UNIVERSITI MALAYSIA PAHANG

BORANG PENGESAHAN STATUS TESIS♦

JUDUL: **AUTOMATION OF 6 WHEELS ROBOT (HOBO L3A1 ROBOT)**

SESI PENGAJIAN: **2011/2012**

Saya **NORFADILAH BT CHE YUSOFF (890627-03-6144)**
(HURUF BESAR)

mengaku membenarkan tesis (Sarjana Muda/~~Sarjana~~ /~~Doktor Falsafah~~)* ini disimpan di Perpustakaan dengan syarat-syarat kegunaan seperti berikut:

1. Tesis adalah hakmilik Universiti Malaysia Pahang (UMP).
2. Perpustakaan dibenarkan membuat salinan untuk tujuan pengajian sahaja.
3. Perpustakaan dibenarkan membuat salinan tesis ini sebagai bahan pertukaran antara institusi pengajian tinggi.
4. **Sila tandakan (√)

☐	**SULIT**	(Mengandungi maklumat yang berdarjah keselamatan atau kepentingan Malaysia seperti yang termaktub di dalam AKTA RAHSIA RASMI 1972)
☐	**TERHAD**	(Mengandungi maklumat TERHAD yang telah ditentukan oleh organisasi/badan di mana penyelidikan dijalankan)
√	**TIDAK TERHAD**	

Disahkan oleh:

_____ (TANDATANGAN PENULIS)

_____ (TANDATANGAN PENYELIA)

Alamat Tetap:

NO.19, KG TELUK BAYU,
BELAHAT,
17600, JELI
KELANTAN.

EN. MOHD FALFAZLI BIN MAT JUSOF
(Nama Penyelia)

Tarikh: **21 JUNE 2012**

Tarikh: : **21 JUNE 2012**

CATATAN:
- * Potong yang tidak berkenaan.
- ** Jika tesis ini SULIT atau TERHAD, sila lampirkan surat daripada pihak berkuasa/organisasi berkenaan dengan menyatakan sekali tempoh tesis ini perlu dikelaskan sebagai atau TERHAD.
- ♦ Tesis dimaksudkan sebagai tesis bagi Ijazah doktor Falsafah dan Sarjana secara Penyelidikan, atau disertasi bagi pengajian secara kerja kursus dan penyelidikan, atau Laporan Projek Sarjana Muda (PSM).

AUTOMATION OF 6 WHEELS ROBOT (HOBO L3A1 ROBOT)

NORFADILAH BT CHE YUSOFF

UNIVERSITI MALAYSIA PAHANG

AUTOMATION OF 6 WHEELS ROBOT (HOBO L3A1 ROBOT)

NORFADILAH BT CHE YUSOFF

A Thesis Submitted in Fulfillment for the
Requirement Award of the degree of
Bachelor of Electrical Engineering (Electronics)

Faculty of Electrical and Electronics Engineering
Universiti Malaysia Pahang

JUNE 2012

"I hereby acknowledge that the scope and quality of this thesis is qualified for the award of the Bachelor Degree of Electrical Engineering (Electronics)"

Signature: ……...

Name: EN MOHD FALFAZLI B. MAT JUSOF

Date: 21st JUNE 2012

Signature: ……

Author: NORFADILAH BT CHE YUSOFF

Date: 21st June 2012

DEDICATIONS

Specially dedicated to
My beloved parents,

CHE YUSOFF BIN CHE DERAHMAN
&
NORMADIAH BT YUNUS

Siblings, siblings' in-law,
UMP lecturers,
My dedicated supervisor,

EN MOHD FALFAZLI BIN MAT JUSOF

and all of my special and best friends and course mates.

ACKNOWLEDGEMENTS

First of all, I want to say syukur Alhamdulillah to ALLAH because He has granted me with faith and determinations along the way of this life He had borrowed to me, granted me with knowledge and with his most great love I am able to stand here with all the strengths.

Then, my sincere thanks and appreciation goes to my supervisor, En Mohd Falfazli B. Mat Jusof as I am greatly indebted to him for his advices, guidance and sacrifices throughout my project progress for the two semesters. Thank you, may ALLAH pay all your good deeds with His blessings here and hereafter.

My heartiest thanks to my beloved Mother, Father, and siblings because always praying for my successful since I live in this world until now. Their endless loves and supports throughout my four years study in University Malaysia Pahang has inspires me so much in becoming a great person. May our life full of blessings with ALLAH's love till *Jannah.*

Special thanks to FKEE staffs for helping me through my project progress. Suggestions and criticisms from my friends have always been helpful in finding solutions to my problems. Thank you all.

Finally, I would like to express my thanks to those who involves directly or indirectly in completion of my project.

Nor Fadilah Che Yusoff, UMP

ABSTRACT

The project is about to design a system which enables the 6 wheel robots named HOBO L3A1 ATM automated. A joystick acts an input to the system which embedded on board by interfacing with microcontroller and must be programmed according to expected result to make the robot in functions. The motor speed direction is controlled by developing H-bridge circuits which interface with microcontroller. The robot consists of 6 wheels of different motors that controlled the robot wheels. The overall robot chassis had finished which was designed with motor controls that driven by independent motor, thus the HOBO ATM robot conveniently switch its locomotion modes according to the operational. This project should be done in accordance with planning requirements. A microcontroller Arduino AtMega1280/2560 was needed in order to achieve the objectives of the project. An appropriate command need to transfer to the microcontroller. The embedded control system is not done yet, so a control system needs to be established to make the robot automated and functioned as well. The method for this project is designing a schematic diagram of circuit then implemented onto board. The microcontroller was programmed according to desired output. At the end of project, it was expected that the robot wheels can automate after the joystick (input) is controlled.

ABSTRAK

Projek ini adalah untuk mereka bentuk satu sistem yang membolehkan robot beroda enam yang bernama HOBO L3A1 berfungsi. Pedal diperlukan sebagai input untuk menggerakkan robot tersebut. Dengan menggunakan mikropengawal yang diprogramkan untuk menjadikan robot berfungsi. Robot yang terdiri daripada 6 roda dikawal oleh motor yang berlainan . Struktur robot keseluruhan telah lengkap yang direka bentuk kawalan motor yang setiap satunya mempunyai motor masing-masing , oleh itu, robot ini mudah bertukar mod gerak alih mengikut operasi. Projek ini perlu dilakukan mengikut perancangan yang telah ditetapkan. Sebuah mikropengawal Arduino AtMega1280 diperlukan untuk membuat ia berfungsi dengan baik. Suatu perintah yang sesuai perlu untuk diprogramkan kepada mikropengawal. Sistem kawalan dalaman tidak dilakukan lagi , maka satu sistem kawalan yang diperbaharui perlu ditubuhkan untuk membuat fungsi robot dan bergerak ke hadapan. Kaedah untuk projek ini adalah mereka bentuk gambar rajah skematik litar yang kemudian dilaksanakan ke board.Mikropengawal yang diprogramkan mengikut output yang dikehendaki. Pada akhir projek, ia adalah dijangka bahawa roda robot boleh bergerak selepas pedal (input) dikawal.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

V	Volt
DC	Direct Current
MHz	Mega Hertz
USB	Universal Serial Bus
cc	Clock Cycle
GND	Ground
Vcc	5V DC
LED	Light Emitting Diode
PCB	Printed Circuit Board
DAC	Digital to Analog Converter
BJT	Bipolar Junction Transistor
MOSFET	Metal Oxide Semiconductor Field-effect Transistor
PWM	Pulse Width Modulation
ATM	Angkatan Tentera Malaysia
USU	Utah State University
ODV	Omni Directional Vehicles
Hz	Hertz
CPU	Central Processing Unit

MSB	Most Significant Bit
LSB	Least Significant Bit
B	Base
E	Emitter
C	Collector
S	Source
G	Gate
D	Drain

LIST OF APPENDICES

CHAPTER	ITEM	PAGE

CHAPTER 1

INTRODUCTION

1.1Background

The project was implemented in collaboration UMP-ATM. This collaboration is to develop research in microcontroller and motor driver interfacing theory and practical. UMP had received a bomb diffusion robot named HOBO ATM robot from Angkatan Tentera Malaysia (ATM). We just receive the robot complete in its frame but cannot move or in other words not fully functioning. Where as, for the ATM they need to use the robot which can be used in military actions when it is functioned, so the robot was hand up to UMP so that we can turn the robot into functioning according to our expert in designing its input hardware and programming which enable the robot to move based on desired output. As for UMP, we cannot afford to build the robot frame according to high expensed, so we are using our knowledge and the ATM using their expenses. The HOBO ATM robot was build at 1997.

The Hobo is used mainly in applications of firefighting, the nuclear industry, airfield damage repair, terrorist or hostage situations, and the removal of debris and toxic chemicals, or in any situation or environment which is hazardous to human life. Hobo is built to exacting military standards and specifications and is currently in use in over 22 countries and with 34 agencies, including the United Nations. Combat proven, Hobo has

earned its reputation for being a tough, versatile and reliable vehicle from its years of operational experience in all extremes of climate and terrain. Hobo's exceptional performance is mainly due to its wheeled configuration. Six wheels, each independently driven, give the Hobo the ability to maneuver through the roughest terrain, through mud and sand, snow and water. Operational under radio control, it has a minimum range of 1 km line of sight. [5]

The automation of robot needs 2 major elements that need to be implemented in order to make the robot function. Firstly is microcontroller for programming and second is DC motor for the robot movement. This project is about interfacing DC motor and microcontroller. By using DC motor , the speed of motor can be controlled. Usually H-bridge is preferred way of interfacing a DC motor with a microcontroller. According to the H-bridge theory, the motion of the motor can be controlled. When there is a need for controlling the speed of a DC motor in an efficient manner, pulse width modulation (PWM) is generated. The duty cycle of PWM can be varied linearly through the current applied to the joystick as input hardware. For example, when 50 % of duty cycle is applied, so the speed of motor is 50 % from its actual speed that has been applied, this will affect the robot motion. Below is the hobo robot picture that consist of 6 wheel and each wheels has independently motor driver that stored in the robot body.

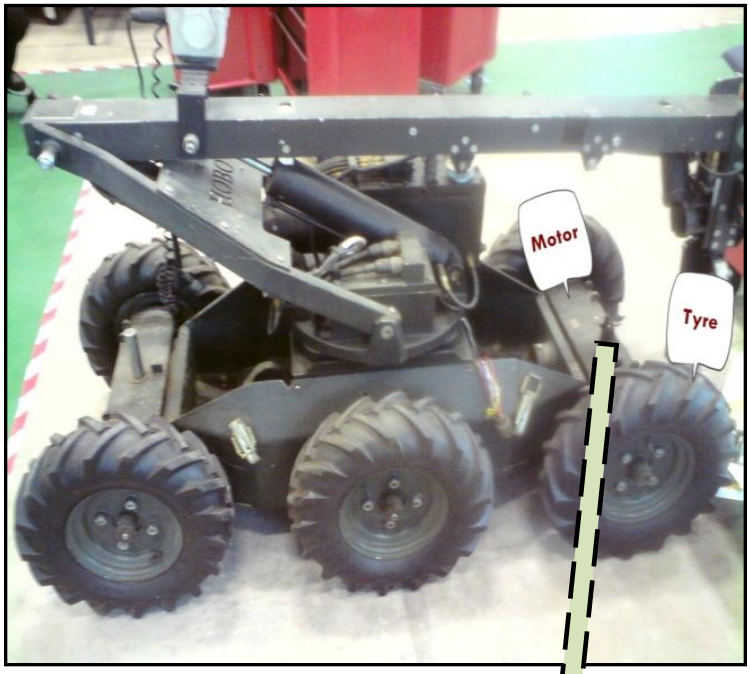


Figure 1.1: Hobo L3A1 Robot

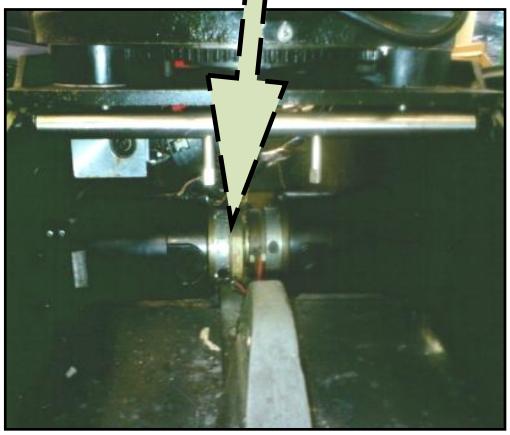


Figure 1.2: Motor driver and HOBO Robot

1.2 Objective of Project

The core objective of this project is to explore about microcontroller and DC motor interfacing in system design to establish hardware (joystick as the input) and make programming which enables the HOBO L3A1 ATM robot move based on pulse width modulation concept by using DC motor and microcontroller of Arduino AtMega1280. The system and the programming will be able to control the motor speed which enables the robot to move according the desired motion when joystick is applied.

1.3 Scope of Project

The scope of this project is as follow;

i. To choose the suitable microcontroller type for robot automation.

ii. To choose the appropriate components to fulfill DC motor specifications in designing H-bridge circuit.

iii. The DC motor will be as output that generates the mechanical energy based on pulse width modulation (PWM) varying from electrical energy in microcontroller.

iv. To describe how the microcontroller can be interface with DC motor in operating the robot automation

1.4 Problem Statement

The HOBO L3A1 robot is consist of six wheels, each wheels is driven by independent dc motor. The robot is not fully functioning and cannot be moved, therefore we need to implement the microcontroller to be interfaced with DC Motor of the robot in the hardware design as well as to develop programming code to the microcontroller and then design a complete system to all six DC motors which enabled the robot to move when input is applied to the system.

1.5 Thesis Outline

This thesis consists five chapters. In first chapter, it discusses about introduction and overview about this project includes background, objectives, and scope and problem statements of the projects.

Chapter two is explanations about literature review as study material, researches and references. The topics that I have studied are about the other method of speed control to compare and analysis their advantages and disadvantages. Furthermore, I can explore about the different types of transistors and other components that meet the requirements with my project specifications. From the literature review, knowledge can be gained thus implement in this project.

The methodology that I have done are discusses on chapter three. This is explanations about the method used to complete hardware and software.

Chapter four are discusses of the result and analysis of this project. This chapter explains the result obtained regarding the performance of the system.

For the last chapter are describes conclusion and future recommendation to make this project greatly. This thesis included with references and appendices. We can refer the further information about this project in references which is states the source and their authors. Datasheets of the components, photos and other information were placed on the appendices part.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

In this chapter include the history of robotics and automation, robots for military purposes, microcontroller interfacing with DC motor circuit theory, hardware and software programming.

2.2 Robotics and Automation History

Robots have been developed that can operate like human behaviors and become as a platform to accomplish human commands. Recently there has been increased interest in unmanned ground vehicles robots, especially for use in the military. This platform features consist of multiple “smart wheels” in which each wheel’s speed and direction can be independently controlled through dedicated processors. The result is a robot with the ability to completely control both the vehicle’s orientation and its motion.

2.2.1 Robots for Military Purpose

Srinivasavaradhan L et al. [2] in the journal title *7TH Sense A Multipurpose Robot for Military, April 2009* said, bomb diffusion can be made in both the automatic and in the user modes. In the automatic mode the robot detects the bomb and diffuses it by disabling the circuitry of the bomb. In case of failure in automatic bomb diffusion the control automatically goes to user mode. Once the user gets the control he can diffuse the bomb from remote location. [2] In the existing system there is only remote monitoring for robots are available. In the system we are going to control the robot from remote location in addition to remote monitoring (for example user mode). Our system also has an automatic mode in which it can take its own decision for combating. We are going to control the robot from remote location by using a computer. Our robot is also capable of detecting and diffusing the bombs more quickly. It can either be done through automatic mode or by user mode.

Kevin L. Moore and Nicholas S. Flann outlined, [3] a series of novel mobile robots based on a specific mobility capability that called as the smart wheel has been developed. For example is a 95-lb ODV vehicle with six smart wheels. Other USU ODV six-wheel vehicles include the ARC III, a 45-lb small-scale robot, and the T2, a 1480-lb robot. The USU smart wheel concept is shown in Fig. 1. Each smart wheel has a drive motor, power and a microcontroller, all in the wheel hub. This is combined with a separate steering motor and with actuation in the *z*-axis to create a three degree-of-freedom mechanism. Infinite rotation in the steering degree of freedom is achieved through an innovative slip ring that allows data and power to pass from the wheel to the chassis without a wired connection.

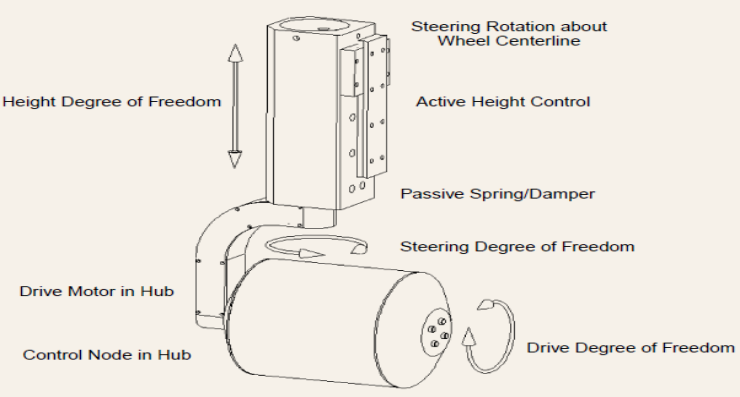


Figure 2.1: The USU smart wheel

The system has three distinct modes of control operation, one of its modes is in the manual control mode; an operator uses a joystick to maneuver the vehicle. A radio modem is used for communication with the joystick. In manual control, commands from the user (movements of the joystick encoded as voltages) are translated into body-referenced motion commands (x, y, θ) by the joystick interpreter [3].

According to Gerald MIES [4] *Military Robots and Industrial Robot* also have a common history. Many developments came out of the military laboratories in the beginning of last century. At this time the governments invest big parts of their budgets in military research and development. The first independent operating systems where used in military applications. Robots in military are still "special machines", build in small lots and very dedicated for their application. Today industrial robots are mass production machines. The mechanical types, drives, controller, sensors and applications are the key items in their development. Robots in Industry are categorized into four main mechanical robot- types: The linear-type- robot, the scara-type robot, the articulated-type-robot, and, the delta-type-robot. With the progress further development on the servo drives, most robots change their mechanical units into the articulated design. The IFR data's shows, that the market share of articulated - and delta robots growths fast. Scara robots and linear robot are also using servo motors, but because of their disadvantage in case of degrees of freedom, is the market share of this designs shrinking.

The revolution in the electronic Industry is another indicator for development in robotics. In the eighties, robot became slower if the demand for periphery communication increased, because the capacity of the CPU could not handle motion control and communication. Current generation of robot controllers is based on dual-core architectures. Motion control and data communications are kept separate and processing is distributed over a pair of CPUs. Sensors, for example, let robots see, feel and let robots work safety in there environment.

