

CS303 – Term Project Report İlker Gül 26352

1. Overview

In the term project, we are assigned to design a simple telephone conversation by having a sequential circuit. Since we have the verbal description of what we are going to do, we should proceed with having a state machine. We can select between Moore or Mealy machine since it was not given as a prerequisite such that we need to perform a specific State machine. I choose to use a Mealy machine since input is also affecting the output of our circuit. I designed my State machine with 6 states, which are IDLE, Ringing, Reject, Caller, Callee and Cost.

IDLE state is the initial state of the machine. When we have a default case, needed to start a procedure again from the beginning or when we have a reset demand the state that we are returning is the IDLE state. When we have startCall request we will go to the Ringing state else we will stay in IDLE state

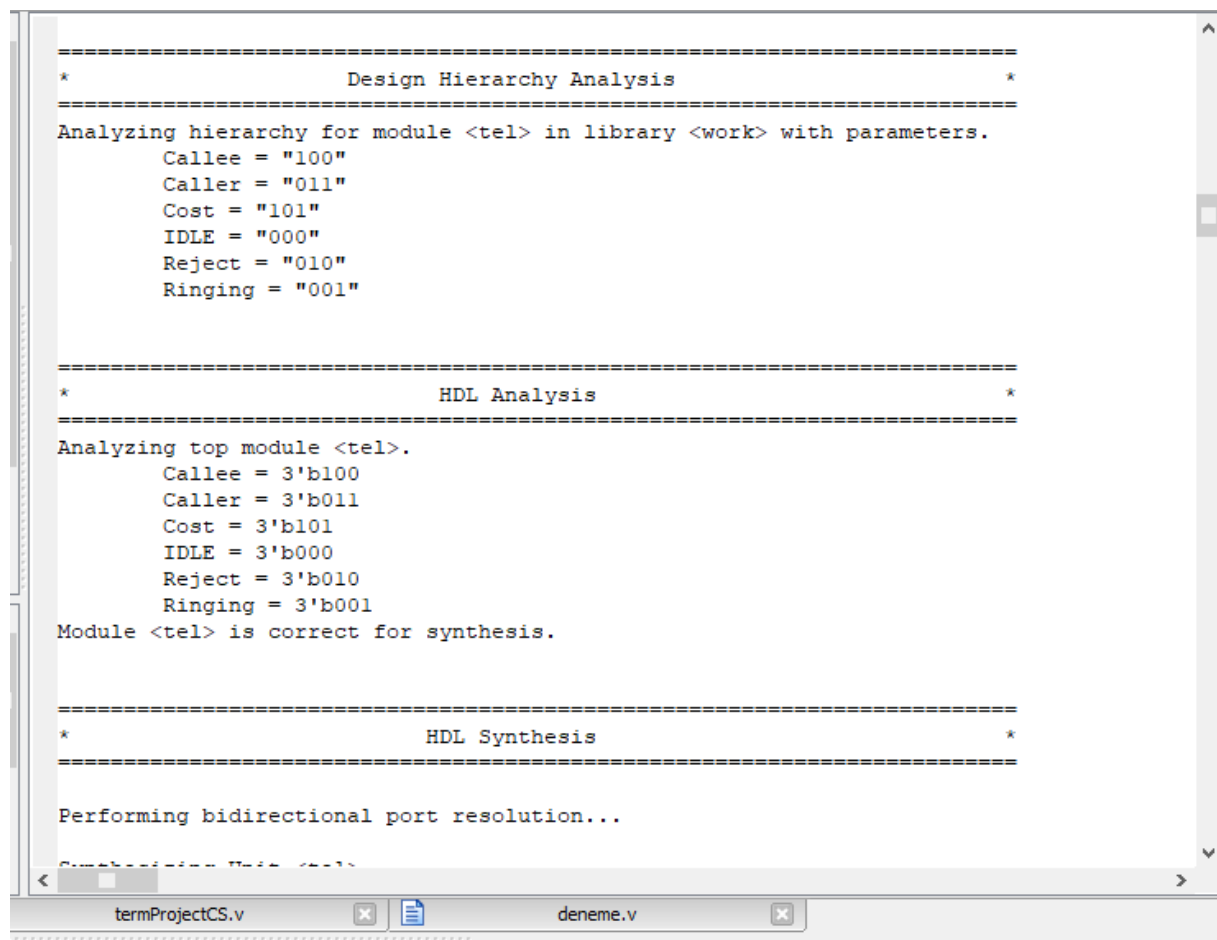
In Ringing state, we will also use a counter to count the total clock cycle that we stayed in this state. If there is no ending request from caller or callee and no answering a call request, we will stay in Ringing state. If we stayed in Ringing state for 10 clock cycles and no request for changing the state, we will return to IDLE state. If there is an end request from caller before answerCall or calleeEndCall we will return to IDLE state. If there is an end request from callee we will go to Reject state. If there is an answerCall request, then we will go into Caller state. For default cases, we will go back to IDLE state.

