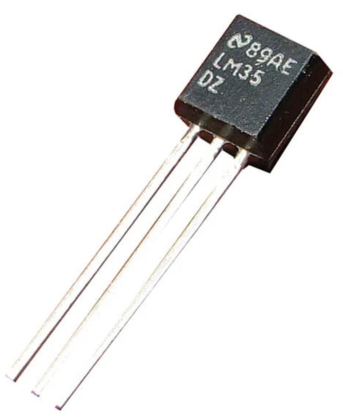
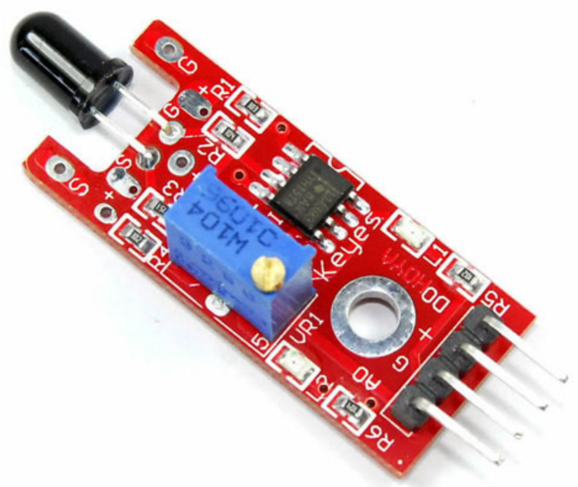
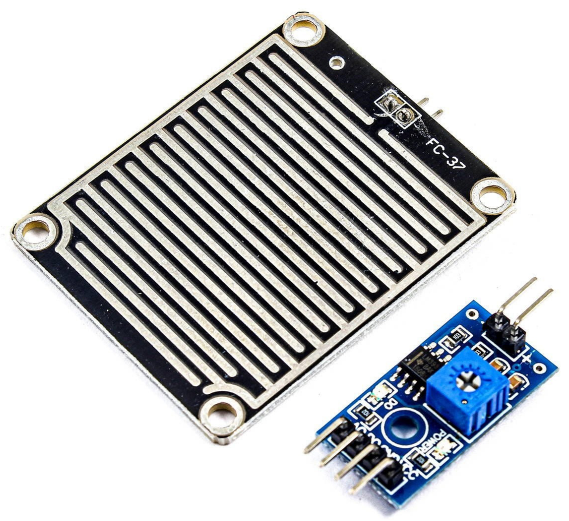
Basic Control System Components

