Function description for RS485 Raspberry Pi 3 B to Teensy4.0, V1.00

2022-04-16, Armin Rehberger

Code for Teensy4.0

```
// ##### Include
#include <LiquidCrystal_I2C.h> // For LCD display HD44780 2004 LCD, 4x20 characters

// ##### Configure LCD display, address 0x27 (39), 4x20 Zeichen
LiquidCrystal_I2C lcd(0x27,20,4);

// ##### Setup
void setup()
{
  // ##### Initialize USB Serial port
  Serial.begin(9600);
  Serial.println("Startup");

  // ##### Initialize serial 1, UART port, RS485
  // UART, Universal Asynchronous Receiver / Transmitter
  // RX default pin 0
  // TX default pin 1
  // Transmit Enable could be any pin
  #define HWSERIAL Serial1
  HWSERIAL.setRX(0); // Pin 0 = RX
  HWSERIAL.setTX(1); // Pin 1 = TX
  HWSERIAL.transmitterEnable(2); // Pin 2 = Transmitter enable
  HWSERIAL.setTimeout(1000); // Read timeout value in ms
  HWSERIAL.begin(115200); // 115200 9600 baud

  // ##### Digital outputs
  pinMode(LED_BUILTIN, OUTPUT); // LED on board

  // ##### Initialize LCD display, 4x20 digits
  lcd.init();
  lcd.backlight();
}

// ##### Loop
void loop()
{
  // ##### Variables
  static bool ledon = false;
  static int BytesRecieved = 0;
  bool WriteData = false;
  static int incomingByteUSB = 0;
  int i;

  // ##### USB read serial port
  if (Serial.available() > 0)
  {
    incomingByteUSB = Serial.read();
    WriteData = true; // Start RS485 write data
  }
```

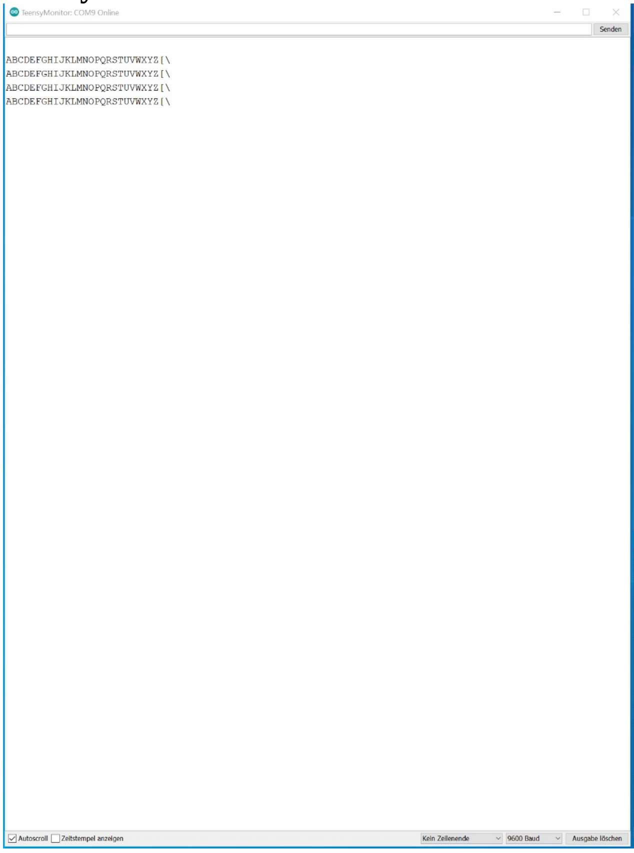
```
 // ##### RS485 write data
 if(WriteData == true)
 {
   WriteData = false;
   // Teensy → Raspberry, Byte 0..22, 23 Byte + \r + \n = 25 Bytes
   // \r = carriage return (Dec 13), \n = linefeed (Dec 24) is added automatically
   char t[25]= "00000000000000000000000"; // Empty array where to put the data

   // Write anything to send data, 23 Byte, Teensy -> Raspberry
   t[0] = 97; // Dec 97 = a
   t[1] = 98;
   t[2] = 99;
   t[3] = 100;
   t[4] = 101;
   t[5] = 102;
   t[6] = 103;
   t[7] = 104;
   t[8] = 105;
   t[9] = 106;
   t[10] = 107;
   t[11] = 108;
   t[12] = 109;
   t[13] = 110;
   t[14] = 111;
   t[15] = 112;
   t[16] = 113;
   t[17] = 114;
   t[18] = 115;
   t[19] = 116;
   t[20] = 117;
   t[21] = 118;
   t[22] = 119;

   HWSERIAL.println(t); // Send string with carriage return and linefeed character \r\n

   // Toggle LED on board
   ledon = ! ledon;
   digitalWrite(LED_BUILTIN, ledon);
 }

 // ##### RS485 read data
 if (HWSERIAL.available() > 0)
 {
   // Buffer for the received data
   const int BUFFER_SIZE = 100;
   char buf[BUFFER_SIZE];

   // Read bytes until
   // 1. The terminator character is detected (LineFeed \n). The terminator itself is not
returned in the buffer.
   // 2. BUFFER_SIZE (100) has reached
   // 3. It times out (1000ms),
   BytesRecieved = HWSERIAL.readBytesUntil('\n', buf, BUFFER_SIZE);
   HWSERIAL.clear(); // Discard any received data that has not been read

   // Print the received data to USB serial port
   for(i = 0; i < BytesRecieved; i++)
     Serial.print(buf[i]);
   Serial.println("");

   // Toggle LED on board
   ledon = ! ledon;
   digitalWrite(LED_BUILTIN, ledon);
 }

 // ##### LCD Display
 static char str[80] = "";

 sprintf(str,"RS485 received: %i", BytesRecieved);
 lcd.setCursor(0,0);
 lcd.print(str);
}
```

