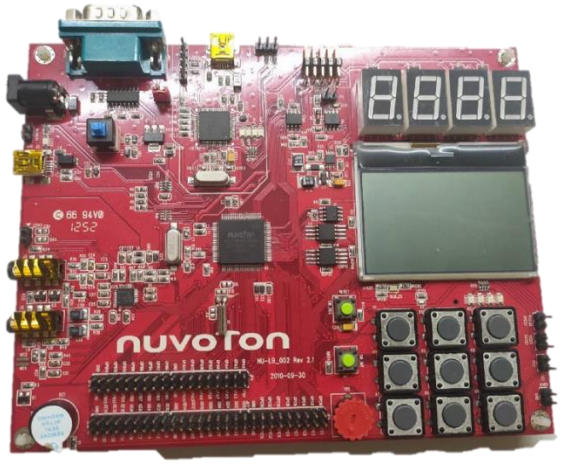
微處理機系統與介面技術 LAB 3

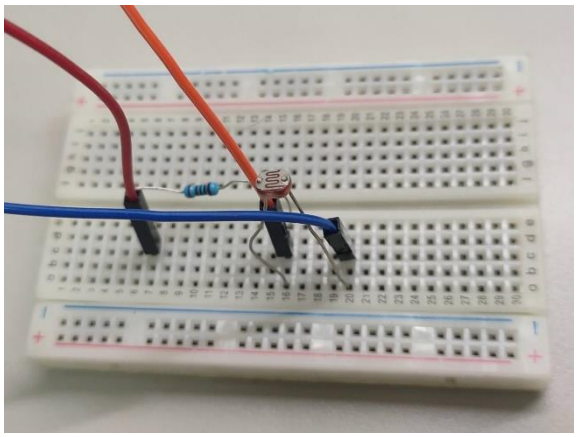
系所：通訊三　　學號 :409430030　　姓名:翁佳煌

<實驗器材>

NUC 140 開發板

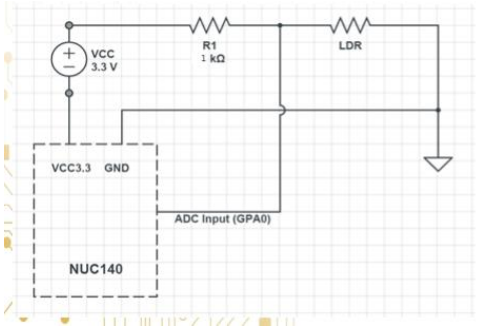


麵包版、1k 電阻、光敏電阻



<實驗過程與方法>

首先一開始，必須按照下圖 1.1 的電路圖在麵包版上接線，不同於圖上的是，我把 ADC input 的腳位接在 GPA(2)的位置。



▲圖 1.1

看到下圖 1.2，main 函示裡的 222 行中，呼叫寫在 109 行的 GPIO 初始化函示，負責設置 LED 燈泡為輸出。在 225 行中，我另外寫出 ADC_INIT 的 function(第 116 行)對 ADC 進行初始化，在 119 行中，ADC_Open 裡面的 0x1<<2 就是設定要把腳位接到 PA2 的原因，如果想要接在 PA1，則改寫成 0X1<<1 即可。

▼圖 1.2

```
109  void GPIO_Init()
110  {
111      GPIO_SetMode(PC, BIT12, GPIO_PMD_OUTPUT);
112      GPIO_SetMode(PC, BIT13, GPIO_PMD_OUTPUT);
113      GPIO_SetMode(PC, BIT14, GPIO_PMD_OUTPUT);
114      GPIO_SetMode(PC, BIT15, GPIO_PMD_OUTPUT);
115  }
116  void ADC_Init()
117  {
118      /* Set the ADC operation mode as single, input mode as single-end and enable the analog input channel 2 */
119      ADC_Open(ADC, ADC_ADCR_DIFFEN_SINGLE_END, ADC_ADCR_ADMD_SINGLE, 0x1 << 2);
120
121      /* Power on ADC module */
122      ADC_POWER_ON(ADC);
123
124      /* Clear the A/D interrupt flag for safe */
125      ADC_CLR_INT_FLAG(ADC, ADC_ADF_INT);
126
127      /* Enable the ADC interrupt */
128      ADC_EnableInt(ADC, ADC_ADF_INT);
129      NVIC_EnableIRQ(ADC_IRQn);
130  }
```

```
201  int main(void)
202  {
203      /* Unlock protected registers */
204      SYS_UnlockReg();
205
206      /* Init System, IP clock and multi-function I/O */
207      SYS_Init();
208
209      /* Lock protected registers */
210      SYS_LockReg();
211
212      /* Init UART0 for printf */
213      UART0_Init();
214
215      /*---------------------------------------------------------
216      /* SAMPLE CODE
217      /*---------------------------------------------------------
218
219      printf("\nSystem clock rate: %d Hz", SystemCoreClock);
220
221
222      GPIO_Init();
223
224      /*Initial ADC*/
225      ADC_Init();
226
227      /* Single Mode test */
228      AdcSingleModeTest();
229
230      /* Disable ADC module */
231      ADC_Close(ADC);
232
233      /* Disable ADC IP clock */
234      CLK_DisableModuleClock(ADC_MODULE);
235
236      /* Disable External Interrupt */
237      NVIC_DisableIRQ(ADC_IRQn);
238
239      printf("\nExit ADC sample code\n");
240      while(1);
241  }
```

