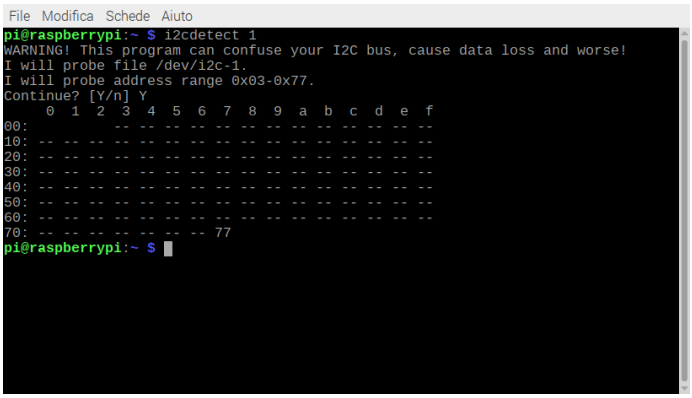
LABORATORIO DIDATTICO 13 - Acquisizione remota dei dati di un sensore I2C con Raspberry Pi, Apache e Python

Introduzione: uso di un dispositivo con interfaccia digitale

Molti sensori e attuatori impiegati nell'ambito IoT si connettono ad un microcontrollore come Arduino o a un microcomputer come Raspberry mediante un'interfaccia digitale: le più diffuse sono **I2C (*Inter-Integrated Circuit*)** e **SPI (*Serial Peripheral Interface*)**. I2C è un bus che impiega due segnali elettrici – uno bidirezionale per i dati e uno unidirezionale per il *clock* – e in cui ogni singolo dispositivo ha un indirizzo di attivazione. SPI invece impiega tre segnali elettrici – due per i dati e uno per il *clock* – e seleziona il dispositivo da attivare mediante un segnale elettrico aggiuntivo (come descritto nei SOTTOPARAGRAFI 1.2 e 1.3 del CAPITOLO 9).
Per utilizzare dispositivi I2C (o SPI) connessi al microcomputer Raspberry Pi è necessario abilitare il sistema operativo Raspbian alla loro gestione utilizzando la scheda "Interfacce" dello strumento di configurazione accessibile dalla voce "Preferenze" del menu principale del sistema operativo.
Per i dispositivi I2C è inoltre necessario installare gli strumenti di utilità ed il modulo Python di gestione SMBus (*System Management Bus*) con il seguente comando della *shell*:

```
$ sudo apt-get install python-smbus i2c-tools
```

Tra gli strumenti installati, il comando "i2cdetetc" – per il quale deve essere specificato come parametro il numero della porta I2C del connettore GPIO utilizzata – consente di rilevare gli indirizzi di eventuali dispositivi connessi e alimentati:



Il modulo Python "smbus" espone i seguenti metodi per la gestione dell'interfaccia I2C da invocare su un oggetto SMBus precedentemente istanziato:

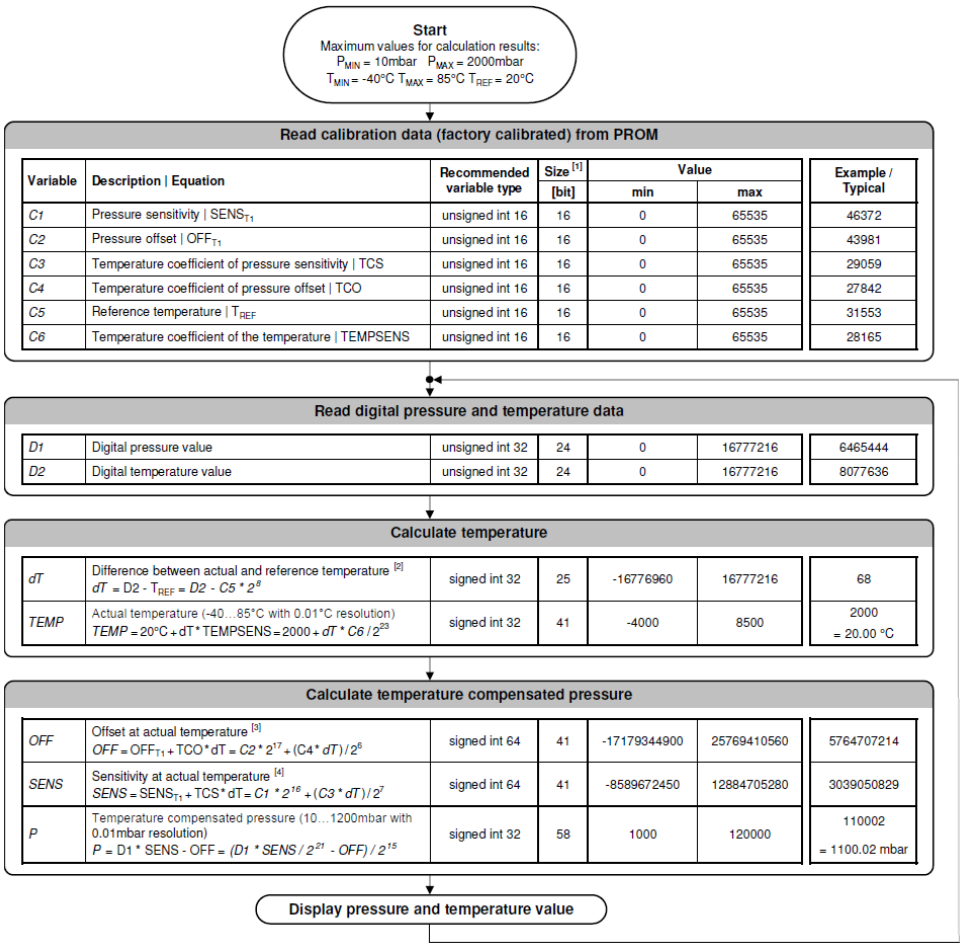
Metodo	Descrizione
`read_byte()`	legge e restituisce un singolo byte dal dispositivo con indirizzo specificato
`write_byte()`	scrive un singolo byte fornito come parametro nel dispositivo con indirizzo specificato
`read_byte_data()`	legge e restituisce un singolo byte dal dispositivo con indirizzo specificato; richiede i dati inviando un indirizzo di registro fornito come parametro
`write_byte_data()`	scrive un singolo byte fornito come parametro nel dispositivo con indirizzo specificato; inizia la trasmissione inviando un indirizzo di registro specificato anch'esso come parametro
`read_i2c_block_data()`	legge e restituisce una sequenza di byte di lunghezza definita come parametro

	dal dispositivo di indirizzo specificato; richiede i dati inviando un indirizzo di fornito anch'esso come parametro
`write_i2c_block_data()`	scrive una sequenza di byte fornita come parametro nel dispositivo di indirizzo specificato; inizia la trasmissione inviando un indirizzo di registro specificato anch'esso come parametro

Esempio

Il dispositivo sensore di temperatura e di pressione atmosferica denominato *Miniature Altimeter Module* di TE *connectivity* presenta sia l'interfaccia I2C che quella SPI.

Per connettere il sensore al connettore GPIO del microcomputer Raspberry Pi utilizzando l'interfaccia I2C si impiegano i pin 3 e 5, rispettivamente per i dati ed il *clock* I2C, e i pin 1 e 6 rispettivamente per l'alimentazione elettrica a 3,3V e la massa.
Il dispositivo comprende due sensori analogici e un singolo ADC con risoluzione di 24 bit: è possibile inviare comandi per calcolare la conversione A/D del sensore di temperatura o del sensore di pressione atmosferica e per acquisire successivamente il risultato della conversione. La trasformazione del risultato della conversione effettuata dallo ADC in centesimi di °C per la temperatura e in centesimi di mbar per la pressione è basata su specifici coefficienti di calibrazione che, come tutti i sensori di questo tipo, sono determinati al momento della fabbricazione e memorizzati permanentemente all'interno del dispositivo stesso.
Il diagramma di flusso che segue – tratto dal *datasheet* del sensore –illustra la sequenza operativa di acquisizione dei valori di calibrazione, di temperatura e di pressione atmosferica:



Tenendo conto che dal *datasheet* del dispositivo si ricavano le seguenti informazioni:

- l'indirizzo I2C del dispositivo è il valore esadecimale 77H,
- i coefficienti di calibrazione C1, C2, C3, C4, C5 e C6 sono memorizzati in locazioni di memoria di 16 bit aventi indirizzo esadecimale pari compreso tra A2H e ADH,
- i comandi per la conversione dei valori di temperatura e pressione sono rispettivamente i valori esadecimali 44H e 54H[1],
- il risultato a 24 bit della conversione è accessibile nel registro di indirizzo 0;