Figure- Sensor and actuator in a system

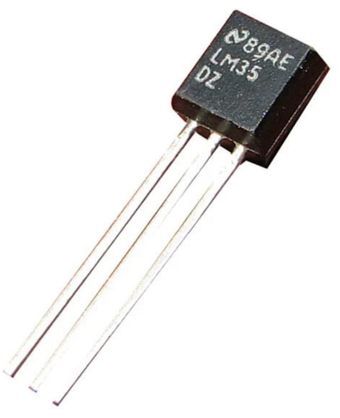
Sensors

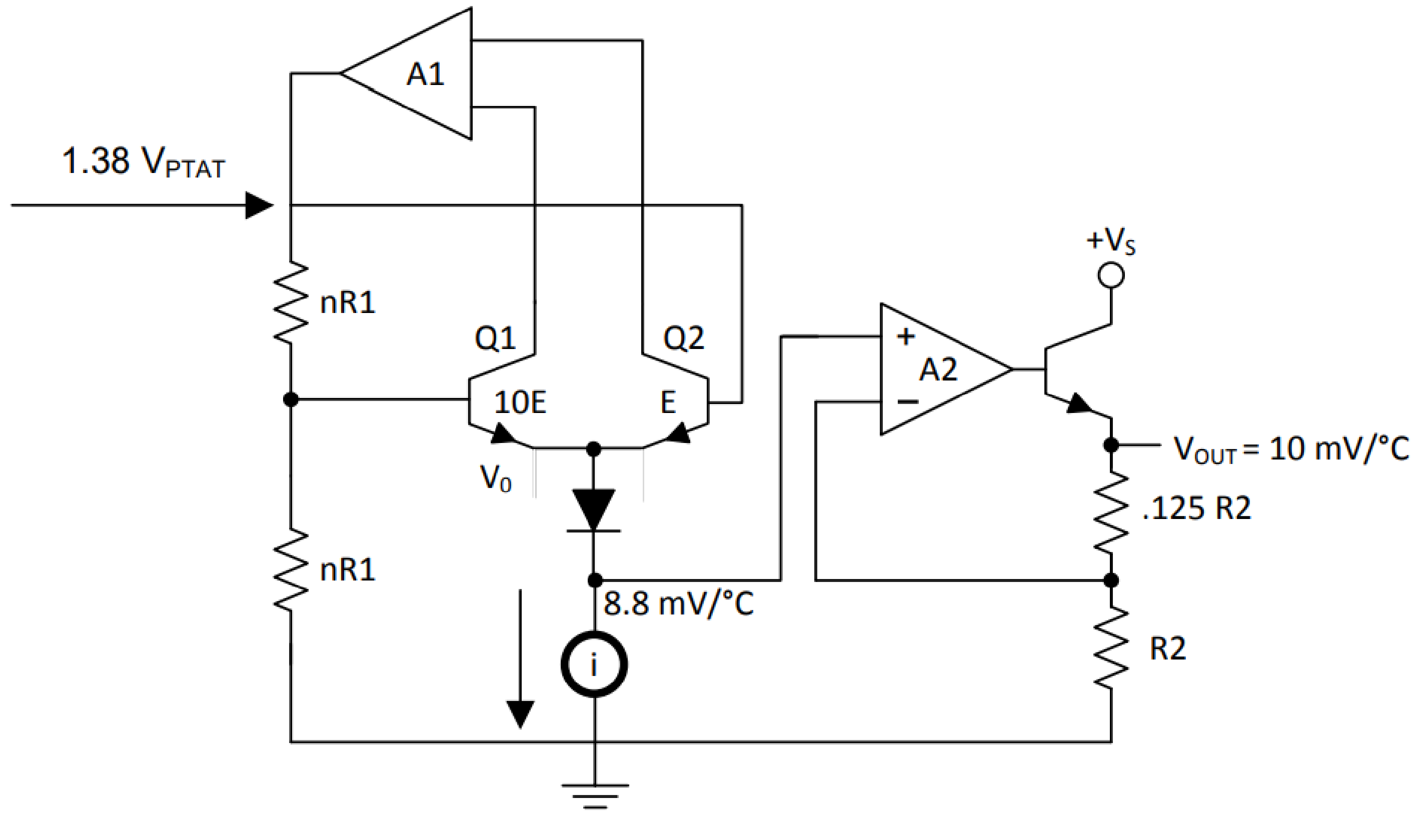






lm35 temp sensor





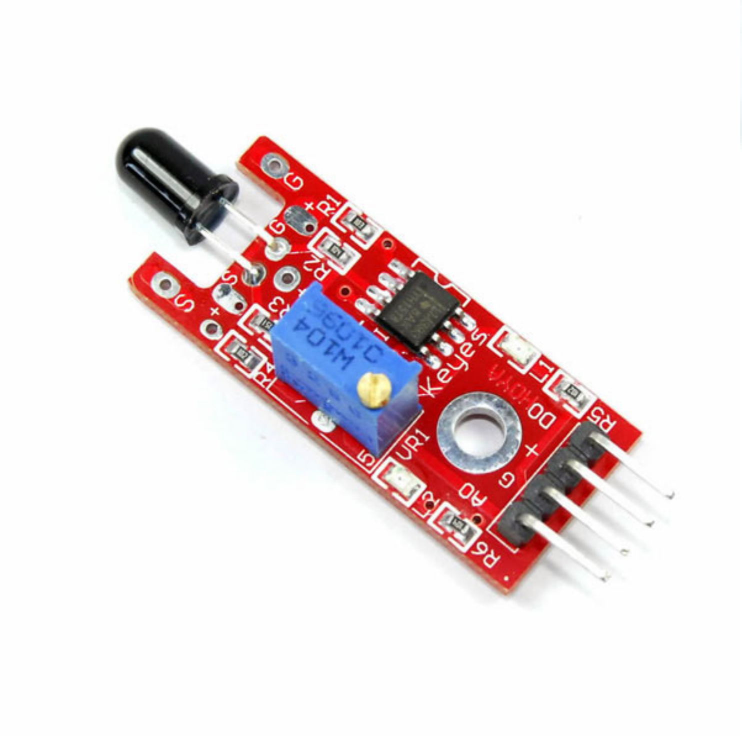
For every 1°C rise in temperature, the output voltage increases by 10 millivolts (mV).

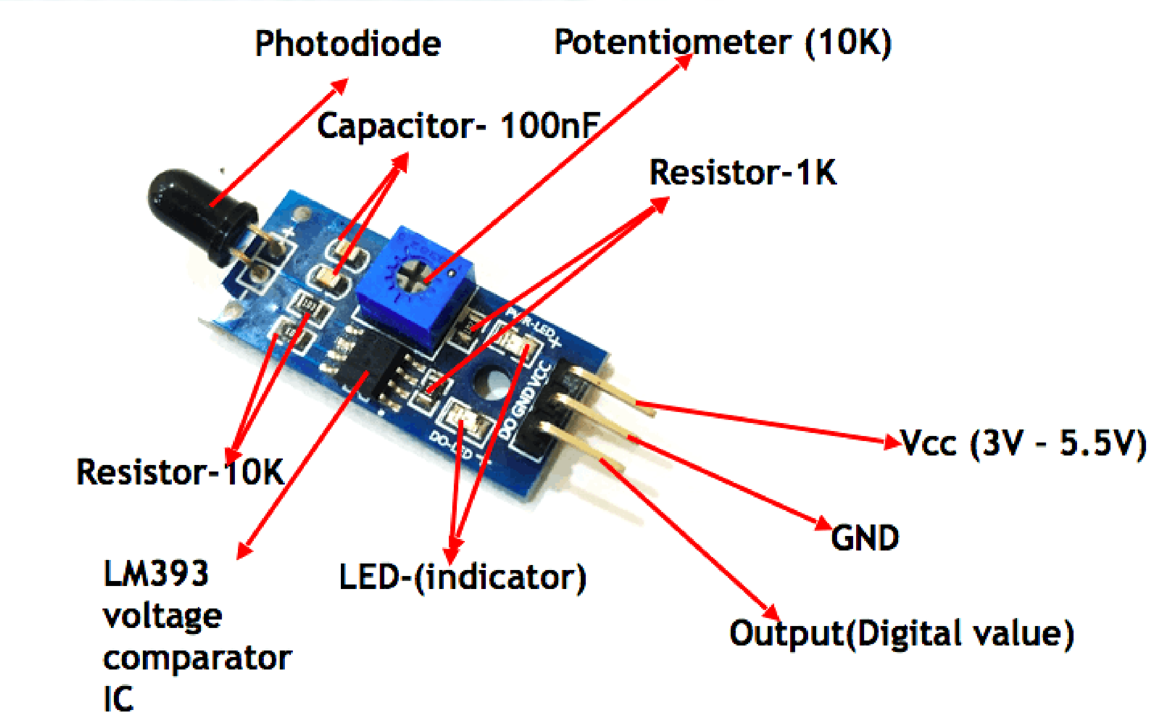
The voltage across this diode changes with temperature in a predictable way due to the temperature dependence of the semiconductor material's characteristics.(there's a silicon-based diode or transistor)

