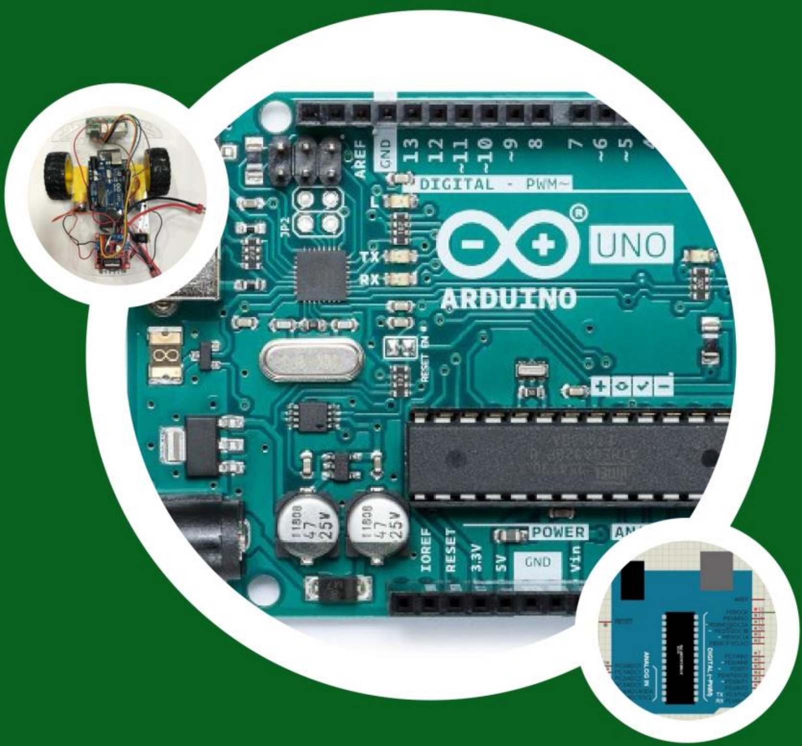
POLITEKNIK MALAYSIA
TUN SYED NASIR

USER MANUAL

DGI40122 TEACHING KIT MODULE

EMBEDDED ROBOTIC



TUAN ROZILAAZAWANI BINTI TUAN MAT
HASYIREEN BINTI ABDUL HALIM
ROZIEANA BINTI ABU

USER MANUAL

DGI40122 TEACHING KIT MODULE

EMBEDDED ROBOTIC

TUAN ROZILAAZAWANI BINTI TUAN MAT
HASYIREEN BINTI ABDUL HALIM
ROZIEANA BINTI ABU

First Publication: October 2022

This ebook is the original work of
Tuan Rozilaazawani binti Tuan Mat
Hasyireen binti Abdul Halim
Rozieana binti Abu

Published by:
Department of Petrochemical Engineering,
Politeknik Tun Syed Nasir Syed Ismail,
Hab Pendidikan Tinggi Pagoh,
KM1 Jalan Panchor
84600 Panchor, Muar, Johor, Malaysia

e ISBN 978-967-2736-13-4

9 7 8 9 6 7 2 7 3 6 1 3 4

PREFACE

Alhamdulillah, in the name of the Almighty Allah S.W.T, the Most Gracious and the Most Merciful who gives us the enlightenment, the truth, the knowledge and with regards to Prophet Muhammad S.A.W. for guiding us to the straight path. We thank Allah S.W.T. for giving us the strength to write this book. May Allah give us the ability to continue our good deeds to the community in this field.

This book is based on the latest syllabus prepared by the Department of Polytechnic and College Community, Ministry of Higher Education. It is structured in three parts; dealing with embedded systems, sensors and actuators, and mobile robot design. These parts are essential to students in mastering the crucial steps of building a complete working robotic system.

This book consists of six practical works; Introduction to Arduino Editor and Proteus Software, Input and Output Programming in C, Introduction to Robotic and Microcontroller, Robotic Controller Programming in C, Sensors and Actuators Programming in C and Mobile Robot Design. Each practical work discusses the concept theory followed by hands-on skill to demonstrate basic concepts. All concepts and skills presented for each practical work are accompanied by practical and theory related question.

Upon completion of these laboratory practices and exercises, you will be able to determine the concept of robot positioning, identification and communication in mobile robot control according to standard robot organization regulation; manipulate the application of sensor and actuator, robot identification and communication during practical work based on land mobile robot design; and organize mini competition among themselves to compete using land mobile robot.

We are happy to receive any comments and suggestions to improve the quality of this book. We hope that this book can be beneficial to both educationist and students. InsyaAllah.

AUTHOR

Tuan Rozilaazawani binti Tuan Mat

is a Lecturer in Electrical and Instrumentation Program at the Department of Petrochemical Engineering at Politeknik Tun Syed Nasir Syed Ismail. Her first degree is in Electrical Engineering from Kolej Universiti Teknologi Tun Hussein Onn. She finished her Master's Degree in Technical and Vocational Education in 2004. She had 18 years of experience in teaching Electrical and Electronic Engineering.

Hasyireen binti Abdul Halim

is a Lecturer of Electrical and Instrumentation Program at the Department of Petrochemical Engineering at Politeknik Tun Syed Nasir Syed Ismail. Her first degree is in Telecommunication Engineering from University of Malaya (UM). She finished her Master of Engineering in Industrial Electronic & Control Engineering in 2012. She had 14 years of experience in teaching Electrical and Electronic Engineering.

Rozieana binti Abu

is a Lecturer in Electrical and Instrumentation Program at the Department of Petrochemical Engineering at Politeknik Tun Syed Nasir Syed Ismail. Her first Degree in Technology with Education (Electric and Electronic Engineering) was conferred by Universiti Teknologi Malaysia. She obtained her Master of Science degree in 2016 and is currently in her fourth year of a Ph.D. program in Chemical Engineering. She has taught Electrical and Electronic Engineering for nearly 19 years.

CONTENT

1.0 Introduction

In recent years, hands-on learning methodologies such as problem-based and project-based learning (PBL) have grown in popularity in the engineering classroom. Since its beginnings, PBL has had a substantial impact on students' knowledge of basic principles taught in courses, and it extends this learning by allowing students to apply this material to real-world applications (Chi & City, 2022).

The advancement of computing technology, such as embedded robotics, has introduced new challenges to the landscape of computer engineering education (Rosa et al., 2021). Numerous research has produced an innovative project-based course to meet the requirement for engineering graduates to have a good grasp and skill in embedded system design. It has the potential to be an educational technique that offers learners realistic learning tasks based on their specific interests (Tian, 2021). As far as the literature is concerned, the popularity of robotics and programming are increasing in educational environments (Uzun, 2020).

For instance, in the year 2020, the DGI40122 Embedded Robotic Course was launched with the latest revised syllabus in Malaysian Polytechnic by the Ministry of Higher Education's Department of Polytechnic and Community College. The course teaches students at the basic and intermediate levels on how to assemble mobile robots and embedded systems.

Despite the fact that PBL is commonly used in embedded robotics, it is apparent that there is still a lack of suitable guidance on the intended approach for the demonstration of practical work processes, especially for the Malaysian polytechnics. Furthermore, it was observed that there is a lack of diverse types of practical work series that best provided a more complete understanding of assessment towards the applicability of embedded robotic teaching methods (Jawaid et al., 2020).

The implications will result in practical pedagogy only focused on one means of communication for electrical and instrumentation engineering students without displaying the interconnection between user experience and soft skills development. The effective teaching of practical work series for embedded robotics, on the other hand, necessarily involves interaction with particular technological goal-setting pertaining to programming and hardware. For instance, using the Arduino microcontroller, C programming development for embedded robotics, and project design experience.

Thus, the initiatives for the development of practical work procedures must be taken in a planned manner. The Embedded Robotics DGI140122 was chosen to illustrate this conceptual analysis because it provides a novel, systematic assessment technique for combining these two types of methods. This practical work series is created with the use of an Arduino microcontroller, a land mobile robot, the Arduino Editor, and the Proteus software.

There are three structured combinations: (1) dealing with embedded systems, to evaluate students' cognitive skills such as programming and application of the C language; (2) sensors and actuators, a psychomotor assessment to observe results and gain judgements from embedded systems design; and (3) mobile robot design, a mix of psychomotor and cognitive abilities to observe results and acquire judgments from embedded systems design. Jawaid et al., (2020) supported that a successful course design strategies of PBL in teaching engineering content to the learners was of the utmost importance in the robotic field.

In embedded robotics, the PBL function is predominantly expressed in three main categories: (i) Pedagogical optimization methods; (ii) Gaming competitions that assess technical criteria; and (iii) Course learning outcomes evaluation. It appears that the course learning outcomes assessment followed by pedagogical optimization methods are by far the most favored in the studies reviewed, indicating a more theoretical or conceptual purpose for PBL.

Engineering students can receive good benefits from PBL because it provides hands-on, real-world experience that helps to comprehend technical ideas and concepts better. By assigning tasks that leads to the development of a final product in terms of design, model, device, program coding, and simulation, PBL gives direction and encourages self-directed and active learning (Larson et al., 2020).

2.0 Overall System Overview

You must be familiar with this manual in order to complete the exercises given within. With all these exercises, you will learn about:

i. Introduction to Arduino Editor and Proteus Software
ii. Input and Output Programming in C
iii. Introduction to Robotic and Microcontroller
iv. Robotic Controller Programming in C
v. Sensors and Actuators Programming in C
vi. Mobile Robot Design

Upon the completion of these laboratory work and exercises, you will be able to determine the concept of robot positioning, identification and communication in mobile robot control according to standard robot organization regulation; manipulates the application of sensor and actuator, robot identification and communication during practical work based on land mobile robot design; and organize mini competition among yourselves to compete using land mobile robot.

3.0 Arduino Uno

Arduino Uno is a microcontroller board based on the ATmega328P. It has 14 digital input/output pins (6 of which can be used as PWM outputs), 6 analog inputs, a 16-MHz quartz crystal, a USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. You can tinker with your UNO without worrying too much about doing something wrong. The worst scenario were you needed to replace the chip and start over again.

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

Summary

Microcontroller	ATmega328
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output) : 0-13
Analog Input Pins	6 (A0-A5)
DC Current per I/O Pin	40mA
DC Current for 3.3V Pin	50mA
Flash Memory	32KB (ATmega328) of which 0.5KB used by bootloader
SRAM	2KB (ATmega328)
EEPROM	1KB (ATmega328)
Clock Speed	16MHz

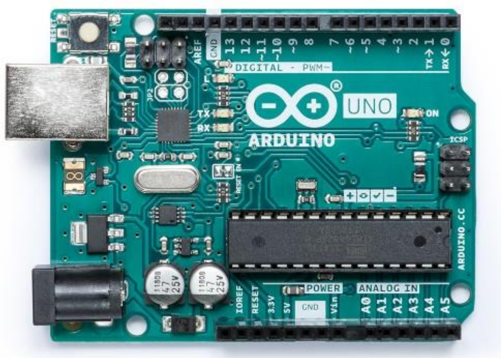


Figure 1: Arduino Uno Board

Figure 1 shows the Arduino Uno Board. For more information please refer to the datasheet. The data sheet can be found at https://www.farnell.com/datasheets/1682209.pdf

4.0 Hardware and Software Preparation

This DGI40122 Embedded Robotic Teaching Kit Module covers the basic concept and application of microcontroller systems based on the Arduino Uno. Users will learn software and hardware development on Arduino development system such as Arduino Software (IDE) and understand how to do interfacing with external devices using suitable internal chip features. Users will also be exposed to the new Microcontroller Unit (MCU) simulation software such as Proteus.

