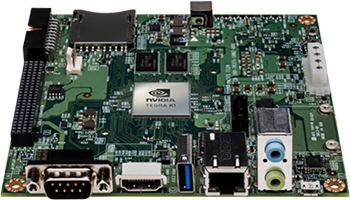
微算機系統 - 實驗六

實作 CHAR DEVICE DRIVER

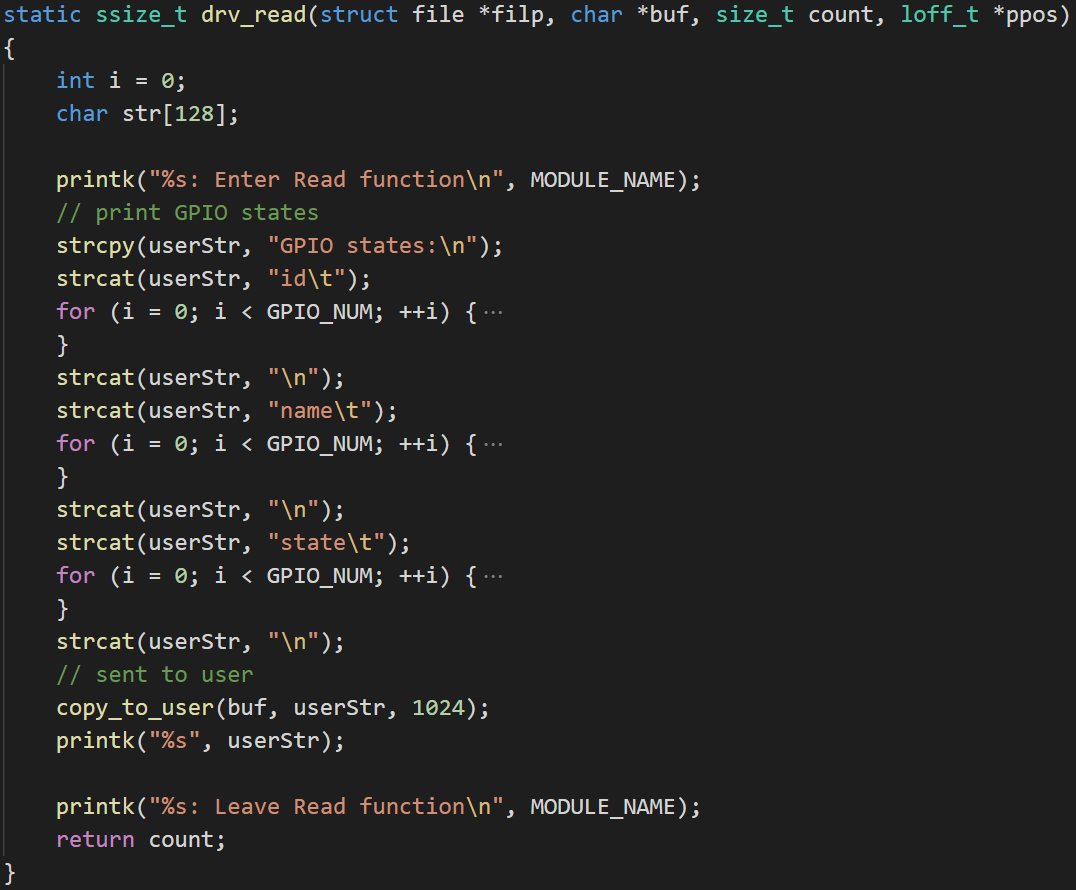
