

# 數位系統技術



# Keyboard

Ren-Der Chen (陳仁德)

Department of Computer Science and  
Information Engineering

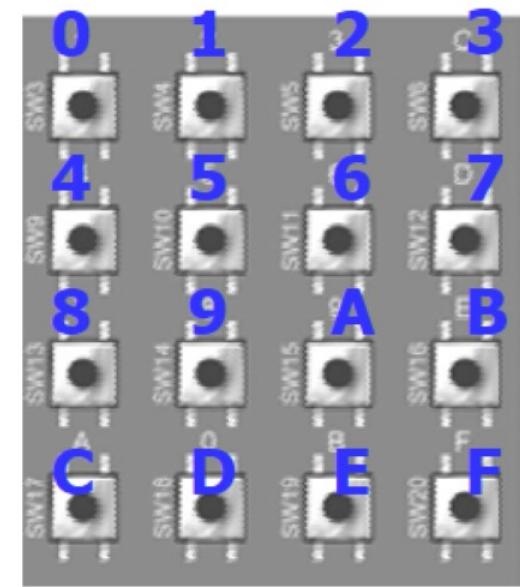
National Changhua University of Education

E-mail: rdchen@cc.ncue.edu.tw

Spring, 2025

# 實驗內容 (1/2)

- 設計一個資料儲存電路**buffer[23:0]**以存放6個數字，每個數字4位元，且這6個數字可同時會顯示在Seg7中。
- 按下keyboard的其中一個16進位數字0~F鍵時，該數字會存入buffer[3:0]位置，原先在buffer[19:0]的數字則會左移。



Keyboard

## 實驗內容 (2/2)

- 當keyboard有按鍵按下時( $\text{press}=1$ )，Led9會顯示亮燈，同時Led3~Led0會顯示所按之鍵的二進位值( $\text{scan\_code}[3:0]$ )，亮燈表示1，不亮表示0。



# Example

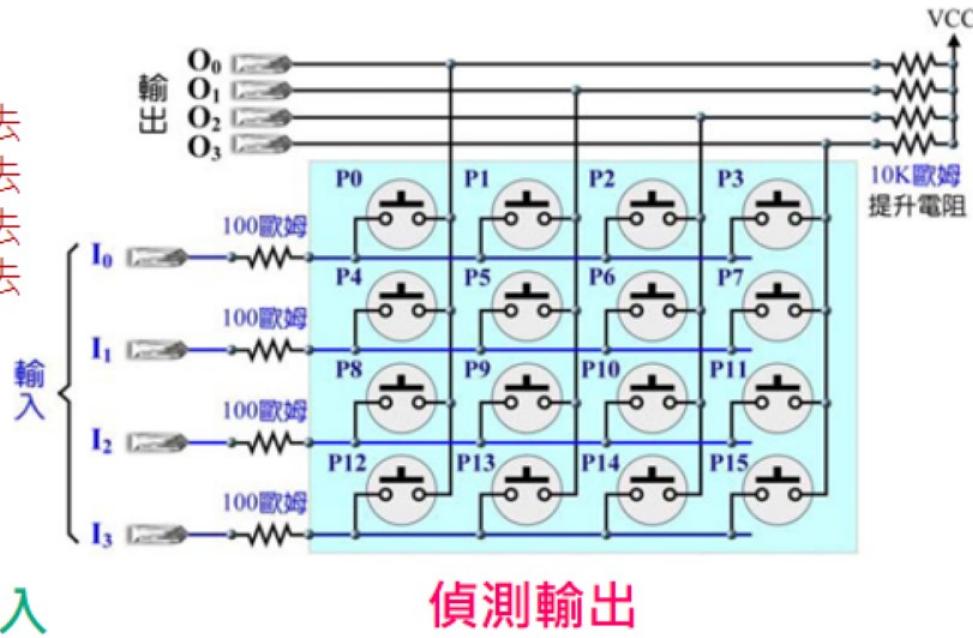
---

- 按下**3** · buffer = \_\_\_\_\_3
- 按下**4** · buffer = \_\_\_\_\_34
- 按下**1** · buffer = \_\_341
- 按下**A** · buffer = \_\_341A
- 按下**5** · buffer = \_341A5
- 按下**F** · buffer = **341A5F**
- 按下**7** · buffer = **41A5F7**
- 按下**8** · buffer = **1A5F78**

