Lab5-

Counter and Birth date Display

助教：東昇、憲億、琮閔、韋廷、文駿

Outline

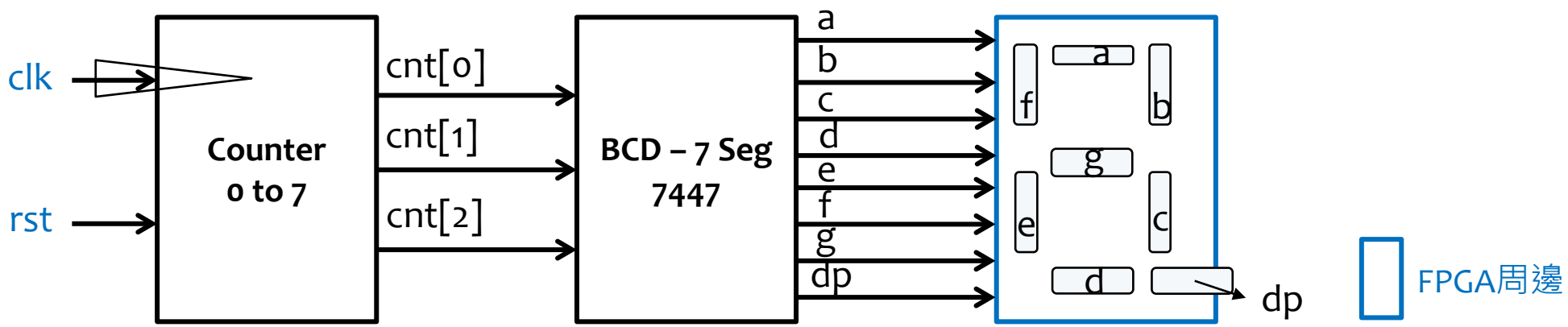
- 課程目標
- 循序顯示 0~7
- 生日碼顯示器
- 驗收內容
- Demo 需知

課程目標

承上個 Lab，大家已經知道如何使用 Nexys4 DDR FPGA 開發版上的七段顯示器、開關等功能，本次課程將教大家

- 使用 Counter、BCD，在七段顯示器上依序顯示 0~7
- 實作 Code Converter 組合電路，並控制七段顯示器顯示 0~7 或生日碼

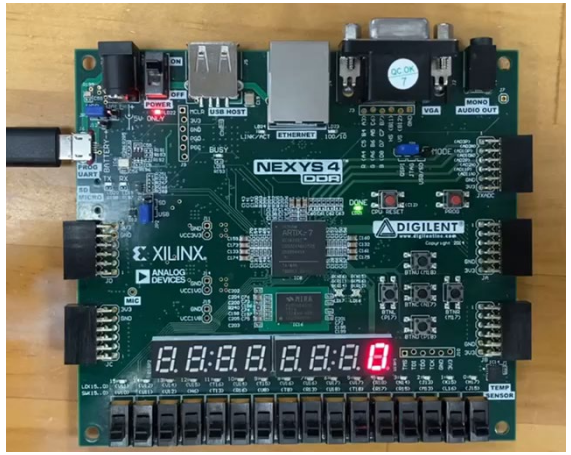
課堂練習 - 循序0~7



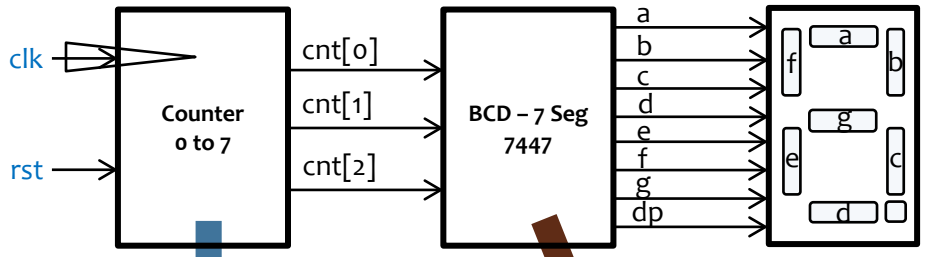
sw15 (rst)