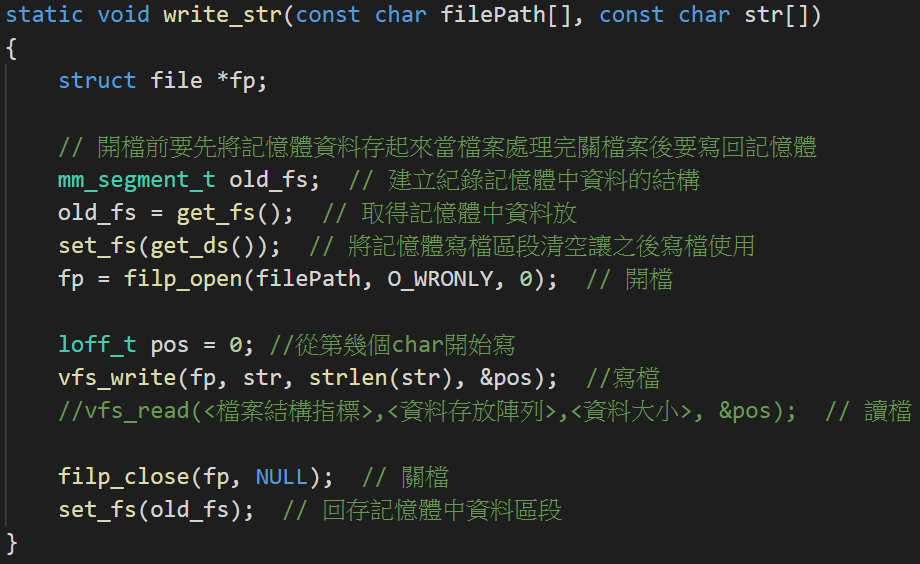
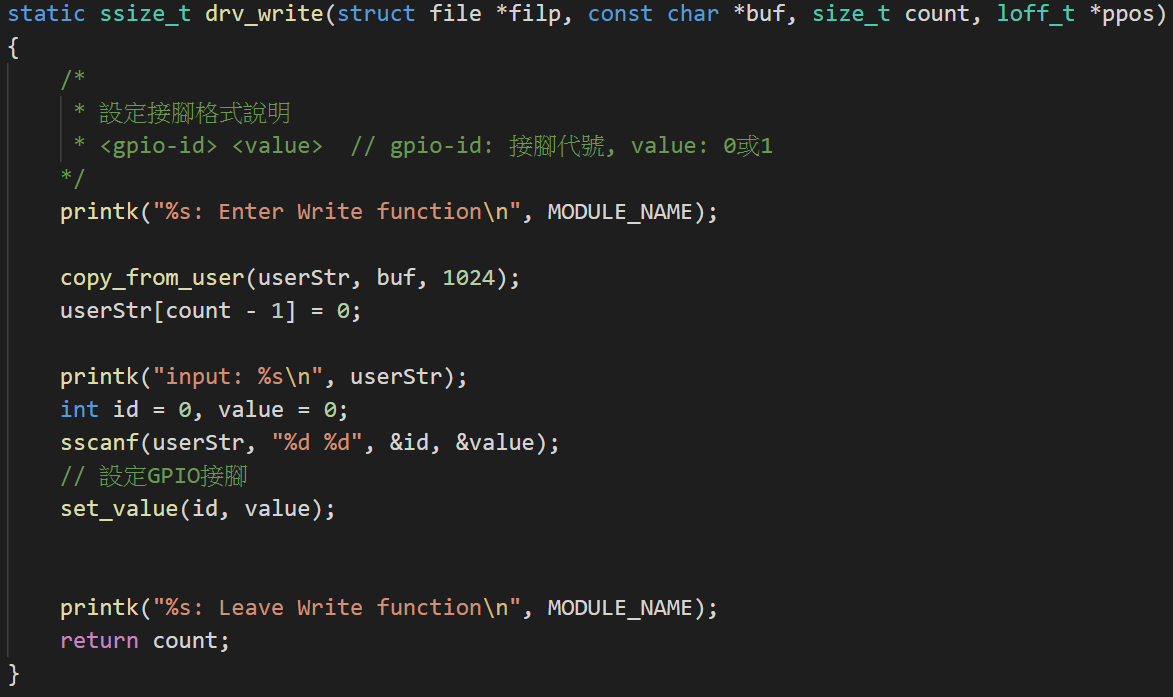
