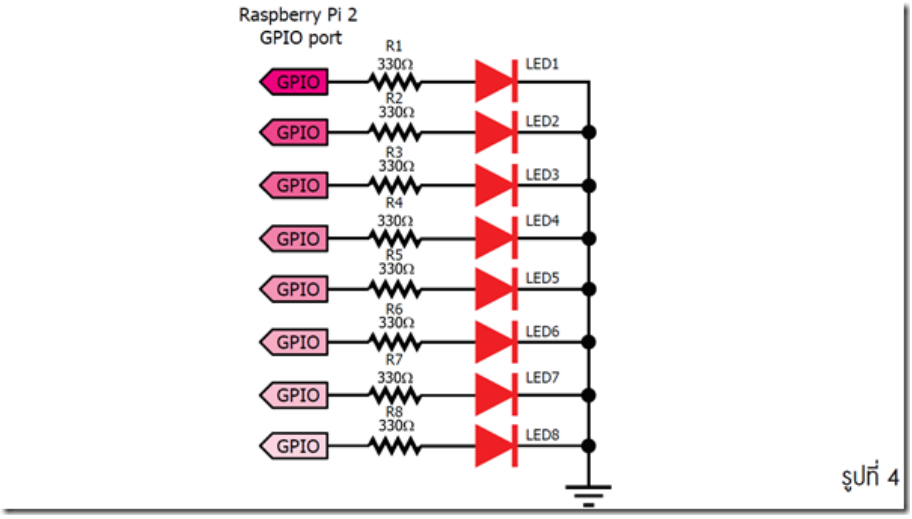
เฉลยโจทย์ การแข่งขันทักษะระดับภาคข้อ 2-6

3. รหัสแอสกี้และเลขฐานสอง



รูปที่ 4

0	0011 0000	O	0100 1111	m	0110 1101
1	0011 0001	P	0101 0000	n	0110 1110
2	0011 0010	Q	0101 0001	o	0110 1111
3	0011 0011	R	0101 0010	p	0111 0000
4	0011 0100	S	0101 0011	q	0111 0001
5	0011 0101	T	0101 0100	r	0111 0010
6	0011 0110	U	0101 0101	s	0111 0011
7	0011 0111	V	0101 0110	t	0111 0100
8	0011 1000	W	0101 0111	u	0111 0101
9	0011 1001	X	0101 1000	v	0111 0110
A	0100 0001	Y	0101 1001	w	0111 0111
B	0100 0010	Z	0101 1010	x	0111 1000
C	0100 0011	a	0110 0001	y	0111 1001
D	0100 0100	b	0110 0010	z	0111 1010
E	0100 0101	c	0110 0011	.	0010 1110
F	0100 0110	d	0110 0100	,	0010 0111
G	0100 0111	e	0110 0101	:	0011 1010
H	0100 1000	f	0110 0110	;	0011 1011
I	0100 1001	g	0110 0111	?	0011 1111
J	0100 1010	h	0110 1000	!	0010 0001
K	0100 1011	I	0110 1001	'	0010 1100
L	0100 1100	j	0110 1010	"	0010 0010
M	0100 1101	k	0110 1011	(	0010 1000
N	0100 1110	l	0110 1100	)	0010 1001

รูปที่ 5

เขียนโปรแกรมรับค่าจากคีย์บอร์ดที่หน้าต่าง Terminal นำค่าที่ได้แสดงผลเป็นค่าเลขฐานสองตามรหัสแอสกี้ที่เกิดขึ้น โดยมีรูปแบบตามรูปที่ 5 ลอจิก “1” LED ติด ลอจิก “0” LED ดับ

*** หมายเหตุ** : รับค่าจาก Serail monitor โดยมีวิธีการดังนี้ (1.) ลองแปลงรหัส ascii เป็นเลขฐาน 2 (2.) รับค่า String เป็นตัวเลข 10100111

(3.1) สามารถแสดงค่าเลขฐานสองที่หน้าต่าง Terminal ได้ถูกต้อง (4 คะแนน)

(3.2) สามารถให้ LED ติดดับตามค่าข้อมูลเลขฐานสองได้ถูกต้อง (6 คะแนน)

จากโจทย์เขียนเป็นโปรแกรมได้ดังนี้

```
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BCM)
my_pin=[6,12,13,19,16,26,20,21]
for pin in my_pin:
    GPIO.setup(pin,GPIO.OUT)
while(1):
    ascii=input("Aacii=")
    bin2str=bin(ord(ascii)+256)
    #0b1xxxxxxxx
    print("bin=",bin2str[3:11])
    GPIO.output(my_pin[0],int(bin2str[2]))
    GPIO.output(my_pin[1],int(bin2str[3]))
    GPIO.output(my_pin[2],int(bin2str[5]))
    GPIO.output(my_pin[3],int(bin2str[6]))
    GPIO.output(my_pin[4],int(bin2str[7]))
    GPIO.output(my_pin[5],int(bin2str[8]))
    GPIO.output(my_pin[6],int(bin2str[9]))
    GPIO.output(my_pin[7],int(bin2str[10]))
```

หัวใจของโปรแกรมนี้คือการระบุตำแหน่งของ LED ให้อยู่ในรูปแบบของ list หรือ array เพื่อระบุตำแหน่งบิตได้สะดวก

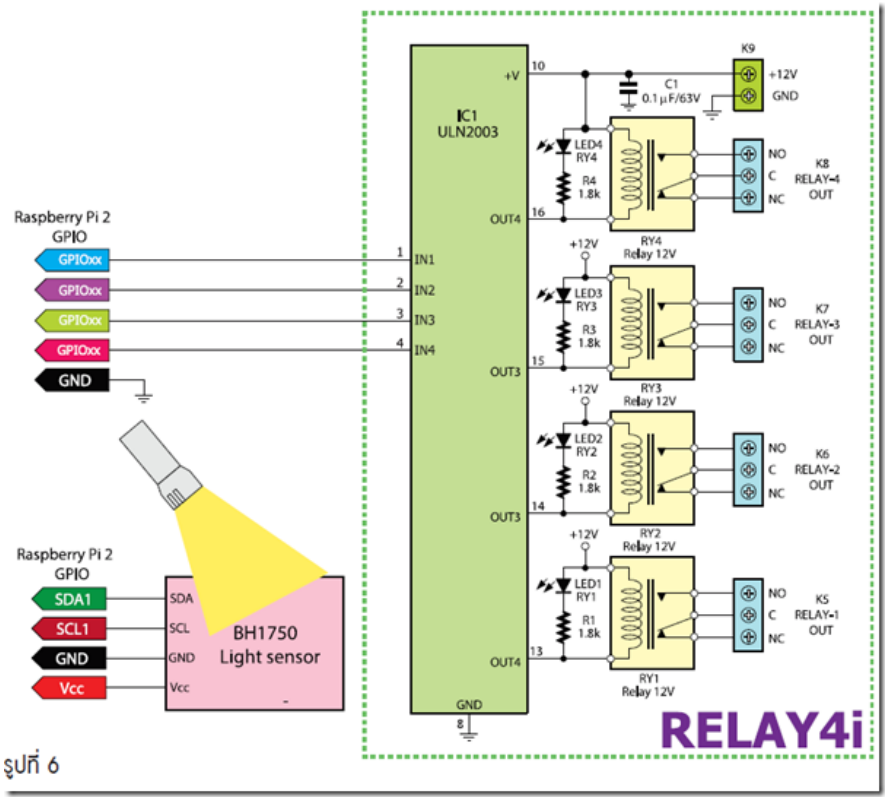
my_pin=[6,12,13,19,16,26,20,21]

อีกคำสั่งคือ ORD สำหรับการเปลี่ยนค่ารหัสแอสกี้ที่อ่านได้จากการกดคีย์บอร์ดให้อยู่ในรูป integer

bin2str=bin(ord(ascii)+256)

หมายเหตุ 256 ที่บวกเพิ่มเพื่อให้เวลาแสดงเลขฐาน 2 แล้วเลข 0 ด้านหน้าไม่ถูกตัดหายไป

4. สวิตช์รหัสกระตุ้นด้วยแสง



รูปที่ 6

สั่งการเปิดปิดรีเลย์ด้วยการสั่งงานด้วยแสง โดยสภาวะปกติ BH1750 จะได้รับแสงปกติของห้อง

(4.1) ใช้กระดาษบังแสงให้ BH1750 แล้วเอาออกรอ 3 วินาที RELAY1 ทำงาน ทำอีกครั้ง RELAY1 หยุดทำงาน สลับกันไปเรื่อย ๆ (Toogle) (4 คะแนน)

(4.2) ใช้กระดาษบังแสงให้ BH1750 แล้วเอาออก 2 ครั้ง รอ 3 วินาที RELAY2 ทำงาน ทำอีกครั้ง RELAY2 หยุดทำงานสลับกันไปเรื่อย ๆ (Toogle) (4 คะแนน)

(4.3) ใช้กระดาษบังแสงให้ BH1750 แล้วเอาออก 3 ครั้ง รอ 3 วินาที RELAY3 ทำงาน ทำอีกครั้ง RELAY3 หยุดทำงาน สลับกันไปเรื่อย ๆ (Toogle) (4 คะแนน)

(4.4) ใช้กระดาษบังแสงให้ BH1750 แล้วเอาออก 4 ครั้ง รอ 3 วินาที RELAY4 ทำงาน ทำอีกครั้ง RELAY4 หยุดทำงาน สลับกันไปเรื่อย ๆ (Toogle) (4 คะแนน)

(4.5) ใช้กระดาษบังแสงให้ BH1750 แล้วเอาออก 5 ครั้ง รอ 3 วินาที RELAY ทุกตัวทำงาน ทำอีกครั้ง RELAY ทุกตัวหยุดทำงาน สลับกันไปเรื่อย ๆ (Toogle) (4 คะแนน)

```
import time
import smbus
bus = smbus.SMBus(1)
import RPi.GPIO as GPIO
import time
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)
addr = 0x23 # i2c adress
my_pin=[16,26,20,21]
```

```
for pin in my_pin:
    GPIO.setup(pin,GPIO.OUT)
st1=0
st2=0
st3=0
st4=0
st5=0
count=0
timer=0

while True:
    time.sleep(0.2)
    data = bus.read_i2c_block_data(addr,0x11)
    lum=(data[1] + (data[0]<<8) / 1.2)
    #print ("Luminosity " ,lum,"lx")

    while (lum<100):
        timer=0
        count+=1
        data = bus.read_i2c_block_data(addr,0x11)
        lum=(data[1] + (data[0]<<8) / 1.2)
        time.sleep(0.2)

        while (lum<100):
            data = bus.read_i2c_block_data(addr,0x11)
            lum=(data[1] + (data[0]<<8) / 1.2)
            time.sleep(0.2)
    print(count)

    timer+=1
    #print ("Timer=",timer)
    if (timer>10):
        if(count==1):
            st1=~st1
            GPIO.output(my_pin[0],st1)
        elif(count==2):
            st2=~st2
            GPIO.output(my_pin[1],st2)
        elif(count==3):
            st3=~st3
            GPIO.output(my_pin[2],st3)
        elif(count==4):
            st4=~st4
            GPIO.output(my_pin[3],st4)
        elif(count==5):
            st5=~st5
            st1=st2=st3=st4=st5
            GPIO.output(my_pin[0],st5)
            GPIO.output(my_pin[1],st5)
            GPIO.output(my_pin[2],st5)
            GPIO.output(my_pin[3],st5)
        count=0
```

การทำงานของโปรแกรม

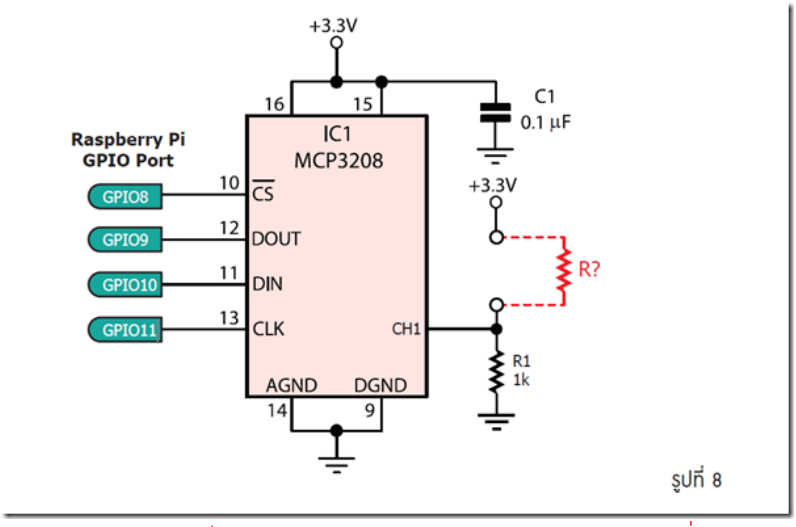
โปรแกรมนี้ใช้ while ในการตรวจสอบ

while (lum<100) ตรวจสอบว่าบังแสง ถ้าใช่ เพิ่มค่าการนับ count

while (lum<100) รอจนกระทั่งเอากระดาษบังแสงออก

ตัวแปร timer จะคอยตรวจสอบว่า ไม่มีการบังแสงนานเกินกว่าเวลาที่กำหนด (2-3 วินาที) ถ้าเกินก็ไปสั่งตรวจสอบ ค่า count เพื่อสั่ง RELAY เปิดหรือปิด

6. โอห์มมิเตอร์ เครื่องวัดค่าความต้านทาน



ต่อวงจรแปลงสัญญาณอะนาลอกเป็นดิจิตอลเข้ากับบอร์ด Raspberry Pi โดยที่เอาต์พุต CH1 ของ MCP2308 ต่อกับวงจรแบ่งแรงดัน โดยกำหนดค่า R1 คงที่ไว้ที่ 1 กิโลโอห์ม ส่วนค่า R? คือจุดที่ใช้ตรวจวัด โดยจะมีการสุ่มตัวต้านทานที่ใช้ในการวัดจำนวน 5 ค่า มีค่าอยู่ระหว่าง 100 โอห์ม ถึง 10 กิโลโอห์ม ค่าที่อ่านได้จะต้องมีค่าความผิดพลาดไม่เกิน 10%

*** หมายเหตุ :** ลองหมุน VR เพื่อดูค่าความต้านทาน

6.1) ค่าความต้านทานเท่ากับ_____อ่านค่าได้ _____(2 คะแนน)

6.2) ค่าความต้านทานเท่ากับ _____อ่านค่าได้_____(2 คะแนน)

6.3) ค่าความต้านทานเท่ากับ_____อ่านค่าได้_____(2 คะแนน)

6.4) ค่าความต้านทานเท่ากับ_____อ่านค่าได้_____(2 คะแนน)

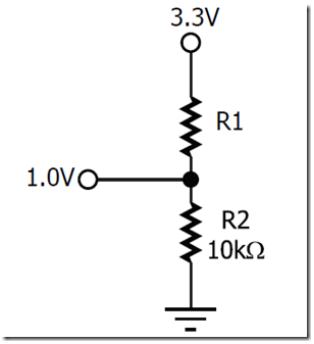
6.5) ค่าความต้านทานเท่ากับ_____อ่านค่าได้_____(2 คะแนน)

จากโจทย์เขียนเป็นโปรแกรมได้ดังนี้

```
import spidev
import time
analog_ch = 1
spi = spidev.SpiDev()
spi.open(0,0)
Vin=3.3
R1=1000
def readADC(adcnum):
    if adcnum > 7 or adcnum < 0:
            return -1;
    r = spi.xfer2([4 | 2 | (adcnum >> 2), (adcnum & 3) << 6, 0])
    adcout = ((r[1] & 15) << 8) + r[2]
    return adcout
while True:
    value = readADC(analog_ch)
    voltage = value*3.3/4096
    if voltage > 0:
        ohm = (R1 * (Vin-voltage))/voltage
        #ohm = ((Vin*R1)/voltage)-R1
        print("R? = %d"  % ohm)
    time.sleep(0.3)
```

การทำงานของโปรแกรม

จากกฎของวงจรแบ่งแรงดัน



R1 = (R2 * (Vin – Vo))/Vo

ค่าของ Vo จะมาจากการอ่านค่าอะนาลอกจาก MCP3208 ซึ่งเป็นแบบ12 บิต แรงดันไฟเลี้ยง 3.3V

voltage = value*3.3/4096

การวัดค่าความต้านในย่าน 1-10 กิโลโอห์มก็ทำได้ไม่ยาก