoor use. The device can be easily adapted to monitor air conditioning systems. The low cost of the device will help to lower the cost. A prototype of the device was fabricated and extensively tested that provide a good results.

2.4 ARDUINO BASED CAN PROTOCOL IMPLEMENTATION IN VECHICLE CONTROL SYSTEM

The main purposes of present automobiles are being developed by more of electrical parts for efficient operation[3]. Generally a vehicle was built with an analogue driver-vehicle interface for indicating various vehicle statuses like speed, fuel level, Engine temperature etc. This paper presents the development and implementation of a digital driving system for a semi-autonomous vehicle to improve the driver-vehicle interface and having Intelligent Braking System (IBS). The advantages of CAN bus based network over traditional point to point schemes will offer better flexibility and expandability for future technology insertions. It uses an ARDUINO based data acquisition system that uses analogue to digital converter (ADC) to bring all control data from analogue to digital format and visualize through the LCD. The communication module used in this project is embedded networking by CAN which has efficient data transfer. It also takes feedback of vehicle conditions like vehicle speed, engine temperature etc., and controlled by the main controller.

2.5 MONITORING AND CONTROLLING OF TEMPERATURE USING HARDWARE DESCRIPTION & OPERATION

CAN is a message arranged wide cost component. Utilizing would we be able to can make piece to square association or hub to hub association. We can send the information from one hub to other hub utilizing can transport. Utilizing would we be able to can send the information through casings. We are having information outline, augmented information outline, remote edge, blunder outline etc. If we need to send the information from one hub other we can utilize information outline, since can is an offbeat correspondence so information must be placed in begin bit and stop bit. Information outline contains SOF, ARBITRATION (IDENTIFIER+RTR) FIELD, DLC, DATA (MAX 8 BYTES), CRC, ACK, EOF[4].

The module is intended for temperature checking and controlling in view of the CAN convention. It manages the temperature changes that happen in any procedure in Industry. The LM35 arrangement are exactness coordinated circuit temperature sensors, whose yield voltage is straightly corresponding to the Celsius (Centigrade) temperature. The Temperature changes are measured by the inbuilt ADC and transmitted to the next hub utilizing the CAN Bus. At that point other hub will show the outcome on the LCD. What's more, in light of the temperature control move is makes put in the temperature hub. At first fundamental hub sends demand to every one of the hubs, in the wake of getting the edge these sub hubs analyzes the identifier, if the identifier coordinates then, the relating sub hub sends the information to the primary hub and after that principle hub will show information on LCD. In this manner the correspondence between hubs will be finished

CHAPTER 3

METHODOLOGY

3.1 INTRODUCTION

This chapter explain the methodology of the project. Research methodology gives a procedure and assist the researcher to perform the study detailed especially techniques, equipment, task and software interfacing to achieve objective within the scope of the project. The main objective methodology is to ensure the smoothness of the working procedure is completed within the estimated time frame. There are six phases in this project:

Phase I: Literature review,

(i) Internet information.
(ii) Books.
(iii) Papers and journal.
(iv) Discussion with lectures and supervisor.

Phase II: Design and develop the system,

(i) Learn how to use Arduino as selected program to develop and compatible with a CAN.
(ii) Draft the project and its flow chart.
(iii) Design and fabricate the system.

Phase III: Simulation of software and hardware.

(i) Decide which software to use based on the discussion with the supervisor.
(ii) Learn the programming language.
(iii) Develop the program for the fabricated system in phase II.

Phase IV: Integration of phase II and III and test whether the system achieve the objective of the project.

Phase V: Result Analysis

(i) Collect the data.
(ii) Analysis the data.

Phase VI: Thesis Writing.

3.2 FLOW CHART

Figure 3.1 shows the flow chart of the overall outline for methodology chapter.

