

**Faculty of Applied Science and Computing**

**Division of Microelectronics & Physic**

**Bachelor of Science**

**Microelectronics with Embedded Technology**

**RMB2**

**BAME 2004 – DIGITAL SYSTEM DESIGN**

|  |  |
| --- | --- |
| **Student Names** | **Student ID** |
| **Chiang Choon Yong** | **13WAR11086** |
| **Au Chee Liang** | **13WAR11593** |
| **Poon Jon** | **13WAR11301** |
| **Soo Fatt** |  |

**DATE : 22 march 2014**

**TITLE : Designing a Traffic Controller (FSM)**

**INTRODUCTION**

**Objective:**

* To study on the traffic controller by using the Finite-state machine (FSM).
* To design a traffic controller in verilog code using FSM approach on Quatus II and synthesize it into the Altera DE1 FPGA board.
* To learn the Altera ModalSim software in order to simulate the result and validate it with real case result.

**Problem statement:**

* To design a simulator for a traffic controller at the intersection of a Main Highway and Country road with interrupt sensor and traffic light represented as switches and LED lights on the Altera DE1 board.
* To implement a timer to calculate and provide time delay that is required for changes between states and display period for a message.

**Brief Background:**

A traffic controller is basically a controller to direct traffic. Nowadays, traffic controllers are widely used on the intersections of roads to control the flow of traffic and prevent congestion of traffic which is automated by the controller. Traffic lights are useful for areas with heavy traffic, the traffic light directs traffic by showing a red light as a sign to stop and a green light to go.

**METHODOLOGY**

**Approach for the design:**

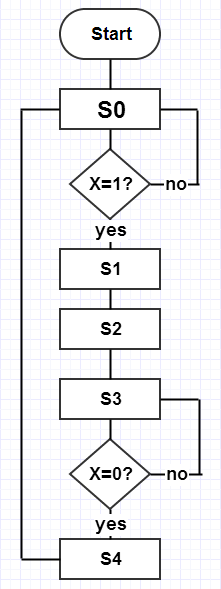
We are required to design a traffic controller for a main highway and a country road by using LED lights on the Altera FPGA DE1 board in verilog code.

We designed the traffic controller using moore Finite-state machine (FSM ) approach. As shown in Figure 1.1. There are a total of 5 states, starting from S0 to S4. On state S0, the LED light for Highway should be green while Country light is red. Followed by the states S1 and S2, Highway light becomes yellow then red while Country Road remains red. Lastly, during state S3 and S4 the light for Country Road becomes green while Highway’s remains red, then it will return to S0 and repeat.

Two different Verilog coding approaches were used to design the traffic controller. The first one was to write the whole traffic controller in one module, it is a straightforward way of designing it. While the other approach is to write sub-modules then instantiate them into the main module, the overall code looks cleaner but it requires a stronger grasp in Verilog language to do it.

The designs were verified by using ModalSim on a testbench to obtain the expected waveform of the program. The designs were also verified by downloading it into the DE1 board to obtain physical results from the LEDs.

There were some problems encountered while designing the traffic controller, it was mostly due to bad design methods for the timing delay of the traffic controller.

****

**Figure 1.0: Flow Chart of traffic controller**

**Procedure:**

1. The traffic signal for the main highway gets the highest priority because the cars are continuously present on the main highway. Thus, the light remains green by default.
2. If there are cars arriving from the country road at the traffic signal. The signal for the country road interrupt is triggered and turns green to let the cars on country road to go, meanwhile the traffic signal on main highway turns red to stop the traffic on the main highway.
3. As soon as the cars on country are cleared, the traffic signal green light on country roads turns yellow and then followed by red and the sensor for the interrupt are set back to default, while the traffic signal on the main highway turns green again.
4. There is a sensor to detect cars that are waiting on the country road. The sensors send an interrupt signal X to the microcontroller to generate the traffic signal. If there are cars present on the country road, the signal X turns to 1, if there are none, remain X as 0.
5. There are delays on transition from S1 to S2 of 10s, from S2 to S3 of 20s and from S4 to S0 of 10s. The delays are controllable by the internal clock delay
6. There is also a count-down timer on the HEX0 and HEX1 7segment display for all delays in step 5 above.
7. The “HrGO” message is displayed on four 7-segment display during the state of S0.
8. The “CrGO” message is displayed during the state of S3.
9. Display a flickering “STOP” message for 5s during the transition from S2 to S3.

X = 0

X = 0

X = 1

X = 1

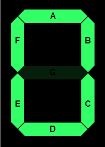
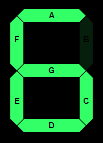
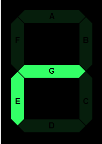
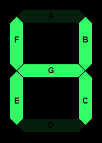
**Figure 1.1: FSM for traffic signal**

As the figure 1.1 shows above, there are delays on transition from S1 to S2, from S2 to S3 and from S4 to S0.

|  |  |  |  |
| --- | --- | --- | --- |
| **State** | **Signal** | | **Delays(s)** |
| **Highway** | **Country** |
| **S0** | **Green** | **Red** | **-** |
| **S1** | **Yellow** | **Red** | **10** |
| **S2** | **Red** | **Red** | **20** |
| **S3** | **Red** | **Green** | **-** |
| **S4** | **Red** | **Yellow** | **10** |

**Table 1.1 FSM table of data**

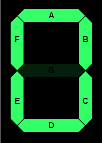
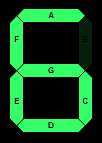
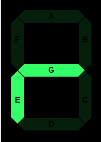
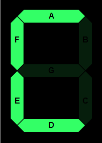
**Active low 7 segment HEX [3:0].**

****

**Figure 1.2: 7-segment display for “HrGO”**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **HrGO** | **g** | **f** | **e** | **d** | **c** | **b** | **a** |
| **HEX0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** |
| **HEX1** | **0** | **0** | **0** | **0** | **0** | **1** | **0** |
| **HEX2** | **0** | **1** | **0** | **1** | **1** | **1** | **1** |
| **HEX3** | **0** | **0** | **0** | **1** | **0** | **0** | **1** |

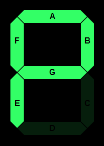
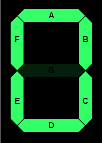
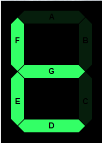
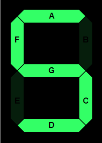
**Table 1.2: state of 7-segment display for “HrGO”**

****

**Figure 1.3: 7-segment display for “CrGO”**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **CrGO** | **g** | **f** | **e** | **d** | **c** | **b** | **a** |
| **HEX0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** |
| **HEX1** | **0** | **0** | **0** | **0** | **0** | **1** | **0** |
| **HEX2** | **0** | **1** | **0** | **1** | **1** | **1** | **1** |
| **HEX3** | **1** | **0** | **0** | **0** | **1** | **1** | **0** |

**Table 1.3: state of 7-segment display for “CrGO”**

****

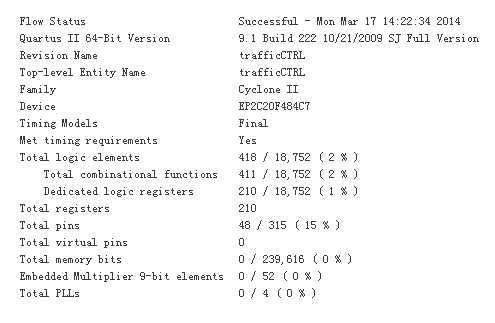
**Figure 1.4: 7-segment display for “STOP”**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **STOP** | **g** | **f** | **e** | **d** | **c** | **b** | **a** |
| **HEX0** | **0** | **0** | **0** | **1** | **1** | **0** | **0** |
| **HEX1** | **1** | **0** | **0** | **0** | **0** | **0** | **0** |
| **HEX2** | **0** | **0** | **0** | **0** | **1** | **1** | **1** |
| **HEX3** | **0** | **0** | **1** | **0** | **0** | **1** | **0** |

