A Practical Activity Report Submitted
For Engineering Design II (UTA-014)

By

101856013 Vansh Tyagi

COMPUTER SCIENCE ENGINEERING DEPARTMENT

THAPAR INSTITUTE OF ENGINEERING & TECHNOLOGY
PATIALA, PUNJAB
(A DEEMED TO BE UNIVERSITY)

INDIA
ODD-SEM 2019

EXPERIMENT – 1

1.1 OBJECTIVE : Introduction to Arduino Microcontroller.

1.2 HARDWARE USED : Arduino UNO board R3 ATmega328p.

1.3 SOFTWARE USED : Arduino IDE (Version 1.8.9)

1.4 THEORY :

A microcontroller is a compact integrated circuit designed to govern a specific operation in an embedded system. A typical microcontroller includes a processor, memory and input/output (I/O) peripherals on a single chip. Sometimes referred to as an embedded controller or microcontroller unit (MCU), microcontrollers are found in vehicles, robots, office machines, medical devices, mobile radio transceivers, vending machines and home appliances among other devices.

Arduino board designs use a variety of microprocessors and controllers. The boards are equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards (shields) or Breadboards (other circuits on them).

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.

The input voltage (7 – 12 V) to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through the Vin pin, or, if supplying voltage via the power jack, access it directly through Vin pin. Here Architecture is of Arduino or precisely the IC of Arduino (ATmega328p). The ATmega328/P is a low-power CMOS 8-bit microcontroller based on the AVR® enhanced RISC architecture. In Order to maximize performance and parallelism, the AVR uses Harvard architecture – with separate memories and buses for program and data. Instruction in the program memory are executed with a single level of pipelining.The clock is controlled by an external 16MHz Crystal Oscillator.

1.5 LOGIC/CIRCUIT DIAGRAM

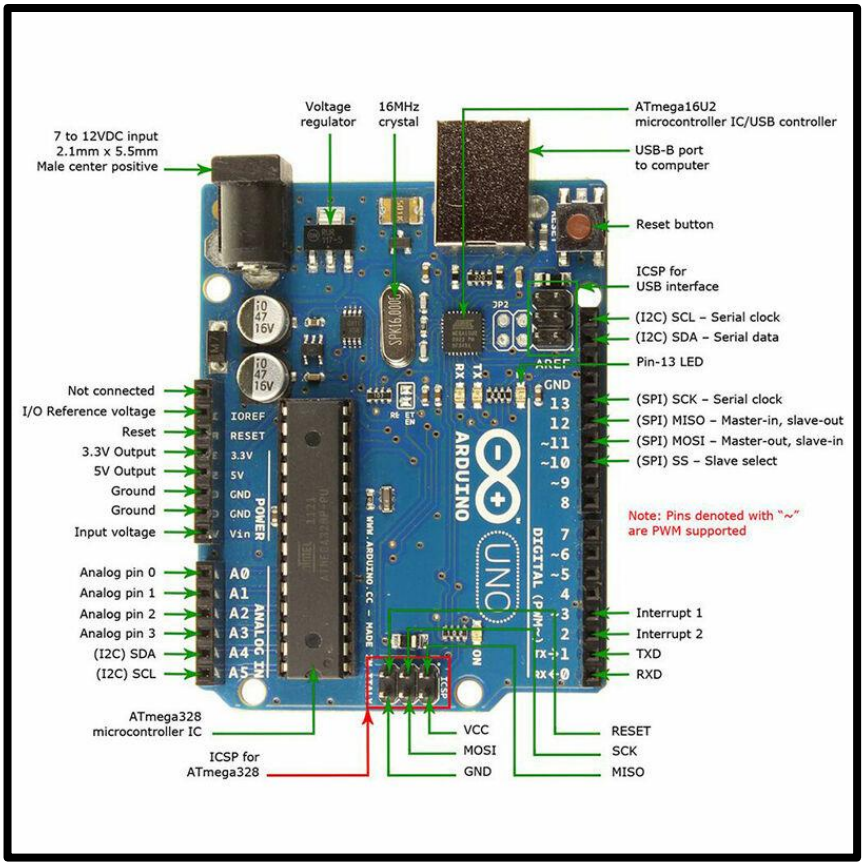


Fig. 1.1 Arduino Uno R3 ATmega328

1.6 RESULT ANALYSIS :

This Arduino microcontroller (specifically ATmega328p IC) will help us to communicate with the buggy (Nvis 3302ARD RoboCar) with some help of Arduino coding. Therefore this component becomes a significant part of our project.