2.3 Microcontroller Interface with DC Motor

Tom Dickens [1], the kind of motor used is a standard motors with a feedback mechanism to sense its position which offer low cost, easy control, and good power. The control input to a motor tells it to be in a certain position, and logic built into the motor will position it. It has three wires comprised of black for ground, red are the motor's voltage, and white is the control line. The voltage is 5 volts. The control line requires a pulse width modulated (PWM) signal. There are two factors when dealing with a PWM signal: frequency and duty cycle. In this article, Tom Dickens [1] used the S148 type as the servo motor. For the S148, the frequency should be about 33 Hz, or 30 mS in width. True duty cycle is the ratio of high time over total time, for example is about 20%. However, with servos, the length of the high time is what indicates the servo's position. The actual frequency of the signal is not critical. Tests with the S148 indicate that a minimum high-pulse of about 0.1 mS will turn the servo full to the right, while a high-pulse of 2.3 mS will turn it full to the left. A high-pulse of about 1.2 mS will position the servo in the middle of its range. Full left-to-right movement takes about 3/4 of a second. A high-pulse in between these two times will cause the servo to position itself accordingly.

High-Pulse (mS)	0.1	0.46	0.65	0.83	1.2	1.57	1.75	1.93	2.3
Position(degrees °)	-90	-60	-45	-30	0	30	45	60	90

Table 2.1**:** Servo pulse width to servo positions.

Servo motors can be used for a variety of robotic uses as-is. Peter Dilworth's article in issue 1 of TRP talks about a walking biped robot that uses 12 servo motors. Whether he was using the S148 servos, another smaller model, and/or a model with faster response or higher power, regardless of the specific servo used, the concepts are the same.

According to the article, *The Secret Motor Driver* from *Solarbotics*, [7] there are many ways to strengthen (”buffer”) a signal so it’s strong enough to drive a large load like a motor. One of the ways is transistor H-bridges circuit. A standard servos use a “Pulse Width Modulated” (PWM) signals to tell a servo where to rotate to. PWM works by sending a rapid train of high/low signals to the servo’s regular driver brains, and depending on how different the high signal is from the low signal, the servo moves to the according position.

Referring to article title *“L293, L293D QUADRUPLE HALF-H DRIVERS”* [6], the L293 and L293D are quadruple high-current half-H drivers. The L293 is designed to provide bidirectional drive currents of up to 1 A at voltages from 4.5 V to 36 V. The L293D is designed to provide bidirectional drive currents of up to 600-mA at voltages from 4.5 V to 36 V. Both devices are designed to drive inductive loads such as relays,

solenoids, dc and bipolar stepping motors, as well as other high-current/high-voltage loads in positive-supply applications.

Whenever talking about robotics, there are always two options in front of the designer whether to use a DC motor or a stepper motor. When it comes to speed, weight, size, and cost, DC motors are always preferred over stepper motors. In addition, Krishna Nand Gupta et al. [8] proposed L293D as the driver IC in interfacing DC Motor and microcontroller. There are many things which we can do with our DC motor when interfaced with a microcontroller. For example we can control the speed of motor; we can control the direction of rotation. Usually H-bridge is preferred way of interfacing a DC motor. These days many IC manufacturers have H-bridge motor driver available in the market like L293D is most used H-Bridge driver IC. H-bridge can also be made with the help of transistors and MOSFETs etc, rather of being cheap, they only increase the size of the design board, which is sometimes not required so using a small 16 pin IC is preferred for this purpose.

The *DC Motor Direction Control Using L293D* [9] article, state and provide tutorials about how to interface or control a simple DC motor using microcontrollers. Controlling a DC motor is nothing but the important part is controlling the direction and speed of a motor. It is very necessary to go through motor controlling concept when we are dealing with robot automation.

Before going through the article, we will see how actually the DC motor runs. The direction control of a DC motor is very simple; it just needs to reverse the polarity, means every DC motor has two terminals out. When DC voltage is applied with proper current to a motor, it rotates in a particular direction but when we reverse the connection of voltage between two terminals, motor rotates in another direction as shown below.

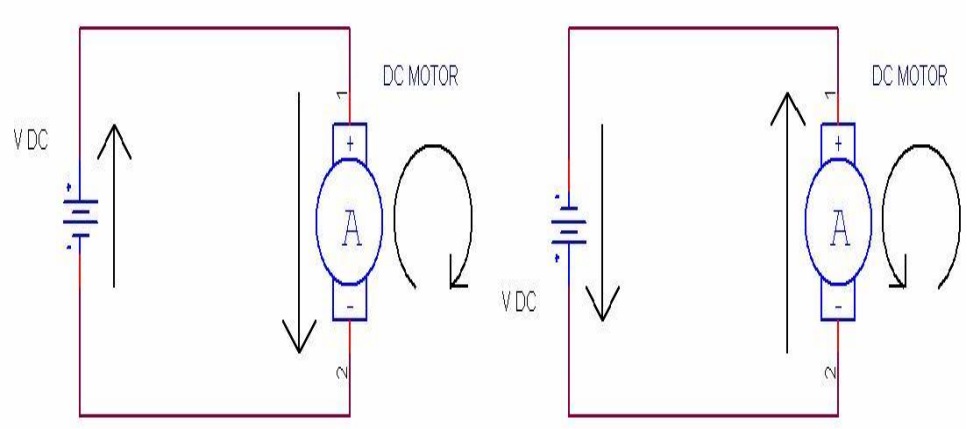


Figure 2.2: DC Motor rotations

Now we will consider how to control motor using Microcontroller provided. The microcontroller provides us with only digital logic (1 or a 0). We cannot provide polarity from microcontroller and we cannot connect motors to controller as mostly motors runs on voltage higher than +5V, and motors demands high current. Therefore, the solution to above limitations is use of an H-Bridge as shown in the figure below.

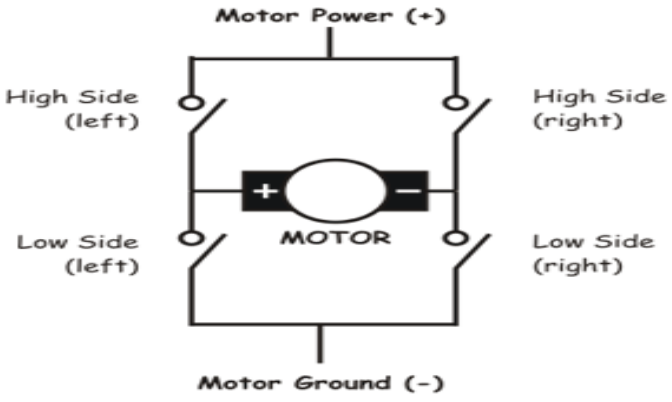


Figure 2.3: H-Bridge circuit

The circuit as shown in the figure below, it allows motor rotation in both directions. From four terminals of H-bridge we can control a DC motor by using L293d Dual Half H Bridge. Furthermore, we can make our own H-Bridge using transistors but as proposed in the article, it will be better if we use a ready made IC named as L293D, its a dual half H bridge IC.

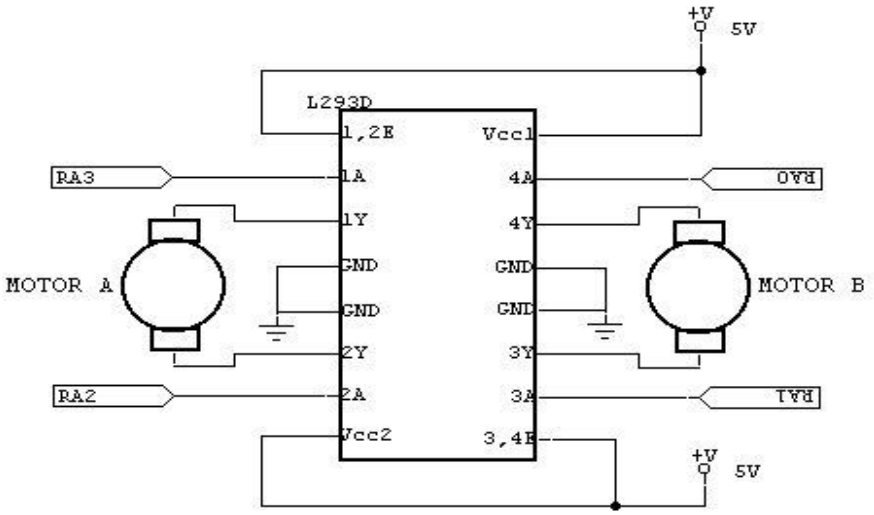


Figure 2.4: Circuit of L293D H-Bridge

IN1	IN2	Motor1
0	1	Rotates in one direction
1	0	Rotates in other direction

Table 2.2: Decision Table

Similarly, it is true for another motor connected to Out3 and Out4 of L293d and can be controlled through IN3 and IN4. This is all about controlling direction of DC

motor using L293d. In addition, to control the speed of DC motor one can use a Pulse Width Modulated signal on Enable1 and Enable2 pins of L293D; this will result in controlled power input on motor, so speed is controlled.

2.3.2 H -bridge circuit

Below is shown H-Bridge Schematic Specifics;

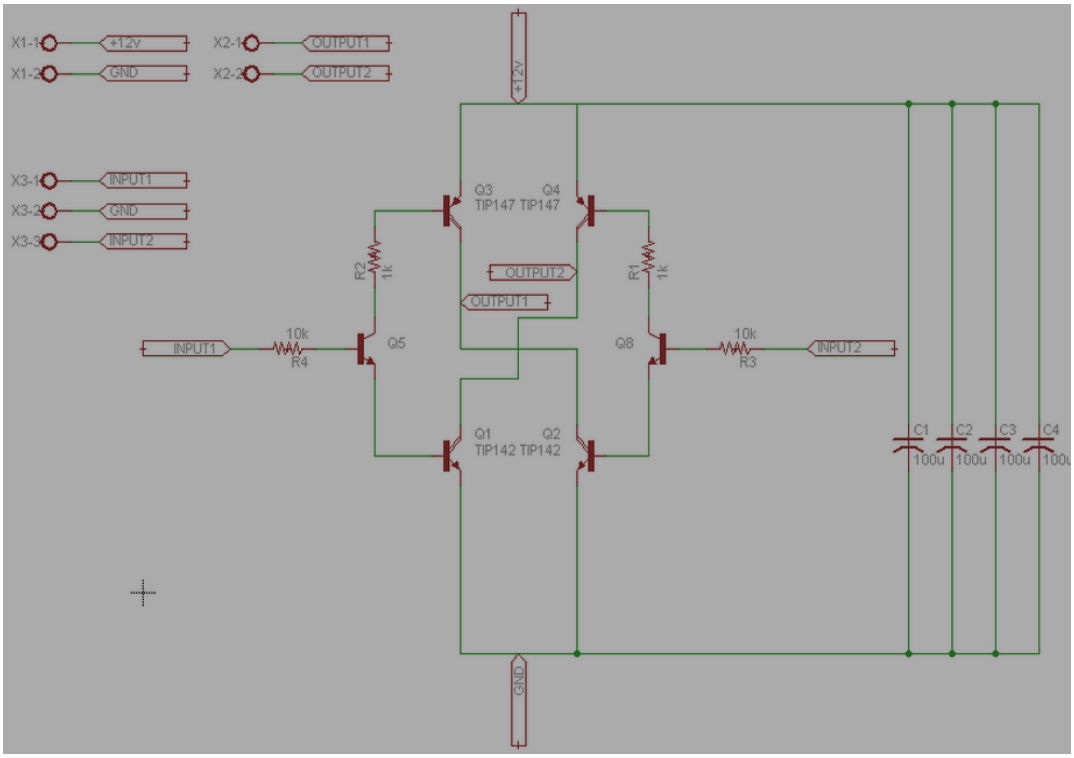


Figure 2.5: Schematic of H bridge circuit

2.3.2.1 Power Circuit

The power circuit is the +12v coming in from the terminal block X1-1/X1-2. It connects to a few 100uF capacitors and you might need more or less to help the motor run smoothly. The power also connects to the very top of the H-bridge (power to collector) and the very bottom (emitter to ground).

2.3.2.2 Motor Control Outputs

Outputs 1 and 2 are found in the middle of the H-bridge circuit, these connections feed into one of the dual terminal blocks, which connects to the DC motor. It is at these points of Output 1/2 where power will flow to drive the motor.

2.3.2.3 Digital Control Input

The triple terminal block X3-1/X3-2/X3-3 offer a way to connect up some digital circuitry and a ground to control the h-bridge. Input 1 controls one side of the H-bridge and Input 2 controls the other side.

2.3.2.1 BJT vs. MOSFET

When it comes to motor control and H-bridges, there are two types of power transistors that take the main stage. The Power BJT and Power MOSFET. The main difference between the two, at least as far as we are concerned is the power loss.

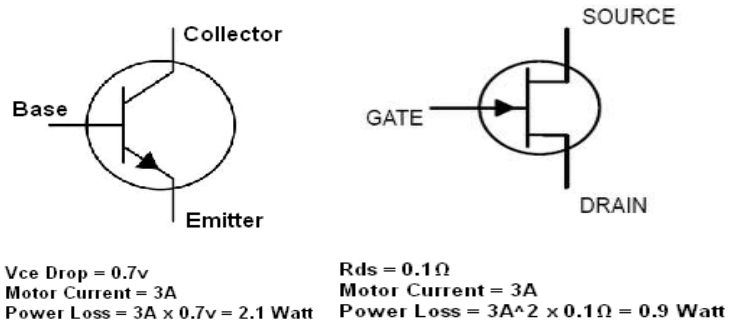


Figure 2.6: Differences between BJT and MOSFET

2.4 Speed Control DC Motor Using Microcontroller [10]

Mohd Fikri Awang [10], the direct current (DC) motor is a device that used in many industries in order to convert electrical energy into mechanical energy. This is all result from the availability of speed controllers is wide range, easily and many ways. In most applications, speed control is very important. DC motor plays a significant role in modern industry. The purpose of a motor speed controller is to take a signal representing the demanded speed, and to drive a motor at that speed. There are numerous applications where control of speed is required, as in rolling mills, cranes, hoists, elevators, machine tools, transit system and locomotive drives. These applications may demand high-speed control accuracy and good dynamic responses. Thus, these applications can be implemented in designing a system for the robot automation in this project.

The electric drive systems used in industrial applications are increasingly required to meet higher performance and reliability requirements. The DC motor is an attractive place of equipment in many industrial applications requiring variable speed and load characteristics due to its ease of controllability. Microcontrollers provide a

suitable means of meeting these needs. Certainly, part of the recent activity on microcontrollers can be ascribed to their newness and challenge. In this project, I used arduino Atmega 1280 microcontroller type as controller for the speed controller of the system. Another system that uses a microprocessor is reported in the work is reported in journal a brief description the system is as follow: The microprocessor computes the actual speed of the motor by sensing the terminal voltage and the current, it then compares the actual speed of the motor with the reference speed and generates a suitable control signal which is fed into triggering unit .The motor speed is controlled through a variable potentiometer for a manually controlled system; the operator mentally compares the actual speed to a desired speed and sets the potentiometer accordingly. [10]

2.5 Speed control DC Motor using Arduino Atmega 1280/2560 [12]

Pulse-width modulation (PWM) can be implemented on the Arduino in several ways. This article explains simple PWM techniques, as well as how to use the PWM registers directly for more control over the duty cycle and frequency. This article focuses on the Arduino Diecimila and Duemilanove models, which use the ATmega168 or ATmega328. Briefly, a PWM signal is a digital square wave, where the frequency is constant, but that fraction of the time the signal is on (the duty cycle) can be varied between 0 and 100%.

Here are the PWM examples with variety of duty cycle;

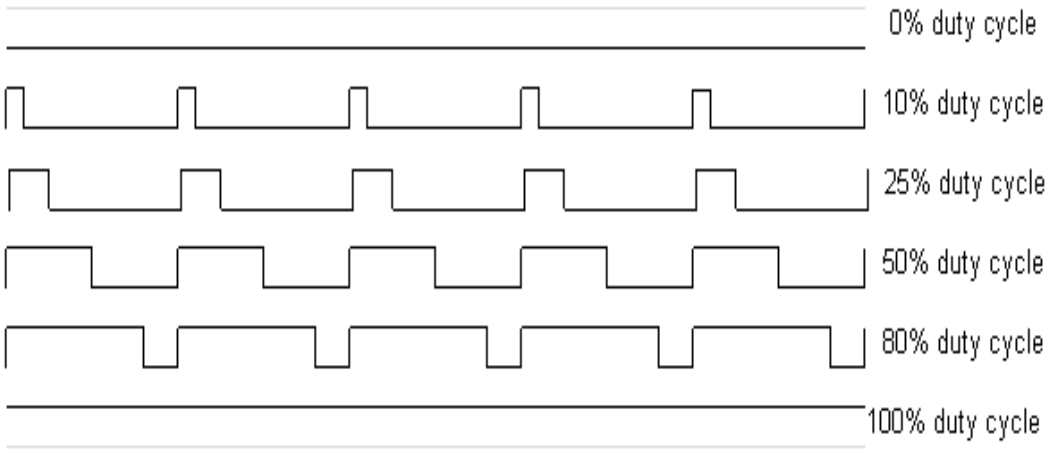


Figure 2.7: PWM with variety of duty cycle

There are several uses of PWM:

- Dimming an LED
- Providing an analog output; if the digital output is filtered, it will provide an analog voltage between 0% and 100 % .
- Generating audio signals.
- Providing variable speed control for motors.
- Generating a modulated signal, for example to drive an infrared LED for a remote control.

2.5.1 Pulse Width Modulation with analogWrite [12]

The Arduino's programming language makes PWM easy to use; simply call analogWrite (pin, dutyCycle), where dutyCycle is a value from 0 to 255, and pin is one of the PWM pins (2, 3, 4, 5, 6, 7, 8, 9, 10, 11, and 12) for Arduino 2560/1280. The analogWrite function provides a simple interface to the hardware PWM, but does not provide any control over frequency. Note that despite the function name, the output is a digital signal, often referred to as a square wave.

In the simplest PWM mode, the timer repeatedly counts from 0 to 255. The output turns on when the timer is at 0, and turns off when the timer matches the output compare register. The higher the value in the output compares register, the higher the duty cycle. This mode is known as Fast PWM Mode.

2.5.1.1 Bit-banging Pulse Width Modulation[12]

Below are the example of manually implement PWM on any pin by repeatedly turning the pin on and off for the desired times.

```
void setup()
{
  pinMode(13, OUTPUT);
}
void loop()
{
  digitalWrite(13, HIGH);
  delayMicroseconds(100); // Approximately 10% duty cycle @ 1KHz
  digitalWrite(13, LOW);
  delayMicroseconds(1000 - 100);
}
```

This technique has the advantage that it can use any digital output pin. In addition, we can control full of the duty cycle and frequency. One major disadvantage is that any interrupts will affect the timing, which can cause considerable jitter unless we disable interrupts. Second disadvantage is we cannot leave the output running while the processor does something else. Finally, it is difficult to determine the appropriate constants for a particular duty cycle and frequency unless we either carefully count cycles, or tweak the values while watching an oscilloscope.

2.6 Arduino Atmega 1280/2560

2.6.1 General System Information

In these topics, we are going to cover the hardware and software setup required to connect an Arduino device with a variety of electronic parts, chips and devices. A related topic covered under this section is the shield, boards that plug directly into an Arduino's pin layout. Information on the creation and use of specific shields belongs in that section. Information on shields in general and their creation belongs here.