在上圖 1.2 的 228 行中呼叫下圖 1.3 的 AdcSingleModeTest 的函示，在 170 行中，設置了一個變數 g_u32AdcIntFlag=0 來去判斷是否要發生中斷，第 171 行中的 ADC_START_CONV 則是開始訊號的轉換，第 174 行則是判斷 u32AdcIntFlag 是否為 0，如果是 0 的話，則會跑迴圈並且卡進中斷，第 191 行 ADC_IRQHandler 則是發生中斷後會執行的事情，在這裡，他把 g_u32AdcIntFlag 設置為 1，因此會跳出 174 行的迴圈，這裡跟上次 LAB2 的中斷概念很像，是由硬體和作業系統所控制的緣故。

跳出迴圈後，在第 177 行，ADC_GET_CONVERSION_DATA(ADC, 2)函示所轉換完成的數值由 i32ConversionData 去存取，也就是本實驗光敏電阻的測量值。測出結果後，在藉由 178 行印出，完成本實驗的 Basic 的部分。

▼圖 1.3

```
154  void AdcSingleModeTest()
155  {
156      int delay;
157
158      uint8_t  u8Option;
159      int32_t  i32ConversionData;
160      printf("\n");
161      printf("+-----------------------------------------------------------------------+\n");
162      printf("|                       ADC single mode sample code                     |\n");
163      printf("+-----------------------------------------------------------------------+\n");
164
165      while(1)
166      {
167          // printf("  Other keys: exit single mode test\n");
168          //u8Option = getchar();
169              /* Reset the ADC interrupt indicator and Start A/D conversion */
170              g_u32AdcIntFlag = 0;
171              ADC_START_CONV(ADC); //start conversion
172
173              /* Wait ADC interrupt (g_u32AdcIntFlag will be set at IRQ_Handler function)*/
174              while(g_u32AdcIntFlag == 0); //wait to finish the converse
175
176              /* Get the conversion result of the ADC channel 2 */
177              i32ConversionData = ADC_GET_CONVERSION_DATA(ADC, 2);
178              printf("Conversion result of channel 2: %d\n\n", i32ConversionData);
179
180              delay=(100000/i32ConversionData) * 8000 ;
181
182              Control_LED(delay);
183
184      }
185  }
```

```
188  /*---------------------------------------------------------------------------------------------------------*/
189  /* ADC interrupt handler                                                                                   */
190  /*---------------------------------------------------------------------------------------------------------*/
191  void ADC_IRQHandler(void)
192  {
193      g_u32AdcIntFlag = 1;
194      ADC_CLR_INT_FLAG(ADC, ADC_ADF_INT); /* clear the A/D conversion flag */
195  }
196
```

Bonus 控制 LED 的部分，則是藉由上圖 1.2 第 180 行所設計的 delay，呼叫第 131 行(下圖 1.4)的函示並把 delay 的值傳入使用，來藉此達到若光敏電阻 low 則閃爍慢一點，若光敏電阻 high 則閃爍快一點的效果。

▼圖 1.4

```
131  void Control_LED(int j)
132  {
133              PC12=0;
134              CLK_SysTickDelay( j);
135              PC12=1;
136              CLK_SysTickDelay( j);
137
138              PC13=0;
139              CLK_SysTickDelay( j);
140              PC13=1;
141              CLK_SysTickDelay( j);
142
143              PC14=0;
144              CLK_SysTickDelay( j);
145              PC14=1;
146              CLK_SysTickDelay( j);
147
148              PC15=0;
149              CLK_SysTickDelay( j);
150              PC15=1;
151              CLK_SysTickDelay( j);
152  }
153
```

<心得與收穫>

這次實驗相對比較簡單一點，但我在控制 delay 的部分還是花很多時間去調它，有時候會忽快忽慢，有時候又回復正常的控制速度，這使我覺得非常困惑，除此之外，這次的實驗我認為最大的目的就是讓我理解這塊 MCU 是如何把類比形式的連續訊號轉換為數位形式，此外，函示裡面還有許多的位元運算還需要去搞懂它們，期許自己持續進步並多看 datasheet。