# Keyboard 工作原理

Ex: 給定  $I=1101$

1. 若偵測到  $O=1110$ ，表示 P4 按下去
2. 若偵測到  $O=1101$ ，表示 P5 按下去
3. 若偵測到  $O=1011$ ，表示 P6 按下去
4. 若偵測到  $O=0111$ ，表示 P7 按下去



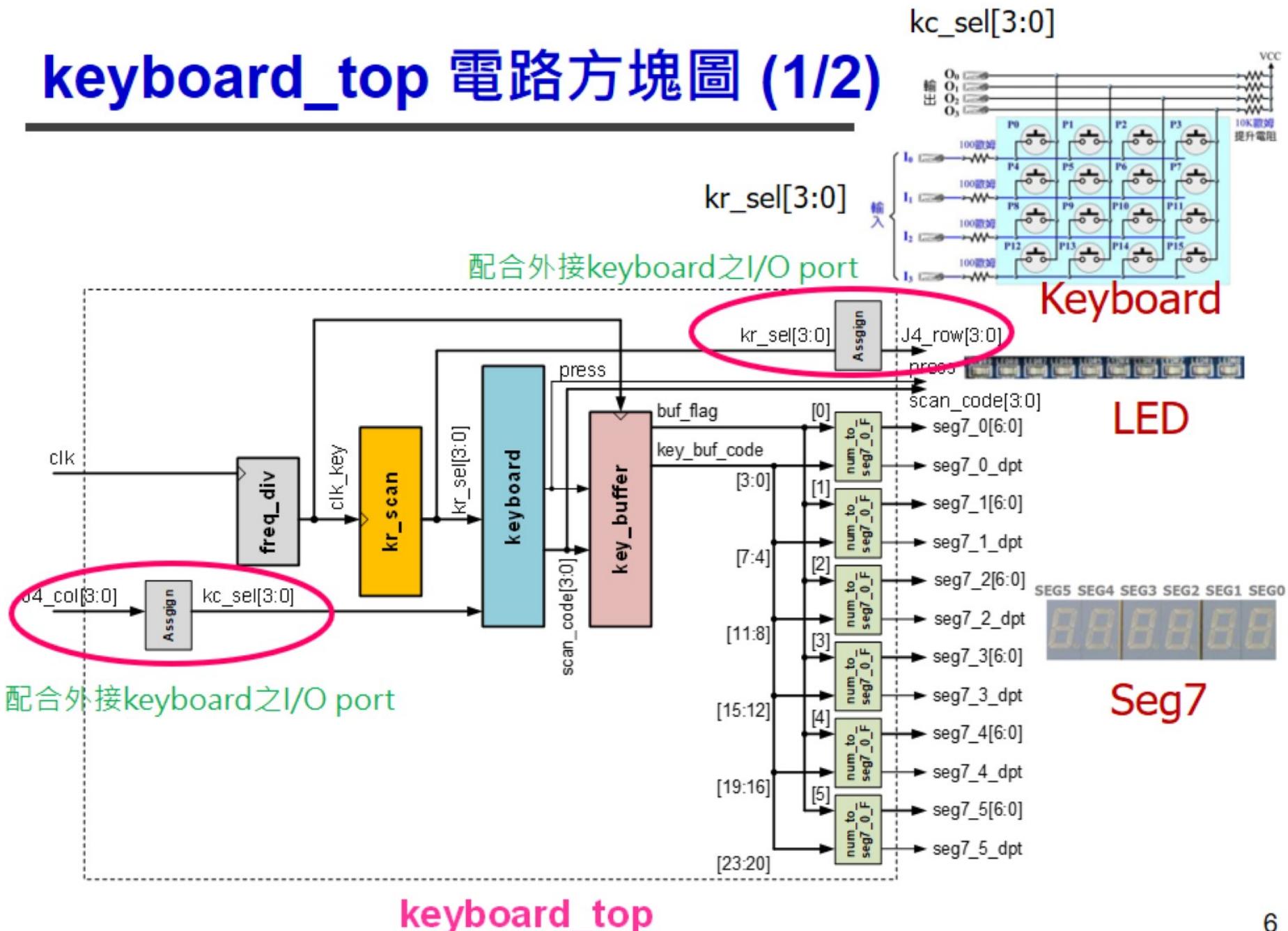
掃描輸入

偵測輸出

順序	對應鍵	$I_3I_2I_1I_0$	$O_3O_2O_1O_0$	$O_3O_2O_1O_0$	$O_3O_2O_1O_0$	$O_3O_2O_1O_0$
0	P <sub>3</sub> P <sub>2</sub> P <sub>1</sub> P <sub>0</sub>	1110	按 P <sub>0</sub> :1110	按 P <sub>1</sub> :1101	按 P <sub>2</sub> :1011	按 P <sub>3</sub> :0111
