Advanced Concepts and Electrical Components

-Harshal Deshpande

Topics:

- Interrupts and Polling
- I2C Protocol
- Components:
  - Motor Driver
  - Voltage Regulator
  - Buck Convertors
  - OP-AMP
  - Logic Gates

What’s the difference between polling and Interrupt?

Polling:

- Polling is the process where the computer or controlling device waits for an external device(sensor in this case) to check for its readiness(sensor initialization) or state.

Interrupts:

- Interrupts is a process where the controller checks the state of the external device if there’s any change in the state of the sensor.

Problem Statement:

Design a room monitoring system, which uses a temperature sensor to get the temperature in the room. You have to get the get the data at a 2 min interval and display the data on a graph.

Interrupts:

There are two types of interrupts –

- **Hardware Interrupts** – They occur in response to an external event, such as an external interrupt pin going high or low.
- **Software Interrupts** – They occur in response to an instruction sent in software. The only type of interrupt that the “Arduino language” supports is the attachInterrupt() function.

Interrupts can come from various sources(eg. digital input pin). You can define a special function called as Interrupt Service Routine which is executed at the rising edge, falling edge or both of the input signal.

If your sketch uses multiple ISRs, only one can run at a time. Other interrupts will be executed after the current one finishes in an order that depends on the priority they have.

Applications:

1. Hardware Interrupt for rotary encoders
2. Hardware Interrupt for observing user input
3. Timer interrupt to get sensor data at specific interrupt

Timer Interrupts:

https://techtutorialsx.com/2017/10/07/esp32-arduino-timer-interrupts/

Syntax:

➜ attachInterrupt(digitalPinToInterrupt(GPIO), function, mode);

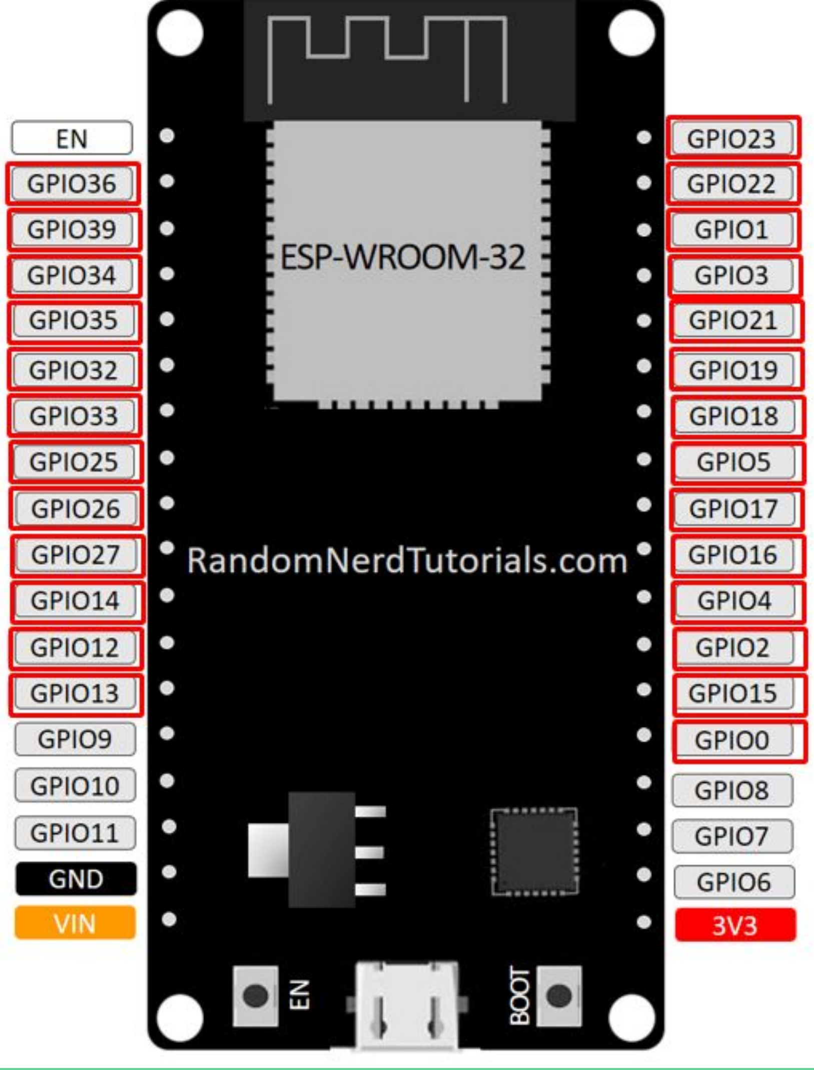
GPIO - pin no.