EXPERIMENT – 2

2.1 OBJECTIVE : To blink a LED.

2.2 HARDWARE USED : LED, Arduino UNO board (ATmega328p), Connecting wires, Breadboard, USB connector, Resistor.

2.3 SOFTWARE USED : Arduino IDE (Version 1.8.9)

2.4 THEORY :

An LED is a small light (it stands for "light emitting diode") that works with relatively little power. The Arduino board has one built-in on digital pin 13.

The second thing we need to do is configure as an output the pin connected to the LED. We do this with a call to the pinMode() function, inside of the sketch's setup() function:

Finally, we have to turn the LED on and off with the sketch's loop() function. We do this with two calls to the digitalWrite() function, one with HIGH to turn the LED on and one with LOW to turn the LED off. If we simply alternated calls to these two functions, the LED would turn on and off too quickly for us to see, so we add two calls to delay() to slow things down. The delay function works with milliseconds, so we pass it 1000 to pause for a second.

LEDs have polarity, which means they will only light up if you orient the legs properly. The long leg is typically positive, and should connect to a digital pin on the Arduino board. The short leg goes to GND; the bulb of the LED will also typically have a flat edge on this side. In order to protect the LED, you will also need use a resistor "in series" with the LED. If the LED doesn't light up, trying reversing the legs (you won't hurt the LED if you plug it in backwards for a short period of time).

2.5 CODE :

```
void setup()
{
  pinMode(1, OUTPUT);
}
void loop()
{
  digitalWrite(1, HIGH);
  delay(1000);
  digitalWrite(1, LOW);
  delay(1000);
}
```

2.8 LOGIC/CIRCUIT DIAGRAM :

2.7 RESULT ANALYSIS : This experiment helped us to use the pins and set them HIGH or LOW on the Arduino microcontroller. We also learned the use of digitalWrite(), void setup() and void loop().