1	P <sub>7</sub> P <sub>6</sub> P <sub>5</sub> P <sub>4</sub>	1101	按 P <sub>4</sub> :1110	按 P <sub>5</sub> :1101	按 P <sub>6</sub> :1011	按 P <sub>7</sub> :0111
2	P <sub>11</sub> P <sub>10</sub> P <sub>9</sub> P <sub>8</sub>	1011	按 P <sub>8</sub> :1110	按 P <sub>9</sub> :1101	按 P <sub>10</sub> :1011	按 P <sub>11</sub> :0111
3	P <sub>15</sub> P <sub>14</sub> P <sub>13</sub> P <sub>12</sub>	0111	按 P <sub>12</sub> :1110	按 P <sub>13</sub> :1101	按 P <sub>14</sub> :1011	按 P <sub>15</sub> :0111

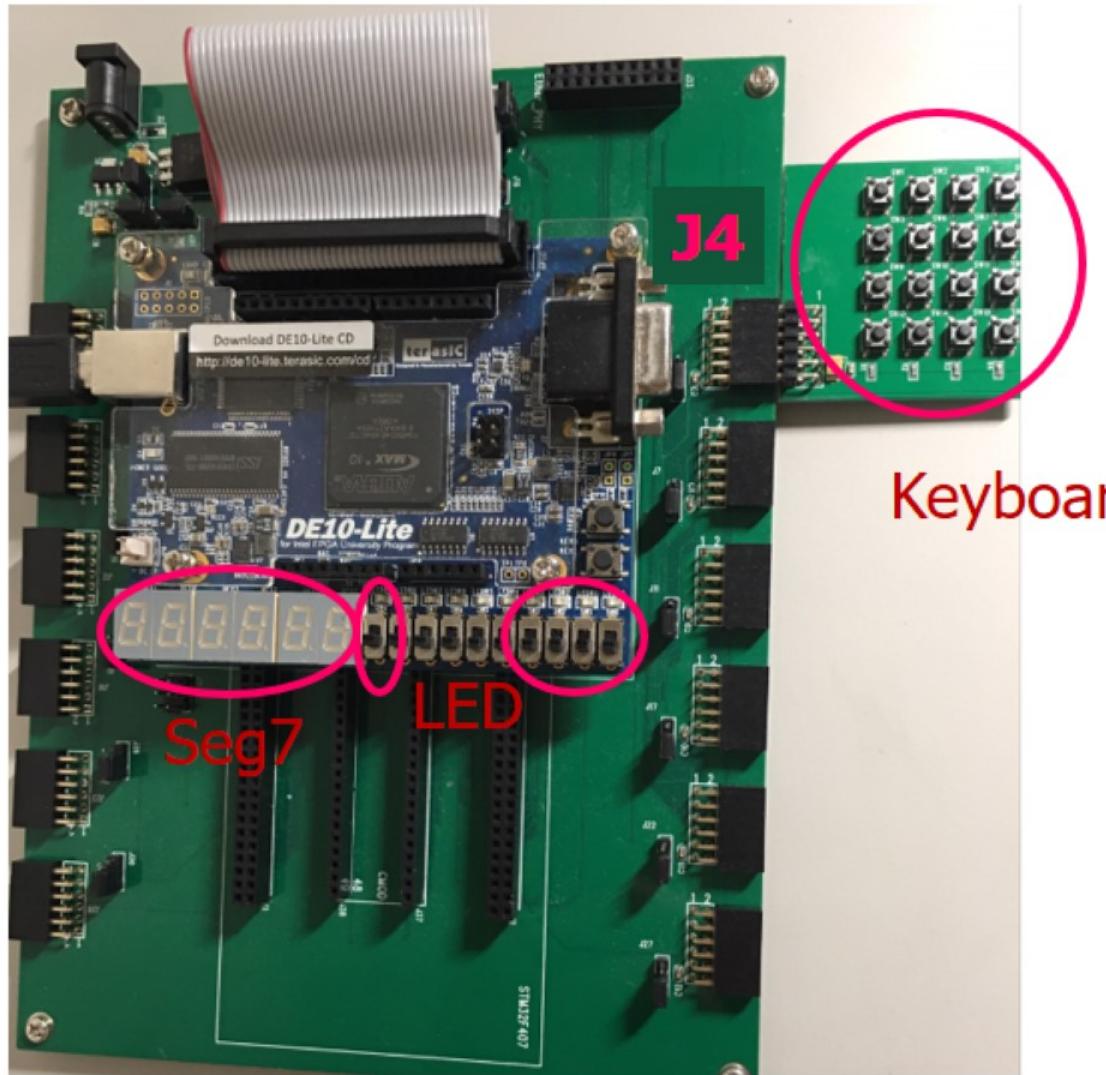
4×4 鍵盤掃瞄原理(無按鍵時  $O_3O_2O_1O_0=1111$ )

