111-1 535313

進階可程式邏輯系統設計與應用

Advanced Programmable Logic System Design and Application

實驗編號 : LAB08

實驗名稱 : HPS FPGA and Custom QSYS Component

結報完成日期 : 2022.12.05

姓名 : 王語

系級 : 機械四

學號 : 0811127

1. 實驗目的

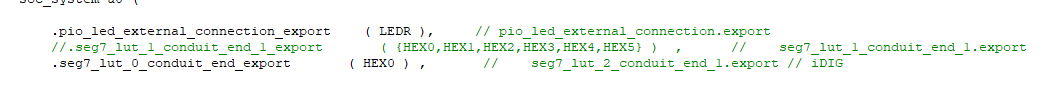
本實驗藉由Qsys連接DE1-SoC上的ARM以及FPGA，以熟悉基於SoC 的 FPGA 嵌入式系統硬體與軟體開發流程。

說明[1] : HPS是基於ARM的處理器，具有豐富的周邊電路和存儲介面。HPS和 FPGA不僅能夠獨立工作，也能通過高性能 AXI 總線橋接實現高速寬頻行數據傳輸，此部分配置可以藉由Quartus II 中的Custom QSYS完成開發，這個總線是雙向的且支持同時讀寫的功能。

ARM部分以 C語言開發，而FPGA部分則支持Verilog。

1. Verilog 程式碼

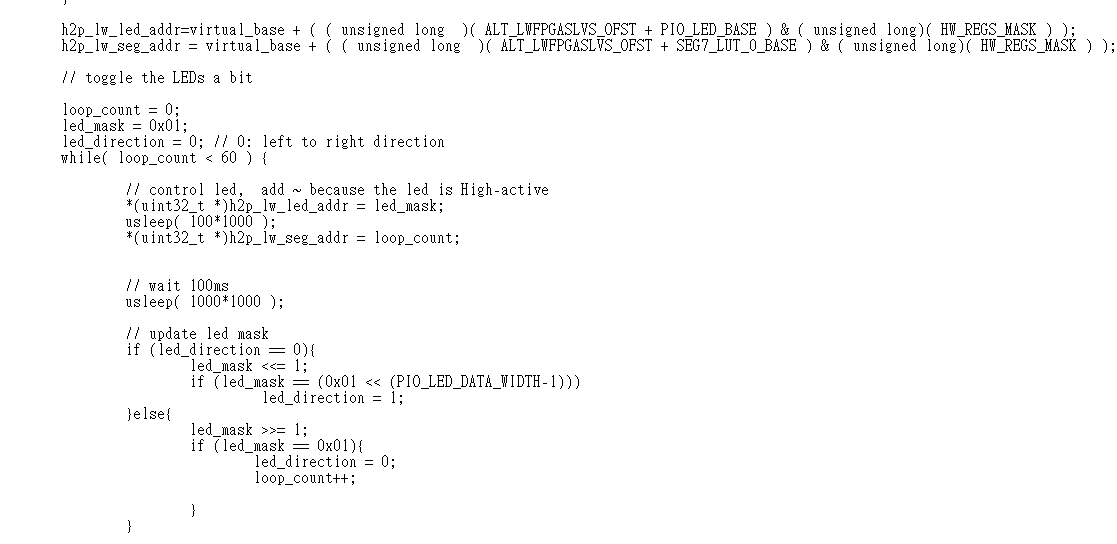
為了節省空間，只貼由我撰寫的兩行程式碼。



另外，E3下載的v檔案中有部分wire沒有宣告，已將其更正。

1. C 程式碼

由於執行過程七節管沒有正常工作，我將助教給的程式碼做了小更動，移除了兩行註解。



1. hps\_0

#ifndef \_ALTERA\_HPS\_0\_H\_

#define \_ALTERA\_HPS\_0\_H\_

/\*

\* This file was automatically generated by the swinfo2header utility.

\*

\* Created from SOPC Builder system 'soc\_system' in

\* file './soc\_system.sopcinfo'.