[1] essendo lo ADC del sensore del tipo Σ-Δ è possibile specificare il numero di iterazioni effettuate: maggiore è questo numero, maggiore è il tempo di conversione e maggiore è la precisione della conversione A/D; i comandi specificati corrispondono ad una condizione di 1024 iterazioni

l'esecuzione del seguente codice Python restituisce i valori attuali di temperatura e pressione rilevati dal sensore:

```
import smbus
import time

# indirizzo del dispositivo I2C
DEV_ADD = 0x77

# inizializzazione dell'interfaccia I2C #1 sul connettore GPIO
dev = smbus.SMBus(1)

# lettura dei 6 coefficienti di calibrazione
C = []
for add in range(0xA2, 0xAE, 2):
    # lettura di due byte dalla locazione di indirizzo specificato
    raw = dev.read_i2c_block_data(DEV_ADD, add, 2)
    # calcolo e memorizzazione del valore a 16 bit
    cooked = raw[1] + raw[0]*256
    C.append(cooked)
    # attesa di 10ms richiesta dal dispositivo
    time.sleep(.01)

# invio del comando di conversione A/D del sensore di temperatura
dev.write_byte(DEV_ADD, 0x44)
# attesa di 10ms richiesta dal dispositivo
time.sleep(0.01)
# lettura di tre byte dalla locazione di indirizzo 0
raw = dev.read_i2c_block_data(DEV_ADD, 0, 3)
# calcolo del valore convertito a 24 bit
D1 = raw[0]*65536 + raw[1]*256 + raw[2]
# attesa di 10ms richiesta dal dispositivo
time.sleep(0.01)
# invio del comando di conversione A/D del sensore di pressione
dev.write_byte(DEV_ADD, 0x54)
# attesa di 10ms richiesta dal dispositivo
time.sleep(0.01)
# lettura di tre byte dalla locazione di indirizzo 0
raw = dev.read_i2c_block_data(DEV_ADD, 0, 3)
# calcolo del valore convertito a 24 bit
D2 = raw[0]*65536 + raw[1]*256 + raw[2]

# conversione dei valori acquisiti con i coefficienti di calibrazione
delta = D2 - C[4]*256
T = 2000 + (delta*C[5])/8388608
offset = C[1]*131072 + (C[3]*delta)/64
sensor = C[0]*65536 + (C[2]*delta)/128
P = (D1*sensor/2097152 - offset)/32768

# visualizzazione dei valori di temperatura (°C) e pressione (mbar)
print(T/100, P/100)
```

Lo *script* Python accede al connettore GPIO e deve essere quindi eseguito come utente *superuser*; in alternativa è possibile aggiungere l'utente predefinito "pi" al gruppo "i2c" che ha i privilegi di accesso al bus I2C utilizzando il seguente comando dalla *shell*:

```
$ usermod -a -G i2c pi
```

Svolgimento del LABORATORIO DIDATTICO 13

In un laboratorio precedente è stato mostrato come installare il web-server Apache per sistema operativo Raspbian nel microcomputer Raspberry Pi: la tecnologia CGI è già abilitata.
Il seguente *script* Python deriva da quello dell'esempio precedente e genera il codice HTML di una semplice pagina web che visualizza i valori attuali di temperatura e pressione acquisiti dal sensore *Miniature Altimeter Module* di TE *connectivity* (la prima riga specifica al modulo CGI del web-server che il codice deve essere eseguito utilizzando l'interprete della versione 3 del linguaggio Python):