# keyboard\_top 電路方塊圖 (1/2)



keyboard\_top

## keyboard\_top 電路方塊圖 (2/2)



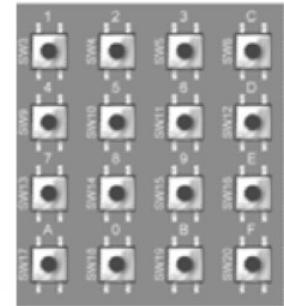
# Design 1: kr\_scan

- 設計keyboard之row掃描電路kr\_scan.v，由上至下掃描keyboard。

```
module kr_scan(clk, rst, kr_out);
    input clk, rst;
    output reg [3:0] kr_out;

    always@ (posedge clk or posedge rst) begin
        if (rst)
            kr_out = [redacted] // top row
        else // left circular shift
            kr_out = [redacted]
        // 1110 -> 1101 -> 1011 -> 0111
        // select row from up to down
    end
endmodule
```

1110  
1101  
1011  
0111



## Design 2: keyboard (1/6)

- 設計keyboard偵測電路keyboard.v，判斷keyboard是否有按鍵按下，並產生按鍵之二進位編碼。

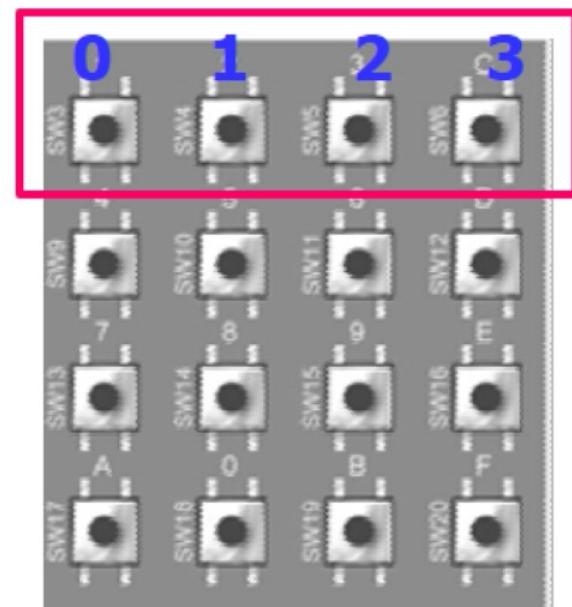
```
module keyboard(kr, kc, press, scan_code);
    input [3:0] kr;
    input [3:0] kc;
    output reg press;
    output reg [3:0] scan_code;

    always@(kr or kc) begin
        case(kr)
```

1. kr[3:0]為keyboard的row掃描輸入
2. kc[3:0]為keyboard的column偵測輸出
3. 若keyboard偵測到有任何鍵按下去，press之值為1
4. scan\_code[3:0]為所按之鍵的二進位編碼

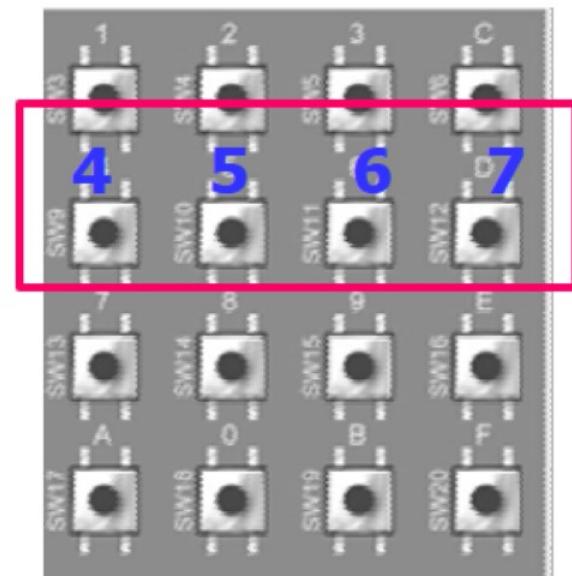
## Design 2: keyboard (2/6)

```
4'b1110: // 1st row
case(kc)
    4'b1110: begin press = 1'b1;
                scan_code = 4'h0;
                end // 0
    4'b1101: begin press = 1'b1;
                scan_code = 4'h1;
                end // 1
    4'b1011: begin press = 1'b1;
                scan_code = 4'h2;
                end // 2
    4'b0111: begin press = 1'b1;
                scan_code = 4'h3;
                end // 3
    default: begin press = 1'b0;
                scan_code = 4'h0;
                end // default
endcase
```