Hardware Setup

The basic hardware in this teaching kit are Arduino Uno, computer/laptop and USB cable. Basic hardware connection to a computer/laptop is as shown in Figure 2.

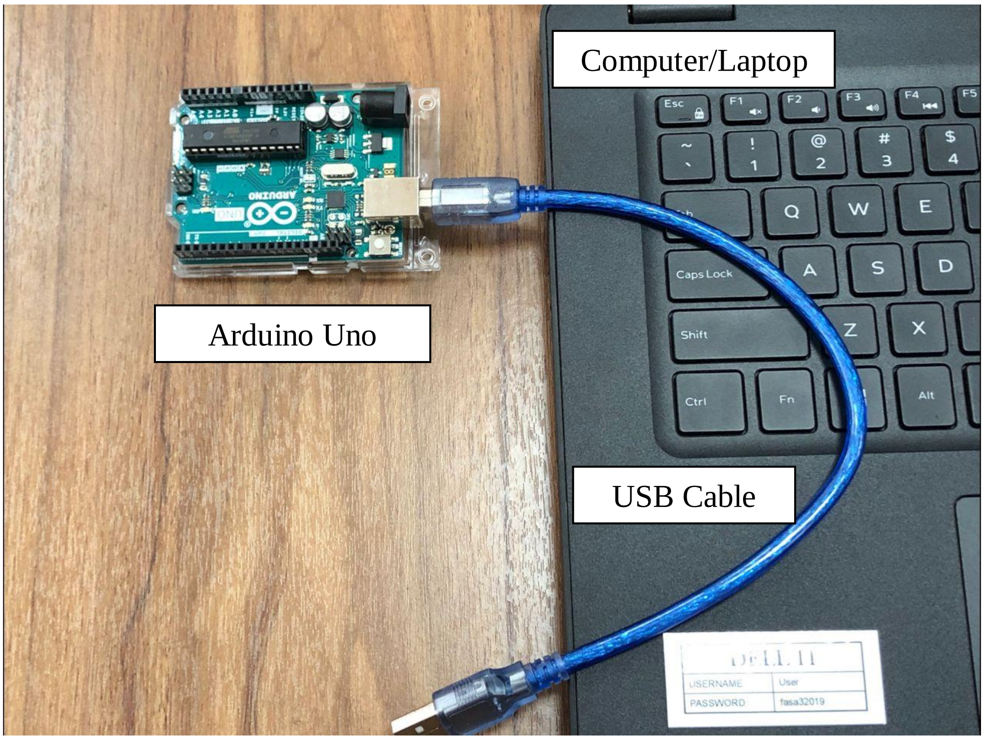


Figure 2: Basic Hardware Connection

Arduino Software (IDE)

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. This software can be used with any Arduino board. Arduino is an open-source physical computing platform based on a simple I/O board and a development environment that implements the Processing/Wiring language. Arduino can be used to develop stand-alone interactive objects or can be connected to software on your computer (e.g. Flash, Processing and MaxMSP). The boards can be assembled manually or purchased preassembled; the open-source IDE can be downloaded for free at https://arduino.cc

Figure 3: Arduino

Active development of the Arduino software is hosted by GitHub. This organisation contains the official Arduino tools (IDE, CLI...), documentation and cores. See @arduino-libraries for the official libraries or https://github.com/arduino

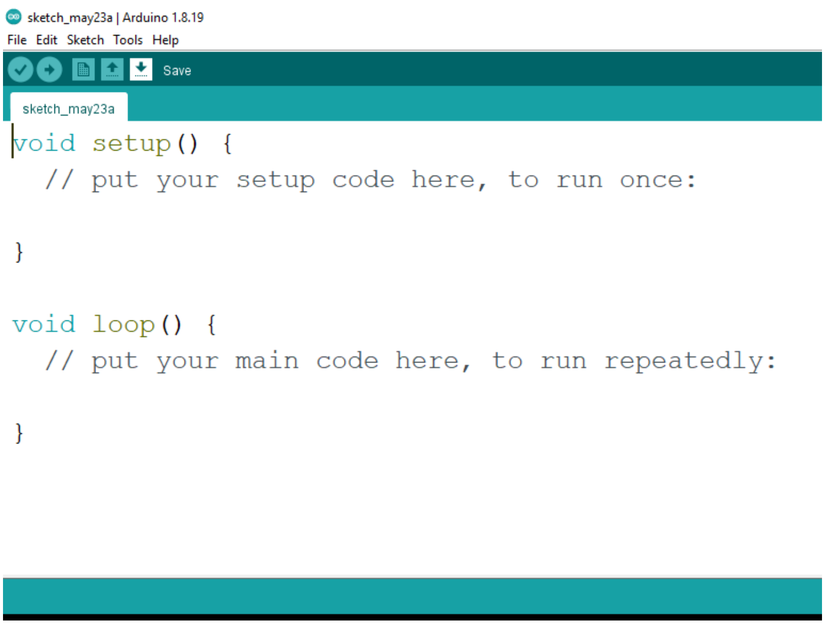


Figure 4: Arduino Editor

Proteus 7/8 Professional

Proteus is simulator software which is capable to simulate any circuit and scenario. Mainly, it is best used for microcontrollers. Proteus professional design combines the ISIS schematic capture and ARES PCB layout programs to provide a powerful, integrated and easy to use tools suite for education and professional PCB Design.

The Proteus Professional is a software for automated design of electronic circuits. The package is a system of circuit simulation, based on the models of electronic components in SPICE. The distinctive feature of the package Proteus Professional package is the possibility of modelling the programmable devices: microcontrollers, microprocessors, DSP and others.

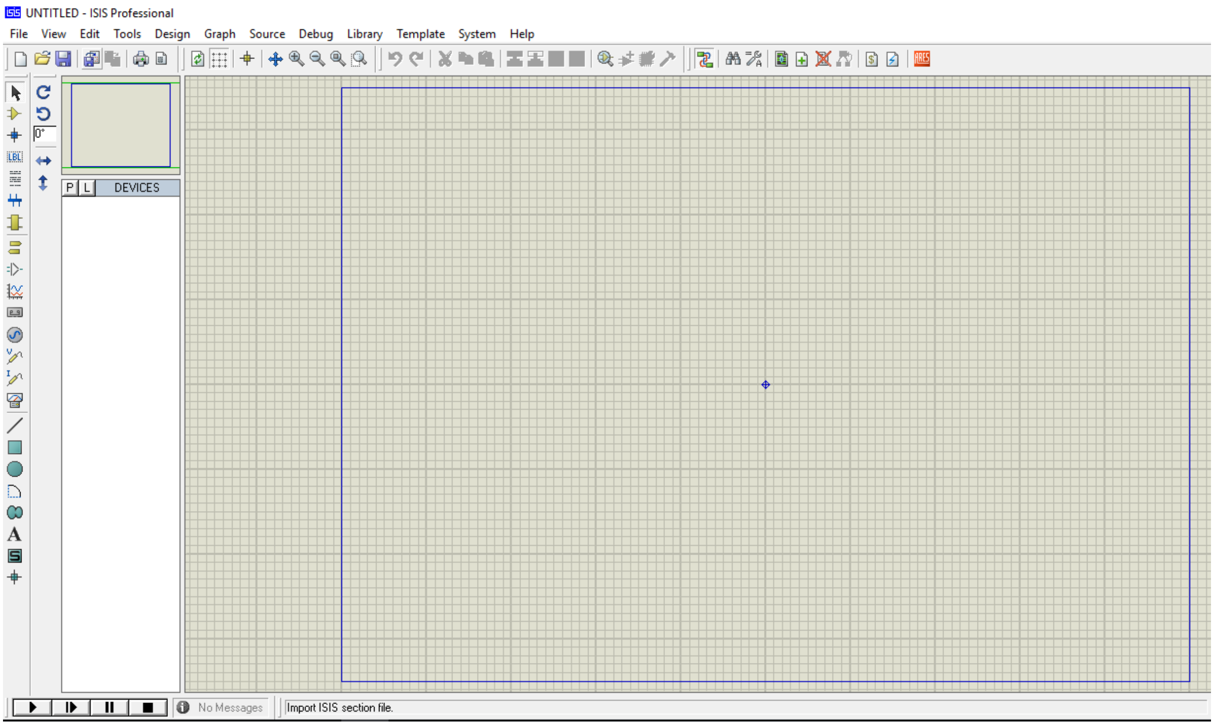


Figure 5: Proteus Workspace

5.0 Hardware Specification

Table 1 shows the hardware specifications for DGI40122 Teaching Kit.

Table 1: DGI40122 Teaching Kit Hardware Specification

No	Components	Quantity
1	Arduino UNO R3	1
2	2WD Smart Robot Car Chassis	1
3	Ultrasonic Sensor HC-SR04	1
4	Auto-calibration Line Sensor	1
5	Bluetooth Module HC-05	1
6	Dual H-Bridge Driver L293D	1
7	LiPO Rechargeable Battery 11.1V 2200mAh	1
8	Multi-Function LiPO Balance Charger	1
9	Arduino Editor	1
10	Proteus Software	1
11	Computer/Laptop	1

