How to Setup Serial Connection Between BeagleBone/Raspberry Pi and Arduino

By Team Pixel Date: Nov - 25 - 2019 (Fall 2019)

Contents

This document has the following contents:

Components needed

1. your mainboard, it can be either BeagleBone or Raspberry Pi
2. USB-A to USB-B cable (check the type of USB port on your Arduino)
3. Arduino, for this guide we are using **Arduino Uno**, other models should also work
4. LCD display (compatible with Hitachi HD44780 driver ,for LCD demo)
5. **1500Ω** and **220Ω** resistor (for LCD demo)
6. Some jumper wires (for LCD demo)

Software needed

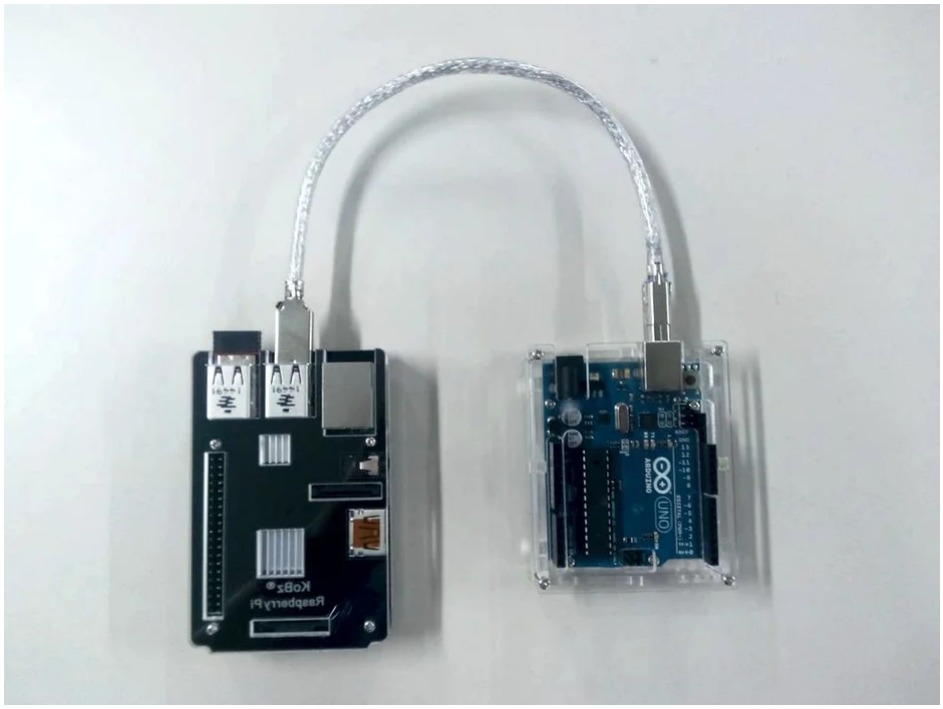
You might need to setup Arduino IDE first to upload the Arduino program (.ino files) to your Arduino board.

Arduino IDE download: https://www.arduino.cc/en/Main/Software Arduino Products list: https://www.arduino.cc/en/Main/Products

Serial Connection

Wiring

Connect your mainboard and Arduino board as follow. You don't need a dedicated power supply to Arduino, as long as you don't have power-hungry components on Arduino. However, you need to power your mainboard as usual.



Wiring is done for serial connection.

Software setting

After boot into your main board, you should use the following command to check if the Arduino is connected.

```
ls /dev/tty*
```

Look for a device **/dev/ttyACM0**, that is your Arduino board.

