Experiment Report of FPGA

160324　16711094　李翰韬

I

1) 4-bit Binary Addition Counters

Logical analysis: input signal CLK clock, RST reset control, output signal P3, P2, P1, P0.

Interface signal and target device pin (binding) connection: clk→P20、rst→U19（F5）；p3→W13（LED4）、p2→W15（LED5）、p1→Y17（LED6）、p0→R16（LED7）。

Download the burning configuration file format:*.sof。

Experimental verification operation:

(1) The input signal CLK clock connecting the FPGA_EA2_p6 (P20) with a wire (FRQ_Q21 1Hz), the corresponding position of RST U19 (F5) is the fifth on the upper right of the "single pulse" in the lower right part of the experimental box; the output signal P3 W13 (LED 4), P2 W15 (LED 5), P1 Y17 (LED 6), P0 R16 (LED 7) shows the position in the experimental box. The middle and the bottom left and lower corner fourth, fifth, sixth or seven light-emitting diodes.

(2) Set the input signal RST to "1". The CLK clock (FRQ_Q21 1Hz) is connected to the P20 by a wire at a frequency of 1Hz. The output information is four light-emitting diodes according to 0000→0001→……→1111 cycle display. Due to the low efficiency of LED, the actual display value is contrary to theory, and the code needs to be corrected.

VHDL：

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
entity count4 is
port(
clk:in std_logic;--in bit
rst:in std_logic;
p:out std_logic_vector(3 downto 0));
end ;
architecture b1 of count4 is
signal p1:std_logic_vector(3 downto 0);
begin
process(rst,clk)
begin
if(clk'event and clk='1')then
p1<=p1+1;
end if;
end process;
p<=p1;
end;
```

2) 1-bit half adder circuit

Logical analysis: input signal a, B; output signal sum (and), carry (carry).

Logical equation: sum=a B; carry=a*b

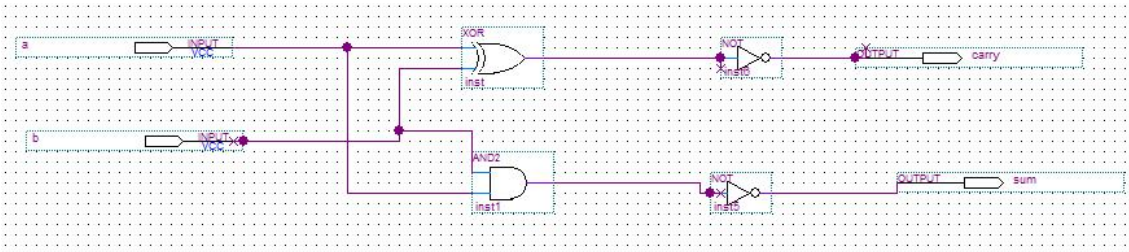
Interface signal and target device pin (binding) connection: input signal a N18 (SW-1), B M20 (SW-2); output signal sum (and) W13 (LED 4), carry (carry) V12 (LED 2).

Download the burning configuration file format: *.sof.

Experimental verification operation:

(1) The corresponding positions of input signals a N18 (SW-1) and B M20 (SW-2) start the first and second at the lower left of the "logic switch group" in the lower right part of the experimental box, and the output signals sum (and) W13 (LED 4), carry (carry) V12 (LED 2) display positions are the fourth and second luminescence at the bottom left corner of the middle part of the experimental box. Diode.

(2) The input signals a and B are all "1", the output result information is the second LED "light"carry (carry), the fourth LED "extinguish"sum (sum); the input signals a and B are changed in turn, and the correct result can be obtained.



II

1) Frequency divider

It is replaced by the frequency division function used in subsequent comprehensive experiments.

2) Binary counter

Replaced by a more sophisticated two bit decimal counter in the IV.

3) 16 x 16 dot matrix display AB

Replaced by VII

4) 4 x 4 keyboard scan display key

Replaced by VI

5) Character type LCD

Replaced by IX

III

1) NAND gate circuit

Logical analysis: input signal a, B; output signal f

Logical equation: f=$\overline{a \cdot b}$

Interface signal and target device pin (binding) connection: input signal a→N18(SW-1), B→M20 (SW-2); output signal f→V12(LED 2).

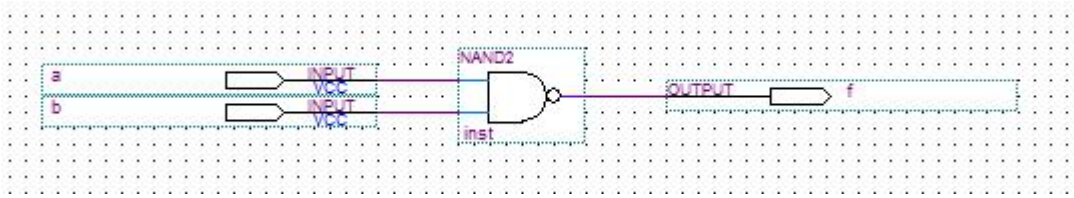
Download the burning configuration file format: *.sof.

Experimental verification operation:

(1) The corresponding positions of the input signals a→N18 (SW-1) and b→M20 (SW-2) start the first and second at the lower left of the "logic switch group" in the lower right part of the experimental box, and the output signals f→V12 (LED 2) display the position of the second light-emitting diode at the bottom left of the middle part of the experimental box.

(2) The input signals a and B are all "1", and the output result information is "out" of the second LED; when at least one of the input signals a and B is "0", the output result information is "light" of the second LED.

(the experimental procedure did not add non gate before the output, so the experimental phenomenon is contrary to the theoretical phenomenon in the report. Non gate correction should be added.)



2) Combination circuit 3-8 decoder

Logical analysis: input signal a, B, C; output signal Y7, y6, Y5, Y4, Y3, Y2, Y1, y0.

Interface signal and target device pin (binding) connection: input signal a→AA15（SW-3）、b→M20（SW-2）、c→N18（SW-1）；y7→U12（LED1），y6→V12（LED2），y5→V15（LED3），y4→W13（LED4），y3→W15（LED5），y2→Y17（LED6），y1→R16（LED7），y0→T17（LED8）.

Download the burning configuration file format: *.sof.

Experimental verification operation:

(1) The corresponding positions of the input signals a→AA15（SW-3）、b→M20（SW-2）、c→N18（SW-1） start at the first, second and third positions at the lower left of the "logic switch group" in the lower right part of the experimental box, and the output signals y7→U12（LED1），y6→V12（LED2），y5→V15（LED3），y4→W13（LED4），y3→W15（LED5），y2→Y17（LED6），

y1→R16（LED7），y0→T17（LED8）display the position of the first to eight light-emitting diodes in the bottom left-hand corner of the middle of the experimental box.

(2) when the initial CBA state is "000", the first seven diodes are all bright. When the input signals are changed to "001", "010", "011", "100", "101", "110" and "111" respectively, the seventh to the first light-emitting diodes are extinguished in turn, and the remaining light-emitting diodes remain "bright" state. The observation shows that the information meets the requirements.

VHDL：

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
entity t32 is
port(
a,b,c:in std_logic;
y:out std_logic_vector(7 downto 0)
);
end;

architecture te of t32 is
signal temp:std_logic_vector(2 downto 0)
signal tt:std_logic_vector(7 downto 0);

begin
temp<=c&b&a;
process(temp)
begin
case temp is
when "000" =>tt<="00000001";
when "001" =>tt<="00000010";
when "010" =>tt<="00000100";
when "011" =>tt<="00001000";
when "100" =>tt<="00010000";
when "101" =>tt<="00100000";
when "110" =>tt<="01000000";
when "111" =>tt<="10000000";
end case;
end process;
y<=tt;
end te;
```

