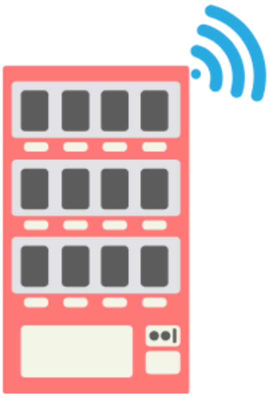


0.1. Smart MDB Controller

status active

Smart MDB Controller

1. 📝 Table of Contents

2. 🧐 About

This repo contains circuit, firmware and backend for Smart MDB Controller Project.

3. 🏁 Getting Started

These instructions will get you a copy of the project up and running on your local machine for development and testing purposes. See deployment for notes on how to deploy the project on a live system.

3.1. Prerequisites

What things you need to install the software and how to install them.

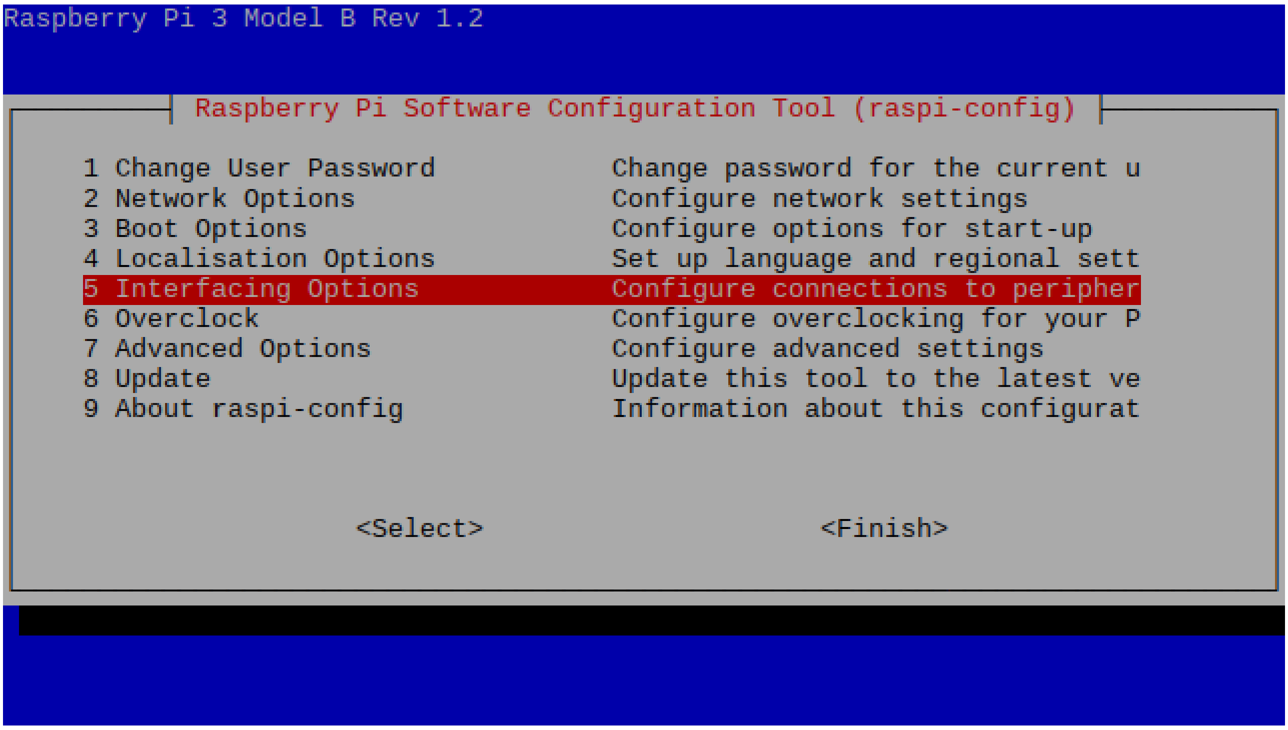
```
- Raspberry Pi Model 3B, 3B+, 4B or CM4
```