The small voltage generated by the temperature-sensitive circuit is amplified to produce a more usable output.

High Accuracy: ±0.5°C at room temperature.
Wide Operating Range: -55°C to +150°C.
Linear Output: Makes it easy to calculate temperature directly.
Low Power Consumption: Ideal for battery-operated devices.

Flame Sensor





Infrared flame sensors are sensitive to wavelengths around 760 nm to 1100 nm, commonly emitted by flames.

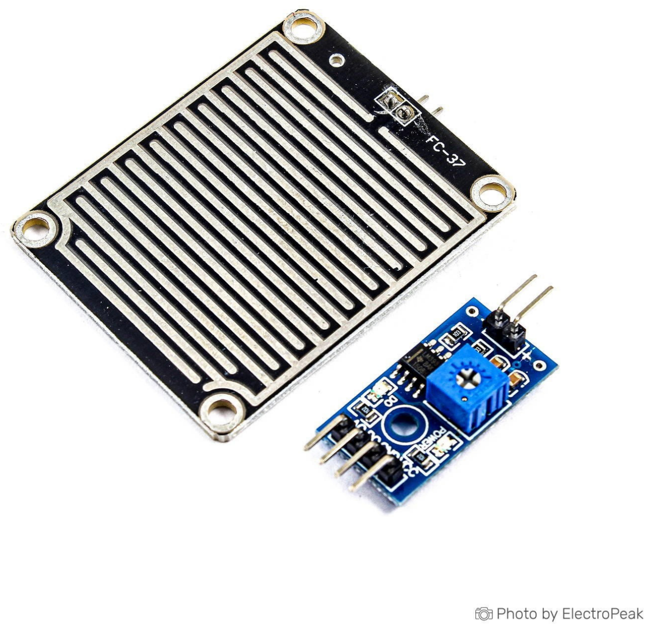
The light detected by the sensor is converted into an electrical signal. This signal is processed and compared to a threshold value to determine if a flame is present.

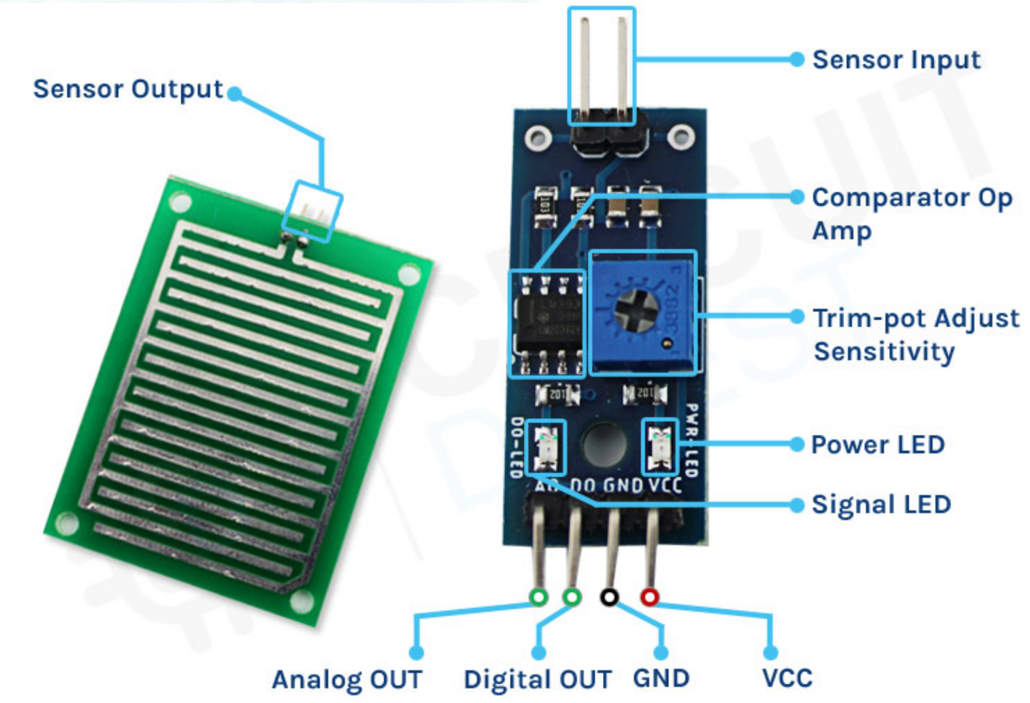
If the intensity of the detected light exceeds the threshold, the sensor outputs a HIGH signal (digital logic 1).

