CS 303 TERM PROJECT

A Simple Telephone Conversation

2021-2022 FALL

Goktug Korkulu

27026

GENERAL OVERVIEW OF THE TERM PROJECT:

In this project, I have designed a circuit using Vivado Xilinx 2018.2 for one sided telephone conversation and then I have implemented it using Vivado HDL. Right after that I demonstrated one by one that the simulation results match with the demanded functional correctness of this project.

As the most basic understanding, there are a caller and a callee in this conversation. The caller initiates the conversation and sends characters to the callee. With my design, I will calculate the cost of the call by some specifications mentioned in the term project introduction. Right after I calculate the overall cost, I will send it as output to the callee.

Inputs;

- rst will set your circuitry to its initial state.
- startCall (1-bit) will be used by the caller and it will represent that the caller pressing a button to start a call.
- answerCall (1-bit) will be used by the callee and it will represent that the callee pressing a button to answer an incoming call.
- endCall (1-bit) will be used by the callee and it will represent that the callee pressing a button to end a call.
- charSent (8-bit) will be used to define 8-bit printable ASCII character to be sent from the caller to the callee according to printable ASCII table shown at Fig. 1.
- sendChar (1-bit) will be used by the caller and it will represent that the caller pressing a button to send an ASCII character (set by charSent input) to the callee.

Outputs;

- statusMsg (64-bit) will be used to display the status of telephone using 8 printable ASCII character. For example, statusMsg output should be “IDLE “ in its initial state. More details will be provided below.
- sentMsg (64-bit) will be used to display the last 8 printable ASCII characters sent by caller. More details will be provided below.

Operation Steps;

The circuitry will start in the IDLE state, in which it should output ‘IDLE ’ to statusMsg output (last 4 characters are space). You should use rst input as an asynchronous reset input to put the sequential circuit into IDLE state. In IDLE state, the caller can initiate a call by pressing startCall when the circuit is in IDLE state. Otherwise, the circuit will stay in IDLE state. If the caller initiates a call by pressing startCall, the telephone will start ringing. When the telephone is ringing, the 8-bit ASCII output “RINGING ” should be at output statusMsg. When the telephone is ringing, there are three possibilities:

1. If the callee rejects the call by pressing endCall, your circuit should output ‘REJECTED’ to statusMsg output for 10 clock cycles and then your circuit should go back to the IDLE state.
2. If the callee does not answer the call for 10 clock cycles, your circuit should go to the BUSY state. Your circuit should output ‘BUSY’ to statusMsg output for 10 clock cycles. Then your circuit should go back to the IDLE state.
3. If the callee answers the call and the conversation starts, there are two possibilities during the conversations:
    a. The caller sends character to the callee by setting charSent and pressing sendChar. If the caller does not press sendChar, telephone takes no input. During the caller sending character to the

callee, statusMsg output should be “CALLER “. Also, sentMsg output should be the last 8 ASCII characters sent by the caller. For example, if caller sends characters “C”, “S”, “3”, “0”, “3”, then sentMsg should be “ CS303”. Note that only ASCII characters with decimal values between 32 and 126 are considered valid input characters. If the caller enters and sends an invalid character, your circuit should ignore that character (do not include this input in cost calculation). If the caller sends the ASCI character with decimal value 127 (DEL) (include this input in cost calculation), it will be caller’s last input character.

b. The callee ends the call at any time by pressing on endCall.

Each sent character costs 2 Krş except for integer digits (0, 1, 2, 3, 4, 5, 6, 7, 8, 9) which cost 1 Krş. Your circuit should calculate total cost of a call. When the call is ended, total cost of the conversation will be sent as an output to sentMsg output in hexadecimal and statusMsg output should be “COST ” for 5 clock cycles. For example, a call with a cost of 55 Krş should be shown on sentMsg as “00000037” for 5 clock cycles. Then, your circuit should go to IDLE state.