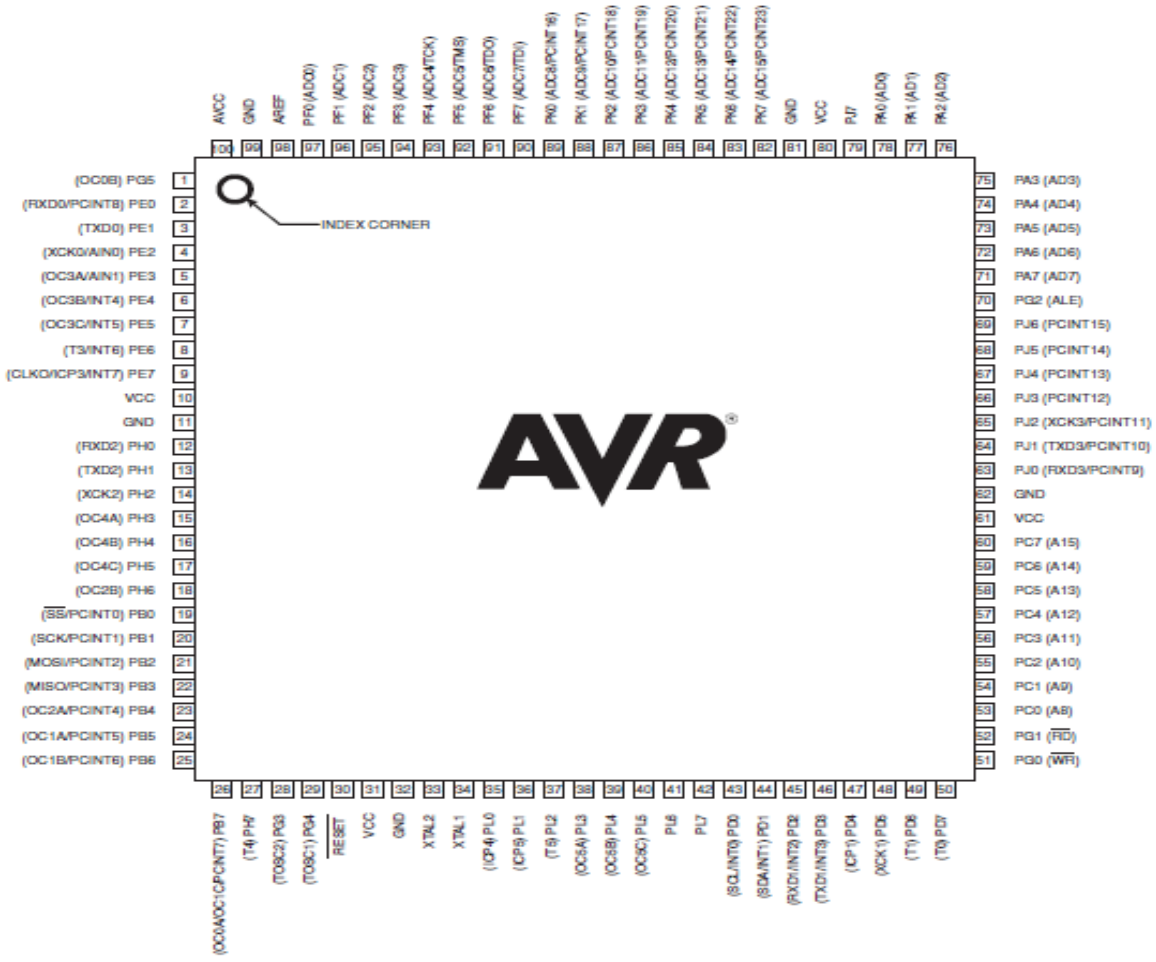


Figure 2.8: Pin configuration of Arduino Atmega 2560

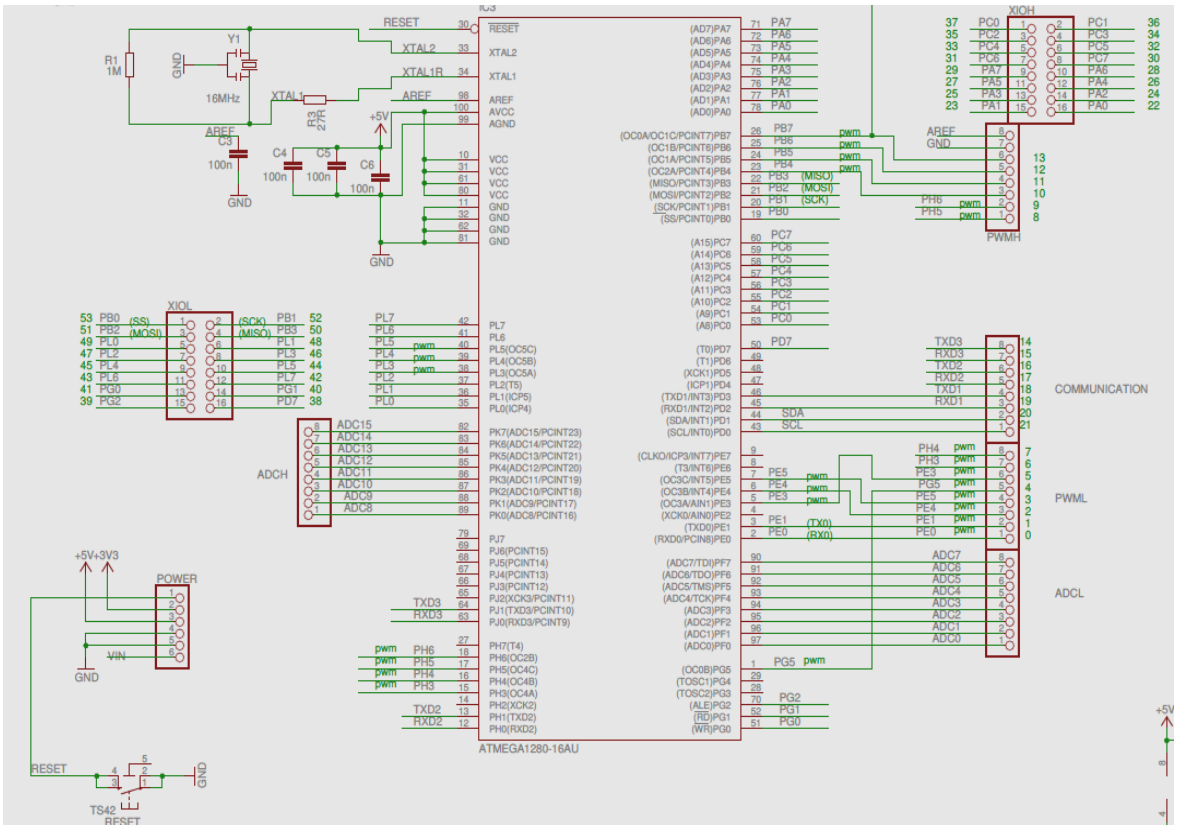


Figure 2.9: Pin Mapping for Arduino 2560/1280

Each device in the ATmega640/1280/1281/2560/2561 family differs only in memory size and number of pins. Table 2-1 summarizes the different configurations for the six devices. Here are the comparison between ATmega1281/2561 and ATmega640/1280/2560 [11]

Device	Flash	EEPROM	RAM	General Purpose I/O pins	16 bits resolution PWM channels	Serial USARTs	ADC Channels
ATmega640	64KB	4KB	8KB	86	12	4	16
ATmega1280	128KB	4KB	8KB	86	12	4	16
ATmega1281	128KB	4KB	8KB	54	6	2	8
ATmega2560	256KB	4KB	8KB	86	12	4	16
ATmega2561	256KB	4KB	8KB	54	6	2	8

Table 2.3: Comparison between ATmega1281/2561 and ATmega640/1280/2560

2.6.2 System description of Atmega 2560/1280

Below are summary of system description about ATmega2560 rev. A

• Non-Read-While-Write area of flash not functional

• Part does not work under 2.4 volts

• Incorrect ADC reading in differential mode

• Internal ADC reference has too low value

• IN/OUT instructions may be executed twice when Stack is in external RAM

• EEPROM read from application code does not work in Lock Bit Mode 3

2.6.2 1 Non-Read-While-Write area of flash not functional

The Non-Read-While-Write area of the flash is not working as expected. The problem is related to the speed of the part when reading the flash of this area. For problem fix/workaround, only use the first 248K of the flash. If boot functionality is needed, run the code in the Non-Read-While-Write area at maximum 1/4th of the maximum frequency of the device at any given voltage. This is done by writing the CLKPR register before entering the boot section of the code.

2.6.2 2 Part does not work under 2.4 volts

The part does not execute code correctly below 2.4 volts. For problem fix/ workaround. Do not use the part at voltages below 2.4 volts.

2.6.2.3 Incorrect ADC reading in differential mode

The ADC has high noise in differential mode. It can give up to 7 LSB errors. For problem fix/workaround, use only the 7 MSB of the result when using the ADC in differential mode.

2.6.2 4 Internal ADC reference has too low value

The internal ADC reference has a value lower than specified. For problem fix/ workaround, use AVCC or external reference. The actual value of the reference can be measured by applying a known voltage to the ADC when using the internal reference. The result when doing later conversions can then be calibrated.

2.6.2 5 IN/OUT instructions

The IN/OUT instructions may be executed twice when Stack is in external RAM If either an IN or an OUT instruction is executed directly before an interrupt occurs and the stack pointer is located in external ram, the instruction will be executed twice. In some cases this will cause a problem, for example:

- If reading SREG it will appear that the I-flag is cleared.

- If writing to the PIN registers, the port will toggle twice.
- If reading registers with interrupt flags, the flags will appear to be cleared.

The Arduino has limits on how much current can be sourced or sunk by its I/O pins. When interfacing with hardware you need to be careful not to exceed these limits. In general, do not exceed 20 mA per pin. In particular, do NOT directly connect LEDs to Arduino outputs! Always use a series resistor (220 ohms is a good value).

Speed (MHz)[2]	Power Supply	Ordering Code	Package[1][2][3]	Operation Range
8	1.8V - 5.5V	ATmega2560V-8AU ATmega2560V-8AUR[4] ATmega2560V-8CU ATmega2560V-8CUR[4]	100A 100A 100C1 100C1	Industrial (-40°C to 85°C)
16	4.5V - 5.5V	ATmega2560-16AU ATmega2560-16AUR[4] ATmega2560-16CU ATmega2560-16CUR[4]	100A 100A 100C1 100C1	

Table 2.4: System information of various type of Atmega

2.6.3 Pin Descriptions [12]

VCC = Digital supply voltage.
GND= Ground.

2.6.3.1 Port A (PA7~PA0)

Port A is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port A output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port A pins that are externally pulled low will source current if the pull-up resistors are activated. The Port A pins are tri-stated when a

reset condition becomes active, even if the clock is not running. Port A also serves the functions of various special features of the ATmega640/1280/1281/2560/2561.

2.6.3 2 Port B (PB7~PB0)

Port B is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port B output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port B pins that are externally pulled low will source current if the pull-up resistors are activated. The Port B pins are tri-stated when a reset condition becomes active, even if the clock is not running. Port B has better driving capabilities than the other ports. Port B also serves the functions of various special features of the ATmega640/1280/1281/2560/2561.

2.6.3.3 Port C (PC7~PC0)

Port C is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port C output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port C pins that are externally pulled low will source current if the pull-up resistors are activated. The Port C pins are tri-stated when a reset condition becomes active, even if the clock is not running. Port C also serves the functions of special features of the ATmega640/1280/1281/2560/2561.

2.6.3.4 Port D (PD7~PD0)

Port D is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port D output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port D pins that are externally pulled low will source current if the pull-up resistors are activated. The Port D pins are tri-stated when a

reset condition becomes active, even if the clock is not running. Port D also serves the functions of various special features of the ATmega640/1280/1281/2560/2561.

2.6.3.5 Port E (PE7~PE0)

Port E is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port E output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port E pins that are externally pulled low will source current if the pull-up resistors are activated. The Port E pins are tri-stated when a reset condition becomes active, even if the clock is not running. Port E also serves the functions of various special features of the ATmega640/1280/1281/2560/2561.

2.6.3.6 Port F (PF7~PF0)

Port F serves as analog inputs to the A/D Converter. Port F also serves as an 8-bit bi-directional I/O port, if the A/D Converter is not used. Port pins can provide internal pull-up resistors (selected for each bit). The Port F output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port F pins that are externally pulled low will source current if the pull-up resistors are activated. The Port F pins are tri-stated when a reset condition becomes active, even if the clock is not running. If the JTAG interface is enabled, the pull-up resistors on pins PF7 (TDI), PF5 (TMS), and PF4 (TCK) will be activated even if a reset occurs. Port F also serves the functions of the JTAG interface.

2.6.3.7 Port G (PG5~PG0)

Port G is a 6-bit I/O port with internal pull-up resistors (selected for each bit). The Port G output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port G pins that are externally pulled low will source current if the pull-up resistors are activated. The Port G pins are tri-stated when a reset condition becomes active, even if the clock is not running. Port G also serves the functions of various special features of the ATmega640/1280/1281/2560/2561.

2.6.3.8 Port H (PH7~PH0)

Port H is a 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port H output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port H pins that are externally pulled low will source current if the pull-up resistors are activated. The Port H pins are tri-stated when a reset condition becomes active, even if the clock is not running. Port H also serves the functions of various special features of the ATmega640/1280/2560.

2.6.3.9 Port J (PJ7~PJ0)

Port J is a 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port J output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port J pins that are externally pulled low will source current if the pull-up resistors are activated. The Port J pins are tri-stated when a reset condition becomes active, even if the clock is not running. Port J also serves the functions of various special features of the ATmega640/1280/2560.

2.6.3.10 Port K (PK7~PK0)

Port K serves as analog inputs to the A/D Converter. Port K is a 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port K output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port K pins that are externally pulled low will source current if the pull-up resistors are activated. The Port K pins are tri-stated when a reset condition becomes active, even if the clock is not running. Port K also serves the functions of various special features of the ATmega640/1280/2560.

2.6.3.11 Port L (PL7~PL0)

Port L is a 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port L output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port L pins that are externally pulled low will source current if the pull-up resistors are activated. The Port L pins are tri-stated when a reset condition becomes active, even if the clock is not running. Port L also serves the functions of various special features of the ATmega640/1280/2560.

2.6.3.12 RESET

Reset input. A low level on this pin for longer than the minimum pulse length will generate a reset, even if the clock is not running. The minimum pulse length is given in “System and Reset Characteristics” on page 372. Shorter pulses are not guaranteed to generate a reset.

2.6.3.13 XTAL1

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

2.6.3.14 XTAL2

Output from the inverting amplifier of oscillator.

2.6.3.15 AVCC

AVCC is the supply voltage pin for Port F and the A/D Converter. It should be externally connected to VCC, even if the ADC is not used. If the ADC is used, it should be connected to VCC through a low-pass filter.

2.6.3.16 AREF

This is the analog reference pin for the A/D Converter

2.6.4 Advantages of Arduino Compiler 0017

There are several advantages of arduino.exe 0017 version which are;

* can support for multiple sketch windows.
* The serial monitor now has its own window.
* Comment / Uncomment menu item (in Edit) and keyboard shortcut.
* Increase and Decrease Indent menu items (in Edit) and keyboard shortcuts.
* Support for third-party libraries in the SKETCHBOOK/libraries folder.

* Libraries are now compiled with the sketch, eliminating the delay when switching boards and the need to delete .o files when changing library source code.
* Arduino now comes as an app file (in a dmg) on the Mac.
* Adding the Arduino Nano w/ ATmega328 to the Tools > Board menu.

CHAPTER 3

METHODOLOGY

3.1 Introduction

At the beginning, the method that will be used in this project is dividing into two parts; software design (microcontroller programming) and hardware design. For the software design, we are using an Arduino Atmega1280 which acts as the system memory which enabled the robot motion when input is applied to the system. Then, second part is designing a circuit diagram of microcontroller interfacing with DC motor by firstly design the circuit diagram in Proteus ISIS Professional as circuit simulation software and then fabricate the design into the board. The hex file generated from arduino.exe compiler will be transfer to the Atmega 1280 in Proteus to be simulated. At the end of each parts will be software and hardware testing to examine whether it fulfill the specifications required as introduce in previous chapters.

For the very first step, study and researches about microcontroller types , DC motor concepts and relations between the components applied to the system as well as the theory that rely are applicable and necessary in order to make the project successful and the objectives can be achieved ultimately.

The study and researches about robotics and its automation, its microcontroller interfacing with DC motor, circuit simulation through appropriate software, programming of the microcontroller and also types of components are needed in the circuit design. The study is done through journals, related articles and also with my supervisor guidance's which is as the best reference. Based on the study that I had made, H- Bridge theory circuit is used when interfacing microcontroller with DC Motor to meet the specifications needed and more suitable in this project.

Furthermore, a study about the power consumption also necessary in order to supply the suitable current when the system need to be execute and used. More over, for doing so, I also come out with a simple block diagram and flow chart before proceed to actual hardware circuit.

3.2 Software Implementation

There are two software were used to complete this project; Proteus ISIS 7 Professional PCB design and arduino.exe compiler.

3.2.1 Proteus ISIS 7 Professional PCB design

Proteus PCB design combines the ISIS schematic capture and ARES PCB layout programs to provide a powerful, integrated and easy to use suite of tools for professional PCB Design. All Proteus PCB design products include an integrated shape based on auto router and a basic simulation capability as standard. More advanced routing modes are included in Proteus PCB Design Level 2 and higher whilst simulation capabilities can be enhanced by purchasing the Advanced Simulation option and/or micro-controller simulation capabilities the products are offered at a number of levels which offer increasing levels of functionality and design capacity. Furthermore, the simulation also included components of Atmega 1280 and other components are provided which will be needed in this project.

The figure 3.22 in next page shows a simple designing page layout of Proteus 7 professional PCB design.

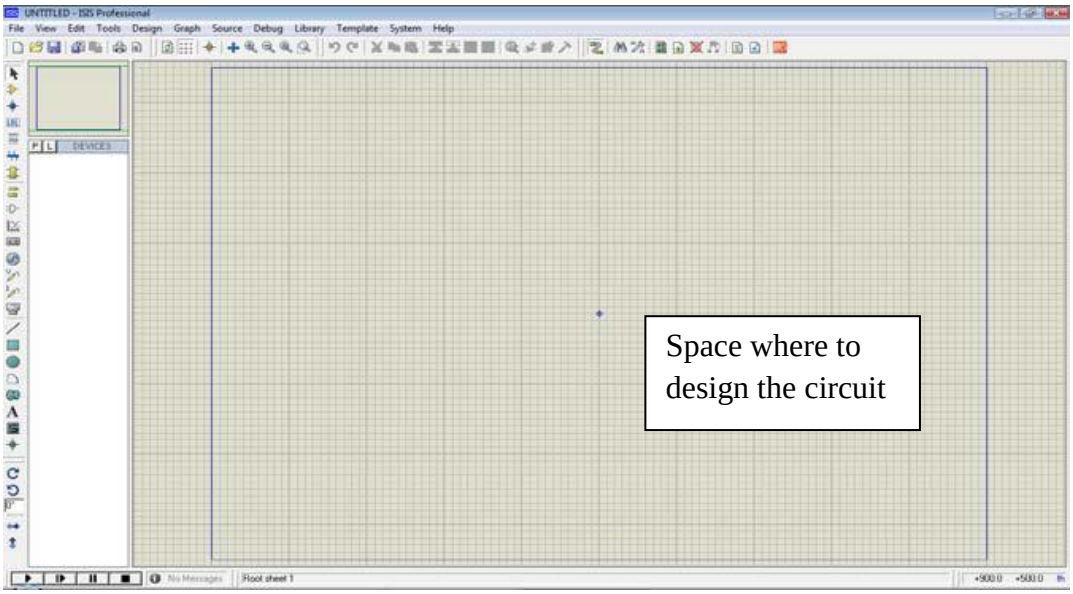


Figure 3.1: Designing page of Proteus 7 professional

3.2.2 Arduino Compiler

Arduino is an open-source physical computing platform based on a simple input/output board and a development environment that implements the processing/wiring language. The Arduino can be used to develop stand-alone interactive objects or can be connected to software on your computer. In this project, the Arduino version 0017 - 2009.07.25 is used as it can support Atmega 1280 programming in C language. There are many updated items that have been added to this version like improving the accuracy of the baud rate calculations for serial communication (fixing double-speed problems on 8 MHz Arduino boards).

