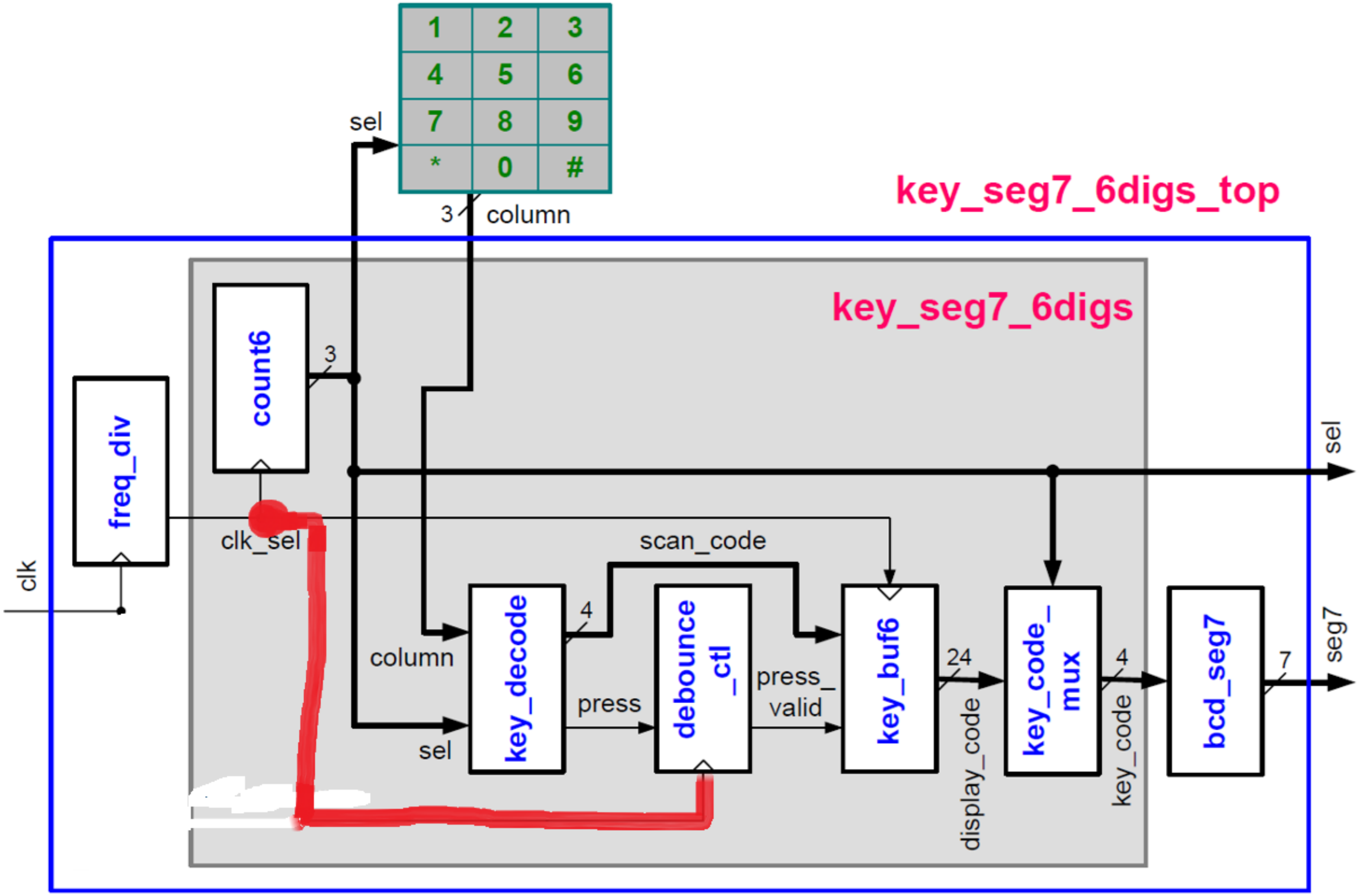
FPGA Lab-06

Keyboard(2)

Content

- Use the keyboard as the input tool to display the keyed-in numbers in order on the six-digit, seven-segment display (the old data is shifted to the left when new data is entered).

