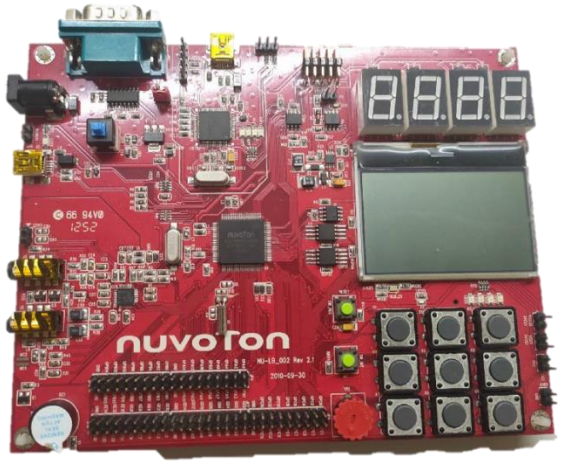
微處理機系統與介面技術 LAB 6

系所：通訊三　　學號 :409430030　　姓名:翁佳煌

<實驗器材>

NUC 140 開發板



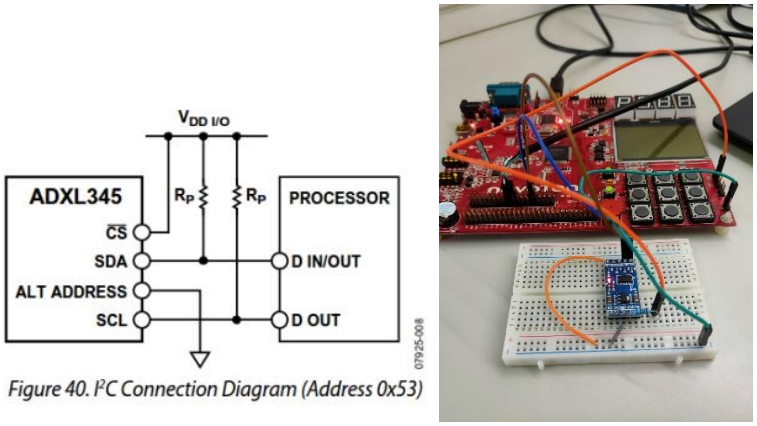
PL2303 USB to UART 線

ADXL345

<實驗過程與方法>

首先，先依照線路圖完成接線，如下**圖 1.1**。

- SDA --------- M0 SDA(GPA8)
- SCL --------- M0 SCL(GPA9)
- Don’t care: VS, INT1, INT2
- CS, SCL, SDA is already pull high for this module
- SDO is already pull low



▲圖 1.1

首先看到 main 函式，第 244 行為 I2C 的初始化，等等下半部會詳細描述，而第 246 行~253 行的 while 迴圈部分是不段讀取 ADXL345 的 x,y,z 三軸檢測到的數值，讀取的腳位在 ADXL 的位置分別為:

- DATAX0(0x32), DATAX1(0x33)
- DATAY0(0x34), DATAY1(0x35)
- DATAZ0(0x36), DATAZ1(0x37)

其中如 xaxis = ((DataX1 << 8) | DataX0)的運算是因為輸出數據為 two complement，DATAx0 為 LSB，DATAx1 為 MSB。

```
228  int32_t main(void)
229  {
230      float axis[3]={0};
231      int i,j;
232      /* Unlock protected registers */
233      SYS_UnlockReg();
234
235      /* Init System, IP clock and multi-function I/O */
236      SYS_Init();
237
238      /* Init UART0 for printf */
239      UART0_Init();
240
241      /* Lock protected registers */
242      SYS_LockReg();
243
244      I2C0_Init();
245
246      while(1){
247        for(i=0;i<3;i++){
248          axis[i]=((float)((I2C_Read(0x33+2*i) << 8) | I2C_Read(0x32+2*i))/256);
249          //default sensitivity is (+-)2g , so we need divide 256 to get (+-)1g
250        }
251        CLK_SysTickDelay(1000000);
252        printf("x=%.2f  y=%.2f  z=%.2f\n",axis[0],axis[1],axis[2]);
253      }
254      I2C0_Close();
255
256      while(1);
257  }
```

▲圖 1.2

再來看到上**圖 1.2** 第 244 行呼叫的 I2C0_Init 函式，下**圖 1.3** 開始對 ADXL 的初始化，第 177~179 行開啟中斷的設定並決定中斷發生後會進入 IRQ Handler，然後在 IRQ Handler 裡面會呼叫 s_I2C0HandlerFn。而 s_I2C0HandlerFn 會接著呼叫 I2C_status 函式。

第 181~186 行為 ADXL345 的初始化:

- DATA_FORMAT(0x31): 0x0B
- POWER_CTL(0x2D): 0x08
- FIFO_CTL(0x38): 0x80

第 188~204 行則為根據 ADXL345 的 datasheet 設定它 x,y,z 三軸的 offset。

```
172  void I2C0_Init(void)
173  {
174      int offset[3],loop=30,i,j;
175      printf("\nADXL init \n");
176      I2C_Open(I2C0, 100000);   // Open I2C module and set bus clock //
177      I2C_EnableInt(I2C0);
178      NVIC_EnableIRQ(I2C0_IRQn);
179      s_I2C0HandlerFn = (I2C_FUNC)I2C_status;
180
181      I2C_Write(0x31,0x0b); //DATA_FORMAT(0x31): 0x0B
182      //The default configuration of the interrupt pins is active high.
183      //It can be changed to active low by setting the INT_INVERT bit.
184      I2C_Write(0x2d,0x08); //POWER_CTL(0x2D): 0x08
185      I2C_Write(0x38,0x80); //FIFO_CTL(0x38): 0x80
186      //The watermark bit is set when the number of samples in FIFO equals the value stored in the samples bits
187
188      //offset
189      I2C_Write(0x1e,0);
190      I2C_Write(0x1f,0);
191      I2C_Write(0x20,0);
192
193      printf("device id: %x\n",(uint8_t)I2C_Read(0x00));
194      for(j=0;j<3000000;j++){}
195      for(j=0;j<loop;j++) //set the offset value. First, we need sample the value.
196      {
197          for(i=0;i<3;i++){
198              offset[i] += ((I2C_Read(0x33+(2*i))<<8) | I2C_Read(0x32+(2*i)));
199  //The output data is twos complement, with DATAx0 as the least significant byte and DATAx1 as the most significant byte
200          }
201      }
202      I2C_Write(0x1e,-1*offset[0]/(4*loop));
203      I2C_Write(0x1f,-1*offset[1]/(4*loop));
204      I2C_Write(0x20,-1*(offset[2]-256)/(4*loop));
205
206  }
```

▲圖 1.3

接下來看到下**圖 1.4**，I2C_Read 和 I2C_Write，設有 g_u8MstEndflag，此變數代表是否終止的 Flag，若 1 則代表終止。

不管事 I2C_Read 或是 I2C_Write 都會送一個 start 的訊號，之後會進入中斷，並在第 35 行利用 u32Status 得到目前的狀態，第 38 行判斷是否超時中斷，如果是就清除旗標，如果不是則會在第 42 行再一次呼叫函示 I2C_status。

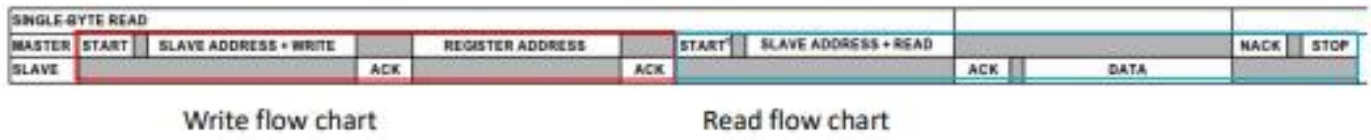
```
31  /*---------------------------------------------------------------------------
32  void I2C0_IRQHandler(void)
33  {
34      uint32_t u32Status;
35      u32Status = I2C_GET_STATUS(I2C0); //(i2c)->I2CSTATUS
36
37      //printf("inperrupt\n");
38      if(I2C_GET_TIMEOUT_FLAG(I2C0))
39          I2C_ClearTimeoutFlag(I2C0);   // Clear I2C0 Timeout Flag
40      else{
41          //printf("Status %x\n", u32Status);  //8 18 28 10 40 58
42          s_I2C0HandlerFn(u32Status);         //Keep going~~
43      }
44  }
```

```
97  int8_t I2C_Read(uint8_t DataAddr)
98  {
99      g_u8MstEndFlag = 0;
100     R=1;
101     g_DataAddr = DataAddr;
102     I2C_SET_CONTROL_REG(I2C0, I2C_I2CON_STA);
103     while(g_u8MstEndFlag == 0);
104     return (I2C_GET_DATA(I2C0)); //I2C0->I2CDAT
105 }
106 void I2C_Write(uint8_t DataAddr,uint8_t Data)
107 {
108     g_u8MstEndFlag = 0;
109     R=0;
110     g_DataAddr = DataAddr; //addr at adxl 345
111     g_Data = Data;
112     I2C_SET_CONTROL_REG(I2C0, I2C_I2CON_STA);//START
113     while(g_u8MstEndFlag == 0);
114 }
115
```