Figure 3.2: Arduino.exe

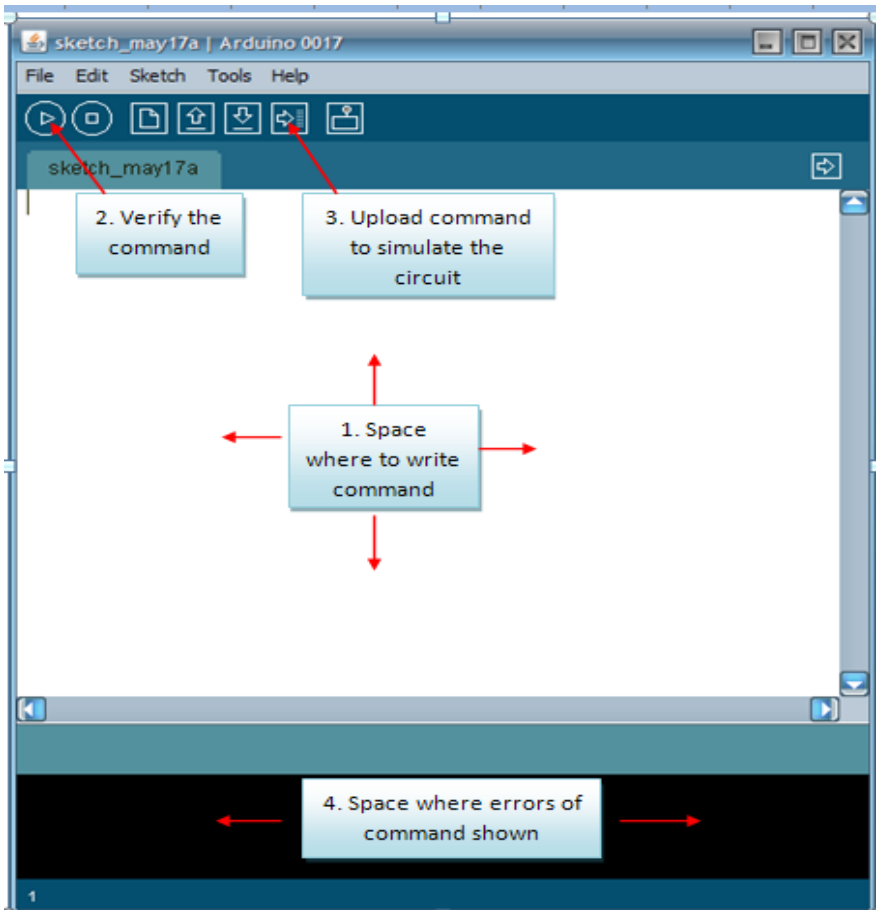


Figure 3.3: Arduino.exe command space

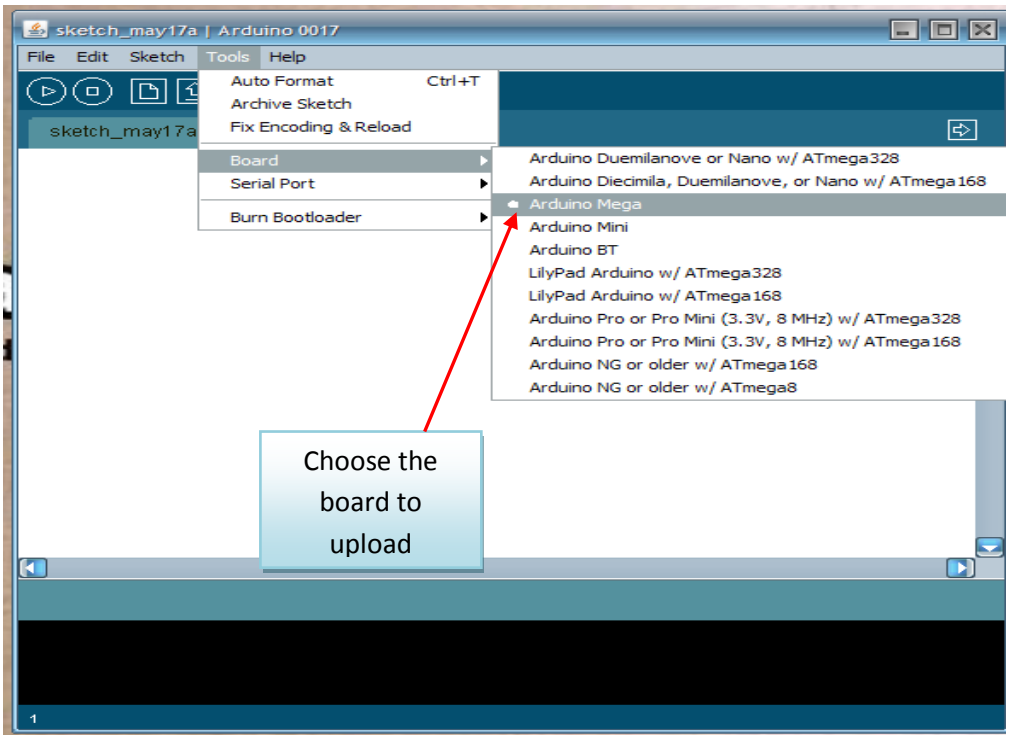


Figure 3.4: Choose board type to upload

Figure 3.5: Choose serial port for Arduino

3.2.3 Flow chart of software development

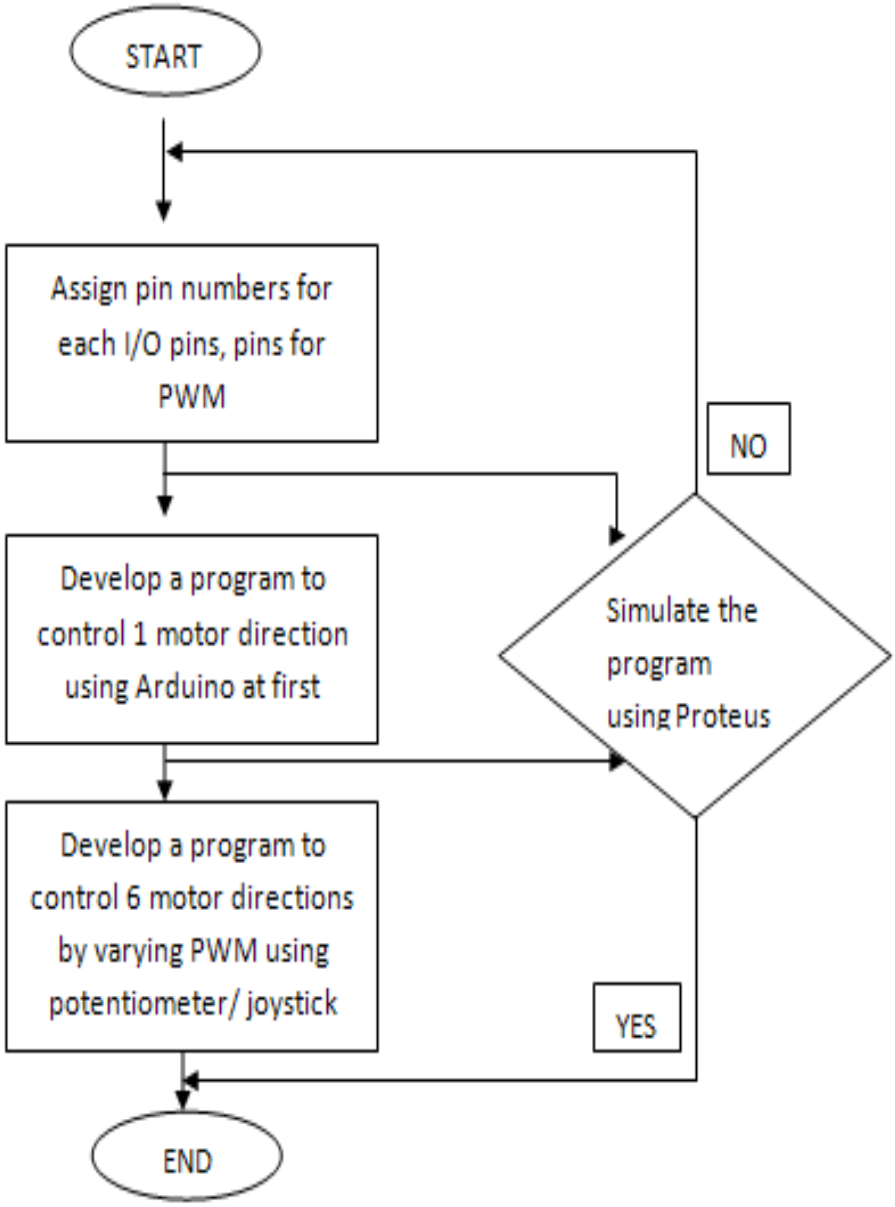


Figure 3.6: Flow chart for software development

3.3 Hardware Implementation

The hardware part is named as a controlling circuit. Controlling circuit includes Arduino Mega ADK Board of 2560 and H-bridge circuits build with transistors TIP142 N-channel, diodes, and resistors. For this project, the Arduino Mega and H-bridge circuits play an important role. The function and operation are briefly explained in next part. There is another important hardware which already established that is the HOBO robot chassis which will be implemented with all these components. Below is a simple block diagram for the system.

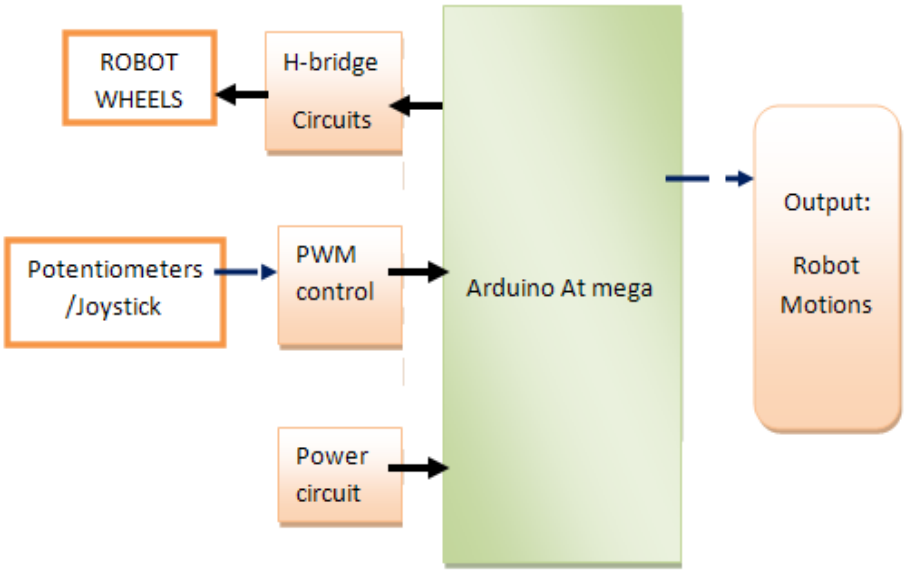


Figure 3.7: Block diagram to design the robot system

3.3.1 H bridge circuit

The H-bridge circuit is the circuit that I had established, by referring to the schematic diagram below, the circuit comprises of five NPN BJT transistors of TIP 142, four diodes and four resistors for each single H-bridge printed on board. Each provided with two terminals to connect to DC motor and five terminals connected to pin forward, reverse, and PWM in order to control forward and reverse direction and to control speed of DC motor. The PWM pin must be in high condition which needs to be set greater than 1V in order to make the circuit functioned as well as forward and reverse pins. Moreover, the diodes used in the circuit as to prevent any destroy to microcontroller while interfacing with it.

Two transistors (Q1 and Q4) were controlled forward direction, while two transistors (Q2 and Q4) were controlled reverse direction. A transistor Q5 placed between the ground and emitter (E) pin of two transistors (Q2 and Q4) as PWM transistor to control the speed of the DC motor.

Theory of operations of the circuit is; when the forward pin is in high condition, the motor will run forward and when the reverse switch is high condition, the motor will run reverse. , the condition of PWM switch must always be in high condition. Through PWM, the speed of motor can be varied by adjusted the duty cycle of the PWM.

From the circuit overview, a harm to the components will happen if the two pins of forward and reverse accidentally at high condition at the same time, therefore, for safety in the circuit, some modifications need to be implemented in programming which, if both pins at high condition at the same time, a command will be sent to microcontroller to enable the robot stopped.

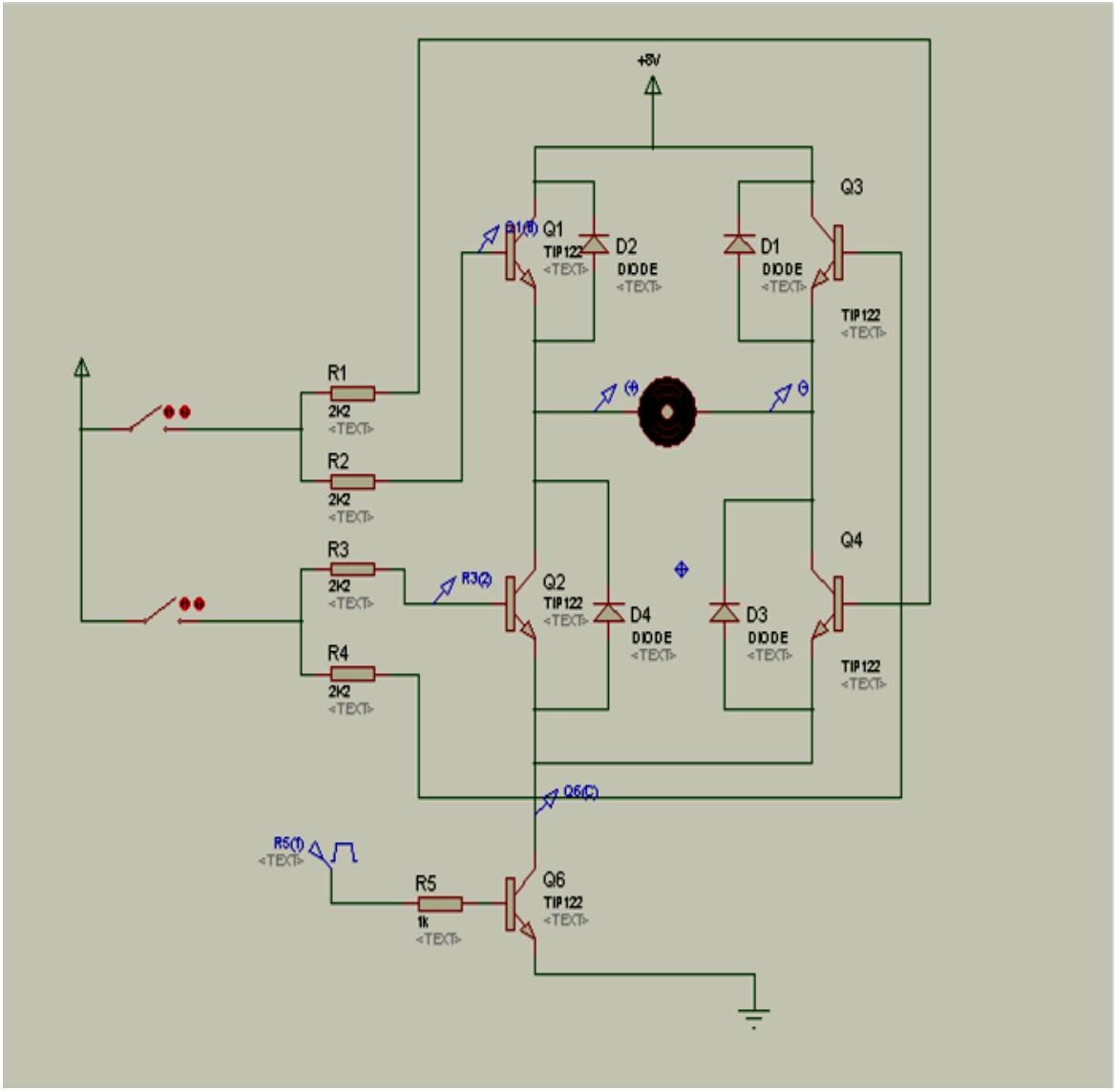


Figure 3.8: H bridge circuit schematic diagram

3.3.2 H bridge circuit testing for 1 motor with Arduino328

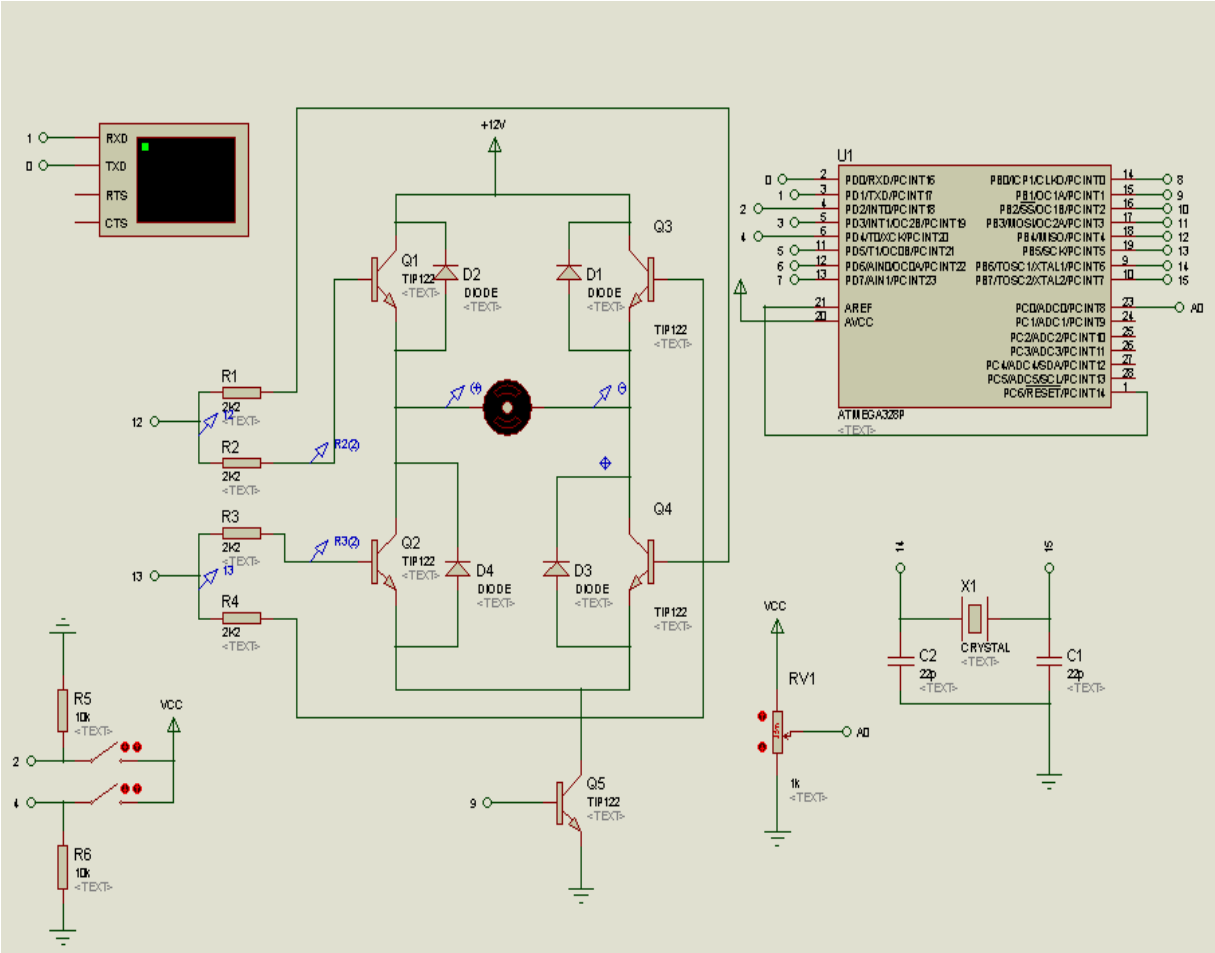


Figure 3.9: Control motor direction by varying the speed using potentiometer

3.3.3 Interfacing with a Joystick

The joystick has two potentiometers that allow us to measure the movement of the stick in 2-D. Potentiometers are variable resistors and, in a way, they act as sensors provided with a variable voltage depending on the rotation of the device around its shaft. The kind of program that we need to monitor the joystick has to make polling to two of the analog pins. We can send these values back to the computer, but then we face the classic problem that the transmission over the communication port has to be made with 8bit values, while our DAC (Digital to Analog Converter - that is measuring the values from the potentiometers in the joystick) has a resolution of 10bits. In other words this means that the potentiometers are characterized with a value between 0 and 1024. The following code includes a method called treatValue () that is transforming the measurements into a value between 0 and 255 and sends it in ASCII back to the computer.