- 0：維持在 0
- 1：循序顯示 0~7

範例：



循序 0~7 - 程式說明



cnt [2:0]	a	b	c	d	e	f	g
0(000)	0	0	0	0	0	0	1
1(001)	1	0	0	1	1	1	1
2(010)	0	0	1	0	0	1	0
3(011)	0	0	0	0	1	1	0
4(100)	1	0	0	1	1	0	0
5(101)	0	1	0	0	1	0	0
6(110)	0	1	0	0	0	0	0
7(111)	0	0	0	1	1	0	1

```
//**COUNTER**//
always@ (posedge clk_2hz or negedge rst) begin
    if (~rst)
        cnt <= 3'b0;
    else
        cnt <= cnt + 3'b1;
end
```

```
//**BCD_to_7seg**//
always@(*) begin
    seg_data[0] = (cnt[2] & !cnt[1] & !cnt[0]) | (!cnt[2] & !cnt[1] & cnt[0]);                                  //CA
    seg_data[1] = (cnt[2] & !cnt[1] & cnt[0]) | (cnt[2] & cnt[1] & !cnt[0]);                                    //CB
    seg_data[2] = !cnt[2] & cnt[1] & !cnt[0];                                                                   //CC
    seg_data[3] = (cnt[2] & !cnt[1] & !cnt[0]) | (!cnt[2] & !cnt[1] & cnt[0]) | (cnt[2] & cnt[1] & cnt[0]);    //CD
    seg_data[4] = (cnt[2] & !cnt[1]) | (!cnt[2] & cnt[0]) | (cnt[2] & cnt[0]);                                  //CE
    seg_data[5] = (!cnt[2] & cnt[0]) | (!cnt[2] & cnt[1]);                                                      //CF
    seg_data[6] = (!cnt[2] & !cnt[1]) | (cnt[2] & cnt[1] & cnt[0]) ;                                            //CG
end
```

- 實驗目的是顯示 0~7，因此 3bit 可以完全表示
- 使 counter 每一個 clock cycle 就加一，達到循序的效果
- 因為 cnt 經過每一個 clock cycle 會加一，代表每一個 cycle 的輸出會不同
- 我們可以使用 BCD – 7 seg (7447) 的真值表並利用 min sop 化簡布林函數，可以在七段顯示器上依序顯示 0~7

循序 0~7 - 實作流程

同學在寫 demo 作業時先利用 testbench 確認自己撰寫的 Code 功能正確後，再上板子進行驗證，流程如下：

① 程式模擬 (於命令提示字元輸出，觀察結果是否與目標一致)

② 周邊整合 (將 verilog 中的 input、output 與 FPGA 上的腳位連接)

③ 上板驗證 (觀察七段顯示器)

循序顯示 0~7 - 程式模擬

① 開啟資料夾 lab5/範例/cnt0_7 並於路徑處輸入 cmd 開啟命令提示字元

② 輸入「iverilog –o test tb_lab5.v」
- 輸入檔案的第一個字母後按「tab」即會出現第一字母相同的檔案

③ 輸入「vvp test」，即可觀察到 output根據 cnt 的不同在輸出

循序顯示 0~7 - 周邊整合

UCF (User Constraint File)

- 參考 Lab4 第七頁，因本次實驗僅用到最右邊的七段顯示器，所以我們只需令ANo為o即可
- 七段顯示器輸出的數字則由 BCD – 7 seg 的輸出「seg_data」來決定
- rst 設定在 sw15

- 修改 Nexys4.xdc：

```
assign {AN7,AN6,AN5,AN4,AN3,AN2,AN1,AN0} = 8'b1111_1110; //turn on the 8th 7seg LED
assign {CG,CF,CE,CD,CC,CB,CA} = seg_data;
```