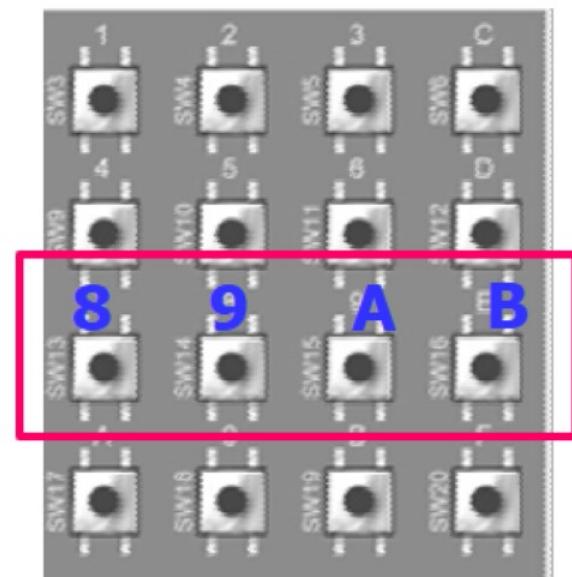
## Design 2: keyboard (3/6)

```
4'b1101: // 2nd row
case(kc)
    4'b1110: begin
        end //
    4'b1101: begin
        end //
    4'b1011: begin
        end //
    4'b0111: begin
        end //
default: begin
    end //
endcase
```



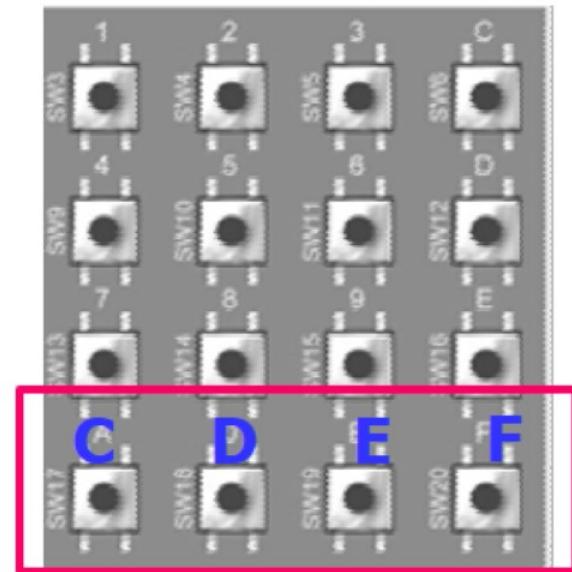
## Design 2: keyboard (4/6)

```
4'b1011: // 3rd row
  case(kc)
    4'b1110: begin
      end // 
    4'b1101: begin
      end // 
    4'b1011: begin
      end // 
    4'b0111: begin
      end // 
  default: begin
    end // 
endcase
```



## Design 2: keyboard (5/6)

```
4'b0111: // 4th row
  case(kc)
    4'b1110: begin
      end // 
    4'b1101: begin
      end // 
    4'b1011: begin
      end // 
    4'b0111: begin
      end // 
  default: begin
    end // 
endcase
```



# Design 2: keyboard (6/6)

```
    default: begin    press = 1'b0;  
                      scan_code = 4'h0;  
    end  
endcase  
end  
endmodule
```

# Design 3: key\_buffer (1/2)

- 設計一個可存放6個數字之buffer電路key\_buffer.v，每個數字4個位元。

```
module key_buffer(clk, rst, press, scan_code,  
buf_flag, key_buf_code);  
  
    input clk, rst, press;  
    input [3:0] scan_code;  
    output [5:0] buf_flag;  
    output [23:0] key_buf_code;  
  
    reg [5:0] buf_flag;  
    reg [23:0] key_buf_code;
```

