



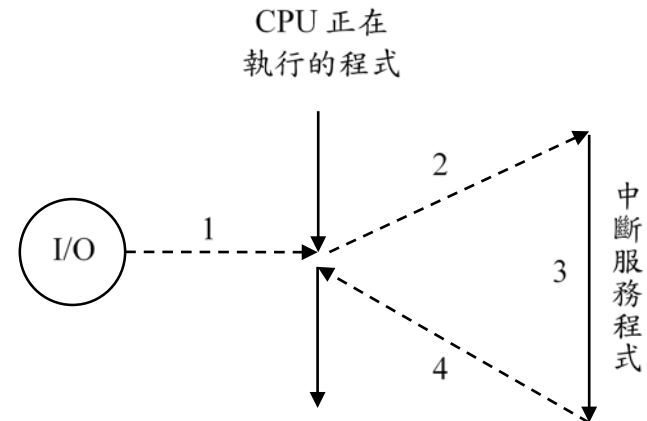
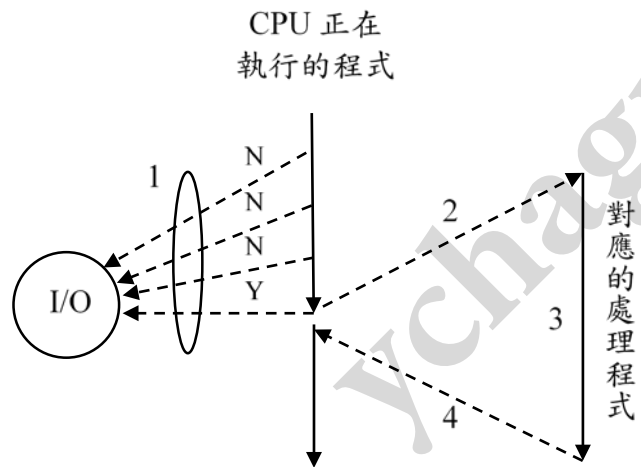
Chap 6

中斷

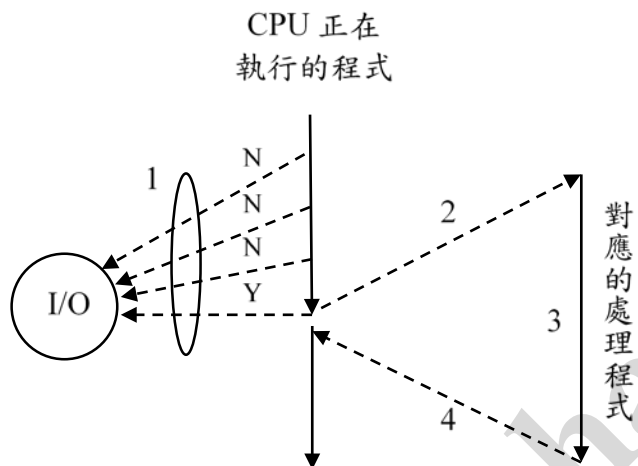
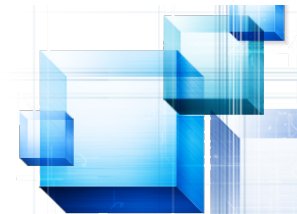
6-1 I/O事件的處理



- 一般而言處理器在執行程式時如果需要與I/O或其他周邊設備進行溝通，通常有二種方法可以使用，分別為輪詢（polling）與中斷（interrupt），這二種方法各有其優缺點，依據應用程式的特性可適用在不同的狀況需求。

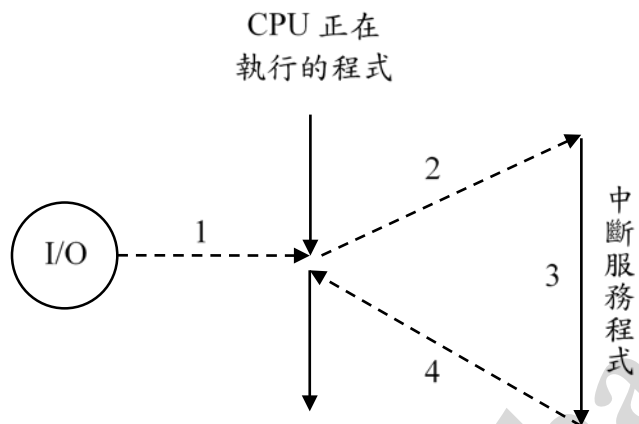


6-1 輪詢 (Polling)