▲圖 1.4

再來下**圖 1.5** 則是和 sample code 一樣，主要再負責判斷當前狀態與相對應要做的事情，依照著最下方圖中的資料格式做傳輸。

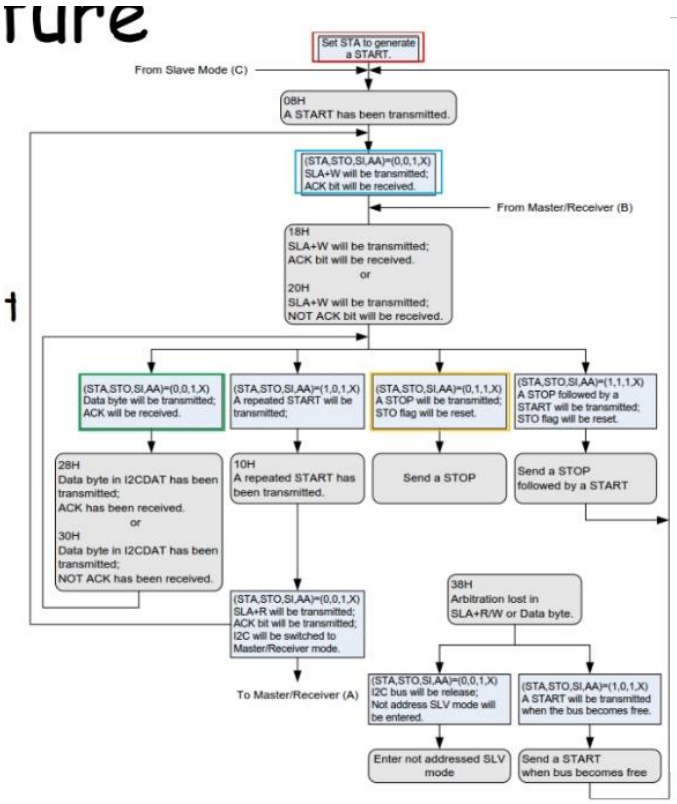
```
46  void I2C_status(uint32_t u32Status)
47  {
48      if(u32Status == 0x08)                       // START has been transmitted and prepare SLA+W
49      {
50          I2C_SET_DATA(I2C0, (DeviceAddr << 1));  //Write SLA+W to Register I2CDAT    (i2c)->I2CDAT = u8Data+0(W)
51          I2C_SET_CONTROL_REG(I2C0, I2C_I2CON_SI);
52      }
53      else if(u32Status == 0x18)                  // SLA+W has been transmitted and ACK has been received
54      {
55          I2C_SET_DATA(I2C0, g_DataAddr);
56          I2C_SET_CONTROL_REG(I2C0, I2C_I2CON_SI);
57      }
58      else if(u32Status == 0x20)                  // SLA+W has been transmitted and NACK has been received
59      {
60          I2C_SET_CONTROL_REG(I2C0, I2C_I2CON_STA_STO_SI);
61      }
62      else if(u32Status == 0x28)                  /* DATA has been transmitted and ACK has been received */
63      {
64          if(!R){
65              if(I2C0->I2CDAT != g_Data){         //Write register
66                  I2C_SET_DATA(I2C0, g_Data);
67                  I2C_SET_CONTROL_REG(I2C0, I2C_I2CON_SI);
68              }
69              else{                               //Write data
70                  I2C_SET_CONTROL_REG(I2C0, I2C_I2CON_STO_SI);
71                  g_u8MstEndFlag = 1;
72              }
73          }
74          else{
75              I2C_SET_CONTROL_REG(I2C0, I2C_I2CON_STA_SI); // Restart!!!
76          }
77      }
78      else if(u32Status == 0x10)                  /* Repeat START has been transmitted and prepare SLA+R */
79      {
80          I2C_SET_DATA(I2C0, ((DeviceAddr << 1) | 0x01));   /* Write SLA+R to Register I2CDAT */
81          I2C_SET_CONTROL_REG(I2C0, I2C_I2CON_SI);
82      }
83      else if(u32Status == 0x40)                  // SLA+R has been transmitted and ACK has been received
84      {
85          I2C_SET_CONTROL_REG(I2C0, I2C_I2CON_SI);
86      }
87      else if(u32Status == 0x48)                  //SLA+R has been transmitted and NOT ACK has been received.
88      {
89          I2C_SET_CONTROL_REG(I2C0, I2C_I2CON_STA_STO_SI);
90      }
91      else if(u32Status == 0x58)                  // DATA has been received and NACK has been returned
92      {
93          I2C_SET_CONTROL_REG(I2C0, I2C_I2CON_STO_SI);
94          g_u8MstEndFlag = 1;
95      }
96  }
97
```



Write flow chart　　Read flow chart

▲圖 1.5

下**圖 1.6** 左邊為對應的狀態流程圖，右邊為 datasheet 中各個數值代表的狀態和對應的描述。



Master mode	
STATUS	Description
0x08	Start
0x10	Master Repeat Start
0x18	Master Transmit Address ACK
0x20	Master Transmit Address NACK
0x28	Master Transmit Data ACK
0x30	Master Transmit Data NACK
0x38	Master Arbitration Lost
0x40	Master Receive ACK
0x48	Master Receive NACK
0x50	Master Receive ACK
0x58	Master Receive NACK
0x00	Bus error

▲圖 1.6

<心得與收穫>

這次的 LAB 真的如助教所言特別的困難阿!我連續好幾天嘗試到半夜都無法單獨寫出來，直到最後去問了各個大神才了解 CODE 如何撰寫，我這才發現雖然明白 I2C 的原理是很簡單的，只需要 2 條線就能實現資料傳輸的方便功能，但在實作 CODE 上真的是燒壞了我的腦袋，這次的實驗雖然不是靠我自己獨立完成的，但大家一起花時間討論出來的結果卻令人雀躍不已。還剩下最後一次 LAB 為 SPI，希望能把握好每次學習的機會，讓自己更加進步。