Main Board Serial Connection Program in C

Here is a sample C program called serial_write.c for sending user input string to Arduino through serial write.

```
#include <stdio.h>
#include <stdbool.h>
#include <stdlib.h>
#include <fcntl.h>   /* File Control Definitions           */
#include <termios.h> /* POSIX Terminal Control Definitions */
#include <unistd.h>  /* UNIX Standard Definitions      */
#include <string.h>

#define BUFFER_SIZE 256

static bool stop = false;
static struct termios SerialPortSettings;

int main(void){
    int fd;/*File Descriptor*/
    fd = open("/dev/ttyACM0",O_RDWR | O_NOCTTY | O_NDELAY);
    if(fd == -1)                        /* Error Checking */
       printf("\n  Error! in Opening ttyACM0  ");
    else
```

```
        printf("\n  ttyACM0 Opened Successfully ");

    tcgetattr(fd, &SerialPortSettings); /* Get the current attributes of the Serial port */
    cfsetispeed(&SerialPortSettings,B9600); /* Set Read  Speed as 9600                   */
    cfsetospeed(&SerialPortSettings,B9600); /* Set Write Speed as 9600                   */
    SerialPortSettings.c_cflag &= ~PARENB;   /* Disables the Parity Enable bit(PARENB),So No Parity   */
    SerialPortSettings.c_cflag &= ~CSTOPB;   /* CSTOPB = 2 Stop bits,here it is cleared so 1 Stop bit */
    SerialPortSettings.c_cflag &= ~CSIZE;    /* Clears the mask for setting the data size             */
    SerialPortSettings.c_cflag |=  CS8;      /* Set the data bits = 8                                 */
    SerialPortSettings.c_cflag &= ~CRTSCTS;       /* No Hardware flow Control                         */
    SerialPortSettings.c_cflag |= CREAD | CLOCAL; /* Enable receiver,Ignore Modem Control lines       */
    SerialPortSettings.c_iflag &= ~(IXON | IXOFF | IXANY);          /* Disable XON/XOFF flow control both i/p and o/p */
    SerialPortSettings.c_iflag &= ~(ICANON | ECHO | ECHOE | ISIG);  /* Non Cannonical mode                            */
    SerialPortSettings.c_oflag &= ~OPOST;/*No Output Processing*/
    if((tcsetattr(fd,TCSANOW,&SerialPortSettings)) != 0) /* Set the attributes to the termios structure*/
        printf("\n  ERROR ! in Setting attributes");
    else
        printf("\n  BaudRate = 9600 \n  StopBits = 1 \n  Parity   = none");

    char buffer[BUFFER_SIZE];
    int bytes_written = 0;
    while(!stop){
        memset(buffer, ' ', sizeof(char)*sizeof(buffer));
        printf("\nEnter text you want to sent\n");
        scanf("%s",&buffer);
        if(!stop){
            // str_strip(buffer);
            bytes_written = write(fd,buffer,strlen(buffer));
            printf("\nNumber of bytes written=%d\n",bytes_written);
        }
    }
    return 0;
}
```

The sample program is modified from the following page, this page also contains a serial read sample.

Troubleshooting note: compiler might not know where to find the definition for CRTSCTS. So CRTSCTS undefined error may occur. CRTSCTS is a 10 bits all 0 bit mask to turn off flow control. You could solve it by adding **-std=gnu99** to the make file, or just add the following line at the top.

```
 # define CRTSCTS  020000000000 //10 bit all 0 bit mask to disable flow control
```

Serial-Port-Programming-on-Linux: https://github.com/xanthium-enterprises/Serial-Port-Programming-on-Linux/blob/master/USB2SERIAL_Write/Transmitter%20(PC%20Side)/SerialPort_write.c

Makefile

This make file is for direct compile on the main board using gcc. If you want to use cross-compiler, please create your own Makefile.

```
all: serial_write
    rm -rf *.o
serial_write: serial_write.o
    gcc serial_write.o -o serial_write
serial_write.o: serial_write.c
    gcc -c serial_write.c
clean:
    rm -rf *o serial_write
```

Arduino

If you want to enable serial in Arduino, all you need to do is add a line **Serial.begin(9600);** in your setup function. Arduino does support other transfer rates than 9600 bps, but 9600 bps is the most common one. If you want to use other transfer speeds, be sure to modify your mainboard serial program accordingly.

