微算機系統

實驗六

組別： 14
班級、姓名與學號：四資二　洪晟毅　104590048
四資二　梁皓鈞　104360098

日期： 2016.12.19

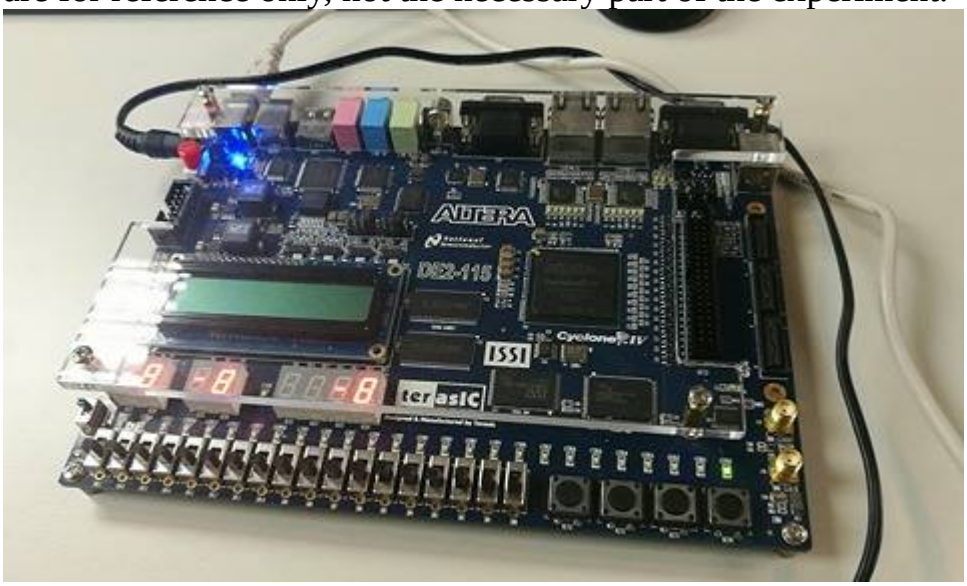
1. 實驗內容：

   This experiment is about the swap function of the three registers. It includes the bus line, three registers, and several tristate buffers. Those tristate buffers is for controlling the flow of data. So that those registers won't crash and should be working properly.