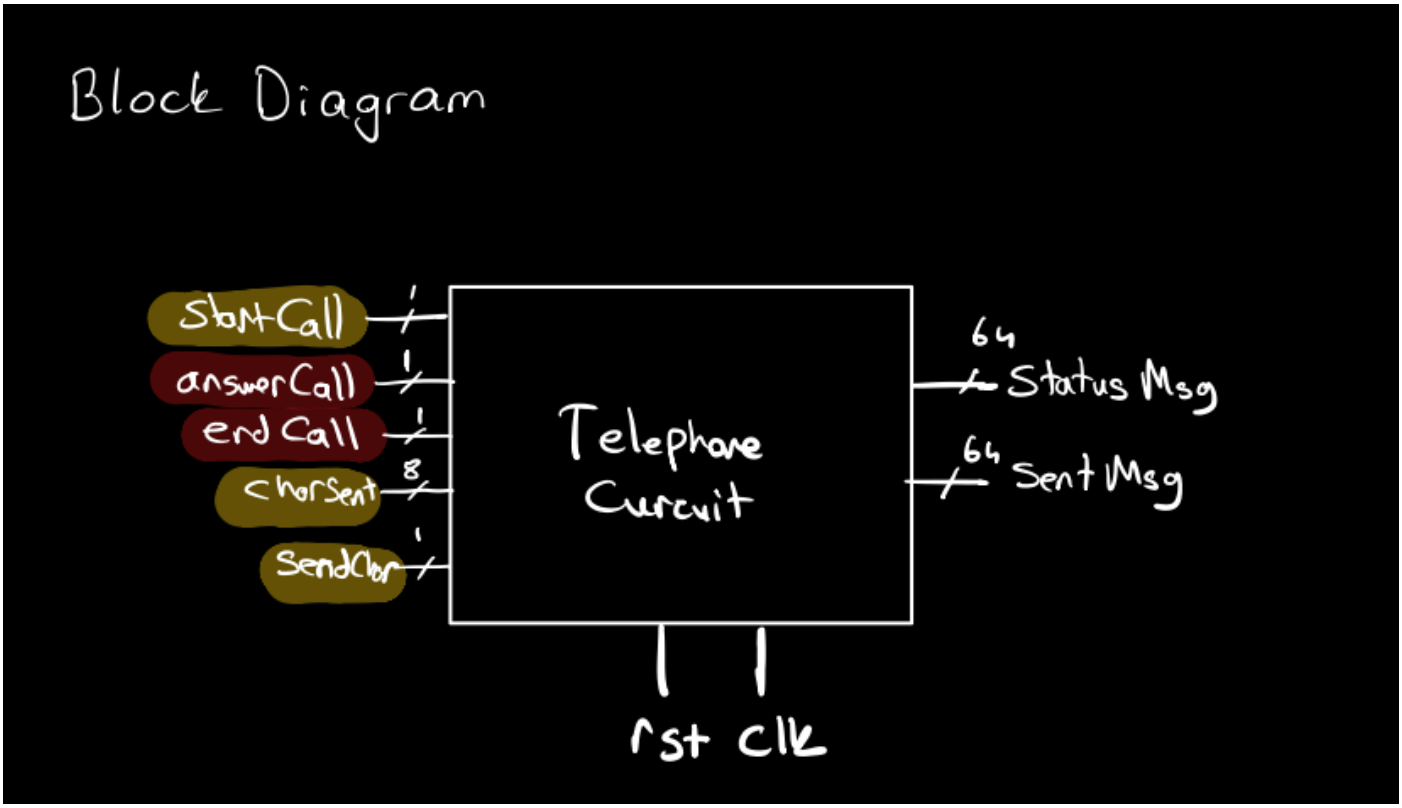


Figure 1: Block Diagram of the Overall Circuit

Definement:

İn this project, there will be 7 inputs (including reset) and 2 outputs. startCall, answerCall, endCall, sendChar, rst and clk inputs will be all 1-bit inputs where charSent input will be 8-bits. Also, the outputs StatusMsg and SentMsg will be 64-bits length. This circuit whose block diagram stated above will do all functionalities demanded above.

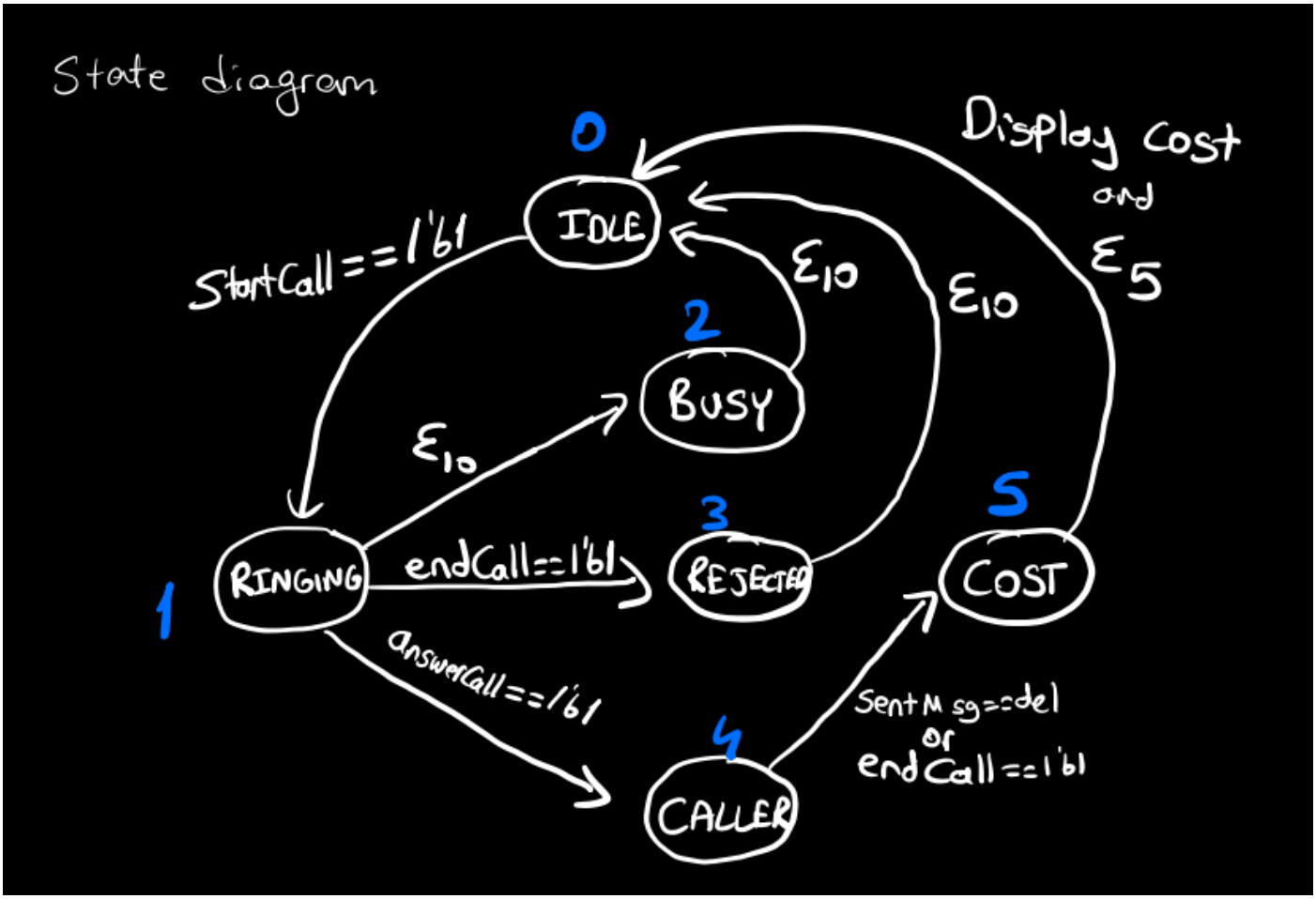


Figure 2: State Diagram of the Overall Circuit

Definement: the state diagram corresponds to all the specifications stated and demanded above.

First of all, when we receive rst = 1, we get in the IDLE state. While transmitting from IDLE state to RINGING state, we should receive only startCall input as 1, the remaining inputs will not affect the current state otherwise. For example. If we receive endCall = 1 while we stand at IDLE state, nothing will be changed.

When we are at RINGING state, there are 3 possibilities; 1: if we receive endCall = 1 at this step, we move to REJECTED state and stay there for 10 clock cycles then move to the IDLE state again. 2: if we don't receive any input change in this RINGING state for 10 clock cycles, we will go to the BUSY state and in 10 more clock cycles we will go to the IDLE state again. 3: if we receive answerCall = 1 in this step, we will move to the CALLER state.

