

I2C Interfacing through RaspberryPi and Arduino

Alok Ranjan Kesari[†], Veera Shanti Ram[†] and G V V Sharma*

CONTENTS

1	Single Slave	1
1.1	Configuring Arduino as slave	1
1.2	On RaspberryPi	1
1.3	Connecting RaspberryPi to Arduino	1
1.4	Hardware Setup	2
1.5	RaspberryPi as Master . . .	2
2	Multiple Slaves	2

1 SINGLE SLAVE

1.1 Configuring Arduino as slave

Problem 1.1. Run the following program on the RaspberryPi using Arduino software. This will configure the Arduino as a slave.

```
#include <Wire.h> //library for
    accessing I2C bus.
#define SLAVE_ADDRESS 0x04 //
    assigning the slave address.

int r=0;
void setup() {
    Serial.begin(9600);
    Wire.begin(SLAVE_ADDRESS); //
        starting the I2C
        communication.
    Wire.onRequest(sendData); //
        sending data to the Master.
    Serial.println("Ready!");
}
void loop() {
    delay(100);
}
```

```
}
void sendData() {
    r=analogRead(2); //port at
        which resistance is
        connected.
    Wire.write(r/5); //scaling the
        value by 5 since analog
        value generated by Arduino
        is in the range of 0 to 1023
        and the library can handle
        only upto 255.
}
```

1.2 On RaspberryPi

The following commands are for RaspberryPi. The process for running it on other Linux distributions is similar.

```
#Remove I2C from blacklist by editing the
    following file:
sudo nano /etc/modprobe.d/raspi-blacklist.conf
#After editing it should look like this:
blacklist spi-bcm2708
#blacklist i2c-bcm2708
#edit the following file:
sudo nano /etc/modules
#Add the following line at the end of the file:
i2c-dev
#Install I2C tools:
sudo apt-get install i2c-tools
#Allow Pi user to access I2C devices
sudo adduser pi i2c
#Reboot the RaspberryPi
#For checking if the slave is connected
sudo i2cdetect -y 1
```

The last command above will display the Arduino address as 0x04 on the terminal screen.

1.3 Connecting RaspberryPi to Arduino

Problem 1.2. Connect the pins as in Table 1 and Fig. 1.2

[†] The authors were interns at IIT Hyderabad during the summer of 2017 email: alok.kesari@yahoo.co.in, veerashanthiram@gmail.com.
*The author is with the Department of Electrical Engineering, Indian Institute of Technology, Hyderabad 502285 India e-mail: gadepall@iith.ac.in.

RaspberryPi	Arduino
GND	GND
3	A4
5	A5

TABLE 1

Raspberry Pi 3 GPIO Header				
Pin#	NAME		NAME	Pin#
01	3.3v DC Power		DC Power 5v	02
03	GPIO02 (SDA1 , I2C)		DC Power 5v	04
05	GPIO03 (SCL1 , I2C)		Ground	06
07	GPIO04 (GPIO_GCLK)		(TXD0) GPIO14	08
09	Ground		(RXD0) GPIO15	10
11	GPIO17 (GPIO_GEN0)		(GPIO_GEN1) GPIO18	12
13	GPIO27 (GPIO_GEN2)		Ground	14
15	GPIO22 (GPIO_GEN3)		(GPIO_GEN4) GPIO23	16
17	3.3v DC Power		(GPIO_GEN5) GPIO24	18
19	GPIO10 (SPI_MOSI)		Ground	20
21	GPIO09 (SPI_MISO)		(GPIO_GEN6) GPIO25	22
23	GPIO11 (SPI_CLK)		(SPI_CE0_N) GPIO08	24
25	Ground		(SPI_CE1_N) GPIO07	26
27	ID_SD (I2C ID EEPROM)		(I2C ID EEPROM) ID_SC	28
29	GPIO05		Ground	30
31	GPIO06		GPIO12	32
33	GPIO13		Ground	34
35	GPIO19		GPIO16	36
37	GPIO26		GPIO20	38
39	Ground		GPIO21	40

Fig. 1.2

1.4 Hardware Setup

Problem 1.3. Connect resistance to be measured and the known resistance value as per the following circuit diagram on a breadboard.

