Logic Design Lab 6

Experiment 1

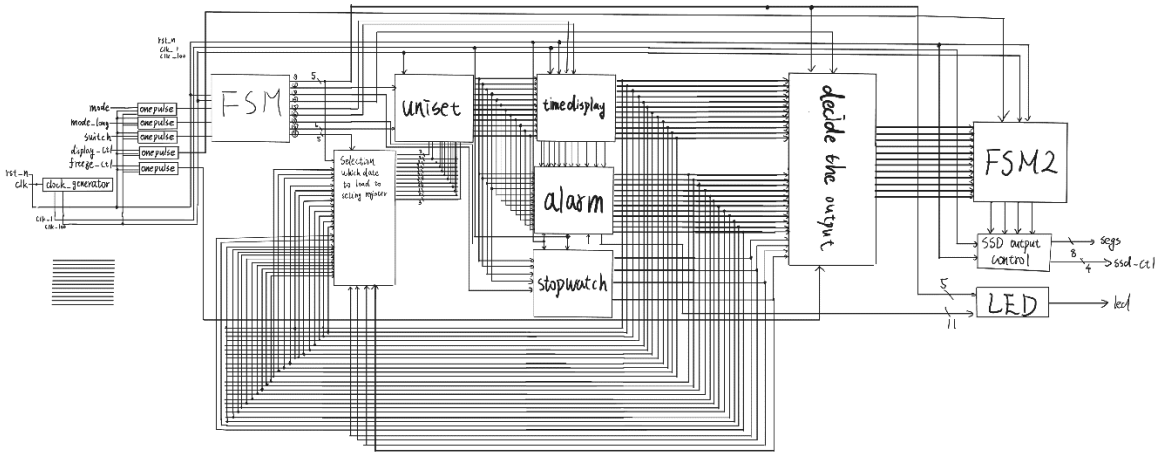
Implement an electronic clock with the following functions.