**Table 1.4: state of 7-segment display for “STOP”**

**Simulation Result & Discussion:**

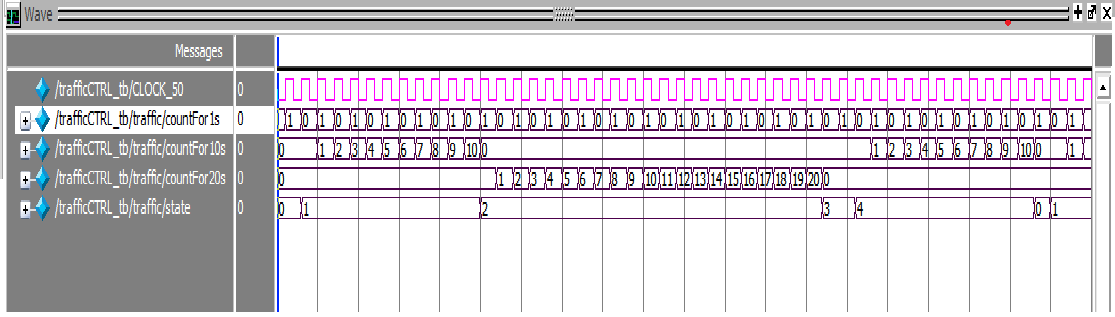
The first approach, where the design was written in only one module



**Figure 1.5 Compilation Report**

The simulated result of the traffic light controller can be improved by decreasing the number of uses of register by using the instantiation methods of coding and the traffic controller program can be more efficient for our current design.

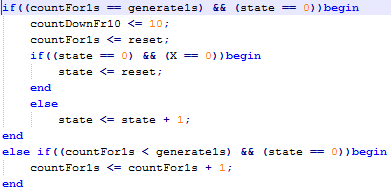
**Test Bench**

****

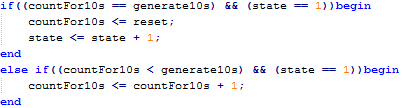
**Figure 1.6: Test Bench of Traffic Controller with the FSM State**

* **State 0 –** If countFor1s not equal to one, it should increase by 1. Otherwise, it should clear and change the state to state 1.
* **State 1 –** After the clock triggered the countFor10s will start to count 1 by 1 for the counter. When it equal to 10, it will be clear and change the state to State 2.
* **State 2 –** This state will hold for 20s before come to the next state. Most of operations are same as before.
* **State 3 –** At this state, it will hold only one cycle before it go to next state if there are no interrupt being detected.
* **State 4 –** This state will change to state 0 and continue repeating the previous 4 task after it finished the counting of 10 second.

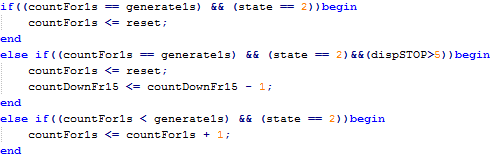
**Source code for state S0**

****

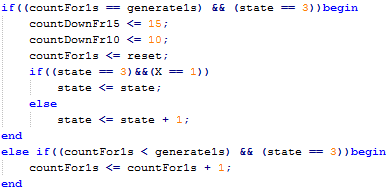
**Source code for state S1**

****

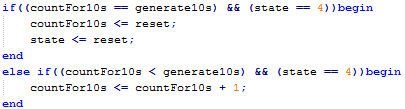
**Source code for state S2**

****

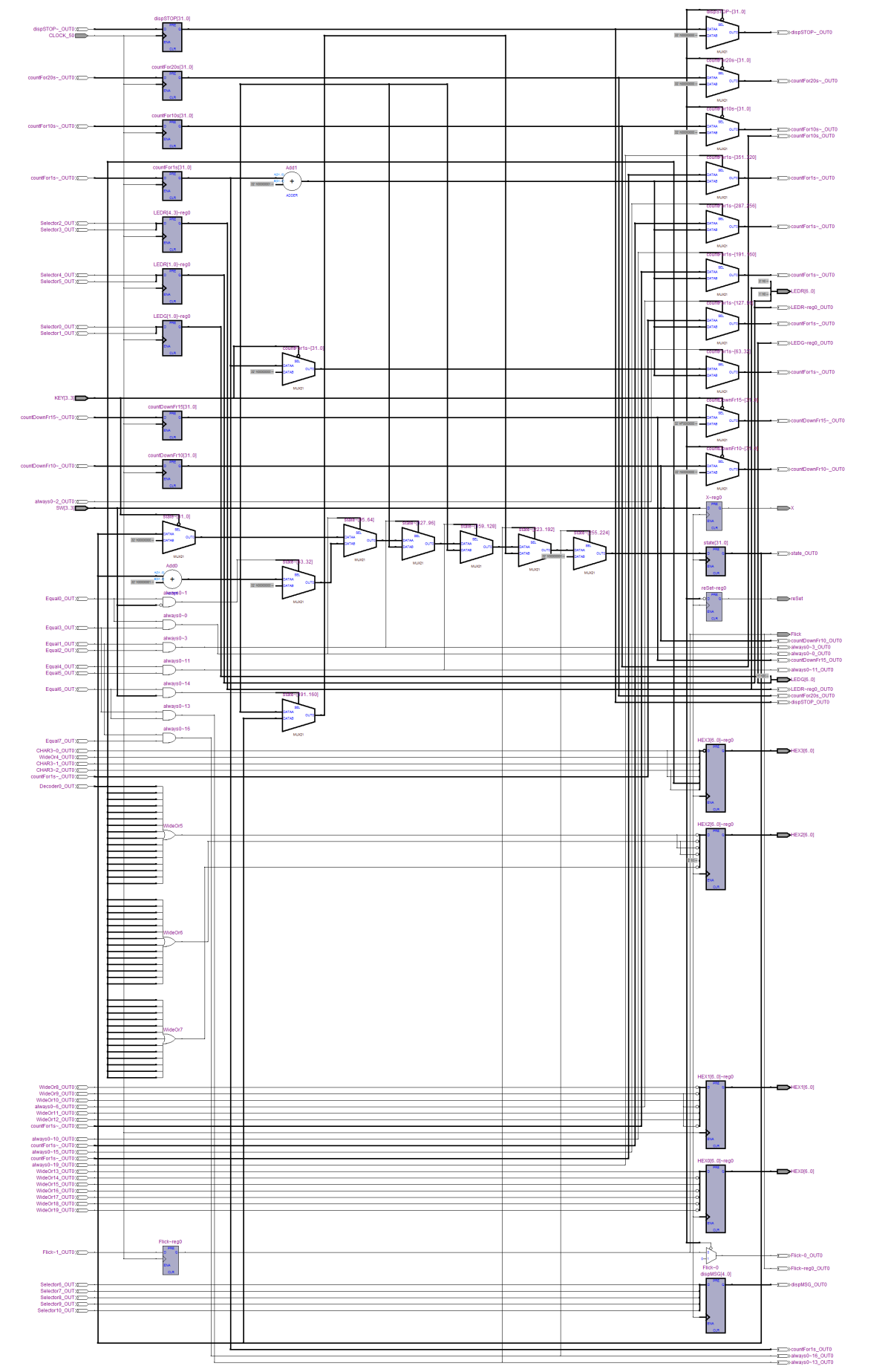
**Source code for state S3**

****

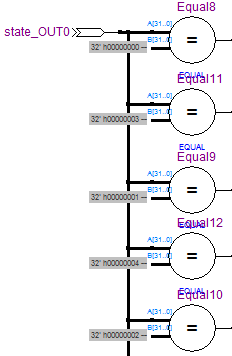
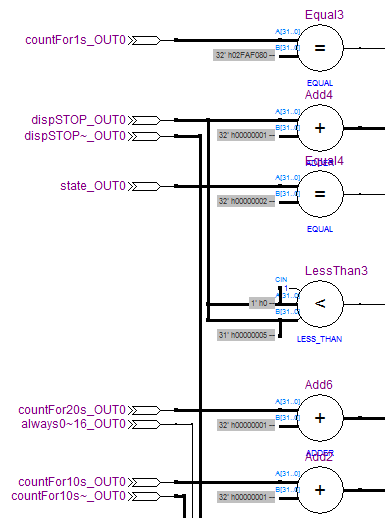
**Source code for state S4**