7 seg 中各個 LED 腳位　　Verilog 中的 output

```
##7 segment display
set_property -dict { PACKAGE_PIN T10   IOSTANDARD LVCMOS33 } [get_ports { CA }]; #IO_L24N_T3_A00_D16_14 Sch=ca
set_property -dict { PACKAGE_PIN R10   IOSTANDARD LVCMOS33 } [get_ports { CB }]; #IO_25_14 Sch=cb
set_property -dict { PACKAGE_PIN K16   IOSTANDARD LVCMOS33 } [get_ports { CC }]; #IO_25_15 Sch=cc
set_property -dict { PACKAGE_PIN K13   IOSTANDARD LVCMOS33 } [get_ports { CD }]; #IO_L17P_T2_A26_15 Sch=cd
set_property -dict { PACKAGE_PIN P15   IOSTANDARD LVCMOS33 } [get_ports { CE }]; #IO_L13P_T2_MRCC_14 Sch=ce
set_property -dict { PACKAGE_PIN T11   IOSTANDARD LVCMOS33 } [get_ports { CF }]; #IO_L19P_T3_A10_D26_14 Sch=cf
set_property -dict { PACKAGE_PIN L18   IOSTANDARD LVCMOS33 } [get_ports { CG }]; #IO_L4P_T0_D04_14 Sch=cg

set_property -dict { PACKAGE_PIN J17   IOSTANDARD LVCMOS33 } [get_ports { AN0 }]; #IO_L23P_T3_FOE_B_15 Sch=an[0]
set_property -dict { PACKAGE_PIN J18   IOSTANDARD LVCMOS33 } [get_ports { AN1 }]; #IO_L23N_T3_FWE_B_15 Sch=an[1]
set_property -dict { PACKAGE_PIN T9    IOSTANDARD LVCMOS33 } [get_ports { AN2 }]; #IO_L24P_T3_A01_D17_14 Sch=an[2]
set_property -dict { PACKAGE_PIN J14   IOSTANDARD LVCMOS33 } [get_ports { AN3 }]; #IO_L19P_T3_A22_15 Sch=an[3]
set_property -dict { PACKAGE_PIN P14   IOSTANDARD LVCMOS33 } [get_ports { AN4 }]; #IO_L8N_T1_D12_14 Sch=an[4]
set_property -dict { PACKAGE_PIN T14   IOSTANDARD LVCMOS33 } [get_ports { AN5 }]; #IO_L14P_T2_SRCC_14 Sch=an[5]
set_property -dict { PACKAGE_PIN K2    IOSTANDARD LVCMOS33 } [get_ports { AN6 }]; #IO_L23P_T3_35 Sch=an[6]
set_property -dict { PACKAGE_PIN U13   IOSTANDARD LVCMOS33 } [get_ports { AN7 }]; #IO_L23N_T3_A02_D18_14 Sch=an[7]
```

七段顯示器驅動腳位

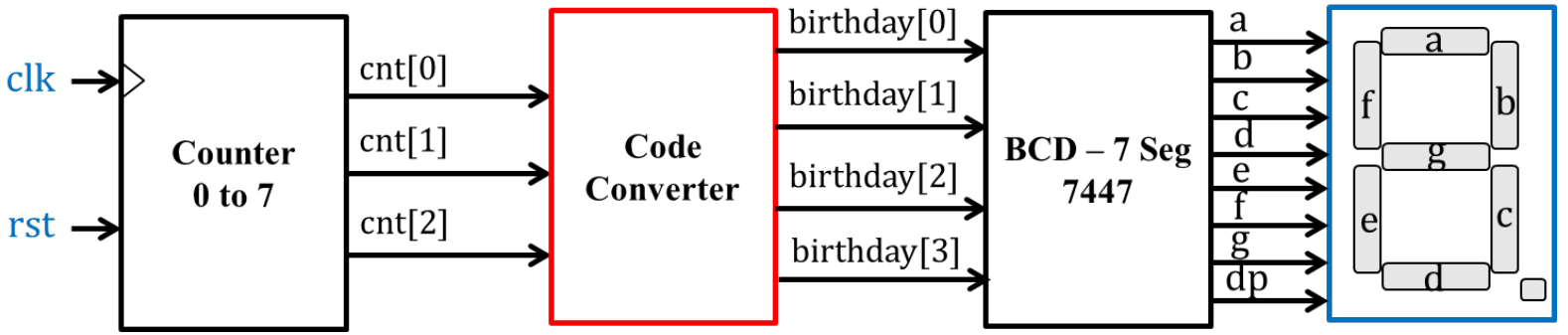
```
set_property -dict { PACKAGE_PIN V10   IOSTANDARD LVCMOS33 } [get_ports { rst }]; #IO_L21P_T3_DQS_14 Sch=sw[15]
```