In Reject state we will stay here for 10 clock cycles then, we will go back to IDLE state.

In Caller state, if there is a sending char request from caller and the character that wanted to be sent is not equal to [DEL] we will stay in Caller state if there is no ending call request from each side. Else, we will go into Cost state since the call has finished. If the character is [DEL] and no ending call request, then you will go into Callee state. Transitions for Callee state is similar to Caller state. The difference is that when we have [DEL] and no ending call request you will go into Caller state. When there is an ending call request, we will go into Cost state from Caller and Callee states.

In Cost state, we will stay for 5 clock cycles and then we will go back to IDLE state. The tricky part is you need to output cost in a hexadecimal manner, but once you get the insight, it is very easy to implement.

Moreover, we need to keep total cost of the call and a counter to check the total amount of clock cycles for Reject, Ringing and Cost state. We use another always loop for checking and changing the counter and total Cost with respect to states and inputs. In the flowing figure you can see the aforementioned states. Xilinx ISE turned binary encoding into one hot encoding while processing the Verilog.



```
=====
*                               Design Hierarchy Analysis                               *
=====
Analyzing hierarchy for module <tel> in library <work> with parameters.
  Callee = "100"
  Caller = "011"
  Cost = "101"
  IDLE = "000"
  Reject = "010"
  Ringing = "001"

=====
*                               HDL Analysis                                           *
=====
Analyzing top module <tel>.
  Callee = 3'b100
  Caller = 3'b011
  Cost = 3'b101
  IDLE = 3'b000
  Reject = 3'b010
  Ringing = 3'b001
Module <tel> is correct for synthesis.

=====
*                               HDL Synthesis                                           *
=====
Performing bidirectional port resolution...
Synthesizing Module <tel>...

termProjectCS.v  deneme.v
```

Figure 1: States of the Finite State Machine

2. Simulation Results

I have used the 2 example cases given in the Sucourse+ to check the accuracy of my circuit. In the following figures, you will see the correctness of my design with respect to given examples.

First, I will show the general statistics of the synthesis report.

```
Unit <tel> synthesized.
INFO:Xst:1767 - HDL ADVISOR - Resource sharing has identified that some arithmetic ope:

=====
HDL Synthesis Report

Macro Statistics
# Adders/Subtractors          : 18
 32-bit adder                 : 1
 4-bit adder                  : 1
 6-bit adder                  : 8
 7-bit adder                  : 8
# Registers                   : 67
 1-bit register               : 64
 32-bit register              : 1
 4-bit register               : 1
 64-bit register              : 1
# Comparators                 : 12
 4-bit comparator less        : 8
 8-bit comparator greater     : 2
 8-bit comparator less        : 2

=====
*                               Advanced HDL Synthesis                               *
=====

Analyzing FSM <FSM_0> for best encoding.
Optimizing FSM <CurrentState/FSM> on signal <CurrentState[1:6]> with one-hot encoding.
|-----
```

Figure 2: General statistics of the synthesis report

Now, I will show the simulation results of the sample cases. I used given example cases as I mentioned before, then I changed one of them to also show the hexadecimal result with a letter in the sentMsg.

First, I will show the results in which Cost should be played as “00000021”.