6.0 DGI40122 Teaching Kit

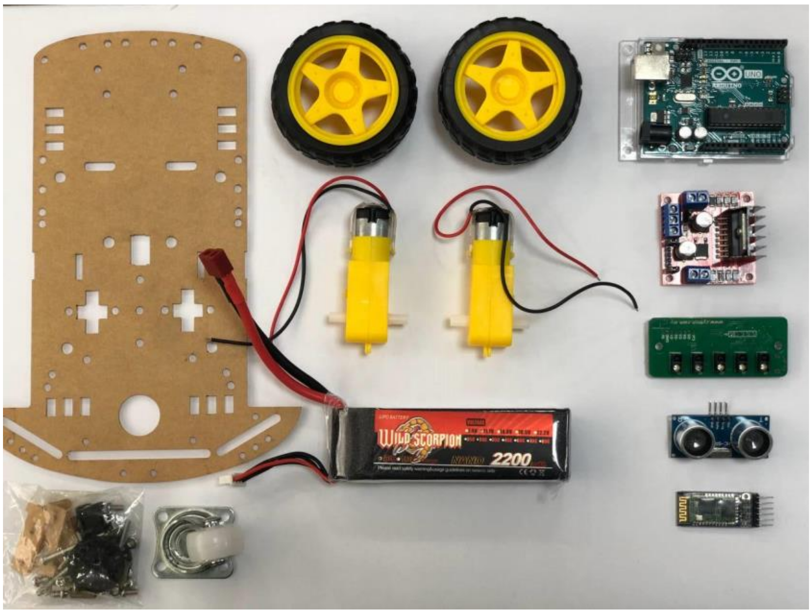


Figure 6: Hardware Specification

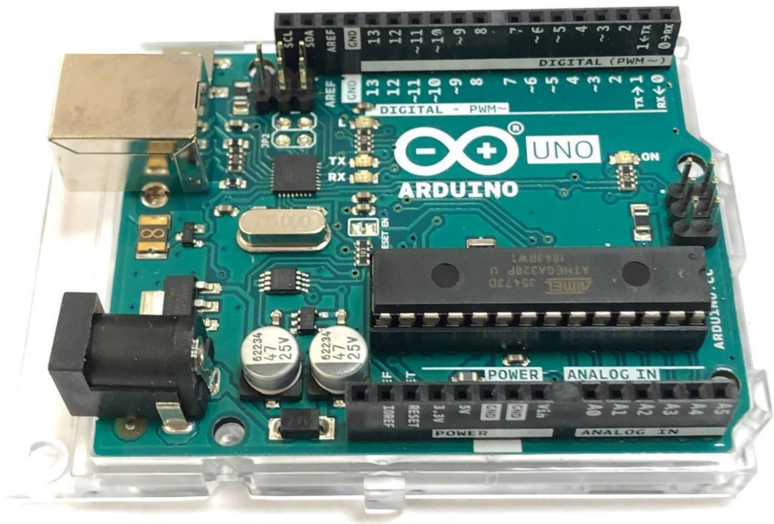
7.0 Teaching Kit Operational Procedure

i. Prepare all equipments and materials to be used such as Arduino UNO R3, input-output (I/O) components, robot mobile and USB cable.
ii. Plug one end of the USB cable into the Arduino board USB. Plug the other end into a USB port on your PC.
iii. The lights indicator on the Arduino board will turn on. It shows that the equipment is in good condition.
iv. Open Arduino Editor workspace window. Select menu Tools – Board: “Arduino Uno” – Select Arduino Uno.
v. Then, select menu File – Preferences – menu Settings – checked box “Show verbose output during: √ compilation – then OK.
vi. Follow the next procedure to complete the practical work.

PRACTICAL WORK

[CONTENT OF PRACTICAL WORK]

1. PW1 : INTRODUCTION TO ARDUINO EDITOR AND PROTEUS SOFTWARE
2. PW2 : INPUT-OUTPUT PROGRAMMING IN C
3. PW3 : INTRODUCTION TO ROBOTIC AND MICROCONTROLLER
4. PW4 : ROBOTIC CONTROLLER PROGRAMMING IN C
5. PW5 : SENSORS AND ACTUATORS PROGRAMMING IN C
6. PW6 : MOBILE ROBOT DESIGN



PRACTICAL WORK 1

INTRODUCTION TO ARDUINO EDITOR AND PROTEUS SOFTWARE

LEARNING OBJECTIVES:

Students will be able to;

a) perform an installation of Proteus software and Arduino Editor.
b) write a C program that will run on the Arduino Uno.
c) assembles a C program to define input and output programming.
d) make an observation on the output of C program using Proteus.

EQUIPMENTS:

1. Arduino IDE Installer
2. Proteus Installer
3. Computer

THEORY:

ARDUINO UNO

Arduino Uno is a microcontroller board based on the ATmega328P (datasheet). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.. You can tinker with your UNO without worring too much about doing something wrong, worst case scenario you can replace the chip for a few dollars and start over again.

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

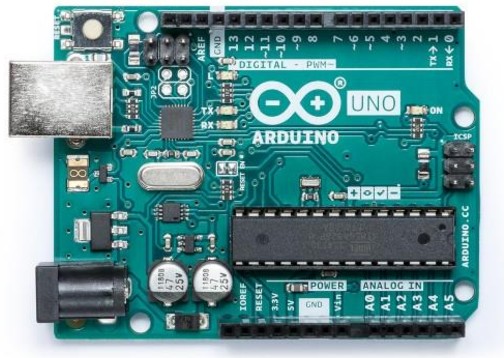


Figure 1 : Arduino Uno Board

PROTEUS

Proteus 7/8 Professional is a software which can be used to draw schematics, PCB layout, code and even simulate the schematic. It is developed by Labcenter Electronic Ltd.

It is a software suite containing schematic, simulation as well as PCB designing. ISIS is the software used to draw schematics and simulate the circuits in real time. The simulation allows human access during run time, thus providing real time simulation.

ISIS has wide range of components in its library. It has sources, signal generators, measurement and analysis tools like oscilloscope, voltmeter, ammeter etc., probes for real time monitoring of the parameters of the circuit, switches, displays, loads like motors and lamps, discrete components like resistors, capacitors, inductors, transformers, digital and analog Integrated circuits, semi-conductor switches, relays, microcontrollers, processors, sensors etc.

PROCEDURE A

A. Arduino Software Installation

1. Double-click to **arduino-1.8.3-windows** file in your Installer Folder.
2. Finish the installation.
3. Create an Arduino desktop shortcut.

B. Proteus Software Installation

1. Follow all the instructions in **Proteus Installation Note** file in your Installer Folder.
2. Finish the installation.
3. Create a Proteus desktop shortcut.

C. How to Add Arduino Board to Proteus 7

1. Copy Arduino Library files.
2. Open file location of Proteus 7.
3. Browse Labcenter Electronics folder in Program Files (x86).
4. Open LIBRARY folder.
5. Paste Arduino Library into LIBRARY folder.
6. Arduino Library allocate in the LIBRARY Proteus 7.

D. How to Add Arduino Board to Proteus 8

1. Copy Arduino Library files.
2. Open Folder Options and on radio button for “Show hidden files, folders, and drivers”. Then select Apply.
3. Browse Labcenter Electronics folder in Program Files.
4. Open LIBRARY folder.
5. Paste Arduino Library into LIBRARY folder.
6. Arduino Library allocate in the LIBRARY Proteus 8.

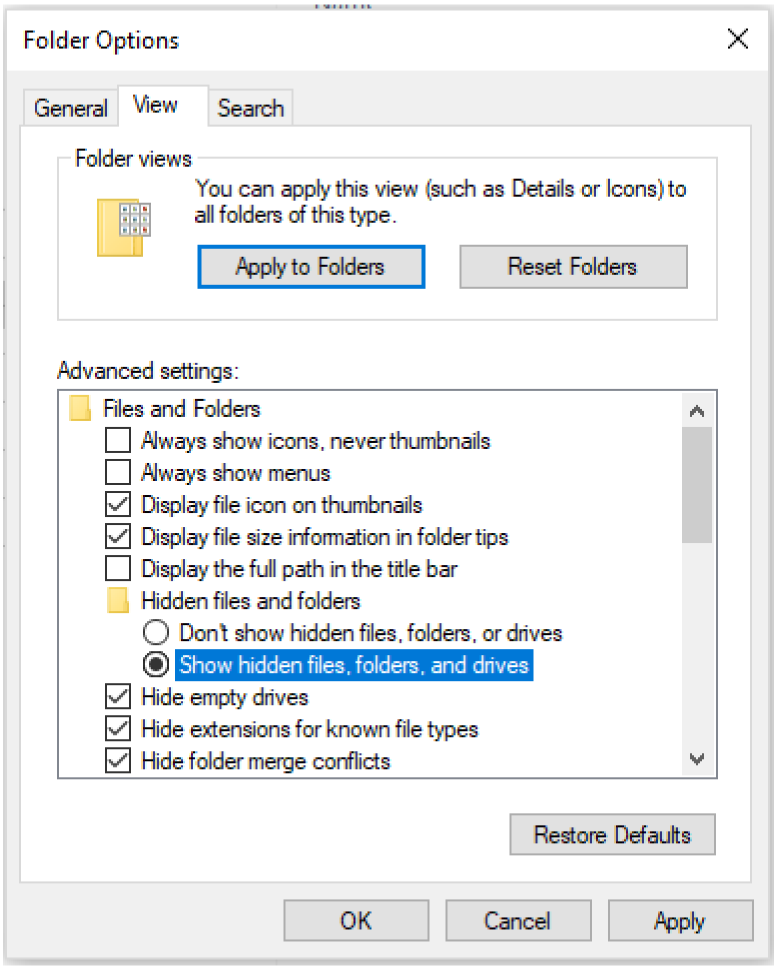


Figure 2 : Folder Options for Show Hidden Files

QUESTIONS

1. Explain the functions of Proteus software.

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

2. What is Arduino Editor?

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

3. What is C programming language?

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

PRACTICAL WORK 2

INPUT AND OUTPUT (I/O) PROGRAMMING IN C

LEARNING OBJECTIVES:

Students will be able to;

a) write a C program for the Delay subroutine and language subroutines.
b) assembles a C program to define analog and digital input output.
c) sketch a schematic circuit of LED Blink, Seven Segment and LCD Display using Proteus.
d) make an observation on the output of C program using Proteus.

EQUIPMENTS:

1. Arduino Editor
2. Proteus Software
3. Computer

THEORY:

ARDUINO EDITOR

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software.

This software can be used with any Arduino board. The **Arduino** Web **Editor** allows you to write code, save it to the cloud storage and upload sketches to any **Arduino** board and Intel®-based platforms from your web browser after installing a simple plug-in.

Figure 1 : Arduino Editor

PROTEUS

Proteus 7 Professional is a software which can be used to draw schematics, PCB layout, code and even simulate the schematic. It is developed by Labcenter Electronic Ltd.

It is a software suite containing schematic, simulation as well as PCB designing. ISIS is the software used to draw schematics and simulate the circuits in real time. The simulation allows human access during run time, thus providing real time simulation.

ISIS has wide range of components in its library. It has sources, signal generators, measurement and analysis tools like oscilloscope, voltmeter, ammeter etc., probes for real time monitoring of the parameters of the circuit, switches, displays, loads like motors and lamps, discrete components like resistors, capacitors, inductors, transformers, digital and analog Integrated circuits, semi-conductor switches, relays, microcontrollers, processors, sensors etc.

SEVEN-SEGMENT DISPLAY

A seven-segment display (SSD), or seven-segment indicator, is a form of electronic display device for displaying decimal numerals that is an alternative to the more complex dot-matrix displays. Seven-segment displays are widely used in digital clocks, electronic meters, and other electronic devices for displaying numerical information.

The seven elements of the display can be lit in different combinations to represent the arabic numerals. Often the seven segments are arranged in an*oblique* (slanted) arrangement, which aids readability. In most applications, the seven segments are of nearly uniform shape and size (usually elongated hexagons, though trapezoids and rectangles can also be used), though in the case of adding machines, the vertical segments are longer and more oddly shaped at the ends in an effort to further enhance readability. The numerals 6, 7 and 9 may be represented by two or more different glyphs on seven-segment displays. The seven segments are arranged as a rectangle of two vertical segments on each side with one horizontal segment on the top, middle, and bottom. Additionally, the seventh segment bisects the rectangle horizontally. There are also fourteen-segment displays and sixteen-segment displays (for full alphanumerics); however, these have mostly been replaced by dot-matrix displays. The segments of a 7-segment display are referred to by the letters A to G, where the optional DP decimal point (an "eighth segment") is used for the display of non-integer numbers.

