AS-E405 Raspberry Pi GPIO experiment board

AS-E405 Raspberry Pi GPIO experimental board usage examples and sample code introductions.

This sample code is
Body: Raspberry Pi 3
OS: 2016-05-27-raspbian-jessie
Python：2.7.9
Created with.

1. Raspberry Pi setup

・Please make initial settings such as network settings and Japanese localization.
* Please refer to Raspberry Pi related books and related sites for the setting procedure.

2. How to use Python on Raspberry Pi

Save the file name as test.py by right-clicking on the desktop and creating a new file.
Double-click the created test.py to open the file in the Python development environment.
Write the following code in test.py and save it.
test.py：https://translate.google.com/translate?hl=en&sl=ja&u=https://www.sunhayato.co.jp/support/skill/AS-E405.html&prev=search

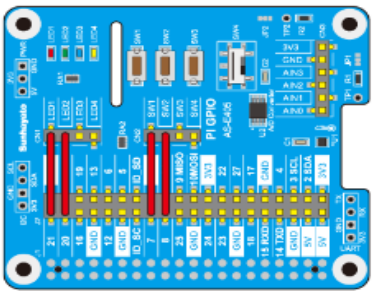
```
#!/usr/bin/python
print "Hello, world!"
```

Save (ctrl + s) and press F5 on the keyboard to open another window called Shell and execute the code
Hello, world!
Should be displayed.
The following introduces sample code of GPIO, temperature sensor, and A / D converter.

3. GPIO usage example

With jump wire GPIO21 and LED1
・GPIO20 and LED2
・GPIO7 and SW1
・GPIO8 and SW2
Connect.

■GPIO 使用例の接続　　（GPIO.py）



【接続内容】
GPIO21 — LED1
GPIO20 — LED2

GPIO7 — SW1
GPIO8 — SW2

When the following code(GPIO.py)is executed: -LED1 and LED2 blink alternately-SW1 and SW2 status is displayed with 0 (OFF) / 1 (ON) every second.

```
#!/usr/bin/python
# coding: utf-8

#Import module
import RPi.GPIO as GPIO
import time

#Specified by GPIO number
GPIO.setmode(GPIO.BCM)

GPIO.setup(21, GPIO.OUT)    #21 - LED1
GPIO.setup(20, GPIO.OUT)    #20 - LED2
GPIO.setup(7,  GPIO.IN)     #7  - SW1
GPIO.setup(8,  GPIO.IN)     #8  - SW2

while 1:
    GPIO.output(21, 1)  #Set pin 21 to H (3.3V) = LED1 lit
    GPIO.output(20, 0)  #Set pin 20 to L (0.0V) = LED2 off
    print "SW1=",
    print GPIO.input(7) #Outputs pin 7 status
    print "SW2=",
    print GPIO.input(8) #Output the state of pin 8

    time.sleep(1)       #wait 1 second

    GPIO.output(21, 0)  #Set pin 21 to H (3.3V) = LED1 lit
    GPIO.output(20, 1)  #Set pin 20 to L (0.0V) = LED2 lit

    time.sleep(1)       #wait 1 second
```

4. Set up to use I2C on Raspberry Pi

By default, the I2C function of Raspberry Pi is disabled. Change the setting to enable it.
Run the following command from the terminal to install the I2C library.

```
$ sudo apt-get install i2c-tools libi2c-dev python-smbus
```

After installing the I2C library, select "Interface" from "Menu-> Settings-> Raspberry Pi settings" and change the I2C item to valid.
Check that I2C is enabled after rebooting so that the setting is reflected.
When the following command is executed from the terminal, the address of the connected device is displayed.

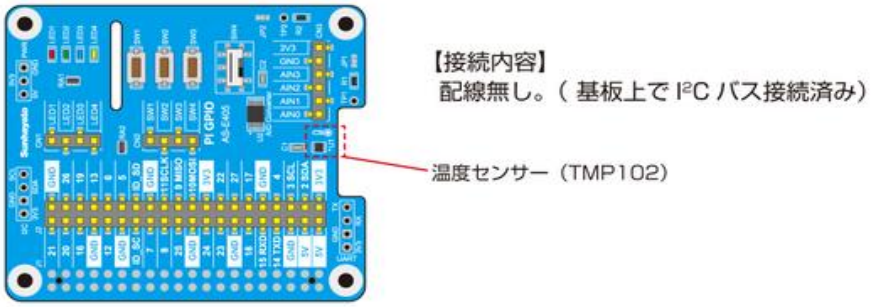
```
$sudo i2cdetect –y 1
 0 1 2 3 4 5 6 7 8 9 a b c d e f
 00: -- -- -- -- -- -- -- -- -- -- -- -- --
 10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
 20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
 30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
 40: -- -- -- -- -- -- -- -- 48 49 -- -- -- -- -- --
 50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
 60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
 70: -- -- -- -- -- -- -- --
```

48 is TMP120 (temperature sensor)
49 is ADS1015 (A / D converter)

5. Temperature sensor (TMP102) usage example

The temperature sensor is connected to the I2C bus on the board, so no wiring is required.

