

R-Pi

Team Emertxe



Interfacing

Rpi with NodeMCU using Serial Libraries



Interfacing

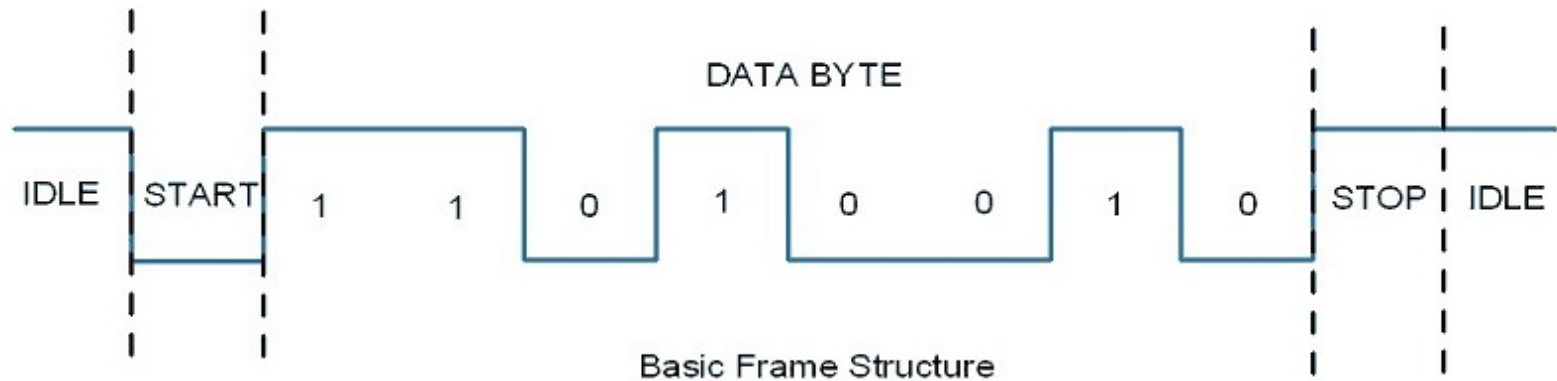
Introduction: UART



- UART (Universal Asynchronous Receiver/Transmitter) is a serial communication protocol in which data is transferred serially i.e. bit by bit.
- Asynchronous serial communication is widely used for byte oriented transmission.
 - ➔ In Asynchronous serial communication, a byte of data is transferred at a time.
- UART serial communication protocol uses a defined frame structure for their data bytes. Frame structure in Asynchronous communication consists:
 - ➔ START bit: It is a bit with which indicates that serial communication has started and it is always low.
 - ➔ Data bits packet: Data bits can be packets of 5 to 9 bits. Normally we use 8-bit data packet, which is always sent after the START bit.
 - ➔ STOP bit: This usually is one or two bits in length. It is sent after data bits packet to indicate the end of frame. Stop bit is always logic high.

