

## CA326 Advanced Embedded Programming

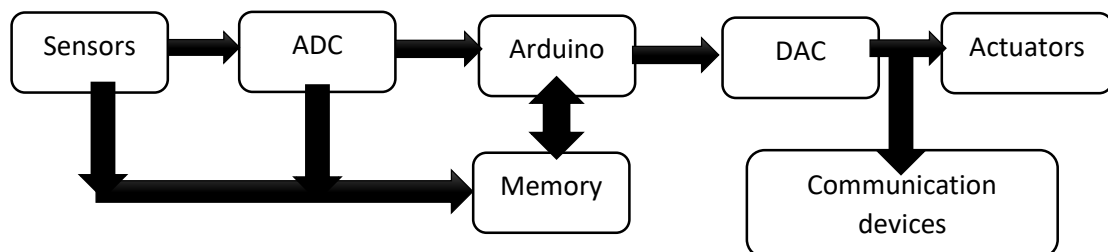
### Lecture Notes:

#### Chapter 1 and Chapter 2

##### **Give the differences between Microprocessor and Microcontroller.**

|    | Microprocessor   | Microcontroller   |
|----|--|---|
| 1. | It has basic units like ALU , control units and registers. | It has units like processing unit, RAM, ROM , ADC and DAC.                  |
| 2. | It can handle multiple operations.                         | It can handle single operation.   |
| 3. | multitasking   | Single tasking  |
| 4. | It requires external memory                                | It doesn't require external memory because it has internal memory available |
| 5. | It is costlier   | It is cheap   |
| 6. | It is heavy weight   | It is light weight  |

##### **Architecture or block diagram of an embedded system using arduino.**



Above figure is diagram of an embedded system.

##### **Embedded system with arduino must have following components.**

1. Sensors / keypad (input)
2. ADC: Analog to Digital Convertor
3. Arduino controller
4. Memory
5. DAC: Digital to Analog Convertor
6. Actuator (output) / Display
7. Communication Device

##### **1. Sensors / keypad (input):**

It reads the input. Input can be from sensor or any other input devices like keypad, switch etc.

**2. ADC:**

Analog to Digital Converter: It converts available input analog signal into digital form because Arduino reads in digital form.

**3. Arduino controller:**

It reads data and processes the data based on instructions. Based on input conditions it controls the outputs.

**4. Memory: (Internal / external)**

It stores data temporarily.

**5. DAC:**

Digital to Analog Converter: It converts digital signal back into analog form because output devices may require analog signal.

**6. Actuator (output):**

It is used to actuate output devices like DC motors, servomotor etc. which convert input energy to mechanical energy.

Instead of actuator simple output devices like LCD, LED or 7 segment LEDs can be used.

**7. Communication Device:**

It is used to transmit data from Arduino to other devices like other Arduino, WiFi module, Bluetooth module etc.

Communication can be of three types

1. I2C: Inter Integrated Circuit
2. SPI: Serial to Peripheral Interface
3. UART: Universal Asynchronous transmitter and receiver.

**Applications of embedded system which can be created using Arduino controller.**

Some of the applications are as under.

**1. Fire alarm system:**

It senses the fire from the sensor and if fire value goes beyond certain threshold then it provides mechanism to prevent the fire.

For IOT system develop one mobile app using Android Studio and connect data with Firebase Cloud. Use the program which sends data to cloud and mobile app to prevent the fire.

**2. Temperature monitoring system**

It monitors the temperature and sets the temperature to certain values only as per the user input.

**3. RFID based authentication system. (attendance system)**

It has RFID cards and RFID reader. Users of the system will use RFID card to mark their attendance.

#### **4. Car safety system**

Cars can have password based ignition system so that it cannot be stolen.

Cars can have mechanism to sense alcohol. If a driver is drunk, then the system will not allow to start the car.

#### **5. Motion detection system**

Motion sensors are used to detect the motion or movement. It can be used at home or offices for security purpose.

#### **6. Object detection system**

It is a system which performs appropriate actions when an object is detected. In mall as soon as object is detected, the door / gate will be opened. Another application is automatic hand sanitizer machine.

#### **7. Water quality measurement system**

This system will find out the **pH level** of the water and TDS level of the water so that proper purifier can be designed to clean the water.

#### **8. Water level detection system**

This can be used to monitor the level of water in overhead tank. If tank is full then further water cannot be supplied to tank. If tank is empty or lower than certain level then water is supplied to tank.

#### **9. Weather monitoring system**

It is used to measure various weather parameters like temperature, humidity, rain, content in the air (air quality index), wind direction etc. Based on value of these parameters appropriate actions can be taken.

#### **10. Weight / height measurement system**

It can be used in gym for calculating BMI (Body Mass Index) based on weight and height. Based on BMI value the customers can be suggested the exercises and diet plans.

#### **11. Obstacle avoidance robot**

Robot can be designed which can detect the object in its path. If object is found then it will decide the new path.

#### **12. Smart Hand sanitizer**

It is an application which can detect the object at the lower part of machine. If distance of an object is in given value range then it automatically spray the sanitizer.

#### **13. Face recognition based attendance**

It is a system where authorized person is allowed to enter the premises if the person has been verified. It can also be used to mark the face based attendance for the students or employee.

#### 14. Heartbeat monitoring system.

This system can be used by hospital to monitor the heartbeat of a patient so that patient can be treated properly.

#### 15. Smart Hand gloves

Smart hand gloves can have various sensors in it so that it can measure parameters of human body who wear the gloves and give the suggestion accordingly.

#### 16. Stick for blind person

Blind person can use the stick to detect any objects during walking. Blind stick can have various motion and object detection sensors to guide the person properly.

#### 17. Farm security system.

This type of system can be used to provide the security of crops from domestic or wild animals.

#### 18. Irrigation system.

It can also be used in the farm for irrigating the crops automatically. System detects the level of moisture and behave accordingly.