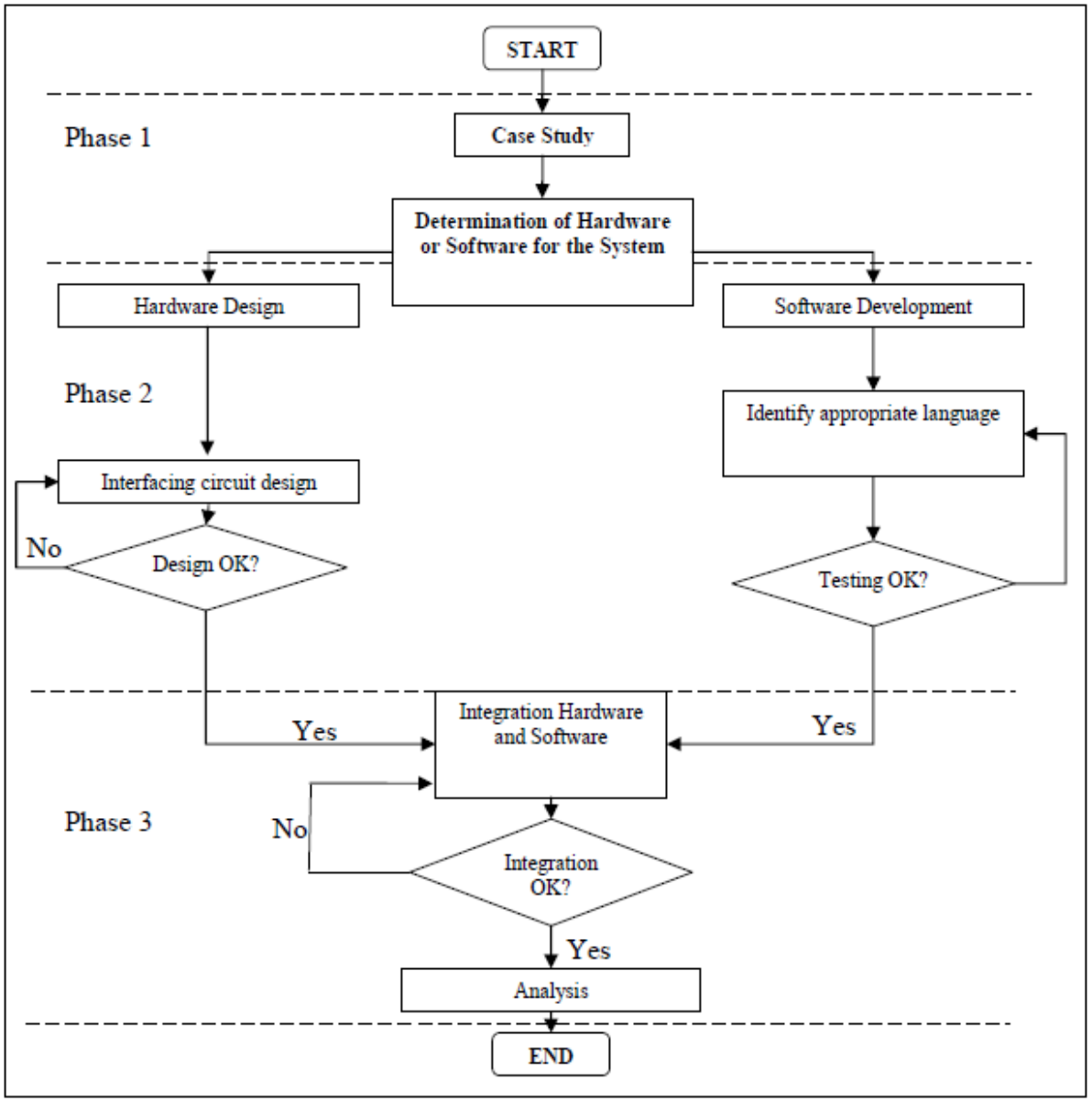


Figure 3.1 Flow chart of the study

3.3 BLOCK DIAGRAM

Figure 3.2 shows the block diagram of the implemented system

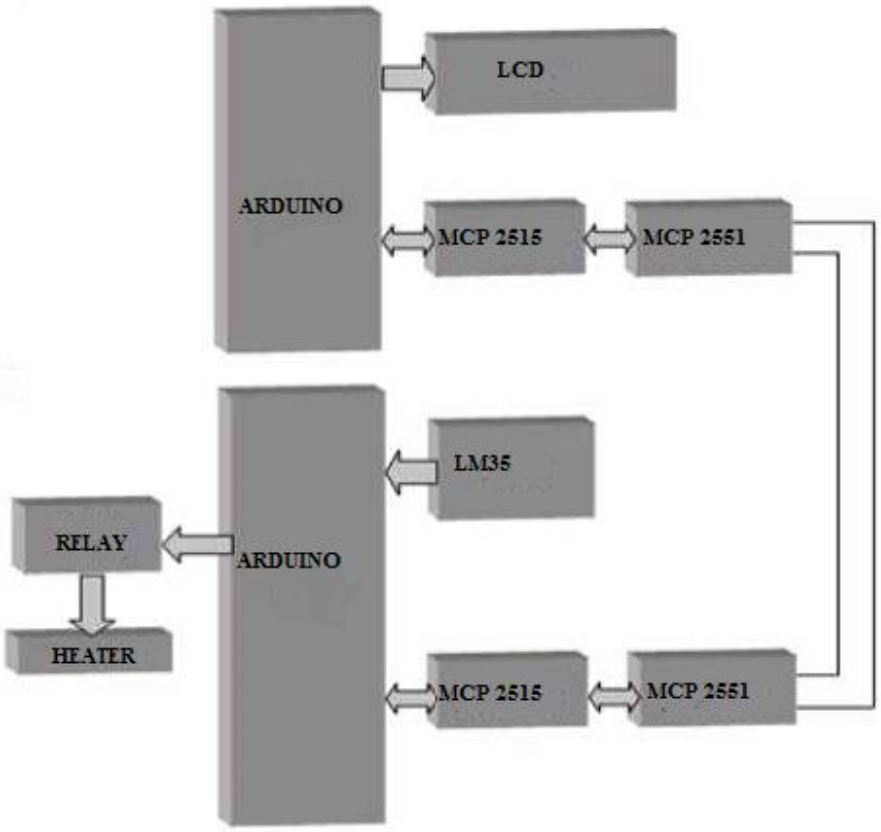


Figure 3.2 Block diagram of implemented system

3.4 SYSTEM DESIGN

In this project, two temperature sensor have been used. The reading was taken from both sensors to compare the reading accuracy between the sensors. The sensor used was LM35 and DS18B20. The Arduino Uno acting as a microcontroller for the system and for display, LCD 2x16 is used to display temperature reading. The heater and fan is used to control the output (temperature) of the system.

3.4.1 Microcontroller

The Arduino Uno is acted as a microcontroller. The Uno is a microcontroller board based on the ATmega328P. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analogue inputs, a 16 MHz quartz crystal, a USB connection, power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with an AC-to-DC adapter or battery to get started. This prevent the microcontroller from damage if the given voltage is higher than the normal voltage.

"Uno" means one in Italian and was chosen to mark the release of Arduino Software (IDE) 1.0. The Uno board and version 1.0 of Arduino Software (IDE) is the reference versions of Arduino, now evolved to newer releases. The Uno board is the first in a series of USB Arduino boards, and the reference model for the Arduino platform; for an extensive list of current, past or outdated boards seethe Arduino index of boards.

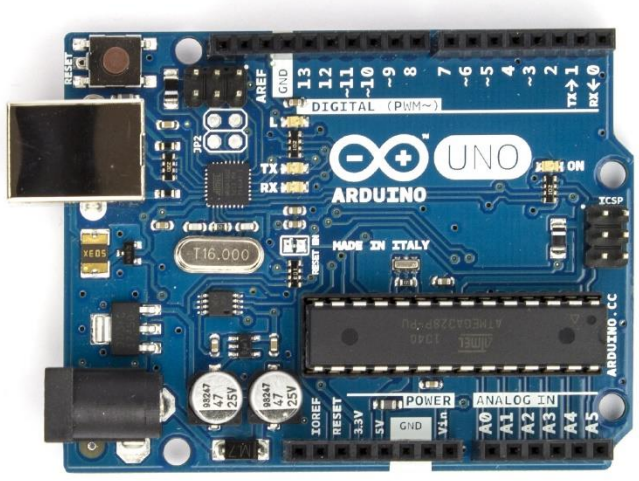


Figure 3.3.1 Arduino UNO board

3.4.2 CAN-BUS Shield

The CAN-BUS is a common industrial bus because of its long travel distance, medium communication speed and high reliability. It is commonly found on modern machine tools and as an automotive diagnostic bus. This CAN-BUS Shield adopts MCP2515 CAN Bus controller with Serial Peripheral Interface (SPI) interface and MCP2551 CAN transceiver to give Arduino/ Seeeduino CAN-BUS capability. The Serial Peripheral Interface (SPI) bus is a synchronous serial communication interface specification used for short distance communication, primarily in embedded systems. With an OBD-II converter cable added on and the OBD-II library imported and B&B's intelligent Streamers extract and translate the complex OBDII data from the vehicle bus and convert them into easy-to-use parameters, allowing Telematics Service Providers to focus on developing best of class applications., the diagnostic device or data logger, can be built on the board.

