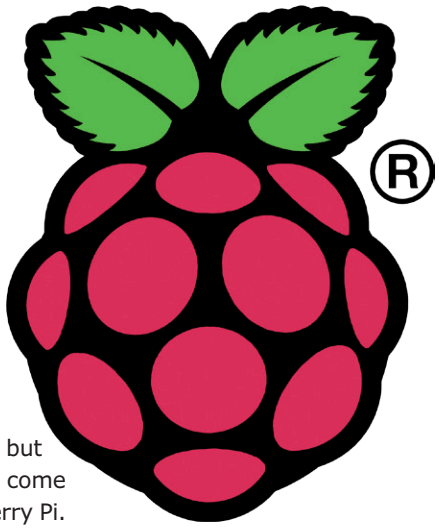
Raspberry Pi Recipes

Part #5

I²C: I Square Seeds (for baking)

By **Tony Dixon** (UK)



In the previous two parts published through Elektor .POST we've looked at the UART and SPI interfaces of the Raspberry Pi's Expansion Header. Now, on our way to becoming *electro-baking experts*, we'll take a look at the last of the Raspberry Pi's serial interfaces, the I²C Bus.

I²C Interface

The Inter-IC or I²C is the final of the three serial interfaces you'll find on the Raspberry Pi's Expansion Header. The other two interfaces are an UART Serial Interface and the SPI interface (see parts #3 and #4 respectively).

Table 1 details the Expansion Header signals and the I²C interface can be found on pin 3 (SDA) and pin 5 (SCL).

Like SPI, the I²C interface is designed to interface other devices with a minimal number of signals. I²C uses only two bidirectional open-drain lines, Serial Data Line (SDA) and Serial Clock (SCL) to provide its bus. These are typically pulled up with resistors to 3.3 V as in the Raspberry Pi's case via two 1.8 kΩ resistors

I²C is not as fast as say the SPI bus, but common bus speeds for it are 100 Kbit/s in Standard Mode and 400 Kbit/s in Fast Mode. The Broadcom SoC chip used by the Raspberry Pi has two I²C interfaces. The original version of the Raspberry Pi only had one I²C interface available, the first of the I²C interfaces (I2C_SDA0 and I2C_SCL0) on its Expansion Header.

The second Raspberry Pi revision added an additional smaller expansion header and allowed access to the second I²C interface, but it also swaps things around a little. The Expansion Header of a Revision 2 is changed to use the second I²C interface (I2C_SDA1 and I2C_SCL1), while the first I²C interface (I2C_SDA0 and I2C_SCL0) was moved to the new smaller expansion header—a small but important thing to remember when we come to do things with the I²C on our Raspberry Pi.

Table 1. Expansion Header Pin Out

Pin Name	Pin Function	Alternative	RPi.GPIO
P1-02	5.0V	-	-
P1-04	5.0V	-	-
P1-06	GND	-	-
P1-08	GPIO14	UART0_TXD	RPi.GPIO8
P1-10	GPIO15	UART0_RXD	RPi.GPIO10
P1-12	GPIO18	PWM0	RPi.GPIO12
P1-14	GND	-	-
P1-16	GPIO23		RPi.GPIO16
P1-18	GPIO24		RPi.GPIO18
P1-20	GND	-	-
P1-22	GPIO25		RPi.GPIO22
P1-24	GPIO8	SPI0_CE0_N	RPi.GPIO24
P1-26	GPIO7	SPI0_CE1_N	RPi.GPIO26

Pin	Board Revision 1		Board Revision 2	
Name	Pin Function	Alternative	Pin Function	Alternative
P1-01	3.3V	-	3.3V	-
P1-03	GPIO0	**I2C0_SDA**	GPIO2	**I2C1_SDA**
P1-05	GPIO1	**I2C0_SCL**	GPIO3	**I2C1_SCL**
P1-07	GPIO4	GPCLK0	GPIO4	GPCLK0
P1-09	GND	-	GND	-
P1-11	GPIO17	RTS0	GPIO17	RTS0
P1-13	GPIO21		GPIO27	
P1-15	GPIO22		GPIO22	
P1-17	3.3V	-	3.3V	-
P1-19	GPIO10	SPI0_MOSI	GPIO10	SPI0_MOSI
P1-21	GPIO9	SPI0_MISO	GPIO9	SPI0_MISO
P1-23	GPIO11	SPI0_SCLK	GPIO11	SPI0_SCLK
P1-25	GND	-	GND	-