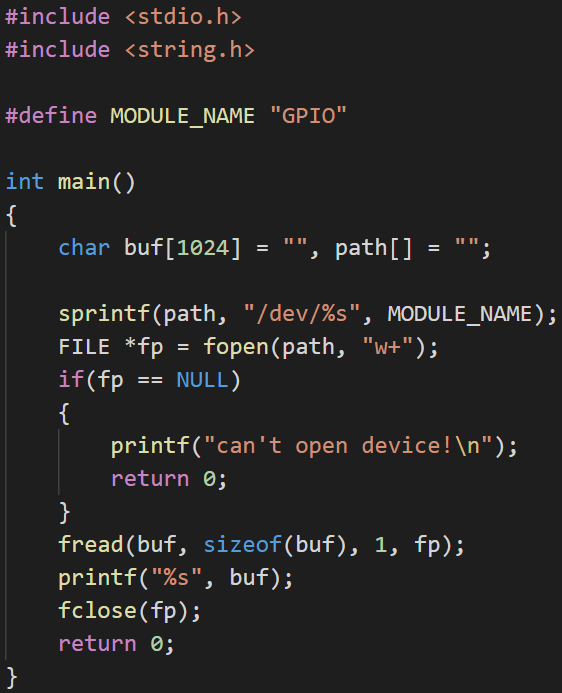
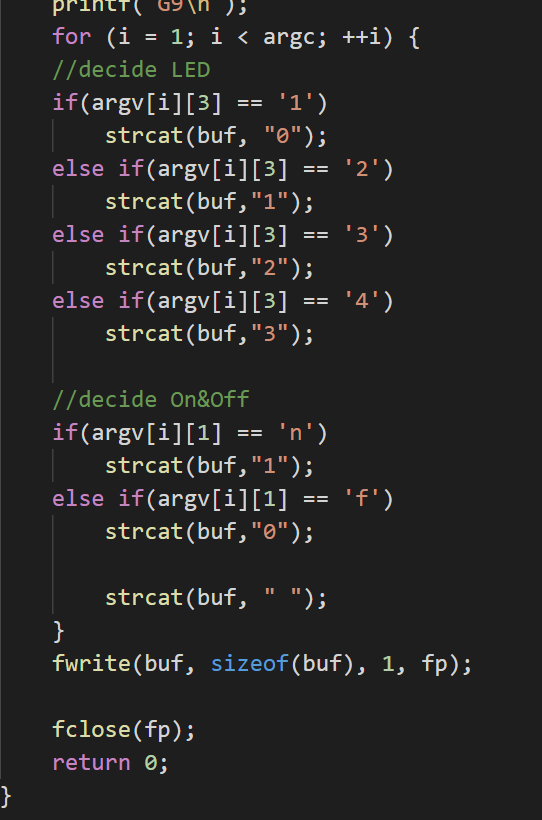
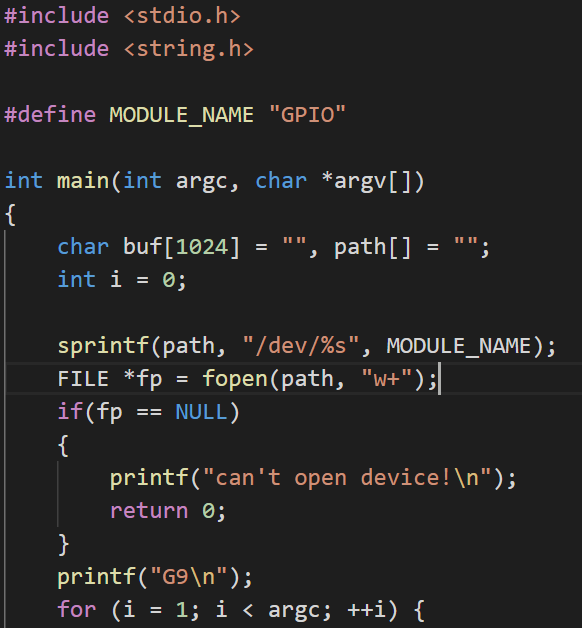