3) Sequential circuit D trigger

Logical analysis: input signal CLK clock, RST reset control, D; output signal Q.

Interface signal and target device pin (binding) connection: input signal CLK clock selection (FRQ_Q15 64Hz) with a wire and FPGA_EA2_p6 connected to the clk→P20、rst→AA15（SW-3）、d→N18（SW-1）；q→V12（LED2）).

Download the burning configuration file format: *.sof.

Experimental verification operation:

(1) The input signal CLK clock connecting the FPGA_EA2_p6 (P20) with a wire (FRQ_Q15 64Hz), RST clearing AA15 (SW-3) corresponding position in the lower right part of the experimental box "logical switch group" the third left lower side, the corresponding position of D N18 (SW-1) in the lower right part of the experimental box "logical switch group" the first left lower side; The signal Q to V12 (LED2) shows the location of the second light emitting diodes in the bottom left corner of the test box.

(2) Set the input signal RST as "1". The CLK clock (FRQ_Q15 64Hz) is connected with the P20 by wire at 64Hz. In theory, the switch D is set to 1 when the light is on and 0 when the light is off. (because the diode low level active code needs to be modified, the actual result of the program is contrary to theory).

VHDL：

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
entity t33 is
port(clk:in std_logic;
        rst:in std_logic;
        d:in std_logic;
        q:out std_logic);
end;
architecture nake of t33 is
        signal q1:std_logic;
           begin
              process(rst,clk)
                 begin
                    if(clk'event and clk='1')then
                       q1<=d;
                    end if;
              end process;
             q<=q1;
End;
```

IV
Two -bit decimal counter

Logic analysis: input signal CLK clock, clk_dis clock, RST zero clearing control; output signal choose1, choose0, d7, d6, d5, d4, d3, d2, d1, d0.

Interface signal and target device pin (binding) connection: clk→P20、clk_dis→W14、rst→N18（SW-1）; choose1→Y21、choose0→Y22、d7→AA20、d6→W20、d5→R21、d4→P21、d3→N21、d2→N20、d1→M21、d0→M19.

Download the burning configuration file format: *.sof.

Experimental verification operation:

(1) Input signal CLK clock connect FPGA_EA2_p6 (P20) with wire (FRQ_Q18 8Hz), clk_dis clock connect FPGA_EA2_p7 (W14) with wire (FRQ_Q15 64Hz), RST clearing N18 (SW-1) corresponding position in the lower right part of the experimental box "logic switch group" the first one on the left; output signal chooses corresponding position Second, third digital bit selection, d7~d0 corresponding 8 segment driver. The dial switch is 00XX at this time.

(2) Set the input signal RST to "1". The CLK clock (FRQ_Q18 8Hz) is connected with P20 by wire at 8Hz. The clk_dis clock (FRQ_Q15 64Hz) is connected with W14 by wire at 64Hz. The output information of the second and third two digital tubes is in accordance with 00→01→02→03……→98→99 cycle display. The information is satisfied by observation.

VHDL:

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
entity t40 is
      port(
            clk:in std_logic;
            clk_dis:in std_logic;
            rst:in std_logic;
            choose:out std_logic_vector(1 downto 0);
            d:out std_logic_vector(7 downto 0));
            end;
architecture b1 of t40 is
      signal q1,q2:std_logic_vector(3 downto 0)
      signal dis:std_logic_vector(3 downto 0);
      signal dis_choose:std_logic_vector(1 downto 0);
      signal co:std_logic;
      begin
            process(clk,rst)
                  begin
                   if rst='0' then
                      q1<="0000";
                      elsif(clk'event and clk='1')then--clk
                         if q1<9 then
                            q1<=q1+1;
```

```
                else q1<="0000";
                end if;
                if q1=9 then
                    co<='1';
                else
                    co<='0';
                end if;
          end if;

end process;
process(clk,rst)
         begin
          if rst='0' then
              q2<="0000";
              elsif(clk'event and clk='1'and co ='1')then
                  if q2<9 then
                      q2<=q2+1;
                  else q2<="0000";
                  end if;
          end if;
end process;
process(clk_dis)
        begin
        if(clk_dis'event and clk_dis='1')then
              if(dis_choose="10") then
                    dis_choose<="01";
                    dis<=q1;
              elsif(dis_choose="01") then
                    dis_choose<="10"
                    dis<=q2;
              else dis_choose<="01";
              end if;
        end if;
        choose<=dis_choose;
end process;

    process(dis)
    begin
        if(rst='1')then
          case dis is
              when "0000"=>d<="11111100";
              when "0001"=>d<="01100000";
              when "0010"=>d<="11011010";
              when "0011"=>d<="11110010";
```

```
                when "0100"=>d<="01100110";
                when "0101"=>d<="10110110";
                when "0110"=>d<="10111110";
                when "0111"=>d<="11100000";
                when "1000"=>d<="11111110";
                when "1001"=>d<="11110110";
                when "1010"=>d<="11101110";
                when "1011"=>d<="00111110";
                when "1100"=>d<="10011100";
                when "1101"=>d<="01111010";
                when "1110"=>d<="10011110";
                when "1111"=>d<="10001110";
            end case;
        end if;
    end process;
end b1;
```

V
Design of Lanterns

Logical analysis: input signal CLK clock, Q7, Q6, Q5, Q4, Q3, Q2, Q1, Q0, RST.

Interface signals are connected to target device pins (binding):clk→P20、q7→U12、q6→V12、q5→V15、q4→W13、q3→W15、q2→Y17、q1→R16、q0→T17、rst→N18.

Download the burning configuration file format: *.sof.

Experimental verification operation:

At the beginning of the experiment, except for the leftmost LED lights, a row of LED monitors on the other experimental boxes were all lit up; then, with the frequency of the CLK trigger signal, each trigger pulse moved one bit to the right of the extinguished light (one on the right of the original extinguished light was extinguished), and the original extinguished light was re-lit; when the rightmost light was extinguished After that, the next trigger pulse will reset the cycle, that is, the left most of the lights will go off, and the rest of the lights will be on.

Visually, a black spot moves one bit at a time from the left to the right. When it reaches the right terminal, the black spot returns to the left and enters the next cycle.