4. Installation and Configuration

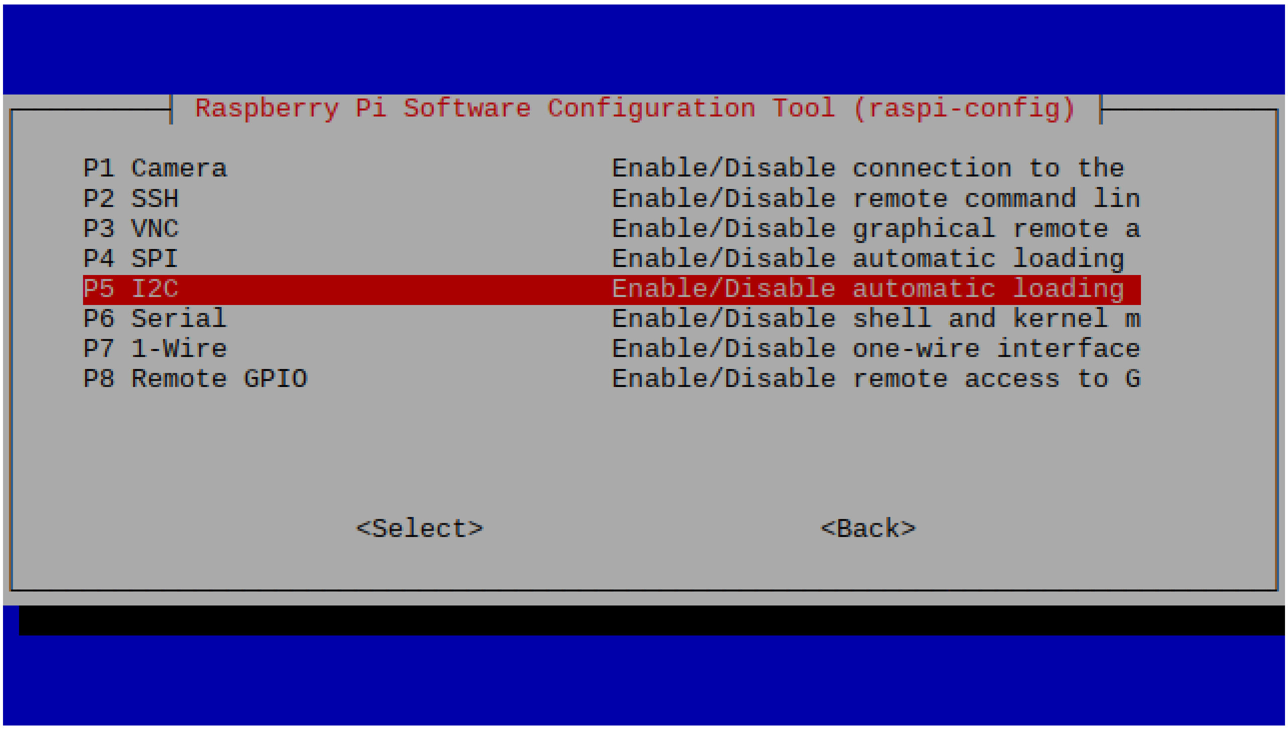
A step by step series that covers how to get the Firmware running.

4.1. Raspberry Pi Firmware Pre-Reqs

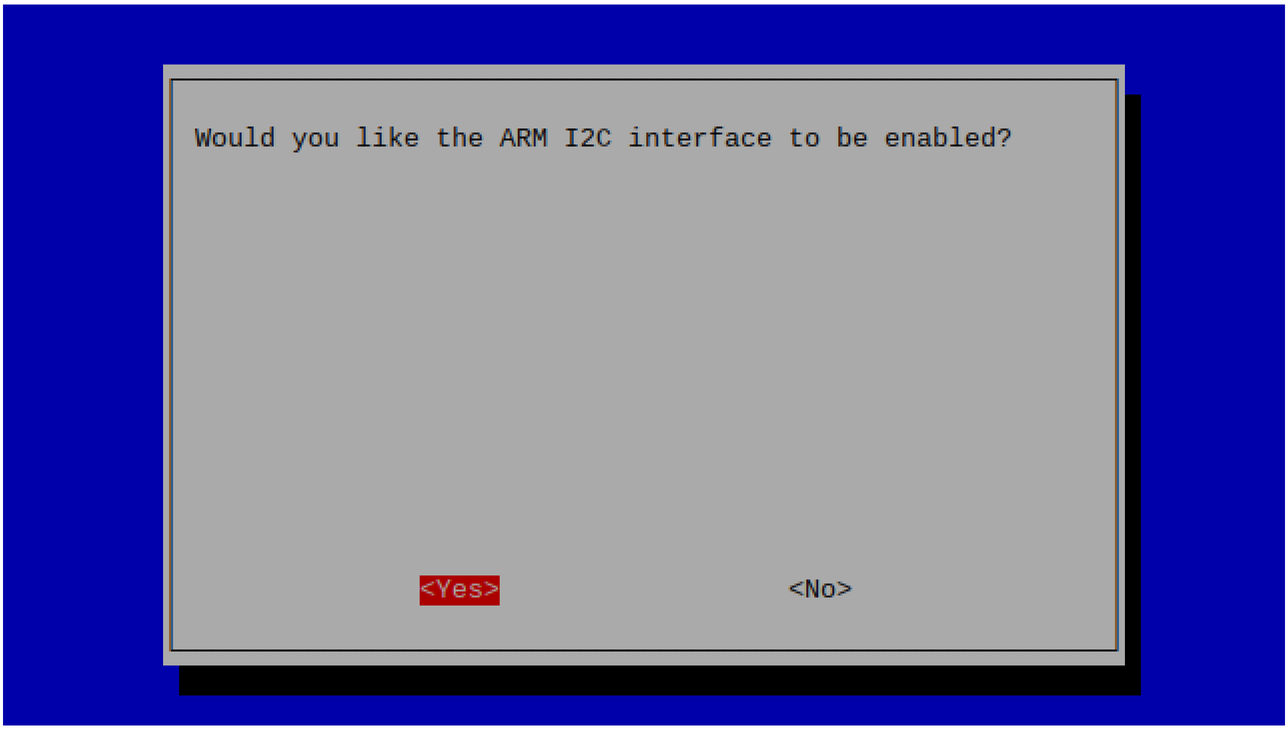
1. Download and install the latest Raspberry Pi OS Desktop image to your SD card
2. Open the terminal and execute the following command `sudo raspi-config`
3. Then follow the following pictures to enable I2C bus on you raspberry pi