#### 19. Soil moisture detection system.

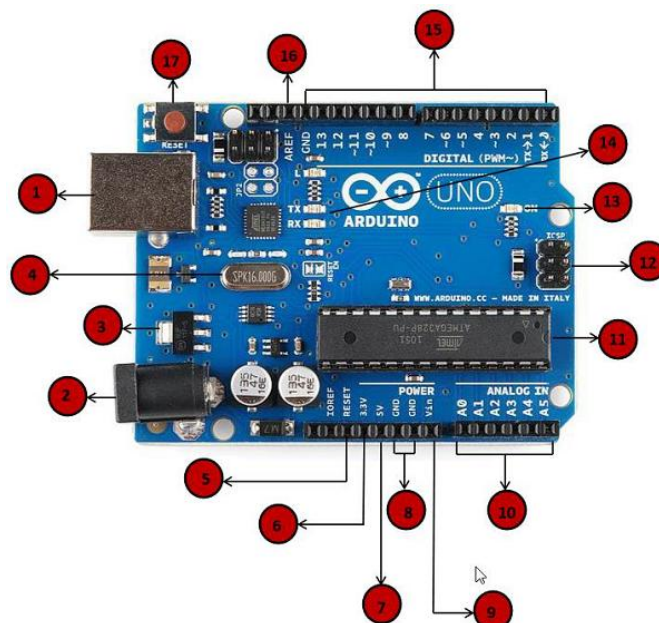
It can detect the level of moisture in the soil.






#### 20. Covid 19 patient monitoring system





The system can be used to measure the health conditions like body temperature, Oxygen level, heartbeat of covid19 patients.

**Explain PIN Diagram of Arduino UNO board in details.**

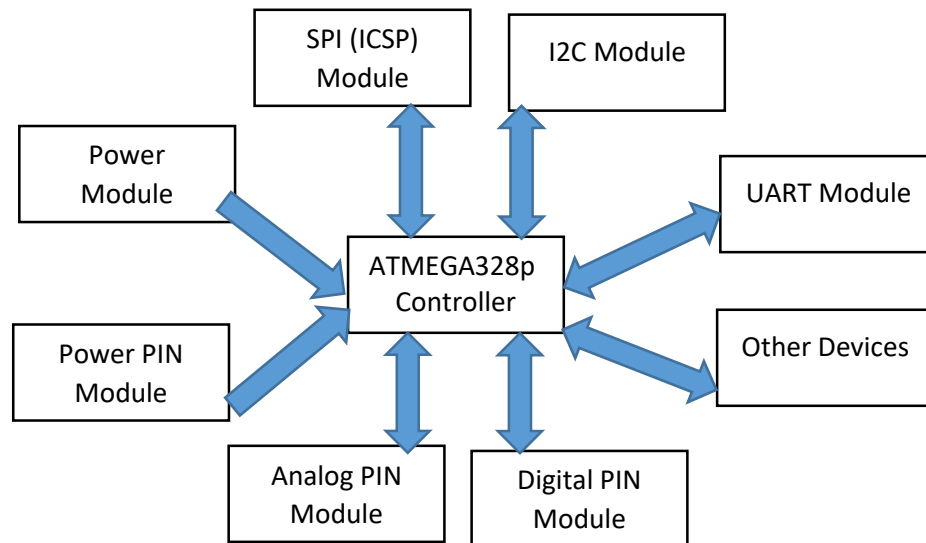
**Draw block diagram of Arduino UNO Board.**



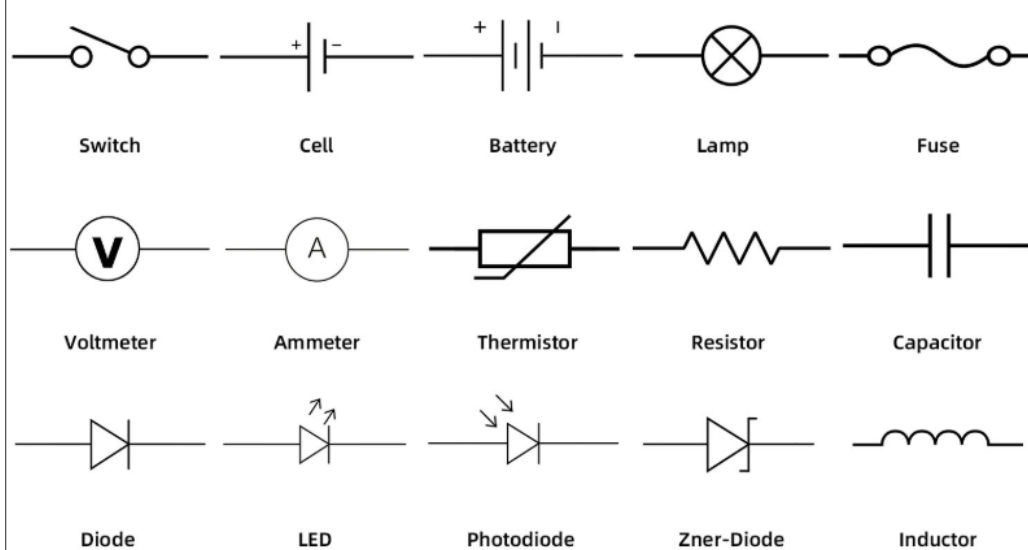
|   |   |
|---|---|
|    | <p><b>Power USB</b></p> <p>Arduino board can be powered by using the USB cable from your computer.</p>  |
|    | <p><b>Power (Barrel Jack)</b></p> <p>Arduino boards can be powered directly from the AC mains power supply by connecting it to the Barrel Jack (2).</p>   |
|    | <p><b>Voltage Regulator</b></p> <p>The function of the voltage regulator is to <u>control the voltage given to the Arduino board</u> and stabilize the DC voltages used by the processor and other elements.</p>  |
|  | <p><b>Crystal Oscillator</b></p> <p>The crystal oscillator helps Arduino in dealing with time issues. How does Arduino calculate time? The answer is, by using the crystal oscillator. The number printed on top of the Arduino crystal is 16.000H9H. It tells us that the frequency is 16,000,000 Hertz or 16 MHz.</p> <p><b>Time = 1/frequency or frequency = 1/time</b></p> <p>Frequency of a signal is number of cycles per second.</p> <p><b>1K = Kilo = <math>10^3=1000</math></b></p> <p><b>1M = Mega = <math>10^6=1000000</math></b></p> <p><b>1G = Giga = <math>10^9=1000000000</math></b></p> <p><b>1T = Tera = <math>10^{12} = 1000000000000</math></b></p> <p><b>1m = mili = <math>10^{-3} = 1/10^3=1/1000=0.001</math></b></p> <p><b>1micro = micro = <math>10^{-6}=0.000001</math></b></p> <p><b>1n = neno = <math>10^{-9} = 0.000000001</math></b></p> |
|  | <p><b>Arduino Reset</b></p> <p>You can reset your Arduino board, i.e., start your program from the beginning. You can reset the UNO board in two ways. First, by using the reset button (17) on the board. Second, you can connect an external reset button to the Arduino pin labelled RESET (5).</p>  |