\*/

/\*

\* This file contains macros for module 'hps\_0' and devices

\* connected to the following masters:

\* h2f\_axi\_master

\* h2f\_lw\_axi\_master

\*

\* Do not include this header file and another header file created for a

\* different module or master group at the same time.

\* Doing so may result in duplicate macro names.

\* Instead, use the system header file which has macros with unique names.

\*/

/\*

\* Macros for device 'onchip\_memory2\_0', class 'altera\_avalon\_onchip\_memory2'

\* The macros are prefixed with 'ONCHIP\_MEMORY2\_0\_'.

\* The prefix is the slave descriptor.

\*/

#define ONCHIP\_MEMORY2\_0\_COMPONENT\_TYPE altera\_avalon\_onchip\_memory2

#define ONCHIP\_MEMORY2\_0\_COMPONENT\_NAME onchip\_memory2\_0

#define ONCHIP\_MEMORY2\_0\_BASE 0x0

#define ONCHIP\_MEMORY2\_0\_SPAN 65536

#define ONCHIP\_MEMORY2\_0\_END 0xffff

#define ONCHIP\_MEMORY2\_0\_ALLOW\_IN\_SYSTEM\_MEMORY\_CONTENT\_EDITOR 0

#define ONCHIP\_MEMORY2\_0\_ALLOW\_MRAM\_SIM\_CONTENTS\_ONLY\_FILE 0

#define ONCHIP\_MEMORY2\_0\_CONTENTS\_INFO ""

#define ONCHIP\_MEMORY2\_0\_DUAL\_PORT 0

#define ONCHIP\_MEMORY2\_0\_GUI\_RAM\_BLOCK\_TYPE AUTO

#define ONCHIP\_MEMORY2\_0\_INIT\_CONTENTS\_FILE soc\_system\_onchip\_memory2\_0

#define ONCHIP\_MEMORY2\_0\_INIT\_MEM\_CONTENT 1

#define ONCHIP\_MEMORY2\_0\_INSTANCE\_ID NONE

#define ONCHIP\_MEMORY2\_0\_NON\_DEFAULT\_INIT\_FILE\_ENABLED 0

#define ONCHIP\_MEMORY2\_0\_RAM\_BLOCK\_TYPE AUTO

#define ONCHIP\_MEMORY2\_0\_READ\_DURING\_WRITE\_MODE DONT\_CARE

#define ONCHIP\_MEMORY2\_0\_SINGLE\_CLOCK\_OP 0

#define ONCHIP\_MEMORY2\_0\_SIZE\_MULTIPLE 1

#define ONCHIP\_MEMORY2\_0\_SIZE\_VALUE 65536

#define ONCHIP\_MEMORY2\_0\_WRITABLE 1

#define ONCHIP\_MEMORY2\_0\_MEMORY\_INFO\_DAT\_SYM\_INSTALL\_DIR SIM\_DIR

#define ONCHIP\_MEMORY2\_0\_MEMORY\_INFO\_GENERATE\_DAT\_SYM 1

#define ONCHIP\_MEMORY2\_0\_MEMORY\_INFO\_GENERATE\_HEX 1

#define ONCHIP\_MEMORY2\_0\_MEMORY\_INFO\_HAS\_BYTE\_LANE 0

#define ONCHIP\_MEMORY2\_0\_MEMORY\_INFO\_HEX\_INSTALL\_DIR QPF\_DIR

#define ONCHIP\_MEMORY2\_0\_MEMORY\_INFO\_MEM\_INIT\_DATA\_WIDTH 64

#define ONCHIP\_MEMORY2\_0\_MEMORY\_INFO\_MEM\_INIT\_FILENAME soc\_system\_onchip\_memory2\_0

/\*

\* Macros for device 'pio\_led', class 'altera\_avalon\_pio'

\* The macros are prefixed with 'PIO\_LED\_'.

\* The prefix is the slave descriptor.