In CALLER state, we will stay there as long as we don't receive either sentMsg = DEL or endCall = 1. İn this step caller will send characters to the callee byte by byte. Right after we receive endCall = 1 or sentMsg = DEL, we will move forward to COST state.

In the COST state, we will calculate the cost of the conversation by calculating the corresponding costs of each character sent by caller to the callee. And right after we show the cost in the hexadecimal form to the callee, in 5 clock cycles we'll go back to the IDLE state again and start everything from scratch there.

SIMULATION RESULTS

1) When press rst button,

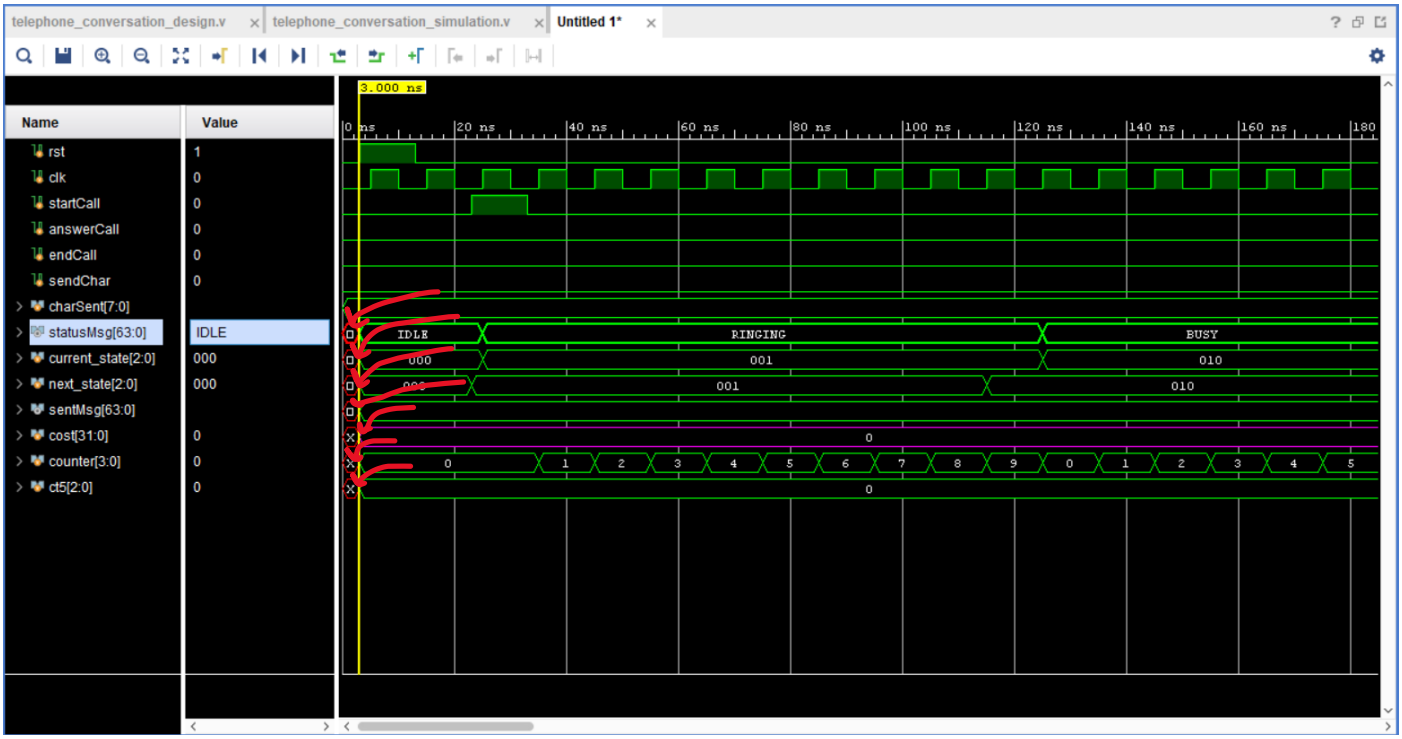


Figure 3: All outputs get their default starting position when rst is pressed.

2) startCall pressed and posedge of clk triggers it.