|   |  |  |
|---|--|--|
|    | <p><b>Pins (3.3, 5, GND, Vin)</b></p> <ul style="list-style-type: none"> <li>• 3.3V (6) – Supply 3.3 output volt</li> <li>• 5V (7) – Supply 5 output volt</li> <li>• Most of the components used with Arduino board works fine with 3.3 volt and 5 volt.</li> <li>• GND (8)(Ground: 0v ) – There are several GND pins on the Arduino, any of which can be used to ground your circuit.</li> <li>• Vin (9) – This pin also can be used to power the Arduino board from an external power source, like AC mains power supply.</li> </ul> |  |
|    | <p><b>Analog pins</b></p> <p>The Arduino UNO board has six analog input pins A0 through A5. These pins can read the signal from an analog sensor like the humidity sensor or temperature sensor and convert it into a digital value that can be read by the microprocessor.</p>  |  |
|  | <p><b>Main microcontroller</b></p> <p>Each Arduino board has its own microcontroller (11). You can assume it as the brain of your board.</p> <p>The microcontrollers are usually of the ATMEL Company.</p> <p>You must know what IC your board has before loading up a new program from the Arduino IDE. This information is available on the top of the IC.</p> <p>For more details about the IC construction and functions, you can refer to the data sheet.</p>   |  |
|  | <p><b>ICSP pin</b></p> <p>ICSP: In Circuit Serial Programming</p> <p>Arduino board is connected with another arduino board or another controller using ICSP module.</p> <p>Mostly, ICSP (12) is an AVR, a tiny programming header for the Arduino consisting of (Master / slave)</p> <p><b>MOSI→ Master output Slave input→ Mater to Slave</b></p> <p><b>MISO→ Master input Slave output→ Slave to Master</b></p> <p><b>SCK→Serial clock</b></p> <p><b>RESET→ Reset</b></p>  |  |

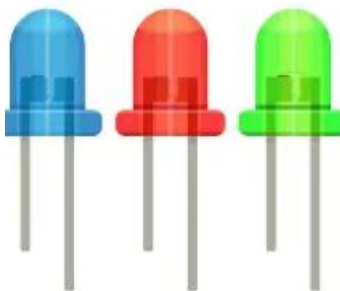
|   |  |
|---|--|
|   | <p><b>VCC→ +5V</b></p> <p><b>GND→ 0v</b></p> <p>It is often referred to as an SPI (Serial Peripheral Interface), which could be considered as an "expansion" of the output. Actually, you are slaving the output device to the master of the SPI bus.</p>  |
| 13  | <p><b>Power LED indicator</b></p> <p>This LED should light up when you plug your Arduino into a power source to indicate that your board is powered up correctly. If this light does not turn on, then there is something wrong with the connection.</p>   |
| 14  | <p><b>TX and RX LEDs</b></p> <p><b>Tx: transmitter</b></p> <p><b>Rx: Receiver</b></p> <p>On your board, you will find two labels: TX (transmit) and RX (receive). They appear in two places on the Arduino UNO board. First, at the digital pins 0 and 1, to indicate the pins responsible for serial communication. Second, the TX and RX led (13). The TX led flashes with different speed while sending the serial data. The speed of flashing depends on the baud rate used by the board. RX flashes during the receiving process.</p> |
| 15  | <p><b>Digital I/O</b></p> <p>The Arduino UNO board has 14 digital I/O pins (15)</p> <p>(of which 6 provide PWM (Pulse Width Modulation) output. These pins can be configured to work as input digital pins to read logic values (0 or 1) or as digital output pins to drive different modules like LEDs, relays, etc. The pins labeled “~” can be used to generate <u>PWM: Pulse width Moduation</u>.</p>  |
| 16  | <p><b>AREF</b></p> <p><b>AREF stands for Analog Reference</b>. It is sometimes, used to set an external reference voltage (between 0 and 5 Volts) as the upper limit for the analog input pins.</p>  |
| <p><b>Draw detailed block diagram of Arduino UNO Microcontroller.</b></p> |  |



### Various IoT components and devices used in Embedded system / IoT Applications.



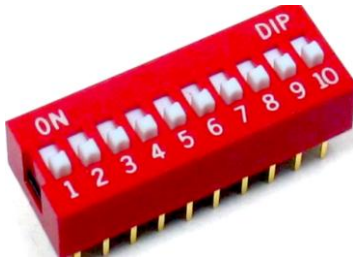
1. **LED:** light emitting diode / output device which can be digital or analog output device having two states ON and OFF. Two ends of LEDs are Anode (+) and Cathode (-). Long end represents Anode and short end is Cathode.





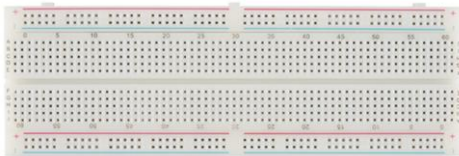
2. **Switch:**

input device having two states ON and OFF switches can be push button, ON/OFF, press once or DIP switches (Dual in Package).



3. **Bread board:**

device for making temporary connection



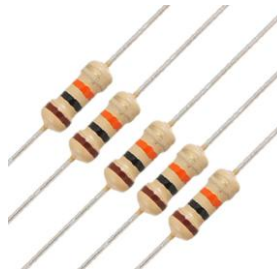
4. **Resistors:**

Passive devices which opposes flow of current

Passive devices are not able to generate the signal by itself.

(resistor, capacitor, inductor)

Active devices can able to generate the signal by itself (diode, transistor)



5. **Capacitors:**

passive device which is used for charging and discharging

when capacitor is connected in parallel with battery it will charge.