Function - Name of the ISR

Modes:

- **LOW** to trigger the interrupt whenever the pin is low.
- **CHANGE** to trigger the interrupt whenever the pin changes value.
- **FALLING** whenever the pin goes from high to low.
- **RISING** whenever trigger changes from low to high
- **FALLING** whenever trigger changes from high to low

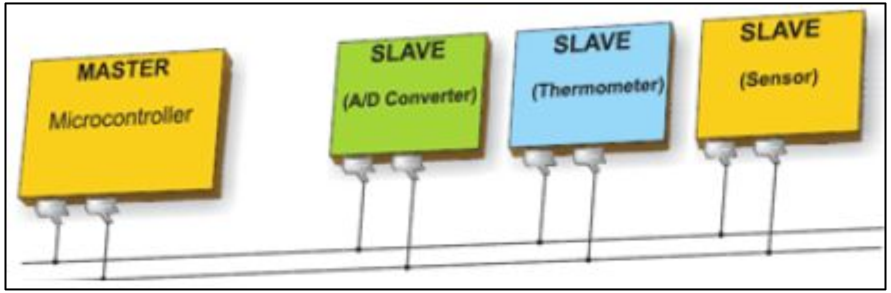
Red pins can used are interrupt pins.



I2C Protocol:

Inter-integrated circuit (I2C) is a system for serial data exchange between the microcontrollers and specialized integrated circuits of a new generation. It is used when the distance between them is short (receiver and transmitter are usually on the same printed board).

Connection takes place over 2 conducting wires. One is used for synchronization and another one is used for data transmission.



In I2C there is a master (which controls the data bus) and there can be multiple slaves.

There can be two possible cases:

1. Master is the transmitter and slaves receive the data
2. Slaves transmit the data and master receives the data

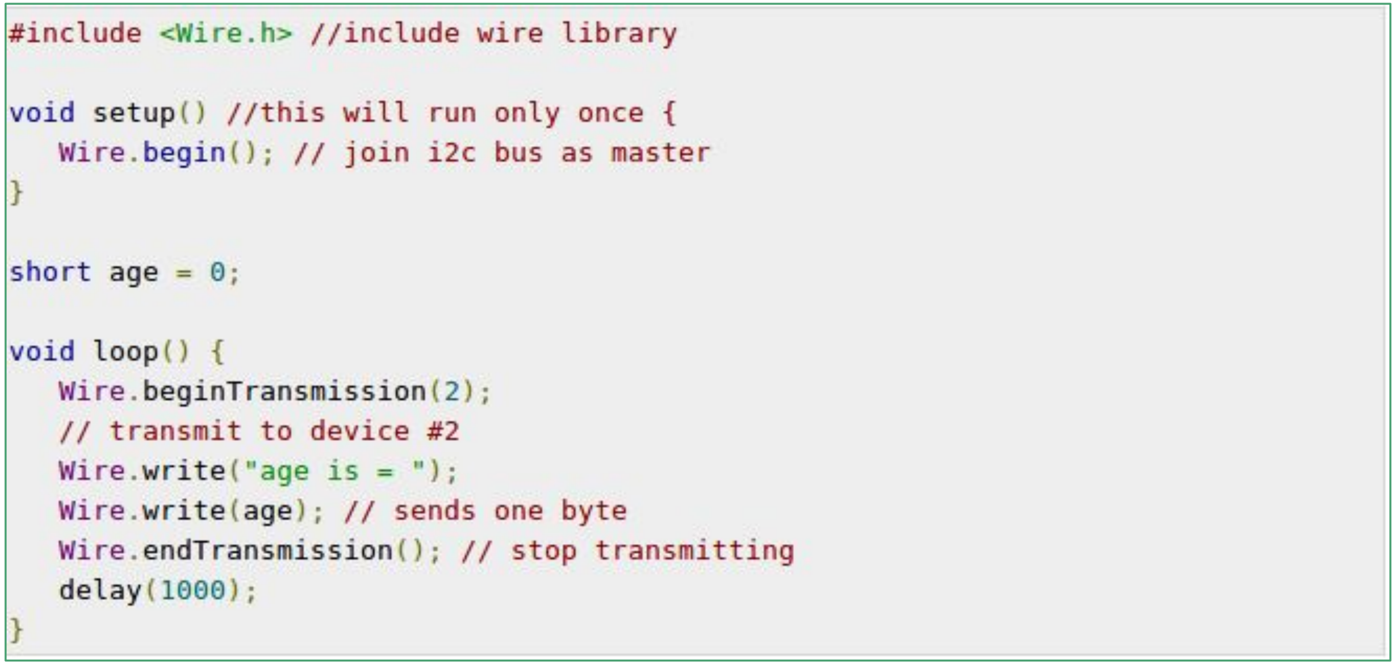
Syntax:

To use the I2C protocol you need to import ‘Wire’ library.

On ESP 32, pin 21 is used for data and pin 22 is used for clock.

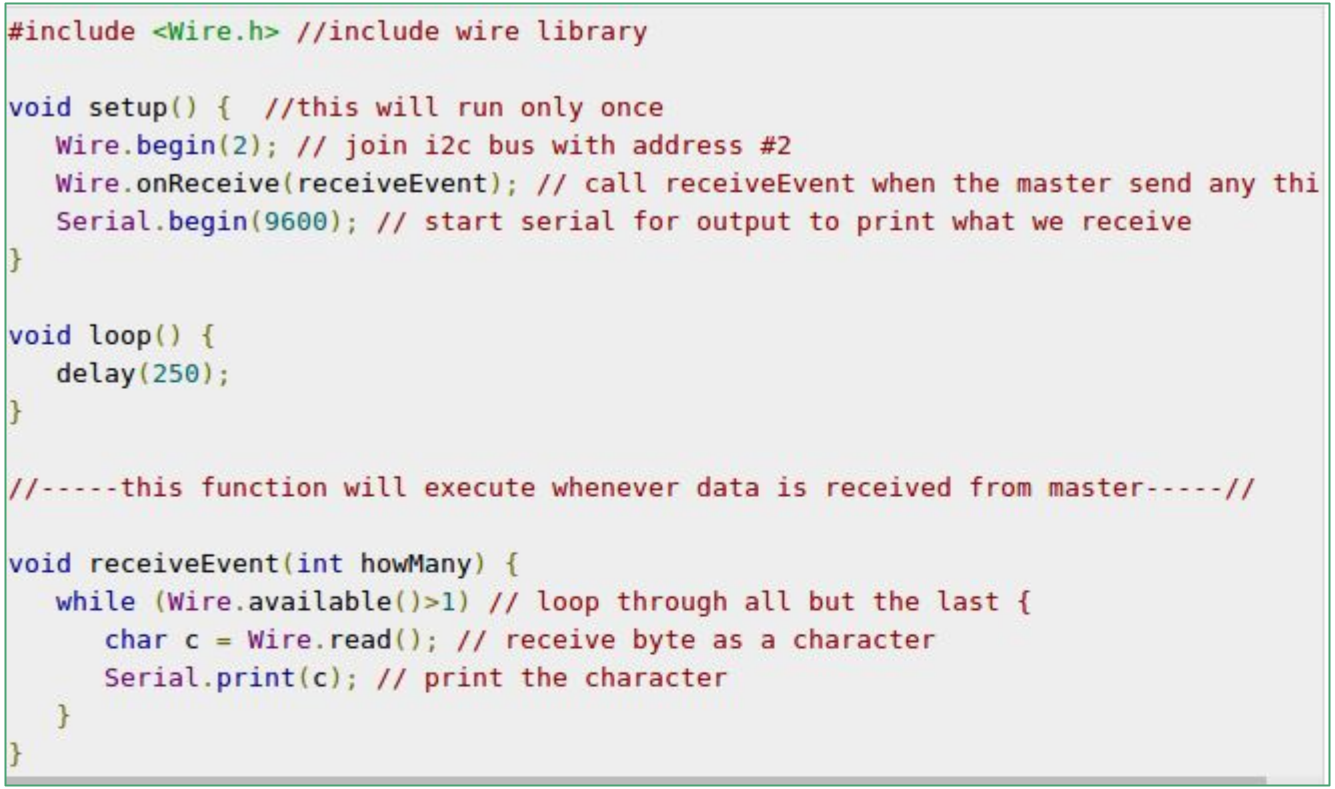
https://www.arduino.cc/en/reference/wire

Case I: Master is transmitting



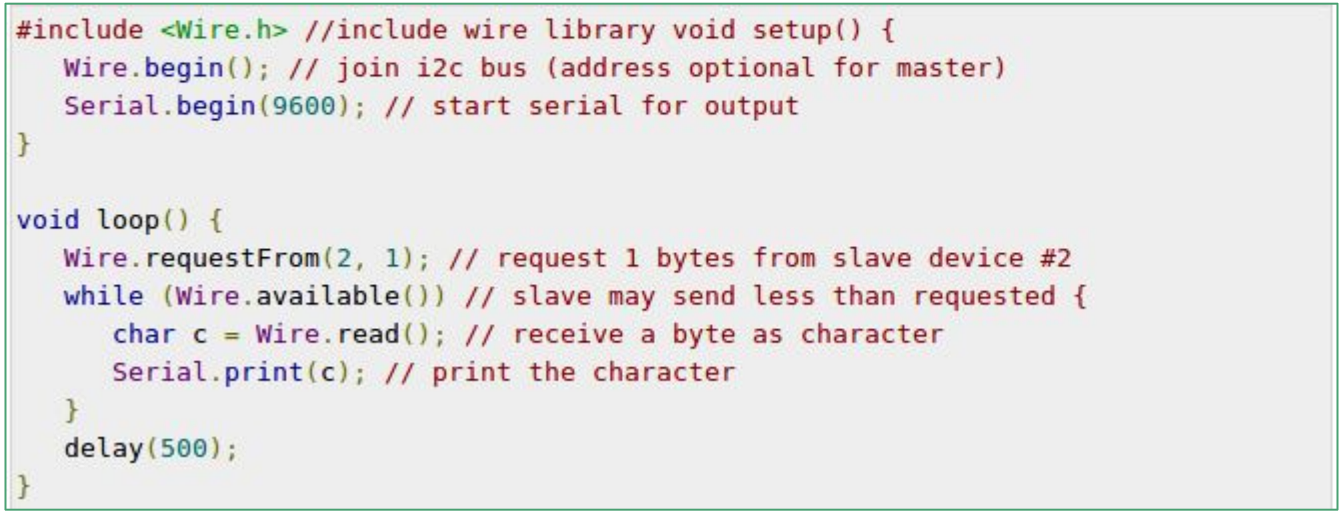
```
#include <Wire.h> //include wire library

void setup() //this will run only once {
   Wire.begin(); // join i2c bus as master
}

short age = 0;

void loop() {
   Wire.beginTransmission(2);
   // transmit to device #2
   Wire.write("age is = ");
   Wire.write(age); // sends one byte
   Wire.endTransmission(); // stop transmitting
   delay(1000);
}
```

Slave receiving



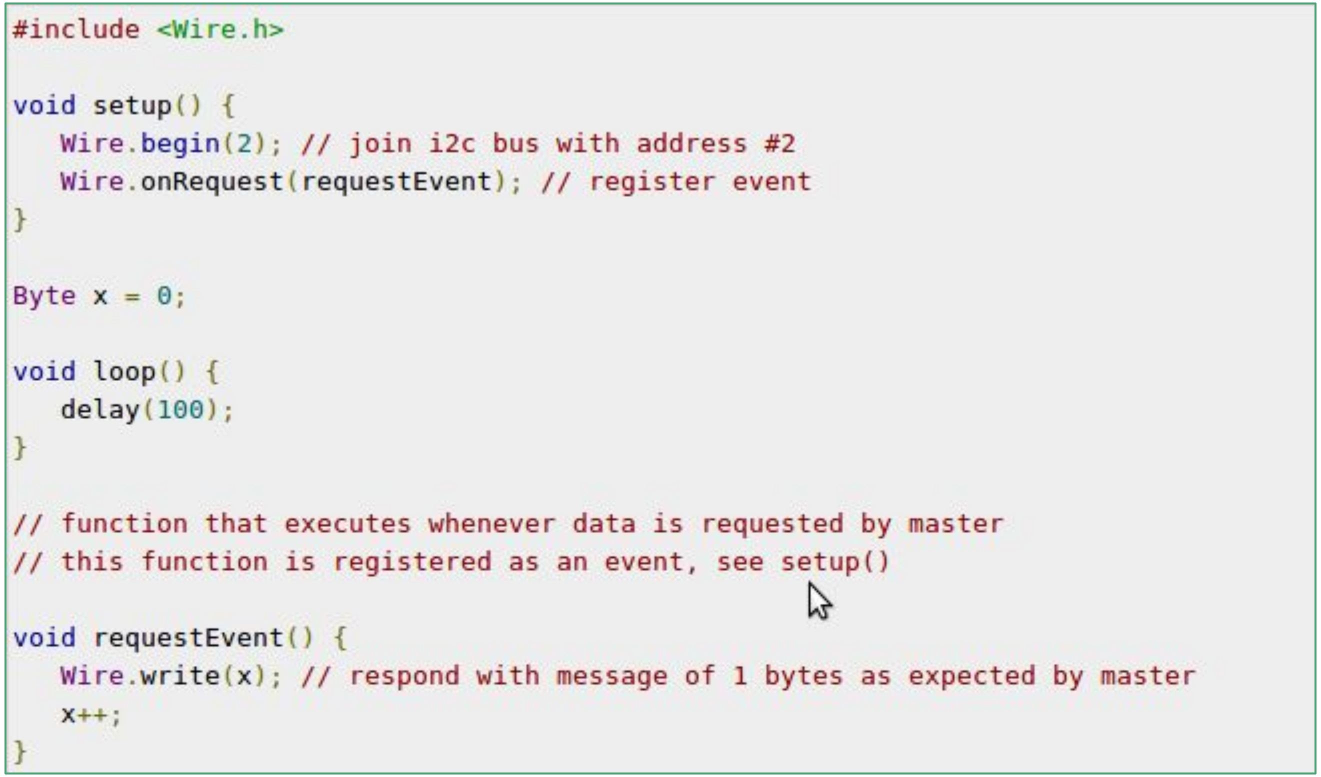
```
#include <Wire.h> //include wire library

void setup() {  //this will run only once
   Wire.begin(2); // join i2c bus with address #2
   Wire.onReceive(receiveEvent); // call receiveEvent when the master send any thi
   Serial.begin(9600); // start serial for output to print what we receive
}

void loop() {
   delay(250);
}

//-----this function will execute whenever data is received from master-----//

void receiveEvent(int howMany) {
   while (Wire.available()>1) // loop through all but the last {
      char c = Wire.read(); // receive byte as a character
      Serial.print(c); // print the character
   }
}
```

Case II: Master is receiver



```
#include <Wire.h> //include wire library void setup() {
   Wire.begin(); // join i2c bus (address optional for master)
   Serial.begin(9600); // start serial for output
}

void loop() {
   Wire.requestFrom(2, 1); // request 1 bytes from slave device #2
   while (Wire.available()) // slave may send less than requested {
      char c = Wire.read(); // receive a byte as character
      Serial.print(c); // print the character
   }
   delay(500);
}
```

Slave - transmitting



```
#include <Wire.h>

void setup() {
   Wire.begin(2); // join i2c bus with address #2
   Wire.onRequest(requestEvent); // register event
}

Byte x = 0;

void loop() {
   delay(100);
}

// function that executes whenever data is requested by master
// this function is registered as an event, see setup()

void requestEvent() {
   Wire.write(x); // respond with message of 1 bytes as expected by master
   x++;
}
```

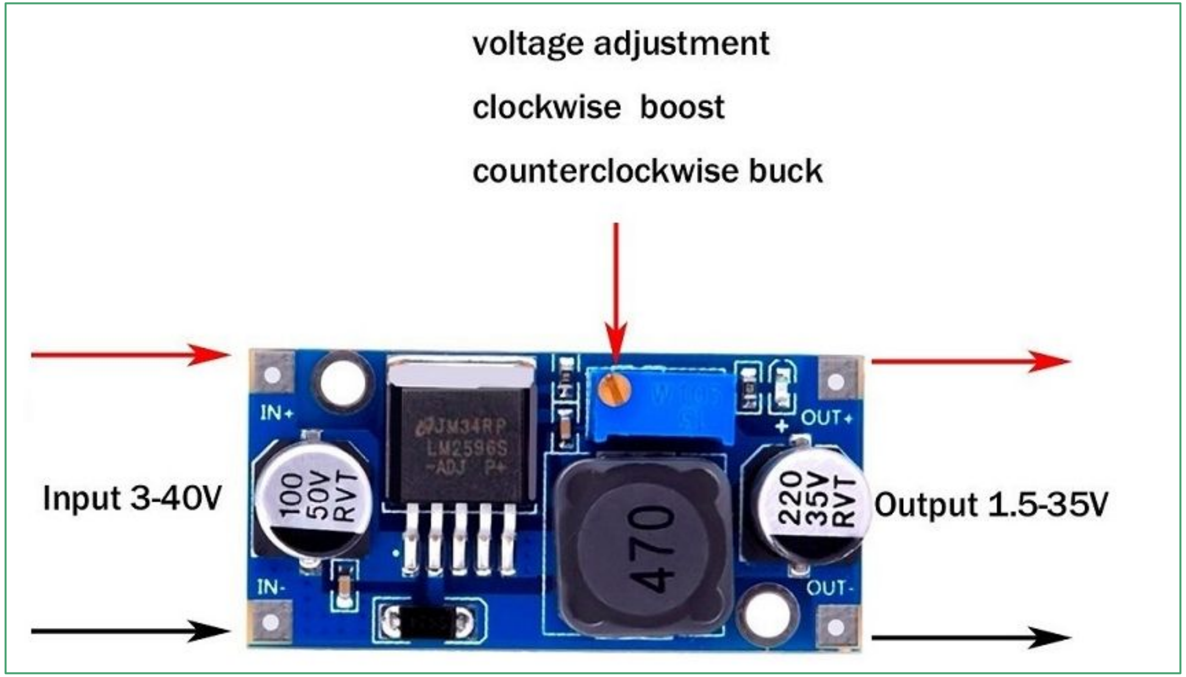
Commonly used Components

1) Voltage Regulator

A **voltage regulator** is a system designed to automatically maintain a constant **voltage** level across its terminals.

Refer:
https://www.elprocus.com/types-of-voltage-regulators-and-working-principle/

2) Buck Convertor



Steps down DC-DC voltage level

Refer:
https://www.instructables.com/id/The-Introduction-of-LM2596-Step-Down-Power-Module-/

3) OP-AMP

OP-AMP(Operational Amplifier), as the name suggests is used for amplification of voltage/current. There are different OPAMPs available which are used for either voltage amplification or current amplification.