1. 處理器會不斷的詢問（正確的說法應該是檢查）I/O的狀態，可以是連續不斷也可以是間隔一段時間的檢查，直到I/O已完成工作或是滿足某特定的條件。。
2. 一旦I/O的狀態或條件符合，CPU就會立刻暫停現在正在執行的程式或工作，優先執行預先寫好的處理程式。
3. 執行對應的處理程式，一般而言沒有固定的格式，可以是程式片段也可以是副程式，以完成I/O事件的服務。
4. 返回並且繼續執行原來的程式或工作。

6-1 中斷 (Interrupt)



1. I/O已完成工作或是滿足某特定的條件就會引發中斷，通知處理器有中斷事件的發生，這時候處理器是處於被動告知的角色。
2. 處理器在收到中斷訊號後，會先暫停現在正在執行的程式或工作，根據中斷的來源查詢中斷向量表，執行預設的中斷服務程式。
3. 執行對應的中斷服務程式，以完成I/O事件的服務。
4. 返回並且繼續執行原來的程式或工作。

6-2 Arduino UNO的中斷



Arduino UNO 開發板 (ATmega328P) 中總共有 26 個中斷，範圍涵蓋了下列功能：

- I/O 接腳訊號變化的偵測。
- 看門狗定時器 (watchdog timer)。
- 一般定時器的中斷。
- SPI (Serial Peripheral Interface) 串列週邊介面資料傳輸。
- I2C (Inter IC) IC 間資料傳輸。
- USART (Universal Synchronous Asynchronous Receiver Transmitter) 通用同步/非同步收發傳輸器資料傳輸。
- ADC (Analog-to-Digital Converter) 類比數位訊號轉換。
- EEPROM 記憶體存取。
- Flash 記憶體存取。

6-2 Arduino UNO的中斷 1/4

表 6.1 Arduino UNO (ATmega328P) 的中斷

中斷 向量 編號	程式 位址	中斷 ISR(中斷服務程序名稱)	功能描述
1	0x0000	RESET	Power-on Reset and Watchdog System Reset 電源重置與看門狗系統重置
2	0x0002	INT0 (INT0_vect)	External Interrupt Request 0 (pin D2) 使用在 D2 接腳的外部中斷要求 0
3	0x0004	INT1 (INT1_vect)	External Interrupt Request 1 (pin D3) 使用在 D3 接腳的外部中斷要求 1
4	0x0006	PCINT0 (PCINT0_vect)	Pin Change Interrupt Request 0 (pins D8 to D13) 接腳 D8 到 D13 的電壓準位改變中斷要求 0
5	0x0008	PCINT1 (PCINT1_vect)	Pin Change Interrupt Request 1 (pins A0 to A5) 接腳 A0 到 A5 的電壓準位改變中斷要求 1
6	0x000A	PCINT2 (PCINT2_vect)	Pin Change Interrupt Request 2 (pins D0 to D7) 接腳 D0 到 D7 的電壓準位改變中斷要求 2
7	0x000C	WDT (WDT_vect)	Watchdog Time-out Interrupt 看門狗 timeout 中斷

6-2 Arduino UNO的中斷 2/4

8	0x000E	TIMER2 COMPA (TIMER2_COMP_A_vect)	Timer/Counter2 Compare Match A Timer2 定時器的比較相符中斷 A
9	0x0010	TIMER2 COMPB (TIMER2_COMPB_vect)	Timer/Counter2 Compare Match B Timer2 定時器的比較相符中斷 B
10	0x0012	TIMER2 OVF (TIMER2_OVF_vect)	Timer/Counter2 Overflow Timer2 定時器的溢位中斷
11	0x0014	TIMER1 CAPT (TIMER1_CAPT_vect)	Timer/Counter1 Capture Event Timer1 定時器的輸入比較相符中斷
12	0x0016	TIMER1 COMPA (TIMER1_COMP_A_vect)	Timer/Counter1 Compare Match A Timer1 定時器的比較相符中斷 A
13	0x0018	TIMER1 COMPB (TIMER1_COMPB_vect)	Timer/Counter1 Compare Match B Timer1 定時器的比較相符中斷 B
14	0x001A	TIMER1 OVF (TIMER1_OVF_vect)	Timer/Counter1 Overflow Timer1 定時器的溢位中斷
15	0x001C	TIMER0 COMPA (TIMER0_COMP_A_vect)	Timer/Counter0 Compare Match A Timer0 定時器的比較相符中斷 A

6-2 Arduino UNO的中斷 3/4

16	0x001E	TIMER0 COMPB (TIMER0_COMPB_vect)	Timer/Counter0 Compare Match B Timer0 定時器的比較相符中斷 B
17	0x0020	TIMER0 OVF (TIMER0_OVF_vect)	Timer/Counter0 Overflow Timer0 定時器的溢位中斷
18	0x0022	SPI STC (SPI_STC_vect)	SPI Serial Transfer Complete SPI 串列週邊介面資料傳輸完成
19	0x0024	USART RX (USART_RX_vect)	USART Rx Complete USART 接收資料完成
20	0x0026	USART UDRE (USART_UDRE_vect)	USART Data Register Empty USART 資料暫存器是空的，沒有資料
21	0x0028	USART TX (USART_TX_vect)	USART Tx Complete USART 傳送資料完成
22	0x002A	ADC (ADC_vect)	ADC Conversion Complete 類比數位訊號轉換完成