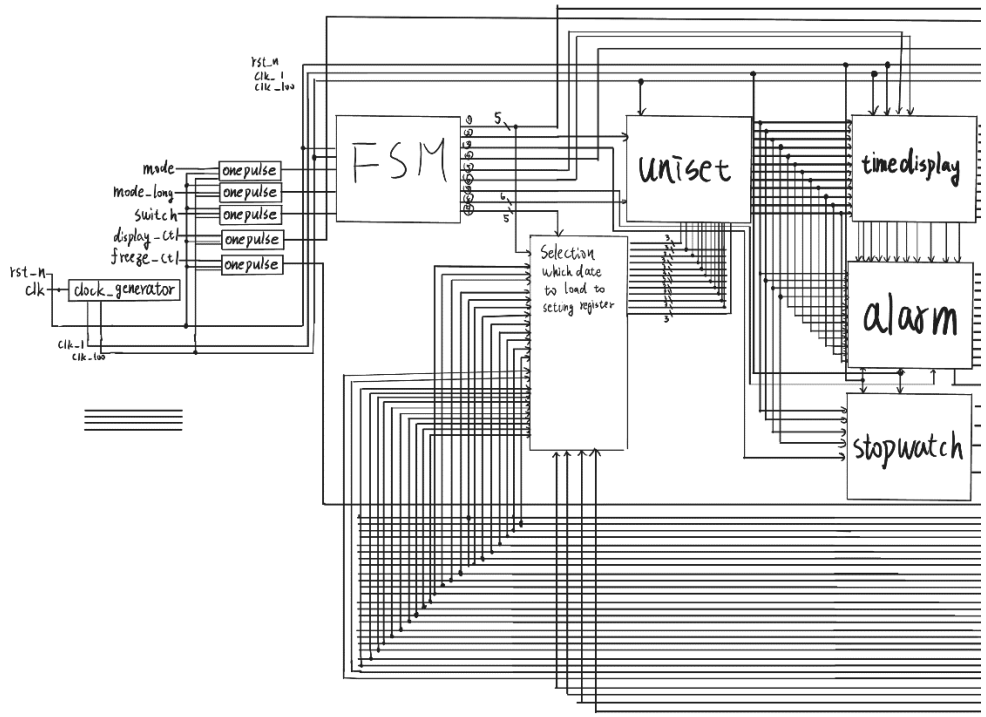
Design Specification：

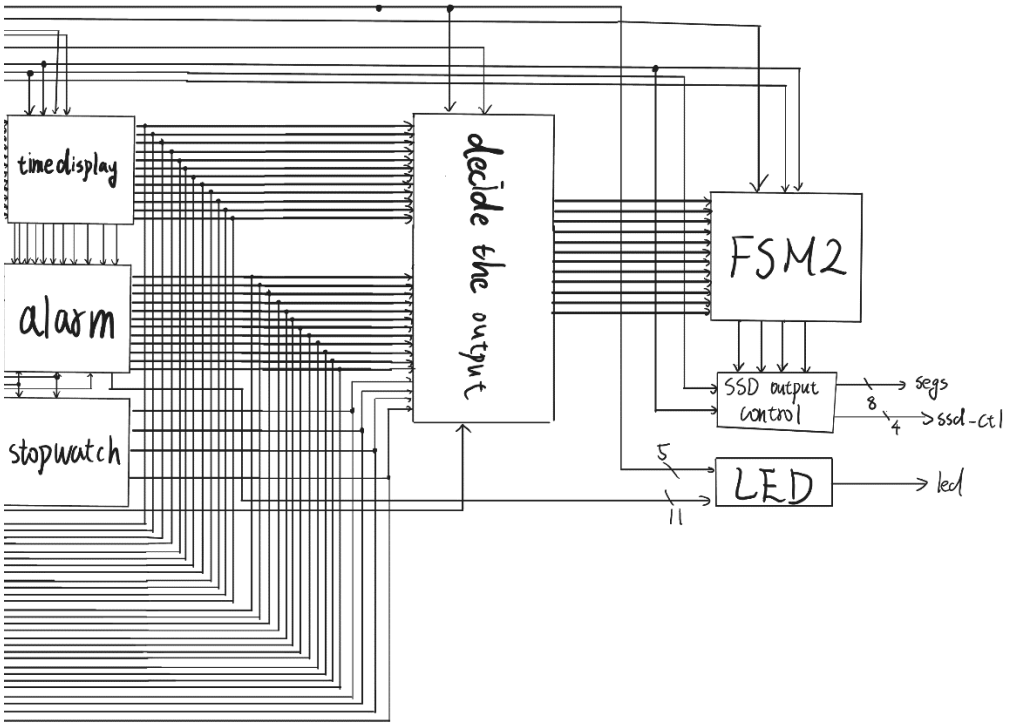
Input: clk, rst_n, mode, mode_long, switch, display_ctl, freeze_ctl.

Output: ssd_ctl [3:0], segs [7:0], led[15:0].

Block diagram:

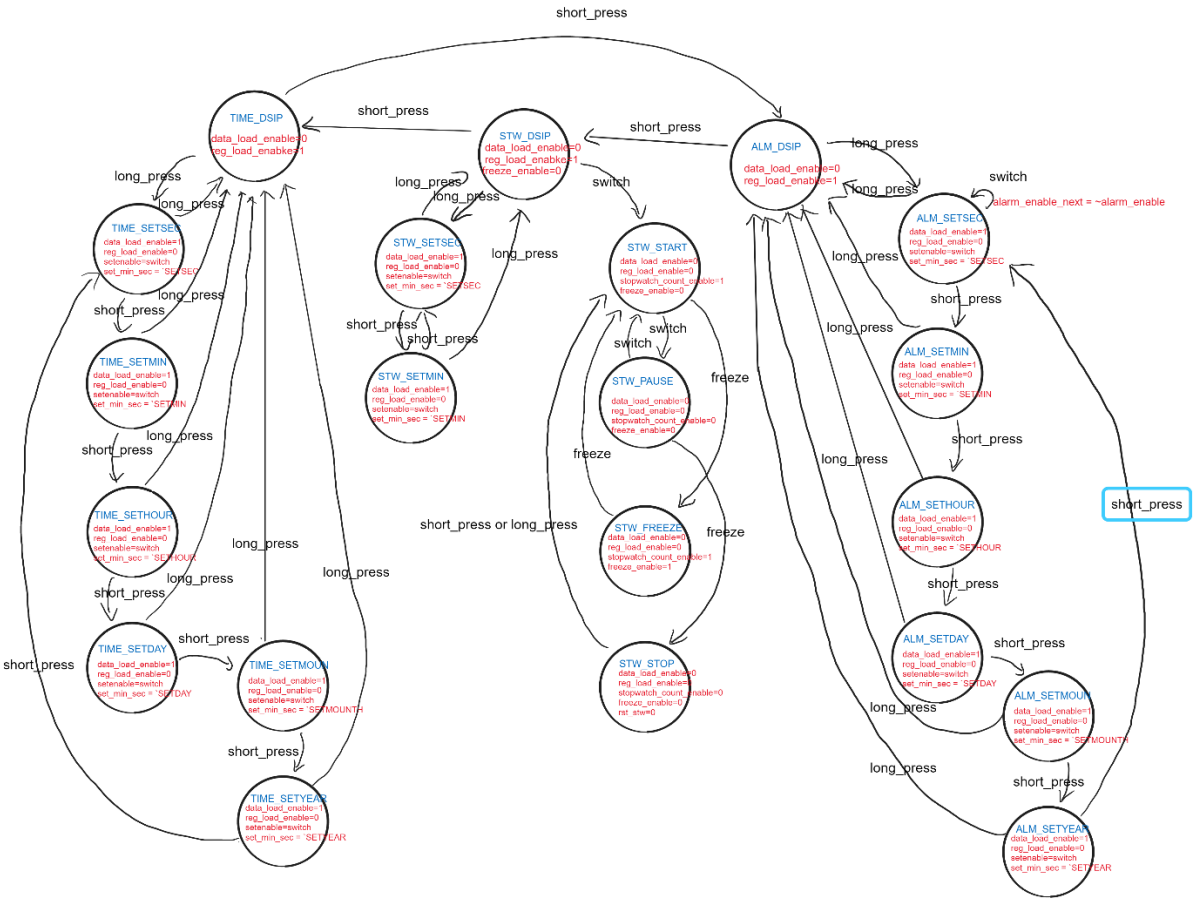


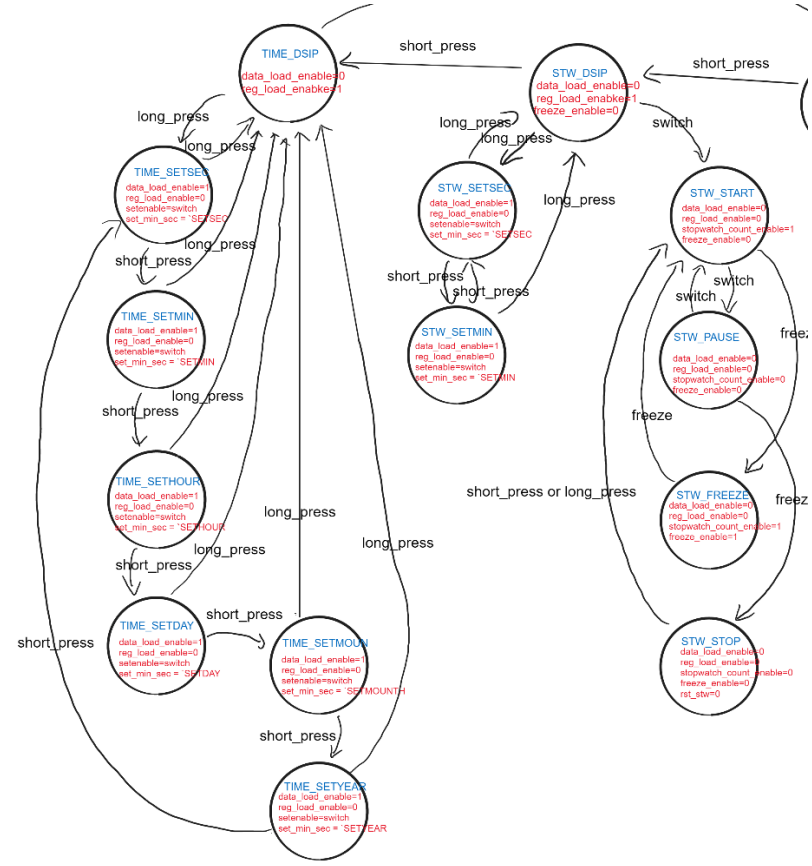


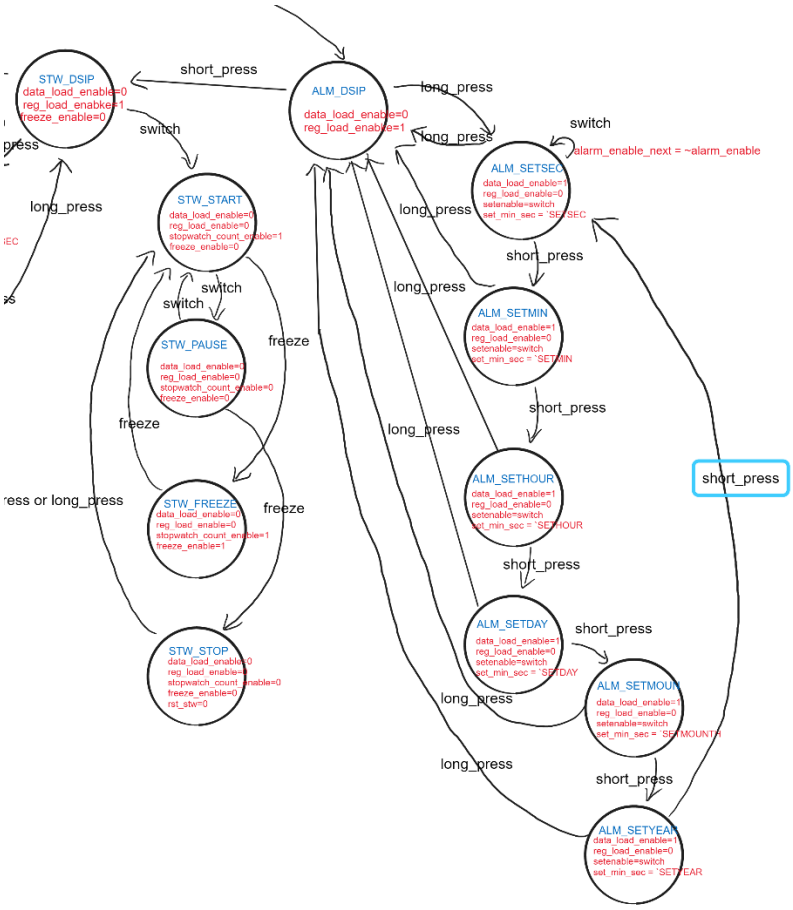


Design Implementation：

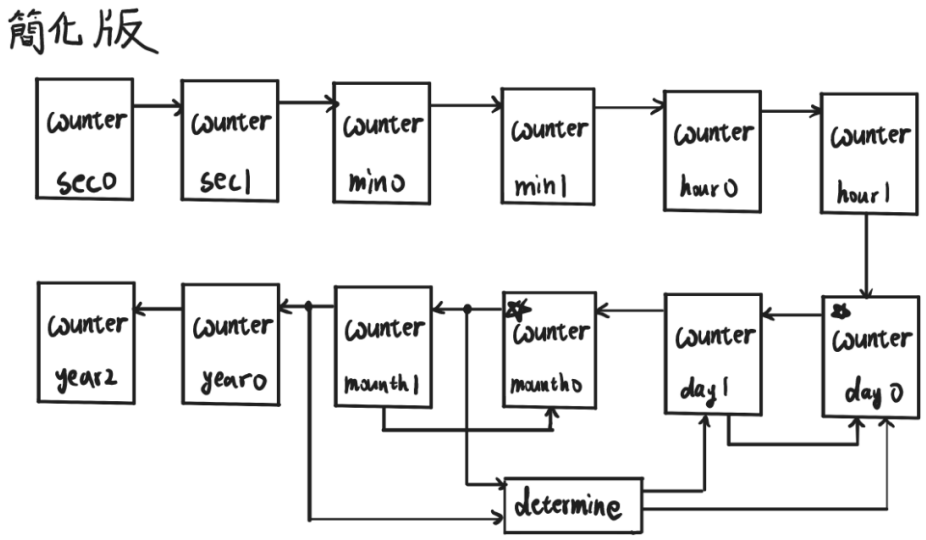
1. FSM：



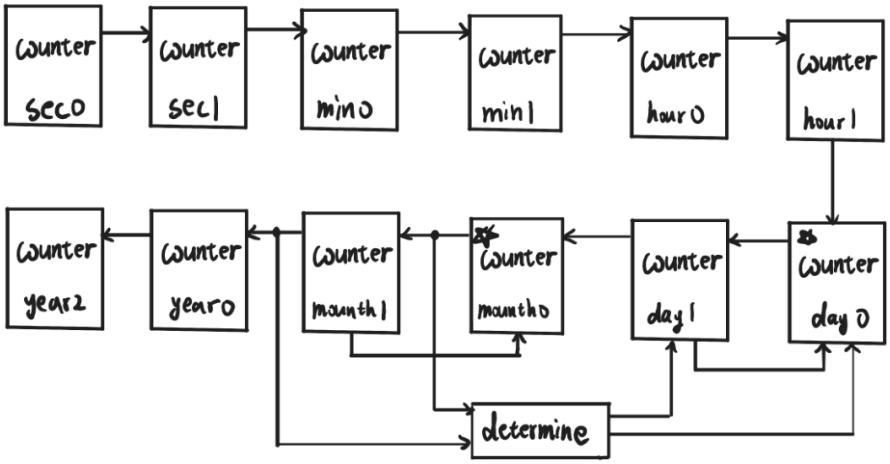

TIME_DSIP
data_load_enable=0
reg_load_enable=1
STW_DSIP
data_load_enable=0
reg_load_enable=1
freeze_enable=0
short_press
long_press
switch
TIME_SETSEC
data_load_enable=1
reg_load_enable=0
setenable=switch
set_min_sec = 'SETSEC
STW_SETSEC
data_load_enable=1