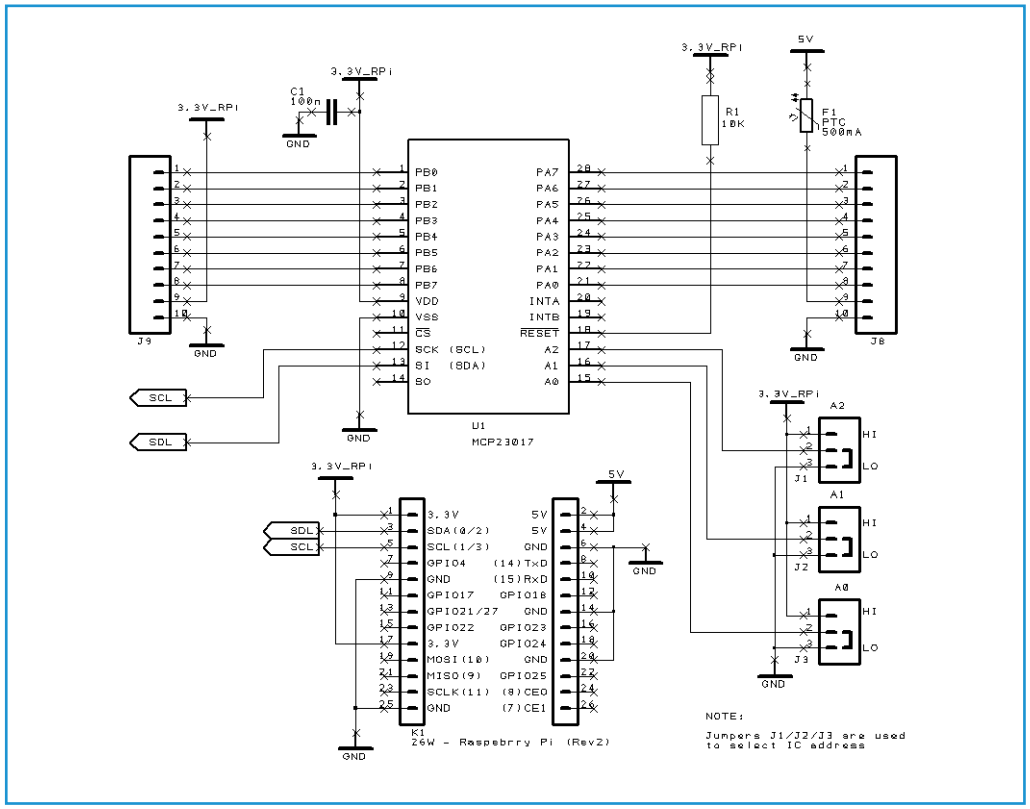


Figure 1.
Schematic for Raspberry Pi MCP23017 Port Expander.

Port Expander Hardware (again)

In our Raspberry Pi I²C project we'll be expanding the number of the RPI GPIOs by adding yet another Port Expander. As before, we'll be using a port expander from Microchip, but this time it's about a 16-channel MCP23017 [1], which is the I²C cousin of the MCP23S17 we used in our SPI Port Expander Project.

Figure 1 shows a simplified MCP23017 circuit. The chip is connected to the RPi's I²C interface. Unlike the SPI version of the circuit there are no chip enable signals required, so the circuit is very simple. Jumpers J1, J2 and J3 provide optional address signals to the port expander, allowing more than one port expander to be connected to the I²C bus.

Figure 2 shows our hardware setup where we are using a small add-on board to provide our MC23017 interface. Eagle eyed readers will notice it's the same board as we used previously [2], which is correct, as this particular board design can use either the I²C MCP23017 or SPI MCP23S17 by changing the jumper selection for either SPI or I²C.

Installing I²C Tools

Before we install the I²C tools we need to do a little housekeeping and tell Raspian that we intend to use the hardware I²C interface. The hardware I²C is disabled by default, so we have to change this by editing the blacklist file:

```
sudo nano /etc/modprobe.d/raspi-blacklist.conf
```

Find the text line with ***blacklist i2c-bcm2708***, insert a # (hash) at the start of the line to comment-out the statement and then save the file.

Next we need to edit the modules file by:

```
sudo nano /etc/modules
```

Add the text ***i2c-dev*** to a new line and save the file.

