DSI2599
智慧加工監測系統

發想動機

Part I
摘要

Part II
硬體介紹

Part III
程式介紹

Part IV
成果展示

第一章 摘要

1-1 案例簡介與應用

本案例適用於工業4.0，監控生產參數，可應對各種加工機台，或是依照個人需求安裝需要的感測器，彙整出加工資訊，並傳至雲端平台，可以實現遠端監控加工狀態，若不符合預期參數，會發出警示予以提醒

另外可以將讀出數據，使用類神經網路訓練成有效的預測模型，將訓練參數輸入至mcu中，就可以即時利用感測器收到的數據，預測出平面粗度，並把結果傳至雲端，可應用於工業之機械加工，當作自動化加工的參考依據

1-2 所需材料

1.DSI2599 x 1

2.強力磁鐵 x 1

3.MPU6050 x 1

4.FC03 x 1

5.排線 x 少許

第二章

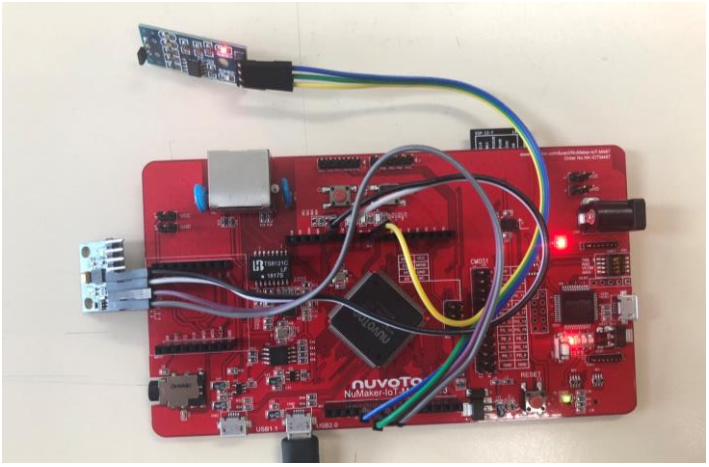
硬體介紹

1-1 接線介紹

1-2-1 腳位介紹-GPIO
1-2-2 腳位介紹-I2C

1-3-1 感應器介紹-MPU6050
1-3-2 感應器介紹-FC03

2-1 接線介紹



本次使用兩個電源輸出，一個GPIO，以及一組I2C

2-2-1 腳位介紹-GPIO

Board	Board Pin Name	CPU Pin Name
NuMaker-PFM-M487	D0	B2
NuMaker-IOT-M487	D1	B3
	D2	C9
	D3	C10
	D4	C11
	D5	C12
	D6	E4
	D7	E5
	D8	A5
	D9	A4
	D10	A3
	D11	A0
	D12	A1
	D13	A2

GPIO為較基本的功能，M487提供14個GPIO，但是本次案例會使用I2C，必須預留腳位，本示範案例需要預留D10跟D13

2-2-1 腳位介紹-GPIO

```
1.  from pyb import Pin
2.
3.  p_d0 = Pin(Pin.board.D0, Pin.OUT)     # create output pin on GPIO B2
4.  p_d0.value(1)                         # set pin to on/high
5.
6.  p_d1 = Pin(Pin.board.D1, Pin.IN)      # create input pin on GPIO B3
7.  print(p_d1.value())                   # get value, 0 or 1
8.
9.  Pin.board.D2.af_list()                # list available alternate functions on GPIO C9
10.
11. def sw2_callback(pin):                # define sw2 (switch button 2) callback
12.     print(pin)
13.
14. sw2 = Pin.board.SW2
15. sw2.irq(handler=sw2_callback, trigger=Pin.IRQ_RISING)  # configure sw2 to interrupt
```

GPIO語法介紹，使用以上語法可呼叫GPIO功能

2-2-2 腳位介紹-I2C

Board	I2C Pin Name	Board Pin Name	CPU Pin Name
NuMaker-PFM-M487 NuMaker-IOT-M487	I2C0_SCL	D8	A5
	I2C0_SDA	D9	A4
	I2C1_SCL	D10	A3
	I2C1_SDA	D13	A2

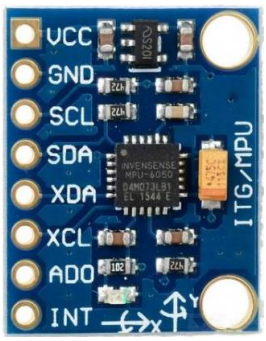
根據nuvoTon提供的User Manual，可以得知I2C的Pin位置，M487提供兩組I2C，使用的時候需要呼叫函式庫，這次示範案例使用的腳位是D10跟D13

2-2-2 腳位介紹-I2C

```
1. from pyb import I2C
2.
3. i2c = I2C(1, I2C.MASTER)     # create and initiate I2C1 as a master
4. i2c.scan()        # scan for slaves on the bus, returning a list of valid addresses.
   Only valid when in master mode.
5. i2c.is_ready(0x42)           # check if slave 0x42 is ready
6. i2c.send('123', 0x42)        # send 3 bytes to slave with address 0x42
7. data = bytearray(3)          # create a buffer
8. i2c.recv(data)               # receive 3 bytes, writing them into data
9. i2c.deinit()                 # turn off the peripheral
```

I2C語法介紹，使用以上語法可呼叫I2C功能

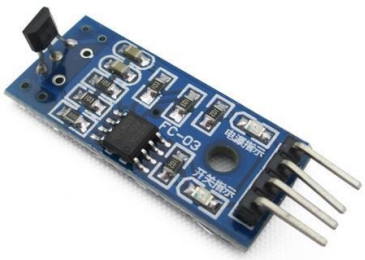
2-3-1 感應器介紹-MPU-6050



工作資訊：

供電電源：3.3V ~ 5V（內部低壓差穩壓）
資料讀取：I2C(16位元資料輸出)
陀螺儀範圍：±250 / 500 / 1000 / 2000°/s
加速度範圍：±2 / ±4 / ±8 / ±16g
引腳間距2.54mm

2-3-2 感應器介紹-FC03



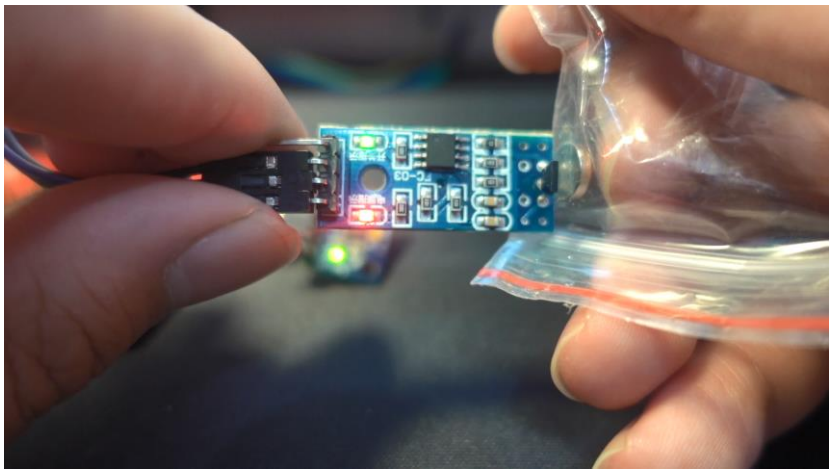
工作原理：

霍爾效應指當電流通過固體導體（或半導體）且放在磁場內，導體內的電荷載子受到勞倫茲力而偏向一邊，進而產生電壓（霍爾電壓）。根據此效應，便可偵測磁場、磁鐵，利用磁鐵置於旋轉物體，根據感應到的次數來計算RPM

工作資訊：

供電電源：3.3V ~ 5V（內部低壓差穩壓）
資料讀取：GPIO

2-3-2 感應器介紹-FC03



感應到磁鐵後，上面的綠色LED燈會亮起

第三章
軟體介紹

2-1-1程式介紹-MPU6050.py
2-1-2 程式介紹-main.py

3-1-1 程式介紹-main.py

```
import pyb
import network
import usocket as socket
from pyb import I2C
from MPU6050 import MPU6050

SSID = "夢繪"
PASS = "yumee0525"
HOST = "ideaschain.com.tw"
API_URL = "iiot.ideaschain.com.tw"
DEVICE_KEY = "0DezkympiDB6pgfSzNsY"
```

這邊設定要載入I2C的函式庫，這樣GPIO才會轉換成I2C，以及載入為MPU6050編輯的函式庫，接下來要設定上傳IDEAS Chain平台的網址跟DEVICE_KEY，平台上稱為存取權杖

3-1-1 程式介紹-main.py

```
def wifi():
    try:
        print("connecting to wifi")
        wlan = network.WLAN()
        wlan.connect(SSID, PASS)

    except:
        print("Wifi module initial error, reconnecting......")
        pyb.delay(1000)
        wifi()
```

這邊的功能是如果成功連上Wi-Fi，就會顯示”connecting to wifi”並進行後續的步驟，若是無法連上Wi-Fi，1秒後重新連接，這邊要注意的是，若是無法連接Wi-Fi ，就不會進行後續的動作，會一直重複連接Wi-Fi

3-1-1 程式介紹-main.py

```
wifi()
print("Wi-Fi connect")

addr = socket.getaddrinfo(HOST, 80)[0][-1] # 取得連線到伺服器的相關訊息
print(addr) # 顯示取得的address訊息內容

i2c = I2C(1, I2C.MASTER) # 選用I2C1
mpu = MPU6050(i2c)

def get_data():

    data = {"GX":mpu.read_Gyro_x(),"GY":mpu.read_Gyro_y(),"GZ":mpu.read_Gyro_z()}

    json_temp = ','.join(["\"%s\":\"%d\"" % (key,value) for key, value in data.items()])

    return '{'+json_temp+'}'
```

M487有兩組I2C，分別為I2C0與I2C1，這邊使用I2C1，詳細的介紹1-1腳位介紹有提到，後面就是列出GX、GY、GZ的值

3-1-2 程式介紹-MPU6050.py

```
from pyb import I2C

MPU_ADDR=0X68
WHO_AM_I_VAL = MPU_ADDR

MPU_PWR_MGMT1_REG = 0x6B
MPU_GYRO_CFG_REG  = 0X1B
MPU_ACCEL_CFG_REG = 0X1C
MPU_SAMPLE_RATE_REG=0X19
MPU_CFG_REG=0X1A

MPU_INT_EN_REG=0X38
MPU_USER_CTRL_REG=0X6A
MPU_FIFO_EN_REG=0X23
MPU_INTBP_CFG_REG=0X37
MPU_DEVICE_ID_REG=0X75
```

MPU_ADDR這邊設定裝置位址引腳，有兩種選擇，low為0x68、high為0x69

3-1-2 程式介紹-MPU6050.py

```
MPU_GYRO_XOUTH_REG=0X43
MPU_GYRO_XOUTL_REG=0X44
MPU_GYRO_YOUTH_REG=0X45
MPU_GYRO_YOUTL_REG=0X46
MPU_GYRO_ZOUTH_REG=0X47
MPU_GYRO_ZOUTL_REG=0X48

MPU_ACCEL_XOUTH_REG=0X3B
MPU_ACCEL_XOUTL_REG=0X3C
MPU_ACCEL_YOUTH_REG=0X3D
MPU_ACCEL_YOUTL_REG=0X3E
MPU_ACCEL_ZOUTH_REG=0X3F
MPU_ACCEL_ZOUTL_REG=0X40
MPU_TEMP_OUTH_REG=0X41
MPU_TEMP_OUTL_REG=0X42
```

設定MPU_GYRO、MPU_ACCEL、MPU_TEMP裝置引腳

3-1-2 程式介紹-MPU6050.py

```
    def read_Gyro_x(self):
        x = self._read_s16(MPU_GYRO_XOUTH_REG)
        return x
    def read_Gyro_y(self):
        y = self._read_s16(MPU_GYRO_YOUTH_REG)
        return y
    def read_Gyro_z(self):
        z = self._read_s16(MPU_GYRO_ZOUTH_REG)
        return z
```

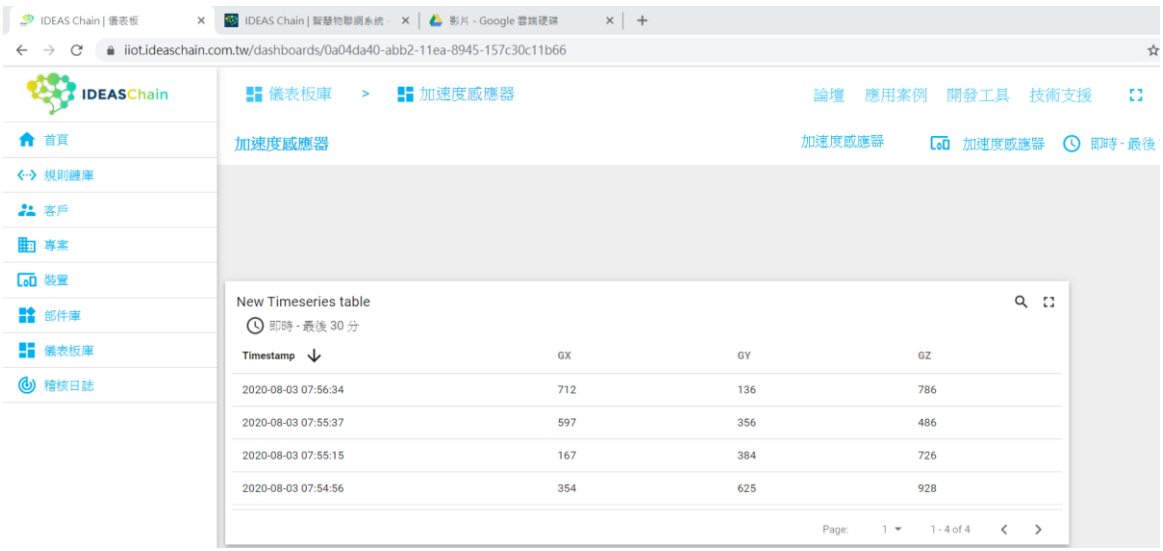
設定X、Y、Z軸的陀螺儀，到時候在main.py若要讀取，以X軸為例，輸入mpu.read_Gyro_x()即可

3-1-2 程式介紹-MPU6050.py

```
    def read_Accel_x(self):
        x = self._read_s16(MPU_ACCEL_XOUTH_REG)
        return x
    def read_Accel_y(self):
        y = self._read_s16(MPU_ACCEL_YOUTH_REG)
        return y
    def read_Accel_z(self):
        z = self._read_s16(MPU_ACCEL_ZOUTH_REG)
        return z
```

設定X、Y、Z軸的加速規，到時候在main.py若要讀取，以X軸為例，輸入mpu.read_Accel_z()即可

第四章
成果展示


IDEAS Chain | 儀表板
IDEAS Chain | 智慧物聯網系統
影片 - Google 雲端硬碟
iiot.ideaschain.com.tw/dashboards/0a04da40-abb2-11ea-8945-157c30c11b66
IDEASChain
儀表板庫 > 加速度感應器
論壇 應用案例 開發工具 技術支援
首頁
規則鏈庫
客戶
專案
裝置
部件庫
儀表板庫
稽核日誌
加速度感應器
加速度感應器 加速度感應器 即時 - 最後
New Timeseries table
即時 - 最後 30 分
Timestamp GX GY GZ
2020-08-03 07:56:34 712 136 786
2020-08-03 07:55:37 597 356 486
2020-08-03 07:55:15 167 384 726
2020-08-03 07:54:56 354 625 928
Page: 1 1 - 4 of 4