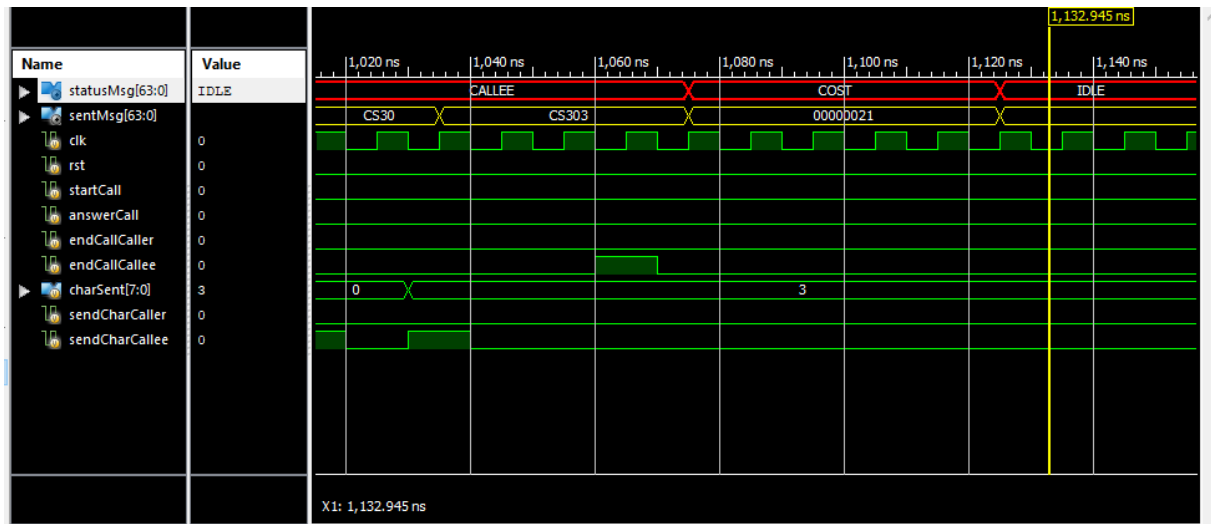


Figure 3: Result of the first example case

As it can be seen, it works correctly with correct amount of clock periods. When we got 5 clock periods, in the next clock it has a transition to IDLE as it was wanted.

Then, in the next figure, I will show the second example case which has two cost outputs.