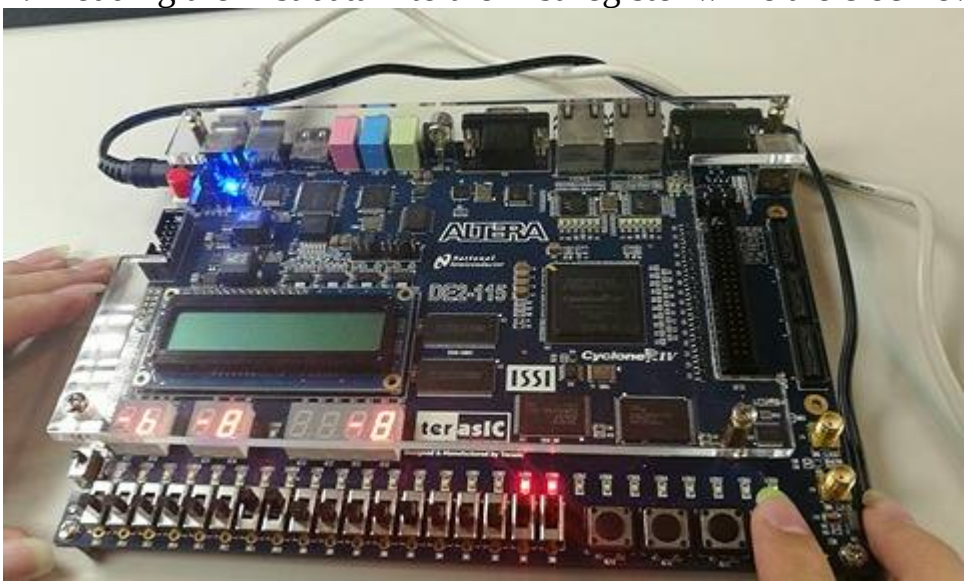
2. 實驗過程及結果：

   實驗記錄：

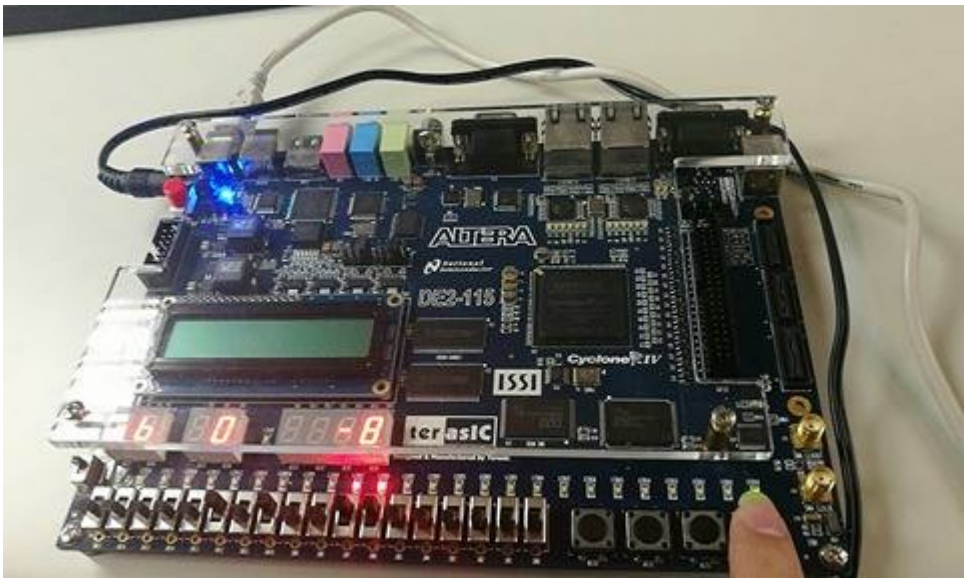
   This is the initial state of the experiment. To be reminded that the seven-segments are for reference only, not the necessary part of the experiment.



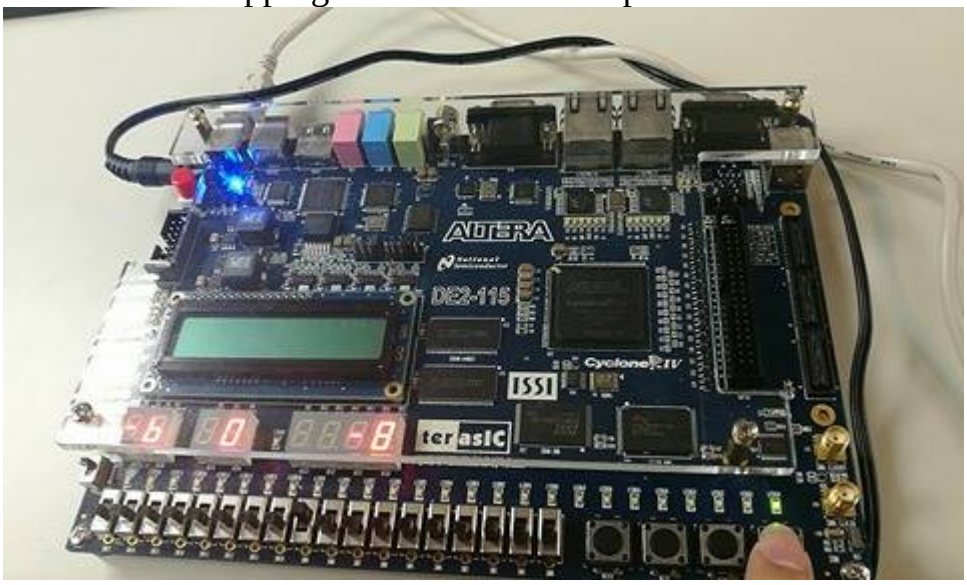
1. Loading the first data into the first register while the clock'event and clock = '1'