Demo Sketch :

```
//Create variables for each button on the Joystick Shield to assign the pin numbers
char button0=3, button1=4, button2=5, button3=6;
char sel=2;

void setup(void){
  pinMode(sel, INPUT);      //Set the Joystick 'Select'button as an input
  digitalWrite(sel, HIGH);  //Enable the pull-up resistor on the select button
  pinMode(button0, INPUT);      //Set the Joystick button 0 as an input
  digitalWrite(button0, HIGH);  //Enable the pull-up resistor on button 0
  pinMode(button1, INPUT);      //Set the Joystick button 1 as an input
  digitalWrite(button1, HIGH);  //Enable the pull-up resistor on button 1
  pinMode(button2, INPUT);      //Set the Joystick button 2 as an input
  digitalWrite(button2, HIGH);  //Enable the pull-up resistor on button 2
  pinMode(button3, INPUT);      //Set the Joystick button 3 as an input
  digitalWrite(button3, HIGH);  //Enable the pull-up resistor on button 3
   Serial.begin(9600);           //Turn on the Serial Port at 9600 bps
}

void loop(void) {
 Serial.print(analogRead(0));  //Read the position of the joysticks X axis and print it on the serial port.
  Serial.print(",");
  Serial.print(analogRead(1)); //Read the position of the joysticks Y axis and print it on the serial port.
  Serial.print(",");
  Serial.print(digitalRead(sel));       //Read the value of the select button and print it on the serial port.
  Serial.print(digitalRead(button0));   //Read the value of the button 0 and print it on the serial port.
  Serial.print(digitalRead(button1));   //Read the value of the button 1 and print it on the serial port.
  Serial.print(digitalRead(button2));   //Read the value of the button 2 and print it on the serial port.
  Serial.println(digitalRead(button3)); //Read the value of the button 3 and print it on the serial port.

  //Wait for 100 ms, then go back to the beginning of 'loop' and repeat.
  delay(100);}
```

Through using joystick with Arduino where we hook up a potentiometer and read the voltage on the wiper with the ADC, we do not need to use another resistor to measure the potentiometers in the joystick by looking at the schematic of a joystick; only two of the terminals of the potentiometer are wired up. The way we would normally read the position of a potentiometer on the Arduino is to hook up the two outside terminals of the potentiometer to +5V and ground and connect the wiper of the potentiometer to one of the analog channels and read the voltage at the wiper terminal. The reason this works is because the entire potentiometer acts as a voltage divider with current flowing through the two outside terminals. The location of the wiper is basically where the two resistors in the voltage divider are split, and the total resistance of these two resistors is always the value of the potentiometer.

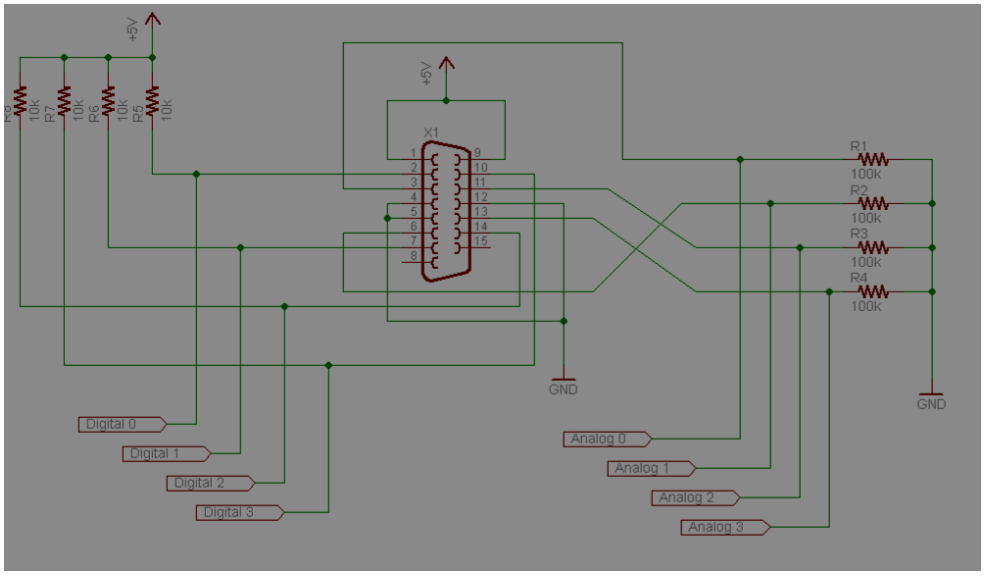


Figure 3.10: Joystick Dongle Schematic

To read the joystick potentiometer, we'll need to set up our own voltage divider with a 100K Ohm resistor:

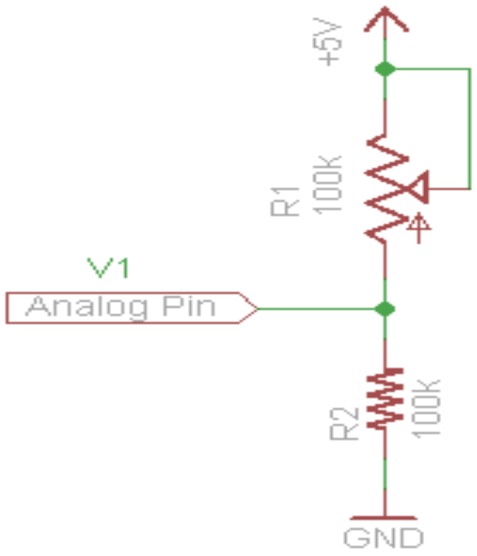


Figure 3.11: Basic Voltage Divider Circuit for Reading a Potentiometer

3.3.4 Flow chart for hardware development

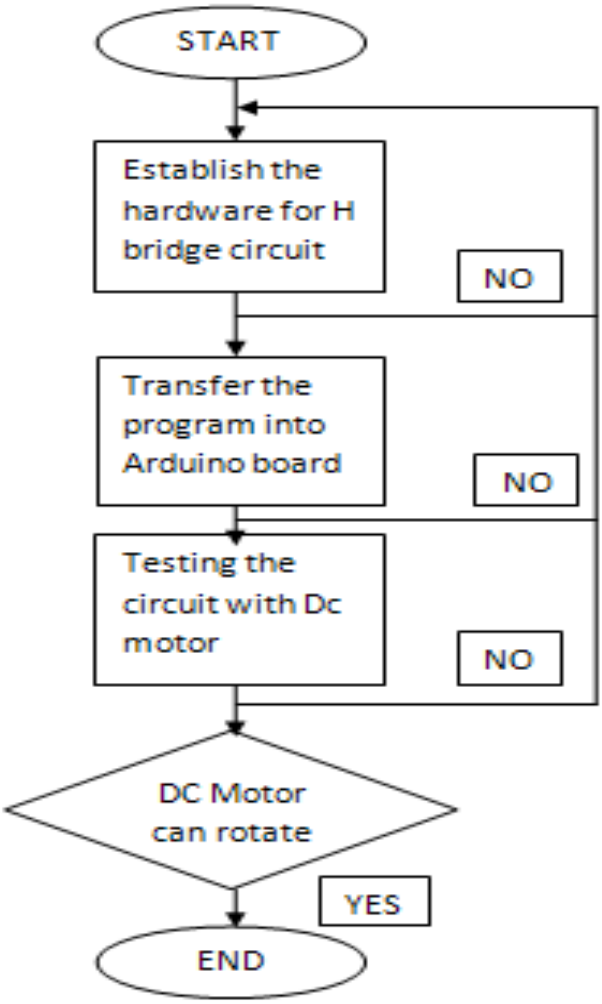


Figure 3.12: Flow chart for hardware development

CHAPTER 4

RESULTS AND DISCUSSIONS

4.1 Introduction

In this section, I will show the results and analysis of projects which divided into results for simulation and results from hardware development. Various tests are conducted in this project. The result of the testing reveal the system has achieved of substantial goal. Since the system developed with some high voltage components like DC motors, it should be tested independently for ensure the suitability of the component to this project and also for safety purpose. Finally all the components are integrated and tested for its functionality.

4.2 Software result

4.2.1 PWM from simulations

The programming language applied in this project is in C language. The arduino Atmega is programmed using C language in arduino.exe compiler. Below are shown some parts of programming that had been applied in this project.

4.2.1.1 Program Declaration Pins

Program declaration pins were done regarding the pins for input/output pins and pins for PWM in Arduino microcontroller for forward and reverse direction and to control PWM. Below is the programming;

```
int fwR=31;       //set pin output for motor left move forward
int revR=33;      //set pin output for motor left move reverse
int fwL=37;       //set pin output for motor right move forward
int revL=35;      //set pin output for motor left move reverse
int pwmR=3;     //set pin output for pwm motor at right side
int pwmL=2;     //set pin output for pwm motor at left side
long x, y;          //x and y axes of joystick
int x_in, y_in;  //analog value at x and y axes while controlling joystick
```

4.2.1.2 Program Setting the Motions

Program for robot automation of robot is divided in two: for program setting the motions and program condition of motions. In program setting the motions, there were eight directions (output) the robot can have: forward, reverse, turn left, turn right, forward right, forward left, reverse right and reverse left. At the program setting, the eight conditions were set up in void function and then the void function is then called again in void loop function based on conditions as stated in the several void directions in programming below.

```
void setup()
{ Serial.begin(9600);      // set up Serial library at 9600 bps
pinMode(fwR,OUTPUT);
pinMode(revR,OUTPUT);
pinMode(fwL,OUTPUT);
pinMode(revL,OUTPUT);}

void forward()    //robot move forward direction
{digitalWrite(fwL,HIGH);
digitalWrite(fwR,HIGH);
analogWrite(pwmL,y);
analogWrite(pwmR,y);
delay(200);}

void reverse()  //robot move reverse direction
{
digitalWrite(revR, HIGH);
digitalWrite(revL,HIGH);
analogWrite(pwmL,(0-y));
analogWrite(pwmR,(0-y));
delay(100);}
```

4.2.1.3 Program Condition of Motion

```
void loop()

{

x_in = analogRead(1);

y_in = analogRead(0);


x = map(x_in, 508, 1023, 0, 255);  // mapped bits resolutions from 10 bits to 8 bits in analog value

y = map(y_in, 503, 1023, 0, 255);


if(x==0 && y>0)    //analog value at joystick at x and y axes

{

forward();

}


if(x==0 && y<0)    //analog value at joystick at x and y axes

{

reverse();

}
```

4.2.1 Interfacing with joystick

While interfacing the joystick with Arduino, we need to determine the position at x axes and y axes in analog value. From there, we can determine how much current need to be applied at each zone direction it will turn to by varying two potentiometers in the joystick. Below is the result that I had summarized for each position along x-axes and y-axes in analog value through arduino.exe compiler.

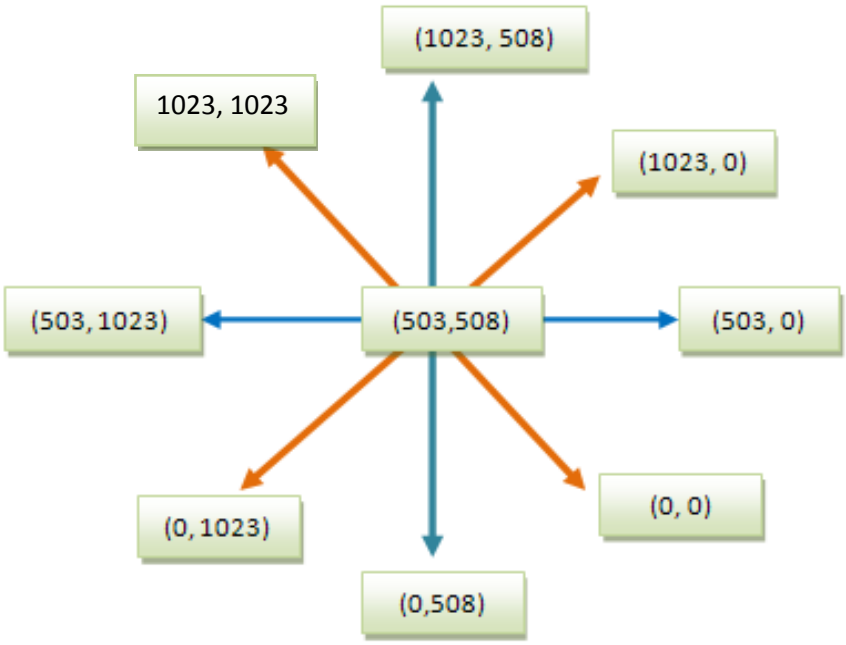


Figure 4.1: Results for joystick direction when interfacing with arduino in analog value

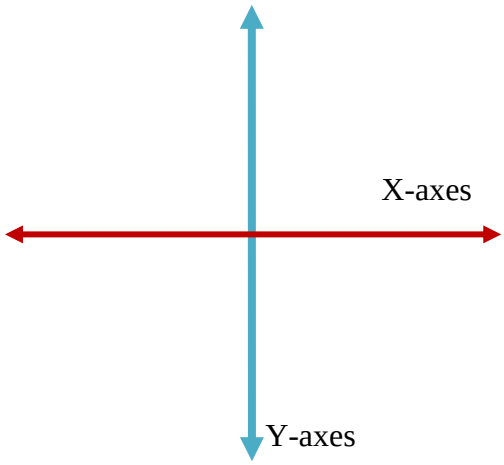


Figure 4.2: Joystick represents as X-axes and Y-axes

4.3 Hardware result

4.3.1 Testing with joystick

The joystick is as the controlling system for the robot automation. When the input is applied to the joystick, the robot will be able to move according to which direction pull or push. The joystick is connected to Arduino board as shown in figures below.

Figure 4.3: Joystick consist of 4 switch buttons and thumb push button

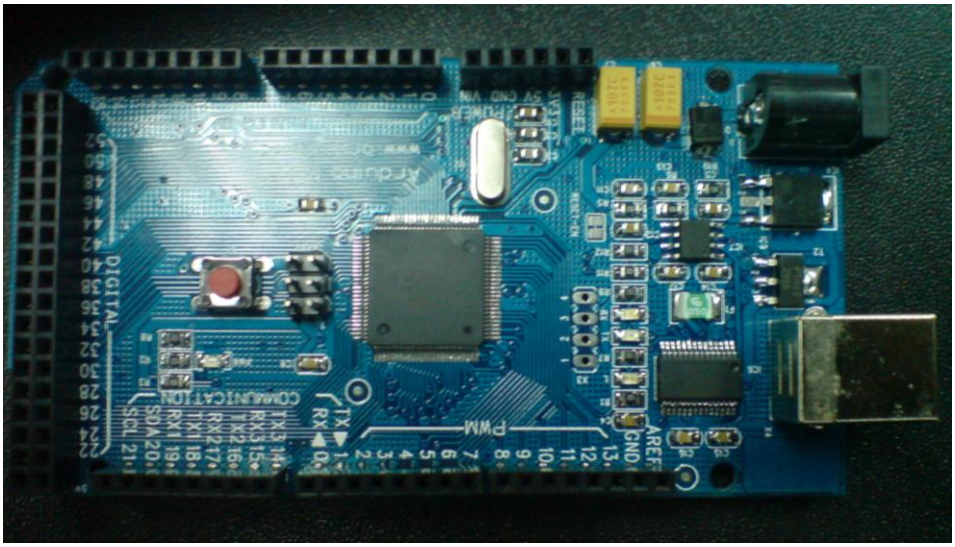


Figure 4.4: Arduino Atmega 2560 board

Figure 4.5: Testing Joystick with arduino and LEDs

The figure above shows result while testing joystick with Arduino and eight LEDs. The LEDs represents as the output that generated to 8 directions of HOBO robots. While push the joystick to front, the first LED will ON, when pull down the second LED ON and followed by next LEDs when push left, right, left up, right up, right down and left down. By varied the joystick positions by half or full, the dimmer of the LED can be controlled which point to PWM values or speed of motors.

4.3.2 H bridge circuit testing with a simple Dc motor

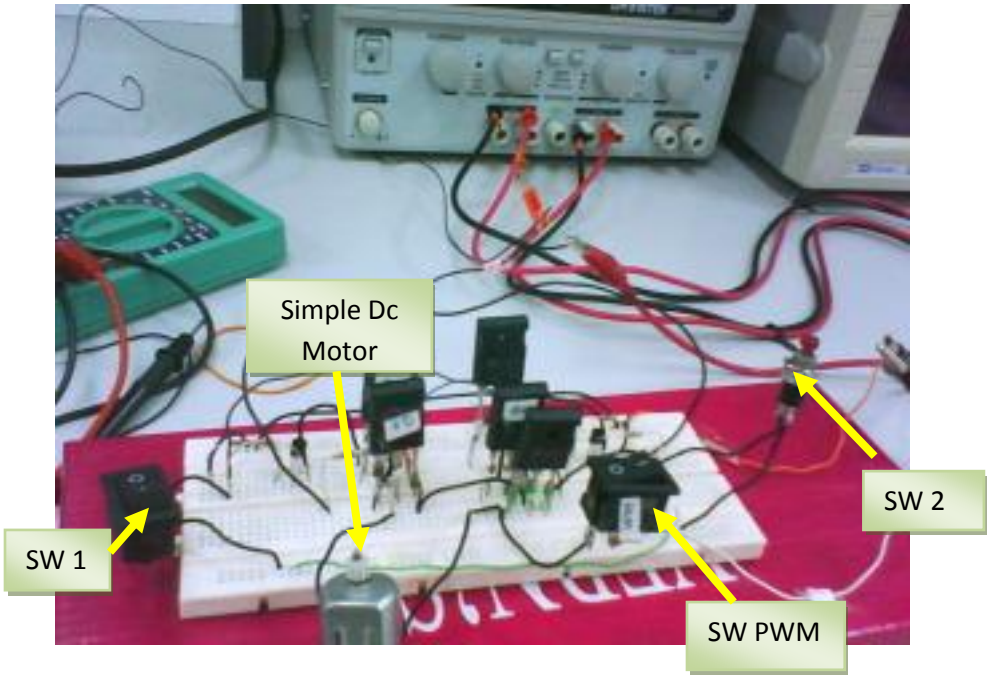


Figure 4.6: Testing circuit on breadboard

From figure above, shows that the transistors and other components were built on breadboard to test it with a simple dc motor. SW1 is for forward motor direction switch and SW2 for reverse motor direction switch. The SW PWM is as enable input to these two switches. If the voltage supplied to SW PWM is equal to zero, the circuit is not functioning though SW1 or SW2 is ON. Furthermore, both SW1 and SW2 cannot be ON at the same time because it can harm the transistors and short circuit will happen. Therefore, as for safety for the hardware, some modifications are needed in programming to protect the circuit damaged as stated in previous chapter. After that, the components were installed onto PCB board as shown below to produce a reliable system for the project.