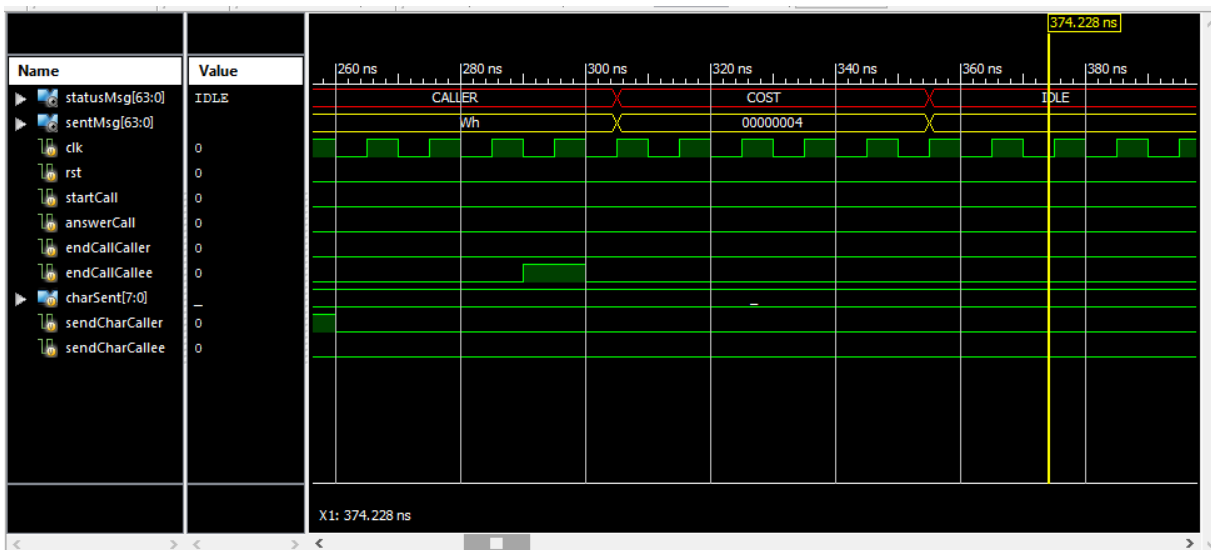


Figure 4: First cost output of the second example

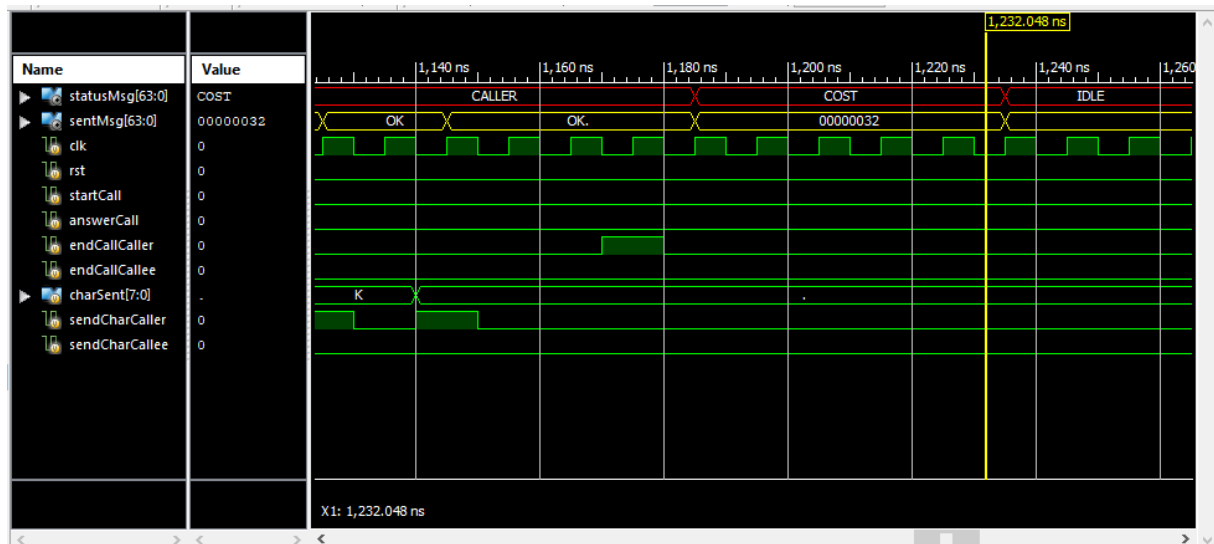


Figure 5: Second cost output of the second example

The output results for sentMsg of the previous two figures was correct according to the comments in the Verilog text fixture.

Then, I changed and some additional characters in which total cost is equal to 62 which equals to 3E in hexadecimal. Hence, we get;

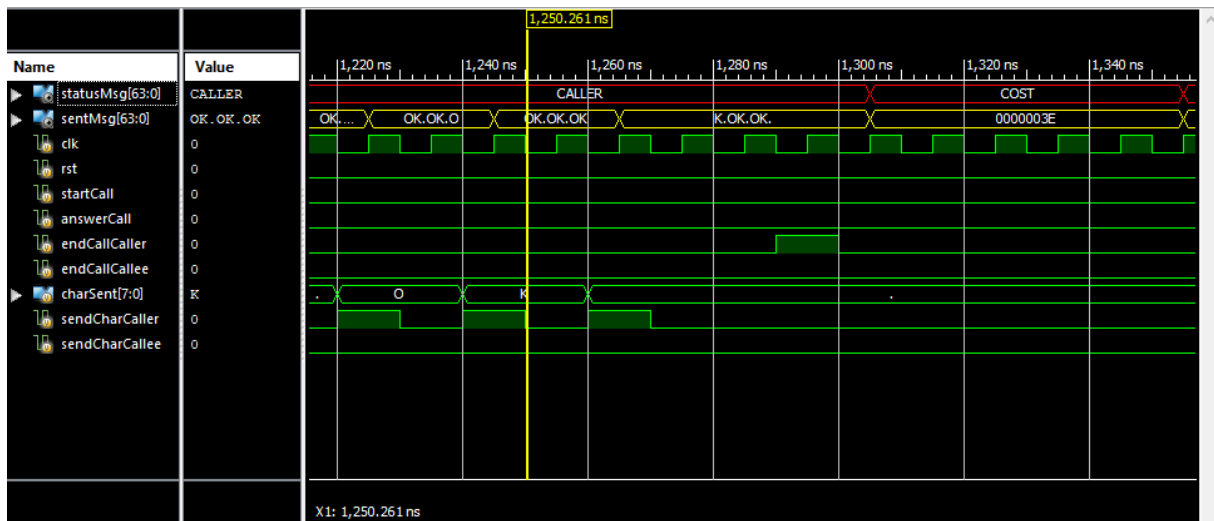


Figure 6: Cost output of the modified second example

Now, we can look at some device utilization results in which there will be time and area related information.

```

Selected Device : 3sl00etql44-4

Number of Slices:                147 out of   960    15%
Number of Slice Flip Flops:      119 out of  1920     6%
Number of 4 input LUTs:          282 out of  1920    14%
Number of IOs:                   144
Number of bonded IOBs:           144 out of   108   133% (*)
Number of GCLKs:                  1 out of    24     4%

!WARNING:Xst:1336 -  (*) More than 100% of Device resources are used

-----
Partition Resource Summary:
-----

    No Partitions were found in this design.
-----

```

Figure 7: Device Utilization Summary

We see that, we have 282 Look-up Tables with 14% utilization. Furthermore, we have seen that, we will check the timing results

```

Timing Summary:
-----
Speed Grade: -4

    Minimum period: 7.172ns (Maximum Frequency: 139.431MHz)
    Minimum input arrival time before clock: 10.442ns
    Maximum output required time after clock: 4.450ns
    Maximum combinational path delay: No path found

Timing Detail:
-----
All values displayed in nanoseconds (ns)

=====

```

Figure 8: Timing Summary

Since we have a sequential circuit, there is no unexpected timing result from our synthesis report.

Furthermore, we do not have any error in our synthesis, which is a great indicator that we will have no error when we want to implement our circuit.

In conclusion, we were trying to design and implement a sequential circuit in which we have tested what we have learned throughout the semester about combinational, sequential and state machine transitions by using Verilog. It was insightful and explanatory to extract the concepts better.