reg_load_enable=0
setenable=switch
set_min_sec = 'SETSEC
STW_START
data_load_enable=0
reg_load_enable=0
stopwatch_count_enable=1
freeze_enable=0
TIME_SETMIN
data_load_enable=1
reg_load_enable=0
setenable=switch
set_min_sec = 'SETMIN
STW_SETMIN
data_load_enable=1
reg_load_enable=0
setenable=switch
set_min_sec = 'SETMIN
STW_PAUSE
data_load_enable=0
reg_load_enable=0
stopwatch_count_enable=0
freeze_enable=0
TIME_SETHOUR
data_load_enable=1
reg_load_enable=0
setenable=switch
set_min_sec = 'SETHOUR
STW_FREEZE
data_load_enable=0
reg_load_enable=0
stopwatch_count_enable=1
freeze_enable=1
short_press or long_press
freeze
TIME_SETDAY
data_load_enable=1
reg_load_enable=0
setenable=switch
set_min_sec = 'SETDAY
TIME_SETMOUN
data_load_enable=1
reg_load_enable=0
setenable=switch
set_min_sec = 'SETMOUNTH
STW_STOP
data_load_enable=0
reg_load_enable=0
stopwatch_count_enable=0
freeze_enable=0
rst_stw=0
TIME_SETYEAR
data_load_enable=1
reg_load_enable=0
setenable=switch
set_min_sec = 'SETYEAR