- 

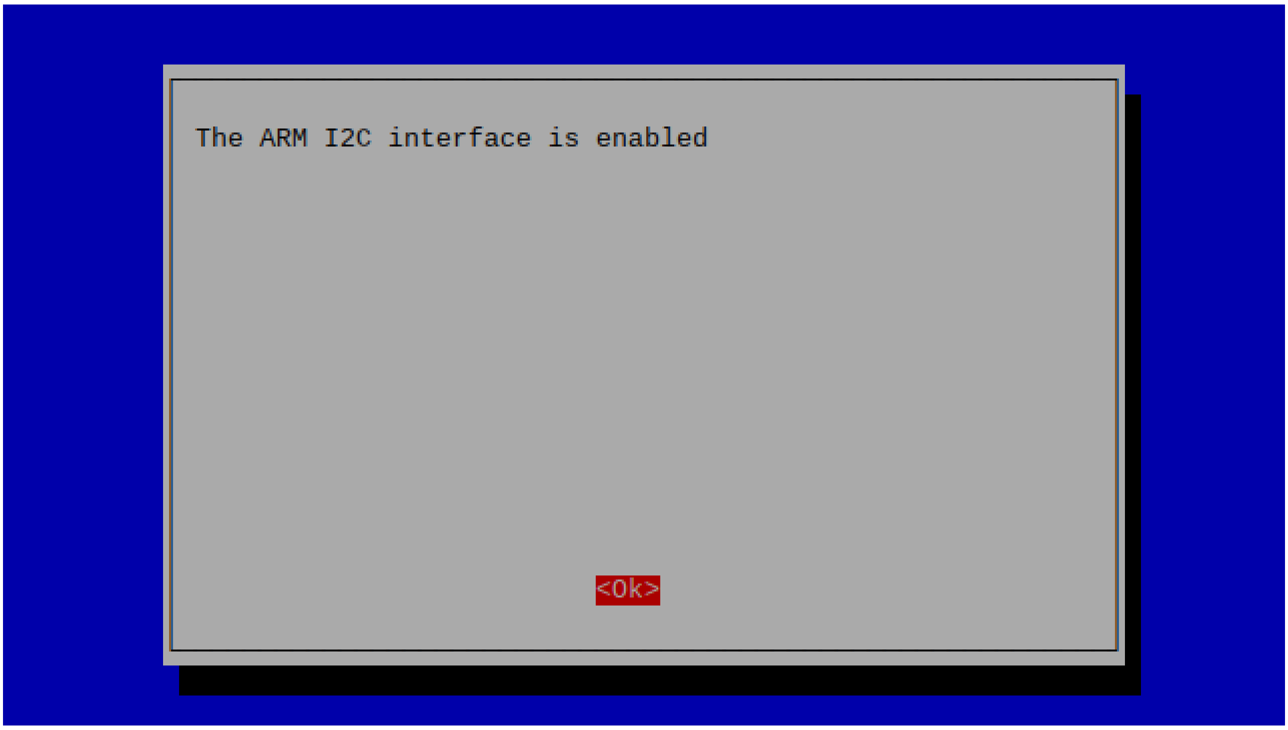
```
Raspberry Pi 3 Model B Rev 1.2

┌──────────┤ Raspberry Pi Software Configuration Tool (raspi-config) ├──────────┐

     1 Change User Password          Change password for the current u
     2 Network Options               Configure network settings
     3 Boot Options                  Configure options for start-up
     4 Localisation Options          Set up language and regional sett
     5 Interfacing Options           Configure connections to peripher
     6 Overclock                     Configure overclocking for your P
     7 Advanced Options              Configure advanced settings
     8 Update                        Update this tool to the latest ve
     9 About raspi-config            Information about this configurat


                  <Select>                     <Finish>

└───────────────────────────────────────────────────────────────────────────────┘
```

- 

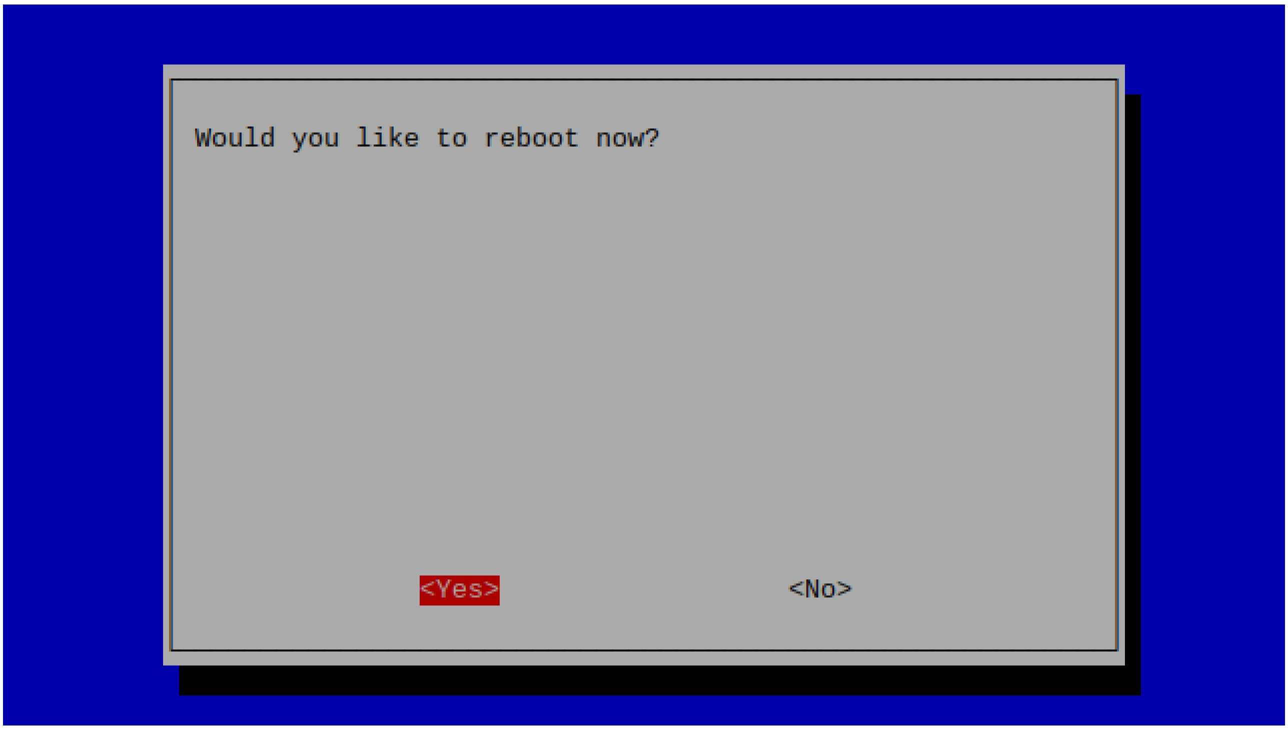
```
┤ Raspberry Pi Software Configuration Tool (raspi-config) ├

 P1 Camera                         Enable/Disable connection to the
 P2 SSH                            Enable/Disable remote command lin
 P3 VNC                            Enable/Disable graphical remote a
 P4 SPI                            Enable/Disable automatic loading
 P5 I2C                            Enable/Disable automatic loading
 P6 Serial                         Enable/Disable shell and kernel m
 P7 1-Wire                         Enable/Disable one-wire interface
 P8 Remote GPIO                    Enable/Disable remote access to G



                 <Select>                     <Back>
```

- 

```
Would you like the ARM I2C interface to be enabled?



                 <Yes>                  <No>
```