EXPERIMENT – 3

3.1 OBJECTIVE : To design patterns from sequence of multiple LEDs using for loop in Arduino

3.2 HARDWARE USED : LED, Arduino UNO board (ATmega328p), Jump wires, Breadboard, USB connector, Resistor.

3.3 SOFTWARE USED : Arduino IDE (Version 1.8.9)

3.4 THEORY :

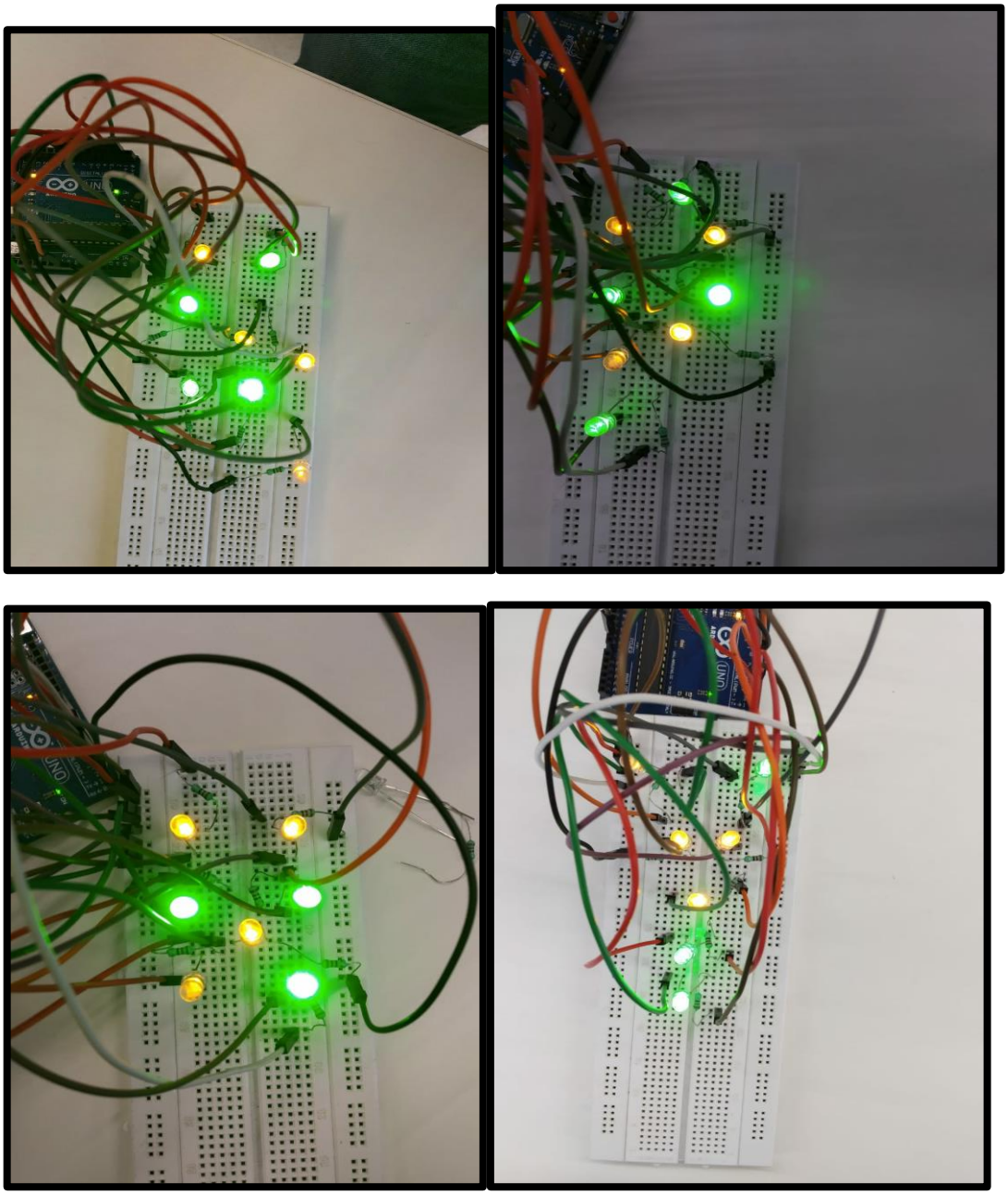
In this experiment we use several LEDS and design a pattern in which they blink. We write the code on the Arduino IDE as per the requirement of the desired pattern. First, all the LEDS will glow after an interval of 1 second each and then all the LEDS will be off one by one after an interval of 1 second.

3.5 CODE :

```
void setup()
{
  for(int i=1;i<=8;i++)
   pinMode(i, OUTPUT);
}


void loop()
{
 for(int i=1;i<=8;i++)
 digitalWrite(i, HIGH);
}
```

3.6 LOGIC/CIRCUIT DIAGRAM :



3.6 RESULT ANALYSIS : This experiment helped us to use multiple pins and set multiple pins as high or low. We hence obtained the desired pattern of the LEDs blinking.

EXPERIMENT – 4

4.1 OBJECTIVE : To demonstrate sending data from the computer to the Arduino board and control brightness of LED

4.2 HARDWARE USED : LED, Arduino UNO board (ATmega328p), Jump wires, Breadboard, USB connector, Resistor.

4.3 SOFTWARE USED : Arduino IDE (Version 1.8.9)

4.4 THEORY :

Arduino microcontroller also provides pin modes for sending of the analog signals. Few pins in the microcontroller helps us to send digital signal and others for the analog signal. We can easily control the brightness of the LED by using analogWrite() function as it can control a range of values as compared to digitalWrite() function which can only turn the LED on or off in the Arduino IDE.