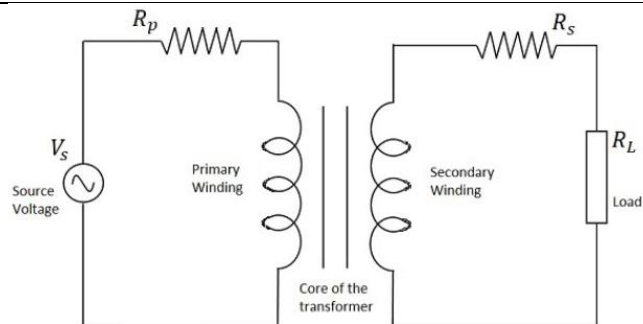
When charged capacitor is connected with load it will discharge.

6. **Inductors:**

passive device which is used for charging and discharging.

It will work with high voltages and power.

Transformer is made up with inductors and used for voltage conversion from high to low or low to high.



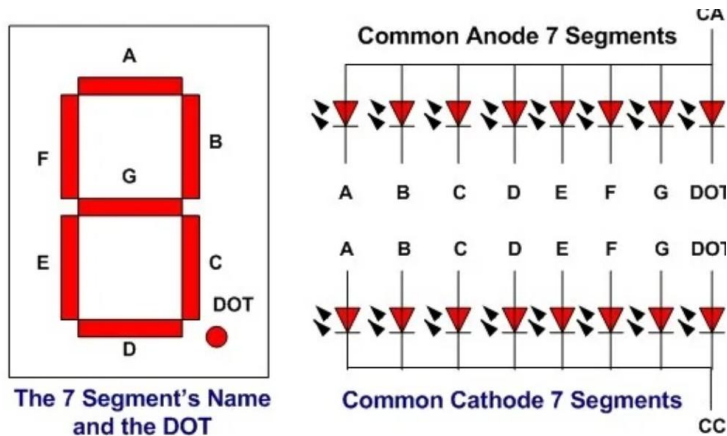
Primary winding of transformer gets charged when it is connected with voltage source. It will charge the secondary winding which is connected with load.

#### 7. 7 segment display:

output device to display alphabets and numbers

there are two configurations of 7 segment display

1. Common cathode (common with Ground)
2. Common anode (common with +5 V)



#### 8. LCD display: (Liquid Crystal Display)

output device to display alphabets and numbers and symbols.

There are two types of LCDs

1. General Purpose (16 pins)
2. I2C LCD



#### 9. Buzzer:

output devices with sound as output

Buzzer can be replaced with LED to give warnings in the embedded system.

#### 10. Sensors

(Temperature sensor, ultrasonic sensor): analog and digital inputs

A device which detects or measures a physical property and records it is sensor.

It is a device which measure changes in environmental conditions / parameters and forward it for further processing.

Transducers convert or transduce energy of one kind into another.

Sensor is a transducer (True /False) → True

Transducer is a sensor (True /False) → False

There are two types of sensors based on Output

1. Analog sensor (temp, pressure) → it produces analog output
2. Digital sensor (Motion, object detection) → it produces digital output

There are two more types of sensors based on Data

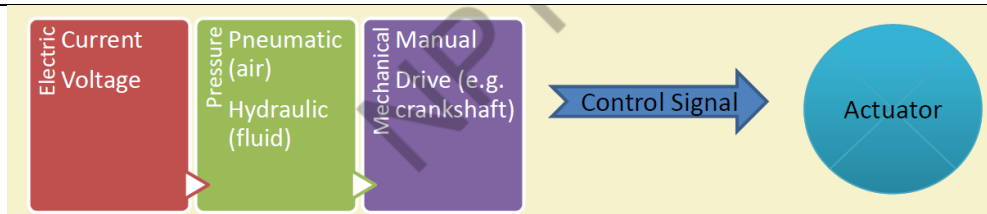
1. Scalar sensor (temp) → sensor output depends on values only. It doesn't depend on direction
2. Vector sensor ( sound, camera) → sensor output depends on both values and direction.

**Sensor resolution** is the smallest change that can be detected by sensor.

| Sensor Types |  |
|--------------|--|
| Light        | <ul style="list-style-type: none"><li>• Light Dependent resistor</li><li>• Photo-diode</li></ul>           |
| Temperature  | <ul style="list-style-type: none"><li>• Thermocouple</li><li>• Thermistor</li></ul>                        |
| Force        | <ul style="list-style-type: none"><li>• Strain gauge</li><li>• Pressure switch</li></ul>                   |
| Position     | <ul style="list-style-type: none"><li>• Potentiometer, Encoders</li><li>• Opto-coupler</li></ul>           |
| Speed        | <ul style="list-style-type: none"><li>• Reflective/ Opto-coupler</li><li>• Doppler effect sensor</li></ul> |
| Sound        | <ul style="list-style-type: none"><li>• Carbon Microphone</li><li>• Piezoelectric Crystal</li></ul>        |
| Chemical     | <ul style="list-style-type: none"><li>• Liquid Chemical sensor</li><li>• Gaseous chemical sensor</li></ul> |

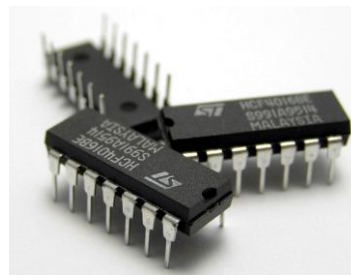
#### 11. Actuator:

- An actuator is a component of a machine or system that moves or controls the mechanism or the system.
- An actuator requires a **control signal** and a source of energy (**power signal**).
- Upon receiving a control signal, the actuator responds by converting the energy into mechanical motion.



- Various types of actuators are
  - Electric
  - Mechanical
  - Hydraulic
  - Pneumatic
  - Magnetic / thermal

12. **Microphone**: input device to capture sound and converts output in electrical form. (Transducer)
13. **Relay**: Power Converter / current convertor (High to low power conversion circuit)
14. **DC / AC motor**: it is an actuator which rotates based on input data.
15. **Servo / stepper motor**: it is an actuator which rotates based on input data.
16. **Integrated circuit (IC)**: it is a device which has multiple components integrated on it.