\*/

#define PIO\_LED\_COMPONENT\_TYPE altera\_avalon\_pio

#define PIO\_LED\_COMPONENT\_NAME pio\_led

#define PIO\_LED\_BASE 0x0

#define PIO\_LED\_SPAN 32

#define PIO\_LED\_END 0x1f

#define PIO\_LED\_BIT\_CLEARING\_EDGE\_REGISTER 0

#define PIO\_LED\_BIT\_MODIFYING\_OUTPUT\_REGISTER 0

#define PIO\_LED\_CAPTURE 0

#define PIO\_LED\_DATA\_WIDTH 10

#define PIO\_LED\_DO\_TEST\_BENCH\_WIRING 0

#define PIO\_LED\_DRIVEN\_SIM\_VALUE 0

#define PIO\_LED\_EDGE\_TYPE NONE

#define PIO\_LED\_FREQ 50000000

#define PIO\_LED\_HAS\_IN 0

#define PIO\_LED\_HAS\_OUT 1

#define PIO\_LED\_HAS\_TRI 0

#define PIO\_LED\_IRQ\_TYPE NONE

#define PIO\_LED\_RESET\_VALUE 1023

/\*

\* Macros for device 'SEG7\_LUT\_0', class 'SEG7\_LUT'

\* The macros are prefixed with 'SEG7\_LUT\_0\_'.

\* The prefix is the slave descriptor.

\*/

#define SEG7\_LUT\_0\_COMPONENT\_TYPE SEG7\_LUT

#define SEG7\_LUT\_0\_COMPONENT\_NAME SEG7\_LUT\_0

#define SEG7\_LUT\_0\_BASE 0x20

#define SEG7\_LUT\_0\_SPAN 4

#define SEG7\_LUT\_0\_END 0x23

/\*

\* Macros for device 'sysid\_qsys', class 'altera\_avalon\_sysid\_qsys'

\* The macros are prefixed with 'SYSID\_QSYS\_'.

\* The prefix is the slave descriptor.

\*/

#define SYSID\_QSYS\_COMPONENT\_TYPE altera\_avalon\_sysid\_qsys

#define SYSID\_QSYS\_COMPONENT\_NAME sysid\_qsys

#define SYSID\_QSYS\_BASE 0x10000

#define SYSID\_QSYS\_SPAN 8

#define SYSID\_QSYS\_END 0x10007

#define SYSID\_QSYS\_ID 2899645186

#define SYSID\_QSYS\_TIMESTAMP 1670153700

/\*

\* Macros for device 'jtag\_uart', class 'altera\_avalon\_jtag\_uart'

\* The macros are prefixed with 'JTAG\_UART\_'.

\* The prefix is the slave descriptor.