sw15 在板上的代碼　　Verilog 中的 rst

生日碼顯示器

前面解說完 Counter 和 BCD – 7 seg (7447) 之間的關係之後，接下來我們要介紹 Code Converter 使我們可以延伸應用前面的 design，在七段顯示器上依序顯示生日

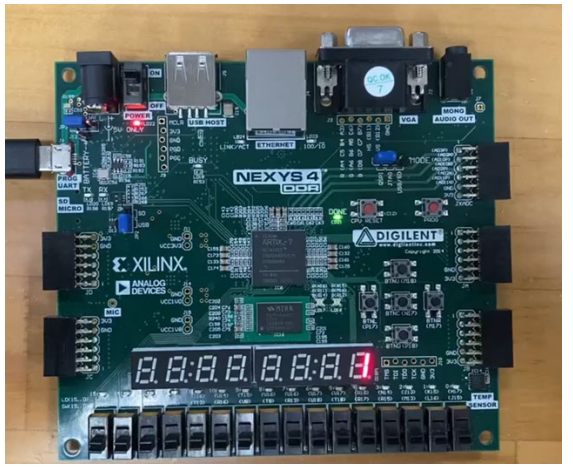
範例練習 - 生日碼顯示器



sw15 (rst)

- 0：維持在1 (生日碼第一個數字)
- 1：循序顯示生日碼

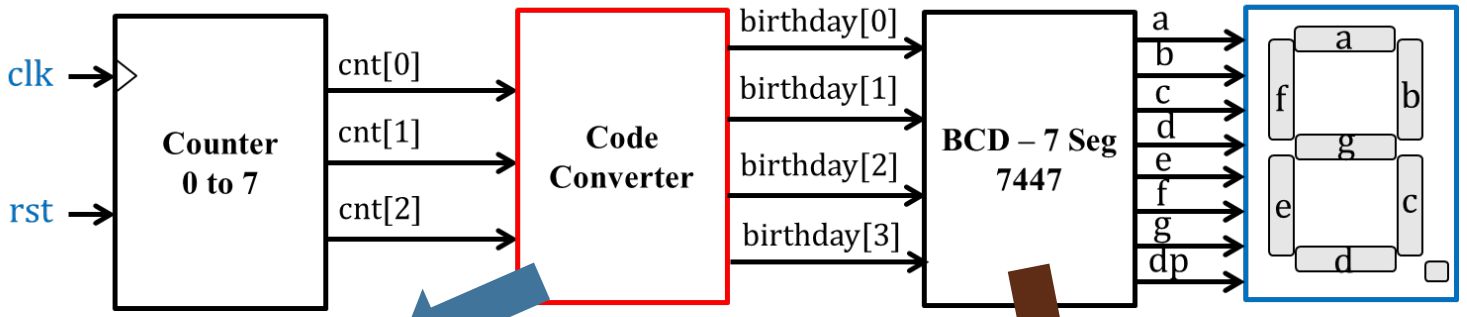
範例：



Code Converter 實作 (1/2)

min SOP (Sum of Product)