SSI: small scale Integration

MSI: Medium scale Integration

LSI: large scale Integration

**VLSI: very large scale Integration**

17. **Fuse**: it is used to protect from high voltage or power.  
It is a wire between main supply and device which protects device from fluctuation of voltages and power.
18. **connecting wires** (Jumper wire): for connections in the circuit (shielding)  
three types of wires are possible
  1. male to male
  2. male to female
  3. female to female

19. **Multi meter**: It is a device to measure AC and DC voltage, current, resistor, connectivity, etc. It can have two wires with tips.
20. **Soldering iron**: It is a device to make permanent connection between two wires or two end of devices. Dry solder is a situation where soldering is not done properly and it must be removed.
21. **Battery**: It is a DC supply used in electronics or electrical circuit to supply voltage / power to electronics device
22. **Potentiometer**: It is a variable resistor which can be used to vary the input resistance as per requirements. For example, 10k potentiometer can vary the value of resistor between 0 to 10 Kohm.
23. **Keypad**: Device to take numbers, alphabets or symbols as input
24. **Touchpad**: It is used to sense the touch input from the user. (capacitive touch)
25. **Speaker**: output device for sound (Transducer)
26. **Camera**: input device to capture the images.

**If a color code printed on resistor is Brown, Black, Red, Gold then find value of resistor.**

**BB ROY GOES TO BOMBAY VIA GATE WAY OF INDIA**

| Color Code | Color Name    | Digit value | Multiplier   |
|------------|---------------|-------------|--------------|
| 0          | Black         | 0           | $10^0$       |
| 1          | Brown         | 1           | $10^1$       |
| 2          | Red           | 2           | $10^2$       |
| 3          | Orange        | 3           | $10^3$       |
| 4          | Yellow        | 4           | $10^4$       |
| 5          | Green         | 5           | $10^5$       |
| 6          | Blue          | 6           | $10^6$       |
| 7          | Violet        | 7           | $10^7$       |
| 8          | Gray          | 8           | $10^8$       |
| 9          | white         | 9           | $10^9$       |
|            | Gold / silver | Gold = 5%   | Silver = 10% |

Resistor value =  $(10 \times 10^2) = 1000 \text{ ohm} = 10^3 = 1 \text{ Kohm}$

Lower Range =  $1K - (1 \times 0.05) = 1K - 0.05 = 0.95 \text{ K}$

Upper Range =  $1K + (1 \times 0.05) = 1K + 0.05 = 1.05 \text{ K}$

Range

**If a color code printed on resistor is Red, Black, orange, Gold then find value / Range of resistor.**

Value of resistor =  $20 \times 10^3 = 20 \text{ Kohm}$

Lower range =  $20\text{K} - (20\text{K} \times 0.05) = 20\text{K} - 1\text{K} = 19\text{Kohm}$

Upper range =  $20\text{K} + (20\text{K} \times 0.05) = 20\text{K} + 1\text{K} = 21\text{Kohm}$

**If a color code printed on resistor is yellow, orange, red, silver then find value of resistor in Kohms.**

Value of resistor =  $43 \times 10^2 = 4.3 \text{ Kohm}$

Lower range =  $4.3\text{K} - (4.3\text{K} \times 0.1) = 4.3\text{K} - 0.43\text{K} = 3.87 \text{ Kohm}$

Upper range =  $4.3\text{K} + (4.3\text{K} \times 0.1) = 4.3\text{K} + 0.43\text{K} = 4.73 \text{ Kohm}$

**What will be the color code printed on resistor if its value is 2.2 Kohm?**

Value =  $2.2 \times 10^3 = 22 \times 10^2$

2: Red color

2: Red Color

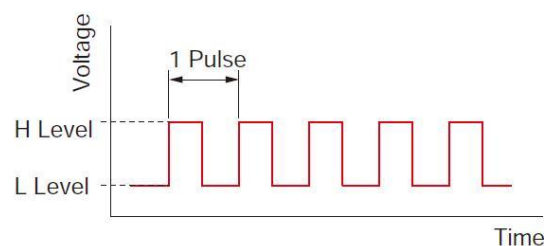
$10^2$  = Red color

So color code printed on resistor = red, red, red

### **Pulse Width Modulation**

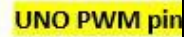
#### **PWM: Pulse Width Modulation**

PWM is a technique where the width of pulse is changing according to input value. Means by passing different values user can control pulse width.





These pins are 3, 5, 6, 9, 10, 11.



## Arduino IDE (Integrated Development Environment)

- Download latest version of arduino IDE from [arduino.cc](https://www.arduino.cc) website.
- Install arduino IDE.



- Install require libraries to run the program.
- Refer the documentation.
- Write an arduino program using embedded C.
- **Arduino program is known as “sketch”.**
- Tick mark tool is used to compile arduino program. (Verify)
- Right arrow is used to upload the program on arduino controller. (Upload)
- Refer inbuilt examples to run the codes.
- Open serial monitor to view the outputs.

### Embedded C / Arduino data types.

Arduino datatypes are used define the type of data used as input or output.

1. void() → it returns null
2. char → it defines characters.
3. int → it defines integer numbers
4. float → it defines floating numbers
5. byte → it defines variable byte with size 8 bits
6. boolean → it defines Boolean type of data
7. const → it defines constant
8. string → it defines string( group of characters)
9. Arr → it defines array
10. unsigned int → it defines unsigned integer number
11. unsigned char → it defines unsigned character variable
12. word → it defines word
13. long → it defines numbers with larger size
14. short → it defines short size number
15. unsigned long → it defines long number without sign



16. unsigned short → it defines short number without sign

17. double → it defines number with higher size

### Embedded C arduino variables.

Arduino variables are of two types.