****

**RTL Viewer**

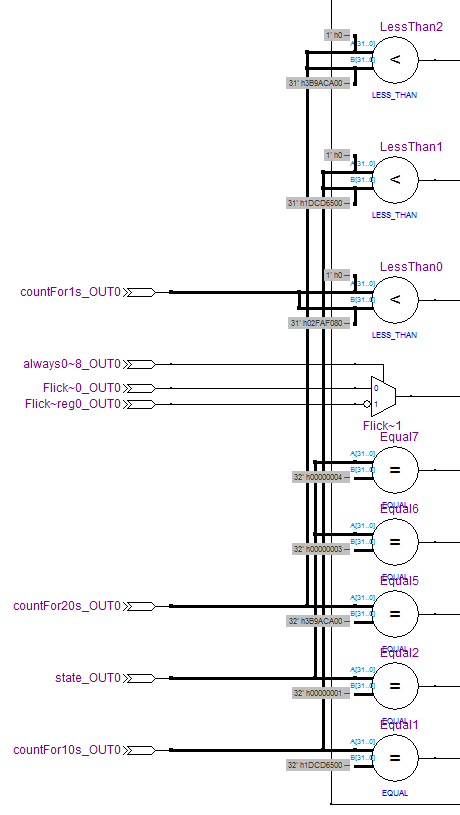
****

**Figure 1.7: Full RTL View of traffic controller.**

****

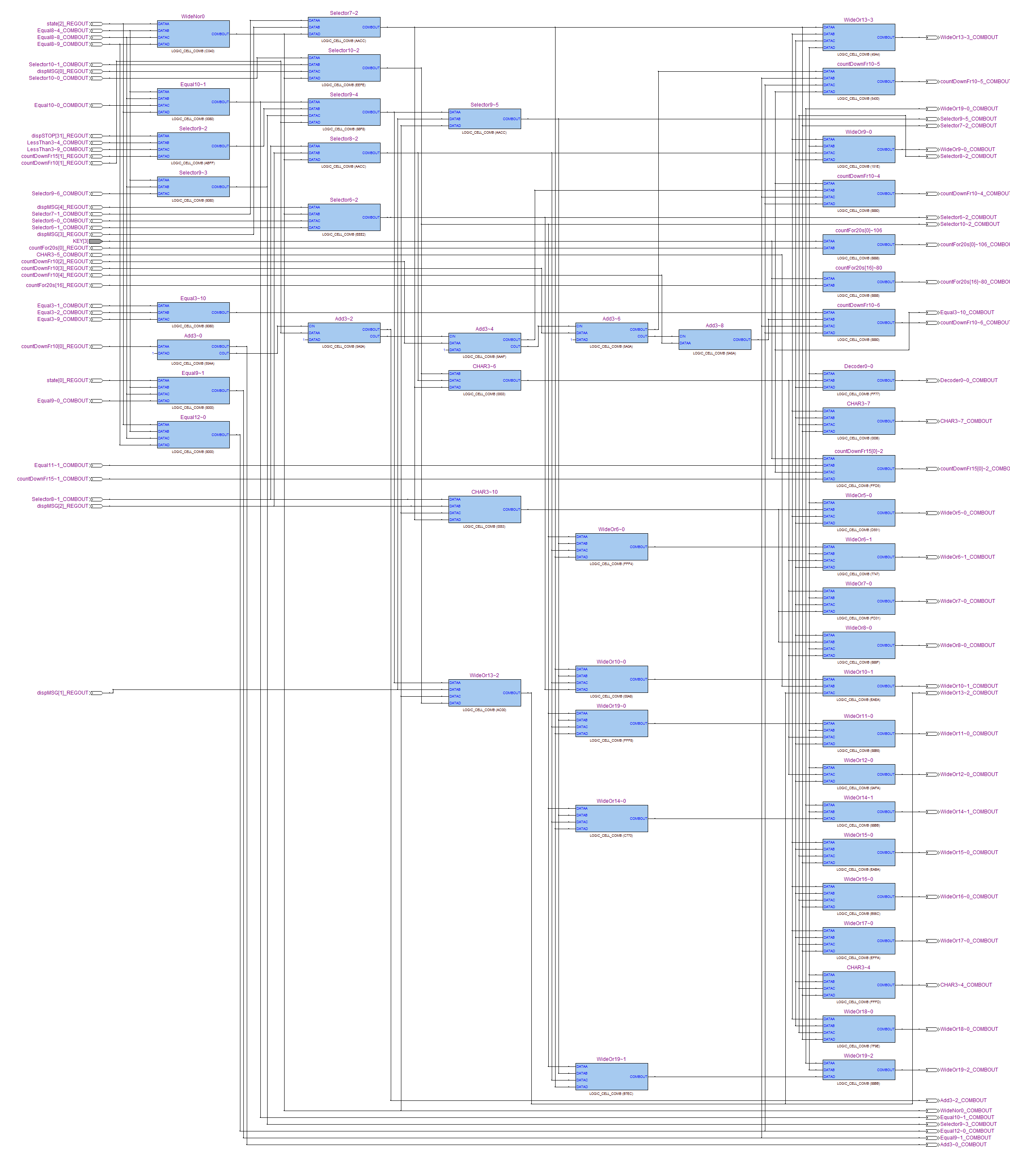
**Figure: 1.8 & Figure 1.9**

* **RTL viewer in condition and decision making of next actions. (left)**
* **State and condition compares (right)**