The amplified signal is compared to a pre-set threshold voltage. If the signal exceeds this threshold, it indicates the presence of a flame.

High sensitivity to flames.
Fast response time.
Reliable in detecting specific flame wavelengths.
Compact and easy to integrate with microcontrollers.

Rain Sensor





The presence of water alters an electrical signal, which is then processed to determine the occurrence and sometimes the intensity of rain.

The rain sensor has a grid of conductive traces printed on a board

When water droplets bridge the conductive traces, the resistance across the grid changes. This change is measured to detect the presence of rain.

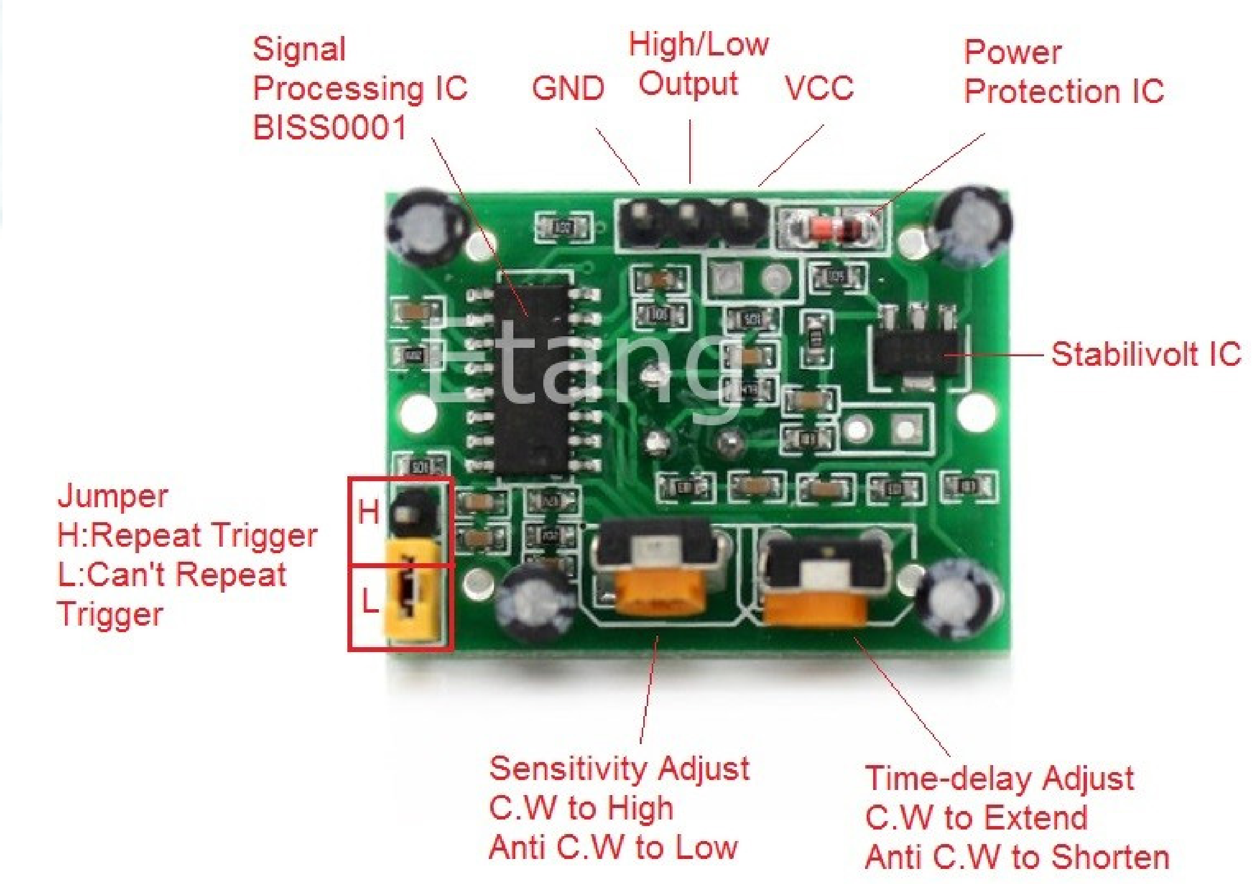
Sensing Plate: A PCB with interlaced conductive traces.

Signal Amplifier: Amplifies the small changes in resistance for processing.

Comparator: Compares the signal to a threshold to output a digital signal.

Output Interface: Provides digital or analog signals for interfacing.

PIR Sensor



The PIR sensor detects these IR changes when an object enters or moves within its detection range.

The core of a PIR sensor is a pyroelectric material that generates a small electrical signal when it detects changes in infrared radiation.

The small electrical signal from the pyroelectric material is amplified by an operational amplifier to make it usable.

The amplified signal is fed into a comparator circuit, which determines if the signal exceeds a preset threshold. If it does, the sensor outputs a HIGH signal to indicate motion.

PIR sensors have additional circuitry to control parameters like sensitivity and delay time (duration for which the output remains HIGH after detecting motion).