1. Local variable
2. Global variable

#### Example:

int led=13; → global variable

void loop()

{

int a; → local variable

--

}

Led → global variable

a → local variable

### Embedded C arduino operators.

#### 1. Arithmetic operators (+, -, \*, /, %)

+ → addition

- → Subtraction

\* → multiplication

/ → division

% → remainder

Arduino based Calculator is an application which demonstrate arithmetic operators.

```
double calculate(char operation, double left, double right) {  
    switch (operation) {  
        case '+': return left + right;  
        case '-': return left - right;  
        case '*': return left * right;  
        case '/': return left / right;  
    }  
}
```

#### 2. Comparison operators (==, >, <, !=, <=, >=, .....

== → equal to operator

> → greater than

< → less than

>= → greater than or equal to

<= → less than or equal to

Sensor based output controls is the example of comparison operators in embedded C program.

```
if (tmpCel ==25.0)
{
    digitalWrite(2,LOW);
    digitalWrite(3,LOW);
    digitalWrite(4,HIGH);
}
Else if (tmpCel >25.0)
{
    digitalWrite(2,LOW);
    digitalWrite(3,HIGH);
    digitalWrite(4,HIGH);
}
Else
{
    digitalWrite(2,HIGH);
    digitalWrite(3,HIGH);
    digitalWrite(4,HIGH);
}
```

### **3. Boolean operators (&&, ||, !)**

&& → logical AND

|| → logical OR

! → logical NOT

```
If(s1=='ON' && s2=='ON')
{
    digitalWrite(led, HIGH);
}
Else
{
    digitalWrite(led, LOW);
}
```

### **4. Bitwise operator (&, |,....)**

& → bitwise AND

| → bitwise OR

Perform bitwise AND and OR operations of following numbers

A=11001100

B=11111001

And=11001000

Or=11111101

## 5. Compound operators (++ , -- , += , -= , \*= , /= .....)

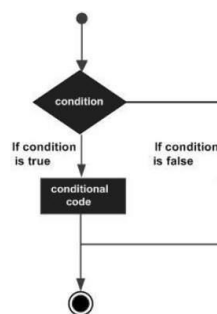
++ → increment

-- → decrement

## **Arduino (Embedded C) control statements.**

- Control statements are used to **control the part of program based on condition or multiple conditions**. These are,

1. if ---else statement
2. switch case statement



(Control statement flow chart)

### 1 If statement

#### **Syntax:**

```
If(condition)
{
}
```

It takes an expression in parenthesis and a statement or block of statements. If the expression is true then the statement or block of statements gets executed otherwise these statements are skipped.

```
If (pot >300)
{
digitalWrite(LED, HIGH);
}
```

### 2 If ...else statement

#### **Syntax:**

```
If(condition)
{true part
}
else
{false part
}
```

An if statement can be followed by an optional else statement, which executes when the expression is false.

```
If (pot >300)
{
digitalWrite(LED, HIGH);
}
else
{
digitalWrite(LED, LOW);
}
```

### **3 If...else if ...else statement**

#### **Syntax:**

```
If(condition)
{
}
elseif (condition)
{
}
Else
{
}
```

The if statement can be followed by an optional else if...else statement, which is very useful to test various conditions using single if...else if statement.

```
If (pot >300)
{
digitalWrite(LED1, HIGH);
digitalWrite(LED2, HIGH);
}
elseif (pot >200)
{
digitalWrite(LED1, LOW);
digitalWrite(LED2, HIGH);
}
else
{
digitalWrite(LED1, LOW);
digitalWrite(LED2, LOW);
}
```

### **4 switch case statement**

#### **Syntax:**

```
switch (variable)

    case label: {
        // statements
        break;
    }
```

```

case label 1: {
    // statements
    break;
}
default: {
    // statements
    break;
}

```

```

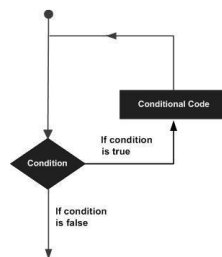
if (key == 'A' || key == 'B' || key == 'C' || key == 'D') {
    op = key;
    switch(op) {
    case 'A': lcd.print(num1 + num2); break;
    case 'B': lcd.print(num1 - num2); break;
    case 'C': lcd.print(num1 * num2); break;
    case 'D': lcd.print(num1 / num2); break;
    }
}

```

Similar to the if statements, switch...case controls the flow of programs by allowing the programmers to specify different codes that should be executed in various conditions.

### looping structure used in arduino programming (Embedded C).

Loop statements are used to execute a statement or group of statements multiple times based on the conditions.



Five different Loops can be created using three different structures.

1. while loop
2. for loop
3. do while loop
4. Nested loop
5. Infinite loop

#### 1 while loop

##### **Syntax:**

```

while (condition)
{
    Statements to be executed when condition is true
}

```

```
X=10
while(x>5)
{
digitalWrite(LED, HIGH);
delay(1000);
digitalWrite(LED, LOW);
delay(1000);
x--;
}
```

while loops will loop continuously, and infinitely, until the expression inside the parenthesis, () becomes false. Something must change the tested variable, or the while loop will never exit.

**While loop is executed at least once.(True/false)**

False.

## **2 do...while loop**

**Syntax:**

```
do
{
Statements to be executed
}
while (condition)
```

```
X=10
do
{
digitalWrite(LED, HIGH);
delay(1000);
digitalWrite(LED, LOW);
delay(1000);
x--;
}
while(x>5)
```

The do...while loop is similar to the while loop. In the while loop, the loop-continuation condition is tested at the beginning of the loop before performed the body of the loop.

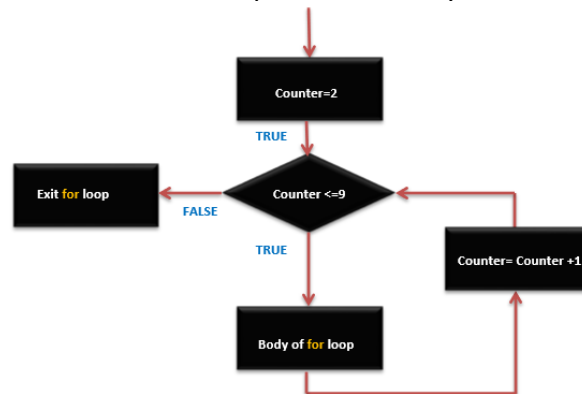
## **3. For Loop:**

**Syntax:**

```
for (initialization; control or condition; increment or decrement)
{
Statements
}
```

```
For(x=0;x<=255;x++)
{
analogWrite(PWMLED, x);
delay(1000);
}
```

A for loop executes statements a predetermined number of times. The control expression for the loop is initialized, tested and manipulated entirely within the for loop parentheses.