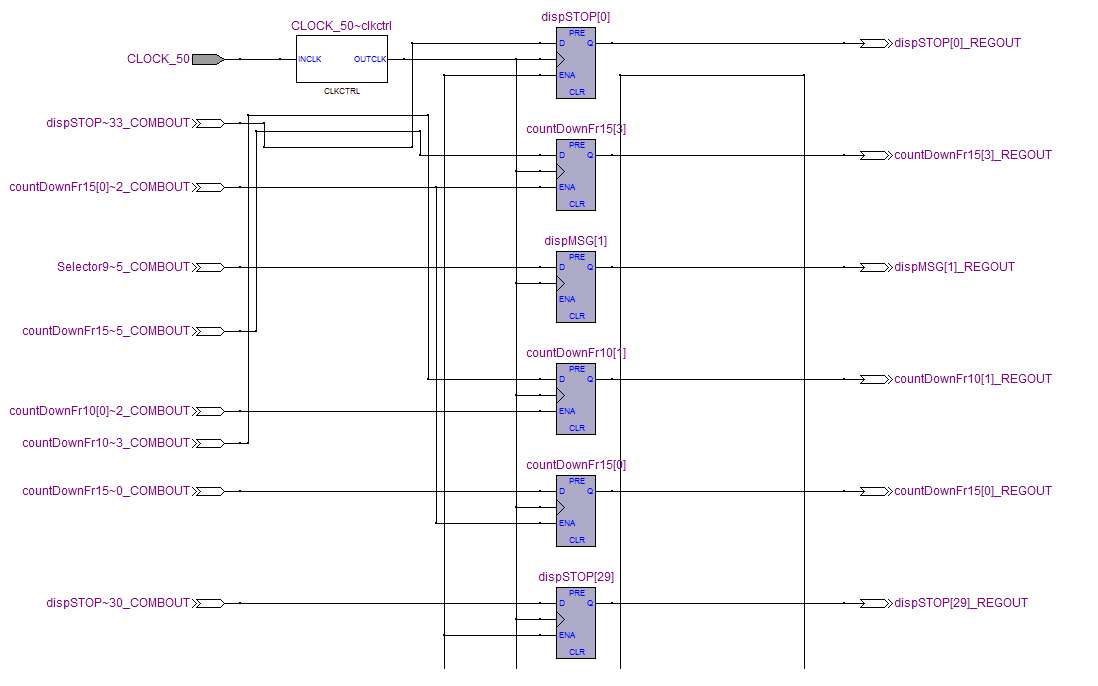
****

**Figure 1.10 Timer for the FSM state S1 to S2, S2 to S3 and S4 to S0.**

**Tech Mapping - Post Mapping**

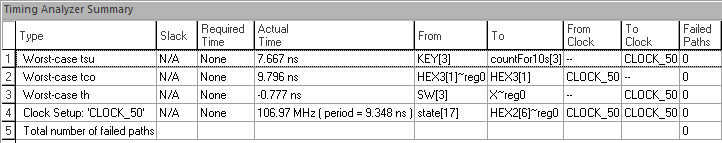
****

**Figure 1.11: Full image of Post mapping on FSM state.**

****

**Figure 1.12: Tech mappling for the clock\_50**

**Timing and Delays**

****

**Figure 1.13: Timing Analyzer Summary**

**Timing Analysis (Worse-Case Path)**

**Tmin = tCQ + tPD + tSU**

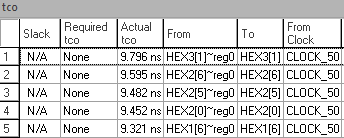
**Delay = tCQ + tPD**

**Tmin = 9.796 ns + 9.107ns + 7.667ns**

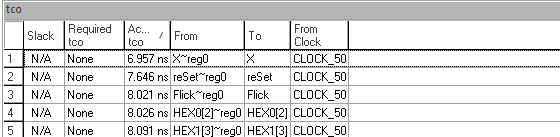
**= 26.57 ns**

**Delay = 6.957 ns + 9.107ns**

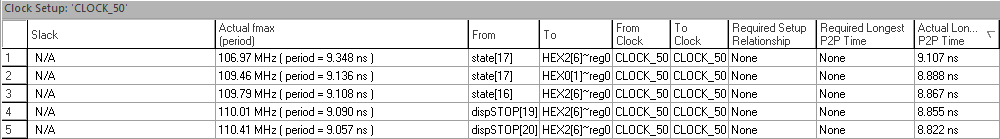
**= 16.064 ns (16.064 ns >9.107ns) No hold time Violation.**

****

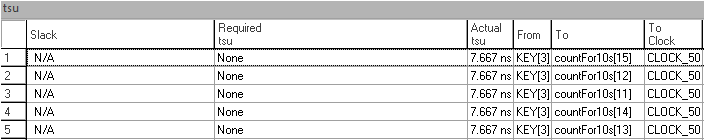
**Figure 1.14: Timing Analysis (tCQ Longest)**

****

**Figure 1.15: Timing Analysis (tCQ Shortest)**

****

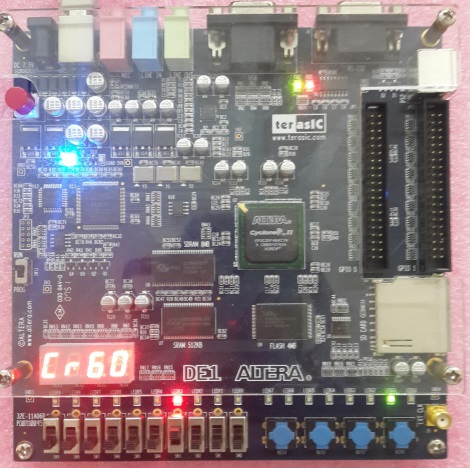
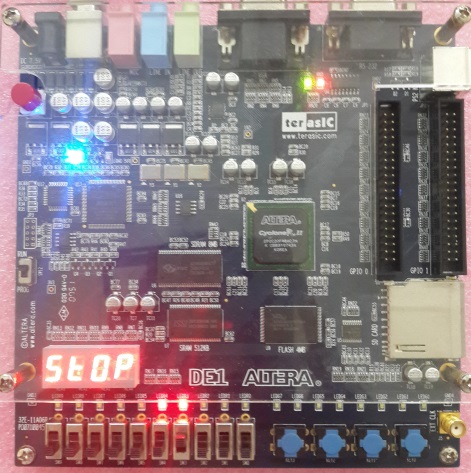
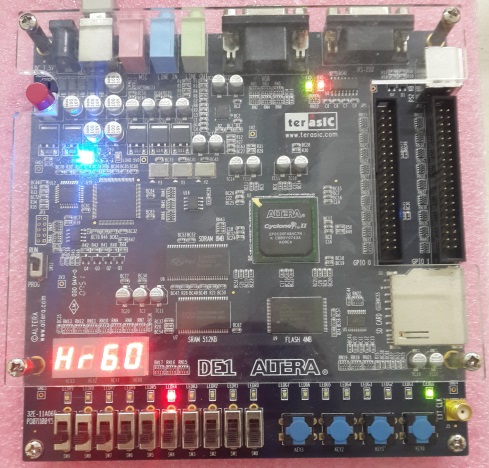
**Figure 1.16: Timing Analysis (th)**

****

**Figure 1.17: Timing Analysis (tSU)**

Setup time is the minimum time before the clock’s active edge by which the data must be stable for it to be latched correctly. If setup violation occurs, data might be incorrect.

Hold time is the minimum time after the clock’s active edge during which the data must be stable. If hold violation occurs, data might be incorrect.



**Figure 1.8: Result of Altera DE1 FPGA board**

LEDG[0] (as green light) = Main Highway (right most)

LEDG[1] (as green light) = Country Road

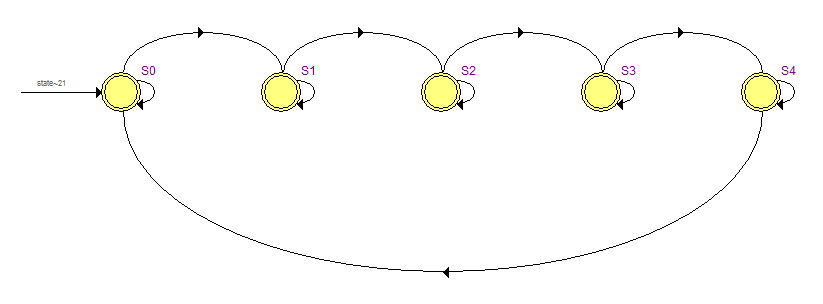
LEDR[0] (as yellow light) = Main Highway