VHDL:

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
entity led is
port(clk:in std_logic;
rst:in std_logic;
q :out std_logic_vector(7 downto 0));
end;
architecture led of led is
constant s0:std_logic_vector(1 downto 0):="00";
constant s1:std_logic_vector(1 downto 0):="01";
constant s2:std_logic_vector(1 downto 0):="10";
constant s3:std_logic_vector(1 downto 0):="11";
signal present:std_logic_vector(1 downto 0);
signal q1:std_logic_vector(7 downto 0);
signal count:std_logic_vector(3 downto 0);
begin
process(rst,clk)
begin
if(rst='0')then
present<=s0;
q1<=(others=>'0');
elsif(clk'event and clk='1')then
```

```
case present is
when s0 => if(q1="00000000")then
q1<="10000000";
else if(count="0111")then
count<=(others=>'0');
q1<="00000001";
present<=s1;
else q1<=q1(0) & q1(7 downto 1);
count<=count+1;
present<=s0;
end if;
end if;
when s1 => if(count="0111")then
count<=(others=>'0');
q1<="10000001";
present<=s2;
else q1<=q1(6 downto 0) & q1(7);
count<=count+1;
present<=s1;
end if;
when s2 => if(count="0011")then
count<=(others=>'0');
q1<="00011000";
present<=s3;
else q1(7 downto 4)<=q1(4) & q1(7 downto 5);
q1(3 downto 0)<=q1(2 downto 0) & q1(3);
count<=count+1;
present<=s2;
end if;
when s3 => if(count="0011")then
count<=(others=>'0');
q1<="10000000";
present<=s0;
else q1(7 downto 4)<=q1(6 downto 4) & q1(7);
q1(3 downto 0)<=q1(0) & q1(3 downto 1);
count<=count+1;
present<=s3;
end if;
end case;
end if;
end process;
q<=q1;
end;
```

VI
4 x 4 keyboard display keys

Logic analysis: input signal CLK clock, start, KBCol3, KBCol2, KBCol1, KBCol0; output signal KBRow3, KBRow2, KBRow1, KBRow0, seg77, seg76, seg75, seg74, seg73, seg72, seg71, seg70, scan7, scan6, scan5, scan4, scan3, scan2, scan1, scan0.

Interface signal and target device pin (binding) connection: clk→P20、start→AA15（SW-3）、KBCol3→A13（SWC3）、KBCol2→F9（SWC2）、KBCol1→D10（SWC1）、KBCol0→B10（SWC0）；KBRow3→C4（SWR3）、KBRow2→A16（SWR2）、KBRow1→A15（SWR1）、KBRow0→A14（SWR0）、scan7→AB20（1）、scan6→Y21（2）、scan5→Y22（3）、scan4→W22（4）、scan3→V22（5）、scan2→U22（6）、scan1→AA17（7）、scan0→V16（8）、seg77→AA20（A）、seg76→W20（B）、seg75→R21（C）、seg74→P21（D）、seg73→N21（E）、seg72→N20（F）、seg71→M21（G）、seg70→M19（H）.

Download the burning configuration file format: *.sof.

Experimental verification operation:

(1) The input signal CLK clock connecting the FPGA_EA2_p6 (P20) with a wire (FRQ_Q11 1024 Hz), starting AA15 (SW-3) corresponding position in the lower right part of the experimental box "logic switch group" left third; the output signal seg77 AA20 (A section), seg76 W20 (B section), seg75 R21 (C section), seg74 P21 (D section) The display position of seg73 N21 (E), seg72 N20 (F), seg71 M21 (G), seg70 M19 (H) corresponds to a digital tube. The dial switch is 10XX at this time.

(2) Set the input signal start to "1". The CLK clock (FRQ_Q11 1024 Hz) is connected with the P20 by wire at 1024 Hz. Press any key of the 4X4 keyboard and display the corresponding number or letter on the digital tube. The observation shows that the information meets the requirements.

VHDL：

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
use ieee.std_logic_arith.all;
entity t60 is
port(clk:in std_logic;
        start:in std_logic;
        KBCol:in std_logic_vector(3 downto 0);
        KBRow:out std_logic_vector(3 downto 0);
        seg7:out std_logic_vector(7 downto 0);
        scan:out std_logic_vector(7 downto 0));
end t60;
architecture bev of t60 is
      signal count:std_logic_vector(1 downto 0);
      signal sta:std_logic_vector(1 downto 0);
      begin
         scan<="11111110";
   a:
       process(clk)
            begin
```

```
        if(clk'event and clk='1')then
                count<=count+1;
        end if;
    end process a;
b:
    process(clk)
        begin
        if(clk'event and clk='1')then
            case count(1 downto 0) is
                when "00"=>KBRow<="0111";
                        sta<="00";
                when "01"=>KBRow<="1011";
                        sta<="01";
                when "10"=>KBRow<="1101";
                        sta<="10";
                when "11"=>KBRow<="1110";
                        sta<="11";
                when others=>KBRow<="1111";
            end case;
        end if;
    end process b;
c:
    process(clk,start)
        begin
        if start='0' then
            seg7<="00000000";
            elsif(clk'event and clk='1')then
              case sta is
                when "00"=>
                case KBCol is
                  when "1110"=>seg7<="10011100";
                  when "1101"=>seg7<="01111010";
                  when "1011"=>seg7<="10011110";
                  when "0111"=>seg7<="10001110";
                  when others=>seg7<="00000000";
                end case;
                when "01"=>
                case KBCol is
                  when "1110"=>seg7<="11111110";
                  when "1101"=>seg7<="11100110";
                  when "1011"=>seg7<="11101110";
                  when "0111"=>seg7<="00111110";
                  when others=>seg7<="00000000";
                end case;
```

```
                when "10"=>
                case KBCol is
                   when "1110"=>seg7<="01100110";
                   when "1101"=>seg7<="10110110";
                   when "1011"=>seg7<="10111110";
                   when "0111"=>seg7<="11100000";
                   when others=>seg7<="00000000";
                end case;
                when "11"=>
                case KBCol is
                   when "1110"=>seg7<="11111100";
                   when "1101"=>seg7<="01100000";
                   when "1011"=>seg7<="11011010";
                   when "0111"=>seg7<="11110010";
                   when others=>seg7<="00000000";
                end case;
                when others=>seg7<="00000000";
             end case;
          end if;
       end process c;
end bev;
```