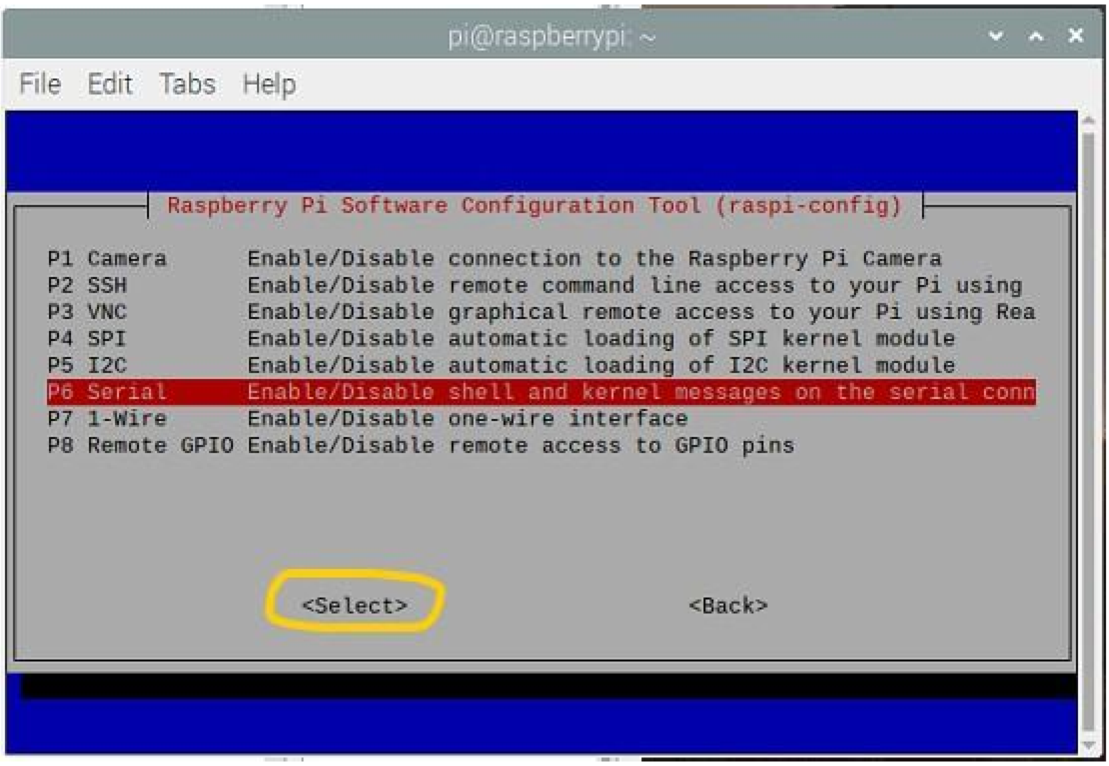
- 

  The ARM I2C interface is enabled

- 

  Would you like to reboot now?