1. press為1時表示keyboard有按鍵被按下
2. scan\_code[3:0]為所按下之鍵的二進位編碼
3. buf\_flag[5:0]為記錄buffer電路中每個位置是否有數字存入
4. key\_buf\_code[23:0]為存放6個數字之24位元buffer電路，每個數字4個位元。

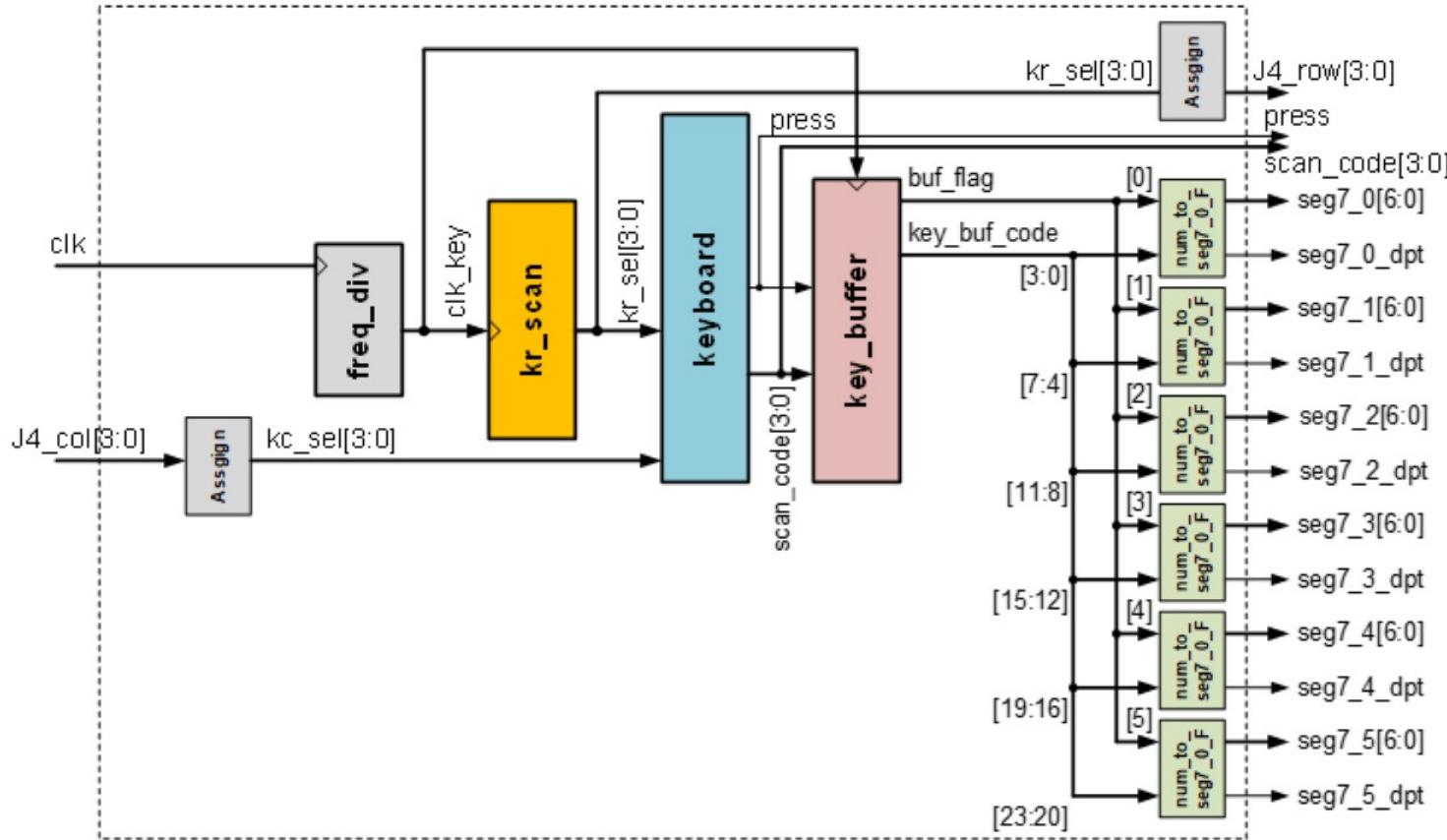
## Design 3: key\_buffer (2/2)

```
always@(posedge clk or posedge rst) begin
    if (rst) begin
        buf_flag = 6'b000000;           // 初始時，buffer電路是空的
        key_buf_code = 24'h000000;
    end
    else
        if (press == 1'b1) begin
            buf_flag = [ ];           // 若有按鍵按下，數字將存入buffer
            key_buf_code[23:0] = [ ];   // , buf_flag[0]設為1，原
                                         //   buf_flag[4:0]之值左移
        end
    end
endmodule
```