Code for Raspberry Pi 3 B

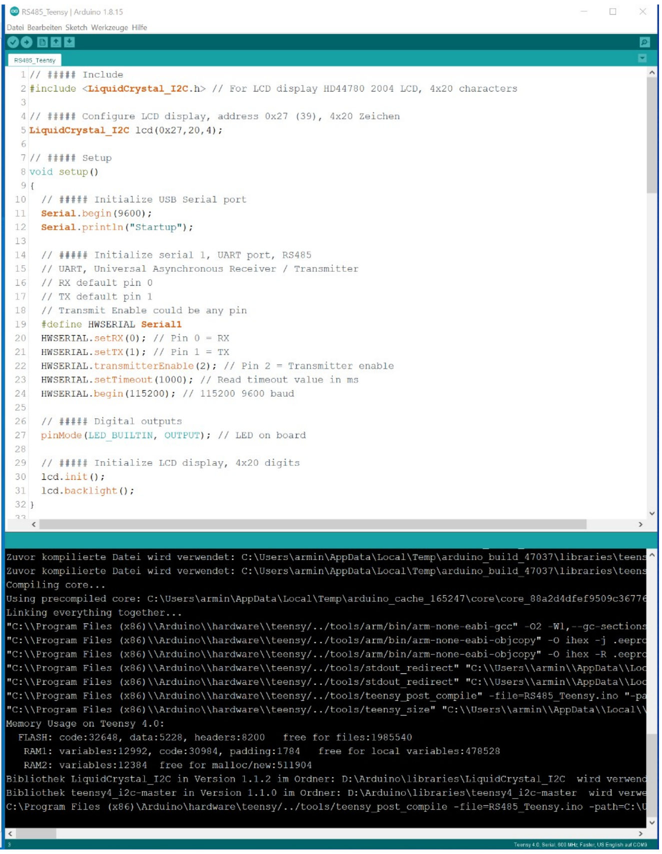
```
""" ######################################################## """
""" Import python modules """
import time
import serial
import RPi.GPIO as GPIO

""" ######################################################## """
""" GPIO """
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD) # GPIO = Pin numbers
# RS485 Transmitter enable. Pin number 7, GPIO4. DE & RE of RS-485 module
# Output low=receive, high=send
GPIO.setup(7, GPIO.OUT, initial=GPIO.LOW)

""" ######################################################## """
""" Initialize serial 0, RS485 """
# 8N1, 8 Data Bits, No Parity, 1 Stopbit
# 115200 Baud,
# Read timeout 1.0s
HWSERIAL = serial.Serial(
   port='/dev/serial0',
   baudrate = 115200,
   parity=serial.PARITY_NONE,
   stopbits=serial.STOPBITS_ONE,
   bytesize=serial.EIGHTBITS,
   timeout=1.0 # Read timeout value in seconds. Value = float
)

""" ######################################################## """
""" Global variables """
NumbersByteWritten = 0
ByteReceived = 0
WriteData = False

# Bytearray 0..22, 23 Byte receive from teensy
ReadDataMotor = bytearray([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, # 23 Byte read data
                    0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
                    0, 0, 0])

# Bytearray 0..29, 30 Byte send to teensy
WriteDataMotor = bytearray([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, # 30 Byte write data
                    0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
                    0, 0, 0, 0, 0, 0, 0, 0, 0, 0])

# Write anything to send data, 30 Byte, Raspberry -> Teensy
WriteByte0 = 65 # Dec 65 = A
WriteByte1 = 66
WriteByte2 = 67
WriteByte3 = 68
WriteByte4 = 69
```

```
WriteByte5 = 70
WriteByte6 = 71
WriteByte7 = 72
WriteByte8 = 73
WriteByte9 = 74
WriteByte10 = 75
WriteByte11 = 76
WriteByte12 = 77
WriteByte13 = 78
WriteByte14 = 79
WriteByte15 = 80
WriteByte16 = 81
WriteByte17 = 82
WriteByte18 = 83
WriteByte19 = 84
WriteByte20 = 85
WriteByte21 = 86
WriteByte22 = 87
WriteByte23 = 88
WriteByte24 = 89
WriteByte25 = 90
WriteByte26 = 91
WriteByte27Checksum = 92
WriteByte28CR = 13 # Carriage return
WriteByte29LF = 10 # Line feed

""" #################################################################################### """
""" Start loop forever """
while True:

""" #################################################################################### """
    """ RS485 read data """
    # Read a line including \n
    # Teensy -> Raspberry, Byte 0..22, 23 Bytes + \r + \n = 25 Bytes
    # \r = carraige return (Dec 13), \n = linefeed (Dec 10)
    # Read timeout must be set in Serial, if \n is missing in the string
    # Return value is a string
    # In Teensy use println, \r + \n is added automatically
    line = HWSERIAL.readline() # Read a line including \n
    HWSERIAL.reset_input_buffer() # Clear input buffer
    if line:
        ByteReceived = len(line)
        if ByteReceived == 25 and line[23] == 13 and line[24] == 10:
            WriteData = True
            print(line)
            for i in range(0, 23): # Copy Bytes 0..22, 23 Bytes to receive array
                ReadDataMotor[i] = line[i]

""" #################################################################################### """
    """ RS485 write data """
    # Write a bytearray
    # Raspberry -> Teensy, Byte 0..29, 30 Byte
    # In Teensy use readBytesUntil
    if WriteData == True:
        WriteData = False

        # Write the send data to a bytearray, 0..29, 30 Byte
        WriteDataMotor = [WriteByte0, WriteByte1, WriteByte2, WriteByte3, WriteByte4,
                          WriteByte5, WriteByte6, WriteByte7, WriteByte8, WriteByte9,
                          WriteByte10, WriteByte11, WriteByte12, WriteByte13, WriteByte14,
                          WriteByte15, WriteByte16, WriteByte17, WriteByte18, WriteByte19,
                          WriteByte20, WriteByte21, WriteByte22, WriteByte23, WriteByte24,
                          WriteByte25, WriteByte26, WriteByte27Checksum, WriteByte28CR,
WriteByte29LF]

        GPIO.output(7, True) # RS485 Transmitter enable, high=send
        HWSERIAL.reset_output_buffer() # Clear output buffer
        NumbersByteWritten = HWSERIAL.write(WriteDataMotor) # Send bytearray
        time.sleep(0.05)
        GPIO.output(7, False) # RS485 Transmitter enable, low=receive

""" ######################################################################## """
""" End loop forever """
GPIO.output(7, False)
GPIO.cleanup()
```