Energy Efficient: Consumes very little power.
Reliable: Detects motion without false triggers from static objects.
Low Cost: Affordable for most applications.
Non-Intrusive: Detects motion passively, without emitting any signals.

LDR Sensor

The resistance of an LDR decreases with an increase in light intensity (photoconductivity).

In darkness or low light, the resistance is very high (in the megohms range).

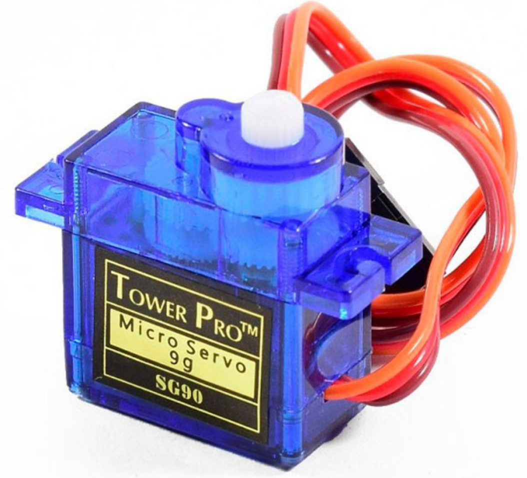
In bright light, the resistance drops significantly (a few hundred ohms).

When light photons strike the surface of the LDR, they provide energy to the electrons in the material, allowing them to move into the conduction band. This increases the conductivity and decreases the resistance.

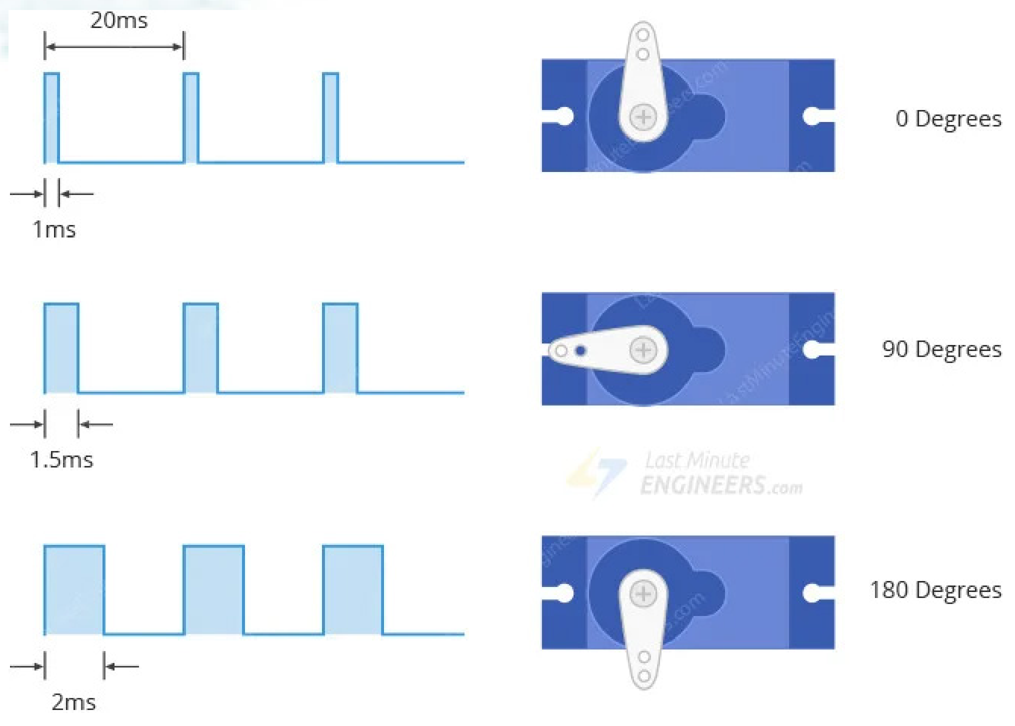
The LDR's resistance change is typically converted into a voltage signal using a voltage divider circuit or an ADC (analog-to-digital converter) for microcontrollers.

Simple and cost-effective.
Low power consumption.
Works well in a wide range of applications.

Actuators



Sg90 Servo Motor

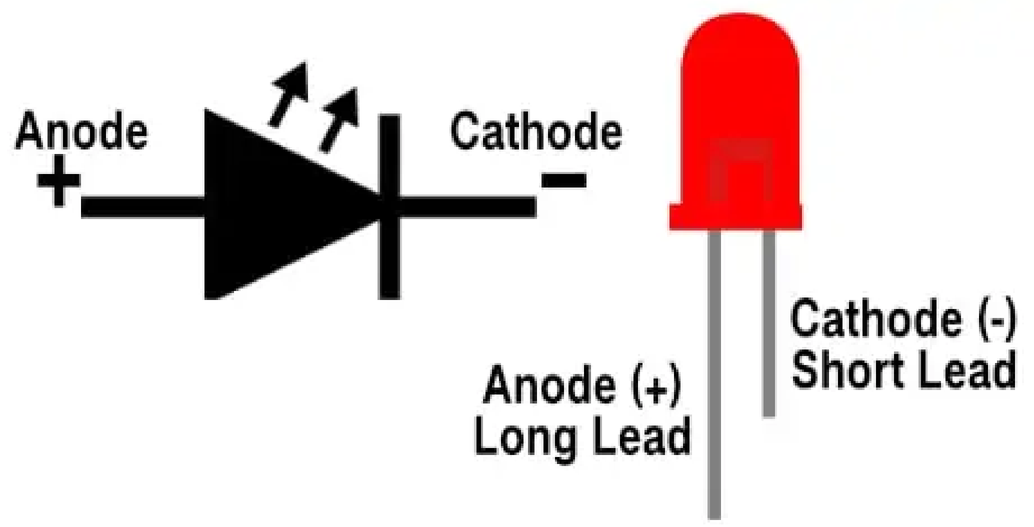


The SG90 uses Pulse Width Modulation (PWM) to control the position of its output shaft.