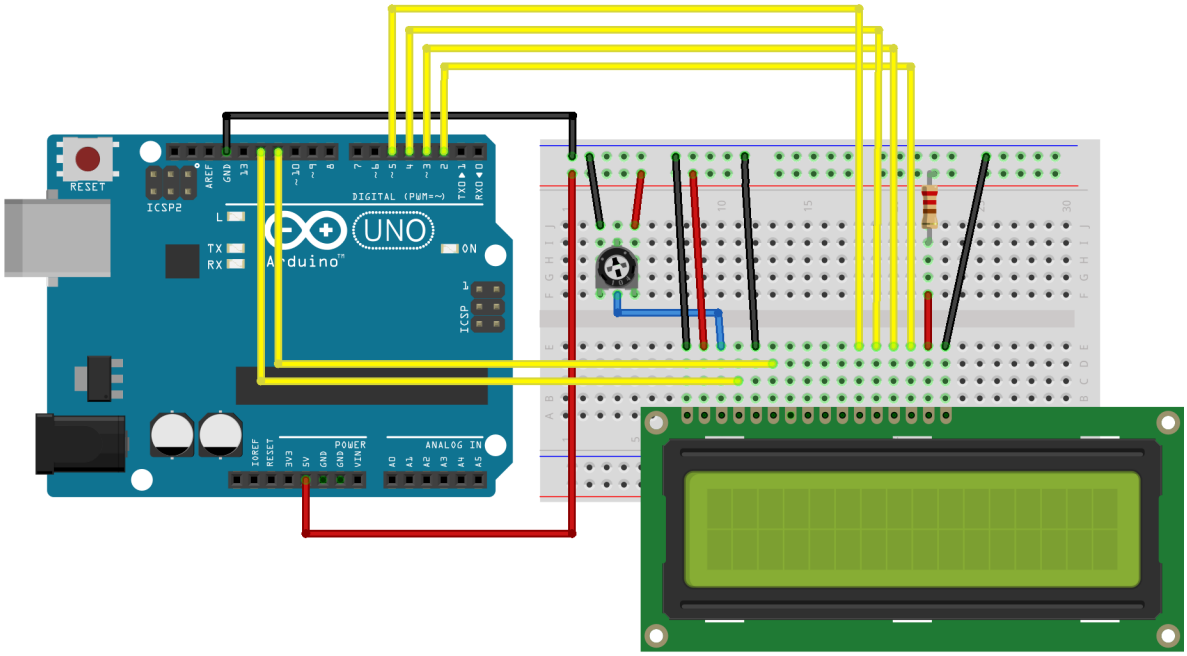
```
void setup() {
  Serial.begin(9600);
}

void loop() {
  if (Serial.available()) {
    // do your serial operations here
  }
}
```

Arduino LCD Display Demo

Right now we have our serial connection setup, but we don't know if it will work. So we add an LCD display to Arduino and program it to display whatever strings come from the mainboard through serial.

Arduino LCD Wiring

Wire the LCD display as follow. One thing to mention is the potentiometer(connected to the third pin of the LCD using a blue wire) in the following graph is for backlight adjustment. For simplicity, you could connect the third pin of LCD to the ground using a normal resistor. In my testing, 1500Ω resistor works best.