所按之鍵的二進位編碼scan\_code[3:0]存入key\_buf\_code[3:0]，原本的[23:20]左移出buffer，[19:0]左移1個數字(4個位元)

# Design 4: keyboard\_top (1/4)

- 設計keyboard\_top.v，整合所有電路模組。



keyboard\_top

## Design 4: keyboard\_top (2/4)

---

```
module keyboard_top (clk, rst, J4_col, J4_row, press, scan_code,  
seg7_0, seg7_0_dpt, seg7_1, seg7_1_dpt, seg7_2, seg7_2_dpt,  
seg7_3, seg7_3_dpt, seg7_4, seg7_4_dpt, seg7_5, seg7_5_dpt);  
  
input clk, rst;  
input [3:0] J4_col;  
output [3:0] J4_row;  
output press;  
output [3:0] scan_code;  
output [6:0] seg7_0, seg7_1, seg7_2, seg7_3, seg7_4, seg7_5;  
output seg7_0_dpt, seg7_1_dpt, seg7_2_dpt,  
seg7_3_dpt, seg7_4_dpt, seg7_5_dpt;
```

# Design 4: keyboard\_top (3/4)

---

```
    wire [3:0] kr_sel, kc_sel;

    assign J4_row[0] = kr_sel[0];
    assign J4_row[1] = kr_sel[1];
    assign J4_row[2] = kr_sel[2];
    assign J4_row[3] = kr_sel[3];
    assign kc_sel[0] = J4_col[0]; 配合外接keyboard之I/O port
    assign kc_sel[1] = J4_col[1];
    assign kc_sel[2] = J4_col[2];
    assign kc_sel[3] = J4_col[3];

    wire clk_key;
    wire [5:0] buf_flag;
    wire [23:0] key_buf_code;
```

# Design 4: keyboard\_top (4/4)

freq_div	# (21)	m0	row scan及資料存入buffer之頻率
kr_scan		m1	
keyboard		m2	rst要反相 -> ~rst
key_buffer		m3	
num_to_seg7_0_F		m4	
num_to_seg7_0_F		m5	
num_to_seg7_0_F		m6	按鍵二進位值對Seg7之編碼
num_to_seg7_0_F		m7	若無資料則Seg7為暗
num_to_seg7_0_F		m8	
num_to_seg7_0_F		m9	

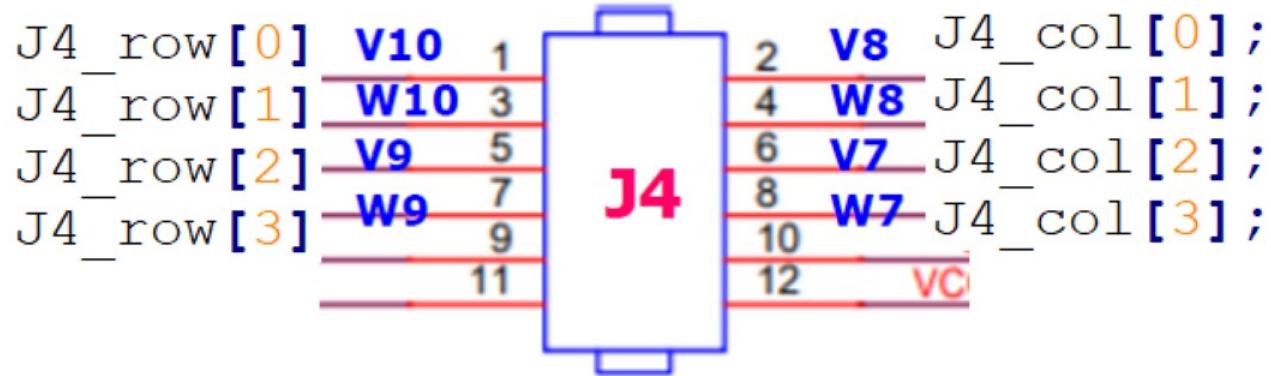
# Pin Assignments (1/2)

