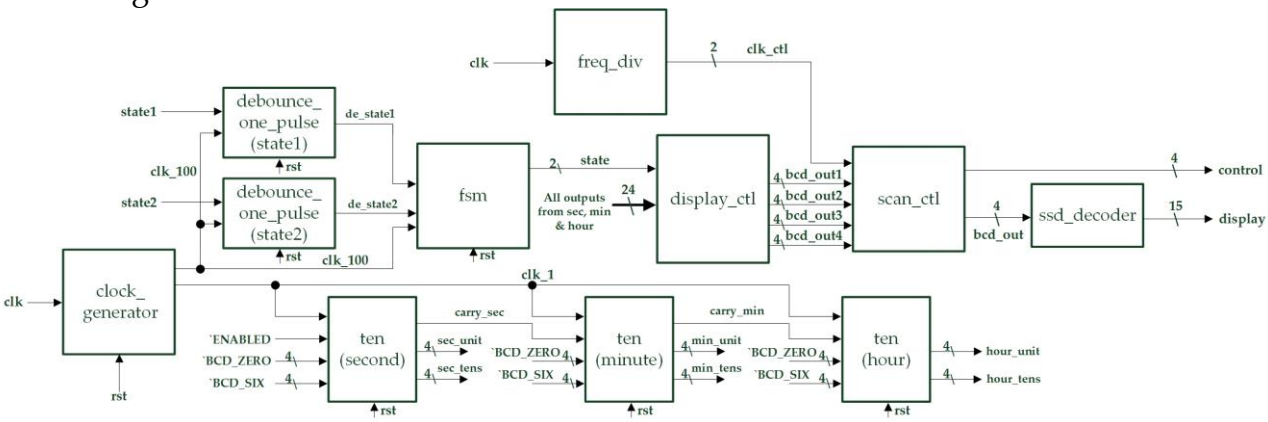
Lab 7: Electronic Clock I (Time Display)

103061207 徐安廷 An-Ting Hsu

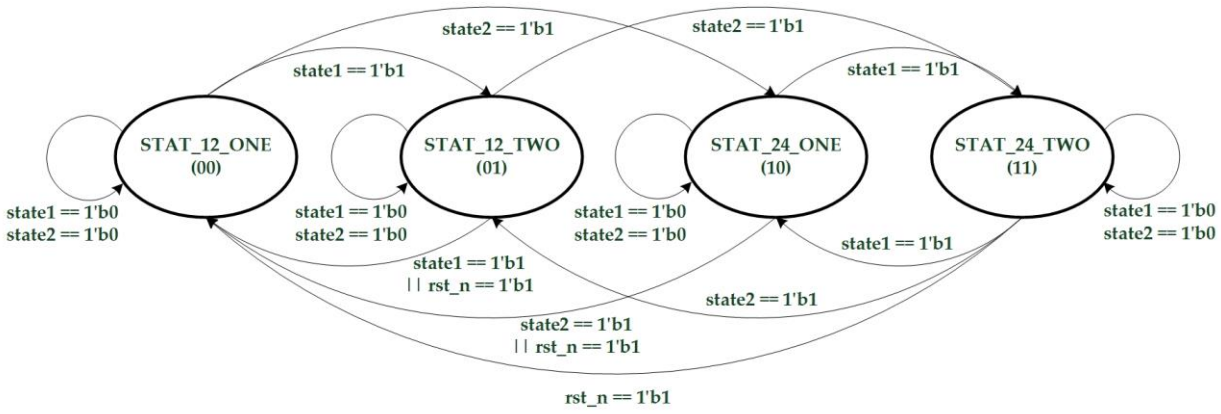
Design Specification

1. Electronic Clock

- ✓ Experiment Goal:
  Construct an electronic clock which can support two modes: AM/PM and 24-hour.
- ✓ Block Diagram:



- ✓ State Diagram:

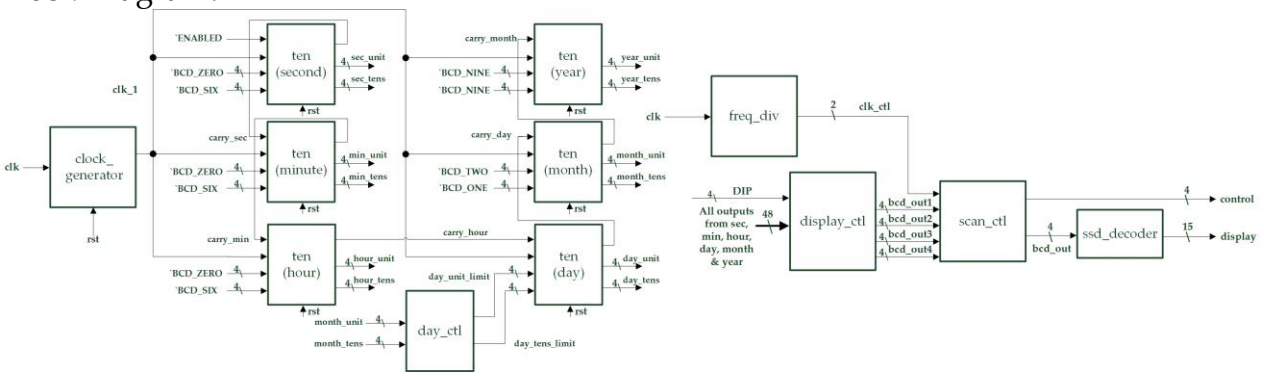


- ✓ I/Os:
  Inputs: clk, state1, state2, rst_n.
  Outputs: [3:0] control, [14:0] display.
- ✓ Details about some module:
  - ■ display_ctl: Determines what is going to show on the 14-degment display according from fsm's state.