In a simple LED package, typically all of the cathodes (negative terminals) or all of the anodes (positive terminals) of the segment LEDs are connected and brought out to a common pin; this is referred to as a "*common cathode*" or "*common anode*" device. Hence a 7 segment plus decimal point package will only require nine pins (though commercial products typically contain more pins, and/or spaces where pins would go, in order to match standard IC sockets. For example, all the anodes of the A segments of each digit position would be connected together and to a driver pin, while the cathodes of all segments for each digit would be connected. To operate any particular segment of any digit, the controlling integrated circuit would turn on the cathode driver for the selected digit, and the anode drivers for the desired segments; then after a short blanking interval the next digit would be selected and new segments lit, in a sequential fashion. In this manner an eight digit display with seven segments and a decimal point would require only 8 cathode drivers and 8 anode drivers, instead of sixty-four drivers and IC pins. A single byte can encode the full state of a 7-segment-display. The most popular bit encodings are *gfedcba* and *abcdefg*, where each letter represents a particular segment in the display. In the *gfedcba* representation, a byte value of 0x06 would (in a common-anode circuit) turn on segments 'c' and 'b', which would display a '1'.

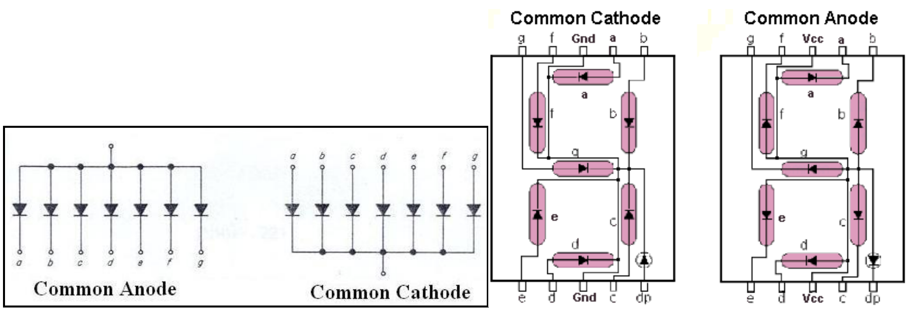
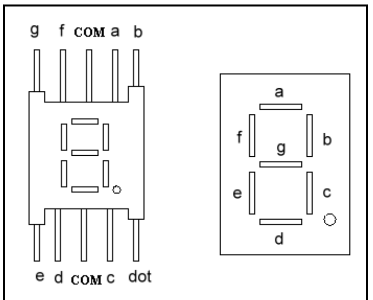


Figure 2 : Interfacing LED's to Microcontroller

Basically there are two types of Seven-Segment displays:

1. Common Cathode where all the segments share the same Cathode.
2. Common Anode where all the segments share the same Anode.

Common Anode is an order to turn ON a segment where the corresponding pin must be set to 0. And to turn it OFF if set to 1. Whereas Common Cathode is an order to turn ON a segment where the corresponding pin must be set to 1. And to turn it OFF if set to 0. Figure 2 and 3 shown truth table for Seven Segment decoder outputs.



Seven-Segment Display

Hex No.	Seven Segment Conversion								7segment equivalent
	dot	g	f	e	d	c	b	a	
0	1	1	0	0	0	0	0	0	CO
1	1	1	1	1	1	0	0	1	F9
2	1	0	1	0	0	1	0	0	A4
3	1	0	1	0	0	0	0	0	B0
4	1	0	0	0	1	0	0	1	99
5	1	0	0	0	0	0	1	0	92
6	1	0	0	0	0	0	1	0	82
7	1	1	1	1	1	0	0	0	F8
8	1	0	0	0	0	0	0	0	80
9	1	0	0	1	1	0	0	0	98

Figure 3: Common Anode (active Low) decoder outputs

Hex No.	Seven Segment Conversion								7segment equivalent
	dot	g	f	e	d	c	b	a	
0	0	0	1	1	1	1	1	1	3F
1	0	0	0	0	0	1	1	0	06
2	0	1	0	1	1	0	1	1	5B
3	0	1	0	0	1	1	1	1	4F
4	0	1	1	0	0	1	1	0	66
5	0	1	1	0	1	1	0	1	6D
6	0	1	1	1	1	1	0	1	7D
7	0	0	0	0	0	1	1	1	07
8	0	1	1	1	1	1	1	1	7F
9	0	1	1	0	1	1	1	1	6F

Figure 4: Common cathode (active High) decoder outputs

LCD Display

The 2x16 character LCD offers character display for an embedded system. It can be used to display numerical information, text message and also special symbol. We can control an LCD using either eight pins (8-bit interface) or four pins (4-bit interface), depending on the I/O pins that we have.

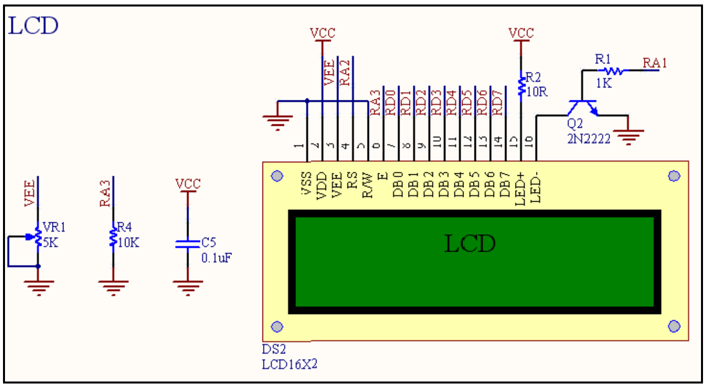


Figure 5 : Connection of a 2x16 character LCD

PROCEDURES

A. Write and Compile Source Code

1. Open Arduino Editor.
2. Write and compile three (3) source code as below.

i. LED Blink Source Code

```
void setup ( ) {
                                          // initialize digital pin LED_BUILTIN as an output.
  pinMode (13, OUTPUT);
}

                                     // the loop function runs over and over again forever
void loop ( ) {
  digitalWrite (13, HIGH);    // turn the LED on (HIGH is the voltage level)
  delay (1000);               // wait for a second
  digitalWrite (13, LOW);     // turn the LED off by making the voltage LOW
  delay (1000);               // wait for a second
}
```

ii. Seven Segment Source Code

```
void setup() {
  // put your setup code here, to run once:
pinMode(2,OUTPUT);
pinMode(3,OUTPUT);
pinMode(4,OUTPUT);
pinMode(5,OUTPUT);
pinMode(6,OUTPUT);
pinMode(7,OUTPUT);
pinMode(8,OUTPUT);
}

void loop() {
  // put your main code here, to run repeatedly:
zero();
one();
two();
three();
four();
five();
six();
seven();
eight();
nine();

}
void zero() {
  digitalWrite(2,HIGH);
  digitalWrite(3,HIGH);
  digitalWrite(4,HIGH);
  digitalWrite(5,HIGH);
  digitalWrite(6,HIGH);
  digitalWrite(7,HIGH);
  digitalWrite(8,LOW);
 delay (1000);
}
void one() {
  digitalWrite(2,LOW);
  digitalWrite(3,HIGH);
  digitalWrite(4,HIGH);
  digitalWrite(5,LOW);
  digitalWrite(6,LOW);
  digitalWrite(7,LOW);
  digitalWrite(8,LOW);
 delay (1000);
}
```

```
void two() {
  digitalWrite(2,HIGH);
  digitalWrite(3,HIGH);
  digitalWrite(4,LOW);
  digitalWrite(5,HIGH);
  digitalWrite(6,HIGH);
  digitalWrite(7,LOW);
  digitalWrite(8,HIGH);
 delay (1000);
}
void three() {
  digitalWrite(2,HIGH);
  digitalWrite(3,HIGH);
  digitalWrite(4,HIGH);
  digitalWrite(5,HIGH);
  digitalWrite(6,LOW);
  digitalWrite(7,LOW);
  digitalWrite(8,HIGH);
 delay (1000);
}
void four() {
  digitalWrite(2,LOW);
  digitalWrite(3,HIGH);
  digitalWrite(4,HIGH);
  digitalWrite(5,LOW);
  digitalWrite(6,LOW);
  digitalWrite(7,HIGH);
  digitalWrite(8,HIGH);
 delay (1000);
}
void five() {
  digitalWrite(2,HIGH);
  digitalWrite(3,LOW);
  digitalWrite(4,HIGH);
  digitalWrite(5,HIGH);
  digitalWrite(6,LOW);
  digitalWrite(7,HIGH);
  digitalWrite(8,HIGH);
 delay (1000);
}
```

```
void six() {
  digitalWrite(2,HIGH);
  digitalWrite(3,LOW);
  digitalWrite(4,HIGH);
  digitalWrite(5,HIGH);
  digitalWrite(6,HIGH);
  digitalWrite(7,HIGH);
  digitalWrite(8,HIGH);
 delay (1000);
}
void seven() {
  digitalWrite(2,HIGH);
  digitalWrite(3,HIGH);
  digitalWrite(4,HIGH);
  digitalWrite(5,LOW);
  digitalWrite(6,LOW);
  digitalWrite(7,LOW);
  digitalWrite(8,LOW);
 delay (1000);
}
```

```
void eight() {
  digitalWrite(2,HIGH);
  digitalWrite(3,HIGH);
  digitalWrite(4,HIGH);
  digitalWrite(5,HIGH);
  digitalWrite(6,HIGH);
  digitalWrite(7,HIGH);
  digitalWrite(8,HIGH);
 delay (1000);
}
void nine() {
  digitalWrite(2,HIGH);
  digitalWrite(3,HIGH);
  digitalWrite(4,HIGH);
  digitalWrite(5,HIGH);
  digitalWrite(6,LOW);
  digitalWrite(7,HIGH);
  digitalWrite(8,HIGH);
 delay (1000);
}
```

iii. LCD Display Source Code

```
// include the library code:
#include <LiquidCrystal.h>

// initialize the library with the numbers of the interface pins
LiquidCrystal lcd (12, 11, 5, 4, 3, 2);

void setup ( ) {
  // set up the LCD's number of columns and rows:
  lcd.begin (16, 2);
  // Print a message to the LCD.
  lcd.print ("Embedded Robotic");
}

void loop ( ) {
  // set the cursor to column 0, line 1
  // (note: line 1 is the second row, since counting begins with 0):
  lcd.setCursor (0, 1);
  // print the number of seconds since reset:
  lcd.print(millis ( ) / 1000);
}
```

B. Schematic Circuits Sketches

1. Open Proteus ISIS Schematic Capture.
2. Select the Component Mode from the left Toolbar.
3. Click On P (Pick From Libraries)
4. Add all the required components.
5. Place the components on the workspace.
6. Wire up the circuit.
7. Click on Play Button on the bottom left to start simulation.