VII
16 x 16 dot matrix display AB

Logic analysis: input signal CLK clock, mode, RST zero; output signal row15, row14, row13, row12, row11, row10, row9, row8, row7, row6, row5, row4, row3, row1, row0, col3, col2, col1, col0.

Interface signal and target device pin (binding) connection: clk→P20、rst→AA15（SW-3）、mode→N18（SW-1）；row15→A13、row14→F9、row13→D10、row12→B10、row11→B9、row10→B8、row9→B7、row8→E14、row7→C15、row6→F11、row5→C13、row4→E11、row3→B6、row2→A6、row1→A5、row0→A4、col3→C4、col2→A16、col1→A15、col1→A14.

Download the burning configuration file format: *.sof.

Experimental verification operation:

(1) The input signal CLK clock connecting the FPGA_EA2_p6 (P20) with a wire (FRQ_Q11 1024 Hz), the corresponding position of RST AA15 (SW-3) in the lower right part of the experimental box is the third on the left lower side of the "logic switch group", and the corresponding position of mode N18 (SW-1) in the lower right part of the experimental box is the first on the left of the "logic switch group"; The display is located at 16 * 16LED dot matrix. Each column of 16 LEDs from left to right corresponds to one of the "XXXX" codes, from bottom to top four sets of corresponding "XXXX" in four bits, X's 0 ~ F value is the display value of each set of lights. The dial switch is 00XX.

(2) Set the input signal RST to "1". Set the mode to "0". Set the input signal start to "1". The CLK clock (FRQ_Q11 1024 Hz) is connected to the P20 with a wire frequency of 1024 Hz. The output phenomenon is that the upper half of the 16 *16 LED lattice shows an upside-down AB.

VHDL:

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

entity t70 is
        generic(n:integer:=72);
        port(clk,mode,rst:in std_logic;--mode n18 sw1 --rst ab15 F1
                    row:out std_logic_vector(15 downto 0);
                    col:out std_logic_vector(3 downto 0));
end t70;
architecture behave of t70 is
      type code is array(0 to n-1)of std_logic_vector(15 downto 0);
      constant

code_0:code:=(x"0003",x"000c",x"0030",x"0090",x"0090",x"0030",x"000c",x"0003",x"0000",x"00ff
",x"0091",x"0091",x"0091",x"00aa",x"0044",x"0000",x"0000",x"0000",x"0000",x"0000",x"0000",x"
0000",x"0000",x"0000",x"0000",x"0000",x"0000",x"0000",x"0000",x"0000",x"0000",x"0000",x"000
0",x"0000",x"0000",x"0000"x"0000",x"0000",x"0000",x"0000",x"0000",x"0000",x"0000",x"0000",x
"0000",x"0000",x"0000",x"0000",x"0000",x"0000",x"0000",x"0000",x"0000",x"0000"x"0000",x"00
00",x"0000",x"0000",x"0000",x"0000",x"0000",x"0000",x"0000",x"0000",x"0000",x"0000",x"0000",
x"0000",x"0000",x"0000",x"0000",x"0000");
```