We'll now install the i2c-tools package by typing:

```
sudo apt-get update
sudo apt-get install i2c-tools
```

Once this is installed, we need to add a new user to the i2c group, type:

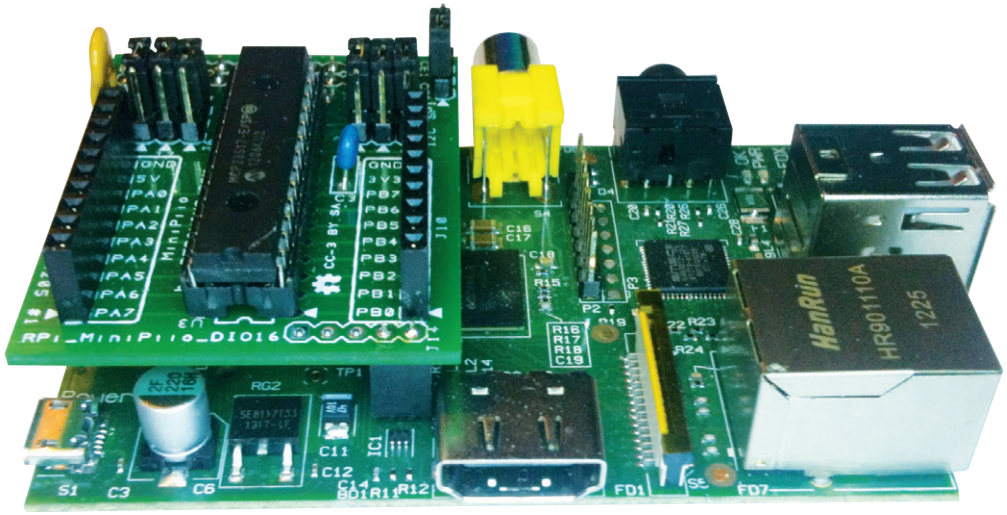


Figure 2.
Pi and MCP23017 Add-On Board.

```
sudo adduser pi i2c
```

Let's do a quick reboot by typing:

```
sudo reboot
```

Once we've rebooted, we can check for the I²C interfaces. Start a new LXTerminal session and type...

```
ls /dev/i2c*
```

...to check that we have two I²C devices listed (one for each I²C interface) and we should have:

```
/dev/i2c-0
/dev/i2c-1
```

We can also test them, if you have a Rev 1 Pi type:

```
sudo i2cdetect -y 0
```

or if you have a Rev. 2 Pi type:

```
sudo i2cdetect -y 1
```

You should see something like this, see **Figure 3**.

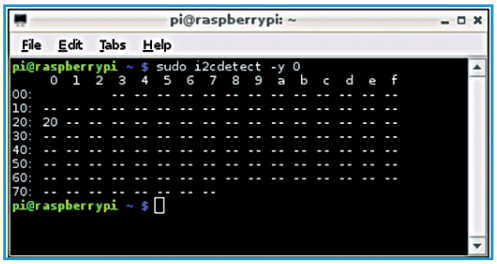


Figure 3.
"i2cdetect" results.

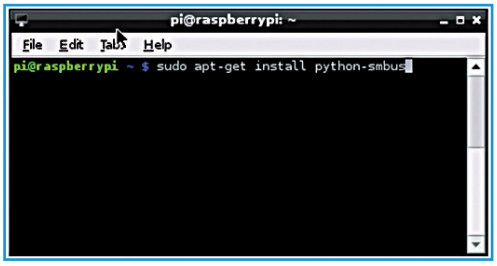


Figure 4.
LXTerminal.

Installing Python's smbus I²C Library

We'll be using Python 2 for the examples in this project. Python, as we should know from our previous parts, is already is installed as standard in the Raspian distribution.

However, there is no provision for the I²C interface. To fix this we will need to install the I²C Python wrapper / library, so let's start a LXterminal session, as shown in **Figure 4**, and type the following commands:

```
sudo apt-get install python-smbus
```

Once this is installed, we're now ready to use the I²C with Python.

Example Program – mcp23017.py

With smbus installed we're now going to write a small test program to illuminate LEDs wired to the Port Expander GPIO.

Double click IDLE icon on your Pi's desktop to start the Python Shell and IDLE (**Figure 5**).

Select File option from the menu and create a

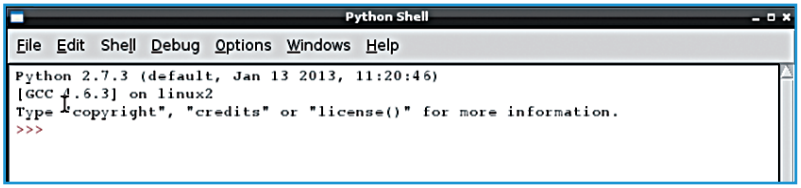


Figure 5.
IDLE Python Shell.

new program. This will start the IDLE editor.

In the IDLE editor (**Figure 6**), type the program as shown in **Listing 1**.