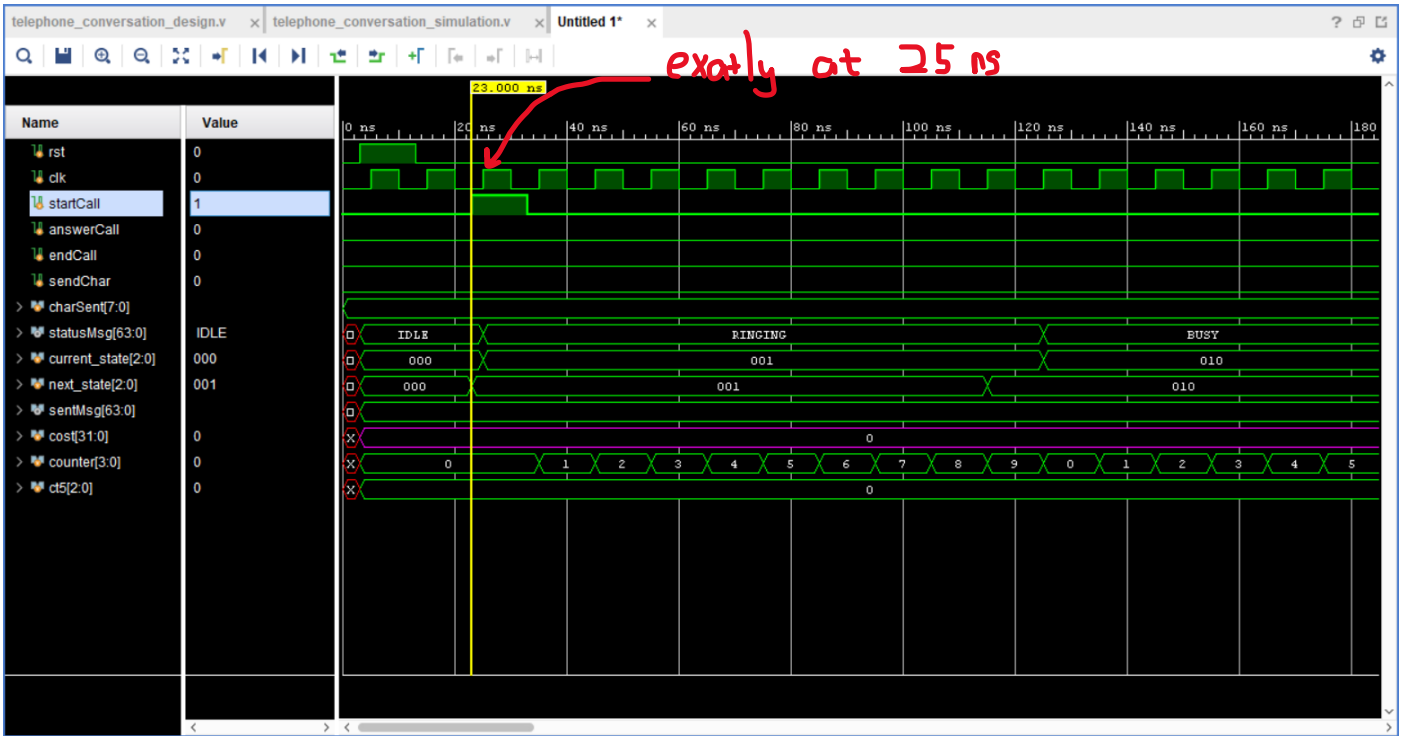


Figure 4: startCall input is pressed at 23ns and circuit is ready to move to the next state 001 determined as RINGING. It waits for positive edge of clk to do that and moves exactly at 25ns (which is posedge of clk).

3) Telephone rings for 10 clock cycles and moves to the BUSY state afterwards.

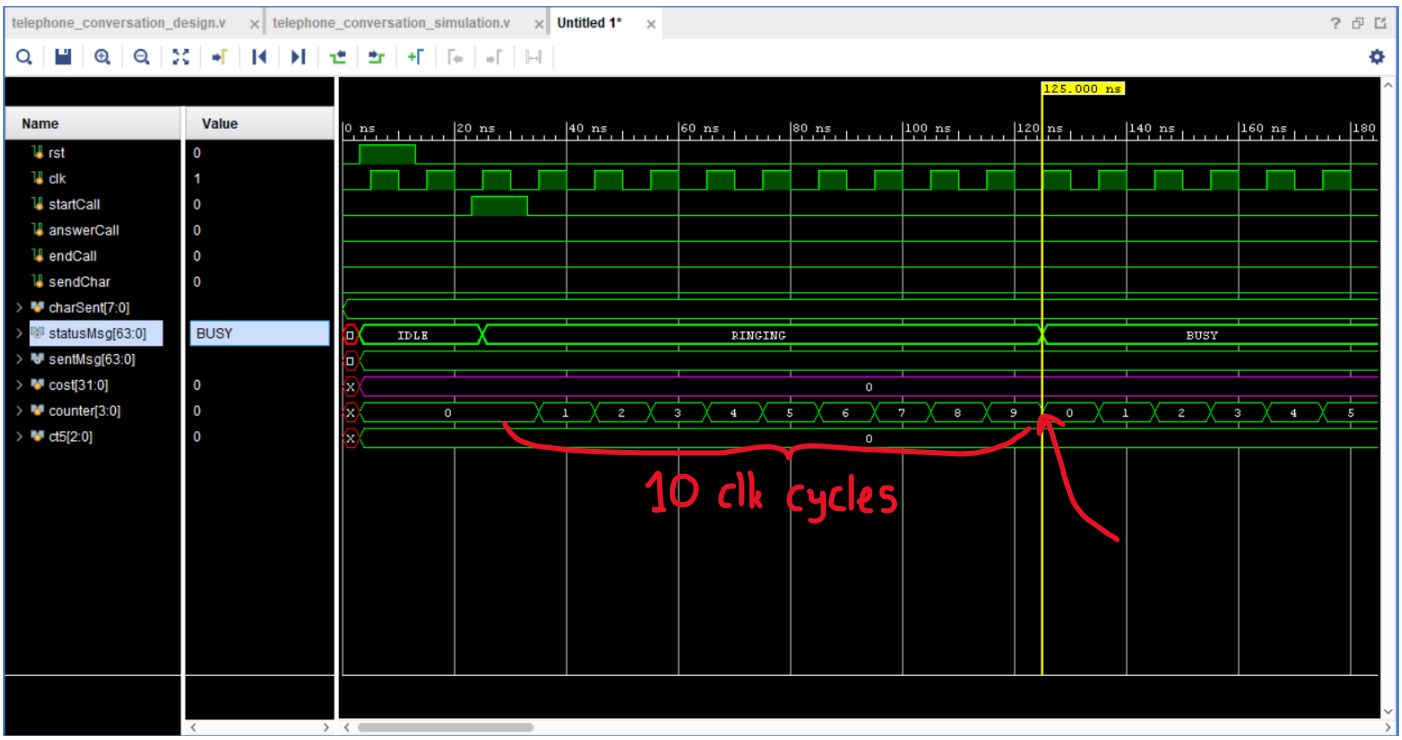


Figure 5: if no input is pressed during the counter reaches to 10 (i.e., RINGING state was active for 10 clock cycles), then automatically state moves to the BUSY state.

4) State waits at BUSY state for 10 clk cycles and moves back to default IDLE state automatically.

Figure 6: circuit waits at BUSY state no matter what for 10 clock cycles and moves back to IDLE state again.

5) Again, startCall pressed when circuit is in IDLE state. Therefore, RINGING state is initiated.