- Then do the same for Serial(UART)

- 

```
Raspberry Pi Software Configuration Tool (raspi-config)

P1 Camera      Enable/Disable connection to the Raspberry Pi Camera
P2 SSH         Enable/Disable remote command line access to your Pi using
P3 VNC         Enable/Disable graphical remote access to your Pi using Rea
P4 SPI         Enable/Disable automatic loading of SPI kernel module
P5 I2C         Enable/Disable automatic loading of I2C kernel module
P6 Serial      Enable/Disable shell and kernel messages on the serial conn
P7 1-Wire      Enable/Disable one-wire interface
P8 Remote GPIO Enable/Disable remote access to GPIO pins

               <Select>                      <Back>
```

4.2. Configuring Raspberry Pi and Running the UI

1. Copy FirmwareRPi folder to the desktop of your Raspberry Pi, open the terminal of your Raspberry Pi and execute the following commands

```
- sudo apt-get update
- sudo apt-get upgrade
- sudo apt install python3-pip
- pip3 install pyserial
- pip3 install paho-mqtt
- sudo adduser $USER dialout
- cd ~/Desktop/FirmwareRPi
- sudo chmod a+rx starter.sh
```

1. To run the program just double click on starter.sh file
    1. or execute `python3 /home/pi/Desktop/FirmwareRPi/Firmware.py`

This program make use of MQTT to communicate with the webapp.

```
+ broker used is borker.hivemq.com
+ vending can be done by publishing an ammount to mdb/invoke topic
```

4.3. Arduino Mega Configurations

The following packages should be installed on your PC

- AVR-GCC
- AVRDUDE
- avr-libc

4.3.1. Installing Pre-Reqs on Linux(Ubuntu or Debian Based Distro)

On Ubuntu or any Debian based Linux distro execute the following commands

```
- sudo apt-get install -y gcc-avr
- sudo apt-get install -y avrdude
- sudo apt-get install avr-libc
```

4.4. Setup

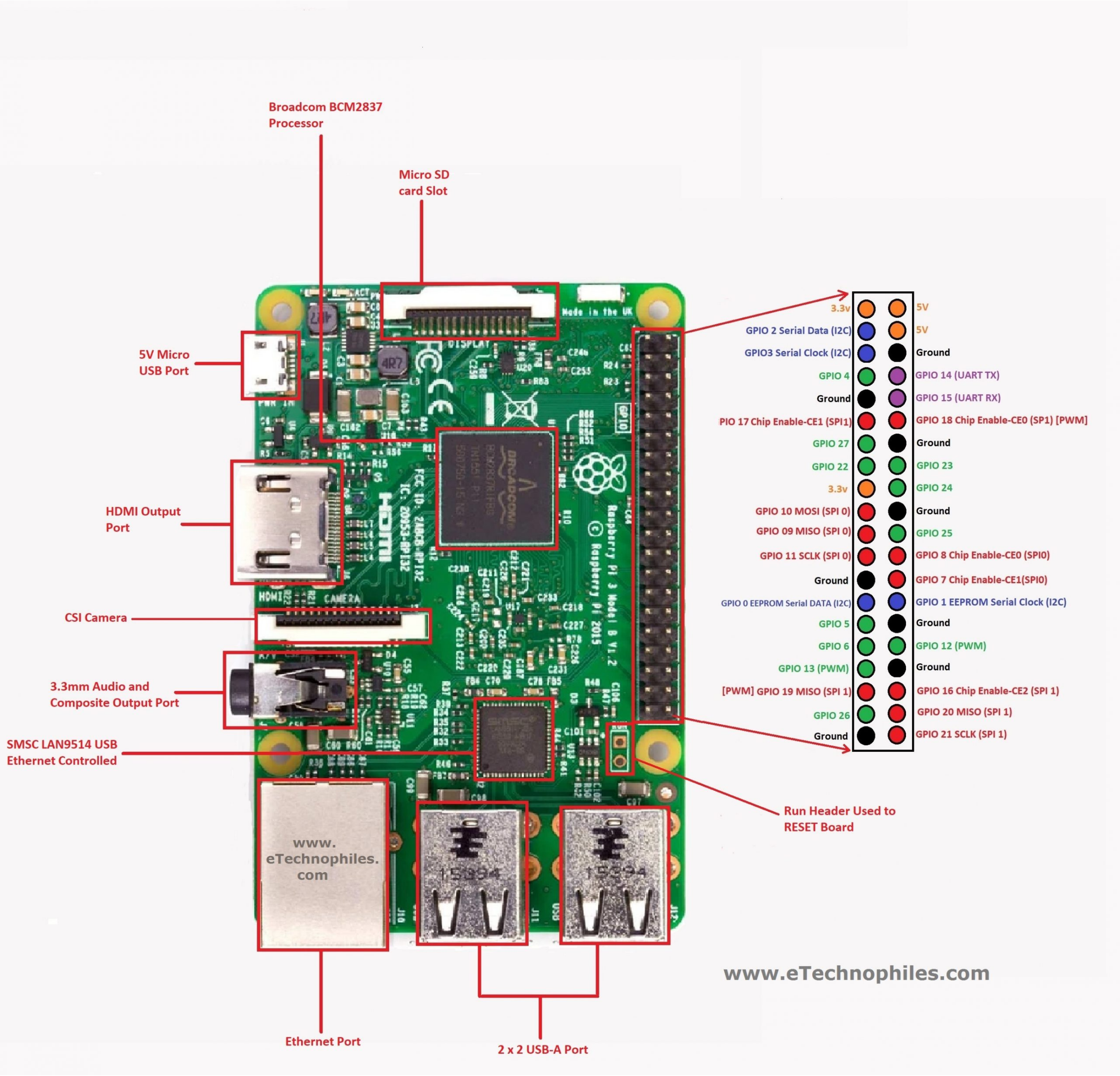
1. Open terminal in the atmega2560FW folder and execute `make clean`
2. Open makefile, replace P=COM1 to the COM port of your arduino mega and save the file
    1. Compile it with `make all`
3. Upload it to your Arduino with `make program`

5. ⛏️ Testing

1. The Firmware can be tested on Raspberry Pi 3B, 3B+ or 4B with the following modifications
    1. Connect the sensor as shown in the Circuit Diagram section below.