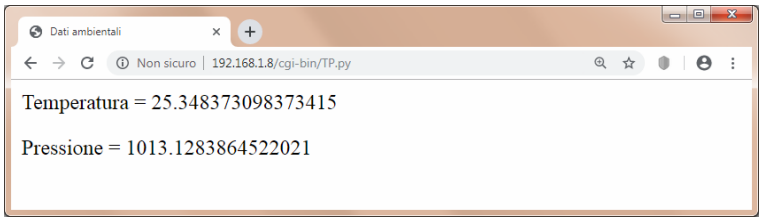
```
#! /usr/bin/env python3

import cgi
import smbus
import time

# acquisizione e conversione dei valori di temperatura e pressione
DEV_ADD = 0x77
dev = smbus.SMBus(1)
C = []
for add in range(0xA2, 0xAE, 2):
    raw = dev.read_i2c_block_data(DEV_ADD, add, 2)
    cooked = raw[1] + raw[0]*256
    C.append(cooked)
    time.sleep(.01)
dev.write_byte(DEV_ADD, 0x44)
time.sleep(0.01)
raw = dev.read_i2c_block_data(DEV_ADD, 0, 3)
D1 = raw[0]*65536 + raw[1]*256 + raw[2]
time.sleep(0.01)
dev.write_byte(DEV_ADD, 0x54)
time.sleep(0.01)
raw = dev.read_i2c_block_data(DEV_ADD, 0, 3)
D2 = raw[0]*65536 + raw[1]*256 + raw[2]
delta = D2 - C[4]*256
T = 2000 + (delta*C[5])/8388608
offset = C[1]*131072 + (C[3]*delta)/64
sensor = C[0]*65536 + (C[2]*delta)/128
P = (D1*sensor/2097152 - offset)/32768

# generazione dello header HTTP che specifica il contenuto HTML
print ("Content-Type: text/html")
# generazione della linea di separazione tra header e body HTTP
print("\r\n")
# generazione dell'intestazione della pagina web in HTML
print("<html>")
print("<head>")
```

```
print("<title>Dati ambientali</title>")
print("</head>")
# generazione del corpo della pagina web in HTML con i valori acquisiti
print("<body>")
print("Temperatura = ",T/100)
print("<p>")
print("Pressione = ",P/100)
print("</body>")
print("</html>")
```

Il file contenente lo *script* Python deve essere reso eseguibile e collocato nella directory `/usr/lib/cgi-bin`; all'utente fittizio "www-data" che esegue il web-server e, di conseguenza, il modulo CGI deve essere assegnato il permesso di accedere all'interfaccia I2C con il seguente comando di *shell*:

```
$ usermod -a -G i2c www-data
```

È possibile visualizzare i valori attuali di temperatura e pressione atmosferica utilizzando un browser da un qualsiasi computer che sia interconnesso in rete con il microcomputer Raspberry Pi (nel caso dell'immagine il file contenente lo *script* è stato denominato "TP.py"):



Se i dati rilevati dal sensore debbono essere utilizzati da un'applicazione software piuttosto che visualizzati per un utente umano è preferibile, anziché inserirli in una pagina web, produrli in un formato di interscambio dei dati standard come JSON (*JavaScript Object Notation*):

```
#! /usr/bin/env python3

import cgi
import smbus
import time

# acquisizione e conversione dei valori di temperatura e pressione
DEV_ADD = 0x77
dev = smbus.SMBus(1)
C = []
for add in range(0xA2, 0xAE, 2):
    raw = dev.read_i2c_block_data(DEV_ADD, add, 2)
    cooked = raw[1] + raw[0]*256
    C.append(cooked)
    time.sleep(.01)
dev.write_byte(DEV_ADD, 0x44)
time.sleep(0.01)
raw = dev.read_i2c_block_data(DEV_ADD, 0, 3)
D1 = raw[0]*65536 + raw[1]*256 + raw[2]
time.sleep(0.01)
dev.write_byte(DEV_ADD, 0x54)
time.sleep(0.01)
raw = dev.read_i2c_block_data(DEV_ADD, 0, 3)
```

```
D2 = raw[0]*65536 + raw[1]*256 + raw[2]
delta = D2 - C[4]*256
T = 2000 + (delta*C[5])/8388608
offset = C[1]*131072 + (C[3]*delta)/64
sensor = C[0]*65536 + (C[2]*delta)/128
P = (D1*sensor/2097152 - offset)/32768

# generazione dello header HTTP che specifica il contenuto JSON
print ("Content-Type: text/json")
# generazione della linea di separazione tra header e body HTTP
print("\r\n")
# generazione dell'oggetto JSON con i valori acquisiti
print("{")
print("\"temperatura\": ", T/100, ",")
print("\"pressione\": ", P/100)
print("}")
```

Anche in questo caso è possibile visualizzare i dati acquisiti mediante un browser: