	เฉลยใบงานที่ 5	ครั้งที่ 9
	หน่วยที่ GPIO ทดลองใช้งาน Char LCD	รวม 8 ชั่วโมง
เรื่องการใช้งานGPIO ส่วนของทดลองใช้งาน Char LCD 16 ตัวอักษร 2 บรรทัด		
ชื่อ.....ชั้น ..... วันที่ ...../...../255.....		

- http://www.raspberrypi-spy.co.uk/2012/07/16x2-lcd-module-control-using-python/

ลำดับขั้นการทดลอง

1. โปรแกรมทดลองใช้งาน Char LCD 16 ตัวอักษร 2 บรรทัดดังนี้

โปรแกรม program ความหมายcomment

```
import RPi.GPIO as GPIO
import time
LCD_RS = 21                                  # Define GPIO to LCD mapping
LCD_E  = 20
LCD_D4 = 26
LCD_D5 = 16
LCD_D6 = 19
LCD_D7 = 13
LCD_WIDTH = 16                               # Maximum characters per line
LCD_CHR = True
LCD_CMD = False
LCD_LINE_1 = 0x80                            # LCD RAM address for the 1st line
LCD_LINE_2 = 0xC0                            # LCD RAM address for the 2nd line
E_PULSE = 0.0005                             # Timing constants
E_DELAY = 0.0005
def main():                                  # Main program block
        GPIO.setwarnings(False)
        GPIO.setmode(GPIO.BCM)               # Use BCM GPIO numbers
        GPIO.setup(LCD_E, GPIO.OUT)          # E
        GPIO.setup(LCD_RS, GPIO.OUT)         # RS
        GPIO.setup(LCD_D4, GPIO.OUT)         # DB4
        GPIO.setup(LCD_D5, GPIO.OUT)         # DB5
        GPIO.setup(LCD_D6, GPIO.OUT)         # DB6
        GPIO.setup(LCD_D7, GPIO.OUT)         # DB7
        lcd_init()                           # Initial display
        while True:
                lcd_string("pathumthani",LCD_LINE_1)     # Send some test
                lcd_string("college",LCD_LINE_2)
                time.sleep(5)                # 3 second delay
                lcd_string("RASPBERRY PI",LCD_LINE_1)
                lcd_string("microcontroller",LCD_LINE_2)
                time.sleep(5)
```

```
def lcd_init():                                 # Initial display
        lcd_byte(0x33,LCD_CMD)                  # 110011 Initialize
        lcd_byte(0x32,LCD_CMD)                  # 110010 Initialize
        lcd_byte(0x06,LCD_CMD)                  # 000110 Cursor move direction
        lcd_byte(0x0C,LCD_CMD)                  # 001100 Display On,Cursor Off, Blink Off
        lcd_byte(0x28,LCD_CMD)                  # 101000 Data length, number of lines, font size
        lcd_byte(0x01,LCD_CMD)                  # 000001 Clear display
        time.sleep(E_DELAY)
def lcd_byte(bits, mode):
        GPIO.output(LCD_RS, mode)
        GPIO.output(LCD_D4, False)
        GPIO.output(LCD_D5, False)
        GPIO.output(LCD_D6, False)
        GPIO.output(LCD_D7, False)
        if bits&0x10==0x10:
                GPIO.output(LCD_D4, True)
        if bits&0x20==0x20:
                GPIO.output(LCD_D5, True)
        if bits&0x40==0x40:
                GPIO.output(LCD_D6, True)
        if bits&0x80==0x80:
                GPIO.output(LCD_D7, True)
        lcd_toggle_enable()
        GPIO.output(LCD_D4, False)
        GPIO.output(LCD_D5, False)
        GPIO.output(LCD_D6, False)
        GPIO.output(LCD_D7, False)
        if bits & 0x01 == 0x01:                 #10
                GPIO.output(LCD_D4, True)
        if bits & 0x02 == 0x02:                 #
                GPIO.output(LCD_D5, True)
        if bits & 0x04 == 0x04: #
                GPIO.output(LCD_D6, True)
        if bits & 0x08 == 0x08: #80
                GPIO.output(LCD_D7, True)
        lcd_toggle_enable()
def lcd_toggle_enable():
        time.sleep(E_DELAY)
        GPIO.output(LCD_E, True)
        time.sleep(E_PULSE)
        GPIO.output(LCD_E, False)
```

```
        time.sleep(E_DELAY)
def lcd_string(message,line):
        message = message.ljust(LCD_WIDTH," ")
        lcd_byte(line, LCD_CMD)
        for i in range(LCD_WIDTH):
                lcd_byte(ord(message[i]),LCD_CHR)
main()
```

3. โปรแกรมทดลองใช้งาน Char LCD 16 ตัวอักษร 2 บรรทัดแสดงค่าเวลาดังนี้

โปรแกรม program ความหมายcomment

```
import RPi.GPIO as GPIO
import time
import datetime
from datetime import datetime


LCD_RS = 21                             # Define GPIO to LCD mapping
LCD_E  = 20
LCD_D4 = 26
LCD_D5 = 16
LCD_D6 = 19
LCD_D7 = 13
LCD_WIDTH = 16                          # Maximum characters per line
LCD_CHR = True
LCD_CMD = False
LCD_LINE_1 = 0x80                       # LCD RAM address for the 1st line
LCD_LINE_2 = 0xC0                       # LCD RAM address for the 2nd line
E_PULSE = 0.0005                        # Timing constants
E_DELAY = 0.0005
def main():                             # Main program block
        GPIO.setwarnings(False)
        GPIO.setmode(GPIO.BCM)          # Use BCM GPIO numbers
        GPIO.setup(LCD_E, GPIO.OUT)     # E
        GPIO.setup(LCD_RS, GPIO.OUT)    # RS
        GPIO.setup(LCD_D4, GPIO.OUT)    # DB4
        GPIO.setup(LCD_D5, GPIO.OUT)    # DB5
        GPIO.setup(LCD_D6, GPIO.OUT)    # DB6
        GPIO.setup(LCD_D7, GPIO.OUT)    # DB7
        lcd_init()                      # Initial display
        while True:
                lcd_string("pathumthani",LCD_LINE_1)    # Send some test
                lcd_string("college",LCD_LINE_2)
```

```
        time.sleep(3)                                   # 3 second delay
        lcd_string("RASPBERRY PI",LCD_LINE_1)
        lcd_string("microcontroller",LCD_LINE_2)
        time.sleep(3)
        now=time.strftime(“%d/%m/%y-%H:%M:%S”)
        lcd_string(str(now),LCD_LINE_1)
        time.sleep(3)
def lcd_init():                                         # Initial display
    lcd_byte(0x33,LCD_CMD)                              # 110011 Initialize
    lcd_byte(0x32,LCD_CMD)                              # 110010 Initialize
    lcd_byte(0x06,LCD_CMD)                              # 000110 Cursor move direction
    lcd_byte(0x0C,LCD_CMD)                              # 001100 Display On,Cursor Off, Blink Off
    lcd_byte(0x28,LCD_CMD)                              # 101000 Data length, number of lines, font size
    lcd_byte(0x01,LCD_CMD)                              # 000001 Clear display
    time.sleep(E_DELAY)
def lcd_byte(bits, mode):
    GPIO.output(LCD_RS, mode)
    GPIO.output(LCD_D4, False)
    GPIO.output(LCD_D5, False)
    GPIO.output(LCD_D6, False)
    GPIO.output(LCD_D7, False)
    if bits&0x10==0x10:
        GPIO.output(LCD_D4, True)
    if bits&0x20==0x20:
        GPIO.output(LCD_D5, True)
    if bits&0x40==0x40:
        GPIO.output(LCD_D6, True)
    if bits&0x80==0x80:
        GPIO.output(LCD_D7, True)
    lcd_toggle_enable()
    GPIO.output(LCD_D4, False)
    GPIO.output(LCD_D5, False)
    GPIO.output(LCD_D6, False)
    GPIO.output(LCD_D7, False)
    if bits & 0x01 == 0x01:                             #10
        GPIO.output(LCD_D4, True)
    if bits & 0x02 == 0x02:                             #
        GPIO.output(LCD_D5, True)
    if bits & 0x04 == 0x04: #
        GPIO.output(LCD_D6, True)
    if bits & 0x08 == 0x08: #80
```

```
                GPIO.output(LCD_D7, True)
        lcd_toggle_enable()
def lcd_toggle_enable():
        time.sleep(E_DELAY)
        GPIO.output(LCD_E, True)
        time.sleep(E_PULSE)
        GPIO.output(LCD_E, False)
        time.sleep(E_DELAY)
def lcd_string(message,line):
        message = message.ljust(LCD_WIDTH," ")
        lcd_byte(line, LCD_CMD)
        for i in range(LCD_WIDTH):
                lcd_byte(ord(message[i]),LCD_CHR)
main()
```