"Theory" : https://www.khanacademy.org/science/electrical-engineering/ee-amplifiers/ee-opamp/v/ee-opamp-intro

4) Logic Gates

Logic gates are the basic building blocks of any digital system. It is an electronic circuit having one or more than one input and only one output.

The output is based on some logic. Based on the logic following gates exist:

1. OR
2. AND
3. NOT
4. NOR
5. NAND
6. EXOR
7. XNOR

1 - AND: When all the inputs are high the output is High

2 - OR: When any one of the input is high the output is high

3 - NOT: Output is opposite of input

4 - XOR: When odd no of inputs are high the output is high

Refer: https://www.tutorialspoint.com/computer_logical_organization/logic_gates.htm

Motor Driver:

A motor driver IC is an integrated circuit chip which is usually used to control motors in autonomous robots.

Motor driver ICs act as an interface between microprocessors in robots and the motors in the robot.

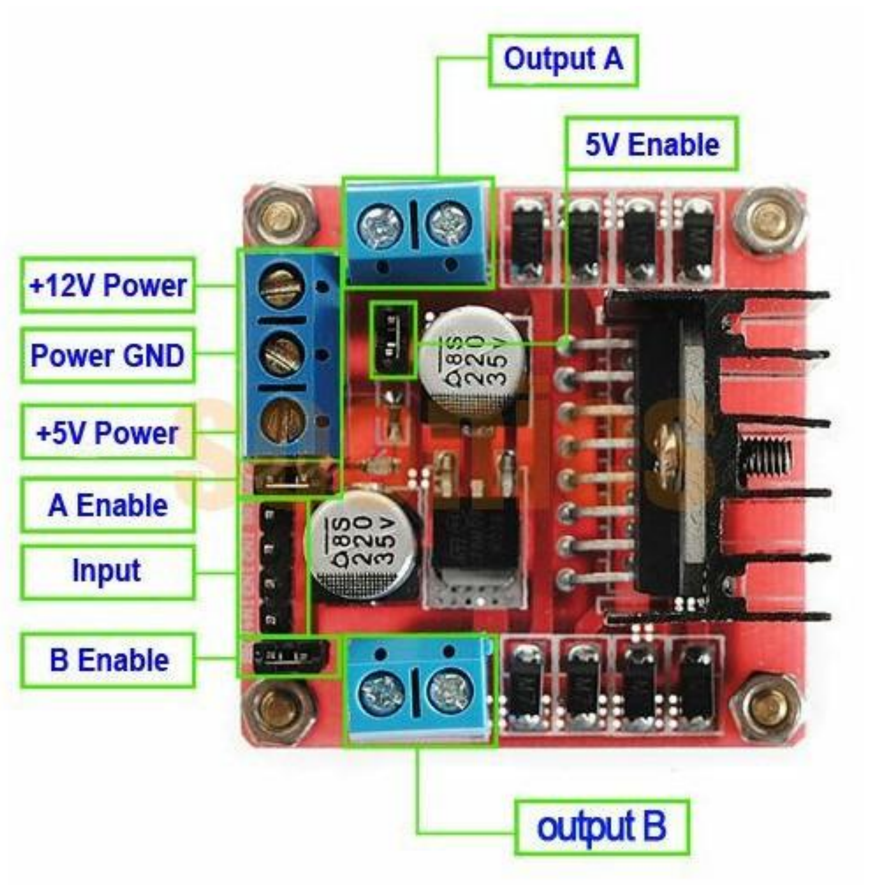
The most commonly used motor driver IC’s are from the L293 series such as L293D, L293NE, etc. These ICs are designed to control 2 DC motors simultaneously.

Why do we need motor driver?