Interfacing

Introduction: UART



Interfacing

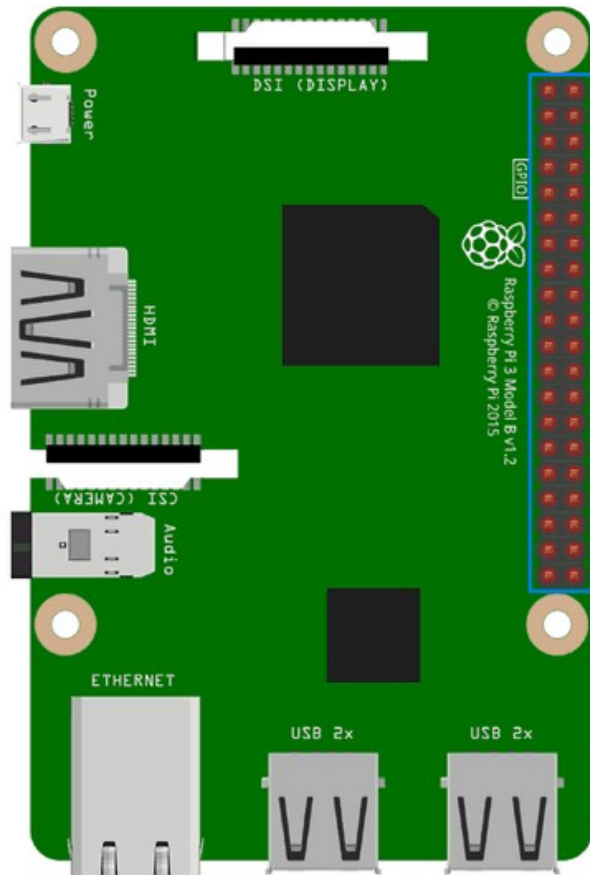
Introduction: Rpi UART



- Raspberry Pi has two in-built UART which are as follows:
 - ➔ PL011 UART
 - ➔ mini UART
- PL011 UART is an ARM based UART. This UART has better throughput than mini UART.
- In RPi-3, mini UART is used for Linux console output whereas PL011 is connected to the On-board Bluetooth module.
- And in the other versions of Raspberry Pi, PL011 is used for Linux console output.
- The PL011 is a stable and high performance UART.

Interfacing

Rpi UART pins



| | | | |
|--------------------|----|----|---------------------|
| 3.3V | 1 | 2 | 5V |
| GPIO2 (SDA1) | 3 | 4 | 5V |
| GPIO3 (SCL1) | 5 | 6 | GND |
| GPIO4 (GPIO_GCLK) | 7 | 8 | GPIO14 (UART_TXD0) |
| GND | 9 | 10 | GPIO15 (UART_RXD0) |
| GPIO17 (GPIO_GEN0) | 11 | 12 | GPIO18 (GPIO_GEN1) |
| GPIO27 (GPIO_GEN2) | 13 | 14 | GND |
| GPIO22 (GPIO_GEN3) | 15 | 16 | GPIO23 (GPIO_GEN4) |
| 3.3V | 17 | 18 | GPIO24 (GPIO_GEN\$) |
| GPIO10 (SPI0_MOSI) | 19 | 20 | GND |
| GPIO9 (SPI0_MISO) | 21 | 22 | GPIO25 (GPIO_GEN6) |
| GPIO11 (SPI0_CLK) | 23 | 24 | GPIO8 (SPI_CE0_N) |
| GND | 25 | 26 | GPIO7 (SPI_CE1_N) |
| ID_SD (I2C EEPROM) | 27 | 28 | ID_SC (I2C EEPROM) |
| GPIO5 | 29 | 30 | GND |
| GPIO6 | 31 | 32 | GPIO12 |
| GPIO13 | 33 | 34 | GND |
| GPIO19 | 35 | 36 | GPIO16 |
| GPIO26 | 37 | 38 | GPIO20 |
| GND | 39 | 40 | GPIO21 |

