2025 Watch Dog

學號：01257027姓名：林承羿

1. 觀賞影片, ”十行代码，就能让你理解看门狗”,

https://www.youtube.com/watch?v=ccb13w0ERI4

1. 觀賞影片, ”看門狗電路 Watchdog Timer”

https://www.youtube.com/watch?v=I49CjGfdCTA

1. 如何正確使用Watchdog？

https://www.sonix.com.tw/article-tw-2255-9139

1. 簡要敘述以上的閱讀心得

　　看門狗（Watchdog Timer）是用來偵測程式是否「卡死」或進入無限迴圈等異常狀態的硬體定時器。若在正常執行流程中未定期重設（餵狗），就會觸發系統重啟（reset）以確保裝置能自動復原。

　　從電路圖最後的OR gate可以得知具有三種情況促使瘋狗，具體情況有輸入錯誤的key(0xAA+0x55)、無正確復位(加法器溢位)、WDCR(101)會使瘋狗情況產生。

　　在 SONIX 微控制器中，看門狗（Watchdog）計時器使用晶片內部獨立的低速 RC 振盪器作為時鐘源，其溢出時間受電壓與溫度影響（例：5V 約 256ms，3V 約 512ms），僅供參考。透過 WTCKS、WTRATE、CLKMD 等位元可調整溢出時間，並可透過 Code Option 設定 Watchdog 為 Always\_On、Enable 或 Disable。正確使用方式是將餵狗（清除）操作僅放在主幹道中一次，避免放在副程式、中斷或 Timer 中斷內，以防止系統跑飛時仍持續復位看門狗失去保護效果。可透過系統暫存器 Pflag 中的 NT0 和 NPD 判斷是否為 Watchdog 所觸發的系統復位。

1. 看門狗與本課程哪兩項主題相關? 中斷 與 計時器 .
2. 造成”瘋狗”的原因是 沒有正確執行餵狗，加法器溢位 .
3. 看門狗的復位(=餵狗)應該放在 [程式的主幹道 | 副程式 ], 原因?

　　於主幹道中，保證會被執行到，如果放在副程式，只要沒被呼叫，可能不會正確執行復位，會發生例外的瘋狗情況。

1. time-out(逾時)是[正常|異常]狀況. 看門狗的餵狗是[正常|異常] 動作
2. (期末加分, 建議100字以內)對於本課程提出具體的改善建議, 只要對於以後上課的教師與同學有實質的幫助, 都歡迎

　　在上課的部分，我對於老師積極回答同學提出問題的行為給予極高評價。但我比較有問題的部分為教學方式，可能是台灣從小體制上的問題，現在本人認為還不是一個可以完全放手自學的時機，直白地說，一直發問問題是否也在某種程度上代表了引導的不夠多？我很清楚這問題是很主觀的，或許通靈能力極高、天資聰穎者不需要老師的提點即可無師自通，但非常不幸，我不是。

　　在考試上，尤其是第一次的期中考，我對於手寫程式碼有點小小意見，雖然考試本身就是為了鑑別高低，但我想反映的是時間問題。對於學習速度本身沒那麼快的人來說，或許可以給予更多的作答時間，畢竟程式不是一個好寫的東西。

　　回到上課步調問題，我個人認為老師在課堂上帶的節奏多快不是問題，但是否可以提供之前上課的錄影黨給同學，老實說，一個將近40葉的PDF不是一個簡簡單單幾分鐘講解就可以解決的，我相信老師也深知如此所以才會有學習單的作業。可在提供可重複觀看的講解後，是否可以花更少的時間獲得更高的學習效率。

　　　　　　　　　　以上，感謝老師這學期的教導 by 01257027 林承羿

|  |  |  |
| --- | --- | --- |
| C program | 實際Optimized asm | 預期Unoptimized asm |
| unsigned char x=0;  void fn1(void){  x++;  fn2();  }  void fn2(void){  x++  } | fn1:  MOV DPTR, #x  MOV A, @DPTR  INC A  MOV @DPTR, A  LJMP fn2  fn2:  MOV DPTR, #x  MOV A, @DPTR  INC A  MOV @DPTR, A  RET | fn1:  MOV DPTR, #x  MOV A, @DPTR  INC A  MOV @DPTR, A  LCALL fn2  RET  fn2:  MOV DPTR, #x  MOV A, @DPTR  INC A  MOV @DPTR, A  RET |

發生的條件是某函數的最後一個指令為呼叫另一個函數, 預期為LCALL (3bytes, 2cycles)+ RET (1byte, 2cycles), 但是實際卻是一個LJMP(3bytes, 2cycles), 節省空間(1 byte), 也節省時間(2cycles)