System Module



Verilog Codes

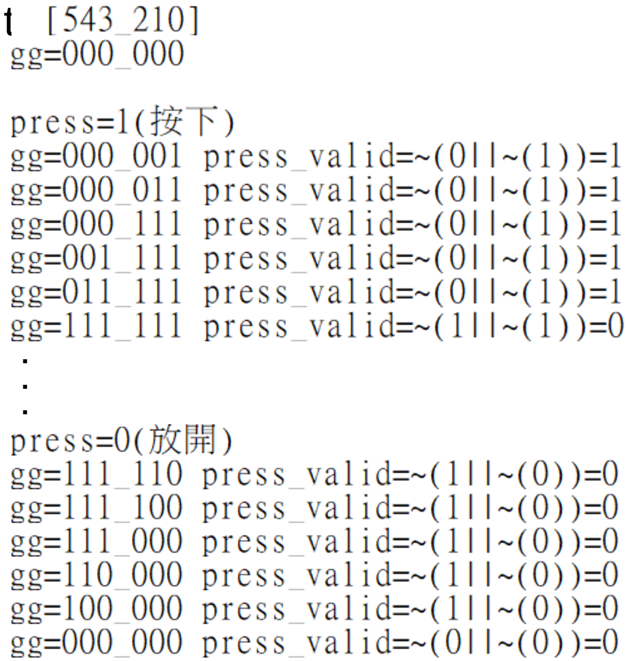
- freq_div.v — COPY FROM PREVIOUS LAB
- count6.v — COPY FROM PREVIOUS LAB
- key_decode.v — COPY FROM PREVIOUS LAB
- debounce_ctl.v
- key_buf6.v
- key_code_mux.v
- key_seg7_6dig.v
- bcd_seg7.v — COPY FROM PREVIOUS LAB
- key_seg7_6dig_top.v

Anti-Bounce Circuitry

- Prevents multiple inputs while manually pressing the keyboard at once.

debounce_ctl.v

```
module debounce_ctl (clk, rst, press, press_valid);
input          press, clk, rst;
output          press_valid;
reg     [5:0]   gg;
//(The number of bits depends on the number of enabling counts, since the seven-
segment display needs to use the count6 in order t
directly set the 6bit.)
assign press_valid = ~(gg[5] || (~press));
always@(posedge clk or posedge rst)
begin
if(rst)
    gg <= 6'b0;
else
    gg <= {gg[4:0], press};
end
endmodule
```

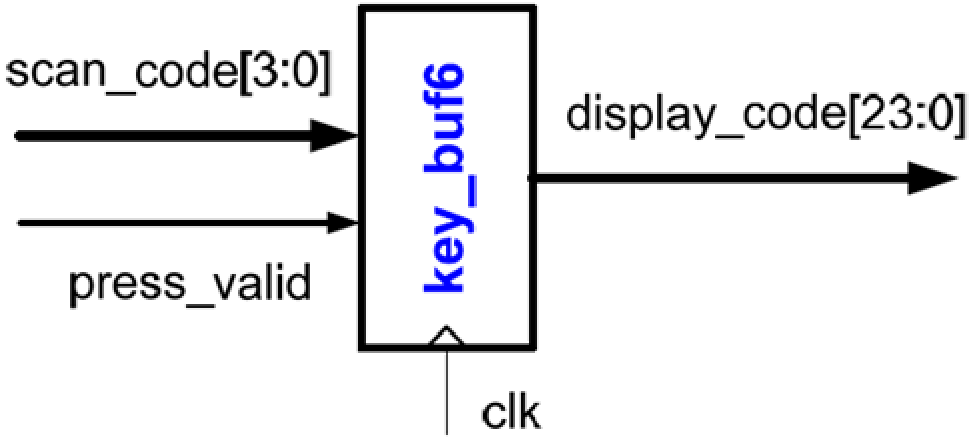


```
[543_210]
gg=000_000

press=1(按下)
gg=000_001 press_valid=~(0||~(1))=1
gg=000_011 press_valid=~(0||~(1))=1
gg=000_111 press_valid=~(0||~(1))=1
gg=001_111 press_valid=~(0||~(1))=1
gg=011_111 press_valid=~(0||~(1))=1
gg=111_111 press_valid=~(1||~(1))=0
 .
 .
 .
press=0(放開)
gg=111_110 press_valid=~(1||~(0))=0
gg=111_100 press_valid=~(1||~(0))=0
gg=111_000 press_valid=~(1||~(0))=0
gg=110_000 press_valid=~(1||~(0))=0
gg=100_000 press_valid=~(1||~(0))=0
gg=000_000 press_valid=~(0||~(0))=0
```