ALM_DSIP
data_load_enable=0
reg_load_enable=1
STW_DSIP
data_load_enable=0
reg_load_enable=1
freeze_enable=0
short_press
long_press
switch
alarm_enable_next = ~alarm_enable
ALM_SETSEC
data_load_enable=1
reg_load_enable=0
setenable=switch
set_min_sec = 'SETSEC
STW_START
data_load_enable=0
reg_load_enable=0
stopwatch_count_enable=1
freeze_enable=0
ALM_SETMIN
data_load_enable=1
reg_load_enable=0
setenable=switch
set_min_sec = 'SETMIN
STW_PAUSE
data_load_enable=0
reg_load_enable=0
stopwatch_count_enable=0
freeze_enable=0
freeze
ALM_SETHOUR
data_load_enable=1
reg_load_enable=0
setenable=switch
set_min_sec = 'SETHOUR
STW_FREEZE
data_load_enable=0
reg_load_enable=0
stopwatch_count_enable=1
freeze_enable=1
ALM_SETDAY
data_load_enable=1
reg_load_enable=0
setenable=switch
set_min_sec = 'SETDAY
ALM_SETMOUN
data_load_enable=1
reg_load_enable=0
setenable=switch
set_min_sec = 'SETMOUNTH
STW_STOP
data_load_enable=0
reg_load_enable=0
stopwatch_count_enable=0
freeze_enable=0
rst_stw=0
ALM_SETYEAR
data_load_enable=1
reg_load_enable=0
setenable=switch
set_min_sec = 'SETYEAR

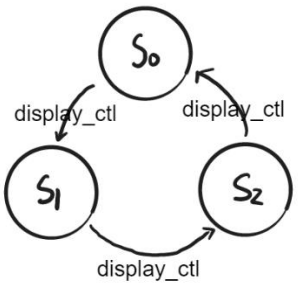
2. Timer



3. Alarm



4. FSM2



S_0 : show (min, sec)
S_1 : show (day, hour)
S_2 : show (year, month)

Input :
sec0, sec1, min0, min1, hour0, hour1
day0, day1, month0, month1, year0, year1

Output :

output	S_0	S_1	S_2
out1	sec0	hour0	month0
out2	sec1	hour1	month1
out3	min0	day0	year0
out4	min1	day1	year1

5. Stopwatch

   It was done well at the lab5_2. Skip.

6. Output control

   We have used the SSD control from time to time. Skip.

7. Selection block (partial code)

```
always @*
  case (state[4:3])
  `TIME:
    begin
      reg_load_q0 = time_sec0;
      reg_load_q1 = time_sec1;
      reg_load_q2 = time_min0;
      reg_load_q3 = time_min1;
      reg_load_q4 = time_hour0;
      reg_load_q5 = time_hour1;
      reg_load_q6 = time_day0;
      reg_load_q7 = time_day1;
      reg_load_q8 = time_mounth0;
      reg_load_q9 = time_mounth1;
      reg_load_q10 = time_year0;
      reg_load_q11 = time_year1;
    end
  `STW:
    begin
      reg_load_q0 = stopwatch_sec0_out;
      reg_load_q1 = stopwatch_sec1_out;
      reg_load_q2 = stopwatch_min0_out;
      reg_load_q3 = stopwatch_min1_out;
      reg_load_q4 = 0;
      reg_load_q5 = 0;
      reg_load_q6 = 0;
      reg_load_q7 = 0;
```