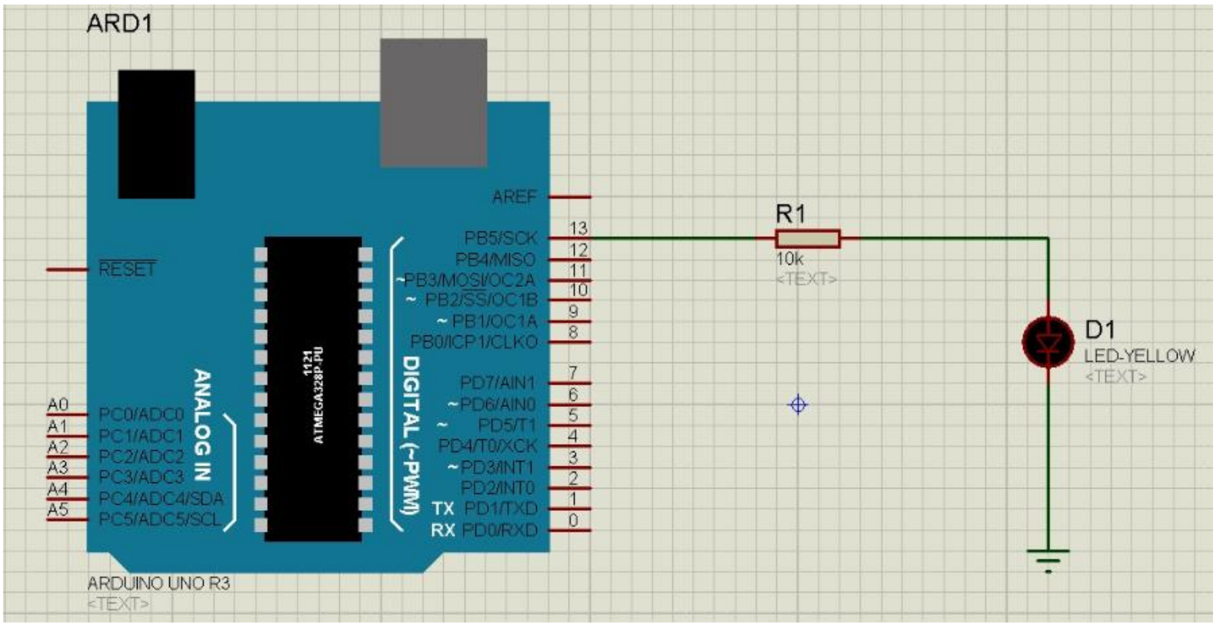


Figure 6 : LED Blink Schematic Circuits

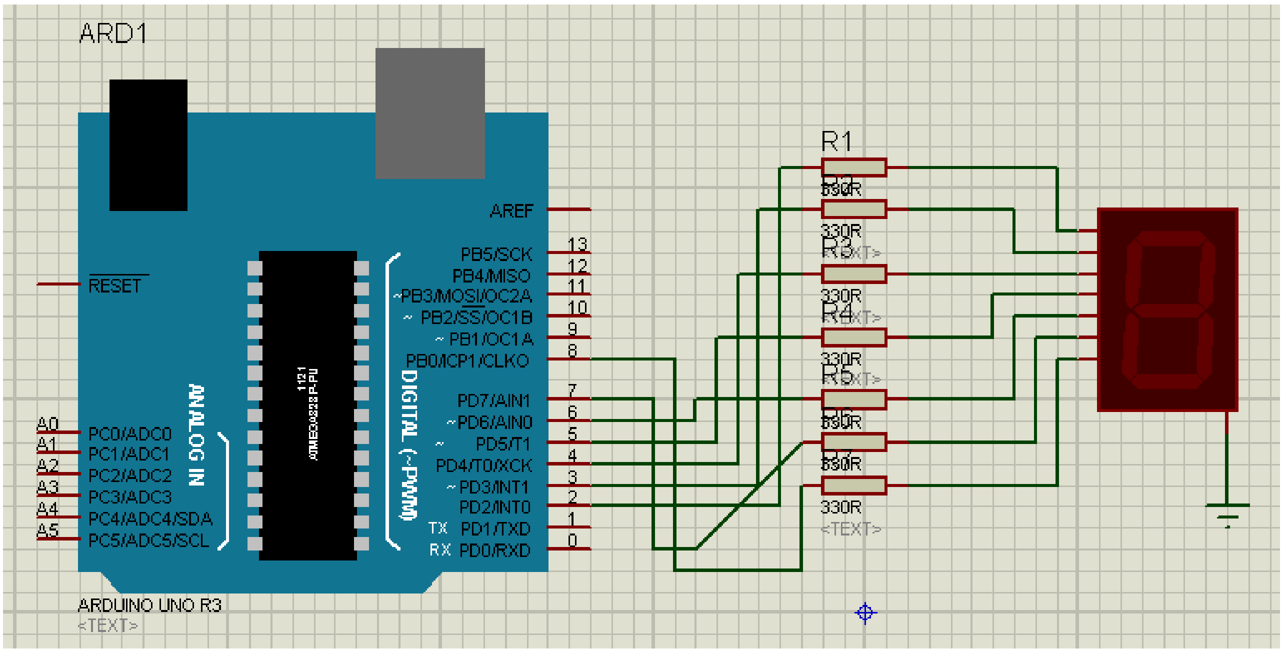


Figure 7 : Seven Segment Schematic Circuits

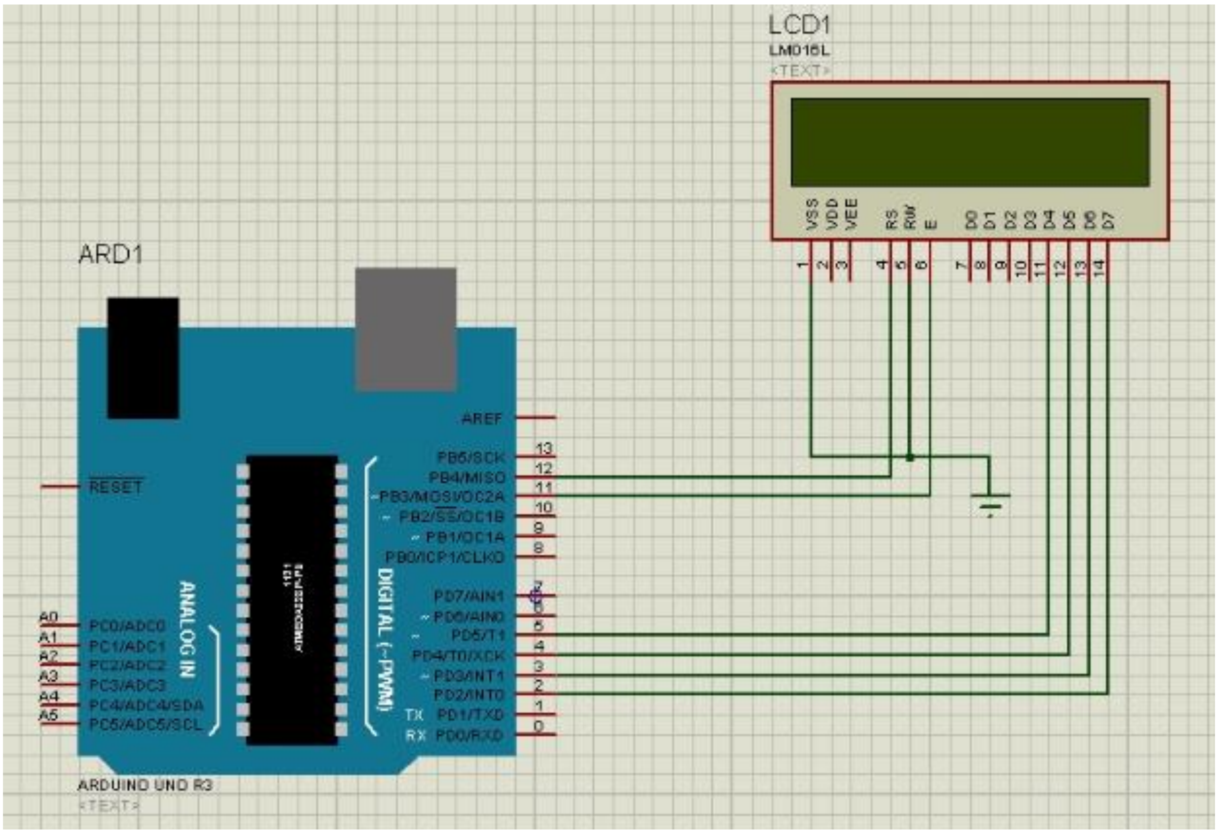


Figure 8 : LCD Display Schematic Circuits

RESULTS

1. LED BLINK

2. SEVEN SEGMENT

3. LCD DISPLAY

QUESTIONS

1. Write a source code to blink TWO LEDs alternately.
2. Write a source code to display number 9 to 0 on the seven segment.
3. Write a source code to display TWO lines message on LCD display.

PRACTICAL WORK 3

INTRODUCTION TO ROBOTIC AND MICROCONTROLLER

LEARNING OBJECTIVES:

Students will be able to;

a) assembles a C program to control DC motors, LM35, LDR and ultrasonic sensor.
b) sketch a schematic circuit of Arduino Uno using Proteus.
c) make an observation on the output of C program using Proteus.

EQUIPMENTS:

1. Arduino Editor
2. Proteus Software
3. Computer

THEORY:

CIRCUIT SIMULATION IN PROTEUS

Proteus in Education Circuit simulation gives students a fast and fun practical learning tool. The software solution allows instructors to prepare and re-use virtual labs. Flexible licensing gives freedom for classes and assignments to be completed anywhere.

It is a software suite containing schematic, simulation as well as PCB designing. ISIS is the software used to draw schematics and simulate the circuits in real time. The simulation allows human access during run time, thus providing real time simulation.

LM35 temperature sensor

LM35 is a temperature sensor that outputs an analog signal which is proportional to the instantaneous temperature. The output voltage can easily be interpreted to obtain a temperature reading in Celsius. The advantage of LM35 over thermistor is it does not require any external calibration.

LDR sensors

An LDR is a component that has a (variable) resistance that changes with the light intensity that falls upon it. This allows them to be used in light sensing circuits. Light Dependent Resistors (LDR) are also called photoresistors. They are made of high resistance semiconductor materials.

Ultrasonic Sensor

An ultrasonic sensor is an instrument that measures the distance to an object using ultrasonic sound waves. An ultrasonic sensor uses a transducer to send and receive ultrasonic pulses that relay back information about an object's proximity.

PROCEDURE A - CIRCUIT DIAGRAM SKETCHES

1. Open Proteus ISIS Schematic Capture.
2. Select the Component Mode from the left Toolbar.
3. Click On P (Pick From Libraries)
4. Add all required components.
5. Place the components on the workspace.
6. Wire up the circuit.
7. Save as your design.