Teensy4.0 with Arduino V1.8.15



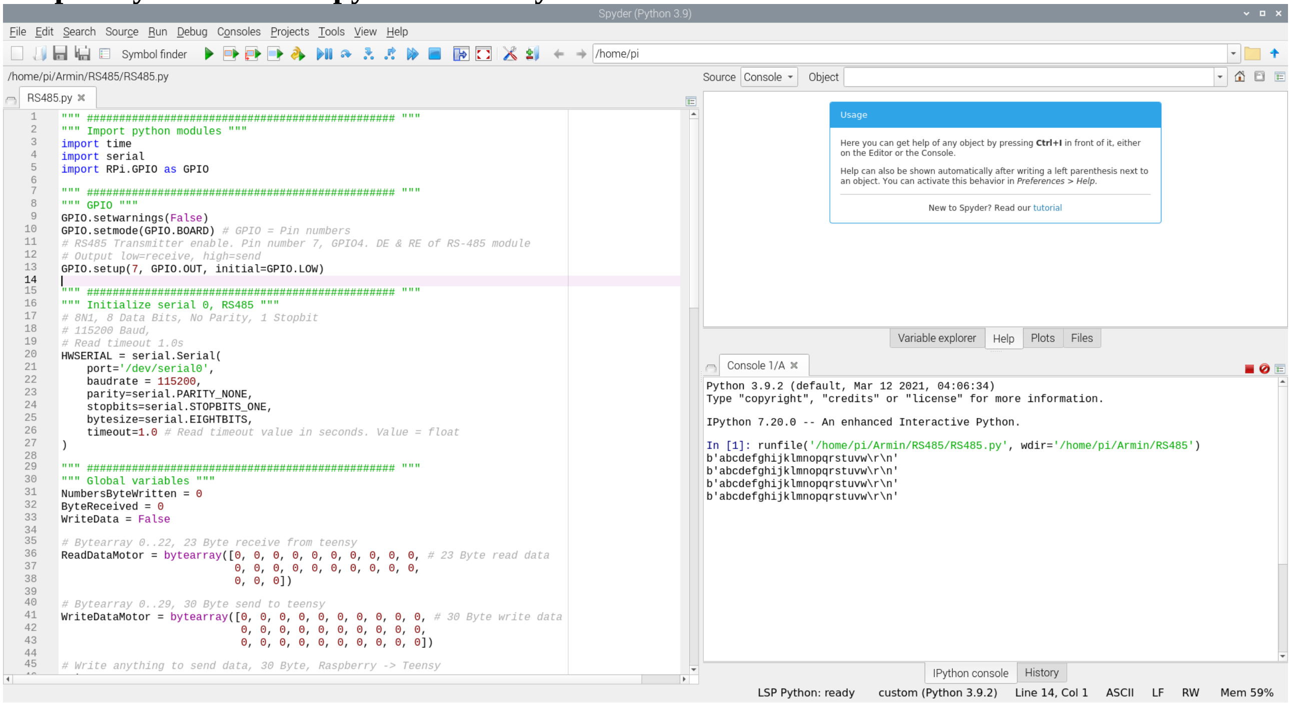
```
ABCDEFGHIJKLMNOPQRSTUVWXYZ[\
ABCDEFGHIJKLMNOPQRSTUVWXYZ[\
ABCDEFGHIJKLMNOPQRSTUVWXYZ[\
ABCDEFGHIJKLMNOPQRSTUVWXYZ[\
```


RS485_Teensy | Arduino 1.8.15
Datei Bearbeiten Sketch Werkzeuge Hilfe
RS485_Teensy
Teensy 4.0, Serial, 600 MHz, Faster, US English auf COM9

```
// ##### Include
#include <LiquidCrystal_I2C.h> // For LCD display HD44780 2004 LCD, 4x20 characters

// ##### Configure LCD display, address 0x27 (39), 4x20 Zeichen
LiquidCrystal_I2C lcd(0x27,20,4);

// ##### Setup
void setup()
{
  // ##### Initialize USB Serial port
  Serial.begin(9600);
  Serial.println("Startup");

  // ##### Initialize serial 1, UART port, RS485
  // UART, Universal Asynchronous Receiver / Transmitter
  // RX default pin 0
  // TX default pin 1
  // Transmit Enable could be any pin
  #define HWSERIAL Serial1
  HWSERIAL.setRX(0); // Pin 0 = RX
  HWSERIAL.setTX(1); // Pin 1 = TX
  HWSERIAL.transmitterEnable(2); // Pin 2 = Transmitter enable
  HWSERIAL.setTimeout(1000); // Read timeout value in ms
  HWSERIAL.begin(115200); // 115200 9600 baud

  // ##### Digital outputs
  pinMode(LED_BUILTIN, OUTPUT); // LED on board

  // ##### Initialize LCD display, 4x20 digits
  lcd.init();
  lcd.backlight();
}
```

```
Zuvor kompilierte Datei wird verwendet: C:\Users\armin\AppData\Local\Temp\arduino_build_47037\libraries\teens
Zuvor kompilierte Datei wird verwendet: C:\Users\armin\AppData\Local\Temp\arduino_build_47037\libraries\teens
Compiling core...
Using precompiled core: C:\Users\armin\AppData\Local\Temp\arduino_cache_165247\core\core_88a2d4dfef9509c36776
Linking everything together...
"C:\\Program Files (x86)\\Arduino\\hardware\\teensy/../tools/arm/bin/arm-none-eabi-gcc" -O2 -Wl,--gc-sections
"C:\\Program Files (x86)\\Arduino\\hardware\\teensy/../tools/arm/bin/arm-none-eabi-objcopy" -O ihex -j .eepro
"C:\\Program Files (x86)\\Arduino\\hardware\\teensy/../tools/arm/bin/arm-none-eabi-objcopy" -O ihex -R .eepro
"C:\\Program Files (x86)\\Arduino\\hardware\\teensy/../tools/stdout_redirect" "C:\\Users\\armin\\AppData\\Loc
"C:\\Program Files (x86)\\Arduino\\hardware\\teensy/../tools/stdout_redirect" "C:\\Users\\armin\\AppData\\Loc
"C:\\Program Files (x86)\\Arduino\\hardware\\teensy/../tools/teensy_post_compile" -file=RS485_Teensy.ino "-pa
"C:\\Program Files (x86)\\Arduino\\hardware\\teensy/../tools/teensy_size" "C:\\Users\\armin\\AppData\\Local\\
Memory Usage on Teensy 4.0:
  FLASH: code:32648, data:5228, headers:8200   free for files:1985540
   RAM1: variables:12992, code:30984, padding:1784   free for local variables:478528
   RAM2: variables:12384  free for malloc/new:511904
Bibliothek LiquidCrystal_I2C in Version 1.1.2 im Ordner: D:\Arduino\libraries\LiquidCrystal_I2C  wird verwend
Bibliothek teensy4_i2c-master in Version 1.1.0 im Ordner: D:\Arduino\libraries\teensy4_i2c-master  wird verwe
C:\Program Files (x86)\Arduino\hardware\teensy/../tools/teensy_post_compile -file=RS485_Teensy.ino -path=C:\U
```

Raspberry Pi 3 B with Spyder V4.2.1 Python 3.9.2


Spyder (Python 3.9)
File Edit Search Source Run Debug Consoles Projects Tools View Help
Symbol finder
/home/pi
/home/pi/Armin/RS485/RS485.py
RS485.py
Source Console Object
Usage
Variable explorer Help Plots Files
Console 1/A
IPython console History
LSP Python: ready custom (Python 3.9.2) Line 14, Col 1 ASCII LF RW Mem 59%

```
""" ################################################ """
""" Import python modules """
import time
import serial
import RPi.GPIO as GPIO

""" ################################################ """
""" GPIO """
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD) # GPIO = Pin numbers
# RS485 Transmitter enable. Pin number 7, GPIO4. DE & RE of RS-485 module
# Output low=receive, high=send
GPIO.setup(7, GPIO.OUT, initial=GPIO.LOW)

""" ################################################ """
""" Initialize serial 0, RS485 """
# 8N1, 8 Data Bits, No Parity, 1 Stopbit
# 115200 Baud,
# Read timeout 1.0s
HWSERIAL = serial.Serial(
    port='/dev/serial0',
    baudrate = 115200,
    parity=serial.PARITY_NONE,
    stopbits=serial.STOPBITS_ONE,
    bytesize=serial.EIGHTBITS,
    timeout=1.0 # Read timeout value in seconds. Value = float
)

""" ################################################ """
""" Global variables """
NumbersByteWritten = 0
ByteReceived = 0
WriteData = False

# Bytearray 0..22, 23 Byte receive from teensy
ReadDataMotor = bytearray([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, # 23 Byte read data
                           0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
                           0, 0, 0])

# Bytearray 0..29, 30 Byte send to teensy
WriteDataMotor = bytearray([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, # 30 Byte write data
                            0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
                            0, 0, 0, 0, 0, 0, 0, 0, 0, 0])

# Write anything to send data, 30 Byte, Raspberry -> Teensy
```

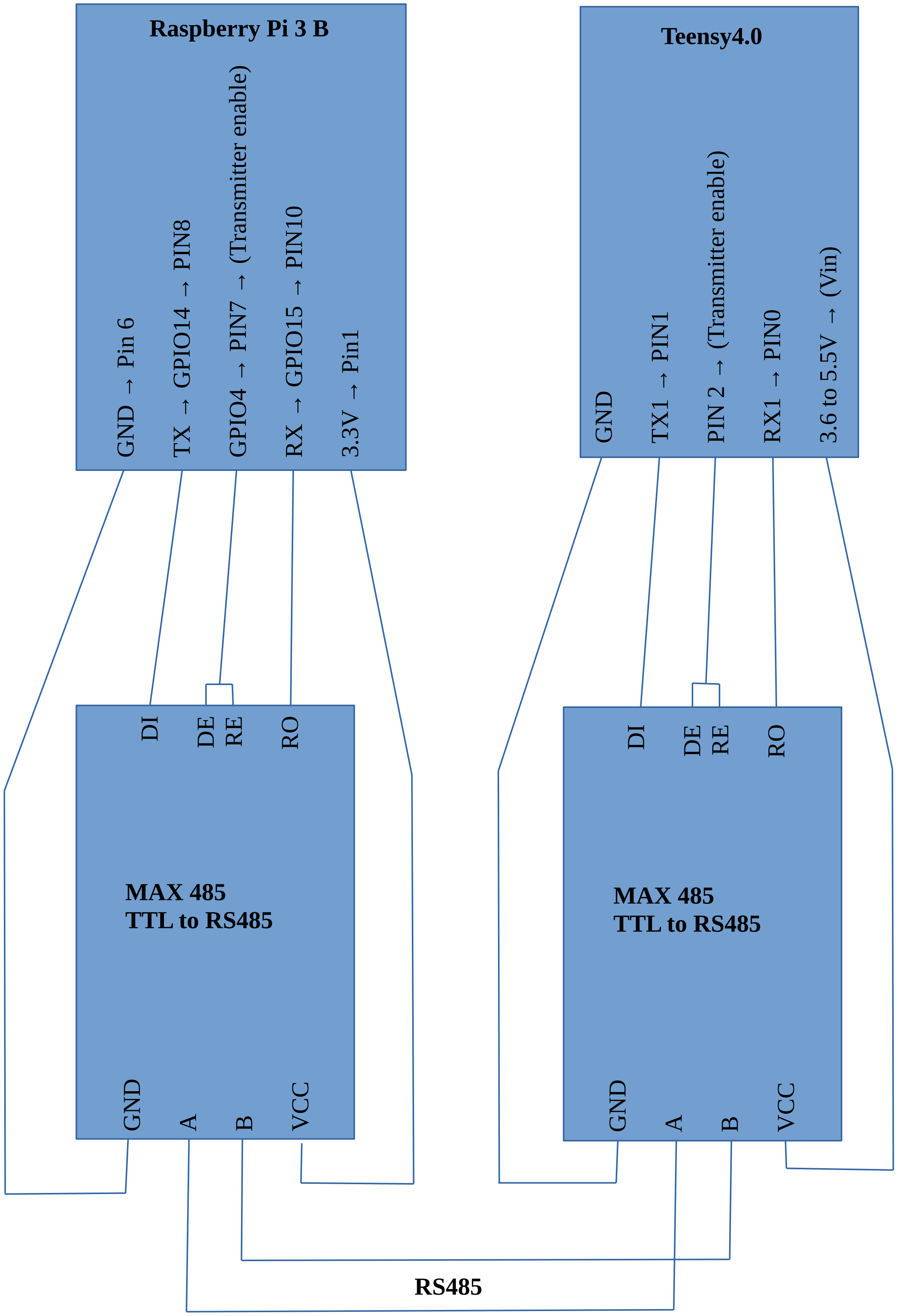
Here you can get help of any object by pressing **Ctrl+I** in front of it, either on the Editor or the Console.

Help can also be shown automatically after writing a left parenthesis next to an object. You can activate this behavior in *Preferences > Help*.

New to Spyder? Read our tutorial

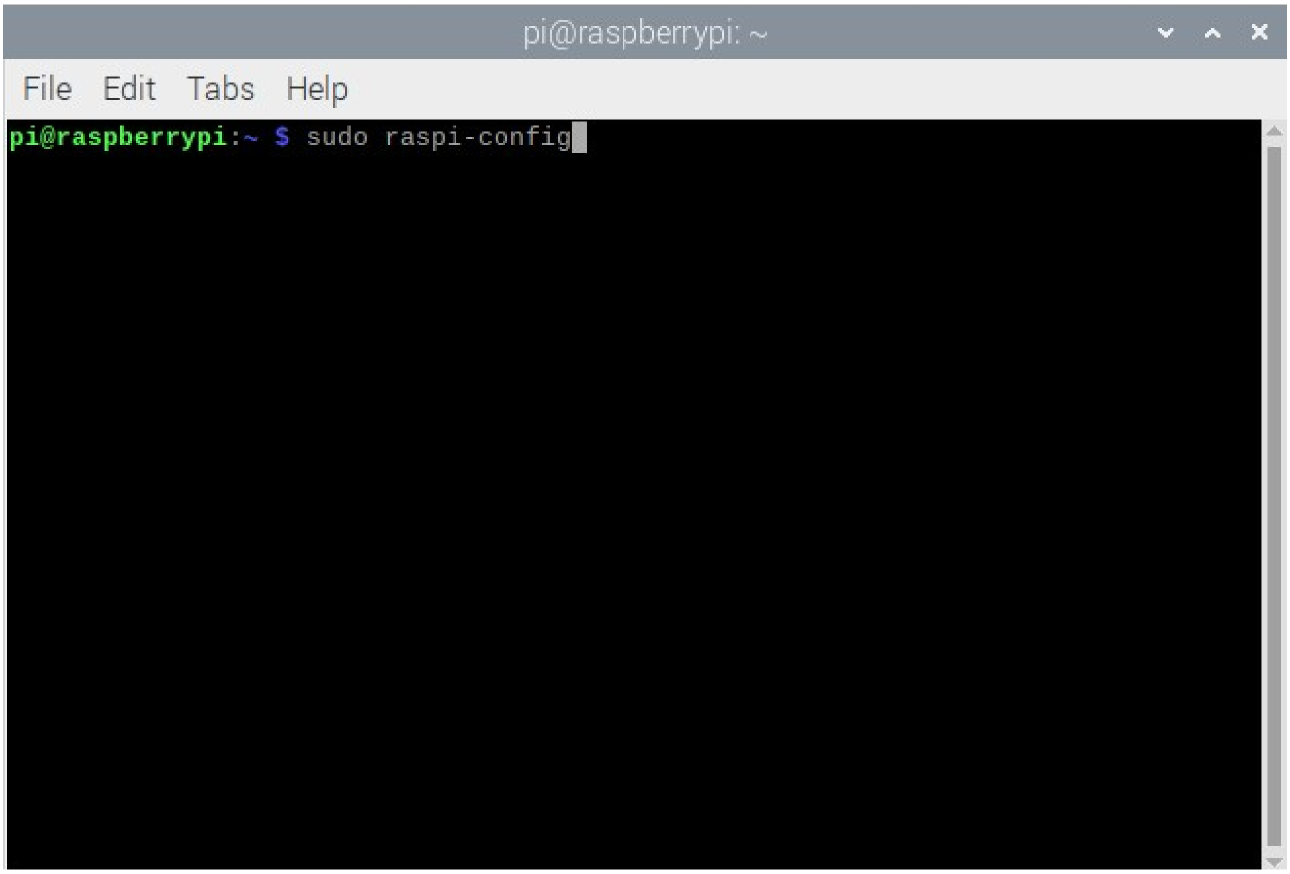
```
Python 3.9.2 (default, Mar 12 2021, 04:06:34)
Type "copyright", "credits" or "license" for more information.

IPython 7.20.0 -- An enhanced Interactive Python.

In [1]: runfile('/home/pi/Armin/RS485/RS485.py', wdir='/home/pi/Armin/RS485')
b'abcdefghijklmnopqrstuvw\r\n'
b'abcdefghijklmnopqrstuvw\r\n'
b'abcdefghijklmnopqrstuvw\r\n'
b'abcdefghijklmnopqrstuvw\r\n'
```