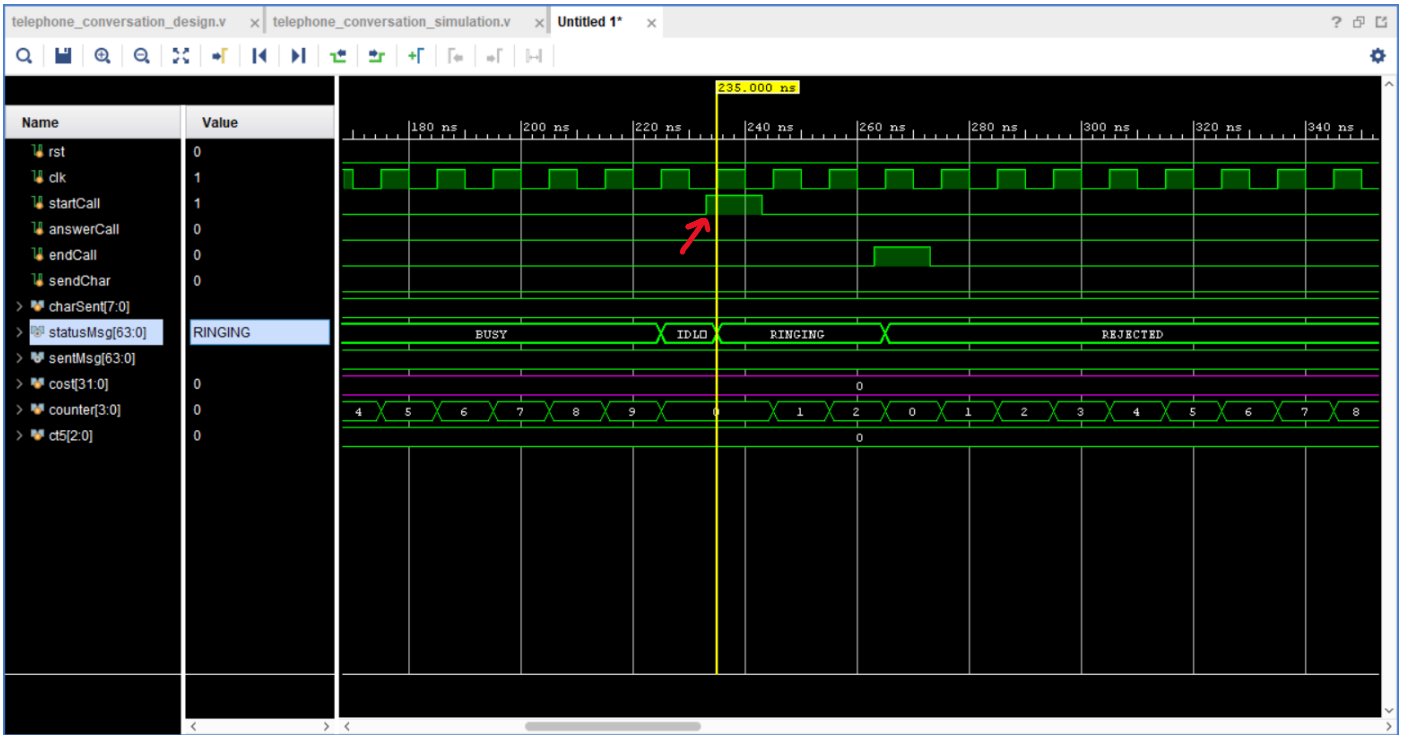


Figure 7: startCall input is pressed and at the next clk cycle circuit moves to the RINGING state again.

6) After a while, endCall input is pressed and callee directly rejects the call. Therefore, circuit goes to the REJECTED state. Waits there 10 clock cycles no matter what and moves back to IDLE state again.

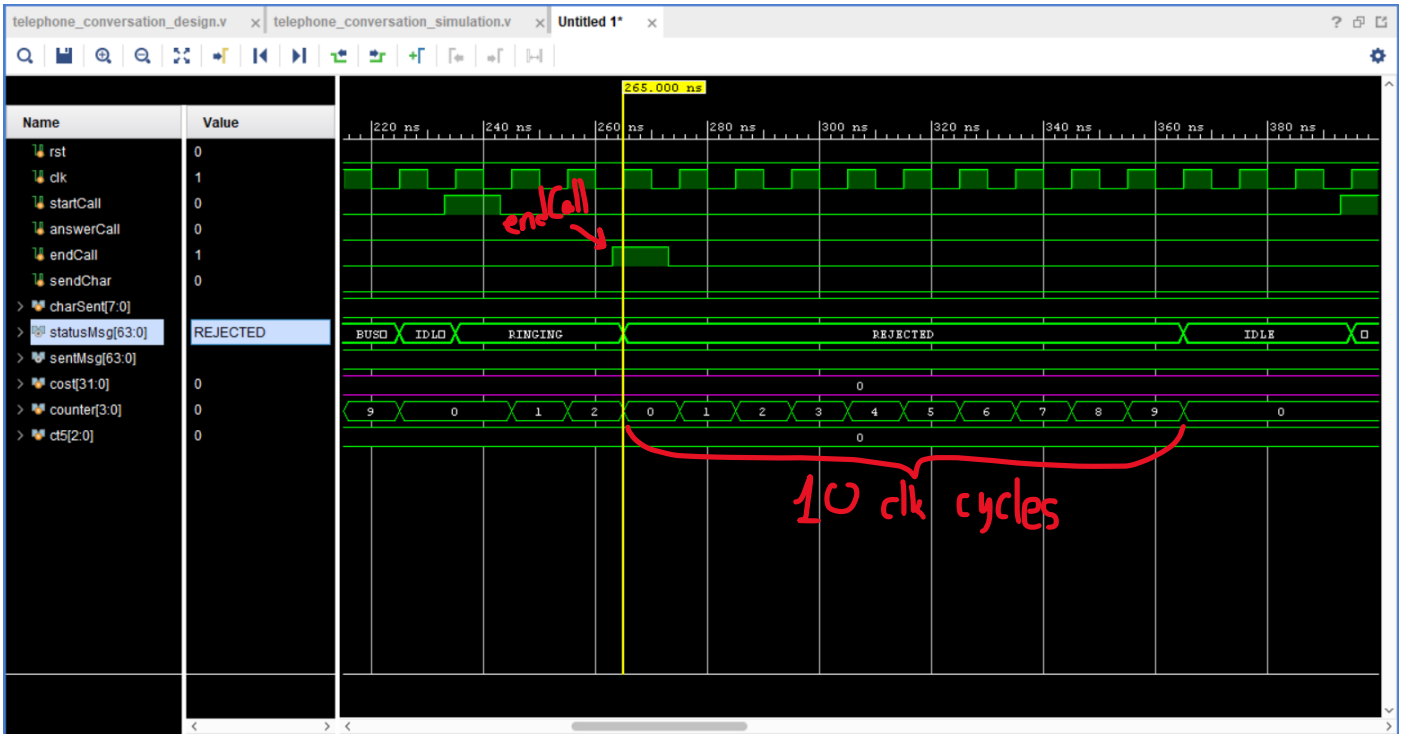


Figure 8: endCall is pressed and at the next clk cycle circuit moved to the REJECTED state. After 10 clk cycles, circuit moves back to the IDLE state.