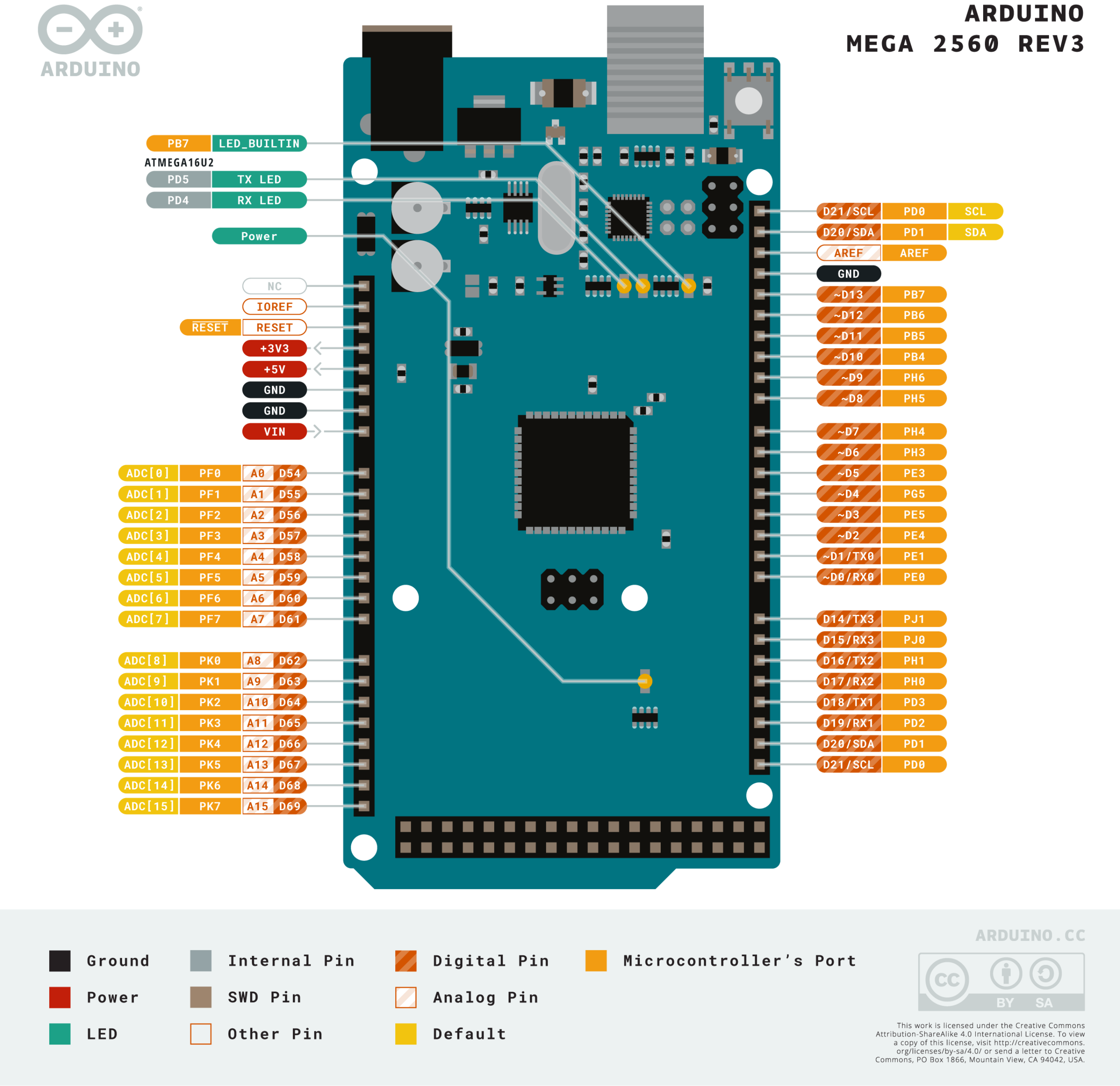
6. 🔌 Circuit Diagram

- RPi 3,4 GPIOs Pinout

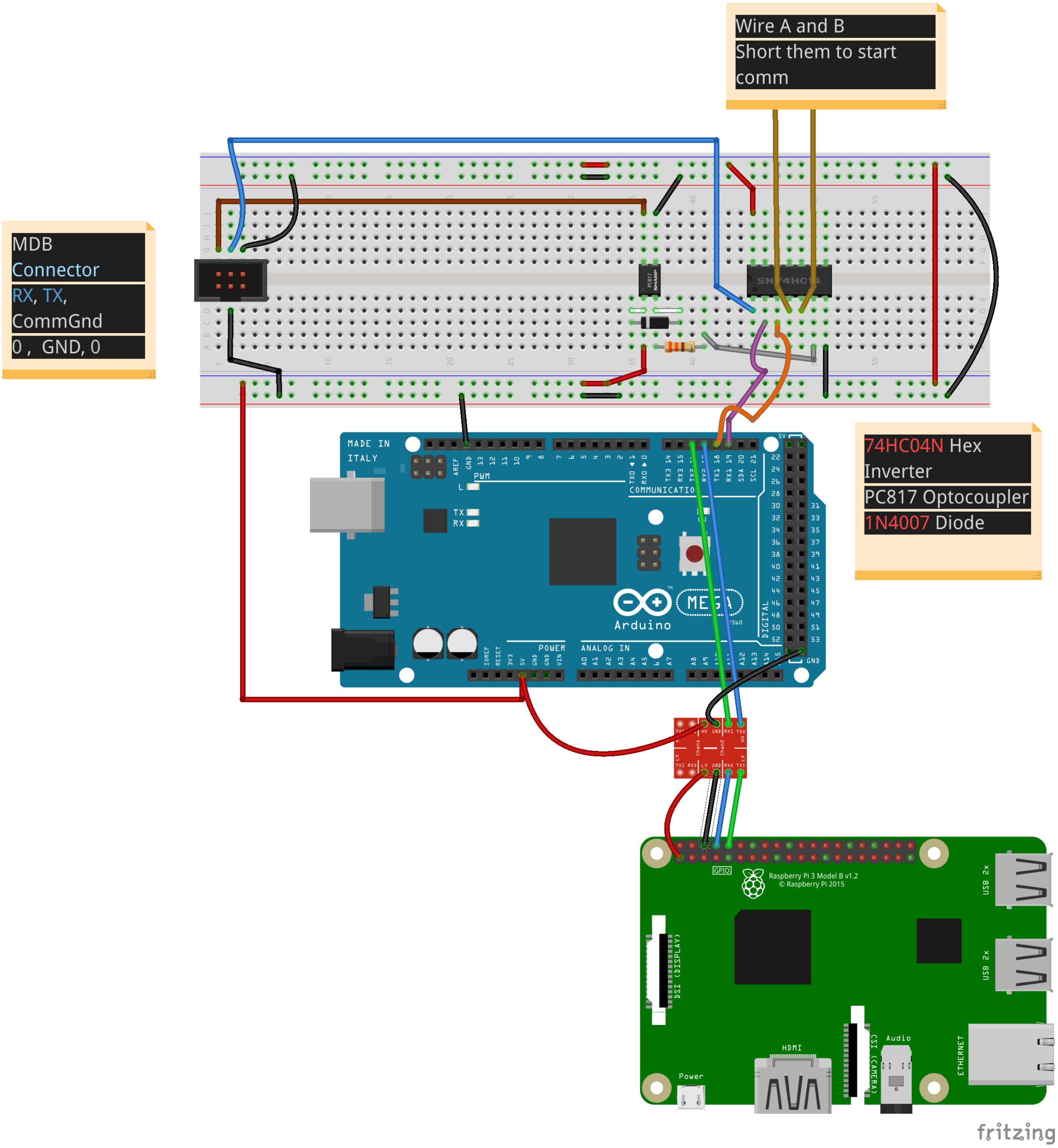


3.3v	5V
GPIO 2 Serial Data (I2C)	5V
GPIO3 Serial Clock (I2C)	Ground
GPIO 4	GPIO 14 (UART TX)
Ground	GPIO 15 (UART RX)
PIO 17 Chip Enable-CE1 (SPI1)	GPIO 18 Chip Enable-CE0 (SP1) [PWM]
GPIO 27	Ground
GPIO 22	GPIO 23
3.3v	GPIO 24
GPIO 10 MOSI (SPI 0)	Ground
GPIO 09 MISO (SPI 0)	GPIO 25
GPIO 11 SCLK (SPI 0)	GPIO 8 Chip Enable-CE0 (SPI0)
Ground	GPIO 7 Chip Enable-CE1(SPI0)
GPIO 0 EEPROM Serial DATA (I2C)	GPIO 1 EEPROM Serial Clock (I2C)
GPIO 5	Ground
GPIO 6	GPIO 12 (PWM)
GPIO 13 (PWM)	Ground
[PWM] GPIO 19 MISO (SPI 1)	GPIO 16 Chip Enable-CE2 (SPI 1)
GPIO 26	GPIO 20 MISO (SPI 1)
Ground	GPIO 21 SCLK (SPI 1)

- Arduino Mega Pinout


ARDUINO
ARDUINO
MEGA 2560 REV3