The width of the control pulse sent to the servo determines the angle of rotation:
A 1 ms pulse positions the shaft at 0°.
A 1.5 ms pulse positions the shaft at 90° (center position).
A 2 ms pulse positions the shaft at 180°.

When a PWM signal is applied to the control pin, the SG90's internal control circuit decodes it.
The desired position is compared to the actual position of the shaft (measured by the potentiometer).
The control circuit drives the DC motor to rotate the shaft until the two positions match.
Once the desired position is reached, the motor stops, holding its position with torque to resist external forces.

LED



LEDs emit light through a process called electroluminescence, where electric energy is converted directly into light energy. When a forward voltage is applied, electrons and holes recombine in the LED's semiconductor material, releasing energy in the form of photons (light).

The wavelength (color) of the emitted light depends on the bandgap energy of the semiconductor material used in the LED.

Energy Efficiency:

LEDs convert most of the electrical energy into light, minimizing energy loss.

Longevity:

LEDs can last up to 100,000–50,000 hours.

Compact Size:

Makes them suitable for a wide range of applications.

Environmentally Friendly:

No toxic materials like mercury (used in fluorescent bulbs).

Fast Switching:

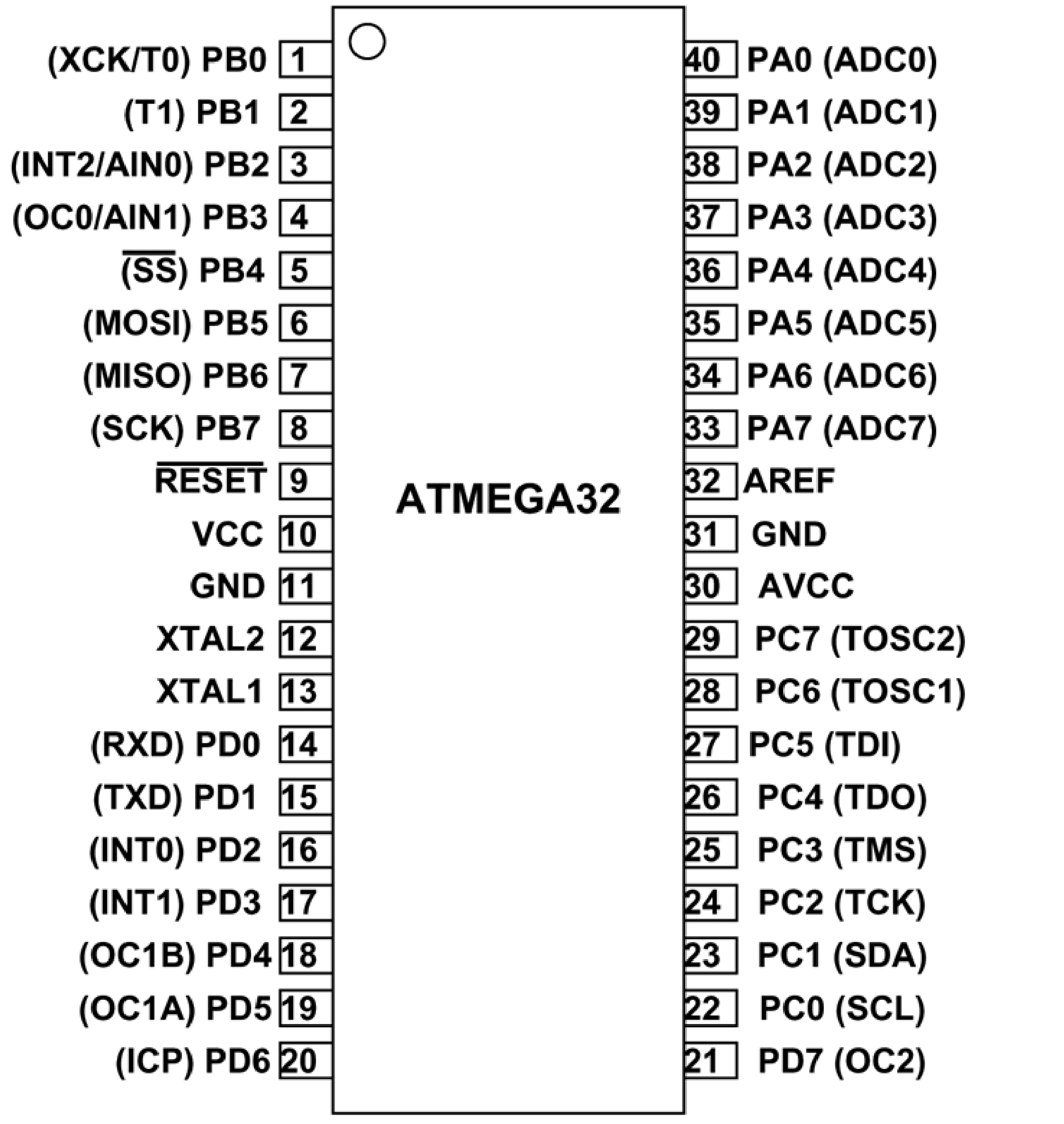
LEDs turn on instantly, making them ideal for signaling and displays.