Wiring Raspberry Pi 3 B → MAX 485 → MAX 485 → Teensy4.0

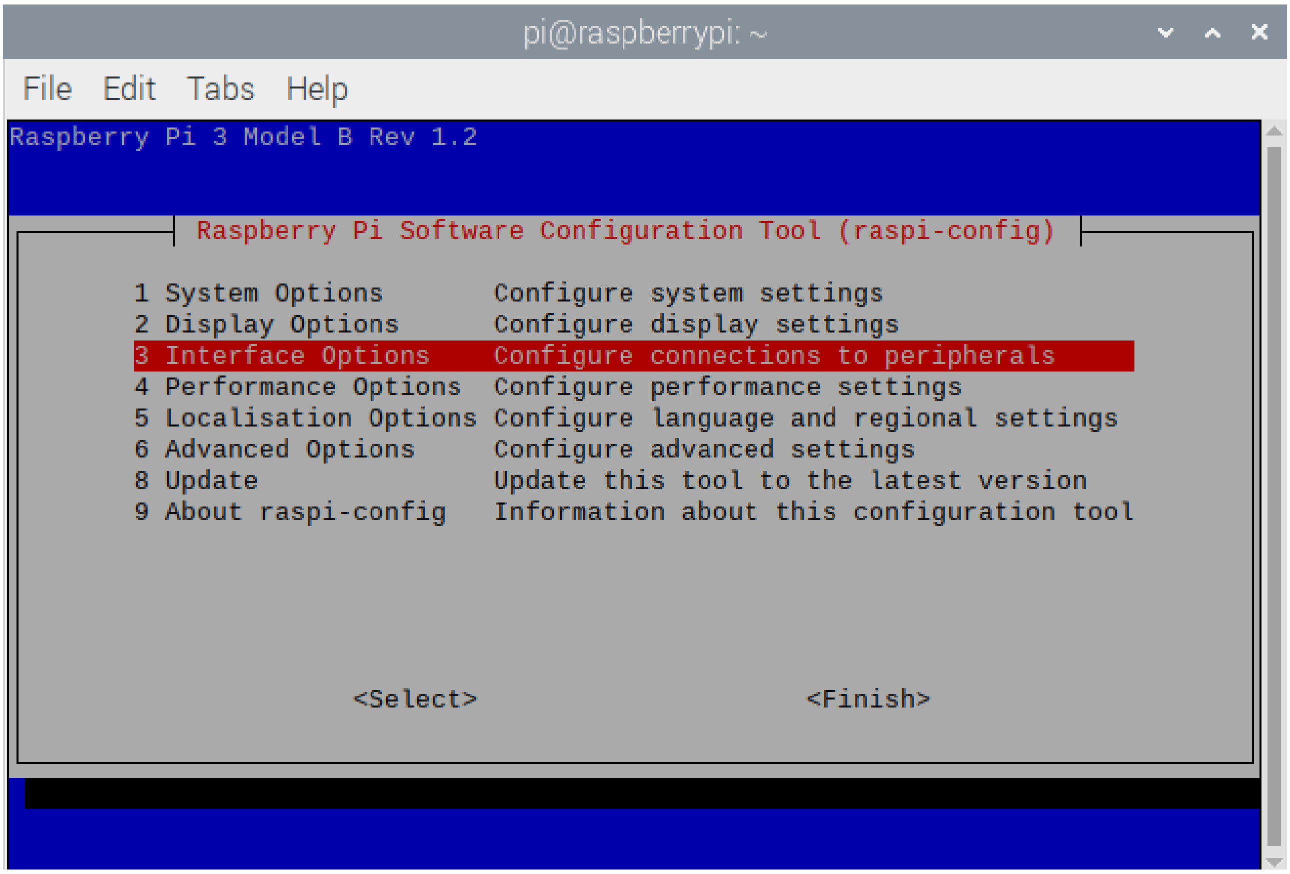


Settings serial port Raspberry Pi 3 B with Linux11 (bullseye)

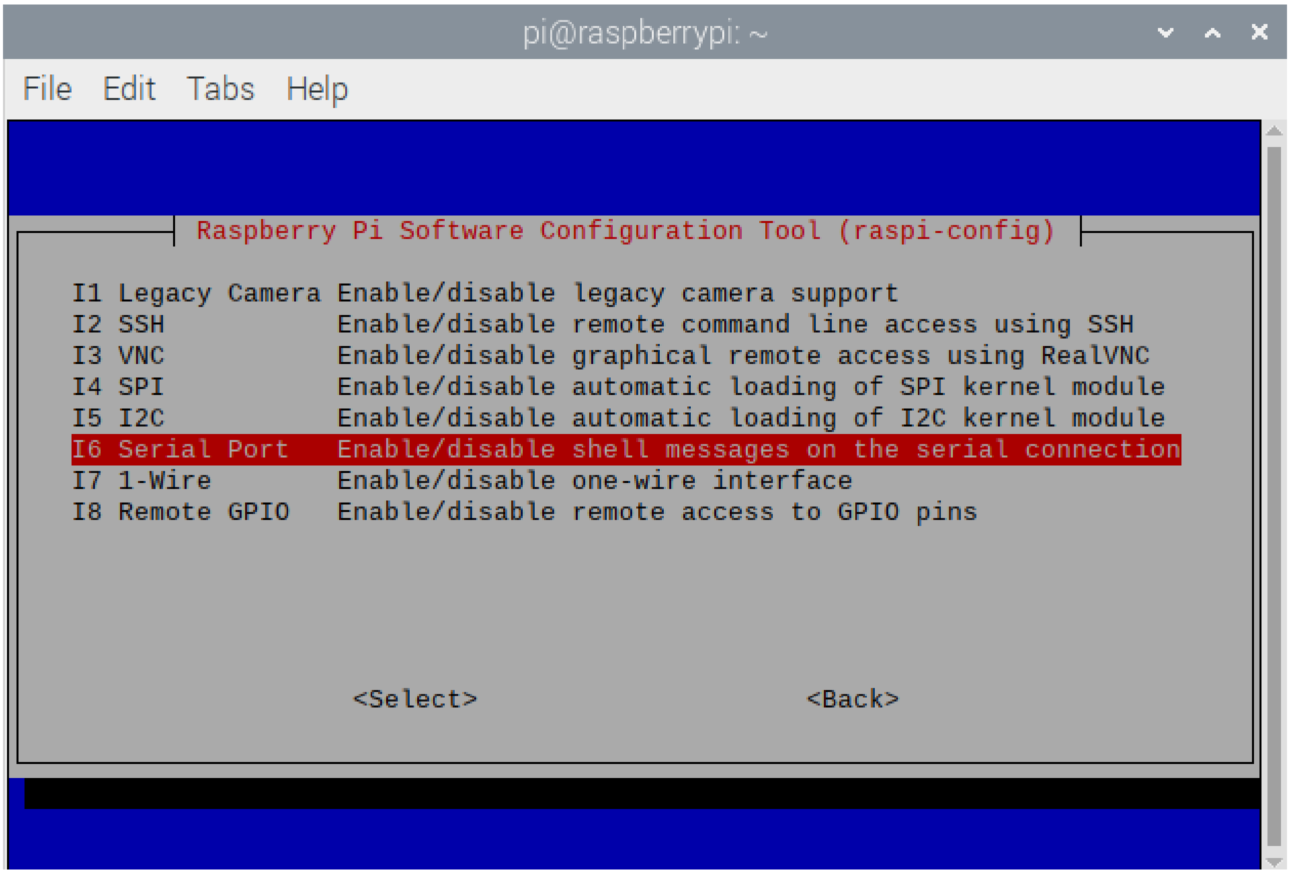
Open a terminal and type sudo raspi-config