Figure 3.3.2 CAN-Bus Shield

3.4.3 Display

In this project Serial LCD (2x16) is used to display the current temperature of water. Now, with only 3 pins from the microcontroller, message can be displayed on the LCD screen. With comparing to parallel LCD which required at least 6 pins of I/O, this LCD offer more cost effective solution. The LCD display is two lines by 16 characters and provides basic text wrapping so that the text looks right on the display. In addition, the Serial LCD also provides full control over all of its advanced LCD features, allowing you to move the cursor anywhere on the display with a single instruction and turn the display on and off in any configuration.

Figure 3.3.3 LCD Display 2x16

3.4.4 DS18B20 Temperature Sensor

Temperature sensor selected for this project is required to be with compatible with CAN protocol. This low cost temperature sensing module provide easy to use sensor. The DS18B20 temperature sensor provide 9-bit to 12-bit Celsius measurement with the alarm function non-volatile user programmable upper and lower trigger points. This sensor communicates over a 1-Wire bus that require only one data line for communication with central microprocessor. Also it able to derive power directly from the data line by eliminating the need for an external power supply.

DS18B20 has a unique 64-bit serial code, allows multiple DS18B20 to function on the same 1-Wire bus. It is simple way one microprocessor to control many sensors distributed over large area.

Figure 3.3.4 DS18B20 Temperature sensor

3.4.5 LM35 Temperature Sensor

The LM35 are accuracy integrated circuit temperature device with a yield voltage strictly relative to the centigrade temperature. The sensor offer the following highlights:

- Directly calibrated in Celsius
- + 10-mV/°C scale factor
- 0.5°C Ensured accuracy at 25°C
- Range -55°C to 150°C
- Suitable remote application
- Low cost
- 4V to 30V operates
- Current drain is less than 60 micro ampere
- Low self-heating
- Non-linearity only ±¼°C Typical
- Output impedance is low, 0.1 ohm for 1 mili ampere load

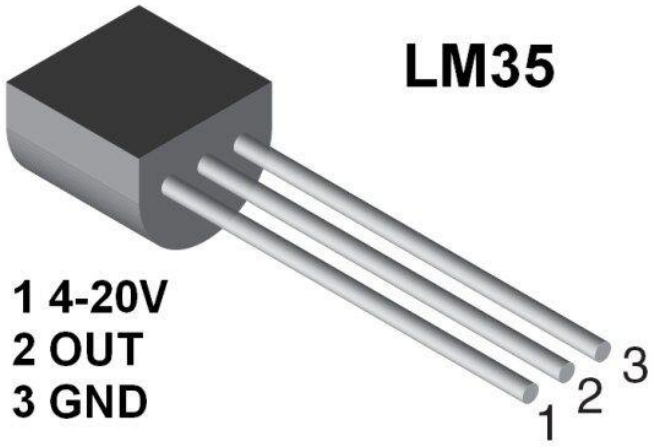


Figure 3.3.5 LM35 temperature sensor

3.4.6 CAN Controller and Transceiver

The controller utilized for this project is MCP2515. The controller is a second generation remain solitary CAN controller. The pin and capacity good with the MCP2510 furthermore incorporates overhauled highlights like speedier throughput, information byte channelling, and bolster for time-activated protocol. The elements of the chip are:

- 18-pin bundle
- Simple SPI interface to any MCU
- One-shot mode guarantees message transmission is endeavoured only once
- Data byte filtering
- Start-of-Frame (SOF) yield
- Low-power CMOS Technology

Figure 3.3.6 MCP2515 Controller

The MCP2551 is chosen to work with the controller. This controller is a fast CAN transceiver, fault tolerant gadget that serves as the interface between a CAN convention controller and the bus physical. The MCP2551 gives differential transmit and get capacity for the CAN convention controller and is completely perfect with the ISO-11898 standard, including 24V necessities. It will work at paces of up to 1 Mb/s. The components of MCP2551 are:

- Slope control input
- Supports 1 Mb/s operation
- Implements ISO-11898 standard physical layer prerequisites
- Suitable for 12V and 24V frameworks
- Externally-controlled slope for diminished RFI discharges
- Permanent predominant recognize
- Low current standby operation
- High clamour invulnerability because of differential bus execution

Figure 3.3.7 MCP2551 Transceiver

3.4.7 Relay

The relay is used in the design system. The relay is used to activate and deactivate the heater and fan. The fan and heater is activated or deactivated depending on current temperature whether is lower or higher than the threshold temperature.

Figures 3.3.8: Relay

3.4.8 Heater

The heater is used to heat up water until the desired or threshold temperature is achieved. Once the desired temperature is achieved, then the relay will deactivate the heater. The process will continue as long as the system is active mode (ON)..

Figure 3.3.9: heating element

3.4.9 Potentiometer 10kΩ

The 10kΩ potentiometer is used to vary the desired temperature. By varying the value of the resistance, we can test whether or not the system react accordingly.

Figure 3.3.10: Potentiometer

3.4.10 USB Fan

The USB fan acting as cooling system. The relay will activate the fan when the current temperature is higher than the desired temperature. The fan continues to be at active state until the desired temperature is achieved and relay deactivates the fan.

Figure 3.3.11: Fan

3.5 PROTOTYPE CIRCUIT SYSTEM DESIGN

Figure 3.4 show the final fabricated hardware that involve all components and devices needed in this project.

Figure 3.4: Prototype hardware design

3.6 SOFTWARE DEVELOPMENT AND CODING

Assembly code is used for speed, compactness or because some function are easier to assembler than in higher level language. Using a high level language will result faster program development. The microcontroller has an easy interface to assembly language. In C program assembler code may be embedded anywhere. The open source Arduino Software (IDE) is use to write code and enables to load program in Arduino Uno R3 board. The Arduino language use is C/C++. This is overall source code for the project is shown in Appendix A.