- 生日碼顯示器可以沿用前一個範例的想法，一樣以 cnt 為輸入，只是這次在 Code Converter 的 output 會依序顯示出預先設計的生日，再利用 7447 將生日輸出至七段顯示器




```
//**birthday_display**//
always@(*)begin
    birth_num[3] = (!cnt[2] & !cnt[1] & cnt[0]) | (cnt[2] & cnt[1] & cnt[0]);
    birth_num[2] = !cnt[2] & cnt[1];
    birth_num[1] = (!cnt[2] & cnt[1]) | (cnt[2] & !cnt[1] & cnt[0]);
    birth_num[0] = (!cnt[2] & !cnt[1]) | (cnt[2] & cnt[0]) | (cnt[1] & !cnt[0]);
end
```

- Input 為 counter，利用 min sop 所化簡的方程式來輸出預先設計的生日碼

```
//**BCD_to_7SEG**//
always@(*) begin
    seg_data[6] = (!birth_num[3] & !birth_num[2] & !birth_num[1]) | (!birth_num[3] & birth_num[2] & birth_num[1] & birth_num[0]);
    seg_data[5] = (!birth_num[3] & !birth_num[2] & birth_num[0]) | (!birth_num[3] & !birth_num[2] & birth_num[1]);
    seg_data[4] = (!birth_num[3] & birth_num[2] & !birth_num[1]) | (!birth_num[2] & !birth_num[1] & birth_num[0]) | (!birth_num[3] & birth_num[2] & birth_num[0]) |  (!birth_num[3] & birth_num[1] & birth_num[0]);
    seg_data[3] = (!birth_num[3] & birth_num[2] & !birth_num[1] & !birth_num[0]) |  (!birth_num[3] & !birth_num[2] & !birth_num[1] & birth_num[0]) |  (!birth_num[3] & birth_num[2] & birth_num[1] & birth_num[0]);
    seg_data[2] = (!birth_num[3] & !birth_num[2] & birth_num[1] & !birth_num[0]);
    seg_data[1] = (!birth_num[3] & birth_num[2] & !birth_num[1] & birth_num[0]) |  (!birth_num[3] & birth_num[2] & birth_num[1] & !birth_num[0]);
    seg_data[0] = (!birth_num[3] & birth_num[2] & !birth_num[1] & !birth_num[0]) |  (!birth_num[3] & !birth_num[2] & !birth_num[1] & birth_num[0]);
end
```

- 跟上一個範例不同，這次需要輸出到9，所以input會用到4bit

Code Converter 實作 (2/2)

High-level Verilog description

- 每次都要找 min SOP 其實很費力，在 verilog 語法中有一個方法為「case」，可以使 input 快速轉換成我們需要的輸出，這邊就以生日碼顯示器作為範例

❖ Code Converter

```
case(cnt)
    3'b000:
        seg_number = 4'd1;
    3'b001:
        seg_number = 4'd9;
    3'b010:
        seg_number = 4'd7;
    3'b011:
        seg_number = 4'd6;
    3'b100:
        seg_number = 4'd0;
    3'b101:
        seg_number = 4'd3;
    3'b110:
        seg_number = 4'd1;
    3'b111:
        seg_number = 4'd9;
endcase
```

- 根據括弧中的 input (cnt) 會有相對應的 output (seg_number)
- 括弧中可為定值或變數，也可放運算子

❖ BCD to 7seg (7447)

```
always@(*) begin
  case(seg_number)
    4'd0:seg_data = 7'b100_0000;
    4'd1:seg_data = 7'b111_1001;
    4'd2:seg_data = 7'b010_0100;
    4'd3:seg_data = 7'b011_0000;
    4'd4:seg_data = 7'b001_1001;
    4'd5:seg_data = 7'b001_0010;
    4'd6:seg_data = 7'b000_0010;
    4'd7:seg_data = 7'b101_1000;
    4'd8:seg_data = 7'b000_0000;
    4'd9:seg_data = 7'b001_0000;
    default: seg_number = seg_number;
  endcase
```