8. Deciding output block

```
always @*
  case (state)
  `TIME_DISP:
    begin
      sec0 = time_sec0;
      sec1 = time_sec1;
      min0 = time_min0;
      min1 = time_min1;
      hour0 = time_hour0;
      hour1 = time_hour1;
      day0 = time_day0;
      day1 = time_day1;
      mounth0 = time_mounth0;
      mounth1 = time_mounth1;
      year0 = time_year0;
      year1 = time_year1;
    end
  `TIME_SETSEC:
    begin
      sec0 = reg_q0;
      sec1 = reg_q1;
      min0 = reg_q2;
      min1 = reg_q3;
      hour0 = reg_q4;
      hour1 = reg_q5;
      day0 = reg_q6;
      day1 = reg_q7;
      mounth0 = reg_q8;
      mounth1 = reg_q9;
      year0 = reg_q10;
      year1 = reg_q11;
    end
```

Discussion：

1. Timer

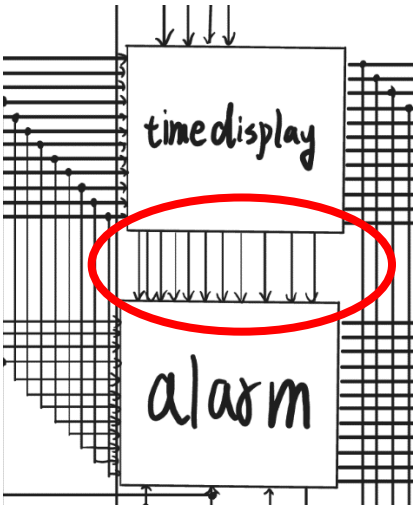
   (1) The counter with star is a special counter that it can receive information from the latter counter. The reason why we need this counter is that when some cases of carrying happens, the first unit of the day and month may be one and

zero. Thus, we need to check the decimal unit to determine whether it is 1 or not.

(2) The determine block is for the sake that when the month is 1, 3, 5, 7, 8, 10, 12 the number of days is 31 while that of other month are 30 mostly (the number of days in February is 28). Thus, we need to check what the month is to determine the days of that month.

2. Alarm

Actually, it is identical to the timer mostly. However, from the block diagram, pieces of information are required to be load to the alarm from the timer in order to determine the setting of the alarm is the same the timer goes to the number.



3. From the block diagram, there is a **FSM2** on that. The function of the FSM2 is to switch the pattern on the SSD from (min,sec) to (day,hour), from (day,hour)to (year,month) and from (year,month) to (min,sec). Furthermore, I use Moore model to implement the FSM, because the problem of hazard may be solved easier.

4. The **uniset** block is made by cascading a series of the counter (also be seen as the register) and receive information from a combination circuit for selecting which data load on this block and load data to the timer, alarm and stopwatch to to set the number on the timer, alarm or stopwatch.

5. The selection block is a combination circuit to decide the load data from timer, alarm or stopwatch.

6. The decision block is a combination of 3 parts. The first part is DISP, the general default state for three functions. The second part is setting, which is to let the information of the register load on the timer, alarm or stopwatch. The third function is about freezing in stopwatch, which is done in lab5 (skip).

Conclusion：

In my opinion, this experiment is not so challenging as lab5 is. The only one point we need to be aware of is that the first days of a month and the first month of a year is one. However, the whole experiment is very complicate. Sometime, a simple typal error may take us a lot of time to find out the bug. So, I spent lots of time doing the lad as I did in lad5. By the way, for me, perhaps this experiment includes a lot of functions, I may be confused when conducting the FPGA board. Maybe I can assign the button more intuitive next lab.

Reference：

1. The handout and the assignment of logic design
   To review some basic concept of the combination circuit and the problem bulls and cows.
2. The handout of logic design lab
   To program our FPGA board by following the method on the handout of logic design lab.