\*/

#define JTAG\_UART\_COMPONENT\_TYPE altera\_avalon\_jtag\_uart

#define JTAG\_UART\_COMPONENT\_NAME jtag\_uart

#define JTAG\_UART\_BASE 0x20000

#define JTAG\_UART\_SPAN 16

#define JTAG\_UART\_END 0x2000f

#define JTAG\_UART\_IRQ 2

#define JTAG\_UART\_READ\_DEPTH 64

#define JTAG\_UART\_READ\_THRESHOLD 8

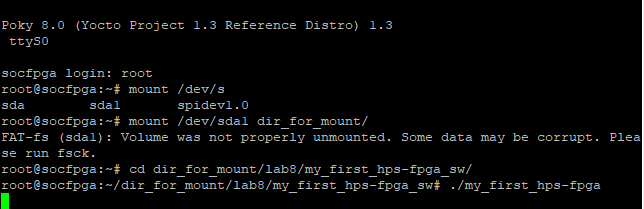
#define JTAG\_UART\_WRITE\_DEPTH 64

#define JTAG\_UART\_WRITE\_THRESHOLD 8

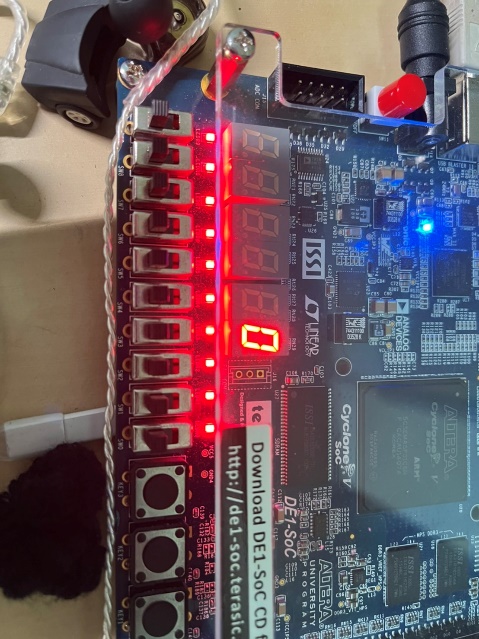
#endif /\* \_ALTERA\_HPS\_0\_H\_ \*/

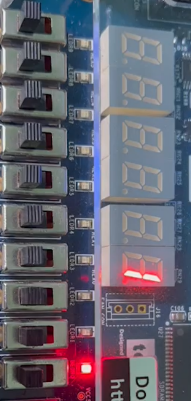
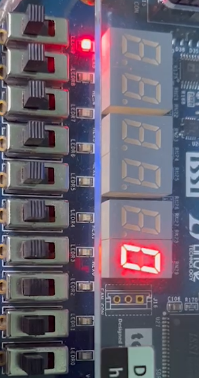
1. 實驗結果