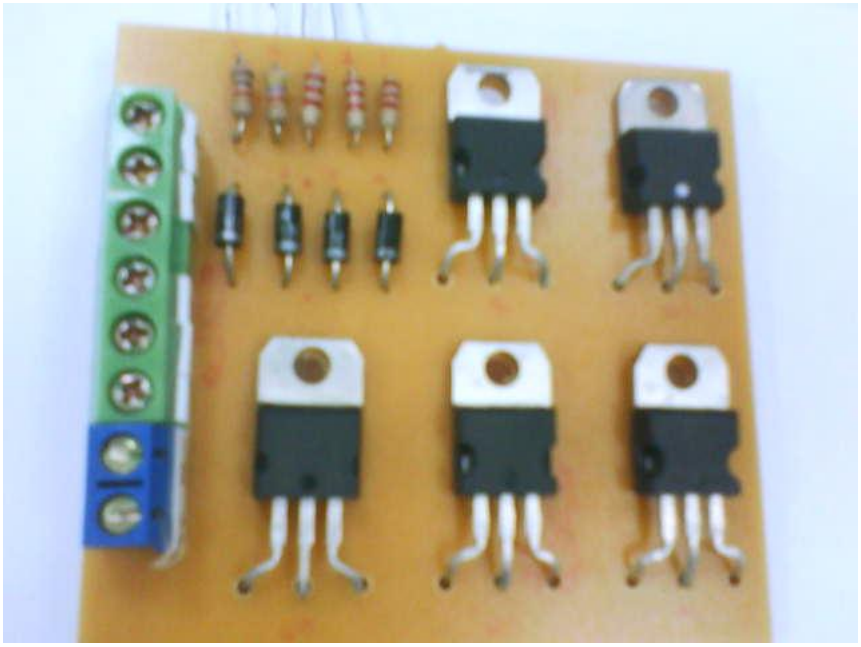


Figure 4.7: Testing on PCB board using transistor TIP 142(NPN channel)

4.3.3 Testing with HOBO DC motor

In this part shows the hardware results of testing DC motor with the H-bridge circuit. Unfortunately, the results that I had come out did not support the DC motor of HOBO robot. By referring to datasheets of transistors TIP 142 and TIP 147, their peak current is up 20 Ampere and collector voltage is up to 100 V. By comparing to the specifications of DC motor of HOBO robot, the starting current is at 4 Ampere and voltage at 24V.Theoretically, the transistors used should support the DC motor, but practically, the DC motor need starting current and voltage at least ten times than its actual current in order to operate. Therefore, the H-Bridge circuit has undergone short circuit and overload due to exceed of limit current flows while testing with the HOBO's DC motor. On the other hand, other suitable components need to be replaced in this circuit which has continuous current up to 40 Ampere.

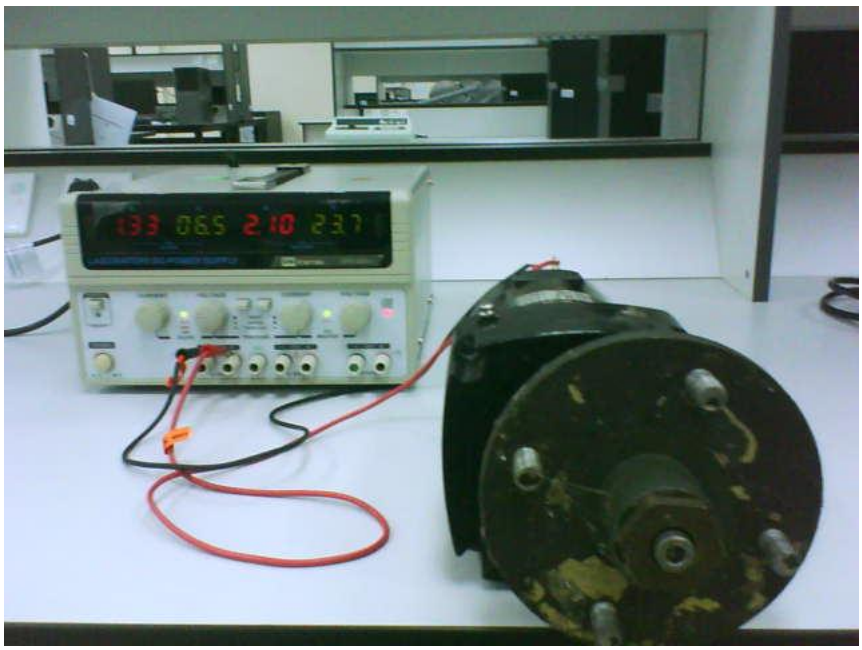


Figure 4.8: DC motor robots testing

Figure above shows while testing Dc Motor using parallel power supply connection in order to produce up to 24V and 4.4A.

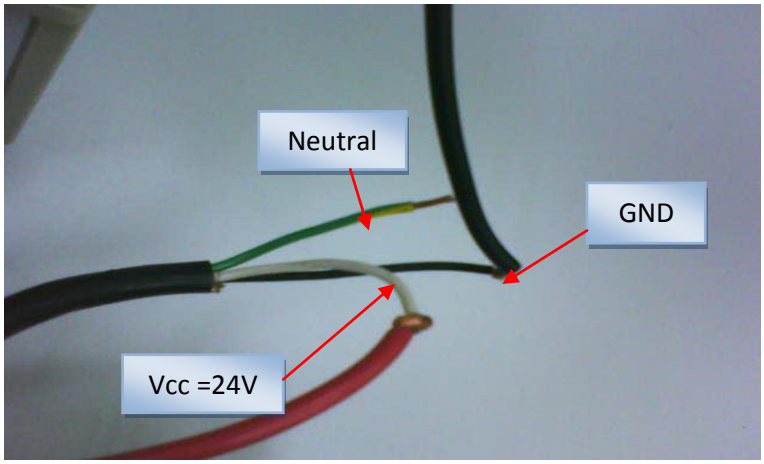


Figure 4.9: Wire connection of DC motor

Referring to figure above, the wire in black colours is for ground, the white wire for Vcc and the yellow green wire for neutral.

4.4 Discussion

In the project, I had encountered various problems during the whole project while testing the hardware. Because of some problems encountered and lack of time in order to achieve the main objectives, then I proceed to next step by using two wheels robot, and then we are able to distinguish 8 motion of the robot. The 8 motions of robot are:

i. Forward

ii. Reverse

iii. Turn to left(TurnL)

iv. Turn to right(TurnR)

v. Forward and turn to left(ForwardL)

vi. Forward and turn to right(ForwardR)

vii. Reverse and turn to left(ReverseL)

viii. Reverse and turn to right(ReverseR)

In enabling both motor directions, the joystick is push to forward, reverse, turn to left, right, reverse left, reverse right, forward left, and forward right and the robot move according to which direction the joystick is push by enabling pins in Arduino connected to H-bridge circuits. Besides, two pins of H-bridge circuit were connected to PWM pins in Arduino in order to control the speed of motor automation.

By using the same concepts of H-bridge circuits operations and same concepts of programming in Arduino, we can control the HOBO robot which consists of 6 wheels. For the six wheels, each wheel needs it independent H-bridge circuit to enable forward and reverse directions and also PWM pin.

The actual expected result of the project is the HOBO robot will be able to move when the input (user mode or the joystick) is push or pull. When the power is ON, the system is enabled and when the joystick is push, because of DC motor rotation, the wheels at the left side will rotate clockwise (forward) and the wheels at the right side will rotate anticlockwise (reverse). The net torque of rotation of left side will cause the robot wheels at that side to move forward and the net torque of rotation of right side will cause the robot wheels at that side will move reverse. When all the wheels at the both side rotate simultaneously, with the same PWM value generated from potentiometer, the HOBO robot will be able to move forward.

CHAPTER 5

CONCLUSIONS AND RECOMMENDATIONS

5.1 Introduction

In this last chapter, will be include with the conclusions through all the projects overall as well as recommendations in order to upgrade the system and to overcome all the problems arise during the project completion.

5.2 Conclusion

As a conclusion, the objectives of the projects to explore about microcontroller of Arduino Atmega and DC motor interfacing in system design to establish hardware (joystick as the input) and make programming has been achieved, but the actual output for the HOBO robot is not comply because of some circumstances occur while testing the circuit which is the components transistors TIP 142 in H-bridge circuit cannot support the high ampere Dc Motor of HOBO robots. Thus, through hardware development, the project was preceding using two wheels robot. The two wheels robot is

able to move according to which direction the joystick is push or pull. The robot able to automate in eight directions and its speed of motion can be controlled as well.

5.3 Recommendations

Although the objective of this project is not achieved yet, there are still some recommendations to the system include software and hardware development. Here are some recommendations that I had outlined:

- Develop more applications of the HOBO robot such as in image processing which can beneficial to ATM
- More researches need to be done about the high ampere of DC motor robot in theory and practical
- Using more efficient power losses components over BJTs transistors and meet specifications of high current DC motor
- Longer time is needed to complete the whole project simulation using software and in hardware troubleshooting

5.4 Costing and Commercialization

The overall cost of the whole project is based on the hardware development. As discussed in previous chapter, the hardware development consists of one main board. Therefore the whole project coat is depends on the cost of electronic devices.

The total cost for the development system is around RM350.00. The system can be redesigned to suit and meet the requirement of the ATM organization. Needless to say, the system has highly potential to be commercialized and can bring much benefits to both organizations.

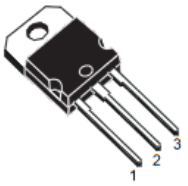
APPENDIX A

TIP 142 DATASHEET

TIP140/141/142
TIP145/146/147

COMPLEMENTARY SILICON POWER DARLINGTON TRANSISTORS

- TIP141, TIP142, TIP145 AND TIP147 ARE STMicroelectronics PREFERRED SALESTYPES
- COMPLEMENTARY PNP - NPN DEVICES
- MONOLITHIC DARLINGTON CONFIGURATION
- INTEGRATED ANTIPARALLEL COLLECTOR-EMITTER DIODE



TO-218

APPLICATIONS

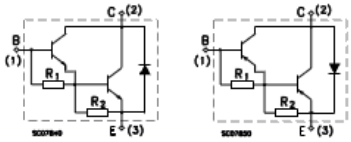
- LINEAR AND SWITCHING INDUSTRIAL EQUIPMENT

DESCRIPTION

The TIP140, TIP141 and TIP142 are silicon Epitaxial-Base NPN power transistors in monolithic Darlington configuration, mounted in TO-218 plastic package. They are intented for use in power linear and switching applications.

The complementary PNP types are TIP145, TIP146 and TIP147 respectively.

INTERNAL SCHEMATIC DIAGRAM



R_1 Typ. = 5 KΩ R_2 Typ. = 150 Ω

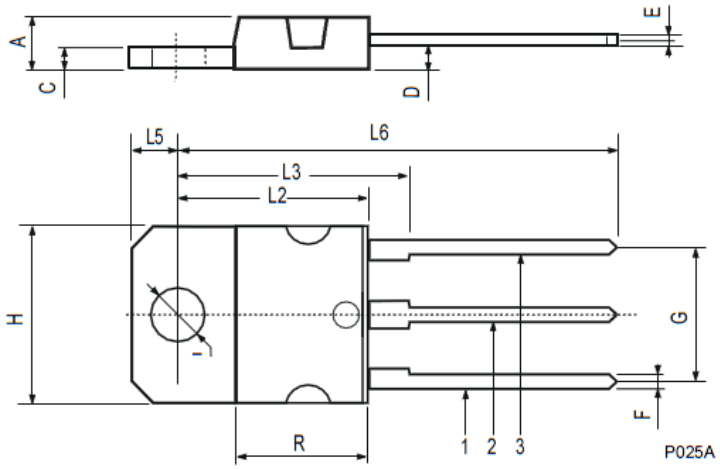
ABSOLUTE MAXIMUM RATINGS

Symbol	Parameter		Value			Unit
		NPN	TIP140	TIP141	TIP142	
		PNP	TIP145	TIP146	TIP147	
V_{CBO}	Collector-Base Voltage (I_E = 0)		60	80	100	V
V_{CEO}	Collector-Emitter Voltage (I_B = 0)		60	80	100	V
V_{EBO}	Emitter-Base Voltage (I_C = 0)		5			V
I_C	Collector Current		10			A
I_{CM}	Collector Peak Current		20			A
I_B	Base Current		0.5			A
P_{tot}	Total Dissipation at $T_{case} \leq 25$ °C		125			W
T_{stg}	Storage Temperature		-65 to 150			°C
T_j	Max. Operating Junction Temperature		150			°C

For PNP types voltage and current values are negative.

TO-218 (SOT-93) MECHANICAL DATA

DIM.	mm			inch		
	MIN.	TYP.	MAX.	MIN.	TYP.	MAX.
A	4.7		4.9	0.185		0.193
C	1.17		1.37	0.046		0.054
D		2.5			0.098	
E	0.5		0.78	0.019		0.030
F	1.1		1.3	0.043		0.051
G	10.8		11.1	0.425		0.437
H	14.7		15.2	0.578		0.598
L2	–		16.2	–		0.637
L3		18			0.708	
L5	3.95		4.15	0.155		0.163
L6		31			1.220	
R	–		12.2	–		0.480
Ø	4		4.1	0.157		0.161



THERMAL DATA

$R_{thj\text{-}case}$	Thermal Resistance Junction-case	Max	1	°C/W

ELECTRICAL CHARACTERISTICS (T_{case} = 25 °C unless otherwise specified)

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
I_{CBO}	Collector Cut-off Current (I_E = 0)	for **TIP140/145**	V_{CB} = 60 V			1	mA
		for **TIP141/146**	V_{CB} = 80 V			1	mA
		for **TIP142/147**	V_{CB} = 100 V			1	mA
I_{CEO}	Collector Cut-off Current (I_B = 0)	for **TIP140/145**	V_{CE} = 30 V			2	mA
		for **TIP141/146**	V_{CE} = 40 V			2	mA
		for **TIP142/147**	V_{CE} = 50 V			2	mA
I_{EBO}	Emitter Cut-off Current (I_C = 0)	V_{EB} = 5 V				2	mA
$V_{CEO(sus)}$*	Collector-Emitter Sustaining Voltage (I_B = 0)	I_C = 30 mA					
		for **TIP140/145**		60			V
		for **TIP141/146**		80			V
		for **TIP142/147**		100			V
$V_{CE(sat)}$*	Collector-Emitter Saturation Voltage	I_C = 5 A	I_B = 10 mA			2	V
		I_C = 10 A	I_B = 40 mA			3	V
$V_{BE(on)}$*	Base-Emitter Voltage	I_C = 10 A	V_{CE} = 4 V			3	V
h_{FE}*	DC Current Gain	I_C = 5 A	V_{CE} = 4 V	1000			
		I_C = 10 A	V_{CE} = 4 V	500			
	RESISTIVE LOAD	I_C = 10 A	I_{B1} = 40 mA				
t_{on}	Turn-on Time	I_{B2} = -40 mA	R_L = 3 Ω		0.9		µs
t_{off}	Turn-off Time				4		µs

For PNP types voltage and current values are negative.

* Pulsed: Pulse duration = 300 µs, duty cycle 1.5 %

APPENDIX B

ATMEGA 2560 DATASHEET

Features

- High Performance, Low Power Atmel® AVR® 8-Bit Microcontroller
- Advanced RISC Architecture
  - 135 Powerful Instructions – Most Single Clock Cycle Execution
  - 32 x 8 General Purpose Working Registers
  - Fully Static Operation
  - Up to 16 MIPS Throughput at 16MHz
  - On-Chip 2-cycle Multiplier
- High Endurance Non-volatile Memory Segments
  - 64K/128K/256KBytes of In-System Self-Programmable Flash
  - 4Kbytes EEPROM
  - 8Kbytes Internal SRAM
  - Write/Erase Cycles:10,000 Flash/100,000 EEPROM
  - Data retention: 20 years at 85°C/ 100 years at 25°C
  - Optional Boot Code Section with Independent Lock Bits
    - In-System Programming by On-chip Boot Program
    - True Read-While-Write Operation
  - Programming Lock for Software Security
    - Endurance: Up to 64Kbytes Optional External Memory Space
- Atmel® QTouch® library support
  - Capacitive touch buttons, sliders and wheels
  - QTouch and QMatrix® acquisition
  - Up to 64 sense channels
- JTAG (IEEE std. 1149.1 compliant) Interface
  - Boundary-scan Capabilities According to the JTAG Standard
  - Extensive On-chip Debug Support
  - Programming of Flash, EEPROM, Fuses, and Lock Bits through the JTAG Interface
- Peripheral Features
  - Two 8-bit Timer/Counters with Separate Prescaler and Compare Mode
  - Four 16-bit Timer/Counter with Separate Prescaler, Compare- and Capture Mode
  - Real Time Counter with Separate Oscillator
  - Four 8-bit PWM Channels
  - Six/Twelve PWM Channels with Programmable Resolution from 2 to 16 Bits (ATmega1281/2561, ATmega640/1280/2560)
  - Output Compare Modulator
  - 8/16-channel, 10-bit ADC (ATmega1281/2561, ATmega640/1280/2560)
  - Two/Four Programmable Serial USART (ATmega1281/2561, ATmega640/1280/2560)
  - Master/Slave SPI Serial Interface
  - Byte Oriented 2-wire Serial Interface
  - Programmable Watchdog Timer with Separate On-chip Oscillator
  - On-chip Analog Comparator
  - Interrupt and Wake-up on Pin Change
- Special Microcontroller Features
  - Power-on Reset and Programmable Brown-out Detection
  - Internal Calibrated Oscillator
  - External and Internal Interrupt Sources
  - Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby, and Extended Standby
- I/O and Packages
  - 54/86 Programmable I/O Lines (ATmega1281/2561, ATmega640/1280/2560)
  - 64-pad QFN/MLF, 64-lead TQFP (ATmega1281/2561)
  - 100-lead TQFP, 100-ball CBGA (ATmega640/1280/2560)
  - RoHS/Fully Green
- Temperature Range:
  - -40°C to 85°C Industrial
- Ultra-Low Power Consumption
  - Active Mode: 1MHz, 1.8V: 500µA
  - Power-down Mode: 0.1µA at 1.8V
- Speed Grade:
  - ATmega640V/ATmega1280V/ATmega1281V:
    - 0 - 4MHz @ 1.8V - 5.5V, 0 - 8MHz @ 2.7V - 5.5V
  - ATmega2560V/ATmega2561V:
    - 0 - 2MHz @ 1.8V - 5.5V, 0 - 8MHz @ 2.7V - 5.5V
  - ATmega640/ATmega1280/ATmega1281:
    - 0 - 8MHz @ 2.7V - 5.5V, 0 - 16MHz @ 4.5V - 5.5V
  - ATmega2560/ATmega2561:
    - 0 - 16MHz @ 4.5V - 5.5V