Interfacing

Configure UART



Interfacing

Step-1

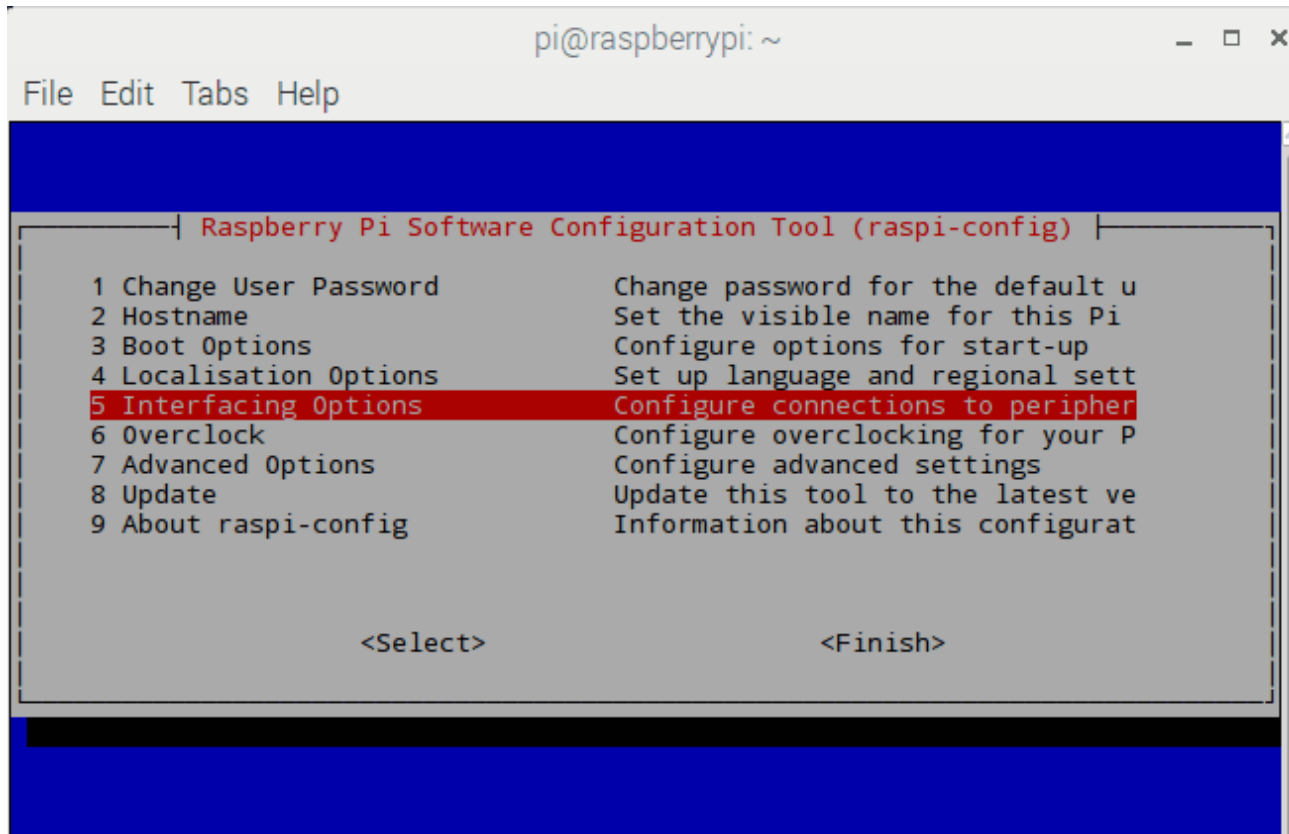
- Open the terminal and type

```
sudo raspi-config
```