This is sample coding of part by part for the project:

(i) LCD

```
#include <LiquidCrystal_I2C.h>
#define I2C_ADDR          0x27        //Define I2C Address where the PCF8574A is
#define BACKLIGHT_PIN     3
#define En_pin            2
#define Rw_pin            1
#define Rs_pin            0
#define D4_pin            4
#define D5_pin            5
#define D6_pin            6
#define D7_pin            7
LiquidCrystal_I2C                                                    lcd(I2C_ADDR,
En_pin,Rw_pin,Rs_pin,D4_pin,D5_pin,D6_pin,D7_pin);
```

(ii) Sensor LM35

```
#include <Wire.h>
float temp;        //Define the temp float variable
int sensor = 1;     // sensor middle pin on analog pin 1
```

```
float desiredTemp;
float TempMin = 20;
float TempMax = 70;
void loop()
{

  temp = analogRead(sensor);        //assigning the analog output to temp
  temp = temp * 0.48828125;         //converting volts to degrees celsius ----- 0.48828125
= [(5V*1000)/1024]10
  lcd.setCursor(8,0);               //move the cursor to position 8 on row 1
  lcd.print(temp);                  //print the temperature in Celsius

// read the input on analog pin 0:
  int sensorValue = analogRead(A2);
  // Convert the analog reading (which goes from 0 - 1023) to a voltage (0 - 5V):
  float voltage = sensorValue * (100.0 / 1023.0);
  float desiredTemp = voltage;
  // print out the value you read:
  Serial.print("Desired Temperature = ");
  Serial.println(desiredTemp);

}
```

(iii) Sensor DS18B20

```
#include <OneWire.h>
#include <DallasTemperature.h>
#define ONE_WIRE_BUS 2
// Setup a oneWire instance to communicate with any OneWire devices (not just Max-
im/Dallas temperature ICs)
OneWire oneWire(ONE_WIRE_BUS);
```

```
// Pass our oneWire reference to Dallas Temperature.
DallasTemperature sensors(&oneWire);
Void loop ()
{
sensors.requestTemperatures(); // Send the command to get temperatures
 // Serial.print("Temperature for the device 1 (index 0) is: ");
  float x=sensors.getTempCByIndex(0);
//  Serial.println(x);
   lcd.setCursor(8,1);
   lcd.print(x);
}
```

3.7 HARDWARE SIMULATION

In this section explains how the hardware being simulated. The Figure 3.5 shows how to hook up the CAN Bus Shield with Arduino Uno. When the module is activated, the temperature sensors LM35 and DS18B20 sense the water temperature and resultant temperature reading is displayed on LCD. Before that, the sensors are completely submerged in the water. The first objective project is accomplished which is monitor the water temperature with LCD via CAN bus.

The potentiometer is used to set desired temperature. When the system detects the current temperature is lower than the desired temperature (say current temperature 29 degrees and set temperature 40 degrees), then the heating element (heater) is switched on automatically. When the heating element increases, the sensor that sensing the temperature of the heater is increase. When the set temperature is meet (40 degrees) the heating element automatically switched off. The current temperature is displayed on LCD. Fan is used for cooling system in this model. The desired temperature is set (say current temperature 40 degrees and desired temperature 29) the heater will off and the fan automatically switched on. The fan will off when current temperature is same as desired temperature. So the second objective to control water temperature based on desired value is accomplished.

Using CAN bus we can make block to block connection. We can send the data from one node to another node using can bus. Also can bus can send the data frames by frames. The can have data frame, error frame, remote frame and extended data frame. To send data from one node to other the data frame is use since CAN is an asynchronous communication so the data will be put in start bit and bit stop.

The aim of this project is designed temperature monitoring and controlling based on the CAN protocol. The LM35 are integrated-circuit temperature sensors whose voltage output is linearly proportional to Celsius temperature. The change temperature are measured and transmitted to other node using CAN Bus. The node will display on the LCD. And the temperature control action is take place in the temperature node. Main node sends to all the nodes the request and receiving frame sub nodes is compare the identifier, if identifier is match the sub node sends data to main data and will display data on LCD. The communication between is done. In this work MCP2551 transceivers is chosen. The flow chart of CAN receiver is explain in Figure 3.6.

The CAN Controller stores received bits one by one from the bus until entire message is available. The CAN Controller then fetched by the host (usually after the CAN has triggered an interrupt). The host processor stores transmit message into CAN Controller, which transmit the bits serially into the bus. In this project MCP2515 controller is chosen.

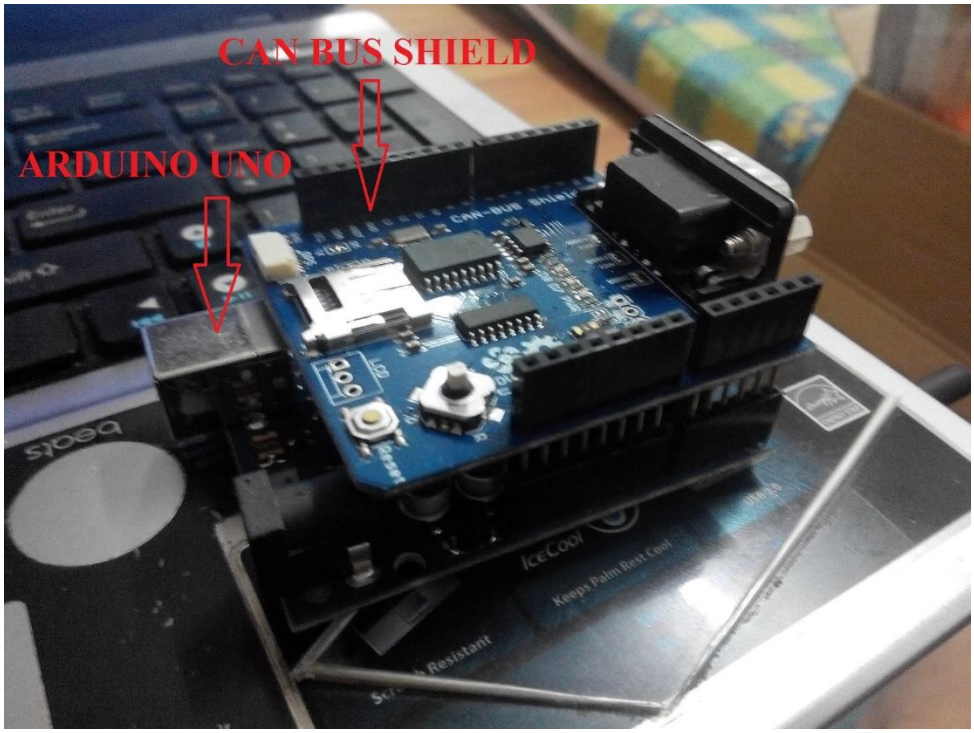


Figure 3.5: Hooked up between Arduino Uno and CAN Bus Shield

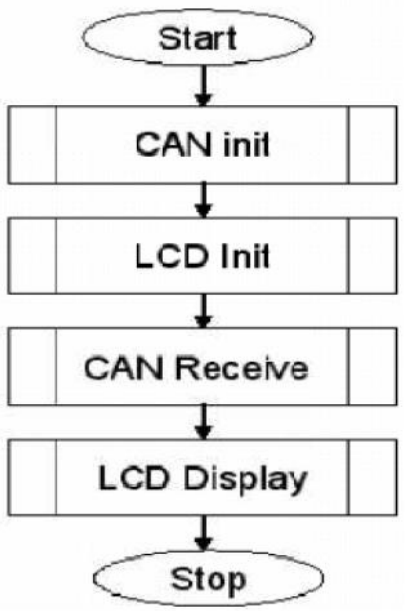


Figure 3.6: CAN Receiver Flow Chart

CHAPTER 4

RESULT AND ANALYSIS

4.1 INTRODUCTION

This chapter explains the results between two sensors reading that are obtained from two different cooling systems. The cooling systems are fan and dc motor water pump. The data were taken for 2 litres of water to decrease the temperature by minute using two different methods. Then, the data obtained from the two different sensors are analysed and compared. The finalised result of data are compared to the lab thermometer and which sensor is more accurate temperature reading is obtained. Figure 4.1 shows the example of data read by the sensors.