組別：第九組

組員名稱：  
資工二 106590034 吳陽生  
資工二 106590036 梁博瑜

日期： 2019-04-28

# 實驗過程

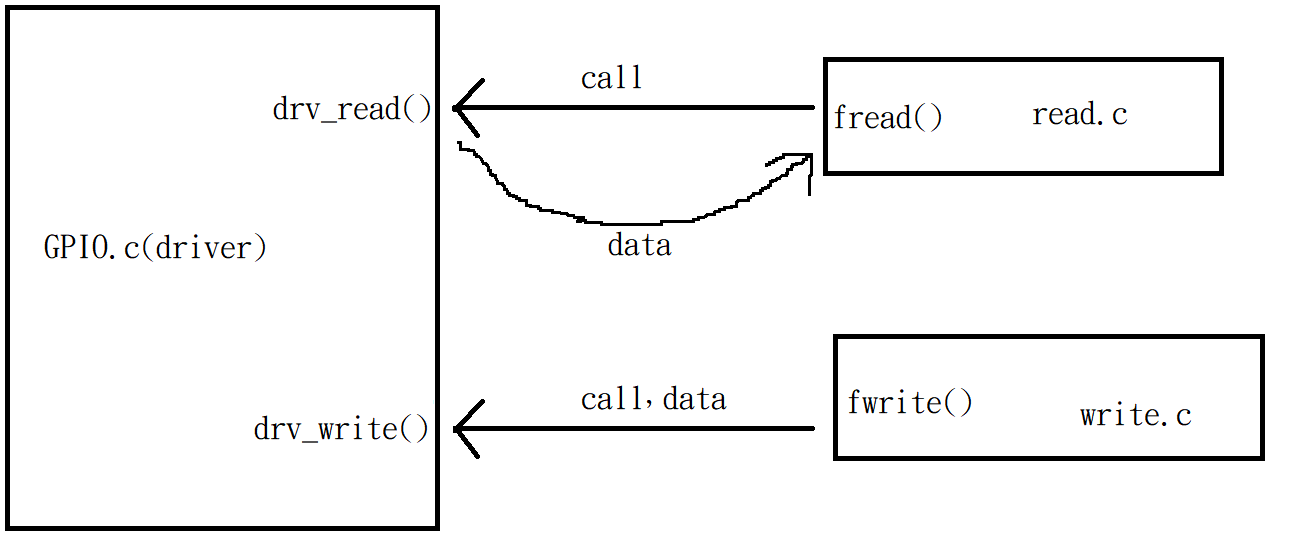
1. 撰寫Char Driver程式碼
   * Driver狀態輸出  
     
   * 寫入字串  
     
   * driver寫入函式  
     
2. 在/dev目錄下建立node檔
3. 掛載驅動GPIO
4. 撰寫對應的寫入及讀取程式  
    
5. 透過驅動控制TK1的GPIO接腳

# 組員貢獻比例

吳陽生 49.9999999999999999999999999999999999999999999999999999999999999999999%

梁博瑜 50.0000000000000000000000000000000000000000000000000000000000000000001%

# 心得

吳陽生:  
 這次實驗有兩個地方比較重要, 一者是要了解所有程式的之間運作, 二者是要學習如何直接利用Linux核心所給的方法來進行寫檔和讀檔  
 以這次實驗來說, 我們有仨檔案分別是: GPIO.c, read.c, write.c, 而GPIO.c才是整個實驗的主角, read.c, write.c則是提供給使用者的介面, 透過read.c, write.c操作GPIO.c, 整個流程是: 使用者執行read並透過fread函式呼叫GPIO的drv\_read, GPIO再通過copy\_to\_user把字串傳回給read; 而wrie也是類似的流程, 不同是fread改成fwrite, copy\_to\_user改成copy\_from\_user, 若以圖片來表示會像是這樣:  
   
 而讀檔的部分則須照著助教給的範例來實作, 但是其中” 開檔前要先將記憶體資料存起來當檔案處理完關檔案後要寫回記憶體”我還是不太明白為什麼要這樣做  
 實驗在過程中有碰到很多問題, 雖然我已經先在家裡把大部分的程式碼都寫好, 因為沒有板子可以測試GPIO, 拿到學校實作的時候還是出現很多錯誤, 比較麻煩的是寫檔有錯誤他並不會直接指出哪裡有誤, 但是結果就是不對, 這時候就只能一點一點地排除錯誤, 最後發現是忘記把int轉成char\*, 好在最後還是解決了  
 為了方便以後的使用, 整理一下安裝Driver的流程(假設程式碼都已經寫好了):

1. 查看kernal版本  
   uname -a
2. 如為首次使用該kernal編譯則需輸入(TK1)  
   sudo make modules\_prepare
3. 建立裝置檔案  
   mknod /dev/moduleName c 60 0
4. Makefile中修改obj-m, kernal\_DIR並執行make
5. insmod 掛載, 用lsmod | grep hellod或dmesg檢查是否掛載成功

梁博瑜:

這一次的實驗花了很多時間才搞懂實際上驅動、程式與板子之間是如何運作的，而且在最後面要進行Debug的時候，就會發現Debug很麻煩，要來回地進行修改，然後將修改過後的檔案傳到板子上，然後在編譯，最後才能執行，最後我們直接先連進去板子，直接修改版子中的檔案，就可以省去傳檔案這個費時的步驟，幸運的是，因為GPIO基本上沒有任何地方出錯，所以只要修改Read跟Write這兩個檔案就好了，不然每次修一個Bug就要重新安裝一次我想一定會改死人，另外，不能把板子帶回家做實驗果然還是頗麻煩的，禮拜一沒做完的部分，回家處裡也無法知道有沒有錯誤，只能隔天再做確認，學校的電腦又時不時掛掉，而且直到現在還是沒有搞清楚連線到板子上的時候，要先在控制台做設定，但是完全不能理解每一個步驟所代表的意義，只能硬背起來，帶了筆電跟網路線來，想說用自己的電腦做也許比較快，卻不知道如何設定連線的部分，最後只好作罷。