```
signal cntscan,frame:std_logic_vector(3 downto 0);
signal i,j,f:integer range 0 to n-1;
signal cnt,cnt1:integer range 0 to 20;
begin
    process(clk,frame,rst)
    begin
      if rst='0'then
          cntscan<="0000";
          frame<="0000";
          i<=0;j<=0;cnt<=0;cnt1<=0;
          row<=x"0000";
          elsif clk'event and clk='1'then
                  if mode='0'then--mode
                    if f=4 then
                        f<=0;
                        j<=0;
                      else
                          row<=code_0(conv_integer(cntscan)+j);
                          col<=cntscan;
                          if cntscan="1111" then
                              cntscan<="0000";
                                cnt1<=cnt1+1;
                                  else
                                  cntscan<=cntscan+1;
                          end if;
                          if cnt1=30 then
                              j<=j+18;
                              f<=f+1;
                                cnt1<=0;
                          end if;
                      end if;
                  else
                      col<=frame;
                      case frame is
                            when"0000"=>row<=code_0((i)mod n);
                            when"0001"=>row<=code_0((i+1)mod n);
                            when"0010"=>row<=code_0((i+2)mod n);
                            when"0011"=>row<=code_0((i+3)mod n);
                            when"0100"=>row<=code_0((i+4)mod n);
                            when"0101"=>row<=code_0((i+5)mod n);
                            when"0110"=>row<=code_0((i+6)mod n);
                            when"0111"=>row<=code_0((i+7)mod n);
                            when"1000"=>row<=code_0((i+8)mod n);
                            when"1001"=>row<=code_0((i+9)mod n);
```

```
                        when"1010"=>row<=code_0((i+10)mod n);
                        when"1011"=>row<=code_0((i+11)mod n);
                        when"1100"=>row<=code_0((i+12)mod n);
                        when"1101"=>row<=code_0((i+13)mod n);
                        when"1110"=>row<=code_0((i+14)mod n);
                        when"1111"=>row<=code_0((i+15)mod n);
                        i<=i+1;
                        cnt<=cnt+1;
                        frame<="0000";
                        when others=>
                                null;
                    end case;
                if i=n-1 then
                    i<=0;
                    else
                     if cnt=10 then
                            i<=i+1;
                            cnt<=0;
                     end if;
                end if;
            frame<=frame+1;
        end if;
        end if;
        end process;
end behave;
```

VIII
Multi-player responder

Logic analysis: input signal CLK clock, rst, light3, light2, light1, light0, s3, s2, s1, s0, states3, states2, states1, states0, warm.

Interface signal and target device pin (binding) connection: clk→P20 、 light3→W13 、 light2→V15、light1→V12、light0→U12、s3→F7、s2→E6、s1→C7、s0→E5、rst→C3.

Download the burning configuration file format: *.sof.

Experimental verification operation:

After starting, press zero key first, counter starts countdown. There are four switches, the first switch is pressed, the corresponding diode will be bright and remain bright. When the other switches are pressed or not, the corresponding diodes are not bright.

VHDL:

Answer module:

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
entity xuanshou is
        port(rst,clk2:in std_logic;
             s0,s1,s2,s3:in std_logic;
             states:buffer std_logic_vector(3 downto 0);
             light:buffer std_logic_vector(3 downto 0);
warm:out std_logic);
end xuanshou ;
architecture one of xuanshou      is
signal st:std_logic_vector(3 downto 0);
begin
p1:process(s0,rst,s1,s2,s3,clk2)
   begin
      if rst='0'    then
         warm<='0';st<="0000";
         elsif clk2'event and clk2='1' then
    if (s0='1' or st(0)='1')and not( st(1)='1' or st(2)='1' or st(3)='1' )
             then st(0)<='1';
    end if ;
 if (s1='1' or st(1)='1')and not( st(0)='1' or st(2)='1' or st(3)='1' )
             then st(1)<='1';
```

```
    end if ;
  if (s2='1' or st(2)='1')and not( st(0)='1' or st(1)='1' or st(3)='1' )
                then st(2)<='1';
      end if ;
  if (s3='1' or st(3)='1')and not( st(0)='1' or st(1)='1' or st(2)='1' )
                then st(3)<='1';
      end if ;
warm<=st(0) or st(1) or st(2) or st(3);
end if ;
end process p1;
p2:process(states(0),states(1),states(2),states(3),light)
  begin
  if (st="0000") then states<="0000";
elsif (st<="0001") then states<="0001";
elsif (st<="0010") then states<="0010";
elsif (st<="0100") then states<="0011";
elsif (st<="1000") then states<="0100";
end if;
light<=st;
end process p2;
end one;
```

Counting module：

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
entity JS is
   port(clk1,rst,start,stop:in std_logic;
           ta,tb:buffer std_logic_vector(3 downto 0));
end JS;
architecture one of JS is
signal co:std_logic;
begin
p1:process(clk1,rst,start,stop,ta)
      begin
```