#### **4. Nested Loop**

**Syntax:**

```
for ( initialize ;control; increment or decrement)
{ // statement block
  for ( initialize ;control; increment or decrement)
  { // statement block
  }
}
```

C language allows you to use one loop inside another loop. The following example illustrates the concept.

```
For(x=0;x<=255;x++)
{
  analogWrite(PWMLED, x);
  delay(1000);
  if(x==255)
  {
    For(x=255;x>=0;x--)
    {
      analogWrite(PWMLED, x);
      delay(1000);
    }
  }
}
```

#### **5. Infinite loop (How to create infinite loop?)**

**Using while:**

```
while(1)
{
  Statements
}
```

It is the loop having no terminating condition, so the loop becomes infinite.

**Using Do----while:**

```
do
{
Statements
}
while(1)
```

It is the loop having no terminating condition, so the loop becomes infinite.

**Using For loop:**

```
For (; ; )
{
statements
}
```

It is the loop having no terminating condition, so the loop becomes infinite.

**Arduino Programming functions used in embedded C.**

| Sno | Name           | Description or syntax of the function   |
|-----|----------------|---|
| 1   | pinMode()      | <ul style="list-style-type: none"><li>• pinMode(pin, Mode)</li><li>• it is used to specify whether the pin used in interfacing is INPUT or OUTPUT.</li><li>• pinMode(13, INPUT) means pin 13 will acts as input pin.</li><li>• pinMode(6, OUTPUT) means pin 6 will acts as output pin.</li><li>• Int a=5<br/>pinMode(a, INPUT)</li><li>• It refers to digital input and output.</li></ul> |
| 2   | digitalRead()  | <ul style="list-style-type: none"><li>• digitalRead(Pin)</li><li>• Reads the value from a specified digital pin, either HIGH or LOW.</li><li>• digitalRead(13) means it reads the digital input from pin 13.</li><li>• Int A=7<br/>digitalRead(a) means it reads from pin 7.</li><li>• This is actually an input fuction.</li></ul>   |
| 3   | digitalWrite() | <ul style="list-style-type: none"><li>• digitalWrite(pin, value)</li><li>• it writes the value high or low to specified Pin.</li><li>• digitalWrite(13, HIGH) means it will write HIGH value of pin 13</li><li>• this is actually an output function.</li><li>• Int A=7</li></ul>   |



|      |               |  |     |     |     |     |    |    |    |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |      |   |   |   |   |   |   |   |   |   |   |
|------|---------------|--|-----|-----|-----|-----|----|----|----|---|---|---|---|--|---|---|---|---|---|---|---|---|---|---|------|---|---|---|---|---|---|---|---|---|---|
|      |               | digitalWrite(A, HIGH) means it writes HIGH value on pin 7.   |     |     |     |     |    |    |    |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |      |   |   |   |   |   |   |   |   |   |   |
| 4    | analogRead()  | <ul style="list-style-type: none"><li>analogRead(Pin)</li><li>Reads the value from the specified analog pin.</li><li>analogRead(A0) means it reads analog value from pin A0.</li><li>Int apin=A0<br/>analogRead(apin) means it reads variable (apin) value.</li></ul> <table><tr><td></td><td>512</td><td>256</td><td>128</td><td>64</td><td>32</td><td>16</td><td>8</td><td>4</td><td>2</td><td>1</td></tr><tr><td></td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>1023</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td></tr></table> |     | 512 | 256 | 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1023 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
|      | 512           | 256  | 128 | 64  | 32  | 16  | 8  | 4  | 2  | 1 |   |   |   |  |   |   |   |   |   |   |   |   |   |   |      |   |   |   |   |   |   |   |   |   |   |
|      | 0             | 0  | 0   | 0   | 0   | 0   | 0  | 0  | 0  | 0 |   |   |   |  |   |   |   |   |   |   |   |   |   |   |      |   |   |   |   |   |   |   |   |   |   |
| 1023 | 1             | 1  | 1   | 1   | 1   | 1   | 1  | 1  | 1  | 1 |   |   |   |  |   |   |   |   |   |   |   |   |   |   |      |   |   |   |   |   |   |   |   |   |   |
| 5    | analogWrite() | <ul style="list-style-type: none"><li>analogWrite(pin, value)</li><li>it basically writes value on specified output pin.</li><li>Int analogpin=3 (connect potentiometer with pin3 to get analog input)</li><li>val = analogRead(analogPin); analogWrite(ledPin, val / 4);</li></ul>  |     |     |     |     |    |    |    |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |      |   |   |   |   |   |   |   |   |   |   |
| 6    | max()         | <ul style="list-style-type: none"><li>max(x,y)</li><li>it finds maximum out of two numbers.</li></ul>  |     |     |     |     |    |    |    |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |      |   |   |   |   |   |   |   |   |   |   |
| 7    | min()         | <ul style="list-style-type: none"><li>min(x,y)</li><li>it finds minimum out of two numbers.</li></ul>  |     |     |     |     |    |    |    |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |      |   |   |   |   |   |   |   |   |   |   |
| 8    | abs()         | <ul style="list-style-type: none"><li>Calculates the absolute value of a number.</li><li>Abs(x)</li></ul>  |     |     |     |     |    |    |    |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |      |   |   |   |   |   |   |   |   |   |   |
| 9    | sq()          | <ul style="list-style-type: none"><li>It finds square of a given number</li><li>Sq(x)</li></ul>  |     |     |     |     |    |    |    |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |      |   |   |   |   |   |   |   |   |   |   |
| 10   | sqrt()        | <ul style="list-style-type: none"><li>It finds square root of a given number</li><li>Sqrt(x)</li></ul>   |     |     |     |     |    |    |    |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |      |   |   |   |   |   |   |   |   |   |   |
| 11   | cos()         | <ul style="list-style-type: none"><li>It finds cosine value of a given number</li></ul>  |     |     |     |     |    |    |    |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |      |   |   |   |   |   |   |   |   |   |   |
| 12   | sin()         | <ul style="list-style-type: none"><li>It finds sine value of a given number</li></ul>  |     |     |     |     |    |    |    |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |      |   |   |   |   |   |   |   |   |   |   |
| 13   | tan()         | <ul style="list-style-type: none"><li>It finds tan value of a given number</li></ul>   |     |     |     |     |    |    |    |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |      |   |   |   |   |   |   |   |   |   |   |
| 14   | map() (IMP)   | <p>It re-map the range.</p> <p><b>map(value, fromLow, fromHigh, toLow, toHigh)</b><br/>value→variable<br/>fromlow→lower value of current range<br/>fromhigh→higher value of current range<br/>tolow→lower value of new range<br/>tohigh→higher value of new range</p> <p><b>map(x, 1, 50, 50, 1) →</b><br/>x = 5 then what is new value of x ? →x = 46</p> <p><b>map(x, 1,100, 1,1000) →</b><br/>x = 50 then what is new value of x ? →x = 500</p>   |     |     |     |     |    |    |    |   |   |   |   |  |   |   |   |   |   |   |   |   |   |   |      |   |   |   |   |   |   |   |   |   |   |