8-bit Atmel Microcontroller with 64K/128K/256K Bytes In-System Programmable Flash

ATmega640/V
ATmega1280/V
ATmega1281/V
ATmega2560/V
ATmega2561/V

Preliminary Summary

2549NS–AVR–05/11

1. Pin Configurations

Figure 1-1. TQFP-pinout ATmega640/1280/2560

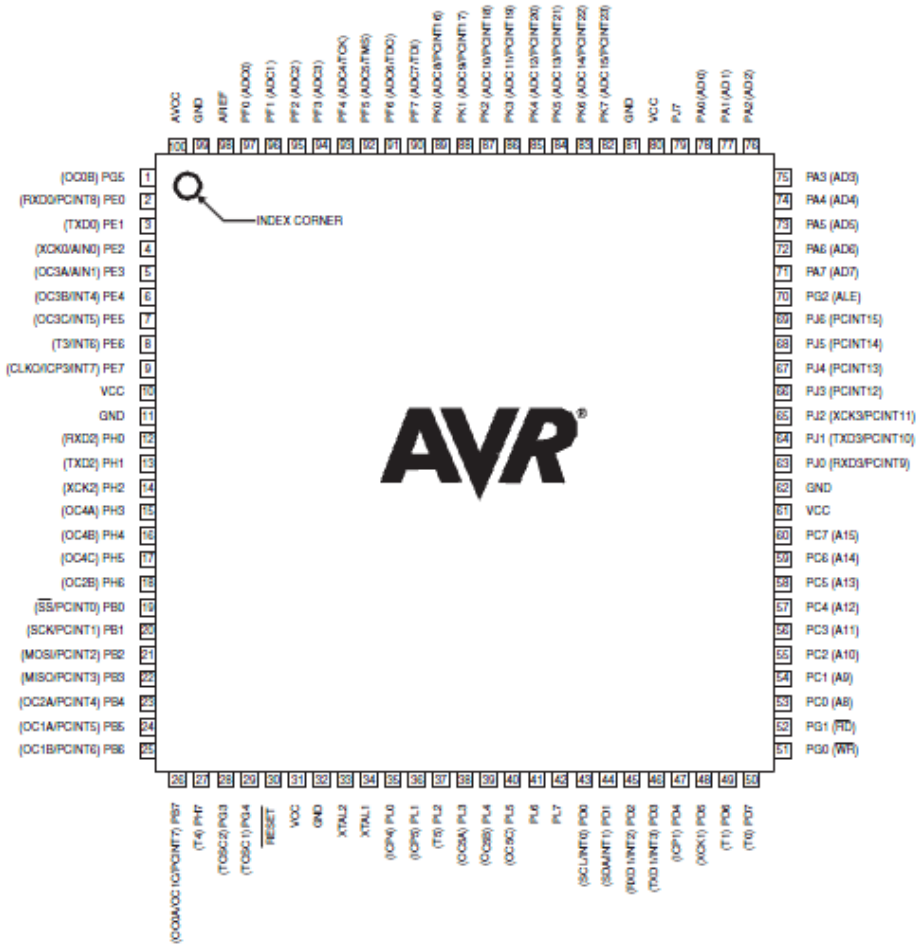


Figure 1-2. CBGA-pinout ATmega640/1280/2560

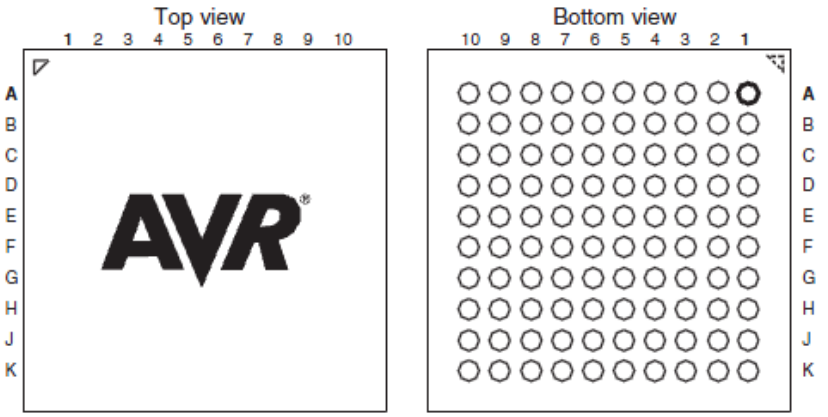


Table 1-1. CBGA-pinout ATmega640/1280/2560

	1	2	3	4	5	6	7	8	9	10
A	GND	AREF	PF0	PF2	PF5	PK0	PK3	PK6	GND	VCC
B	AVCC	PG5	PF1	PF3	PF6	PK1	PK4	PK7	PA0	PA2
C	PE2	PE0	PE1	PF4	PF7	PK2	PK5	PJ7	PA1	PA3
D	PE3	PE4	PE5	PE6	PH2	PA4	PA5	PA6	PA7	PG2
E	PE7	PH0	PH1	PH3	PH5	PJ6	PJ5	PJ4	PJ3	PJ2
F	VCC	PH4	PH6	PB0	PL4	PD1	PJ1	PJ0	PC7	GND
G	GND	PB1	PB2	PB5	PL2	PD0	PD5	PC5	PC6	VCC
H	PB3	PB4	RESET	PL1	PL3	PL7	PD4	PC4	PC3	PC2
J	PH7	PG3	PB6	PL0	XTAL2	PL6	PD3	PC1	PC0	PG1
K	PB7	PG4	VCC	GND	XTAL1	PL5	PD2	PD6	PD7	PG0

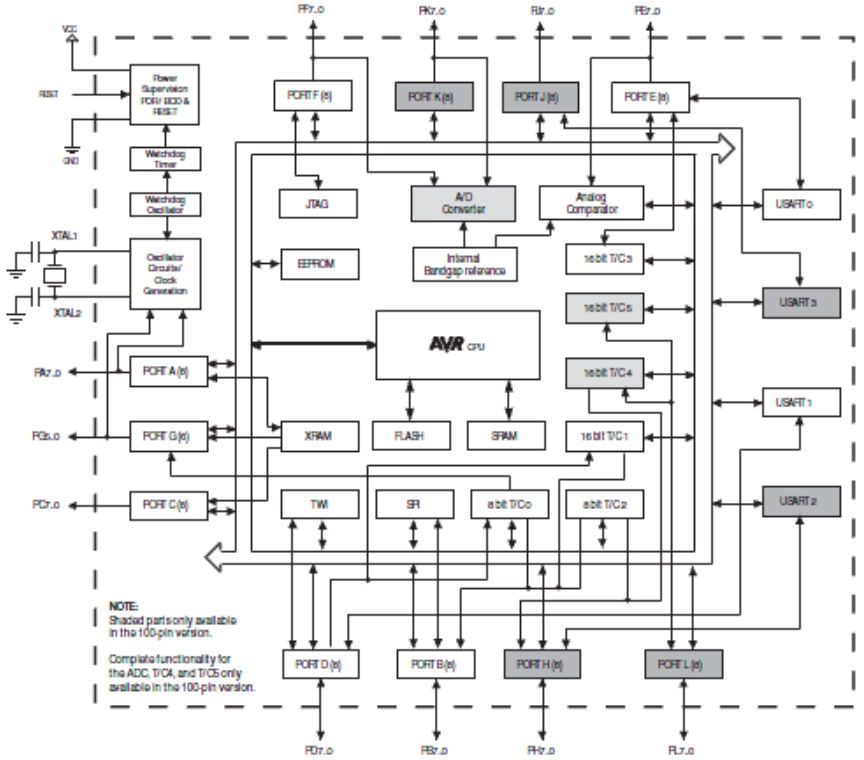
Note: The functions for each pin is the same as for the 100 pin packages shown in Figure 1-1 on page 2.

2. Overview

The ATmega640/1280/1281/2560/2561 is a low-power CMOS 8-bit microcontroller based on the AVR enhanced RISC architecture. By executing powerful instructions in a single clock cycle, the ATmega640/1280/1281/2560/2561 achieves throughputs approaching 1 MIPS per MHz allowing the system designer to optimize power consumption versus processing speed.

2.1 Block Diagram

Figure 2-1. Block Diagram



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

The ATmega640/1280/1281/2560/2561 provides the following features: 64K/128K/256K bytes of In-System Programmable Flash with Read-While-Write capabilities, 4Kbytes EEPROM, 8 Kbytes SRAM, 54/86 general purpose I/O lines, 32 general purpose working registers, Real Time Counter (RTC), six flexible Timer/Counters with compare modes and PWM, 4 USARTs, a byte oriented 2-wire Serial Interface, a 16-channel, 10-bit ADC with optional differential input stage with programmable gain, programmable Watchdog Timer with Internal Oscillator, an SPI serial port, IEEE® std. 1149.1 compliant JTAG test interface, also used for accessing the On-chip Debug system and programming and six software selectable power saving modes. The Idle mode stops the CPU while allowing the SRAM, Timer/Counters, SPI port, and interrupt system to continue functioning. The Power-down mode saves the register contents but freezes the Oscillator, disabling all other chip functions until the next interrupt or Hardware Reset. In Power-save mode, the asynchronous timer continues to run, allowing the user to maintain a timer base while the rest of the device is sleeping. The ADC Noise Reduction mode stops the CPU and all I/O modules except Asynchronous Timer and ADC, to minimize switching noise during ADC conversions. In Standby mode, the Crystal/Resonator Oscillator is running while the rest of the device is sleeping. This allows very fast start-up combined with low power consumption. In Extended Standby mode, both the main Oscillator and the Asynchronous Timer continue to run.

Atmel offers the QTouch® library for embedding capacitive touch buttons, sliders and wheels-functionality into AVR microcontrollers. The patented charge-transfer signal acquisition offersrobust sensing and includes fully debounced reporting of touch keys and includes Adjacent KeySuppression® (AKS™) technology for unambiguous detection of key events. The easy-to-use QTouch Suite toolchain allows you to explore, develop and debug your own touch applications.

The device is manufactured using Atmel's high-density nonvolatile memory technology. The On-chip ISP Flash allows the program memory to be reprogrammed in-system through an SPI serial interface, by a conventional nonvolatile memory programmer, or by an On-chip Boot program running on the AVR core. The boot program can use any interface to download the application program in the application Flash memory. Software in the Boot Flash section will continue to run while the Application Flash section is updated, providing true Read-While-Write operation. By combining an 8-bit RISC CPU with In-System Self-Programmable Flash on a monolithic chip, the Atmel ATmega640/1280/1281/2560/2561 is a powerful microcontroller that provides a highly flexible and cost effective solution to many embedded control applications.

The ATmega640/1280/1281/2560/2561 AVR is supported with a full suite of program and system development tools including: C compilers, macro assemblers, program debugger/simulators, in-circuit emulators, and evaluation kits.

9. Ordering Information

9.1 ATmega640

Speed (MHz)[2]	Power Supply	Ordering Code	Package[1][3]	Operation Range
8	1.8 - 5.5V	ATmega640V-8AU ATmega640V-8AUR[4] ATmega640V-8CU ATmega640V-8CUR[4]	100A 100A 100C1 100C1	Industrial (-40°C to 85°C)
16	2.7 - 5.5V	ATmega640-16AU ATmega640-16AUR[4] ATmega640-16CU ATmega640-16CUR[4]	100A 100A 100C1 100C1	

Notes:
1. This device can also be supplied in wafer form. Please contact your local Atmel sales office for detailed ordering information and minimum quantities.
2. See "Speed Grades" on page 369.
3. Pb-free packaging, complies to the European Directive for Restriction of Hazardous Substances (RoHS directive). Also Halide free and fully Green.
4. Tape & Reel

9.2 ATmega1280

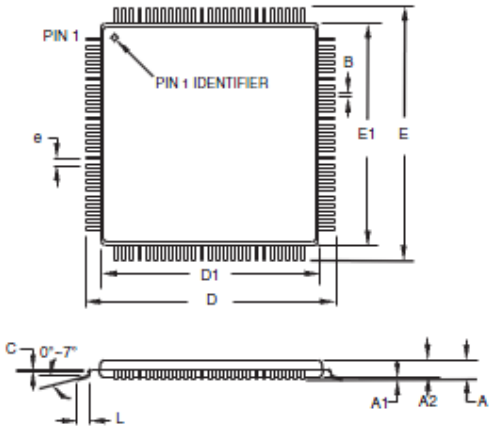
Speed (MHz)[2]	Power Supply	Ordering Code	Package[1][3]	Operation Range
8	1.8V - 5.5V	ATmega1280V-8AU ATmega1280V-8AUR[4] ATmega1280V-8CU ATmega1280V-8CUR[4]	100A 100A 100C1 100C1	Industrial (-40°C to 85°C)
16	2.7V - 5.5V	ATmega1280-16AU ATmega1280-16AUR[4] ATmega1280-16CU ATmega1280-16CUR[4]	100A 100A 100C1 100C1	

Notes:
1. This device can also be supplied in wafer form. Please contact your local Atmel sales office for detailed ordering information and minimum quantities.
2. See "Speed Grades" on page 369.
3. Pb-free packaging, complies to the European Directive for Restriction of Hazardous Substances (RoHS directive). Also Halide free and fully Green.
4. Tape & Reel

Package Type	
100A	100-lead, Thin (1.0mm) Plastic Gull Wing Quad Flat Package (TQFP)
100C1	100-ball, Chip Ball Grid Array (CBGA)

10. Packaging Information

10.1 100A



COMMON DIMENSIONS
(Unit of Measure = mm)

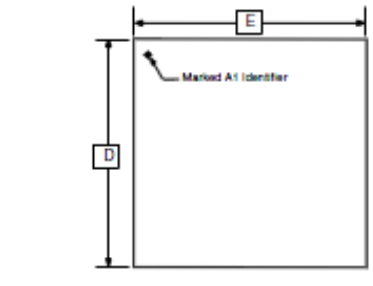
SYMBOL	MIN	NOM	MAX	NOTE
A	–	–	1.20	
A1	0.05	–	0.15	
A2	0.95	1.00	1.05	
D	15.75	16.00	16.25	
D1	13.90	14.00	14.10	Note 2
E	15.75	16.00	16.25	
E1	13.90	14.00	14.10	Note 2
B	0.17	–	0.27	
C	0.09	–	0.20	
L	0.45	–	0.75	
e	0.50 TYP			

Notes:
1. This package conforms to JEDEC reference MS-026, Variation AED.
2. Dimensions D1 and E1 do not include mold protrusion. Allowable protrusion is 0.25 mm per side. Dimensions D1 and E1 are maximum plastic body size dimensions including mold mismatch.
3. Lead coplanarity is 0.08 mm maximum.

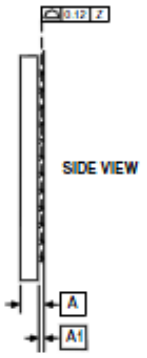
2010-10-20

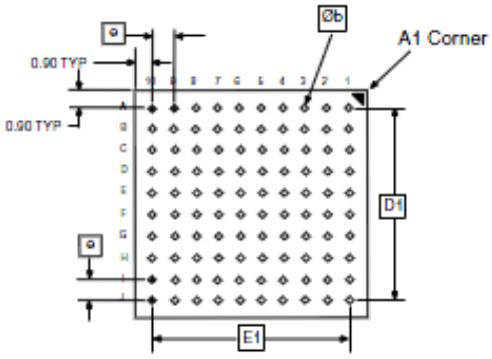
	TITLE	DRAWING NO.	REV.
2325 Orchard Parkway San Jose, CA 95131	**100A**, 100-lead, 14 x 14 mm Body Size, 1.0 mm Body Thickness, 0.5 mm Lead Pitch, Thin Profile Plastic Quad Flat Package (TQFP)	100A	D

10.2 100C1



TOP VIEW





BOTTOM VIEW

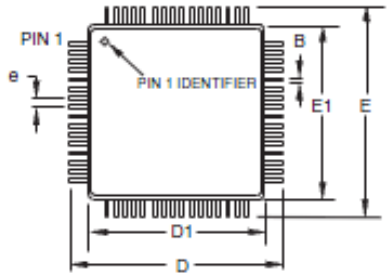
COMMON DIMENSIONS
(Unit of Measure = mm)

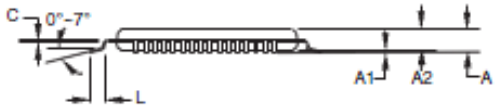
SYMBOL	MIN	NOM	MAX	NOTE
A	1.10	–	1.20	
A1	0.30	0.35	0.40	
D	8.90	9.00	9.10	
E	8.90	9.00	9.10	
D1	7.10	7.20	7.30	
E1	7.10	7.20	7.30	
Øb	0.35	0.40	0.45	
e	0.80 TYP			

5/25/06

	TITLE	DRAWING NO.	REV.
2325 Orchard Parkway San Jose, CA 95131	**100C1**, 100-ball, 9 x 9 x 1.2 mm Body, Ball Pitch 0.80 mm Chip Array BGA Package (CBGA)	100C1	A

10.3 64A





COMMON DIMENSIONS
(Unit of Measure = mm)

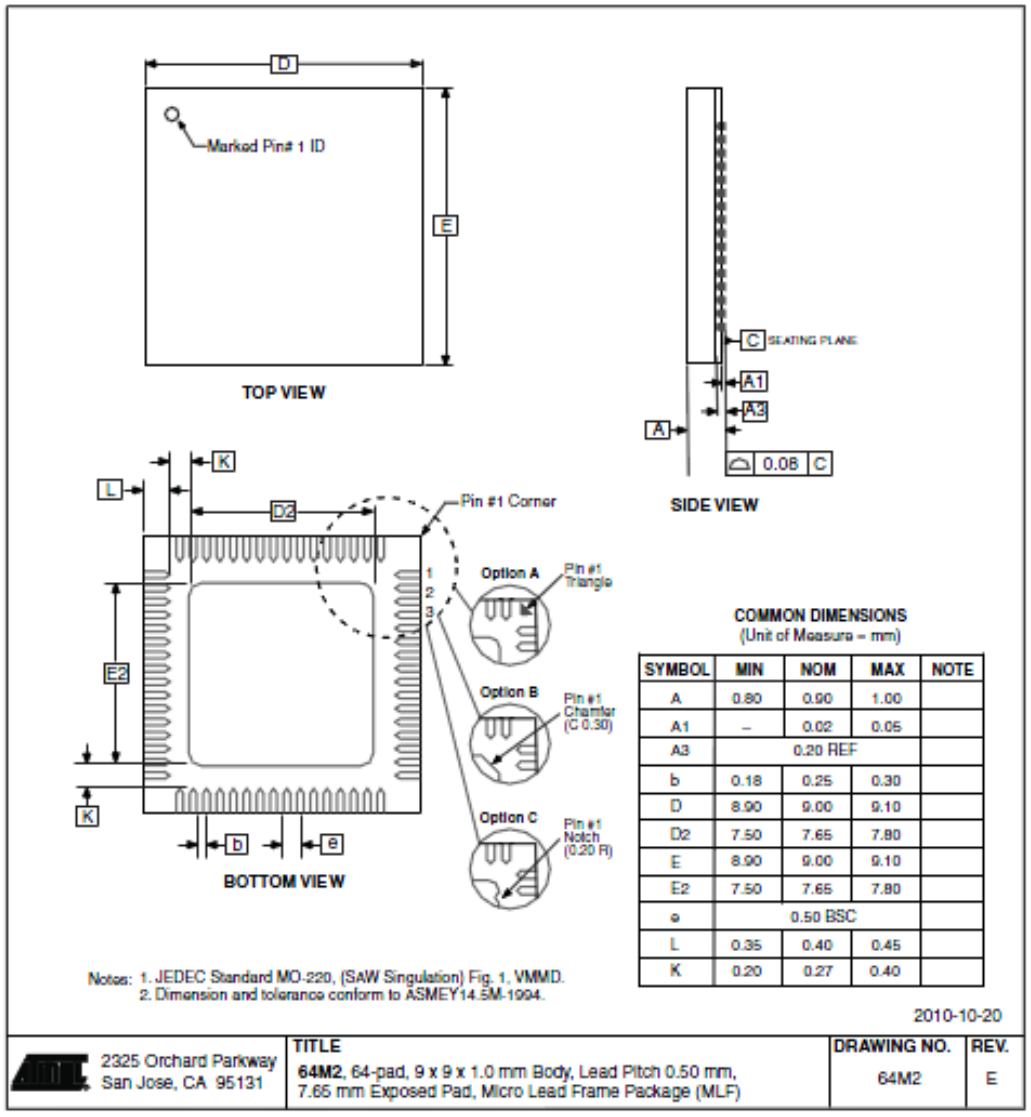
SYMBOL	MIN	NOM	MAX	NOTE
A	–	–	1.20	
A1	0.05	–	0.15	
A2	0.95	1.00	1.05	
D	15.75	16.00	16.25	
D1	13.90	14.00	14.10	Note 2
E	15.75	16.00	16.25	
E1	13.90	14.00	14.10	Note 2
B	0.30	–	0.45	
C	0.09	–	0.20	
L	0.45	–	0.75	
e	0.80 TYP			

Notes:
1. This package conforms to JEDEC reference MS-026, Variation AEB.
2. Dimensions D1 and E1 do not include mold protrusion. Allowable protrusion is 0.25 mm per side. Dimensions D1 and E1 are maximum plastic body size dimensions including mold mismatch.
3. Lead coplanarity is 0.10 mm maximum.