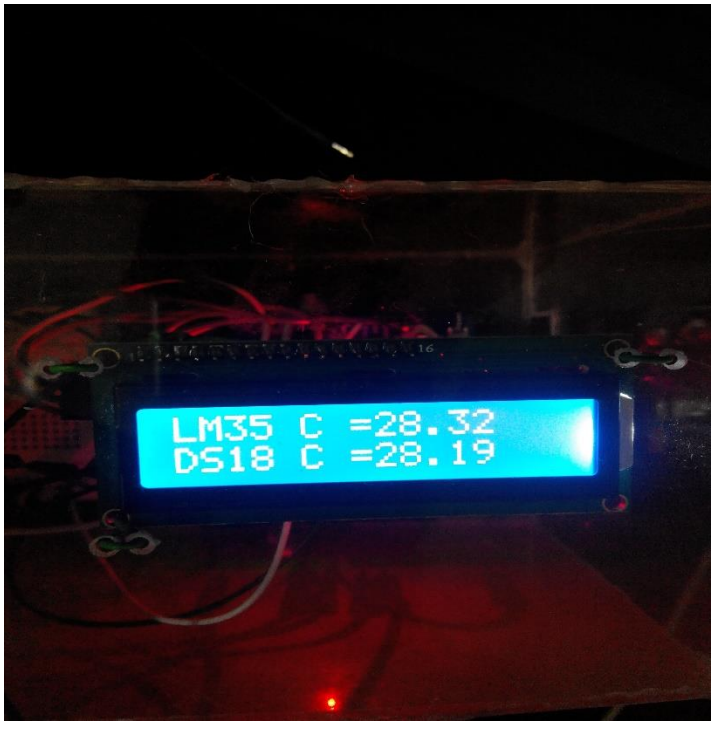


Figure 4.1 Example of Data Read by Sensor

4.2 FAN

The fan was used to cooling down water temperature. The two different distance between the fan and water were used that is 18 cm and 7cm.The sensor are fully submerged in the water. The figure 4.1 shows the setup of hardware. bef how to setup the hardware. Before collecting the data, the water temperature is set to 40°C and stopwatch is used to set time. The temperature reading is taken every 5 minute of the time interval for five times. Figure 4.3 and 4.4 show the graph from the data obtained from the sensors.

Figure 4.2: The fan hardware setup

Amount of time to cooling down temperture (minutes)	LM35 (°C)	DS18B20 (°C)
0	42.53	41.88
5	39.57	38.13
10	36.62	35.81
15	35.16	34.75
20	34.18	33.63
25	32.63	32.63

Table 4.1: The Result by Using Fan (distance 18cm)

Amount of time to cooling down temperture (minutes)	LM35 (°C)	DS18B20 (°C)
0	42.88	41.19
2	41.03	39.88
4	39.22	38.12
6	37.91	36.54
34.88	34.88	34.88

Table 4.2: The Result by using Fan (distance 7cm)

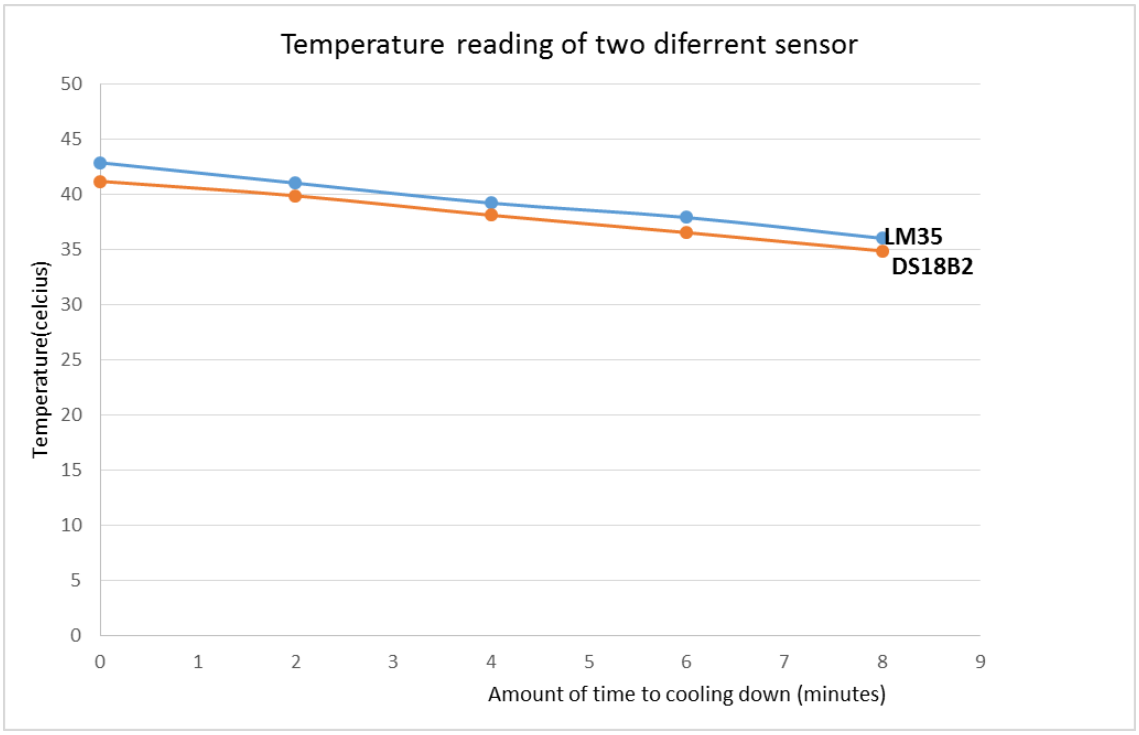


Figure 4.3: Graph (distance 18cm)