7) After a while, startCall is pressed again and this time callee answers the phone by pressing answerCall input. Then circuit moves to the CALLER state.

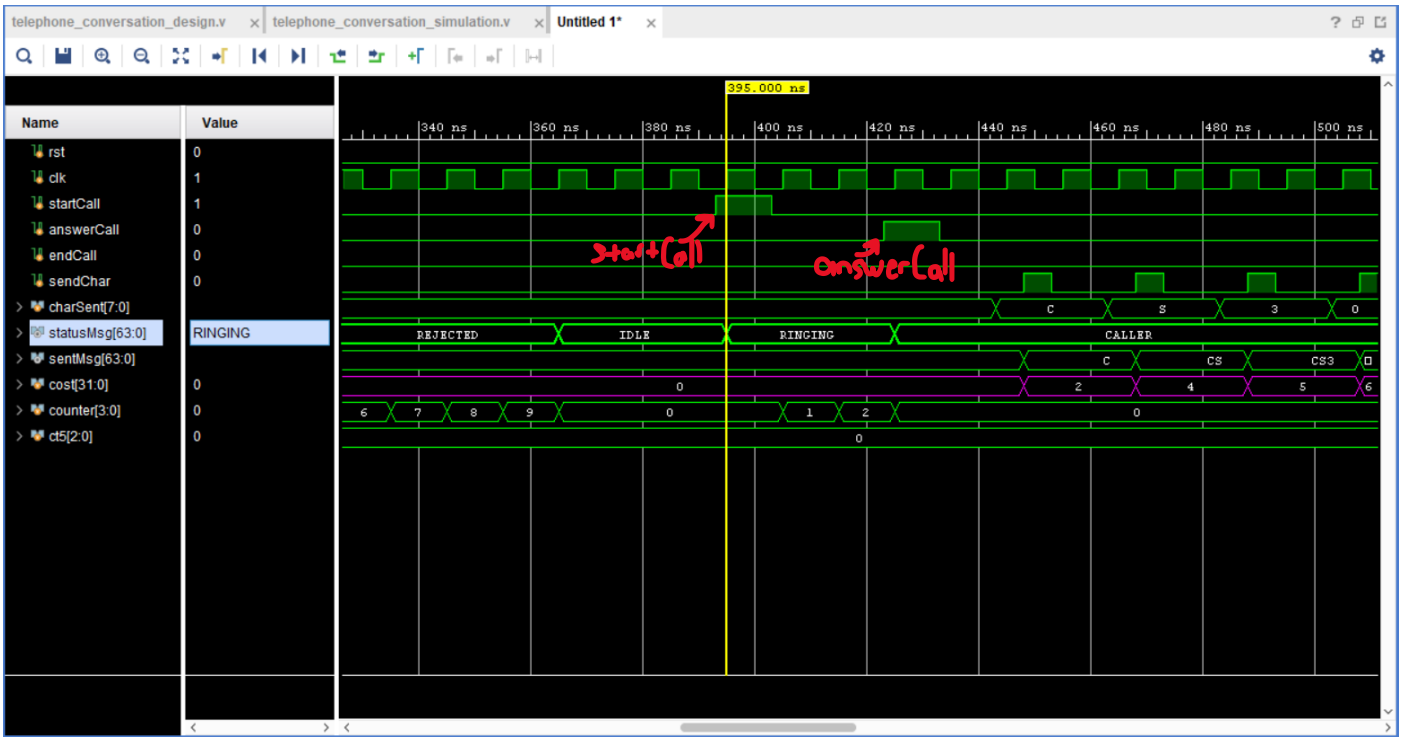
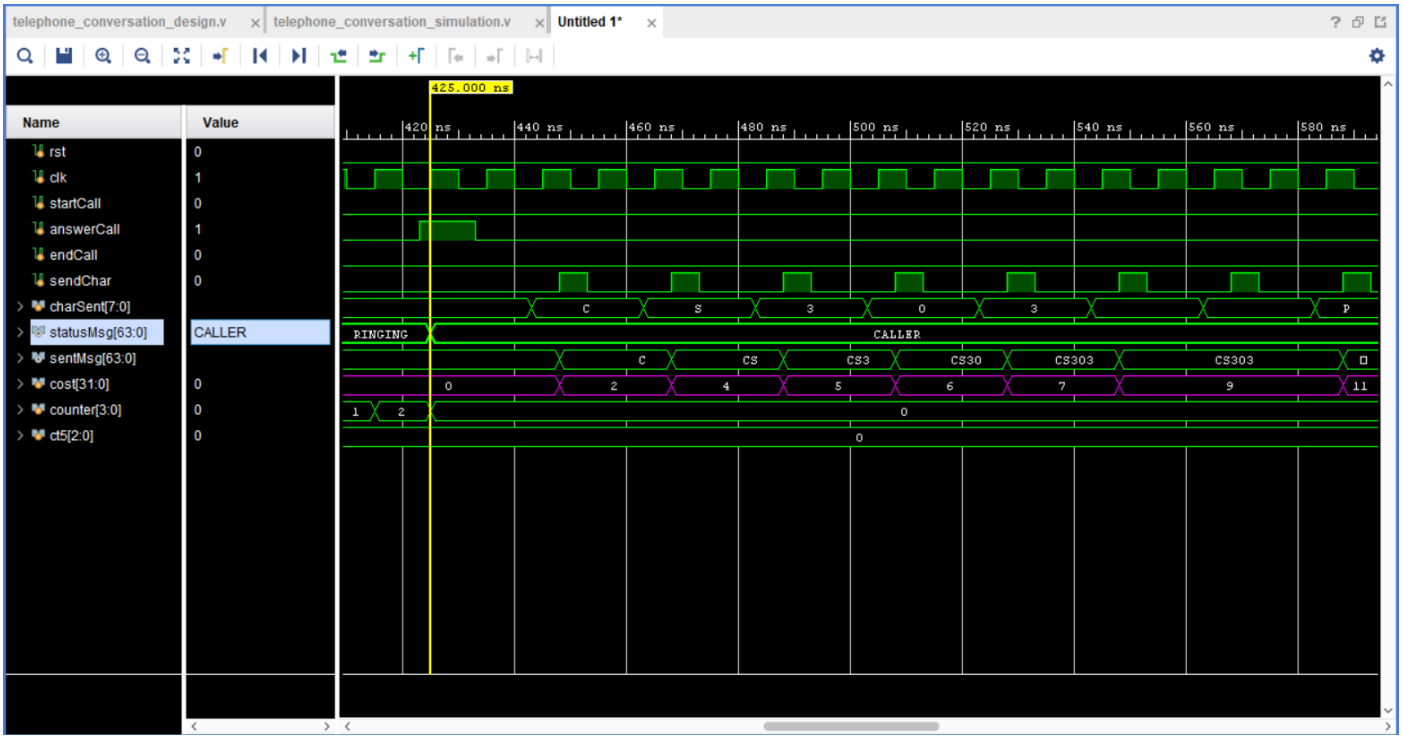