■温度センサー（TMP102）使用例の接続　（TMP102.py）



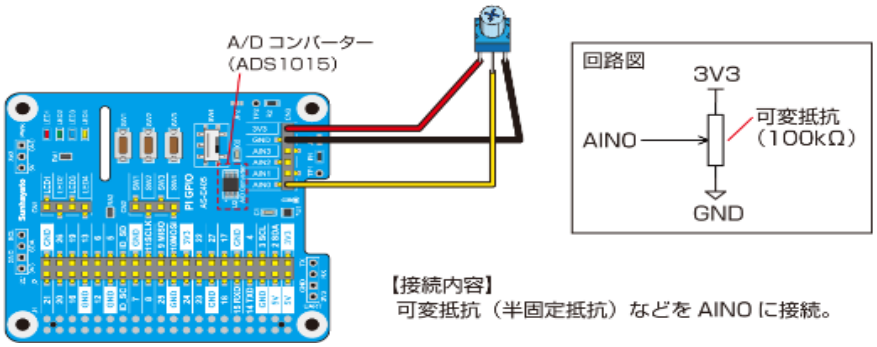
When the following code (**TMP102.py**) is executed, the temperature on the board is displayed every second.

```
#!/usr/bin/python
# coding: utf-8

import smbus
import time

bus = smbus.SMBus(1)    #I2C bus number
address = 0x48          #TMP102 I2C address

#I2C data write (1byte)
def write(reg, value):
    bus.write_byte_data(address, reg, value)

#I2C data read (1byte)
def read(reg):
    value = bus.read_byte_data(address, reg)
    return value

#I2C data write (block)
def blockread(reg, value):
    value = bus.read_i2c_block_data(address, reg, value)
    return value

while 1:
    temp_raw = blockread(0x00, 2)   #I2C gets temperature data (2 bytes)
    #Shaping temperature data
    temp = ((temp_raw[0] << 8) | temp_raw[1]) >> 4
    #Resolution（0.0625℃）times
    temp = temp * 0.0625
    #Display output
    print("temp = " + str(temp) + "*C")

    time.sleep(1)   #1 second
```

6. A / D converter (ADS1015) usage example

Input a voltage in the range of 0-3.3V from a variable resistor to the A / D converter.

■A/D コンバーター（ADS1015）使用例の接続　（ADS1015.py）



When the following code (**ADS1015.py**) is executed, the voltage input to AIN0 is displayed every second.

```
#!/usr/bin/python
# coding: utf-8

import smbus
import time

bus = smbus.SMBus(1)    #I2C bus number
address = 0x49          #I2C address of ADS1015

#I2C data write (1byte)
def write(reg, value):
    bus.write_byte_data(address, reg, value)

#I2C data write (block)
def blockwrite(reg, s):
    bus.write_i2c_block_data(address, reg, s[0:])

#I2C data read (1byte)
def read(reg):
    value = bus.read_byte_data(address, reg)
    return value

#I2C data write (block)
def blockread(reg, value):
    value = bus.read_i2c_block_data(address, reg, value)
    return value

while 1:
    #Initial setting（single,+/-4.096V range(GAIN=1),single-shot mode,1600 sample per sec）
    blockwrite(0x01, [0xC3,0x83])   #Write 0xC383 to the #CONFIG register (AIN0)
#    blockwrite(0x01, [0xD3,0x83])   #Write 0xD383 to the CONFIG register (AIN1)
#    blockwrite(0x01, [0xE3,0x83])   #Write 0xE383 to the CONFIG register (AIN2)
#    blockwrite(0x01, [0xF3,0x83])   #Write 0xF383 to the CONFIG register (AIN3)

    time.sleep(0.01)    #10ms (Slightly wait for A / D conversion processing)

    #Data acquisition
    ad_raw = blockread(0x00, 2)
    #Data formatting
    ad_val = ((ad_raw[0] & 0xFF) << 8) | (ad_raw[1] & 0xFF)
    ad_val = ad_val >> 4
    #+ /-11bit resolution at 4.096V (GAIN = 1 setting)
    ad = ad_val * 4.096 / 2048

    #Display output
    print "ad = " + str(ad) + " V"

    time.sleep(1)   #1 second
```