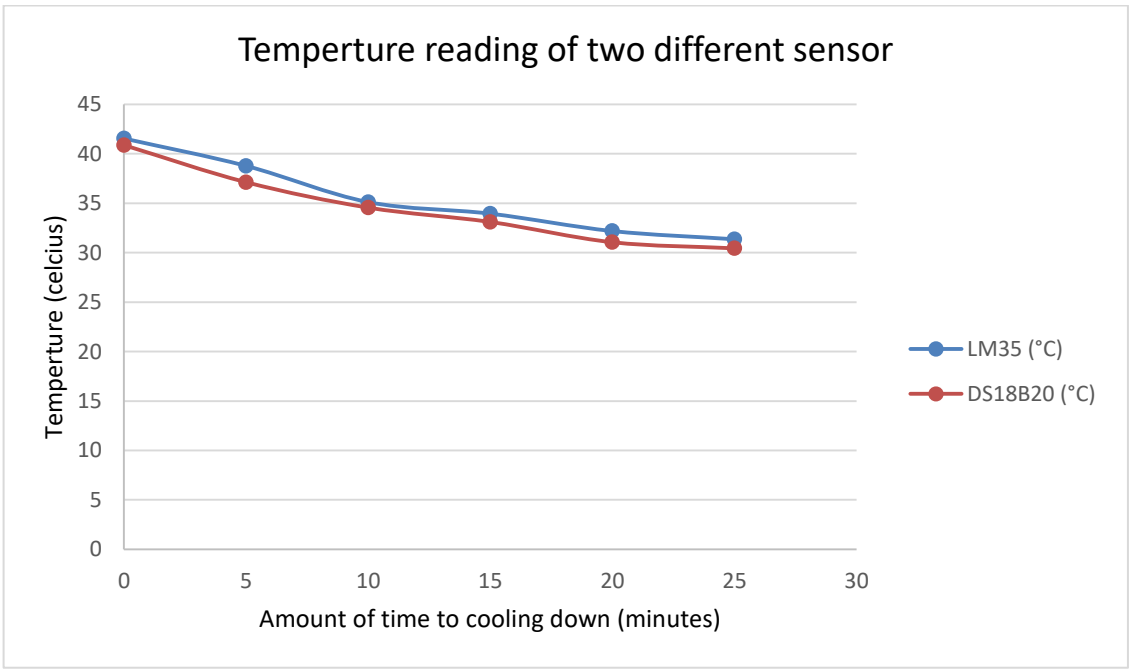


Figure 4.4: Graph (distance 7cm)

4.3 DC WATER PUMP

The DC water pump was used as cooling water temperature. The output voltage DC water pump was 5V and 9V. The figure 4.5 below showed how to setup the hardware including water pump. The temperature reading taken for 2 minute each passed. Set the temperature to 40 degree. The table 4.3 and 4.4 below show the result and figure 4.6, figure 4.7showed the graph result.

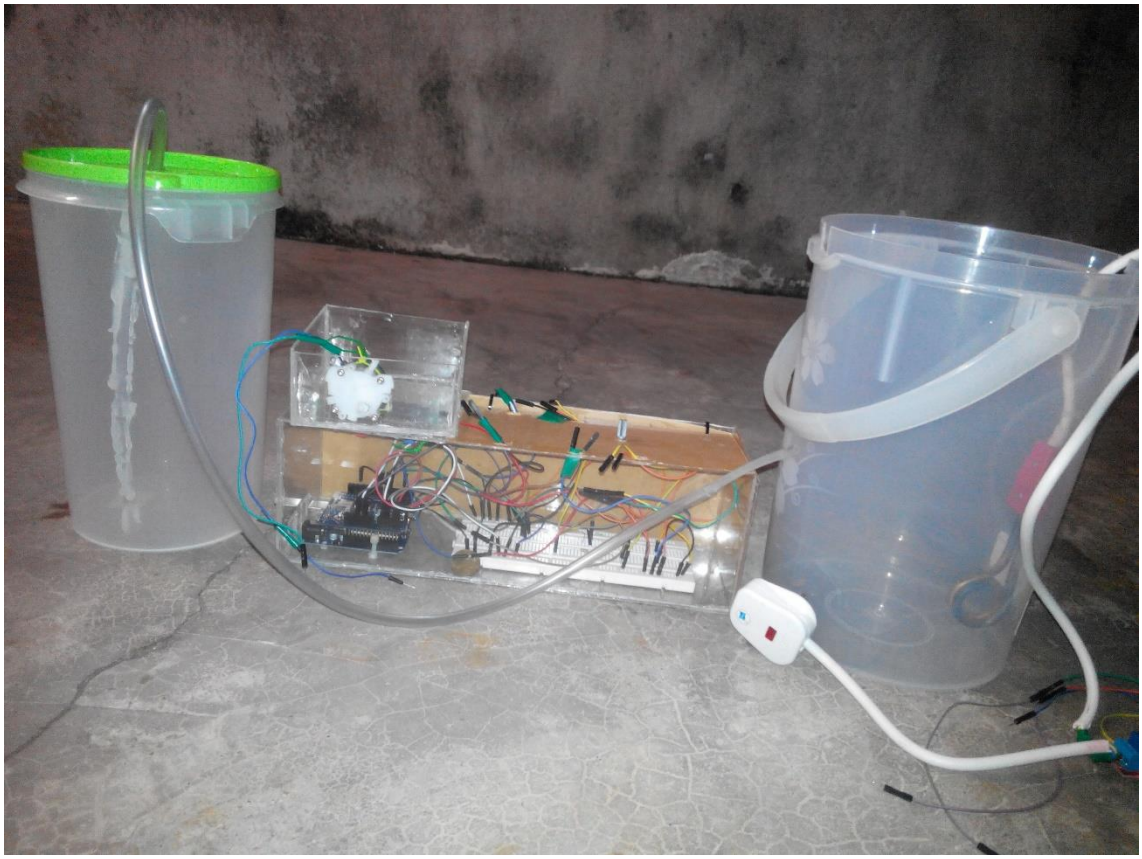


Figure 4.5: The DC water pump hardware setup

Amount of time to cooling down temperture (minutes)	LM35 (°C)	DS18B20 (°C)
0	41.55	40.88
5	38.78	37.13
10	35.12	34.56
15	33.95	33.11
20	32.19	31.06
25	31.36	30.44

Table 4.3: The Result by using water pump (5V)

Amount of time to cooling down temperture (minutes)	LM35 (°C)	DS18B20 (°C)
0	42.88	41.19
2	39.21	38.06
4	35.33	34.12
6	32.51	31.63
8	29.56	29.03