6-2 Arduino UNO的中斷 4/4

23	0x002C	EE READY (EE_READY_vect)	EEPROM Ready EEPROM 記憶體準備就緒
24	0x002E	ANALOG COMP (ANALOG_COMP_vect)	Analog Comparator 類比比較器
25	0x0030	TWI (TWI_vect)	2-wire Serial Interface (I2C) I2C 串列傳輸介面
26	0x0032	SPM READY (SPM_READY_vect)	Store Program Memory Ready 儲存程式記憶體準備就緒
<p>註：(1)中斷向量編號同時也代表著優先權（priority），編號越小優先權越高。</p> <p>(2)中斷服務程序名稱是系統內定用來呼叫該中斷專用的，包含大小寫都要完全一模一樣。</p>			

6-3 中斷服務程序



- 在 Arduino 的程式中，要撰寫中斷服務程序（Interrupt Service Routine，ISR）是一件非常簡單的事。你不需要自己設定中斷向量表，只要使用 ISR(中斷服務程序名稱) 這個巨集指令，就可以將中斷與其中斷服務程序之間的連結自動建立好，例如：。

```
ISR (USART_TX_vect)
{
    //使用者自己寫的中斷處理程式
}
```

6-3 中斷服務程序



● 有別於一般副程式的撰寫，因為中斷的處理是屬於例外的狀況，會暫停一般程式碼的執行，優先執行中斷服務程序，所以在撰寫ISR時要掌握下列幾個重點：

- (1) ISR 程式碼要越短越好，以免中斷時間過長產生非預期的錯誤。
- (2) 切記 ISR 無法使用參數也不會有傳回值。
- (3) 因為 ISR 無法使用參數，所以與一般函式之間必須共用相同的全域變數才能達成資料的傳遞，如果 ISR 執行期間會改變共用全域變數的值，那這些共用的全域變數一定要使用 volatile 修飾子宣告，以避免資料一致性(Data Consistency)的問題，也就是編譯器最佳化所造成資料不一致的狀況。
- (4) 因為在進入 ISR 之後，所有的中斷都會禁止發生（預設），所以在 ISR 內不要使用 Arduino 內建的時間函式 millis()，micro()，delay()，因為這些函式必須依賴定時器中斷才能正常動作。
- (5) 同理，也不要使用串列埠函式 Serial.print()，Serial.read()，Serial.write()...等，因為這些函數必須依賴 USART 中斷才能正常動作。
- (6) 在 ISR 中不要嘗試去啟用/關閉中斷，除非你很有把握不會造成錯誤。

6-4 外部中斷範例

【範例 6.4】Polling

```
1 #define LedPin    13 //Uno 開發板上內建 LED 的接腳固定為 D13
2 #define ButtonPin 2  //指定按鈕開關的接腳為 D2
3 int flag=1;
4
5 void setup() {
6     pinMode(LedPin, OUTPUT);
7     pinMode(ButtonPin, INPUT_PULLUP); //啟用內建的上拉電阻
8     digitalWrite(LedPin, LOW);        //初始 LED 為熄滅的狀態
9 }
10
11 void loop() {
12     if(digitalRead(ButtonPin)==HIGH) flag=1;
13     //D2 的電壓為 LOW，代表按鈕被按下
14     if(digitalRead(ButtonPin)==LOW && flag==1) {
15         digitalWrite(LedPin, !digitalRead(LedPin)); //反轉 LED 燈現在的狀態
16         flag=0;
17     }
18 }
```

6-4 外部中斷範例

【範例 6.5】attachInterrupt

```
1  #define ButtonPin=2;    //設定按鈕的接腳
2  #define LedPin=13;      //Uno 開發板上內建 LED 的接腳固定為 D13
3
4  void setup() {
5      pinMode(LedPin, OUTPUT);
6      pinMode(ButtonPin, INPUT_PULLUP); //enable pull-up resistor
7      digitalWrite(LedPin, LOW);        //初始 LED 為關閉的狀態
8      attachInterrupt(digitalPinToInterrupt(ButtonPin), my_ISR, LOW);
9  }
10
11 void loop() {
12 }
13
14 Void my_ISR() {
15     digitalWrite(LedPin, !digitalRead(LedPin)); //反轉 LED 燈現在的狀態
16 }
```