LEDR[1] (as yellow light) = Country Road

LEDR[3] (as red light) = Main Highway

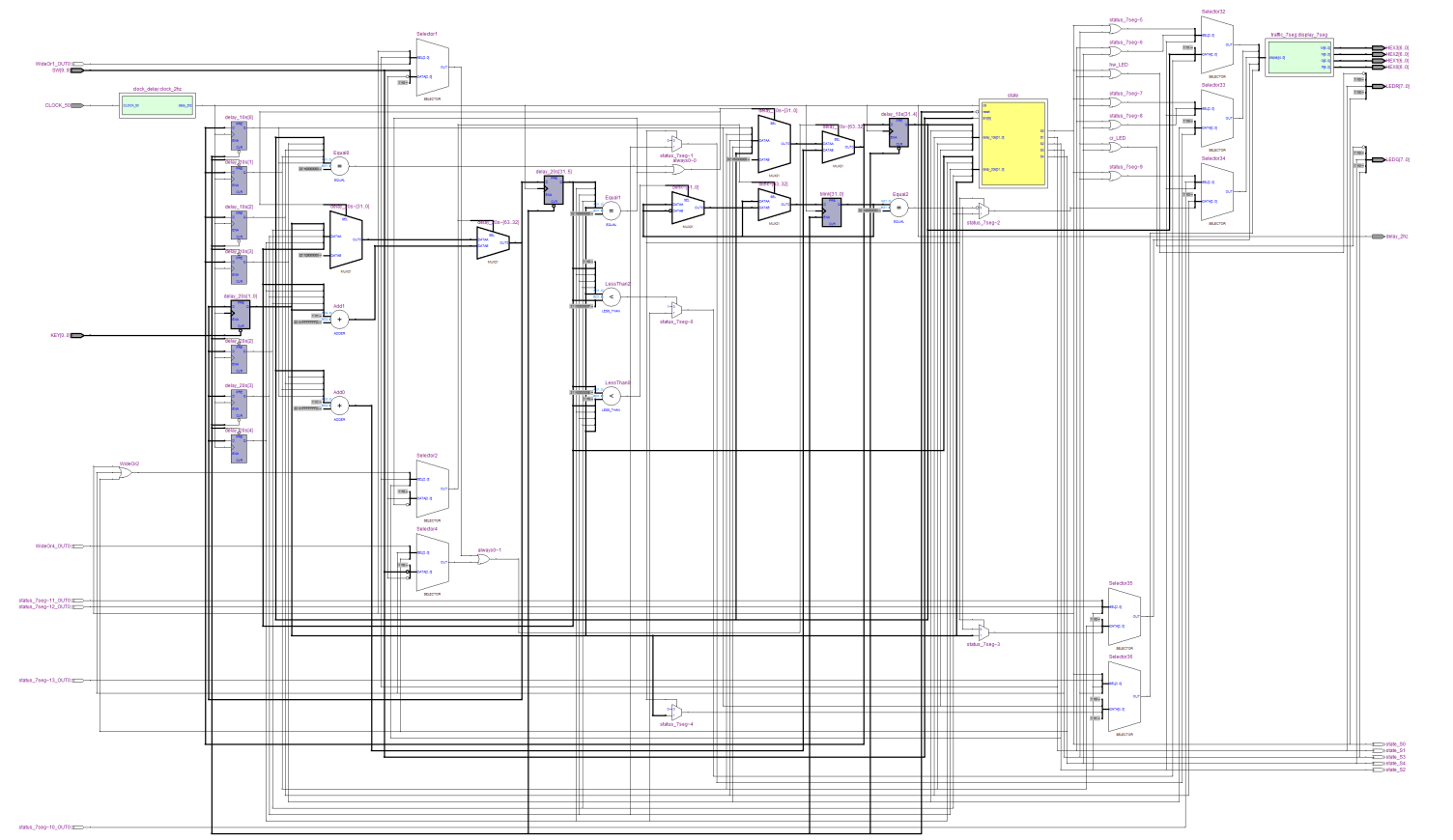
LEDR[4] (as red light) = Country Road

Second design approach, by writing sub-modules then instantiate it into the main module. Unfortunately we failed to obtain the testbench for this design approach. However we were able to obtain the state diagram



All the states will have a next state of itself, because it was required for the delay to work properly.

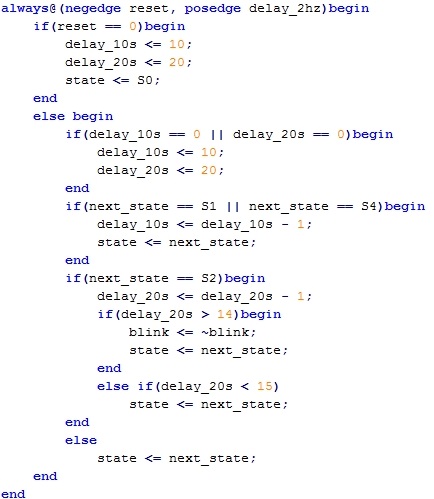
**RTL Viewer**

****

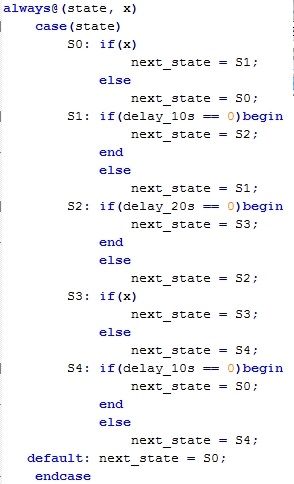
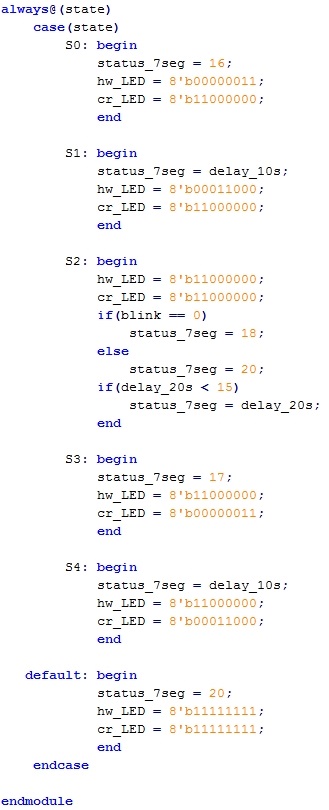
**DIAGRAM 1.9 RTL Viewer**

The design code was separated into 3 parts, 1 sequential, 2 combinatorial code and 2 sub-modules.

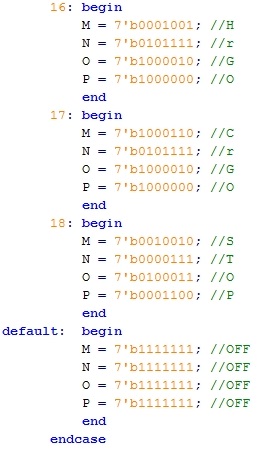
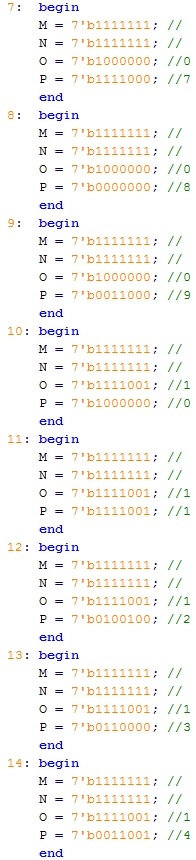
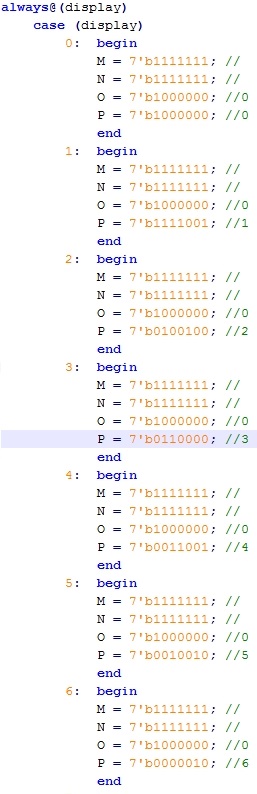
**Source code for sequential**