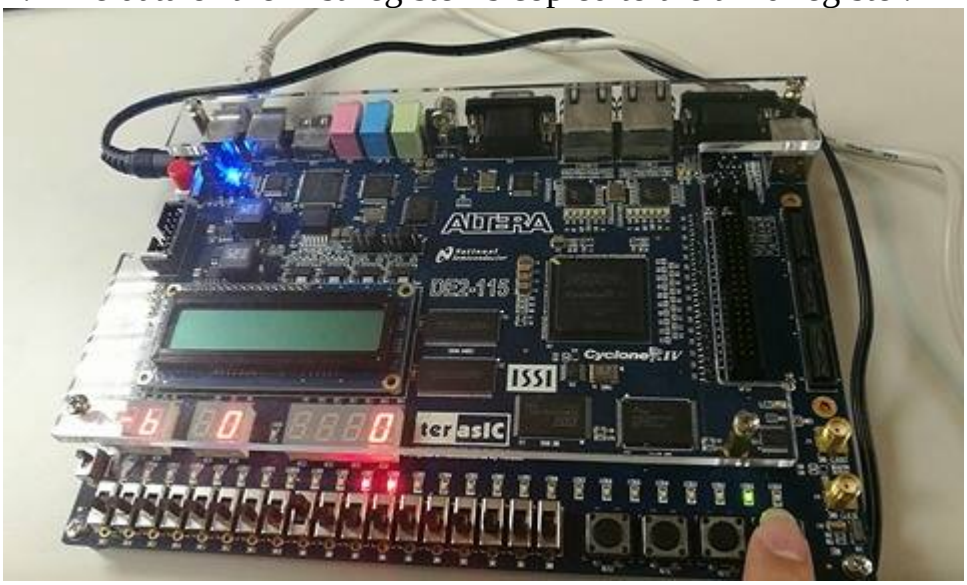
2. Loading the second data into the second register while clock'event and clock = '1'



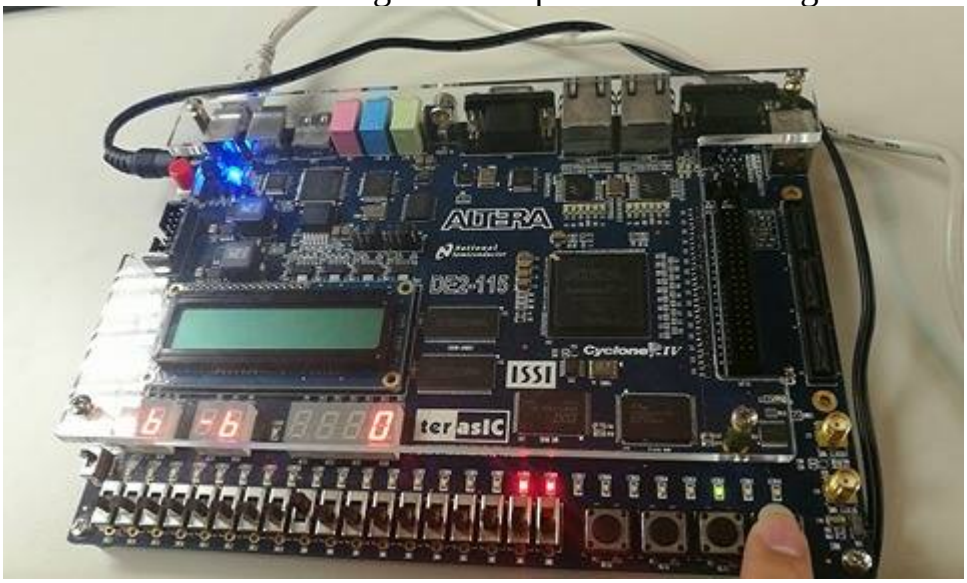
3. Start the swapping function while swap = '1' and clock'event and clock = '1'