Figure 9: circuit stays at IDLE state for a while. After a while, caller press startCall input again and this time callee answers the phone by pressing answerCall. Therefore, circuit moves to the CALLER state and transmission of characters are ready to apply.

8) We're in the CALLER state



9) After a while, in the CALLER state, we enter any characters as 8-bits charSent input and send it to callee by pressing sendChar. All of the alphabetical characters cost 2 krs and all the numbers cost 1 krs to the caller.

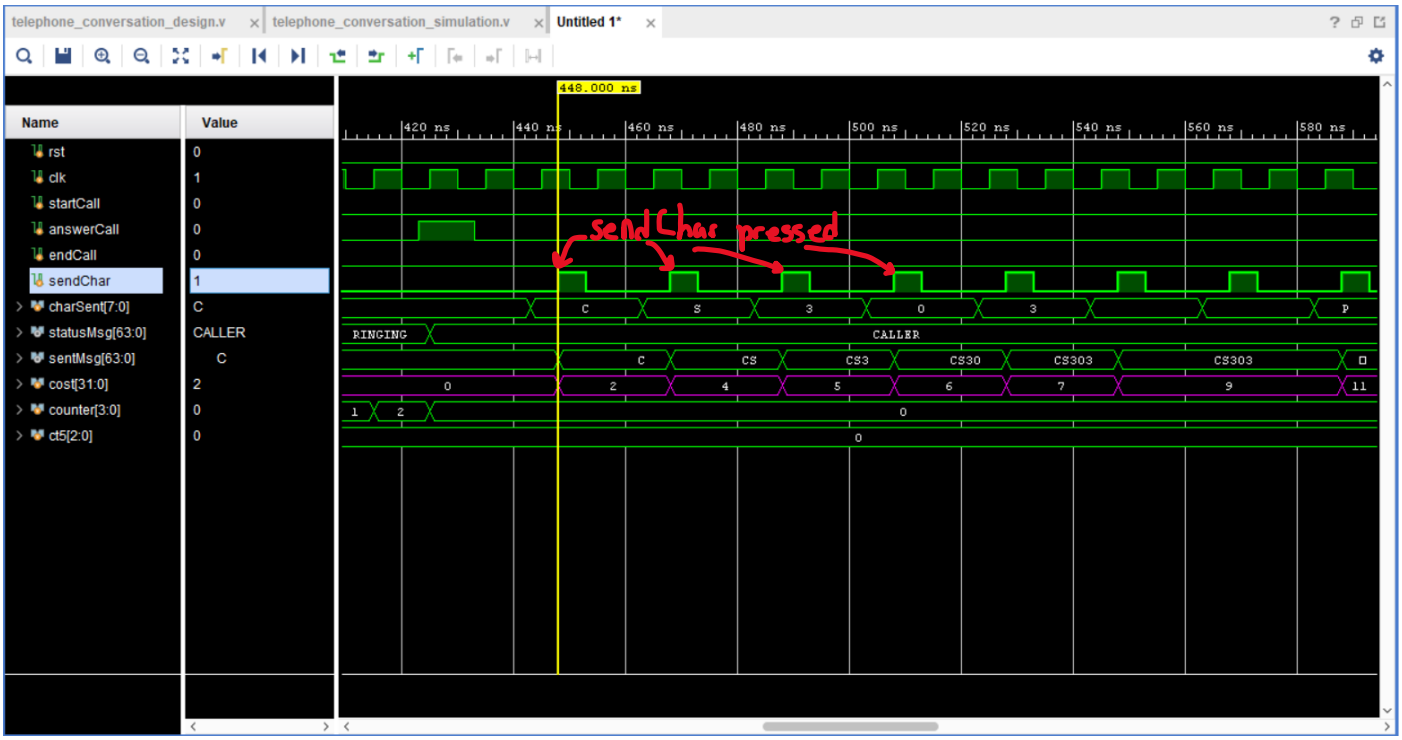


Figure 10: "C" sent and cost 2 krs. Total cost = 2.