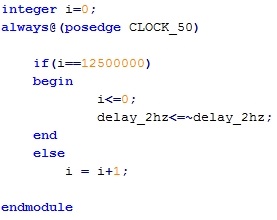
**Source code for combinatorial**

****

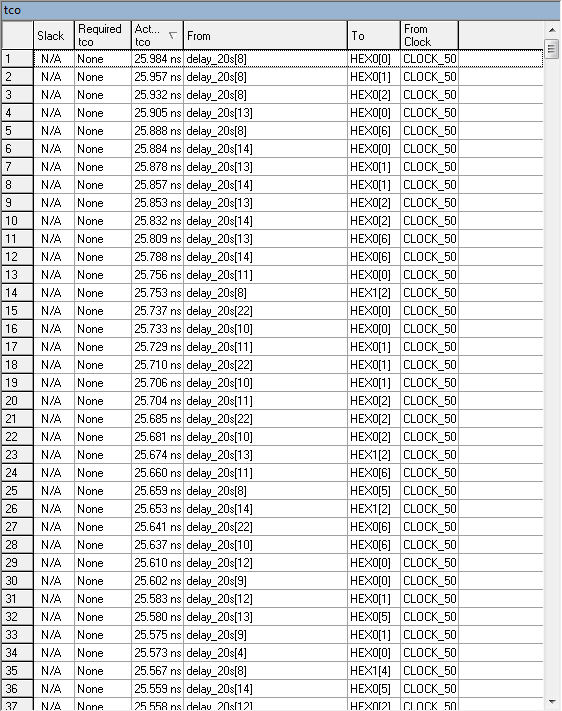
**7 Segment sub-module**

****

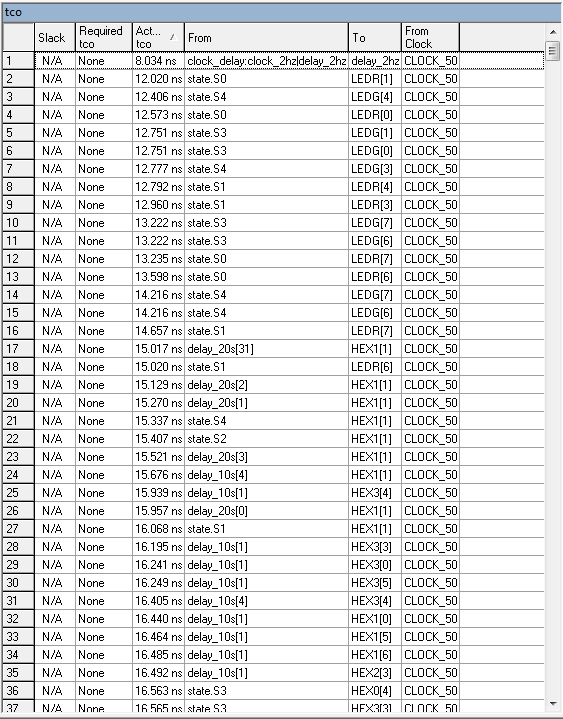
**Clock Delay sub-module**

****

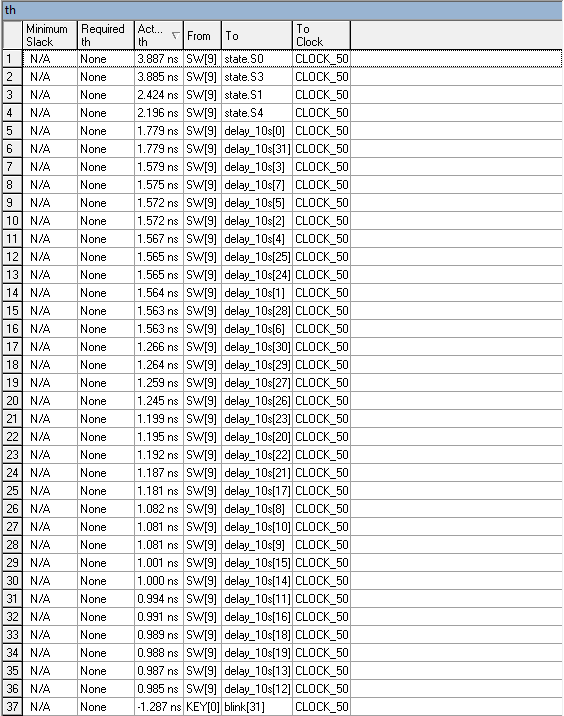
**Timing and Delays**

****

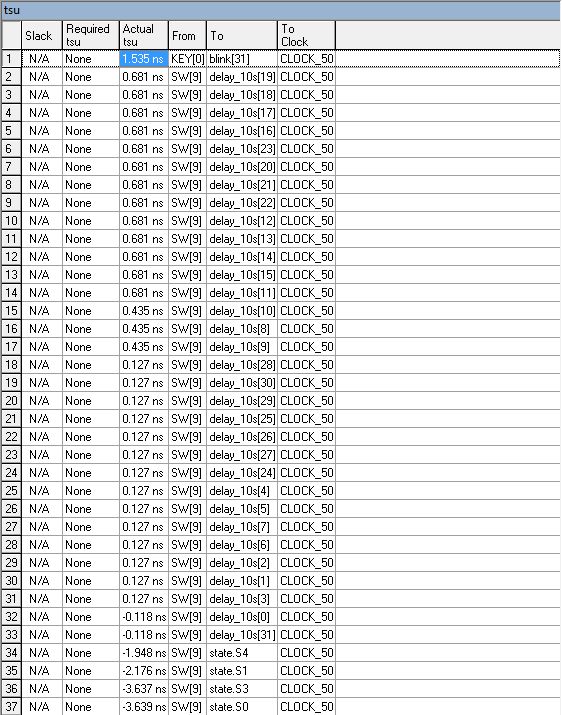
**Figure 1.18: Timing Analysis (tCQ Longest)**

****

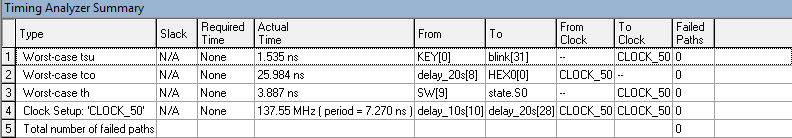
**Figure 1.19: Timing Analysis (tCQ Shortest)**

****

**Figure 1.20: Timing Analysis (th)**

****

**Figure 1.21: Timing Analysis (tSU)**

****

**Figure 1.22: Timing Analyzer Summary**

**Conclusion:**

The traffic controller theory which came from the Finite-state machine (FSM) in this project has been verified with the help of Verilog coding program to run the task and perform all the state in the FSM. The Altera DE1 FPGA board provides the checking process of state changing for the traffic controller; the program is downloaded to the FPGA board to run the process.