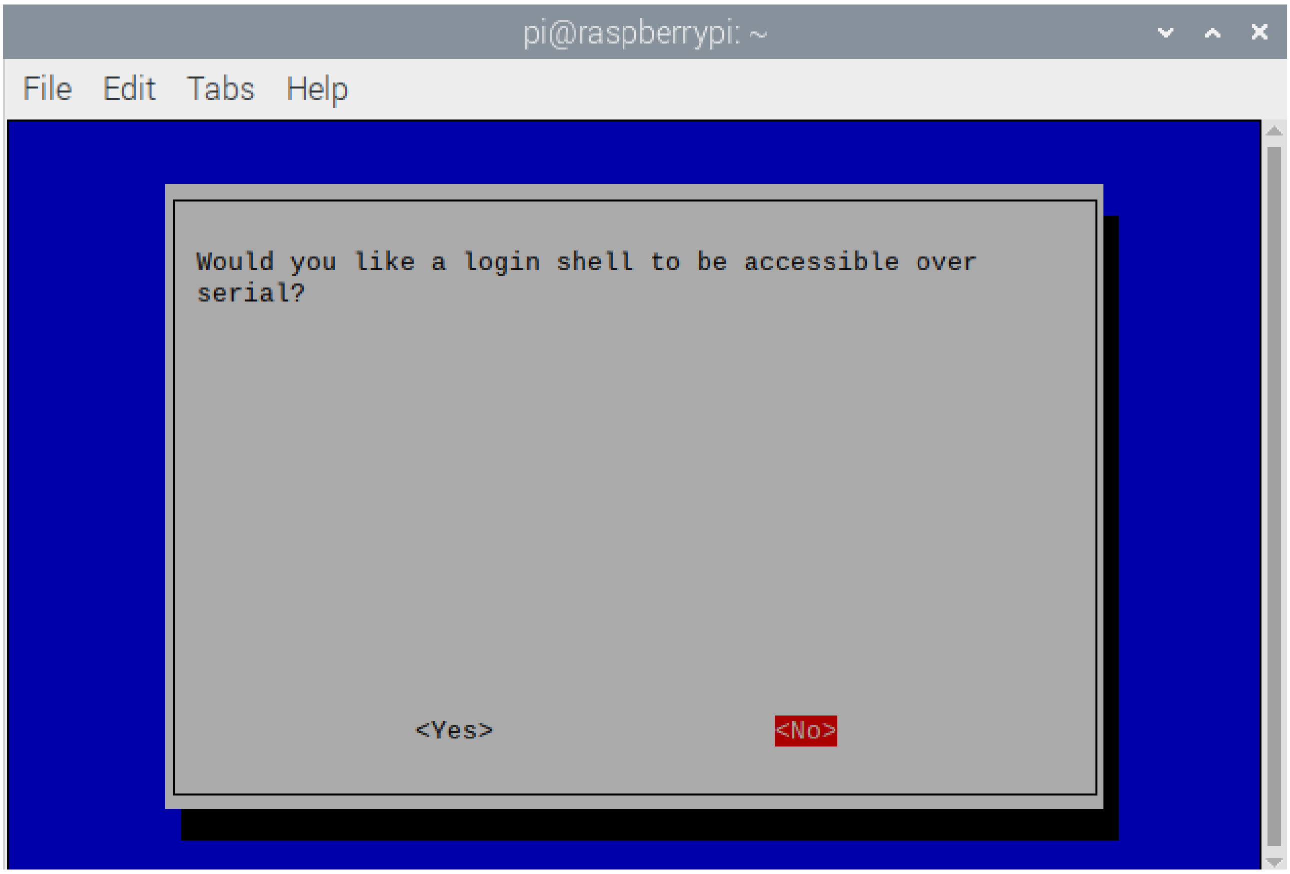
```
pi@raspberrypi:~ $ sudo raspi-config
```



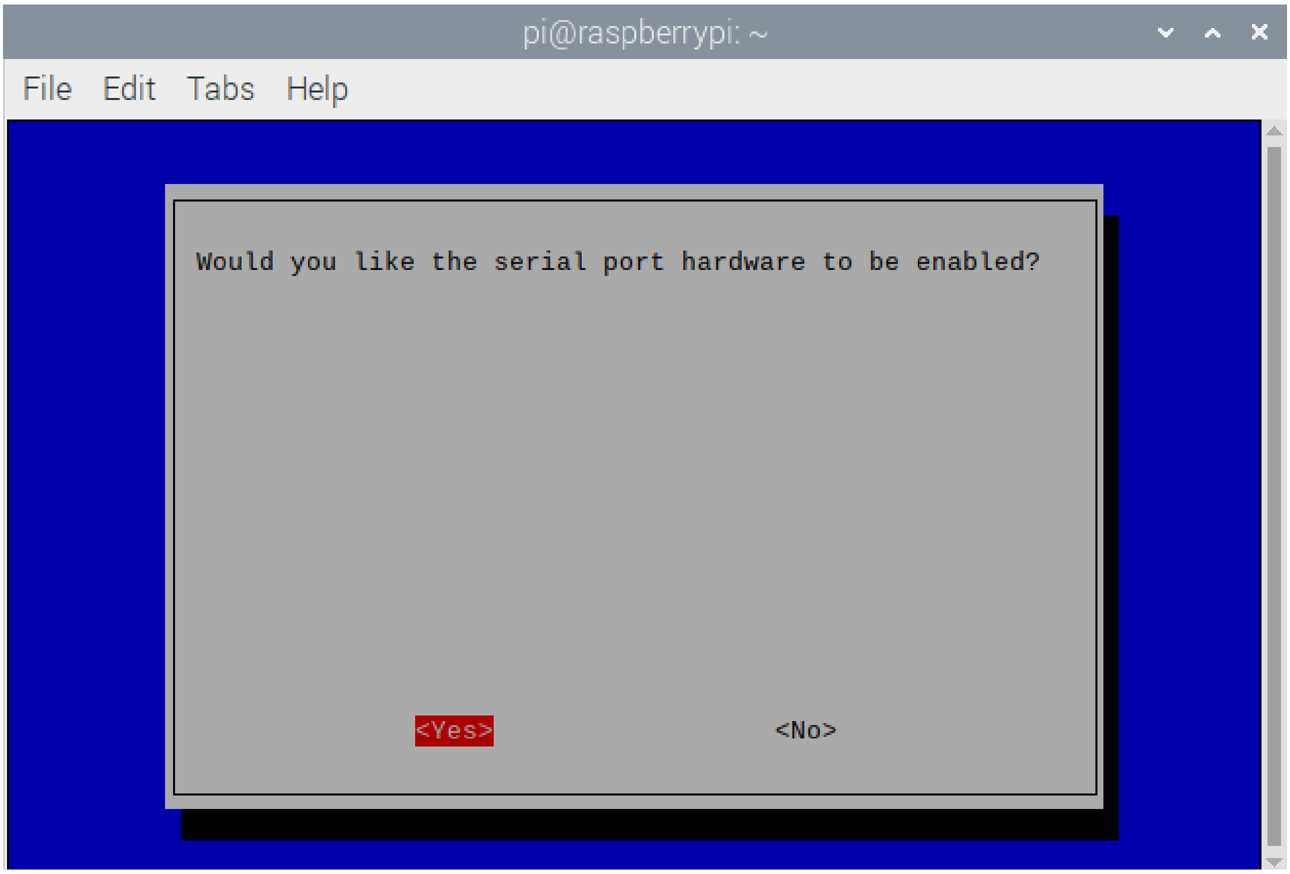
```
Raspberry Pi 3 Model B Rev 1.2

        Raspberry Pi Software Configuration Tool (raspi-config)

       1 System Options       Configure system settings
       2 Display Options      Configure display settings
       3 Interface Options    Configure connections to peripherals
       4 Performance Options  Configure performance settings
       5 Localisation Options Configure language and regional settings
       6 Advanced Options     Configure advanced settings
       8 Update               Update this tool to the latest version
       9 About raspi-config   Information about this configuration tool

                    <Select>                    <Finish>
```