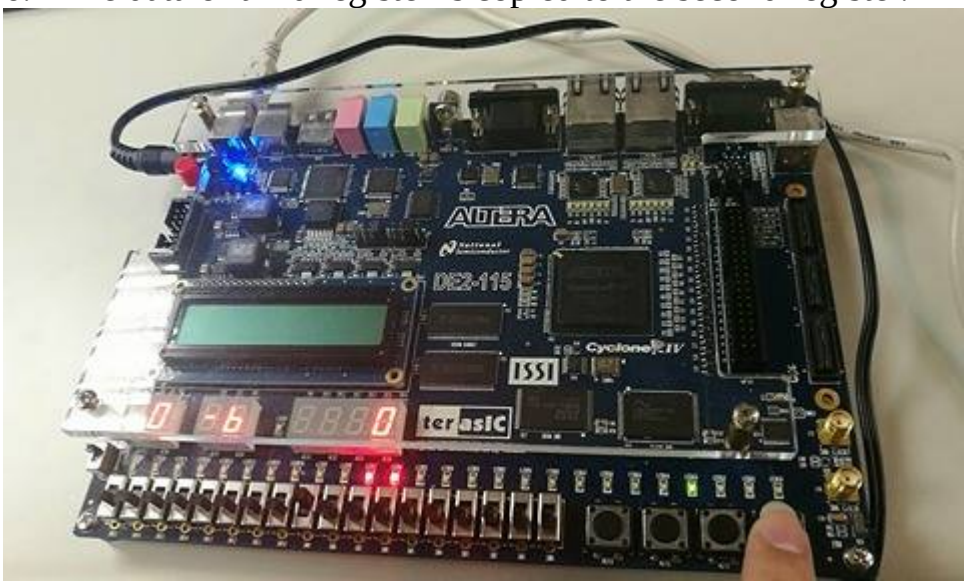
4. The data of the first register is copied to the third register.



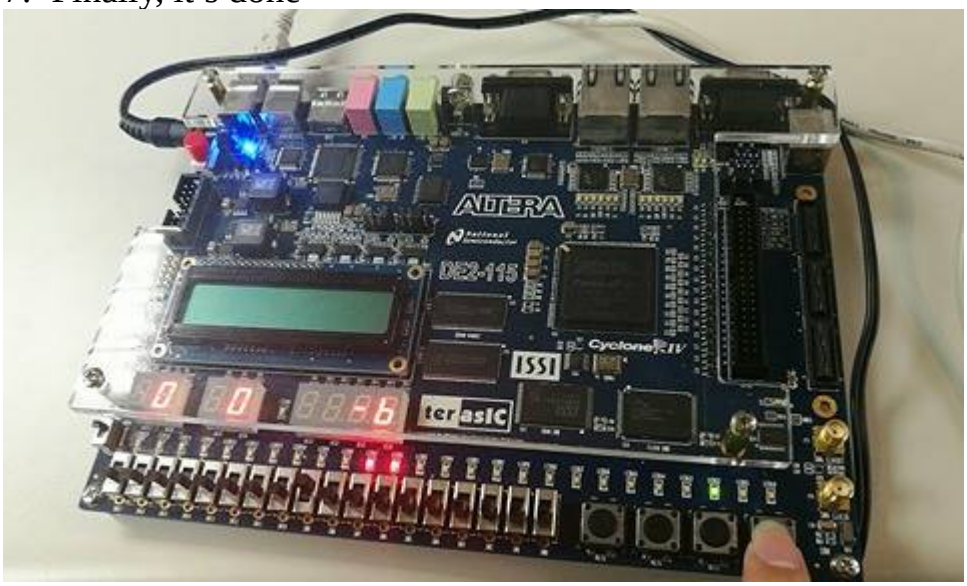
5. The data of second register is copied to the first register