Table 4.4: The Result by using water pump (9V)

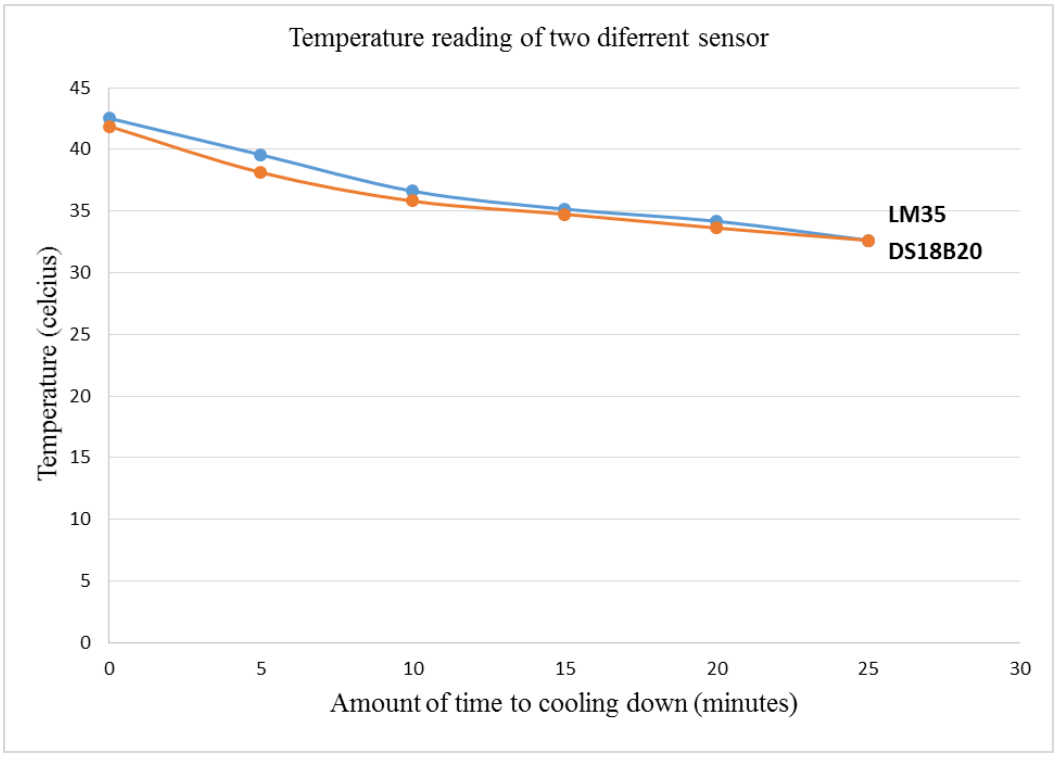


Figure 4.6: Graph (5V)

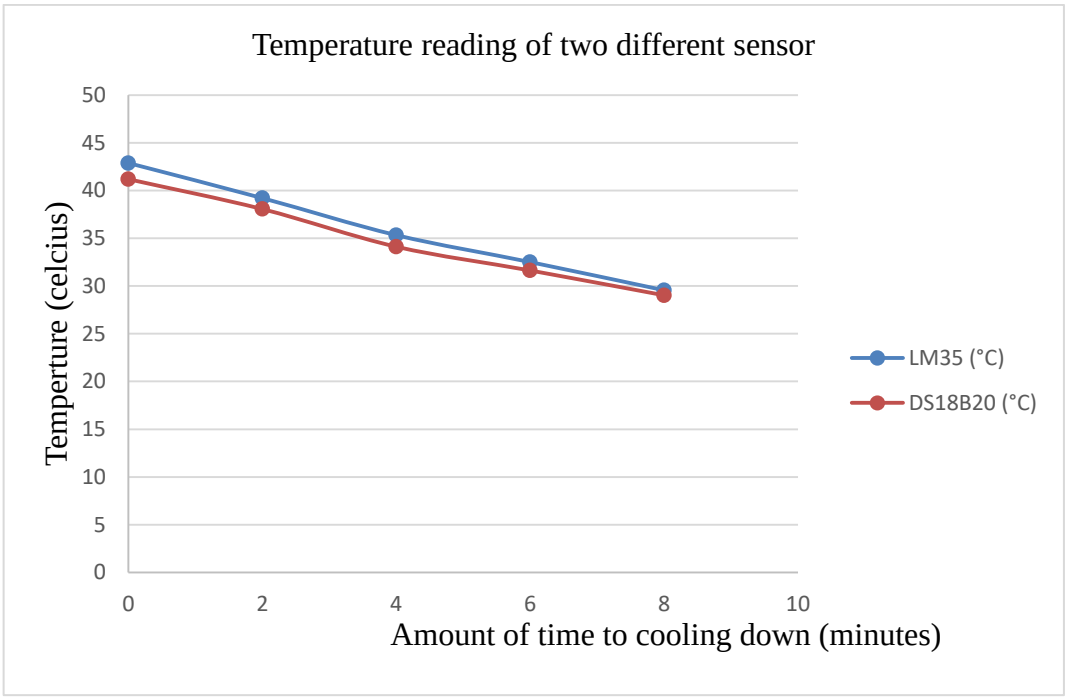


Figure 4.7: Graph (9V)

4.4 ANALYSIS

From the analysed data obtained from the temperature sensors, the temperature sensor DS18B20 is more accurate reading compare to the temperature sensor LM35. This is because LM35 is commonly used in to measure temperature in open area and not to measure the water temperature. The LM35 cannot be submerged into the water. In this project, LM35 was modified so that it can be submerged into the water. The DS18B20 specifically created to measured water temperature. Moreover, the DS18B20 is easier to handle or use than LM35.The DC water pump is more suitable to use as cooling system than using fan because it take less time to decrease water temperature than the fan.

CHAPTER 5

CONCLUSIONS AND RECOMMENDATION

5.1 CONCLUSIONS

This chapter presents the design of water temperature monitoring and control system based on CAN Bus protocol. The implementation of CAN Bus for water temperature monitoring and controlling system was successfully compeleted. The system is able to monitor the current temperature and also able to control the temperature to be at the desired temperature that is set by the user. The same idea can be applied to monitor under water robot monitoring system, vehicle power window, monitoring tire pressure and in automotive system engine management. This idea leads to decentralization of control system in car industry. This system can be extended to industrial car production. Mainly in decentralization the Arduino Microcontroller as control mechanism. There are mny potential future implementations of CAN Bus for industrial control system such as Arduino, and PLC (programmable logic controller).