Buzzer

The buzzer operates by converting electrical energy into sound through mechanical vibrations.

MCU


ATMEGA32A
PU
1018


ATMEGA32
(XCK/T0) PB0 1
(T1) PB1 2
(INT2/AIN0) PB2 3
(OC0/AIN1) PB3 4
$\overline{SS}$ PB4 5
(MOSI) PB5 6
(MISO) PB6 7
(SCK) PB7 8
$\overline{RESET}$ 9
VCC 10
GND 11
XTAL2 12
XTAL1 13
(RXD) PD0 14
(TXD) PD1 15
(INT0) PD2 16
(INT1) PD3 17
(OC1B) PD4 18
(OC1A) PD5 19
(ICP) PD6 20
40 PA0 (ADC0)
39 PA1 (ADC1)
38 PA2 (ADC2)
37 PA3 (ADC3)
36 PA4 (ADC4)
35 PA5 (ADC5)
34 PA6 (ADC6)
33 PA7 (ADC7)
32 AREF
31 GND
30 AVCC
29 PC7 (TOSC2)
28 PC6 (TOSC1)
27 PC5 (TDI)
26 PC4 (TDO)
25 PC3 (TMS)
24 PC2 (TCK)
23 PC1 (SDA)
22 PC0 (SCL)
21 PD7 (OC2)