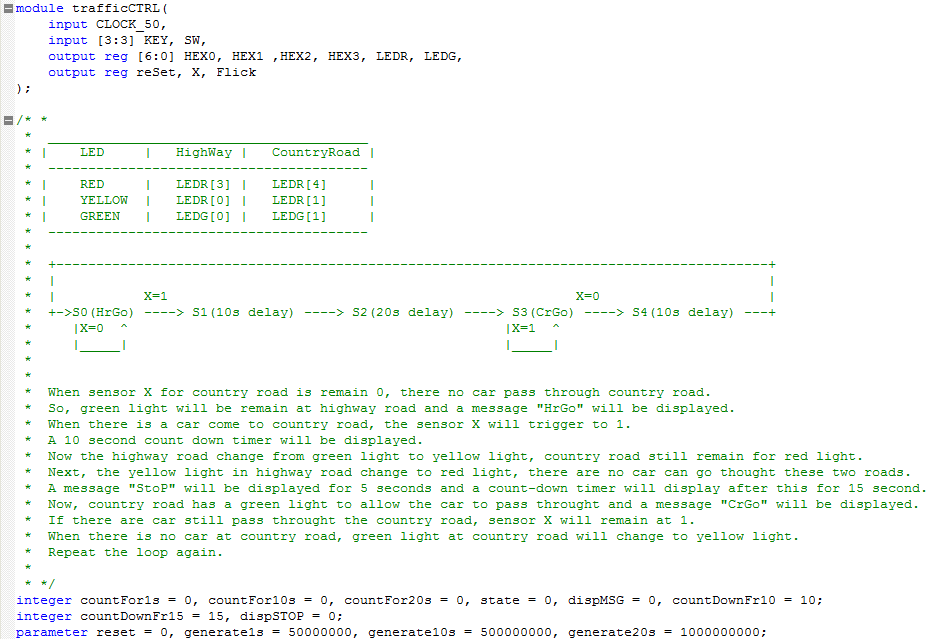
From the result that was obtained from the Altera ModalSim, it shows that the delays has worked for delay in S1, S2 and S4 and the result of the experiment met the requirement for the traffic controller by using the Altera DE1board.

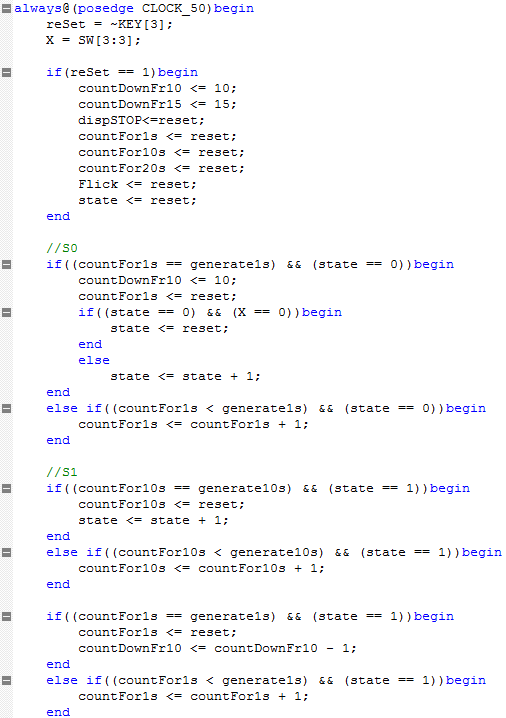
The calculation above, no hold time violation occurred because the calculated delay time exceeded the minimum hold time delay which is 9.107ns. The Tmin obtained from above is 26.57 ns which the minimum time for the clock period for this circuit works.

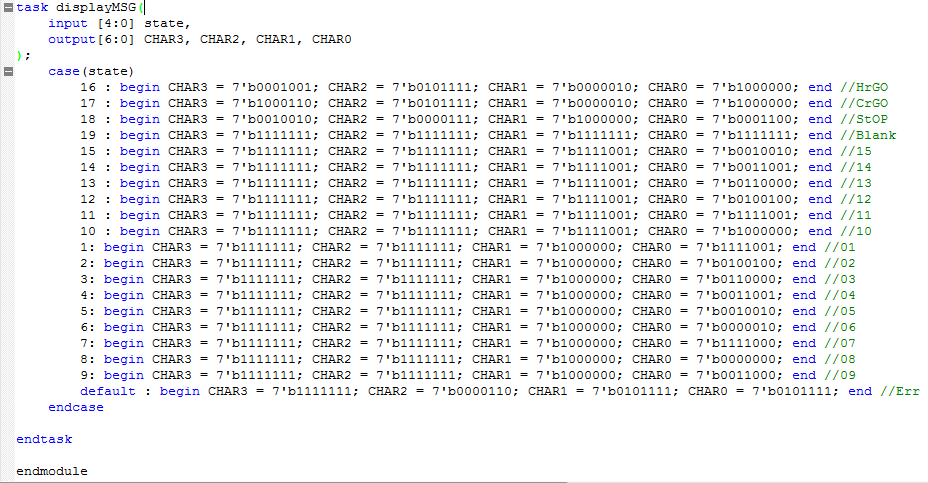
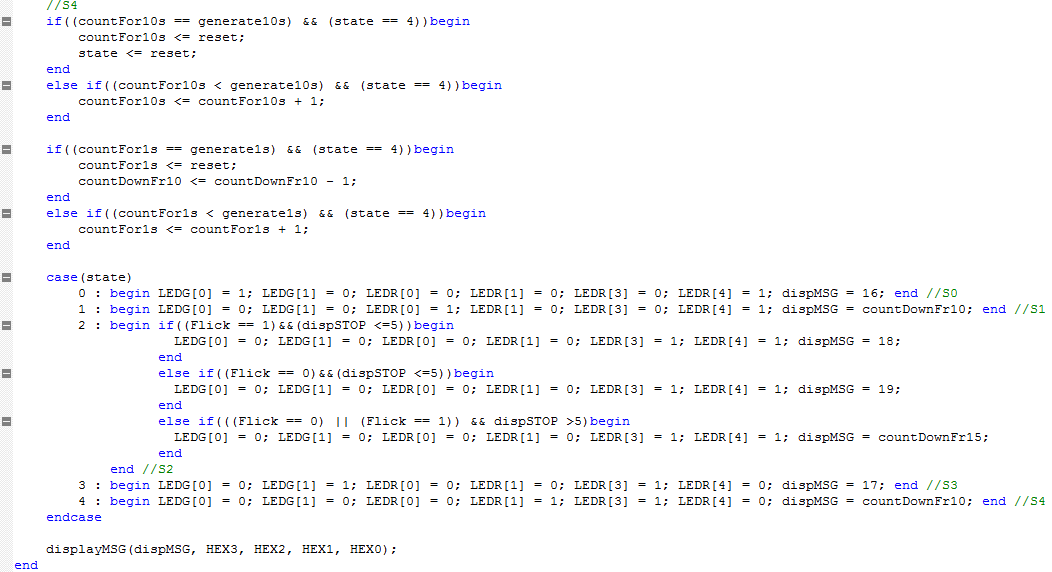
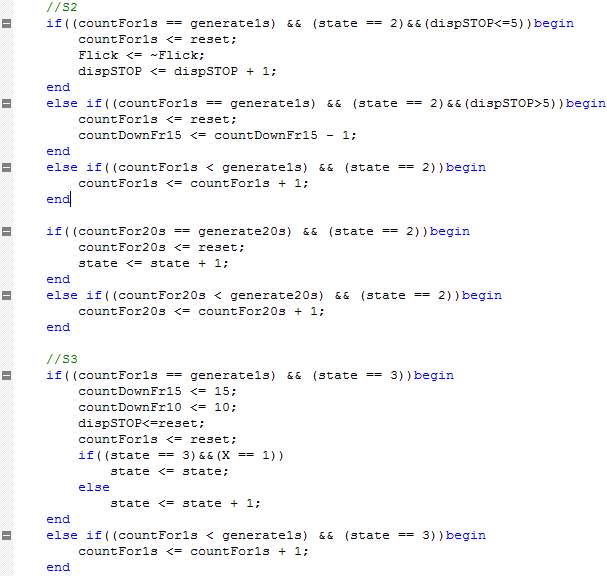
In conclusion, the traffic light controller can be improved by using the instantiation methods of coding and the traffic controller program can be more efficient for our current design. For the future development, the power consumption s for this design will be lower and the size would be smaller. In this case, the speed will increase as the area of size decrease.