6. The data of third register is copied to the second register.



7. Finally, it’s done



程式碼解釋：

右圖為三態緩衝器的component的程式碼，可以看出若E等於1時，會將X輸出至F，否則就輸出高阻抗(Z)給F。

```vhdl
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE WORK.components.trin;

ENTITY trin IS
    GENERIC (
        N : INTEGER := 8
    );
    PORT (
        X : IN STD_LOGIC_VECTOR(N-1 DOWNTO 0);
        E : IN STD_LOGIC;
        F : OUT STD_LOGIC_VECTOR(N-1 DOWNTO 0)
    );
END trin;

ARCHITECTURE Behavior OF trin IS
BEGIN
    PROCESS(E, X)
    BEGIN
        IF E = '1' THEN
            F <= X;
        else
            FOR i IN N-1 DOWNTO 0 LOOP
                F(i) <= 'Z';
            END LOOP;
        END IF;
    END PROCESS;
END Behavior;
```

```vhdl
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE WORK.components.regn;

ENTITY regn IS
    GENERIC (
        N : INTEGER := 8
    );
    PORT (
        R : IN STD_LOGIC_VECTOR(N-1 DOWNTO 0);
        Rin, Clock : IN STD_LOGIC;
        Q : OUT STD_LOGIC_VECTOR(N-1 DOWNTO 0)
    );
END regn;

ARCHITECTURE Behavior OF regn IS
BEGIN
    PROCESS
    BEGIN
        WAIT UNTIL Clock'event AND Clock = '1';
        IF Rin = '1' THEN
            Q <= R;
        END IF;
    END PROCESS;
END Behavior;
```

左圖為控制輸入暫存器的components，只有在Clock正緣觸發且Rin等於1時才會觸發將R寫入Q之中。

下圖為主要電路的 ENTITY 的部分，我們為了方便檢查結果，額外新增了 Show_R1~R3的 PORT，將暫存器 R1~R3的值顯示於七段顯示器上。

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE WORK.components.ALL;

ENTITY Lab6 IS
    PORT (
        Data                : IN STD_LOGIC_VECTOR(7 DOWNTO 0);
        Resetn, w           : IN STD_LOGIC;
        Clock, Extern       : IN STD_LOGIC;
        RinExt              : IN STD_LOGIC_VECTOR(1 TO 3);
        BusWires            : INOUT STD_LOGIC_VECTOR(7 DOWNTO 0);
        StateA              : OUT STD_LOGIC;
        StateB              : OUT STD_LOGIC;
        StateC              : OUT STD_LOGIC;
        StateD              : OUT STD_LOGIC;
        Show_R1             : OUT STD_LOGIC_VECTOR(7 DOWNTO 0);
        Show_R2             : OUT STD_LOGIC_VECTOR(7 DOWNTO 0);
        Show_R3             : OUT STD_LOGIC_VECTOR(7 DOWNTO 0)
    );
END Lab6;
```

內部含有作為狀態機切換狀態的 state、作為暫存器 R1~R3的 SIGNAL 訊號。

```
ARCHITECTURE Behavior OF Lab6 IS
    TYPE state_type IS (A, B, C, D);
    SIGNAL state : state_type := A;

    SIGNAL Rin, Rout, Q : STD_LOGIC_VECTOR(1 TO 3);
    SIGNAL R1, R2, R3       : STD_LOGIC_VECTOR(7 DOWNTO 0);
```

右圖為負責處理狀態機的狀態轉換的 PROCESS。

```
    PROCESS
    BEGIN
        WAIT UNTIL Clock'EVENT AND Clock = '0';
        IF Resetn = '1' THEN
            state <= A;
        ELSE
            CASE state IS
                WHEN A =>
                    IF w='1' THEN
                        state <= B;
                    END IF;
                WHEN B =>
                    state <= C;
                WHEN C =>
                    state <= D;
                WHEN D =>
                    state <= A;
                WHEN OTHERS =>
                    state <= A;
            END CASE;
        END IF;
    END PROCESS;
```

最後主要的部分則是依據狀態控制各個暫存器的輸入輸出與否，只要將前面所完成的 components 串接即可。