5.2 RECOMMENDATION

There are still several improvements can be pursued in the future. This project is concerned on monitoring and control the temperature. This can be extended by connecting GSM module to the circuit. With GSM module technology we able monitor the temperature remotely place by sending the message. The example in industrial area, when machine crosses set temperature, the system will inform the control room manager/personnel by sending the message or make a call to the person in charge that will avoid damage the machine. Also the system will automatically disconnect the machine using GSM technology.

REFERENCES

[1] Mercendes-Benz, “Controller Area Network,” 2002.

[2] S. S. Patil, S. S. Sarade, and S. V Chavan, “Zigbee Based Sensor Networks for Temperature Monitoring and Controlling,” 2013.

[3] S. Shanmathi, P. C. Kamalanathan, S. M. Ramesh, S. Valarmathy, and B. Preethi, “Arduino Based Can Protocol Implementation In Vechicle Control System,” 2014.

[4] I. Engineering, A. Pradesh, I. Engineering, A. Pradesh, C. Engineering, and A. Pradesh, “Monitoring and Controlling of Temperature Using Hardware Description & Operation,” vol. 2, no. 6, pp. 61–68, 2014.

[5] Maxim Integrated 2008 DS18B20 Programmable Resolution 1-Wire Digital Thermometer *System* **92** 1–22.

[6] Controller T, Network A, Can E and Can T Advanced PIC18 Projects — CAN Bus Projects.

[7] Toolset C 2001 CAN-bus project Using the CANbus Toolset TM software and the and the SELECONTROL®MAS automation system.

[8] Anon 2015 Final Design Report : CAN Lighting System

[9] Mazran E, Redzuan A M, Badrul H A, Adie M K and Amat A B Security System Using CAN Bus . 3–7

APPENDIX A

```
#include <Wire.h>
#include <LCD.h>
#include <LiquidCrystal_I2C.h>
#include <Canbus.h>
#include <OneWire.h>
#include <DallasTemperature.h>
#include <SPI.h>
#include <mcp_can.h>
INT32U canId = 0x000;


#define I2C_ADDR          0x27        //Define I2C Address where the PCF8574A is
#define BACKLIGHT_PIN     3
#define En_pin            2
#define Rw_pin            1
#define Rs_pin            0
#define D4_pin            4
#define D5_pin            5
#define D6_pin            6
#define D7_pin            7


#define ONE_WIRE_BUS 2
#define fan1 7     // Set FAN pin on digital pin 7
#define relay 6    // Set Relay digital pin 6


//define variables for the LM35 temperature sensor
```

```
float temp;        //Define the temp float variable
int sensor = 1;     // sensor middle pin on analog pin 1
float desiredTemp;
float TempMin = 20;
float TempMax = 70;

//CAN BUS SHIELD

unsigned char len = 0;
unsigned char buff[8];
char str[20];
const int SPI_CS_PIN = 10; // select your can shield pin

MCP_CAN CAN (SPI_CS_PIN); //CS pin is set


// Setup a oneWire instance to communicate with any OneWire devices (not just Max-
im/Dallas temperature ICs)
OneWire oneWire(ONE_WIRE_BUS);

// Pass our oneWire reference to Dallas Temperature.
DallasTemperature sensors(&oneWire);



//Initialise the LCD
LiquidCrystal_I2C      lcd(I2C_ADDR,
En_pin,Rw_pin,Rs_pin,D4_pin,D5_pin,D6_pin,D7_pin);

void setup()
 {
   Serial.begin(115200);
```

```
  Serial.println("CAN-Bus ");
  pinMode(fan1, OUTPUT);
  pinMode(relay, OUTPUT);
  sensors.begin();

if(Canbus.init(CANSPEED_500))  /* Initialise MCP2515 CAN controller at the speci-
fied speed */
 {
  Serial.println("CAN Initialise ok");
 }
else
 {
  Serial.println("Can't initialise CAN");
  Serial.println("Initialise CAN BUS Shield again");
  delay(500);

 }


  lcd.begin (16,2);  //Define the LCD as 16 column by 2 rows

  //Switch on the backlight
  lcd.setBacklightPin(BACKLIGHT_PIN,POSITIVE);
  lcd.setBacklight(HIGH);

  //goto first column (column 0) and first line (Line 0)
  lcd.setCursor(0,0);

  //Print at cursor Location
  lcd.print("LM35 C = ");

  lcd.setCursor(0,1);
```

```
   lcd.print("DS18 C = ");


 }


void loop()
{

  temp = analogRead(sensor);        //assigning the analog output to temp
  temp = temp * 0.48828125;         //converting volts to degrees celsius ----- 0.48828125
= [(5V*1000)/1024]10
  lcd.setCursor(8,0);               //move the cursor to position 8 on row 1
  lcd.print(temp);                  //print the temperature in Celsius

// read the input on analog pin 0:
  int sensorValue = analogRead(A2);
  // Convert the analog reading (which goes from 0 - 1023) to a voltage (0 - 5V):
  float voltage = sensorValue * (100.0 / 1023.0);
  float desiredTemp = voltage;
  // print out the value you read:
  Serial.print("Desired Temperature = ");
  Serial.println(desiredTemp);

sensors.requestTemperatures(); // Send the command to get temperatures
 // Serial.print("Temperature for the device 1 (index 0) is: ");
  float x=sensors.getTempCByIndex(0);
//  Serial.println(x);
   lcd.setCursor(8,1);
   lcd.print(x);

   if (x < desiredTemp)
```

```
  {
    digitalWrite (fan1,0);
    digitalWrite(relay,1);
  }



  if (x > desiredTemp)
  {
    digitalWrite (fan1,1);
    digitalWrite(relay,0);
  }



  //wait 5 seconds
  delay(1000);
}
```

APPENDIX B

Figure 5.1: Integrate Software and Hardware

APPENDIX C

GANTT CHART FOR PSM 1

Activities	week1	week2	week 3	week 4	week 5	week 6	week 7	week 8	week 9	week 10	week 11	week 12	week 13	week14
PSM idea research and study														
Literature review														
Arduino study														
Report														
Methodology research														
Surveying and purchase of equipments														
Software design														
Hardware design														

GANTT CHART FOR PSM 2

Activities	week 1	week 2	week 3	week 4	week 5	week 6	week 7	week 8	week 9	week 10	week 11	week 12	week 13	week 14
Hardware design finalization														
Hardware Construction														
Hardware system testing and modifying														
Software design finalization														
Software system Development (Programming)														
Software system testing and debugging														
Prototype testing, modifying and finalization														
Thesis wiriting														