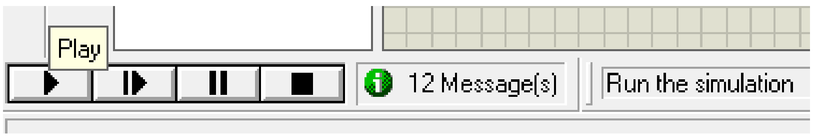


Figure 1 : Click Play to Run Simulation

PROCEDURE B - WRITING SOURCE CODE

1. Verify Arduino source code (A-D) until **Done compiling**.
2. Copy *.hex file from the location:
   C://users/user/AppsData/Local/Temp/ArduinoBuild/
3. Copy *.hex file to your folder.
4. Open Proteus circuit.
5. Double-click Arduino Uno R3 board.
6. Browse *.hex file in your folder.
7. Click on Play Button on the bottom left to start simulation.
8. Observe the output of the simulation.

A. DC MOTOR

```
int val = 200;     //analog value 0-255

void setup ( ) {
  // put your setup code here, to run once:

}

void loop ( ) {
  // put your main code here, to run repeatedly:
analogWrite (9,val);
analogWrite (10,0);
delay (10);
}
```

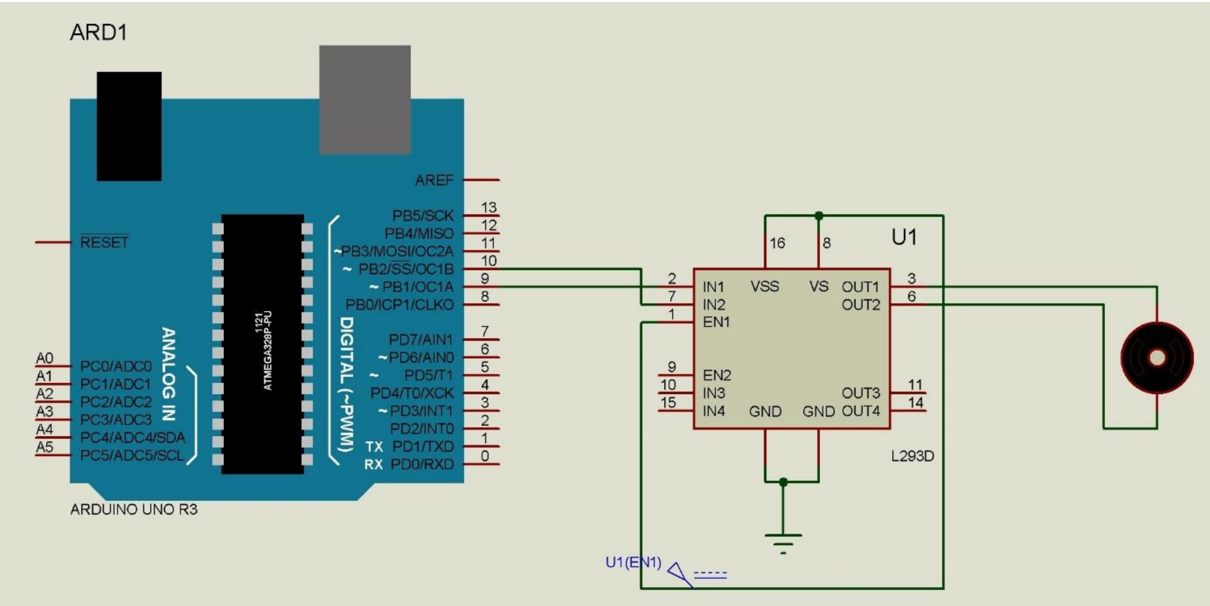


Figure 2 : DC Motor Schematic Diagram

B. LM35

```
int val;
int tempPin = 1;

void setup() {
  // use a for loop to initialize each pin as an output:
pinMode (12, OUTPUT);
pinMode (13, OUTPUT);
Serial.begin (9600);
}

void loop() {
  // loop from the lowest pin to the highest:
val = analogRead (tempPin);
float mv = (val/1024.0) *5000;
float cel = mv/10;

Serial.print (“TEMPRATURE =    “);
Serial.print (cel);
Serial.print (“*C”);
Serial.println ( );

if (cel<30)
{
   digitalWrite (12,HIGH);
   digitalWrite (13,LOW);
}
else {
   digitalWrite (13,HIGH);
   digitalWrite (12,LOW);
}
delay (500);
}
```

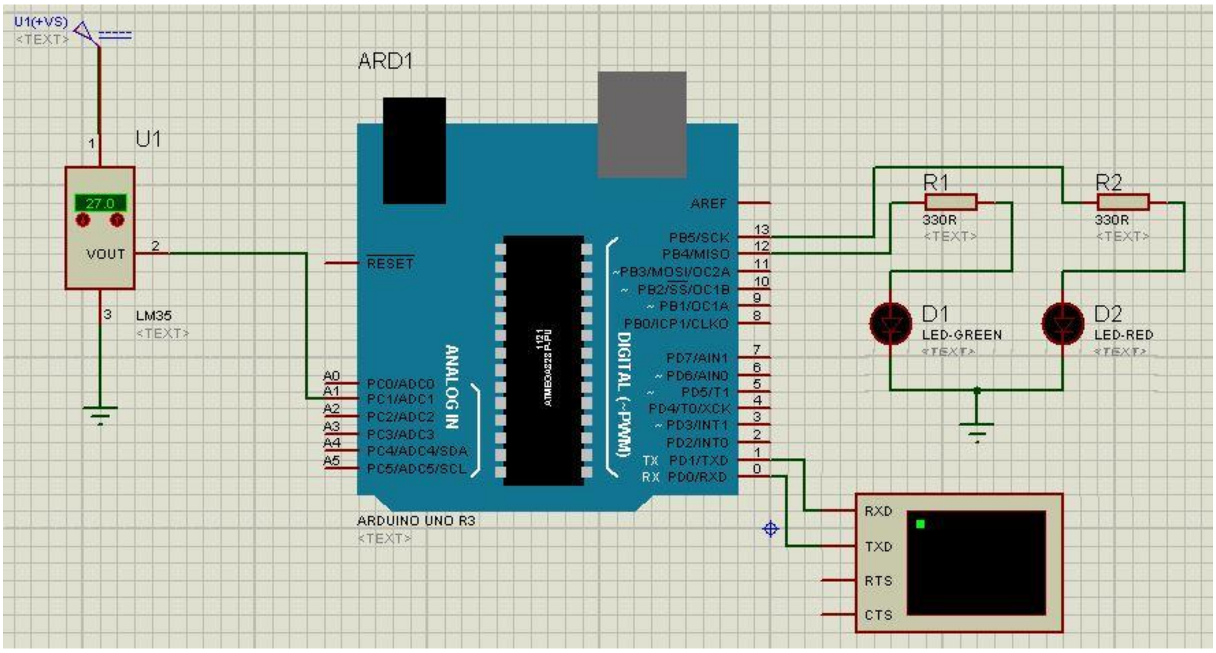


Figure 3 : LM35 Schematic Diagram

C. LDR WITH TORCH

```
void setup ( ) {
  // use a for loop to initialize each pin as an output:
pinMode (10, OUTPUT);
Serial.begin (9600);
}

// the loop routine runs over and over again forever:
void loop ( ) {
// read the input on analog pin 0:
Int sensorValue = analogRead(A0);      //A0 is set here

//printout the value you read:
Serial.println(sensorValue);

if (sensorValue<15)
{
  digitalWrite (10,HIGH);
}
else
{
  digitalWrite (10,LOW);
}
delay (1);       //delay in between reads for stability
}
```

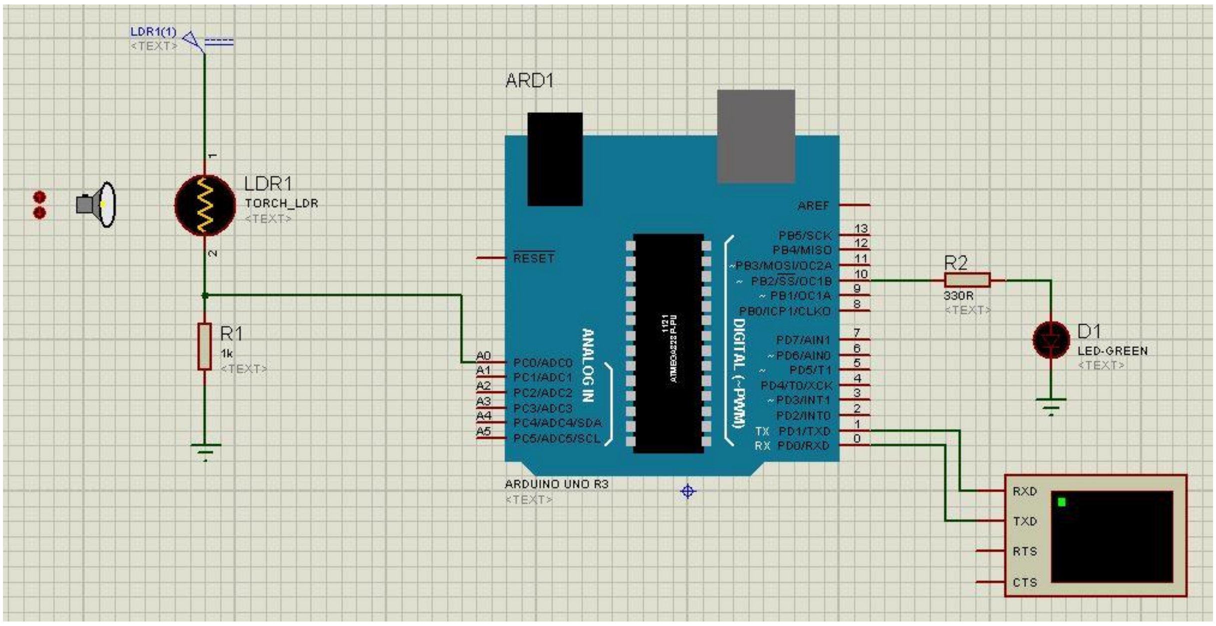


Figure 4 : LDR With Torch Schematic Diagram

D. LED KNIGHT RIDER

```
int timer = 100;     //the higher the number, the slower the timing

void setup() {
  // use a for loop to initialize each pin as an output:
for (int thisPin = 2; thisPin < 8; thisPin++) {
   pinMode (thisPin, OUTPUT);    }
}

// the loop routine runs over and over again forever:
void loop() {
// loop from the lowest pin to the highest:
for (int thisPin = 2; thisPin < 8; thisPin++) {
   //turn the pin on:
   digitalWrite (thisPin, HIGH);
   delay(timer);
   //turn the pin off:
   digitalWrite (thisPin, LOW);   }

// loop from the highest pin to the lowest:
for (int thisPin = 7; thisPin >= 2; thisPin--) {
   //turn the pin ON:
   digitalWrite (thisPin, HIGH);
   delay(timer);
   //turn the pin OFF:
   digitalWrite (thisPin, LOW);    }
}
```