4.5 CODE :

a) To control brightness of a single LED :-

```
int bright=0;
int fade=5;
void setup()
{
  pinMode(6,OUTPUT);
}
void loop()
{
  analogWrite(6,bright);
  bright=bright+fade;
  if(bright<=0||bright>=255)
  {
   fade= -fade;
 }
  delay(50);
```

```
}
```

b) To control brightness of a multiple LEDs :-

```
int fade;
void setup()
{
  pinMode(9,OUTPUT);
  pinMode(10,OUTPUT);
  pinMode(11,OUTPUT);
}
void loop()
{
  for(fade=0;fade<=255;)
  {
     analogWrite(9,fade);
     analogWrite(10,fade);
     analogWrite(11,fade);
     delay(1000);
     fade=fade+40;
  }
  for(fade=255;fade>0;)
  {
     analogWrite(9,fade);
     analogWrite(10,fade);
     analogWrite(11,fade);
     delay(1000);
     fade=fade-40;
  }
}
```

4.6 LOGIC/CIRCUIT DIAGRAM :

4.7 RESULT ANALYSIS :

We learnt how to use the analog signal pins provided in the microcontroller. This helped to increase the brightness of LED slowly to the maximum value and then to the lowest.

EXPERIMENT – 5

5.1 OBJECTIVE : To demonstrate control of DC Motor using forward, backward, left, right turn motion and clock-wise/anti clock- wise rotation.

5.2 HARDWARE USED : Arduino Microcontroller, USB cable, Nvis 3302ARD RoboCar

5.3 SOFTWARE USED : Arduino IDE

5.4 THEORY:

Nvis 3302ARD RoboCar is an electro-mechanical platform with the capability of sensing the environment, processing the data, and acting according to the preprogrammed sequence. It is a miniature prototype car powered by batteries whose motion is controlled by microcontroller. Various Sensors can be interfaced like IR (Infrared) Sensor, Ultrasonic Sensor, Analog Sensor, and many more. It is a multitasking Robocar that can perform actions such as line follower, obstacle detection with wireless operation with DC motor drive. Robocar displays all action on LCD. Nvis 3302ARDRoboCar is a Robot designed for Robotics students. It will help them to get comfortable with the world of Robotics and Embedded Systems.

We can design user defined functions in the Arduino IDE to make the buggy move in our own specified directions like left, right, forward, backward, clockwise and anti clockwise by setting the pins 5, 6, 7 ,8 on Nvis 3302ARD RoboCar either HIGH/LOW.

5.5 CODE :

```
void forward()
{
digitalWrite(5,HIGH);
digitalWrite(6,LOW);
digitalWrite(7,LOW);
digitalWrite(8,HIGH);
}
void backward()
{
```

```
 digitalWrite(5,LOW);
 digitalWrite(6,HIGH);
 digitalWrite(7,HIGH);
 digitalWrite(8,LOW);
 }
 void right()
 {
 digitalWrite(5,LOW);
 digitalWrite(6,LOW);
 digitalWrite(7,LOW);
 digitalWrite(8,HIGH);
 }
 void left()
 {
 digitalWrite(5,HIGH);
 digitalWrite(6,LOW);
 digitalWrite(7,LOW);
 digitalWrite(8,LOW);
 }
 void anticlockwise()
 {
 digitalWrite(5,HIGH);
 digitalWrite(6,LOW);
 digitalWrite(7,HIGH);
 digitalWrite(8,LOW);
 }
 void clockwise()
 {
 digitalWrite(5,LOW);
 digitalWrite(6,HIGH);
 digitalWrite(7,LOW);
digitalWrite(8,HIGH);
```

```
}
void setup()
{

}
```

5.6 LOGIC/CIRCUIT DIAGRAM :