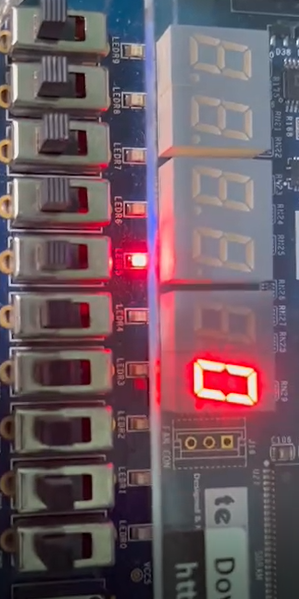
Putty執行



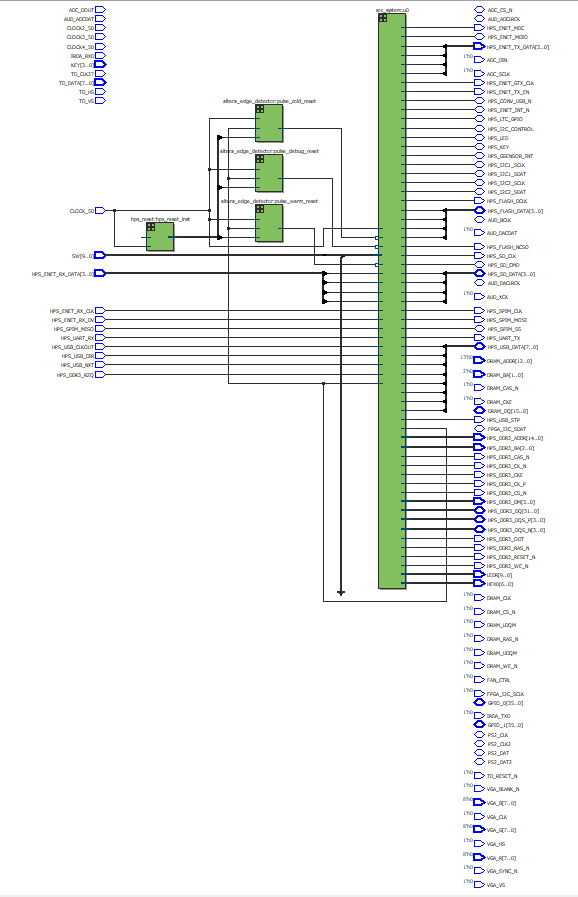
Sof檔案燒錄完成



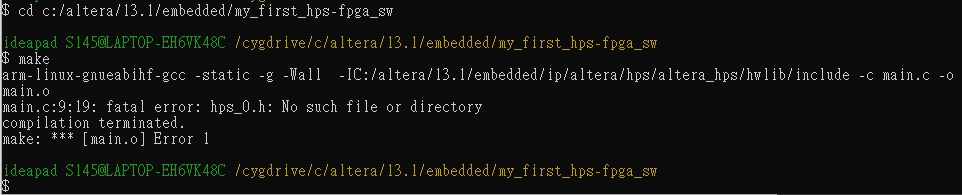
運行結果如同DEMO影片，執行跑馬燈，LED燈從最右邊開始閃爍至最左邊，再由最左邊閃爍回到最右邊，當最右邊的LED燈再次亮起時，七節管的數字+1。



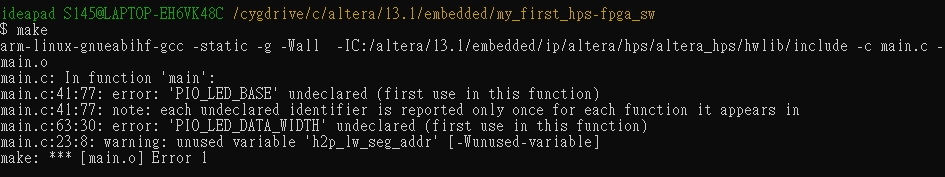
1. RTL

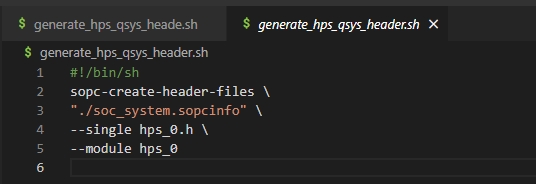


1. 問題與討論
2. 照著實驗講義之步驟操作，卻出現缺少hps\_0的錯誤

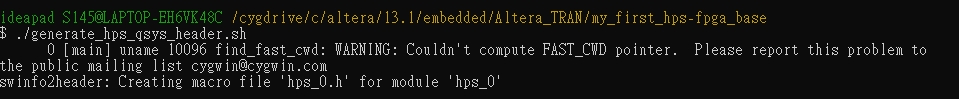


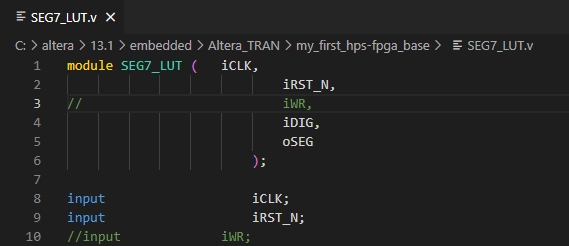
而且生成出來的hps\_0似乎也存在錯誤

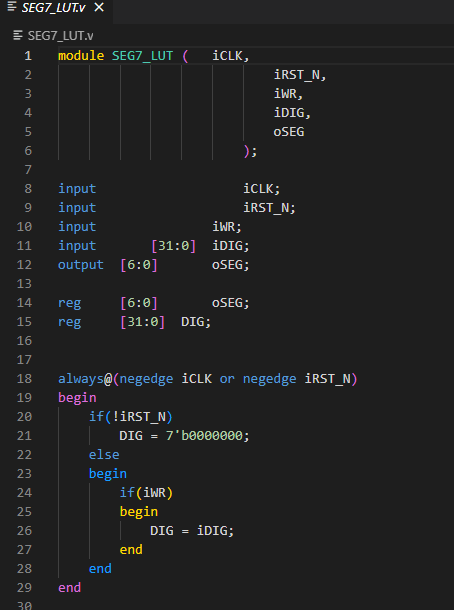


為了解決此問題，自行創建一個新的generate\_hps\_qsys\_header.sh檔，方法如同參考資料[2] 

