ESP32 (38Pin version)

- 18 Analog-to-Digital Converter (ADC) channels
- 3 SPI interfaces
- 3 UART interfaces
- 2 I2C interfaces
- 16 PWM output channels
- 2 Digital-to-Analog Converters (DAC)
- 2 I2S interfaces
- 10 Capacitive sensing GPIO’s
are the LEDs connected with each other?
Now?
Now?
3 options to power up ESP32.

1. Directly via micro-USB port.

2. Unregulated power to GND and 5V pins (Between 5 to 12 v)

3. Regulated power to GND and 3.3V pins (ONLY 3.3v!)

Always only power the microcontroller with one option.
GPIO (General Purpose IO) for both digital input and output

Always refer to the pin layout
Digital Output – Blink an LED
Digital Output

Set the logic value of a pin
– **LOW** (0V) or **HIGH** (3.3V)

Arduino functions
– `pinMode(pin, OUTPUT)` to set the pin direction
  *Often in the `setup()` function*
– `digitalWrite(pin, value)` to write the current value of a pin

Limitations
– Only 0 or 3.3 V with limited current;
Blink the built-in LED

// constants definition
const int ledPin = 2; // Default LED is connected to GPIO 2
// The setup() method runs once, when the sketch starts
void setup() {
  // initialize the digital pin as an output:
  pinMode(ledPin, OUTPUT);
}

// the loop() method runs over and over again,
// as long as the Arduino has power
void loop() {
  digitalWrite(ledPin, HIGH); // set the LED on
  delay(5000); // wait for 5 second
  digitalWrite(ledPin, LOW); // set the LED off
  delay(5000); // wait for 5 second
}
Select
Board: -> esp32 -> NodeMCU-32S

Hit Upload

On the ESP32,
Press and hold the BOOT button until you see the code starts uploading
Practice: Light up the RED Led
3.3V 2.1V

20mA

add a resistor (lowers the current)

Ohm’s Law

\[ \Delta V = R \times I \]
Blink an external LED

// constants definition
const int ledPin = 23; // Default LED is connected to GPIO 23

// The setup() method runs once, when the sketch starts
void setup()
{
    // initialize the digital pin as an output:
    pinMode(ledPin, OUTPUT);
}

// the loop() method runs over and over again,
// as long as the Arduino has power
void loop()
{
    digitalWrite(ledPin, HIGH); // set the LED on
    delay(5000); // wait for 5 second
    digitalWrite(ledPin, LOW); // set the LED off
    delay(5000); // wait for 5 second
}
Serial Communication – talk to PC
Serial Communication

Setup
– Serial.begin(<baud_speed>)//9600

Receiving information
– Test is data is available
  Serial.available()
– Read one byte
  Serial.read()

Sending information
– Raw data transfer
  Serial.write(val) or Serial.write(buf, len)

Other commands -> https://www.arduino.cc/reference/en
– Formatted output
  Serial.print(x,{BIN,OCT,DEC,HEX})
– Read formatted data
  Serial.parseFloat()
  Serial.parseInt()
Echo program
// setup performs initializations
void setup()
{
    // initialize the serial port setting its speed to 9600 Baud:
    Serial.begin(9600);
}

// the loop() method runs over and over again,
// as long as the Arduino has power
void loop()
{
    // Temporary buffer
    byte incoming_byte;
    // check if the something is pending
    if (Serial.available() > 0)
    {
        // read the pending byte;
        incoming_byte = Serial.read();
        // Sending it back;
        Serial.write(incoming_byte);
    }
}
Pulse Width Modulation (PWM)
Now modify your program to

- blink the LED with 100% light intensity when type ‘1’ from the PC
- turn it off when type ‘0’
- light up with 50% light intensity when type ‘2’

analogWrite() is on a scale of 0 - 255