Interfacing

Step-2

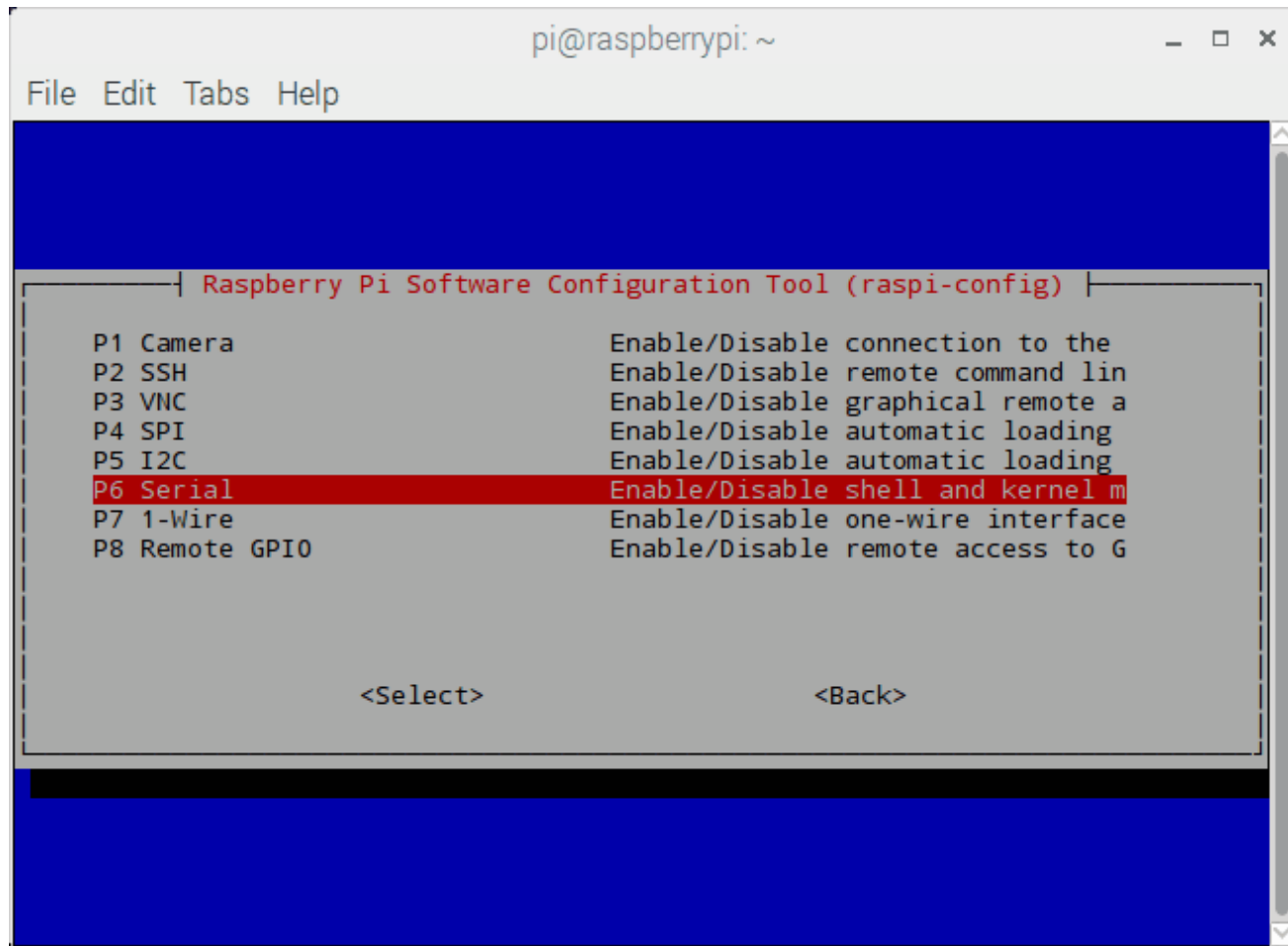
- Select -> Interfacing Options



Interfacing

Step-3

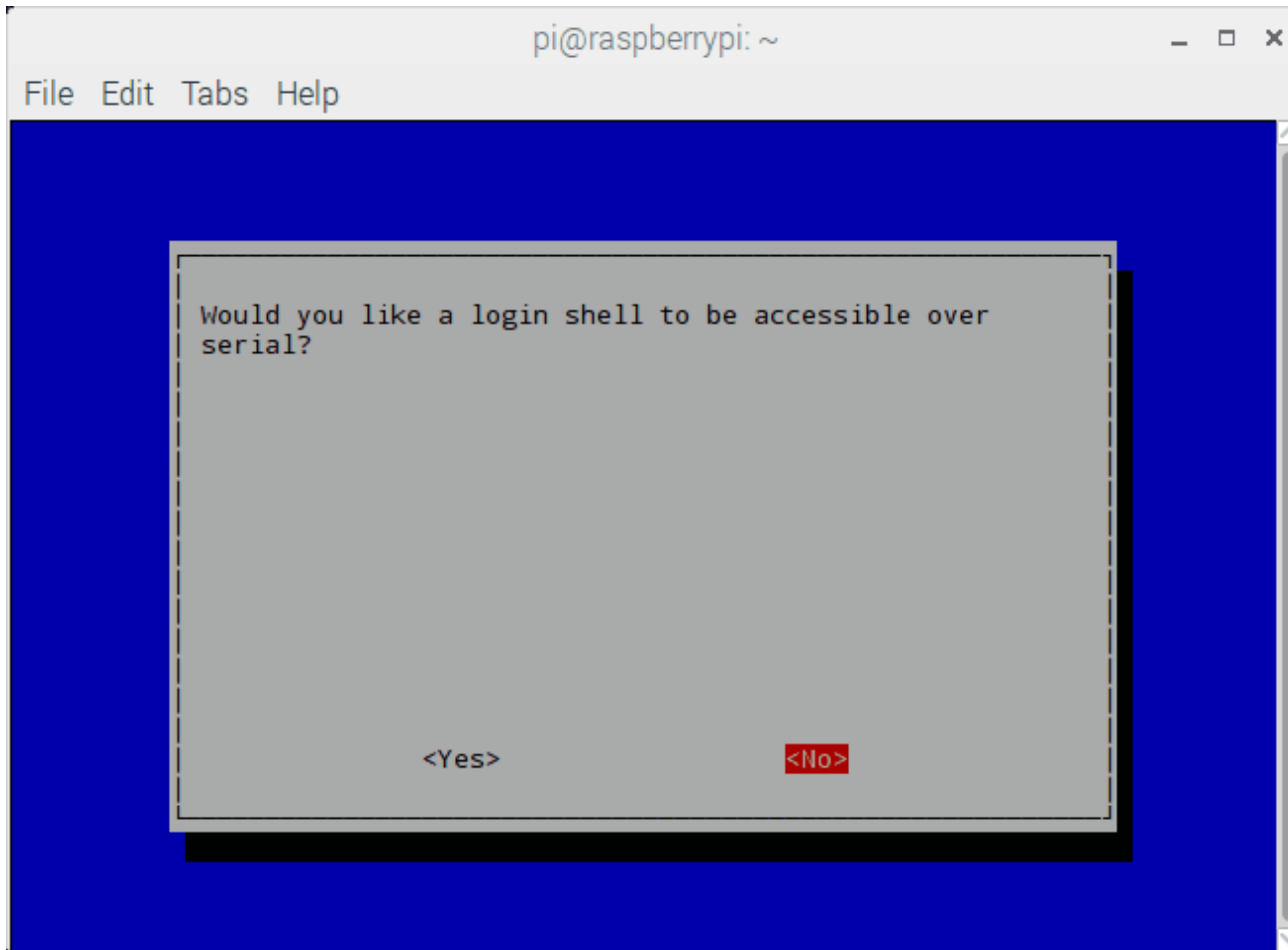
- select Serial option to enable UART



Interfacing

Step-4

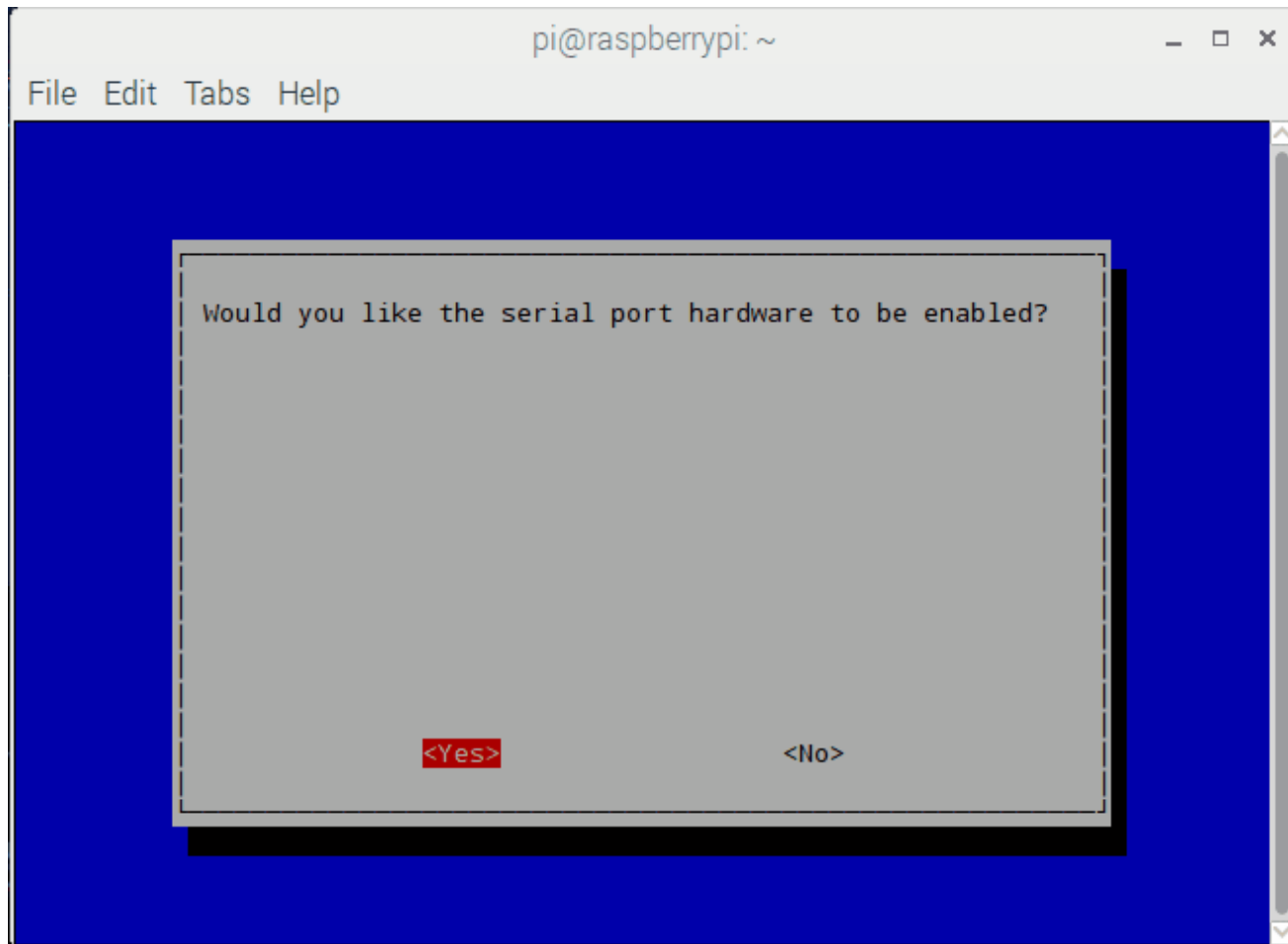
- select No



Interfacing

Step-5

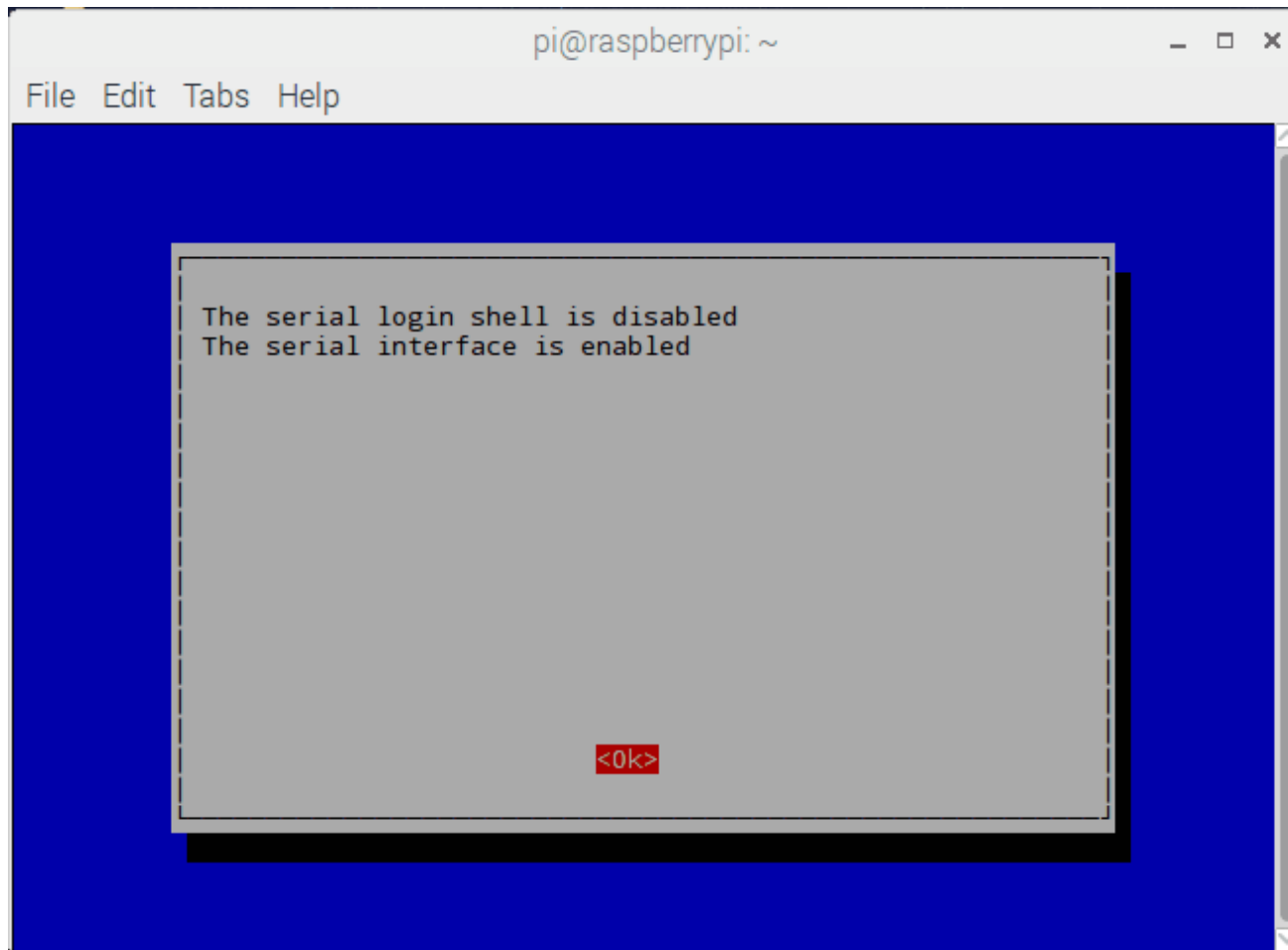
- select Yes



Interfacing

Step-6

- Finally, OK



Interfacing

Step-7

- At last, REBOOT the RPi