```
        if rst='0' or stop='1' then
             ta<="0000";
        elsif clk1'event and clk1='1' then
             co<='0';
           if start='1' then
                if ta="0000" then
                ta<="1001";co<='1';
             else ta<=ta-1;
             end if;
           end if;
        end if;
end process p1;
p2:process(co,rst,start,stop,tb)
    begin
      if rst='0' or stop='1' then
       tb<="0010";
      elsif co'event and co='1' then
           if start='1' then
             if tb="0000" then tb<="0011";
             else tb<=tb-1;
             end if;
          end if;
      end if;
end process p2;
end one ;
```

IX
1602LCD display information

Logic analysis: input signal CLK clock, reset, oe, rs, rw, data7, data6, data5, data4, data3, data2, data1, data0.

Interface signals are connected to target device pins (bindings): clk→P20 、 data7→A3 、 data6→F7、data5→E6、data4→C7、data3→E5、data2→C3、data1→AB18、data0→AB17、oe→A4、rs→A6、 rw→A5.

Download the burning configuration file format: *.sof.

Verification of the experiment:

After the start, press the zero key, LCD refresh, the top row shows the clock count, the next row shows "www.BUAA.edu.cn".

VHDL:

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;
entity test is
generic (N:integer:=200;
delay:integer:=100);
port (clk:in std_logic;
      reset: in std_logic;
      oe: out std_logic;
      rs: out std_logic;
      rw: out std_logic;
      data: out std_logic_vector(7 downto 0));
end;
architecture behavioral of test is
type state is
(clear_lcd, entry_set, display_set, funtion_set, position_set1, write_data1, position_set2, write_data2,stop);
signal current_state: state:=clear_lcd;
type ram is array(0 to 23) of std_logic_vector(7 downto 0);
signal dataram: ram:=(("00110000"), ("00110000"), ("00111010"), ("00110000"),
      ("00110000"), ("00111010"), ("00110000"), ("00110000"),
      x"77", x"77", x"77", x"2E", x"42", x"55", x"41", x"41", x"2E", x"65", x"64", x"75", x"2E", x"63", x"6e", x"80");
signal clk_250Khz,clk_1Hz:std_logic;
signal cnt1, cnt2: integer range 0 to 200000;
signal
      hour_h_tmp, hour_l_tmp, min_h_tmp, min_l_tmp, sec_h_tmp, sec_l_tmp: std_logic_vector(3 downto 0):="0000";
begin
lcd_clk:
```

```
        process(clk, reset)
                variable c1: integer range 0 to 100;
                variable c2: integer range 0 to 50000000;
                variable clk0, clk1:std_logic;
        begin
if(reset='0')then
        c1:=0; c2:=0;
elsif (clk'event and clk='1') then
If (c1=N/2-1) then
        c1:=0;
        clk0:=not clk0;
else
        c1:=c1+1;
end if;
        if c2=50000000/2-1 then
                c2:=0;
                clk1:=not clk1;
        else
                c2:=c2+1;
        end if;
end if;
        clk_250Khz<=clk0;
        clk_1hz<=clk1;
end process;
write:
        process(clk_250Khz, reset)
        begin
        if (clk_250Khz'event and clk_250Khz='1') then
        dataram(0)<="0011" & hour_h_tmp;
        dataram(1)<="0011" & hour_l_tmp;
        dataram(3)<="0011" & min_h_tmp;
        dataram(4)<="0011" & min_l_tmp;
        dataram(6)<="0011" & sec_h_tmp;
        dataram(7)<="0011" & sec_l_tmp;
end if;
end process;
control:
        process(clk_250Khz, reset)
        --variable cnt3: std_logic_vector (3 downto 0);
        begin
        if (reset='0') then
        current_state<=clear_lcd;
        cnt1<=0; cnt2<=0;
        elsif rising_edge(clk_250Khz) then
```

```
            case current_state is
        when clear_lcd=>
        oe<='1';
        rs<='0';
        rw<='0';
        data<=x"01";
        cnt1<=cnt1+1;
if cnt1>delay*1 and cnt1<=delay*6 then