生日碼顯示器 - 周邊整合

因生日碼顯示器的 input、output 皆與循序顯示 0~7 的範例相同，故 XDC 的設定皆與 p. 8 相同，在此不再贅述

```
assign {AN7,AN6,AN5,AN4,AN3,AN2,AN1,AN0} = 8'b1111_1110; //turn on the 8th 7seg LED
assign {CG,CF,CE,CD,CC,CB,CA} = seg_data;
```

7 seg 中各個 LED 腳位　　Verilog 中的 output

```
##7 segment display
set_property -dict { PACKAGE_PIN T10   IOSTANDARD LVCMOS33 } [get_ports { CA }]; #IO_L24N_T3_A00_D16_14 Sch=ca
set_property -dict { PACKAGE_PIN R10   IOSTANDARD LVCMOS33 } [get_ports { CB }]; #IO_25_14 Sch=cb
set_property -dict { PACKAGE_PIN K16   IOSTANDARD LVCMOS33 } [get_ports { CC }]; #IO_25_15 Sch=cc
set_property -dict { PACKAGE_PIN K13   IOSTANDARD LVCMOS33 } [get_ports { CD }]; #IO_L17P_T2_A26_15 Sch=cd
set_property -dict { PACKAGE_PIN P15   IOSTANDARD LVCMOS33 } [get_ports { CE }]; #IO_L13P_T2_MRCC_14 Sch=ce
set_property -dict { PACKAGE_PIN T11   IOSTANDARD LVCMOS33 } [get_ports { CF }]; #IO_L19P_T3_A10_D26_14 Sch=cf
set_property -dict { PACKAGE_PIN L18   IOSTANDARD LVCMOS33 } [get_ports { CG }]; #IO_L4P_T0_D04_14 Sch=cg

set_property -dict { PACKAGE_PIN J17   IOSTANDARD LVCMOS33 } [get_ports { AN0 }]; #IO_L23P_T3_FOE_B_15 Sch=an[0]
set_property -dict { PACKAGE_PIN J18   IOSTANDARD LVCMOS33 } [get_ports { AN1 }]; #IO_L23N_T3_FWE_B_15 Sch=an[1]
set_property -dict { PACKAGE_PIN T9    IOSTANDARD LVCMOS33 } [get_ports { AN2 }]; #IO_L24P_T3_A01_D17_14 Sch=an[2]
set_property -dict { PACKAGE_PIN J14   IOSTANDARD LVCMOS33 } [get_ports { AN3 }]; #IO_L19P_T3_A22_15 Sch=an[3]
set_property -dict { PACKAGE_PIN P14   IOSTANDARD LVCMOS33 } [get_ports { AN4 }]; #IO_L8N_T1_D12_14 Sch=an[4]
set_property -dict { PACKAGE_PIN T14   IOSTANDARD LVCMOS33 } [get_ports { AN5 }]; #IO_L14P_T2_SRCC_14 Sch=an[5]
set_property -dict { PACKAGE_PIN K2    IOSTANDARD LVCMOS33 } [get_ports { AN6 }]; #IO_L23P_T3_35 Sch=an[6]
set_property -dict { PACKAGE_PIN U13   IOSTANDARD LVCMOS33 } [get_ports { AN7 }]; #IO_L23N_T3_A02_D18_14 Sch=an[7]
```