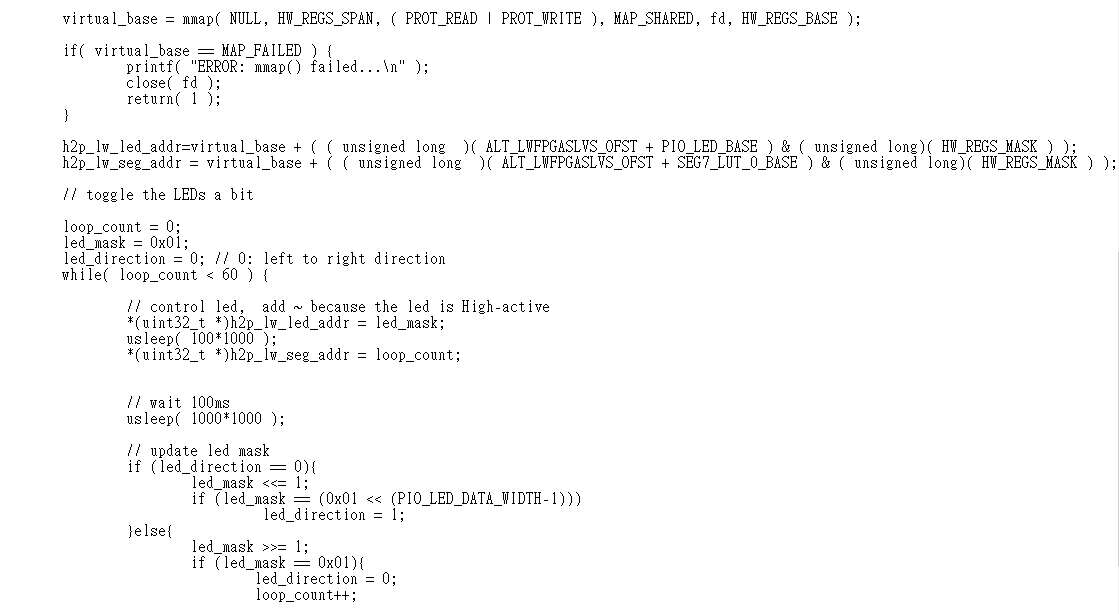
並以新生成的檔案取代舊版的即可解決問題。



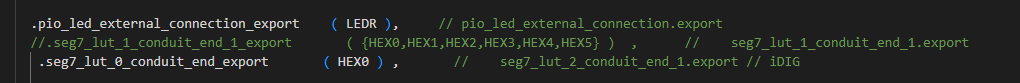
1. 在Qsys中創建SEG7\_LUT時，因為缺少了write\_en，因此無法順利生成，將SEG7\_LUT.v中的註解取消即可順利運行。



1. 燒錄sof檔並在putty上運行，不過LED跑馬燈跑完一輪後，七節管上的數字並沒有增加，花費數個小時，反覆檢查Qsys走線，以及SEG7\_LUT.v是否存在錯誤，結果發現是C檔案中七節管數字的程式碼備註解掉了，不知道是不是助教故意設下的陷阱，但有一點考以很確定，這個程式的設計者是個狠人，因為使用記事本，程式碼沒有顏色，真的很難發現。



1. 在講義中並沒有說明.seg7\_lut\_0\_conduit\_end\_export要接到哪裡，經過一番研究後發現要接在HEX0上。



1. 若要使結果和DEMO影片一致，則不可以用4’b0000作為HEX1~HEX5的輸入，即便沒有給予訊號也不可使他默認接地，否則七節管會輸出8。為了解決此問題，我直接將Output的HEX1~HEX5移除。
2. 心得

這次實驗好困難，做的我心好累，不是不會打，是不知道錯在哪，對於嵌入式我一向很害怕，如果只是要打程式，我很會抓BUG，不過這種系統類、架構相關的問題，在加上對於開發軟體的不熟悉，讓我挫敗連連。

心好累，希望期末專題可以順順利利。

1. 參考資料

[1] HPS基本概念及其設計 - noticeable - 博客園 (cnblogs.com)

https://www.cnblogs.com/noticeable/p/9378394.html

[2] 【友晶科技Terasic】用sopc-create-header-files工具生成FPGA..

https://www.cnblogs.com/DoreenLiu/p/15308574.html