2010-10-20

	TITLE	DRAWING NO.	REV.
2325 Orchard Parkway San Jose, CA 95131	**64A**, 64-lead, 14 x 14 mm Body Size, 1.0 mm Body Thickness, 0.8 mm Lead Pitch, Thin Profile Plastic Quad Flat Package (TQFP)	64A	C

10.4 64M2



COMMON DIMENSIONS
(Unit of Measure – mm)

SYMBOL	MIN	NOM	MAX	NOTE
A	0.80	0.90	1.00	
A1	–	0.02	0.05	
A3		0.20 REF		
b	0.18	0.25	0.30	
D	8.90	9.00	9.10	
D2	7.50	7.65	7.80	
E	8.90	9.00	9.10	
E2	7.50	7.65	7.80	
e		0.50 BSC		
L	0.35	0.40	0.45	
K	0.20	0.27	0.40	

Notes: 1. JEDEC Standard MO-220, (SAW Singulation) Fig. 1, VMMD.
2. Dimension and tolerance conform to ASMEY14.5M-1994.

2010-10-20

	TITLE	DRAWING NO.	REV.
2325 Orchard Parkway San Jose, CA 95131	**64M2**, 64-pad, 9 x 9 x 1.0 mm Body, Lead Pitch 0.50 mm, 7.65 mm Exposed Pad, Micro Lead Frame Package (MLF)	64M2	E

APPENDIX C

CODE PROGRAM

```
//AUTOMATION OF 2 WHEELS ROBOTS//

//PROGRAM TO ENABLE 8 DIRECTION OF 2 WHEELS MOTOR ROBOTS WHEN INTERFACING WITH JOYSTICK//

//NAME:  NORFADILAH BT CHE YUSOFF//

//ID:EA08038   CODE PROJECT: EE044//

//SUPERVISOR: EN MOHD FALFAZLI B. MAT JUSOF//

//UNIVERSITI MALAYSIA PAHANG//

//SEPT 2011-JUNE 2012//


int fwR=31;      //set pin output for motor left move forward

int revR=33;     //set pin output for motor left move reverse

int fwL=37;      //set pin output for motor right move forward

int revL=35;     //set pin output for motor left move reverse

int pwmR=3;      //set pin output for pwm motor at right side

int pwmL=2;      //set pin output for pwm motor at left side

long x, y;       //x and y axes of joystick

int x_in, y_in;  //analog value at x and y axes while controlling joystick


void setup()

{

Serial.begin(9600);      // set up Serial library at 9600 bps

pinMode(fwR,OUTPUT);

pinMode(revR,OUTPUT);

pinMode(fwL,OUTPUT);

pinMode(revL,OUTPUT);

}
```

```
void forward()    //robot move forward direction
{
digitalWrite(fwL,HIGH);
digitalWrite(revR, LOW);
digitalWrite(fwR,HIGH);
digitalWrite(revL,LOW);
analogWrite(pwmL,y);
analogWrite(pwmR,y);
delay(200);
}

void reverse()  //robot move reverse direction
{
digitalWrite(fwL,LOW);
digitalWrite(revR, HIGH);
digitalWrite(fwR,LOW);
digitalWrite(revL,HIGH);
analogWrite(pwmL,(0-y));
analogWrite(pwmR,(0-y));
delay(100);
}
```

```
void turnR()   //robot turn to right direction
{
digitalWrite(fwL,HIGH);
digitalWrite(revR,HIGH);
digitalWrite(fwR,LOW);
digitalWrite(revL,LOW);
analogWrite(pwmL,(x-100));
analogWrite(pwmR,(x-100));
delay(500);
}

void turnL()  //robot turn to left direction
{
digitalWrite(fwL,LOW);
digitalWrite(revR,LOW);
digitalWrite(fwR,HIGH);
digitalWrite(revL,HIGH);
analogWrite(pwmL,(0-x-100));
analogWrite(pwmR,(0-x-100));
delay(500);
}
void forwardR()  //robot move forward and turn right direction
{
digitalWrite(fwL,HIGH);
digitalWrite(revR,LOW);
```

```
digitalWrite(fwR,LOW);
digitalWrite(revL,LOW);
analogWrite(pwmL,y);
analogWrite(pwmR,y);
delay(100);
}

void forwardL()  //robot move forward and turn left direction
{
digitalWrite(fwL,LOW);
digitalWrite(revR,LOW);
digitalWrite(fwR,HIGH);
digitalWrite(revL,LOW);
analogWrite(pwmL,y);
analogWrite(pwmR,y);
delay(100);
}

void reverseR() //robot move forward and turn right direction
{
digitalWrite(fwL,LOW);
digitalWrite(revR,LOW);
digitalWrite(fwR,LOW);
digitalWrite(revL,HIGH);
analogWrite(pwmL,(0-y));
```

```
analogWrite(pwmR,(0-y));
delay(100);
}

void reverseL() //robot move reverse and turn right direction
{
digitalWrite(fwL,LOW);
digitalWrite(revR, HIGH);
digitalWrite(fwR,LOW);
digitalWrite(revL,LOW);
analogWrite(pwmL,(0-y));
analogWrite(pwmR,(0-y));
delay(100);
}

void Stop()   //robot stop
{
digitalWrite(fwL,LOW);
digitalWrite(revR,LOW);
digitalWrite(fwR,LOW);
digitalWrite(revL,LOW);
analogWrite(pwmL,(0));
analogWrite(pwmR,(0));
//delay(100);
}
```

```
void loop()

{

x_in = analogRead(1);

y_in = analogRead(0);


x = map(x_in, 508, 1023, 0, 255);  // mapped bits resolutions from 10 bits to 8 bits in analog value

y = map(y_in, 503, 1023, 0, 255);


if(x==0 && y>0)    //analog value at joystick at x and y axes

{

forward();

}


if(x==0 && y<0)    //analog value at joystick at x and y axes

{

reverse();

}


if(y==0 && x<0)    //analog value at joystick at x and y axes

{

turnR();

}
```

```
if(y==0 && x>0)    //analog value at joystick at x and y axes
{
turnL();
}

if(y>0 && x<0)     //analog value at joystick at x and y axes
{
forwardR();
}

if(x>0 && y>0)     //analog value at joystick at x and y axes
{
forwardL();
}

if(x<0 && y<0)     //analog value at joystick at x and y axes
{
reverseR();
}

if(x>0 && y<0)     //analog value at joystick at x and y axes
{
reverseL();
}
```

```
else
{
Stop();
}

Serial.print("X: ");
Serial.print(x, DEC);
Serial.print("   Y: ");
Serial.print(y, DEC);
Serial.print(", ");
Serial.print(fwR,DEC);
Serial.print(", ");
Serial.print(", ");
Serial.print(revR,DEC);
Serial.print(", ");
Serial.print(revL,DEC);
Serial.print(", ");
Serial.print(pwmR,DEC);
Serial.print(", ");
Serial.print(x_in,DEC);
Serial.print(", ");
Serial.println(y_in,DEC);

//END PROGRAM//
}
```

APPENDIX D

Schematic diagram

Pictures

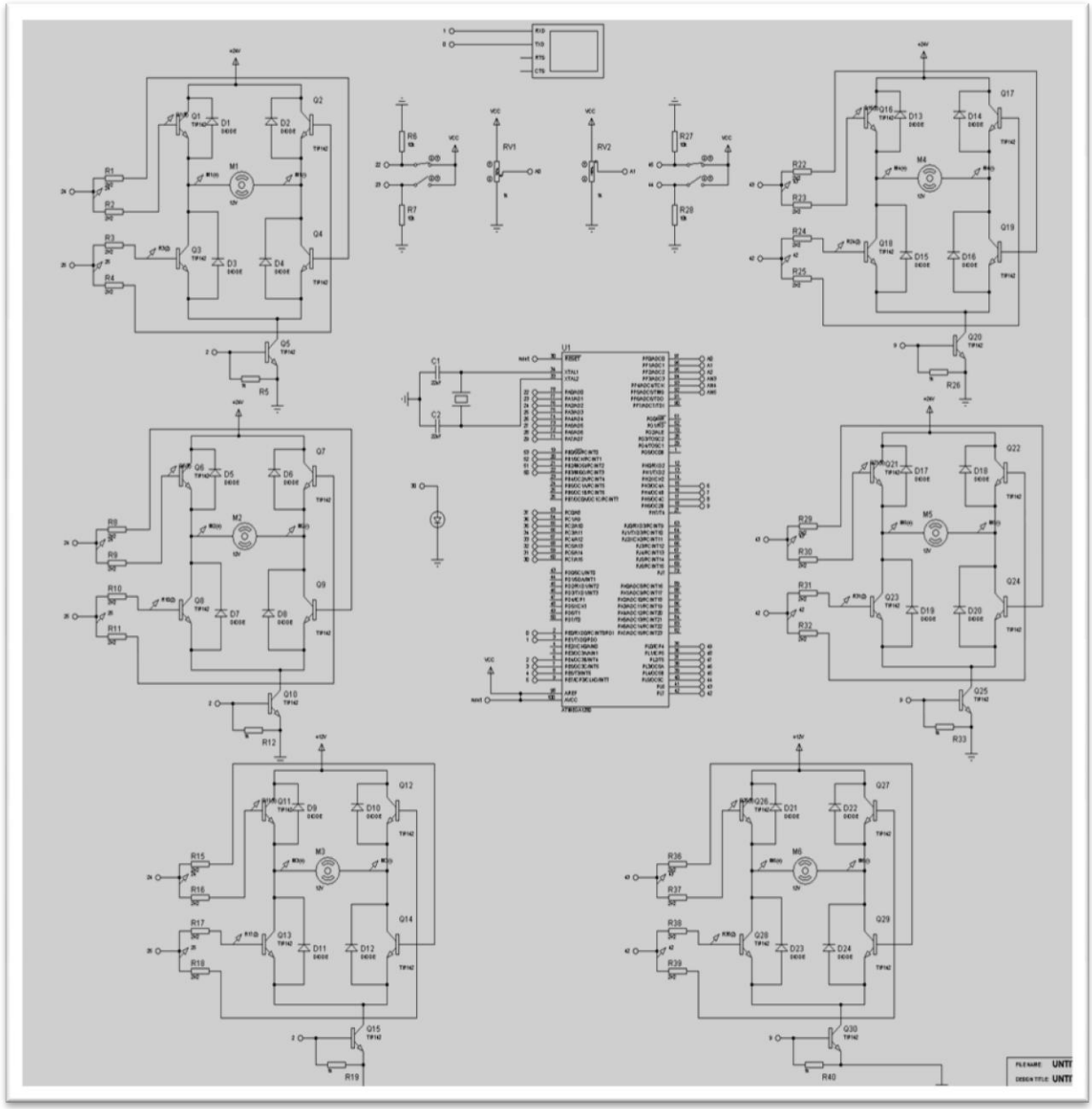
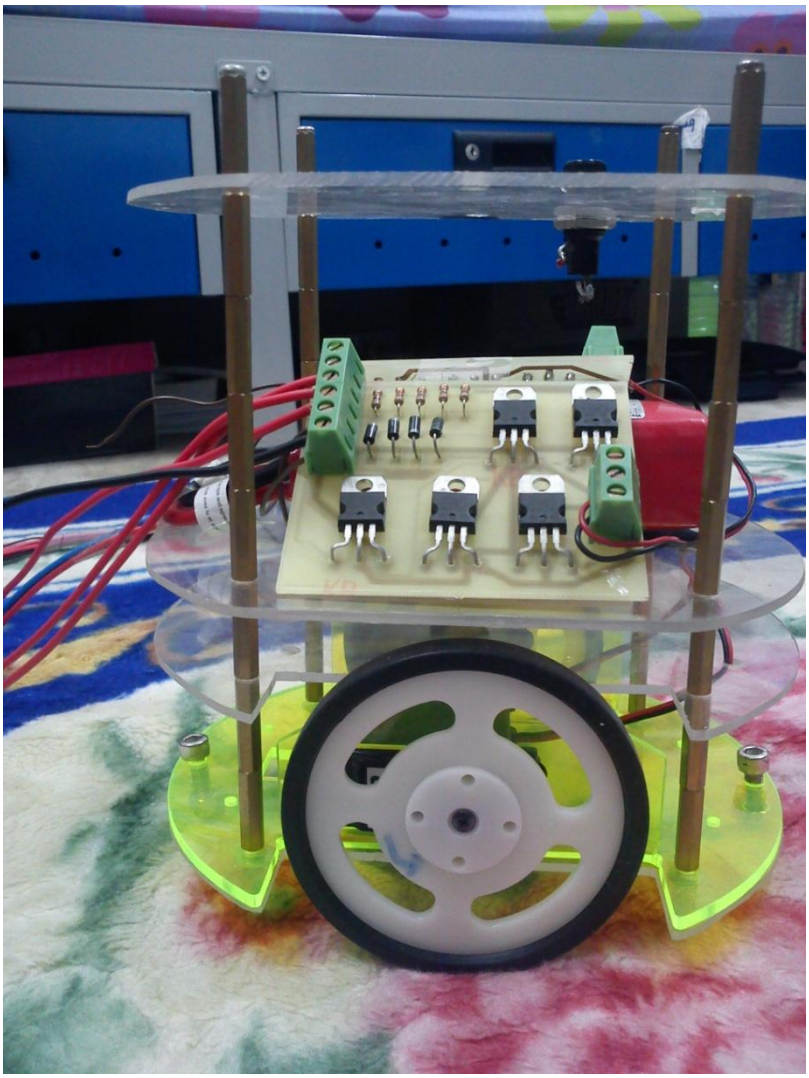
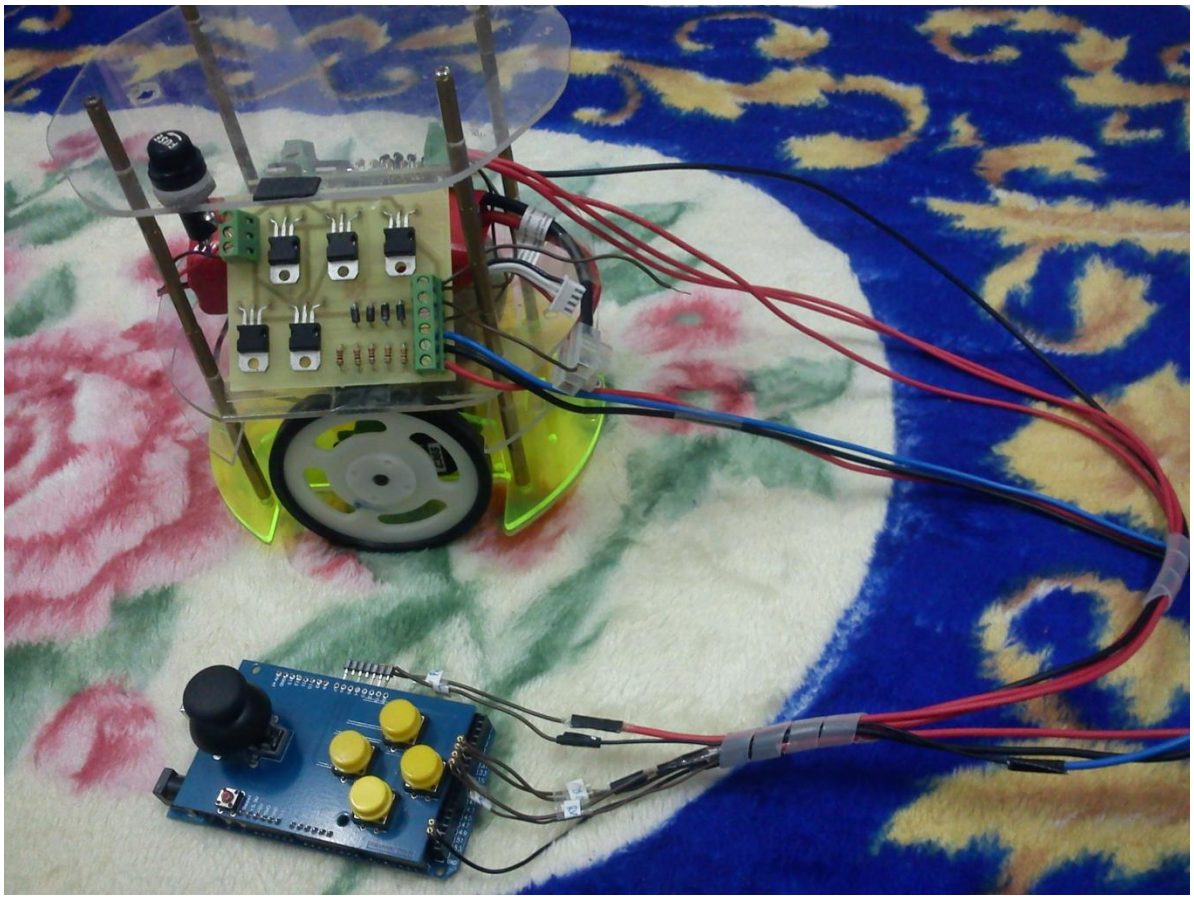


Figure 3.10: Arduino 2560 connected to 6 DC Motor



2 wheels robot from side view



Two wheels robot with joystick