Interfacing

Coding: NodeMCU



Interfacing

Coding: NodeMCU



- Compile the below code in Arduino IDE and upload to nodeMCU

```
#include<SoftwareSerial.h> //Included SoftwareSerial Library
//Started SoftwareSerial at RX and TX pin of NodeMCU
SoftwareSerial s(16, 17);

void setup() {
  //Serial S Begin at 9600 Baud
  s.begin(9600);
}

void loop() {
  s.write("Hello");
  delay(500);
}
```

Note: If SoftwareSerial library not installed, install it.

Interfacing

Coding: RPi-3



Interfacing

Coding: RPi-3



- vi serial_reciever.py

```
import serial
from time import sleep

ser = serial.Serial("/dev/ttyS0", 9600)
ser.baudrate = 9600

while True:
    data = ser.read()
    sleep(0.5)
    data_left = ser.inWaiting()
    data += ser.read(data_left)
    print(data)
```

Interfacing

Connections



Interfacing

Coding: RPi-3

- Connect

| Arduino | RPi |
|---------|----------|
| Tx2 | UART0 RX |
| Rx2 | UART0 TX |
| GND | GND |

Interfacing

Coding: RPi-3

- Connect



Interfacing

Output



Interfacing Output



- After connections, run the codes in both the devices (NodeMCU and Rpi3)
- Press the reset (EN) button in NodeMCU
- Python3 serial_reciever.py
- Output is shown below,

```
pi@raspberrypi:~/emertxe_lab $ python3 serial_recieve.py
b'Hello'
b'Hello'
b'Hello'
b'Hello'
b'Hello'
b'Hello'
b'Hello'
```

THANK YOU