PB7 LED_BUILTIN
ATMEGA16U2
PD5 TX LED
PD4 RX LED
Power
NC
IOREF
RESET RESET
+3V3
+5V
GND
GND
VIN
ADC[0] PF0 A0 D54
ADC[1] PF1 A1 D55
ADC[2] PF2 A2 D56
ADC[3] PF3 A3 D57
ADC[4] PF4 A4 D58
ADC[5] PF5 A5 D59
ADC[6] PF6 A6 D60
ADC[7] PF7 A7 D61
ADC[8] PK0 A8 D62
ADC[9] PK1 A9 D63
ADC[10] PK2 A10 D64
ADC[11] PK3 A11 D65
ADC[12] PK4 A12 D66
ADC[13] PK5 A13 D67
ADC[14] PK6 A14 D68
ADC[15] PK7 A15 D69
D21/SCL PD0 SCL
D20/SDA PD1 SDA
AREF AREF
GND
~D13 PB7
~D12 PB6
~D11 PB5
~D10 PB4
~D9 PH6
~D8 PH5
~D7 PH4
~D6 PH3
~D5 PE3
~D4 PG5
~D3 PE5
~D2 PE4
~D1/TX0 PE1
~D0/RX0 PE0
D14/TX3 PJ1
D15/RX3 PJ0
D16/TX2 PH1
D17/RX2 PH0
D18/TX1 PD3
D19/RX1 PD2
D20/SDA PD1
D21/SCL PD0
ARDUINO.CC
■ Ground
■ Power
■ LED
■ Internal Pin
■ SWD Pin
□ Other Pin
▨ Digital Pin
▨ Analog Pin
■ Default
■ Microcontroller's Port
BY SA