```
52      StateA <= '1' WHEN state = A ELSE '0';
53      StateB <= '1' WHEN state = B ELSE '0';
54      StateC <= '1' WHEN state = C ELSE '0';
55      StateD <= '1' WHEN state = D ELSE '0';
56
57
58      Rin(1) <=   RinExt(1) WHEN state = A ELSE
59                  '1' WHEN state = D ELSE '0';
60      Rin(2) <=   RinExt(2) AND (NOT Rin(1)) WHEN state = A ELSE
61                  '1' WHEN state = C ELSE '0';
62      Rin(3) <=   RinExt(3) AND (NOT Rin(1)) AND (NOT Rin(2)) WHEN state = A ELSE
63                  '1' WHEN state = B ELSE '0';
64      Rout(1) <=  '1' WHEN state = C ELSE '0';
65      Rout(2) <=  '1' WHEN state = B ELSE '0';
66      Rout(3) <=  '1' WHEN state = D ELSE '0';
67
68      tri_ext : trin PORT MAP(Data, Extern, BusWires);
69      reg1 : regn PORT MAP(BusWires, Rin(1), Clock, R1);
70      reg2 : regn PORT MAP(BusWires, Rin(2), Clock, R2);
71      reg3 : regn PORT MAP(BusWires, Rin(3), Clock, R3);
72      tri1 : trin PORT MAP(R1, Rout(1), BusWires);
73      tri2 : trin PORT MAP(R2, Rout(2), BusWires);
74      tri3 : trin PORT MAP(R3, Rout(3), BusWires);
75
76      Show_R1 <= R1;
77      Show_R2 <= R2;
78      Show_R3 <= R3;
```

Modelsim 模擬：

3. 實驗心得：

梁皓鈞(104360098)：

這一次實驗主要在於匯流排跟Tri-State Buffer 的概念。對於這一點, VHDL 已經幫我們做了很多, 不需要自己去想盡辦法做出Tri-State 的情況, 只要直接引用library ieee 中的Z 就可以當作Tri-State 去做。而Tri-State Buffer 也只是一個Process IF 去輸出Z 就可以了。

但這一次最難的是我能理解整個運作的流程跟概念, 可是我對於怎樣用程式碼去寫出來, 以及Hardware Description Language 同步的概念開始有點混亂。

畢竟VHDL跟我們平常寫習慣的Programming Language 是不一樣的, 因此對於同步的處理要小心。因此這一次主要是由晟毅去寫程式, 我在旁邊學習著, 這樣有助我們的實驗效率同時也令我可以更快學習起來, 並且在期末專題發揮一下。

有關彈跳問題真的很難解決, 只能不要太大力去按下去慢慢按來減少出錯機會。我們從使用Switch 改成使用push-button, 出錯機會大大減少。

洪晟毅(104590048)：

本次實驗最困難的地方，我認為是對觀念的理解與整合，整體的程式碼非常的簡單，但是最初的理解則是最困難的。必須先了解到暫存器的輸入、輸入與匯流排控制之間的關係，要能確實地讓各個控制信號正確地輸入，否則就容易發生一些衝突，剛開始我在撰寫程式碼時，因為忽略了讓三態緩衝器輸出高阻抗(Z)，導致compile的過程中一直產生信號衝突的錯誤。而到後面，雖能理解匯流排操作的機制、暫存器交換的摩爾狀態機的原理，但卻遲遲無法將兩者結合，經過一番努力及測試，才終於完成此次的實驗。好不容易完成實驗，卻遇到一個突發狀況，原先採用switch開關模擬clock的輸入，但卻有彈跳現象的發生，後來我們選擇不繼續撰寫防彈跳電路，採用push-button作為替代方案，雖仍有機率發生彈跳現象，但因其純為clock的模擬，所以我們也就未繼續深入。

4. 組員貢獻度及工作內容：

名字	負責項目內容	貢獻比例	貢獻總和
皓鈞	負責實驗結果驗證	20%	50%
	報告實驗內容、實驗過程撰寫	20%	
	報告實驗心得撰寫	10%	
晟毅	負責主要程式碼撰寫	25%	50%
	報告實驗內容的程式碼解釋	15%	
	報告實驗心得撰寫	10%	
總計		100%	100%