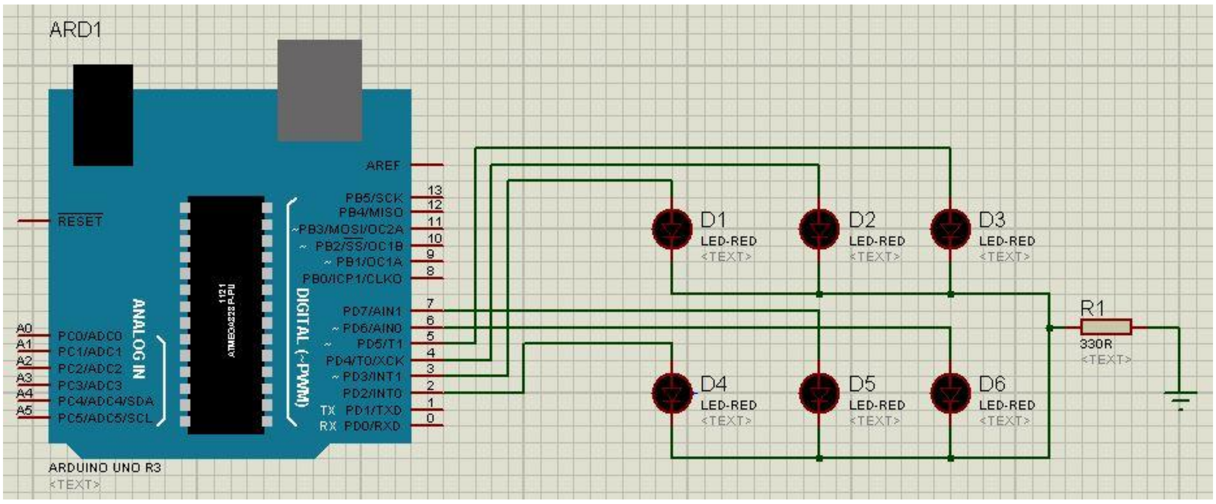


Figure 5 : LED Knight Rider Schematic Diagram

RESULTS

1. DC MOTOR

....

....

....

2. LM35

....

....

....

3. LDR WITH TORCH

....

....

....

4. LED KNIGHT RIDER

....

....

....

QUESTIONS

Explain the function of the source codes below:

1. int val = 200;
2. if (cel<30)……..else……..
3. Serial.begin (9600);
4. int sensorValue = analogRead(A0);
5. for (int thisPin = 7; thisPin >=2; thisPin--)

PRACTICAL WORK 4

ROBOTIC CONTROLLER PROGRAMMING IN C

LEARNING OBJECTIVES:

Students will be able to;

a) assembles a C program to control DC motors.
b) complete the hardware installation of robot base with two DC motors.
c) build a C program to control DC motor directions forward and reverse.
d) make an observations of program C on DC motors.

EQUIPMENTS / COMPONENTS:

ARDUINO UNO Board and DC motors, L298 (Dual full-bridge driver)

THEORY:

Let us find out how a DC motor actually runs. Direction control of a DC motor is very simple, just reverse the polarity to make reverse rotation. It means that every DC motor has two terminals out. When we apply DC voltage 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.

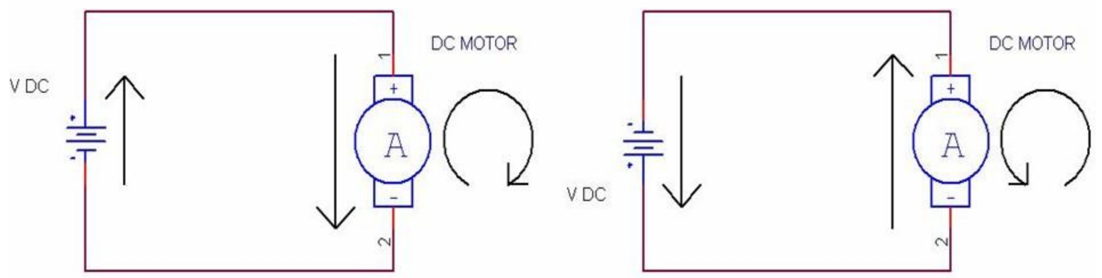


Figure 1 : Motor Direction Control

It is a special circuit which allows motor rotation in both directions. From four terminals of a H bridge, you can control the direction of a DC motor. Depending on current and power requirements, we can make our own H bridge using transistors/MOSFETs but it will be better to demonstrate the working, if we use some ready-made IC such as L298, as it is a dual full-bridge driver.

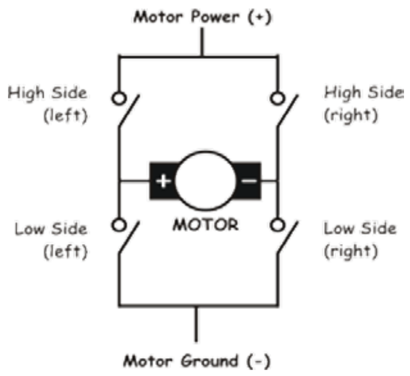


Figure 2 : Dual Full-Bridge Driver

PROCEDURE

PART A

1. Open the Proteus software, select the following part from library
    a. Arduino Uno R3
    b. L298, dual full-bridge driver
    c. Motor, simple DC motor model
2. Create new project, use naming profile <PW4_X>, with X indicate your group number.
3. Drag all component into schematic layout, and use your own creativity for component arrangement and pin connection. Use below schematic diagram below for guideline.
4. Save your project.

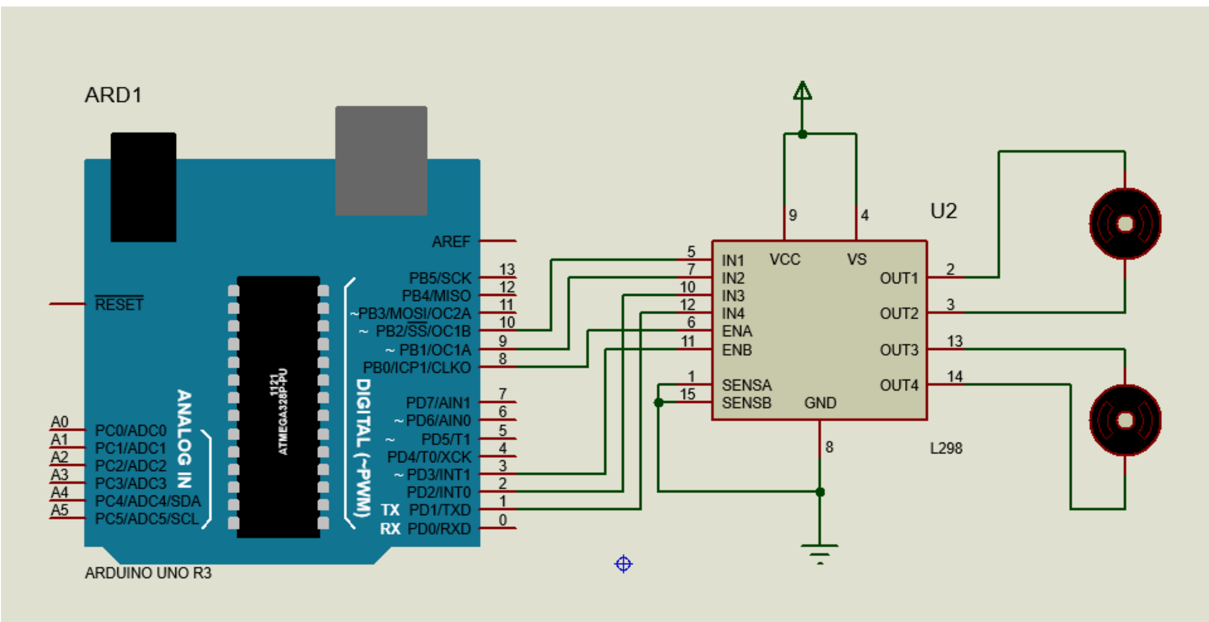


Figure 3 : DC Motors Schematic Diagram

PART B

1. Open the Arduino software. Click on the ‘File’ menu and select ‘Preferences’.

   Click on the checkbox as below.

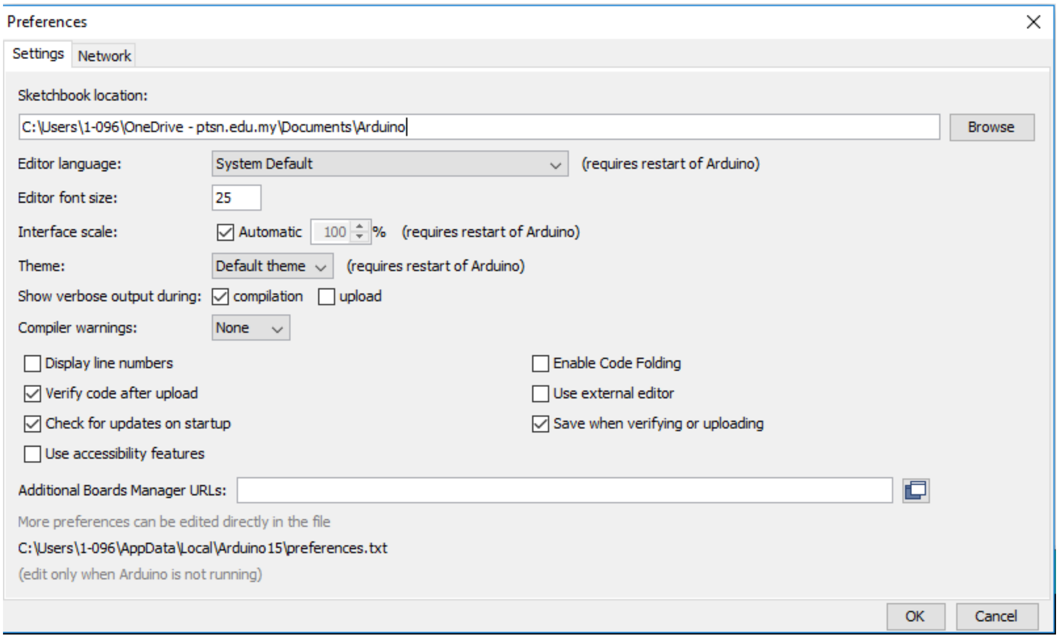


Figure 4 : Preferences Setting

2. Click ‘OK’ to close the pop-up window.
3. Write the following code and save it using the same naming profile as before.
4. Compile the code and find the hex file location. If any error occurs, solve the error until it is “Done compiling”.

```
//declaration
#define    motorA   8          //motor A  enable  pin connect to pin  8
#define    motorB   3          //motor B  enable  pin  connect  to pin 3

void setup () {
  //put your  setup  code here, to run  once:
  pinMode (1,OUTPUT);         //make    pin   as output
  pinMode (2, OUTPUT);
  pinMode  (motorB, OUTPUT);
  pinMode (motorA, OUTPUT);
  pinMode (9, OUTPUT);
  pinMode (10, OUTPUT);   }

void loop () {
 // put your main code here, to run repeatedly:
//rotate motorA CW
digitalWrite (motorA, HIGH);      //enable pin for motorA
digitalWrite (9, HIGH);           //change both signal to change rotation
digitalWrite (10, LOW);

delay (1000);                     //delay 1 second

//rotate motorB CCW
digitalWrite (motorB, HIGH);      //enable pin for motorB
digitalWrite (1, HIGH);           //change both signal to change rotation
digitalWrite (2, LOW);   }
```

PART C

1. Double click on the Arduino Uno R3 component in Proteus to invoke the 'Edit Component' window. On 'Program Files' part, point it to your hex file location on PART B, then click 'OK'.
2. On the lower left side of your schematic layout, click the play button to start the simulation. Observe the output.