|  |  |   |
|--|--|---|
|  |  | <p>Map is used to convert potentiometer value (0 to 1023) into servomotor rotation (0 to 180 degree).</p> <pre>map(x, 0, 1023, 0, 180);</pre> <p>it maps value x which is in the range 1 to 50 to 50 to 1</p> <pre>x = analogRead(A0); y = map(x, 0, 1023, 0, 180); myServo.write(y);</pre> |
|--|--|---|

### Time related function used in arduino programming.

#### 1. delay():

Pauses the program for the amount of time (in milliseconds) specified as parameter. (There are 1000 milliseconds in a second.)

delay(500): it delays or pauses for 500 milliseconds=500/1000=0.5 sec

delay(100): it delays or pauses for 100 milliseconds=100/1000=0.1 sec

```
void loop()
{
  digitalWrite(ledPin, HIGH); // sets the LED on
  delay(1000);                // waits for a 1 second (1000 milliseconds)
  digitalWrite(ledPin, LOW);  // sets the LED off
  delay(1000);                // waits for a 1 second (1000 milliseconds)
}
```

#### 2. delayMicroseconds():

Pauses the program for the amount of time (in microseconds) specified as parameter. (There are 1000 microseconds in a millisecond and there are  $10^6$  microseconds in 1 second.)

```
void loop()
{
  digitalWrite(outPin, HIGH); // sets the pin on
  delayMicroseconds(50);      // pauses for 50 microseconds
  digitalWrite(outPin, LOW);  // sets the pin off
  delayMicroseconds(50);      // pauses for 50 microseconds
}
```

#### 3. micros():

Returns the number of microseconds since the Arduino board began running the current program. This number will overflow (go back to zero), after approximately 70 minutes.

```
void loop()
{
  Serial.print("Time: ");
```

```
time = micros();
Serial.println(time); //prints time since program started
delay(1000);        // wait a second so as not to send massive amounts of data
}
```

#### 4. millis():

Returns the number of milliseconds since the Arduino board began running the current program. This number will overflow (go back to zero), after approximately 50 days

```
void loop()
{
  Serial.print("Time: ");
  time = millis();
  Serial.println(time); //prints time since program started
  delay(1000);        // wait a second so as not to send massive amounts of data
}
```

### Various Constants used in Arduino Programming.

Following is the list of arduino constants.

| Constants | Meaning / Uses   |
|-----------|--|
| INPUT     | This constant is used to declare specified pin as input.<br>pinMode(b, INPUT) : b pin is working as an input<br>It is used to declare sensor or any other input devices connected with Arduino as an input.                                |
| OUTPUT    | This constant is used to declare specified pin as output.<br>pinMode(a, OUTPUT): a pin is working as an output<br>It is used to declare LCD, 7-segment LED or any other output devices connected with Arduino as an output.                |
| HIGH      | It is used to assign high value (+5v) to given variable or pin.<br>digitalWrite(LED, HIGH): it will send high value to an LED. (LED is ON)<br>it can be used to send HIGH value to make output devices such as dc motor or servomotor ON.  |
| LOW       | It is used to assign low value (+0v) to given variable or pin.<br>digitalWrite(LED, LOW): it will send low value to an LED.<br>(LED is OFF)<br>it can be used to send LOW value to make output devices such as dc motor or servomotor OFF. |
| TRUE      | True is defined as '1'. It is correct<br>It can be used with Arduino looping structure such as while and for loop.   |

|             |   |
|-------------|---|
|             | While(TRUE) or while(1) can be used to run the loop continuously or infinite loop.  |
| FALSE       | False is defined as '0'.it is incorrect<br>It can be used to compare the Boolean condition when it FALSE.<br>If(condition==FALSE)<br>{<br>}<br>}                      |
| LED_BUILTIN | It refers to built in LED on arduino board.<br>Basically inbuilt is connected with Pin 13.<br><br>digitalwrite(LED_BUILTIN, HIGH)→ it will glow the on board LED (13) |

### Functions used with arduino programming.

1. pinMode()
2. digitalRead()
3. digitalWrite()
4. analogRead()
5. analogWrite()

#### 1. pinMode()

- **pinMode(pin, Mode)**
- it is used to specify whether the pin used in interfacing is INPUT or OUTPUT.
- pinMode(13, INPUT) means pin 13 will acts as input pin.
- pinMode(6, OUTPUT) means pin 6 will acts as output pin.
- Int a=5  
pinMode(a, INPUT)
- It refers to digital input and output.

#### 2. digitalRead()

- **digitalRead(Pin)**
- Reads the value from a specified digital pin, either HIGH or LOW.
- digitalRead(13) means it reads the digital input from pin 13.
- Int a=7  
digitalRead(a) means it reads from pin 7.
- This is actually an input function.

#### 3. digitalWrite()

- **digitalWrite(pin, value)**
- It writes the value high or low to specified Pin.
- digitalWrite(13, HIGH) means it will write HIGH value of pin 13
- This is actually an output function.
- Int a=7  
digitalWrite(a, HIGH) means it writes HIGH value on pin 7.

#### **4. analogRead()**

- **analogRead(Pin)**
- Reads the value from the specified analog pin.
- analogRead(A0) means it reads analog value from pin A0.
- Int apin=A0  
analogRead(apin) means it reads variable (apin) value.

#### **5. analogWrite()**

- **analogWrite(pin, value)**
- it basically writes value on specified output pin.
- Int analogpin=3  
(connect potentiometer with pin3 to get analog input)  
val = analogRead(analogPin);  
analogWrite(ledPin, val / 4)