**clk**

Clocks: 50 MHZ

CLK1: P11

CLK2: N14



**rst**



KEY0: B8 Push: 0

KEY1: A7 Not push: 1

## Pin Assignments (2/2)

SEG5 SEG4 SEG3 SEG2 SEG1 SEG0



SEG07 SEG06 SEG05 SEG04 SEG03 SEG02 SEG01 SEG00  
**D15 C17 D17 E16 C16 C15 E15 C14**  
SEG17 SEG16 SEG15 SEG14 SEG13 SEG12 SEG11 SEG10  
**A16 B17 A18 A17 B16 E18 D18 C18**  
SEG27 SEG26 SEG25 SEG24 SEG23 SEG22 SEG21 SEG20  
**A19 B22 C22 B21 A21 B19 A20 B20**  
SEG37 SEG36 SEG35 SEG34 SEG33 SEG32 SEG31 SEG30  
**D22 E17 D19 C20 C19 E21 E22 F21**  
SEG47 SEG46 SEG45 SEG44 SEG43 SEG42 SEG41 SEG40  
**F17 F20 F19 H19 J18 E19 E20 F18**  
SEG57 SEG56 SEG55 SEG54 SEG53 SEG52 SEG51 SEG50  
**L19 N20 N19 M20 N18 L18 K20 J20**

press

scan\_code[3:0]

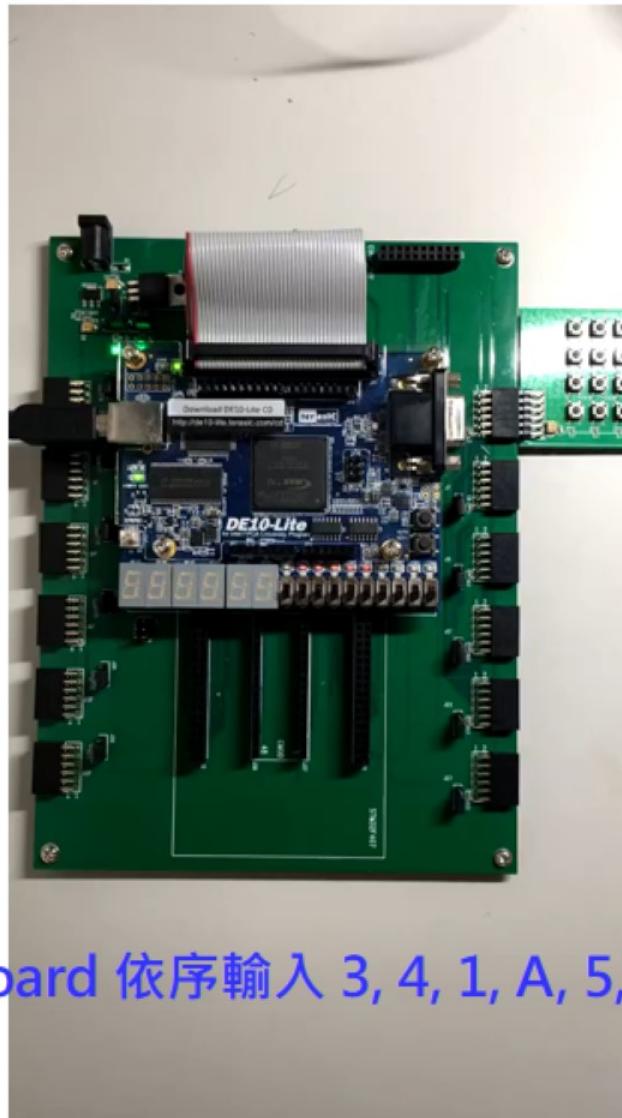


1: LED on  
0: LED off

LED9 LED8 LED7 LED6 LED5 LED4 LED3 LED2 LED1 LED0  
**B11 A11 D14 E14 C13 D13 B10 A10 A9 A8**

# 執行結果

---



# File List

---

## ■ Verilog files

- freq\_div.v (二的次方之除頻電路)
- num\_to\_seg7\_0\_F.v (16進位數字轉Seg7編碼)
- kr\_scan.v (產生列掃描的信號)
- keyboard.v (產生按鍵編碼)
- key\_buffer.v (將按鍵資料送入buffer)
- keyboard\_top.v (最上層top電路)

# 實驗結果驗收

---

- 請老師或助教驗收 keyboard\_top 電路於實驗板顯示之圖形