        oe<='0';
else
        oe<='1';
end if;
if cnt1=delay*7 then
        current_state<=entry_set;
        cnt1<=0;
end if;
when entry_set=>
            oe<='1';
            rs<='0';
            rw<='0';
            data<=x"06";
            cnt1<=cnt1+1;
if (cnt1>delay*1 and cnt1<=delay*2 )then
        oe<='0';
else
        oe<='1';
end if;
if (cnt1=delay*3 )then
        current_state<=display_set;
        cnt1<=0;
end if;
        when display_set=>
        oe<='1';
        rs<='0';
        rw<='0';
        data<=x"0C";
        cnt1<=cnt1+1;
if (cnt1>delay and cnt1<=delay*2 )then
        oe<='0';
else
        oe<='1';
end if;
if (cnt1=delay*3) then
        current_state<=funtion_set;
```

```
        cnt1<=0;
end if;
when funtion_set=>
        oe<='1';
        rs<='0';
        rw<='0';
        data<=x"38";
        cnt1<=cnt1+1;
if (cnt1>delay and cnt1<=delay*2 )then
        oe<='0';
else
        oe<='1';
end if;
if (cnt1=delay*3) then
        current_state<=position_set1;
        cnt1<=0;
end if;
when position_set1=>
        oe<='1';
        rs<='0';
        rw<='0';
        data<=x"38";
        cnt1<=cnt1+1;
if (cnt1>delay and cnt1<=delay*2 )then
        oe<='0';
else
        oe<='1';
end if;
if (cnt1=delay*3 )then
        current_state<=write_data1;
        cnt1<=0;
end if;
when write_data1=>
        oe<='1';
        rs<='1';
        rw<='0';
if cnt2<=7 then
        data<=datarам(cnt2);
        cnt1<=cnt1+1;
if (cnt1>delay and cnt1<=delay*2) then
        oe<='0';
else
        oe<='1';
end if;
```

```
if (cnt1=delay*3) then
        current_state<=write_data1;
        cnt1<=0;
        cnt2<=cnt2+1;
end if;
else
        cnt2<=0;
        current_state<=position_set2;
end if;
when position_set2=>
        oe<='1';
        rs<='0';
        rw<='0';
        data<=x"c0";
        cnt1<=cnt1+1;
if (cnt1>delay and cnt1<=delay*2) then
        oe<='0';
else
        oe<='1';
end if;
if (cnt1=delay*3) then
        current_state<=write_data2;
        cnt1<=0;
end if;
        when write_data2=>
        oe<='1';
        rs<='1';
        rw<='0';
if cnt2<=15 then
data<=dataram (cnt2+8);
cnt1<=cnt1+1;
if( cnt1>delay and cnt1<=delay*2 )then
        oe<='0';
else
        oe<='1';
end if;
if (cnt1=delay*3 )then
        current_state<=write_data2;
        cnt1<=0;
        cnt2<=cnt2+1;
end if;
else
cnt2<=0;
current_state<=position_set1;
```

```
end if;
When stop=>
null;
end case;
end if;
end process;
clock:
process (clk_1hz, reset)
        begin
if reset='0' then
        hour_h_tmp<="0000";
        hour_l_tmp<="0000";
        min_h_tmp<="0000";
        min_l_tmp<="0000";
        sec_h_tmp<="0000";
        sec_l_tmp<="0000";
elsif (clk_1hz'event and clk_1hz='1')then
        if sec_l_tmp="1001" then
                sec_l_tmp<="0000";
        if sec_h_tmp="0101" then
                sec_h_tmp<="0000";
                if min_l_tmp="1001" then
                min_l_tmp<="0000";
                if min_h_tmp="0101" then
                min_h_tmp<="0000";
                if hour_h_tmp="0010" then
                if hour_l_tmp="0011" then
                hour_l_tmp<="0000";
                hour_h_tmp<="0000";
else
        hour_l_tmp<=hour_l_tmp+'1';
end if;
        else
        if hour_l_tmp="1001"then
        hour_l_tmp<="0000";
        hour_h_tmp<=hour_h_tmp+'1';
        else
        hour_l_tmp<=hour_l_tmp+'1';
        end if;
        end if;
else
min_h_tmp<=min_h_tmp+'1';
end if;
else
```

```
min_l_tmp<=min_l_tmp+'1';
end if;
else
sec_h_tmp<=sec_h_tmp+'1';
end if;
else
sec_l_tmp<=sec_l_tmp+'1';
end if;
end if;
end process;
end behavioral;
```