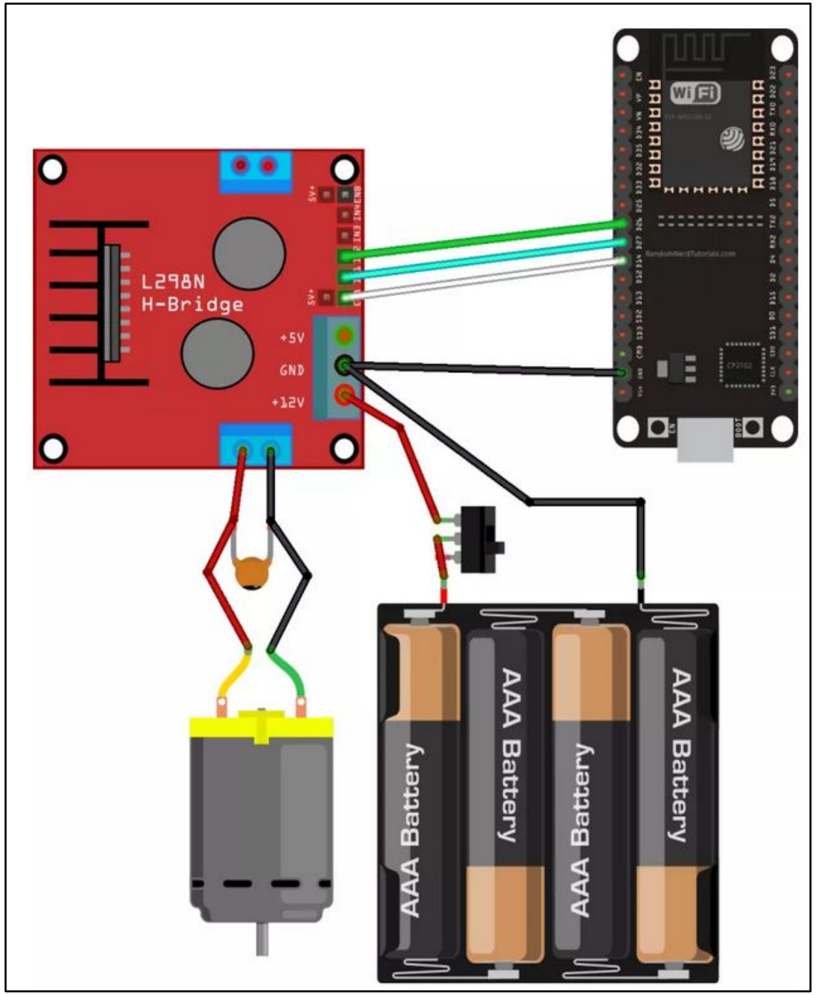
Microcontrollers/microprocessors operate at low voltages and require a small amount of current to operate while the motors require a relatively higher voltages and current . Thus current cannot be supplied to the motors from the microprocessor. This is the primary need for the motor driver IC.

L298N

5v pin can give output current of 0.5A. You can use this to power up your microcontroller



- A DC Motor needs a Motor driver to control its output. In our case we will be using a L298N based motor driver. To get the details of the motor driver go through it’s datasheet.
- The DC motor is connected to OUT1 and OUT2 on the motor driver. The motor driver is powered used 4 AAA batteries.(You can also use a 3-cell lithium polymer battery)
- As shown in the schematic there are 3 wire coming from ESP to motor driver. White-Enable wire, Cyan- IN1, Green-IN2. IN1 and IN2 decide the direction of rotation of the motor and the Enable decides the speed of the rotation.
- A PWM signal is given on the Enable pin on the motor driver.



Tasks

Task - 1

Connect 2 Arduino Unos on i2c bus. Send a string from slave and print the string on serial console of the master.

Note:

On Arduino Uno - SDA: A4

SCL: A5

Task - 2

Write an ISR which is called when a there’s change in the output from a slide switch.

References:

https://www.tutorialspoint.com/arduino/arduino_interrupts.htm

https://www.tutorialspoint.com/arduino/arduino_inter_integrated_circuit.htm

What will I do next?

http://esp32.net/