MCU

High-performance, Low-power AVR ® 8-bit Microcontroller

Up to 16 MIPS Throughput at 16 MHz

32K Bytes of In-System Self-programmable Flash program memory

Two 8-bit Timer/Counters with Separate Prescalers and Compare Modes

One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode

Four PWM Channels

8-channel, 10-bit ADC

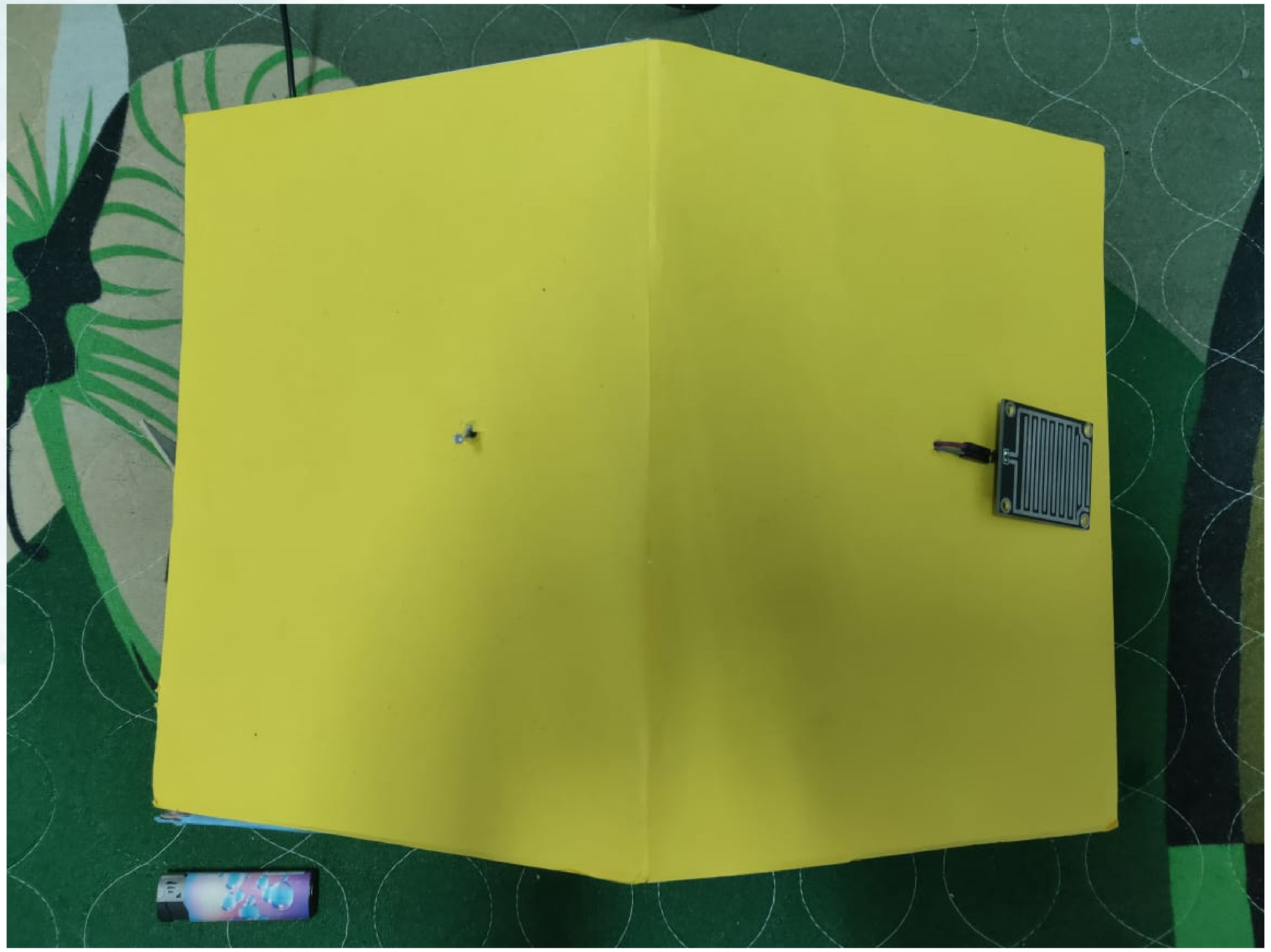
External and Internal Interrupt Sources

40-pin PDIP - 32 Programmable I/O Lines

Code

```
#include <util/delay.h>
#include "STD_TYPES.h"
#include "BIT_MATH.h"
#define MCUCSR *((volatile u8*)0x55)
#include "GPIO_private.h"
#include "GPIO_interface.h"
#include "GIE_interface.h"
#include "ADC_interface.h"
#include "TIMER_interface.h"
#include "LCD_interface.h"
#include "BUZ_interface.h"
#include "LED_interface.h"
#include "SERVO_interface.h"
#include "EXIT_interface.h"

u16 Result;
u16 LDR;
u16 LDR_value;
u16 value;
u8 flame;
u8 rain;
u8 motion;

BUZ_Type buz = { PORTD_ID, PIN3_ID, ACTIVE_HIGH };
LED_Type led = { PORTB_ID, PIN0_ID, ACTIVE_HIGH };

void main() {
        MCUCSR |= (7 >> 1);
        MCUCSR |= (7 >> 1);  // Needs to be written twice within four clock cycles

        GPIO_SetPinDirection(PORTD_ID, PIN2_ID, PIN_INPUT); //==> Flame sensor
        GPIO_SetPinDirection(PORTD_ID, PIN1_ID, PIN_INPUT); //==> Rain sensor
        GPIO_SetPinDirection(PORTD_ID, PIN0_ID, PIN_INPUT); // ==> Motion sensor


        GPIO_SetPinDirection(PORTD_ID,PIN4_ID,PIN_OUTPUT); // Rain Servo
        GPIO_SetPinDirection(PORTD_ID, PIN5_ID, PIN_OUTPUT); // Motion Servo


        BUZ_Init(buz);
        LED_Init(led);
        ADC_Init();
        LCD_Init();

        TIMER1_Init();

        ADC_Enable();
        GIE_Enable();

        LCD_SendString("T=");
while (1) {
                value = 0;
                LDR_value = 0;
                for (int i = 0; i < 10; i++)
                {
                        ADC_GetResultSingle(ADC_CHANNEL0, &Result);
                        value += ((((u16) Result * 2560ul) / 1023ul) / 10ul);
                }
                for (int i = 0; i < 10; i++)
                {
                        ADC_GetResultSingle(ADC_CHANNEL1, &LDR);
                        LDR_value += ((((u16) LDR * 2560ul) / 1023ul));
                }
                value /= 10;
                LDR_value /= 10;

                flame = GPIO_GetPinValue(PORTD_ID, PIN2_ID);
                rain = GPIO_GetPinValue(PORTD_ID, PIN1_ID);
                motion = GPIO_GetPinValue(PORTD_ID, PIN0_ID);

                LCD_SetPosition(LCD_ROW_1, LCD_COL_3);

if (LDR_value > 1500) {
        LED_On(led);
}
else
{
        LED_Off(led);
}

if (value >= 100)
{
        LCD_SendString(" ");
        LCD_SendNumber(value);
        LCD_SendData('C');
}
else if (value > 10 && value < 100)
{
        LCD_SendString("  ");
        LCD_SendNumber(value);
        LCD_SendData('C');
}
else if (value < 10)
{
        LCD_SendString("   ");
        LCD_SendNumber(value);
        LCD_SendData('C');
}

if (flame)
{
        LCD_SetPosition(LCD_ROW_2, LCD_COL_1);
        LCD_SendString("Fire!!!");
        BUZ_On(buz);
}
else
{
        LCD_SetPosition(LCD_ROW_2, LCD_COL_1);
        LCD_SendString("       ");
        BUZ_Off(buz);
}

if (rain == 0)
{
        LCD_SetPosition(LCD_ROW_1, LCD_COL_9);
        LCD_SendString("Raining!");
        TIMER1_SetCTCB(2000);
}

else
{
        LCD_SetPosition(LCD_ROW_1, LCD_COL_9);
        TIMER1_SetCTCB(500);
        LCD_SendString("        ");
}

if (motion)
{
        LCD_SetPosition(LCD_ROW_2, LCD_COL_9);
        LCD_SendString("D_Opened");
        TIMER1_SetCTCA(2300);
} else
{
        LCD_SetPosition(LCD_ROW_2, LCD_COL_9);
        LCD_SendString("D_Closed        ");

        TIMER1_SetCTCA(500);
}
    }
}
```

Final Project



Thank you