**Appendix: Verilog Code Listing**

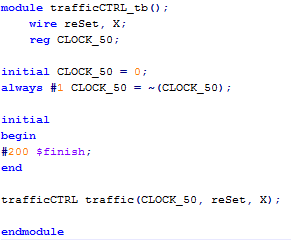
**Method 1**

****

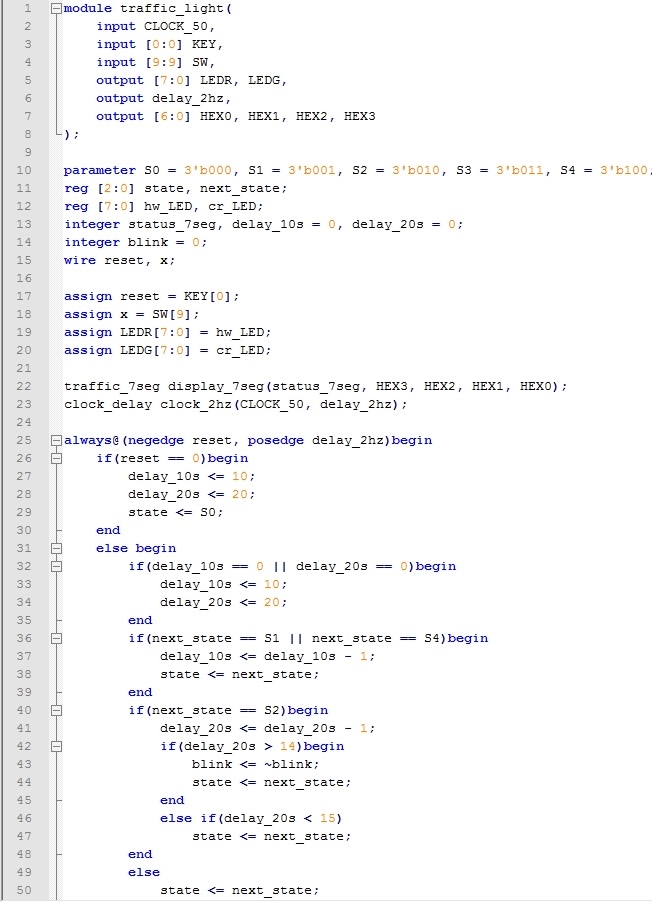
****

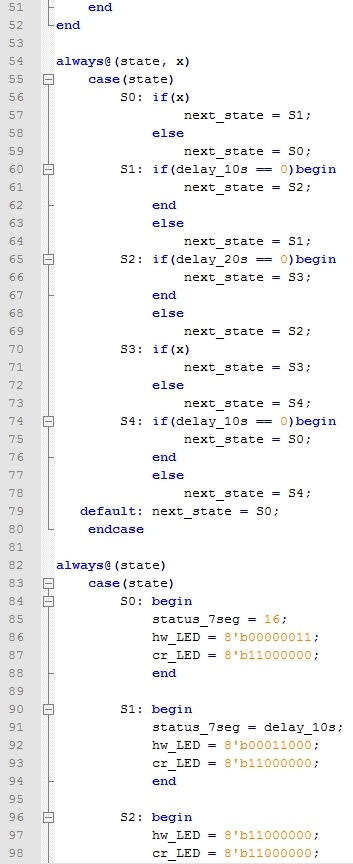
****

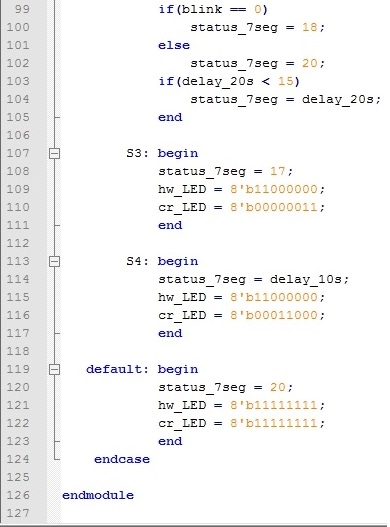
**Test Bench Code Listing**

****

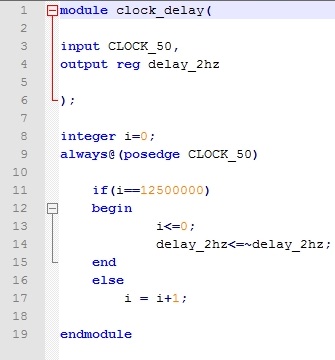
**Method 2**

****

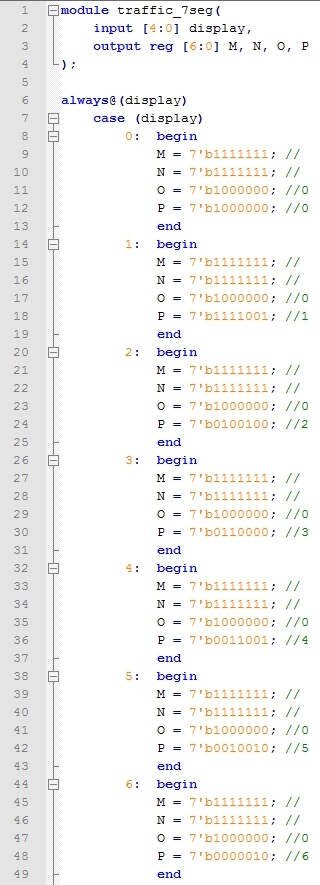
****

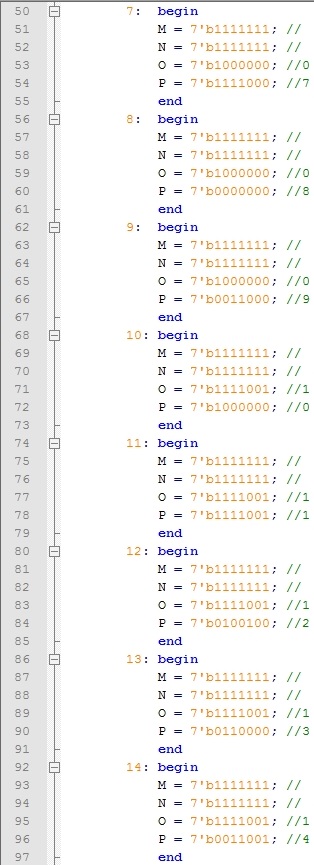
****

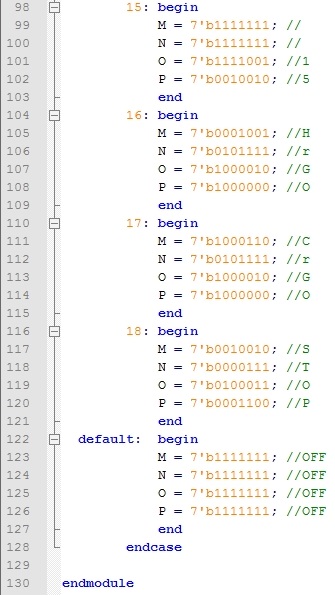
**Sub-module clock delay**

****

**Sub-module traffic 7 segment**

****

****

****