Six Numeric Keyboard Buffers

- Stores the six digits typed by the keyboard.
- Whenever clk is triggered, if press_vaild is 1, it means there is a number typed on the keyboard (scan_code[3:0]). then scan_code will be saved in display_code[23:0] (value will shift left to be save).
- If press_valid is 0 when clk is triggered at the positive edge, it means there is no number keyed into the display_code, the content of display_code remains unchanged.display_code.
- can save up to six numbers, and when it is full, the old data will continue to move left and disappear.

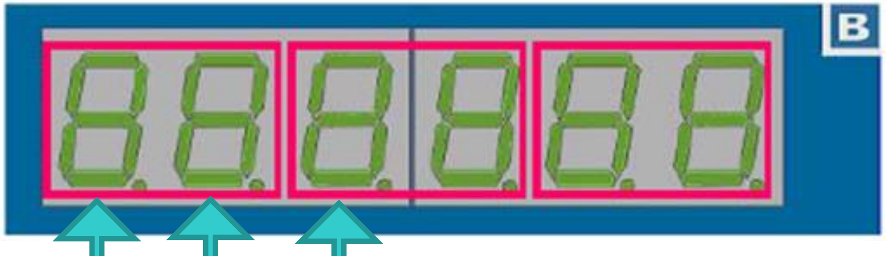


VerilogCode -key_buf6.v

```
module key_buf6(clk, rst, press_valid, scan_code, display_code);
input clk, rst, press_valid;
input[3:0] scan_code;
output[23:0]display_code;
reg[23:0]display_code;
always@(posedge clk or posedge rst) begin
if(rst)
display_code= 24'hffffff;// initial value
else
display_code= press_valid?//{Left_shift_value} :Previous_ value;
end
endmodule
```

Number selection multiplexer

- According to the input control signal sel[2:0], the BCD code of a number in display_code[23:0] is selected and generated in key_code[3:0]
- sel generation order: 000, 001, 010, 011, 100, 101, 000, …
- When sel= 000，choose display_code[23:20]
- When sel= 001，choose display_code[19:16]
- When sel= 010，choose display_code[15:12]
- When sel= 011，choose display_code[11:8]
- When sel= 100，選display_code[7:4]
- When sel= 101, 選display_code[3:0]
- Scanning sequence with 7-segment display



key_code_mux.v

```
module key_code_mux(display_code, sel, key_code);
input[23:0] display_code;
input[2:0]sel;
output[3:0] key_code;
assign key_code= (sel== 3'b101) ? display_code[3:0] :
        (sel== your code) ? display_code[your code] :
        (sel== your code) ? display_code[your code] :
        (sel== your code) ? display_code[your code] :
        (sel== your code) ? display_code[your code] :
        (sel== your code) ? display_code[your code] :     4'b1111;
endmodule
```

Code Integration

- key_seg7_6dig.v
- key_seg7_6dig_top.v

key_seg7_6dig.v

- module key_seg7_6dig(clk_sel, rst, column, sel, key_code);
- input clk_sel, rst;
- input[2:0]column;
- output[2:0]sel;
- output[3:0]key_code;
- wire press, press_valid;

```
wire[3:0] scan_code, key_code;
wire[23:0]display_code;
count6(your code);
key_decode(your code);
debounce_ctl(your code);
key_buf6( your code);
key_code_mux(your code);
endmodule
```

key_seg7_6dig_top.v

```
module key_seg7_6dig_top (clk, rst, column, sel, seg7);
input clk, rst;             //pin W16,C16
input[2:0]column;                      // pin AA13, AB12, Y16
output[2:0]sel;                        // pin AB10, AB11, AA12
output[6:0]seg7;                       // pin AB7,AA7,AB6,AB5,AA9,Y9,AB8
wire clk_sel;
wire[3:0] key_code;
freq_div#(13) (your code);
key_seg7_6dig( your code);
bcd_to_seg7(your code);
endmodule
```