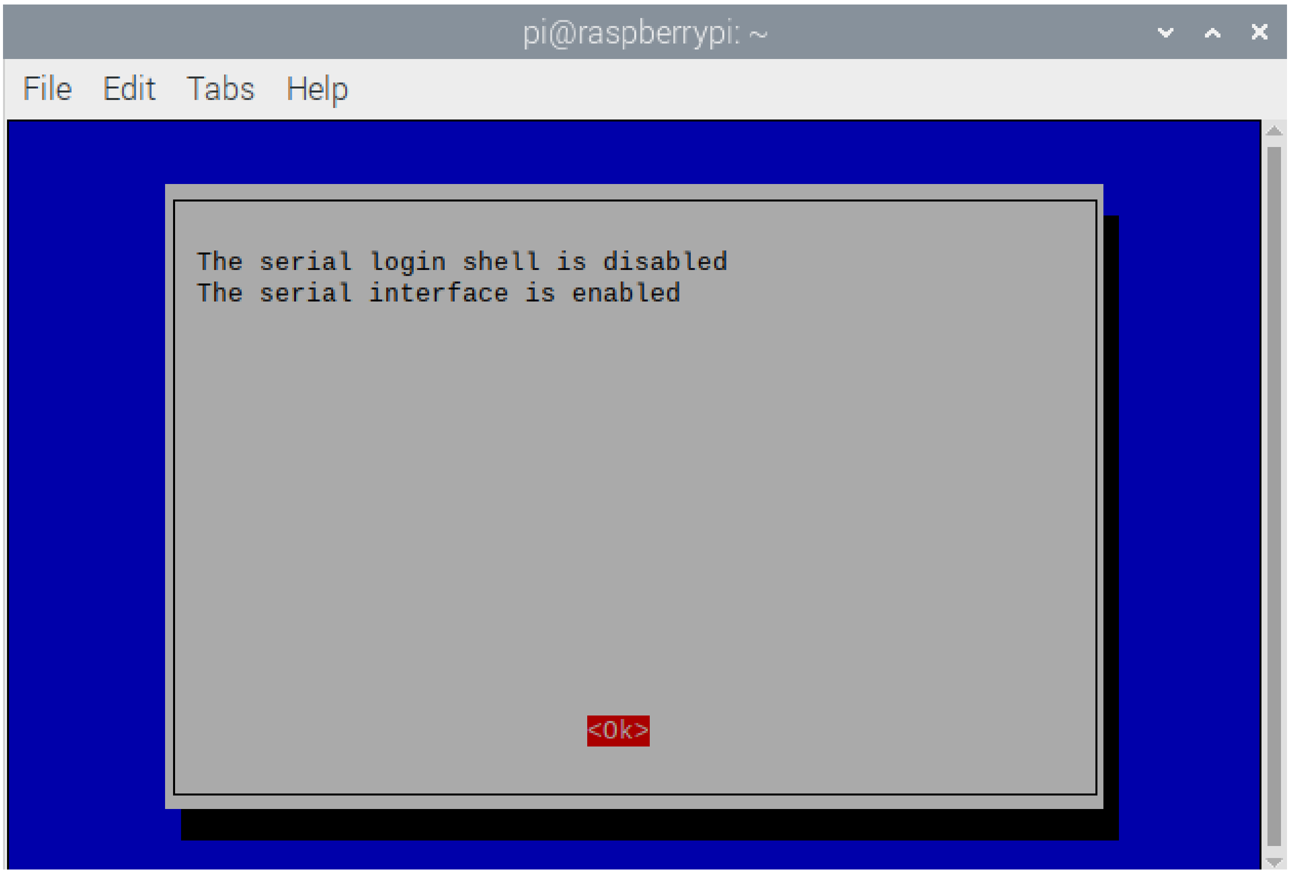
```
Raspberry Pi Software Configuration Tool (raspi-config)

I1 Legacy Camera Enable/disable legacy camera support
I2 SSH           Enable/disable remote command line access using SSH
I3 VNC           Enable/disable graphical remote access using RealVNC
I4 SPI           Enable/disable automatic loading of SPI kernel module
I5 I2C           Enable/disable automatic loading of I2C kernel module
I6 Serial Port   Enable/disable shell messages on the serial connection
I7 1-Wire        Enable/disable one-wire interface
I8 Remote GPIO   Enable/disable remote access to GPIO pins

                 <Select>                     <Back>
```



```
Would you like a login shell to be accessible over
serial?

               <Yes>                     <No>
```



```
Would you like the serial port hardware to be enabled?

                <Yes>                      <No>
```



```
The serial login shell is disabled
The serial interface is enabled

                     <Ok>
```