6.1. Circuit



Wire A and B
Short them to start
comm

MDB
Connector
RX, TX,
CommGnd
0 , GND, 0

74HC04N Hex
Inverter
PC817 Optocoupler
1N4007 Diode

```
Pins connections
```

Arduino Mega	Logic Level Shifter
`TX2`	`RX1`
`GND`	`GND`
`HV`	`5V`
`RX2`	`TX0`

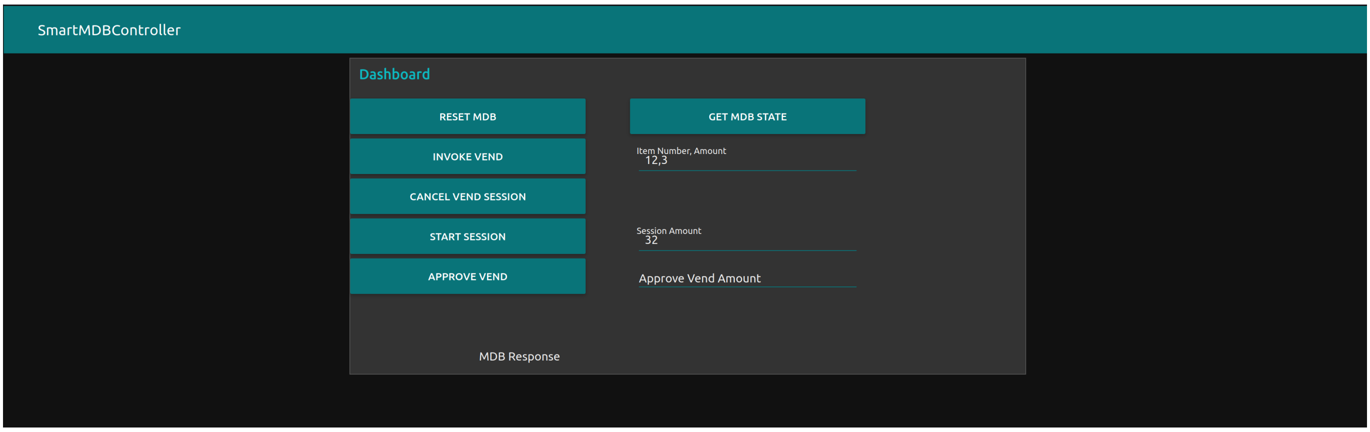
Logic Level Shifter	Raspberry Pi
`RX0`	`UART0_TX`
`GND`	`GND`
`LV`	`3.3V`
`TX1`	`UART0_RX`

SN74HC14	Arduino Mega
`2`	`RX1`
`3`	`TX1`
`4 & 5`	`Shorted`
`14`	`5V`

7. Dashboard

You can use the dashboard with the link below:

https://nodered-proxy.production.wrapdrive.tech/ui/#!/0



Item Number, Amount
12,3

Session Amount
32

Approve Vend Amount

```
- The dashboard is running on a production server that is meant for testing only.

-You should install and configure the dashboard on your Raspberry Pi or local server to keep it running.
```

7.1. Installing and Configuring Node-RED on Raspberry Pi

Conifguring NodeRED, MQTT is required only one time.

Open the terminal and execute the following commands

```
- sudo chmod a+rx starter.sh
- sudo apt install ufw
- sudo ufw enable
- sudo ufw allow tcp http https 1883 8883 1880
```

7.1.0.1. Installing MQTT(Mosquitto)

Open the terminal and execute the following commands

```
- sudo apt install -y mosquitto mosquitto-clients
- sudo systemctl enable mosquitto.service
- mosquitto -v
```

7.1.0.2. Installing and Configuring NodeRED

Open the terminal and execute the following commands

```
- bash <(curl -sL https://raw.githubusercontent.com/node-red/linux-
installers/master/deb/update-nodejs-and-nodered)
- sudo systemctl enable nodered.service
- npm install node-red-dashboard
- sudo npm install node-red-dashboard
- sudo systemctl restart nodered.service
```

Then open NodeRED in your raspberry pi or using any other device which is connected to the same network as your Raspberry Pi is. In the browser you can type http://raspberrypi.local:1880 to open the node-red

- Once node-red is opened, click on the menu button on the top left corner of the app and click on import.

- Click on `select file to import` and select flows.json present in the `dashboard` directory of this repo.
- After flows are imported, click on Deploy button on the top of the screen to save the changes.
- You can access the Dashboard using http://raspberrypi.local:1880/ui

8. Components Used

1. Raspberry Pi
2. Logic Level Converter
3. SN74HC14 Hex Inverter
4. PC817 Optocoupler
5. Arduino Mega
6. 330 Ohm Resistor
7. 1N4007 Diode

8.1. Datasheets

- Optocoupler **OHPC 817C F919G**
- - Datasheet
- Schimitt Trigger Hex Inverter **PHILIPS 74HC14N B9784PS Hnn9840 E**
- - Datasheet
- Diode **1N4007 MIC AXIAL SILASTIC GUARD JUNCTION STANDARD RECTIFIER**
- - Datasheet

9. ⛏️ Built Using

- Python3 - Raspberry Pi FW
- Node-RED - Platform for creating dashbaords and backend logic

10. ✍️ Authors

- @Nauman3S - Development and Deployment