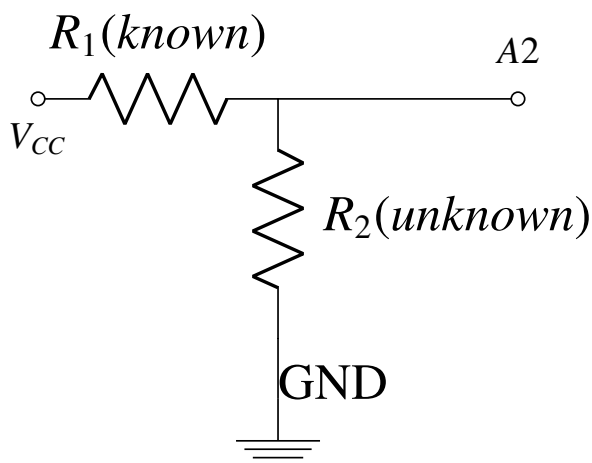


Fig. 1.3

Problem 1.4. Connect one extreme end of breadboard to the 5V of Arduino and another extreme end to the GND of Arduino.

Problem 1.5. Connect +ve pin of V_{cc} from the resistance circuit setup on the breadboard to 5V on the breadboard and -ve pin of V_{cc} to GND on the breadboard.

Problem 1.6. Connect the +ve pin of V_{out} from the resistance circuit setup on the breadboard to A2 of Arduino and -ve pin of V_{out} to GND on the breadboard.

1.5 RaspberryPi as Master

Problem 1.7. Run the following code on RaspberryPi.

```
import smbus
import time
#(1) denotes the I2C port being used.
bus = smbus.SMBus(1)
vi=5;
b=0;
r1=1000;
# These are the Arduino addresses.
address1 = 0x04

def readNumber(address):#Passing the address
as a parameter to the the read function .
number = (bus.read_byte(address)) #Reading
from the slave.
return number

while True:
time.sleep(1)
n=(readNumber(address1)*5)
vo=(n*vi)/1024.0;
b=(vi/vo)-1;
r2=r1/b;
print ('The resistance value measured is
:', r2)
```

2 MULTIPLE SLAVES

Problem 2.1. Configure another Arduino as a slave with address 0x05. Use problem 1.1.

Problem 2.2. Connect one more Arduino to the RaspberryPi using a breadboard according to Table

1. You will have to make multiple connections using the breadboard.

Problem 2.3. Make a similar resistance circuit setup using the circuit diagram of 1.3 and instructions in 1.4: Hardware Setup.

Problem 2.4. Run the following command to check if both arduinos are detected.

```
sudo i2cdetect -y 1
```

Problem 2.5. Modify the code in problem 1.7 to verify if the i2c connection with the second arduino is active.

Problem 2.6. Run the following program to poll from both arduinos on the i2c bus.

```
import smbus
import time
#(1) denotes the I2C port being used.
bus = smbus.SMBus(1)
vi=5;
b=0;
r1=1000;
# These are the Arduino addresses.
address1 = 0x04
address2 = 0x05

def readNumber(address):#passing the address
as a parameter to the the read function.
number = (bus.read_byte(address)) #
Reading from the slave.
return number

while True:
time.sleep(1)

n=(readNumber(address1)*5)
vo=(n*vi)/1024.0;
b=(vi/vo)-1;
r2=r1/b;
print ('The resistance value measured_1 is
:', r2)

n=(readNumber(address2)*5)
vo=(n*vi)/1024.0;
b=(vi/vo)-1;
r2=r1/b;
print ('The resistance value measured_2 is
:', r2)
```