Once you've typed the program, make sure you save it, then switch to a LXTerminal and type the following command to make your program an executable:

```
sudo chmod +x mcp23017.py
```

Once done, you can run your program by typing the following command:

```
sudo ./mcp23017.py
```

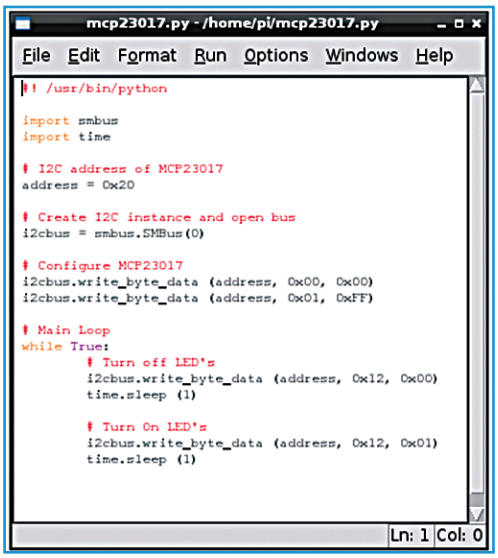


Figure 6.
IDLE Editor.

Listing 1.

```
#! /usr/bin/python

import smbus
import time

# I2C address of MCP23017
address = 0x20

# Create I2C instance and open bus
i2cbus = smbus.SMBus(0)

# Configure MCP23017
i2cbus.write_byte_data(address,0x00,0x00) # Set Bank A to outputs
i2cbus.write_byte_data(address,0x01,0xFF) # Set Bank B to inputs

# Main loop
while True:
  # Turn off LEDs
  i2cbus.write_byte_data (address,0x12,0x00)
  time.sleep(1)

  # Turn on PortA.0
  i2cbus.write_byte_data (address,0x12,0x01)
  time.sleep(1)
```

Note: For Rev 2 Pi boards change the line:

`i2cbus = smbus.SMBus(0)` **to** `i2cbus = smbus.SMBus(1)`

Table 2 includes a quick summary of MCP23x17 control registers.

(130236)

Internet Links

[1] ww1.microchip.com/downloads/en/devicedoc/21952b.pdf

[2] www.dtronixs.com

Table 2. MCP23x17 Register Address Map

Address IOCON.BANK = 1	Address IOCON.BANK = 0	Register	Description
0x00 / 0 dec	0x00 / 0 dec	IODIRA	I/O Direction Register for Port A
0x10 / 16 dec	0x01 / 1 dec	IODIRB	I/O Direction Register for Port B
0x01 / 1 dec	0x02 / 2 dec	IPOLA	Input Polarity Port Register for Port A
0x11 / 17 dec	0x03 / 3 dec	IPOLB	Input Polarity Port Register for Port B
0x02 / 2 dec	0x04 / 4 dec	GPINTENA	Interrupt-n-Change Control Register Port A
0x12 / 18 dec	0x05 / 5 dec	GPINTENB	Interrupt-n-Change Control Register Port B
0x03 / 3 dec	0x06 / 6 dec	DEFVALA	Default Compare Register for GPINTENA
0x13 / 19 dec	0x07 / 7 dec	DEFVALB	Default Compare Register for GPINTENB
0x04 / 4 dec	0x08 / 8 dec	INTCONA	Interrupt Control Register for Port A
0x14 / 20 dec	0x09 / 9 dec	INTCONB	Interrupt Control Register for Port B
0x05 / 5 dec	0x0A / 10 dec	IOCON	I/O Expander Configuration Register
0x15 / 21 dec	0x0B / 11 dec	IOCON	I/O Expander Configuration Register
0x06 / 6 dec	0x0C / 12 dec	GPPUA	Pull-Up Resistor Configuration Register Port A
0x16 / 22 dec	0x0D / 13 dec	GPPUB	Pull-Up Resistor Configuration Register Port B
0x07 / 7 dec	0x0E / 14 dec	INTFA	Interrupt Flag Register for Port A
0x17 / 23 dec	0x0F / 15 dec	INTFB	Interrupt Flag Register for Port B
0x08 / 8 dec	0x10 / 16 dec	INTCAPA	Interrupt Capture Register for Port A
0x18 / 24 dec	0x11 / 17 dec	INTCAPB	Interrupt Capture Register for Port B
0x09 / 9 dec	0x12 / 18 dec	GPIOA	Port Register for Port A
0x19 / 25 dec	0x13 / 19 dec	GPIOB	Port Register for Port B
0x0A / 10 dec	0x14 / 20 dec	OLATA	Output Latch Register for Port A
0x1A / 26 dec	0x15 / 21 dec	OLATB	Output Latch Register for Port B