七段顯示器驅動腳位

```
set_property -dict { PACKAGE_PIN V10   IOSTANDARD LVCMOS33 } [get_ports { rst }]; #IO_L21P_T3_DQS_14 Sch=sw[15]
```

sw15 在板上的代碼　　Verilog 中的 rst

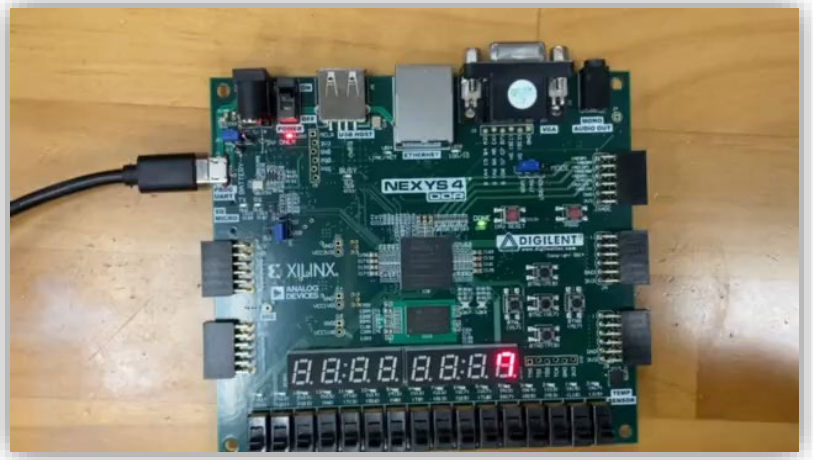
驗收內容

Demo 時請 show 出以下兩個內容：

① 利用 testbench 於命令提示字元中顯示生日　ex：

```
D:\acadamic\LPAP\DD_lab5\cnt0_7>vvp test
VCD info: dumpfile lab5.fsdb opened for output.
              250005  birth =  9
              500005  birth =  7
              750005  birth =  6
             1000005  birth =  0
             1250005  birth =  3
             1500005  birth =  1
             1750005  birth =  9
             2000005  birth =  1
             2250005  birth =  9
             2500005  birth =  7
             2750005  birth =  6
             3000005  birth =  0
             3250005  birth =  3
             3500005  birth =  1
             3750005  birth =  9
```

② 結合循序輸出 0~7 與生日碼顯示器　ex：



- 以sw14為開關，**為0時最右邊的七段顯示器會依序顯示0~7，為1時則會依序顯示出自己的生日**
  - Demo時需準備可證明自己生日之證件！

Demo 需知

地點 : 501A

時間 : 4/19(一) 及 4/21(三)，詳細時間依公告時間表為準

- 可提前5分鐘入場準備，其餘時間違規進入，一次扣總成績3分
- 安排時段內無法展示請即刻離場，違者一次扣總成績5分
- 請帶著Nexys-4開發板以及 .bit 檔 (請使用隨身碟不要用雲端)
- 可使用自己的筆電 Demo

附錄

Min SOP

卡諾圖是一種以圖形化的方式來進行布林代數化簡的表示圖

由於卡諾圖本身亦記錄布林函數的數值，因此我們也可以視其為一種特別版本的真值表

範例：

a/b	b	b’
a	1	1
a’	1	0

紅色圈選處所代表的函數為 a
(因為 b的數值不會影響結果)

綠色圈選處所代表的函數為 b
(因為 a的數值不會影響結果)

故化簡後布林函數為 F = a + b

Code Converter

Code Converter Table Example：

count[2:0]	birthday[3:0]
0(000)	1
1(001)	9
2(010)	7
3(011)	6
4(100)	0
5(101)	3
6(110)	1
7(111)	9

Birthday [3:0]	a	b	c	d	e	f	g
0 (0000)	0	0	0	0	0	0	1
1 (0001)	1	0	0	1	1	1	1
2 (0010)	0	0	1	0	0	1	0
3 (0011)	0	0	0	0	1	1	0
4 (0100)	1	0	0	1	1	0	0
5 (0101)	0	1	0	0	1	0	0
6 (0110)	0	1	0	0	0	0	0
7 (0111)	0	0	0	1	1	0	1
8 (1000)	0	0	0	0	0	0	0
9 (1001)	0	0	0	0	1	0	0