QUESTION:

1. Sketch and simulate a code to make both of your DC motors to rotate at the same time.
2. Sketch and simulate a code to make motor A and motor B to rotate CW for 1 second and rotate CCW for 1 second. Make it run on infinite looping.
3. Sketch a program to make robot spin CW for 5 seconds, Stop 1 second and then spin CCW for 5 seconds and completely stop.

PRACTICAL WORK 5

SENSORS AND ACTUATORS PROGRAMMING IN C

LEARNING OBJECTIVES:

Students will be able to;

a) assembles a C program to control DC motors and sensors.
b) complete the hardware installation of Line Follower Robot (LFR).
c) perform an application of LFR using DC motors, ultrasonic sensor and line sensor.
d) make an observation on the movement of LFR.

EQUIPMENTS / COMPONENTS:

1. ARDUINO UNO Board and DC motors, L298 (Dual full-bridge driver)
2. HC-SR04 ultrasonic sensor

THEORY:

The human ear can hear sound with the frequency of around 20Hz and 20KHz, and ultrasonic is the sound wave beyond the human ability of 20KHz.

Ultrasonic distance measurement principle

Ultrasonic transmitter emitted an ultrasonic wave in one direction, and started timing when it launched. Ultrasonic spread in the air, and would return immediately when it encountered obstacles on the way. At last, the ultrasonic receiver would stop the timing when it received the reflected wave.

As ultrasonic spread velocity is 340 m/s in the air, based on the timer record t, we can calculate the distance (s) between the obstacle and transmitter, namely: $s = 340*t/2$, which is so called time difference distance measurement principle.

The principle of ultrasonic distance measurement uses the already-known air spreading velocity, measuring the time from launch to reflection when it encountered obstacle, and the calculation of the distance between the transmitter and the obstacle according to the time and the velocity. Thus, the principle of ultrasonic distance measurement is the same with radar. Distance measurement formula is expressed as: $L = CXT$. In the formula, L is the measured distance, and C is the ultrasonic spreading velocity in air, also, T represents time (T is half the time value from transmitting to receiving).

Ultrasonic Application Technology has been actively developed in recent decades. With the ultrasonic advance, and the electronic technology development, especially when high-power semiconductor device technology matures, the application of ultrasonic has become increasingly widespread:

i. Ultrasonic measurement of distance, depth and thickness;
ii. Ultrasonic testing;
iii. Ultrasound imaging;
iv. Ultrasonic machining, such as polishing, drilling;
v. Ultrasonic cleaning;
vi. Ultrasonic welding;

PROCEDURE

PART A

1. Open the Proteus software, select the following parts from library
   a. Arduino Uno R3
   b. LM016L, 16x2 LCD display
   c. SRF04, ultrasonic sensor
2. Create a new project, use naming profile as PW5_X, with X indicating your group number.
3. Drag all components into schematic layout, and use your own creativity for component arrangement and pin connection. Use schematic diagram below for guideline.
4. Save your project.

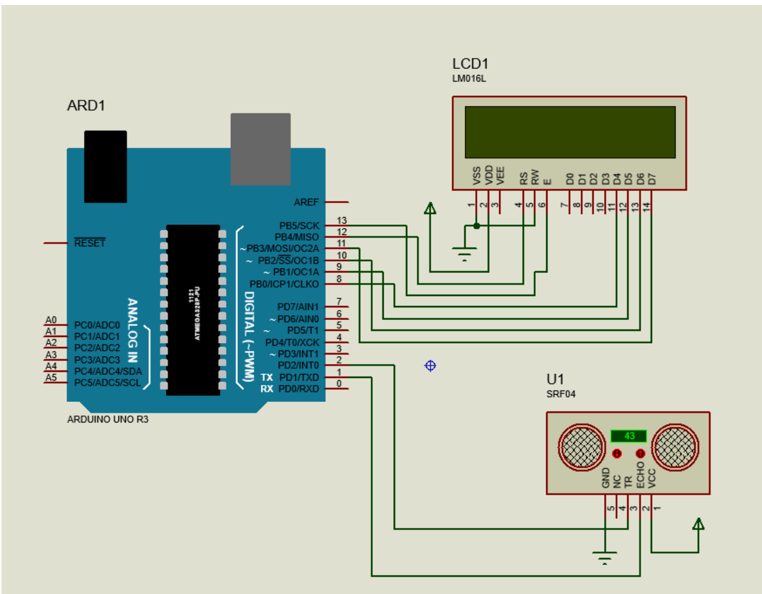


Figure 1 : Ultrasonic Schematic Diagram

PART B

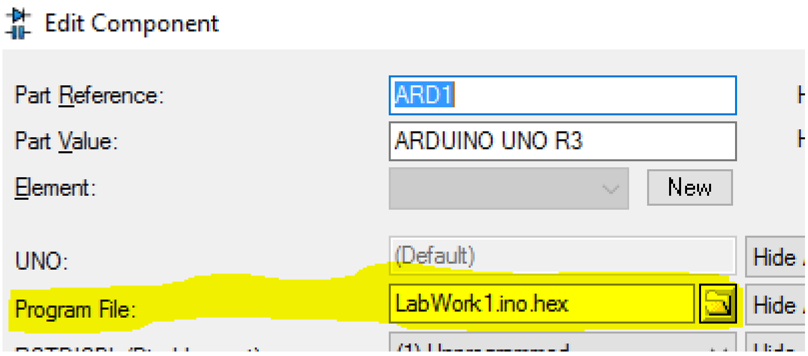
1. Open the Arduino editor.
2. Write the following code and save it using the same naming profile as before.

```
#include <NewPing.h>                //ultrasonic library
#include<LiquidCrystal.h>           //lcd display library

//for ultrasonic sensor setup pin
#define TRIGGER_PIN 2        // Arduino pin tied to trigger pin on the ultrasonic sensor.
#define ECHO_PIN 1           // Arduino pin tied to echo pin on the ultrasonic sensor.
#define MAX_DISTANCE 200  // Maximum distance we want to ping for (in centimeters). Maximum sensor
                                  distance is rated at 400-500cm

NewPing sonar (TRIGGER_PIN, ECHO_PIN, MAX_DISTANCE);      // NewPing setup of pins and maximum
                                           distance.  built-in function wihtin newping library

LiquidCrystal lcd (12,13,8,9,10,11);    //for lcd display. Initialize the library with the numbers of
                                    the interface pins

void setup ( ) {                  // set up the LCD's number of columns and rows:
lcd.begin (16, 2);
lcd.print ("Ultrasonic  Sensor");     //Print a message to the LCD
}

void loop ( ) {
delay (50);               // Wait 50ms between pings (about 20 pings/sec). 29ms should be the shortest
                     delay between pings

//(note:line 1 is the second row, since counting begins with 0):
lcd.setCursor (0, 1);      // set the cursor to column 0, line 1

 lcd.print ("Ping: ");
 lcd.print (sonar.ping_cm( ));   // Send ping, get distance in cm and print result (0 = outside set distance
                            range)
 lcd.println ("cm"); }
```

3. Compile the code and find the hex file location. If any error occurs, solve the error until it fixes.

PART C

1. Double click on the Arduino Uno R3 component in Proteus to invoke the 'Edit Component' window. On 'Program Files' part, point it to your hex file location on PARTB, then click 'OK'.



2. On the lower left side of your schematic layout, click the play button to start the simulation. Observe the output.

QUESTION:

Sketch and simulate a code to make both of your DC motor to STOP rotation IF ultrasonic sensor reading is below 20cm. DC motor will resume rotation if the ultrasonic sensor reading is more than 20cm. You may remove the LCD display during simulation.

PRACTICAL WORK 6

MOBILE ROBOT DESIGN

LEARNING OBJECTIVES:

Students will be able to;

a) complete the hardware installation of Land Mobile Robot (LMR) using DC motors and Bluetooth module.
b) perform an application of Robot Sumo using LMR and Bluetooth module.
c) organizes a Robot Sumo Competition.

EQUIPMENTS / COMPONENTS:

1. ARDUINO UNO Board and DC motors, L298 (Dual full-bridge driver)
2. HC-SR04 ultrasonic sensor
3. Differential drive mobile robot

PROCEDURE

1. With the help from the previous practical work, built an obstacle avoidance differential drive mobile robot.
2. Student must use ultrasonic sensor as the detector for obstacle.
3. Set the limit range of 15 cm between mobile robot and obstacle. Mobile robot will turn right at about 90 degrees before moving straight again.
4. Upload your code into mobile robot and observe the output.

HINTS

1. Use every knowledge from previous practical work.
2. Divide your code into smaller section (create user define function) for more convenient control and so it is easy to troubleshoot.
3. Optimize the use of built-in delay function to make the mobile robot turn approximately 90 degrees.
4. There is no right or wrong in programming IF you can achieve your objective, the difference is only about efficiency, optimization and how you manage your resource.

QUESTION:

1. Draw a flow chart for a procedure above.
2. Suggest way/strategies to make the mobile robot to move smoothly. Describe your solution in detail.

REFERENCES

1. Alba-Flores, R. (2007). Laboratory Enhancements for Improving Embedded Systems Education. *Proceeding of the 2007 American Society for Engineering Educational Annual Conference & Exposition*. American Society for Engineering Education.
2. Jawaid, I., Javed, M. Y., Jaffery, M. H., Akram, A., Safder, U., & Hassan, S. (2020). Robotic system education for young children by collaborative-project-based learning. *Computer Applications in Engineering Education*, *28*(1), 178–192. https://doi.org/10.1002/cae.22184
3. Larson, J., Jordan, S. S., Lande, M., & Weiner, S. (2020). Supporting Self-Directed Learning in a Project-Based Embedded Systems Design Course. *IEEE Transactions on Education*, *63*(2), 88–97. https://doi.org/10.1109/TE.2020.2975358
4. Rosa, A. H. R., Ferreira, R. V., & Pereira, C. A. (2021). Integrated PBL and HIL practices for real-time simulations applied in technical and engineering teaching using embedded systems. *Przeglad Elektrotechniczny*, *97*(1), 46–52. https://doi.org/10.15199/48.2021.01.08
5. Tian, J. (2021). Optimization of Embedded Mobile Teaching Model Based on Network Streaming Media Technology. *Complexity*, *2021*(4). https://doi.org/10.1155/2021/3449338
6. Uzun, A. (2020). Using Educational Robotics as a Cognitive Tool for ICT Teachers in an Authentic Learning Environment. *International Education Studies*, *13*(4), 27. https://doi.org/10.5539/ies.v13n4p27
7. https://www.arduino.cc/
8. https://www.theengineeringprojects.com/
9. https://www.labcenter.com/downloads/