2. Calendar (No Leap Year)

- ✓ Experiment Goal:
  Adjust experiment 1's electronic clock to support day, month and year timer.
  - o Jan/March/May/July/Aug/Oct/Dec: 1-31, Feb: 28, Apr/June/Sept/Nov: 30
  - o No leap year function.

- ✓ Block Diagram:

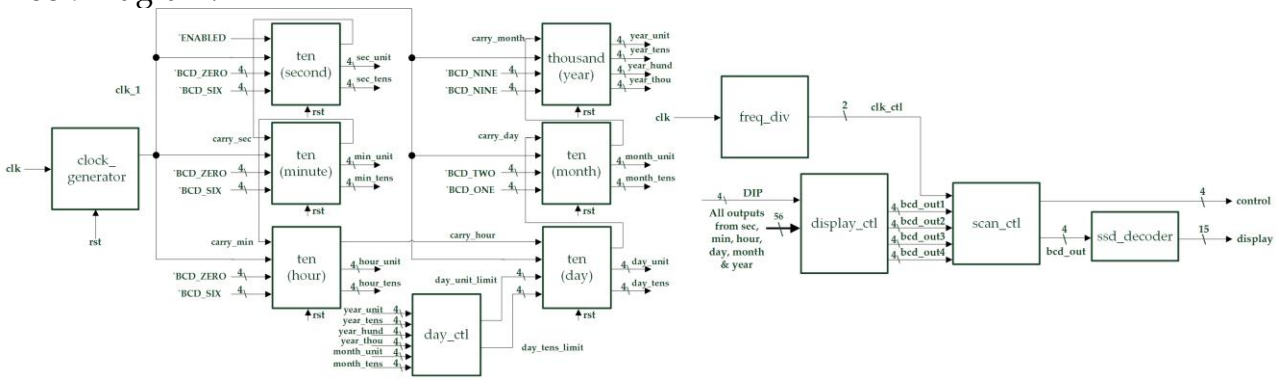


- ✓ I/Os:
  Inputs: clk, rst_n, [3:0] DIP.
  Outputs: [3:0] control, [14:0] display.
- ✓ Details about some module:
  - day_ctl: Determines what the day limit is according from the month value.
  - display_ctl: Determines what is going to show on the 14-degment display according from 4-bit DIP input.

DIP[3]~DIP[0]	Things to show on 14-SD
0000	Seconds & minutes in 12-hr mode
0001	Seconds in 24-hr mode
0010	Hours in 12-hr mode
0011	Hours & minutes in 24-hr mode
01xx	Month & Day
1xxx	Year

3. (Bonus) Calendar (Leap Year Support)

- ✓ Experiment Goal:
  Adjust exp 3 to support leap year function.
- ✓ Block Diagram:



- ✓ I/Os:
  Inputs: clk, rst_n, [3:0] DIP.
  Outputs: [3:0] control, [14:0] display.
- ✓ Details about some module:
  - day_ctl: Determines what the day limit is according from the month and year value.

Design Implementation

1. Electronic Clock

- ✓ Construct three basic units which contains two up counter each, and each unit represents second, minute and hour. The second unit will always increase its value when reach a positive edge clock signal. Minute and hour units will increase its value when second or minute reaches their limit value. Send these units' value to a display controller which will determine what the thing are going to show according from the finite state machine.
- ✓ I/O Pins Assignment:

clk	R10
rst_n	T2
state1	N3
state2	P4
control[0]~control[3]	V8, U8, V6, T6
display[0]~display[14]	U5, T7, R7, V7, V4, T4, T3, R5, N5, R3, U7, T5, V5, N4, P6

2. Calendar (No Leap Year)

- ✓ Add day, month and year counters into first experiment. And, the day limit is determined by a controller which will passes correct day limit by the current month value.
- ✓ I/O Pins Assignment:

clk	R10
rst_n	T2
DIP[0]~DIP[3]	L2, M1, M3, N1
control[0]~control[3]	V8, U8, V6, T6
display[0]~display[14]	U5, T7, R7, V7, V4, T4, T3, R5, N5, R3, U7, T5, V5, N4, P6

3. (Bonus) Calendar (Leap Year Support)

- ✓ Adjust the day controller in experiment 2 to support leap year function.
- ✓ I/O Pins Assignment:

clk	R10
rst_n	T2
DIP[0]~DIP[3]	L2, M1, M3, N1
control[0]~control[3]	V8, U8, V6, T6
display[0]~display[14]	U5, T7, R7, V7, V4, T4, T3, R5, N5, R3, U7, T5, V5, N4, P6

Discussion

- ✓ I spent lots of time debug on the first experiment. One of the main bugs is the hour counter will count to 20 not 00 when it reach the limit value. I found that the bug is in the upcounter module. I miss exchange the place of two if else statement "load_default == `ENABLED" & "increase == `DISABLED", so the counter will not load the default value properly. The other problem I met is that I can't use don't care symbol "x" in the case

statement, so I googled the solution to my problem. I learn that I need to change "case" statement into "casex" statement, so the don't cares can work properly.

Conclusion

- ✓ After the lab I understand the electronic clock in our daily life basic working principle and know how to construct my own clock. Though there was some problems during the experiment, I think those precious experiences will improve my coding skills. And, it will help me to finish the final project easier.

References

1. Electronic Clock

- ✓ Teaching Handout <Universal Counter> p.2~p.7
  →Helps me to construct a counter which can count to a particular value and back to the initial value.