The conclusion of FPGA experiment

Field programmable gate array, or FPGA, is the product of further development on the basis of programmable devices such as PAL, GAL, CPLD. As a semi-customized circuit in the field of application specific integrated circuits (ASIC), it not only solves the shortcomings of customized circuits, but also overcomes the shortcomings of the limited number of gates in the original programmable devices.

The FPGA experiments in these two sessions gave me a preliminary understanding of the general design process of FPGA. From ignorance of the FPGA at the beginning to continuous access to relevant information and experimental instructions, step by step in accordance with the tutorial to learn VHDL syntax, programming simulation, pin distribution, the program downloaded to the experimental board to observe the phenomenon, complete a complete FPGA program design and debugging process. At the very beginning, nature will not be plain sailing. In particular, there are various problems in program debugging and pin allocation. For example, when writing programs, because of the rigor of VHDL language, the whole program run error is likely to be because of a bracket missing, more than a semicolon this seemingly small error, but this small error will make the whole program can not be compiled through, affecting the follow-up simulation and burning. Therefore, we must be careful and careful in programming. The same is true for pin assignment, the dense pin correspondence diagram should not be careless, a switch location allocation error, will directly lead to incorrect program results. However, after solving one problem after another, I still feel very fulfilled when I see the digital tube or LED light on the experimental board turn on in turn.

I believe these two classes are just the beginning of FPGA learning. In the future study and work, I believe that I will often use the FPGA, so should be in the holidays or after-school time in-depth study, the power of the FPGA to play out.