Arduino: LCD Hello World Tutorial: https://www.arduino.cc/en/Tutorial/HelloWorld

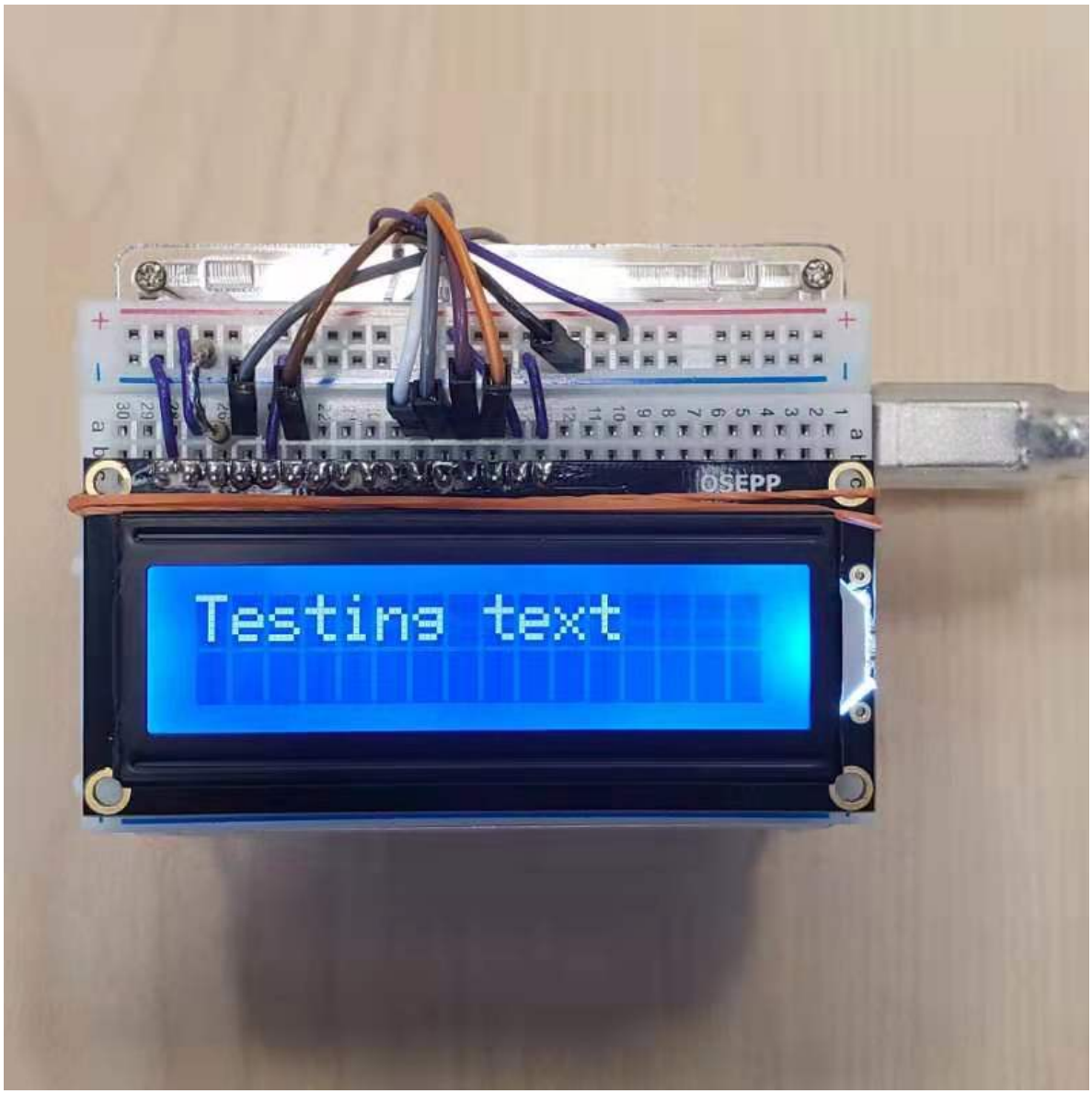
Arduino LCD Program

```
#include <LiquidCrystal.h>

// initialize the library with the numbers of the interface pins
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

void setup() {
  // set up the LCD's number of columns and rows:
  lcd.begin(16, 2);
  Serial.begin(9600);
  lcd.clear();
}

void loop() {
  // when characters arrive over the serial port...
  if (Serial.available()) {
    // wait a bit for the entire message to arrive
    delay(100);
    // clear the screen
    lcd.clear();
    // read all the available characters
    while (Serial.available() > 0) {
      // display each character to the LCD
      lcd.write(Serial.read());
    }
  }
}
```

Arduino: LCD Serial Display: https://www.arduino.cc/en/Tutorial/LiquidCrystalSerialDisplay

Result

After type in “Testing text” on the mainboard, the C program transfers a string of characters to Arduino, and Arduino displays the text on the LCD display.



Have fun with your new toy 😁