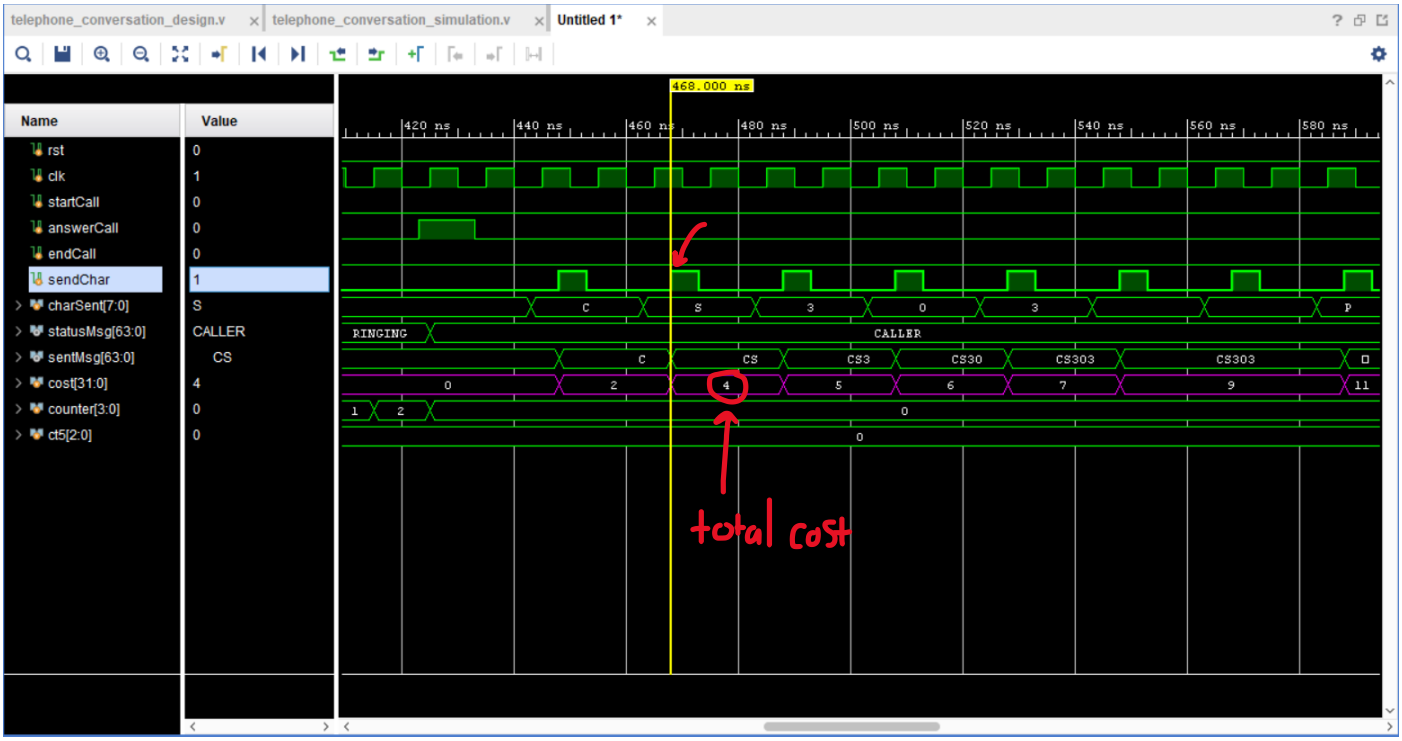


Figure 11: "S" sent and cost 2 krs. Total cost = 4 krs.

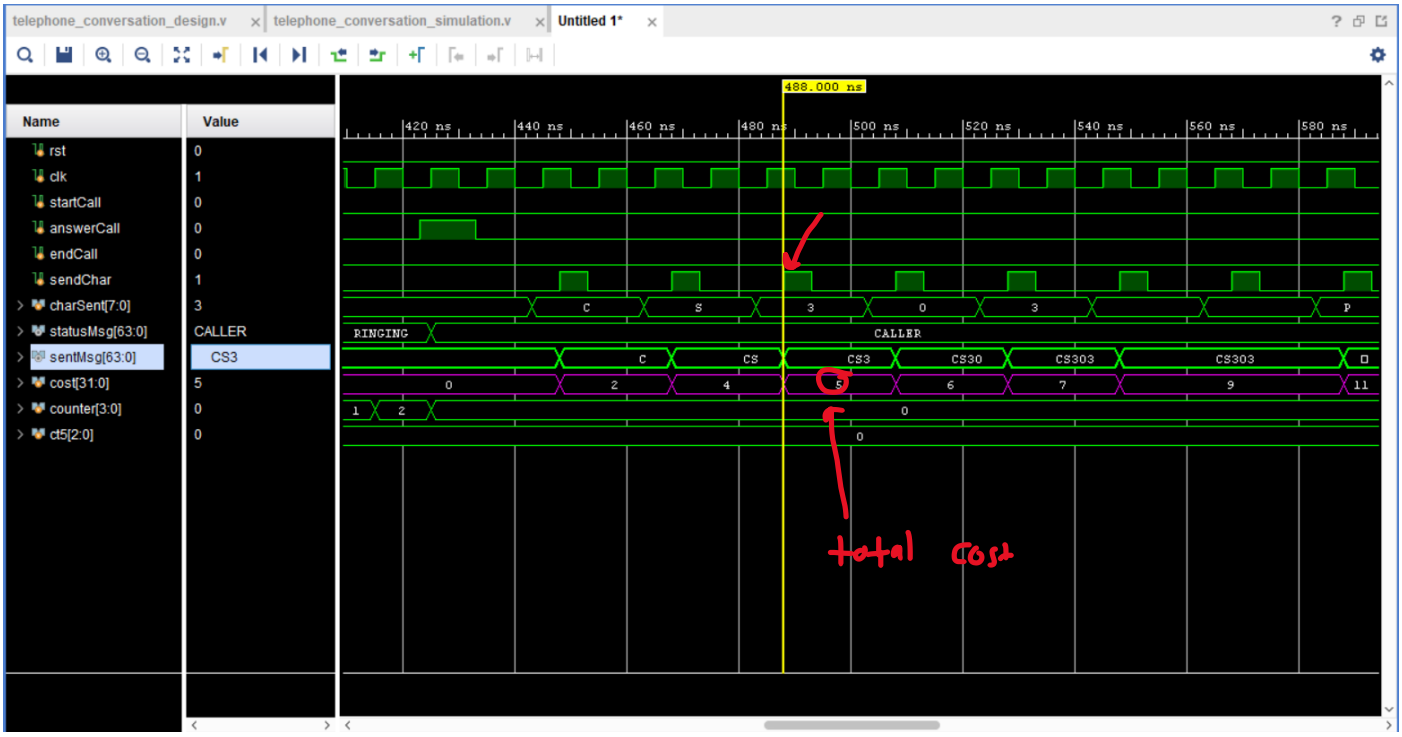


Figure 12: "3" sent and cost 1 krs. Total cost = 5 krs.

10) Notice that, when we send a dummy character (i.e., not in the demanded range in ASCII. For example, 8'd16), there is no change in total cost because we deliberately didn't include the dummy characters to the cost calculation and didn't send them to the callee.

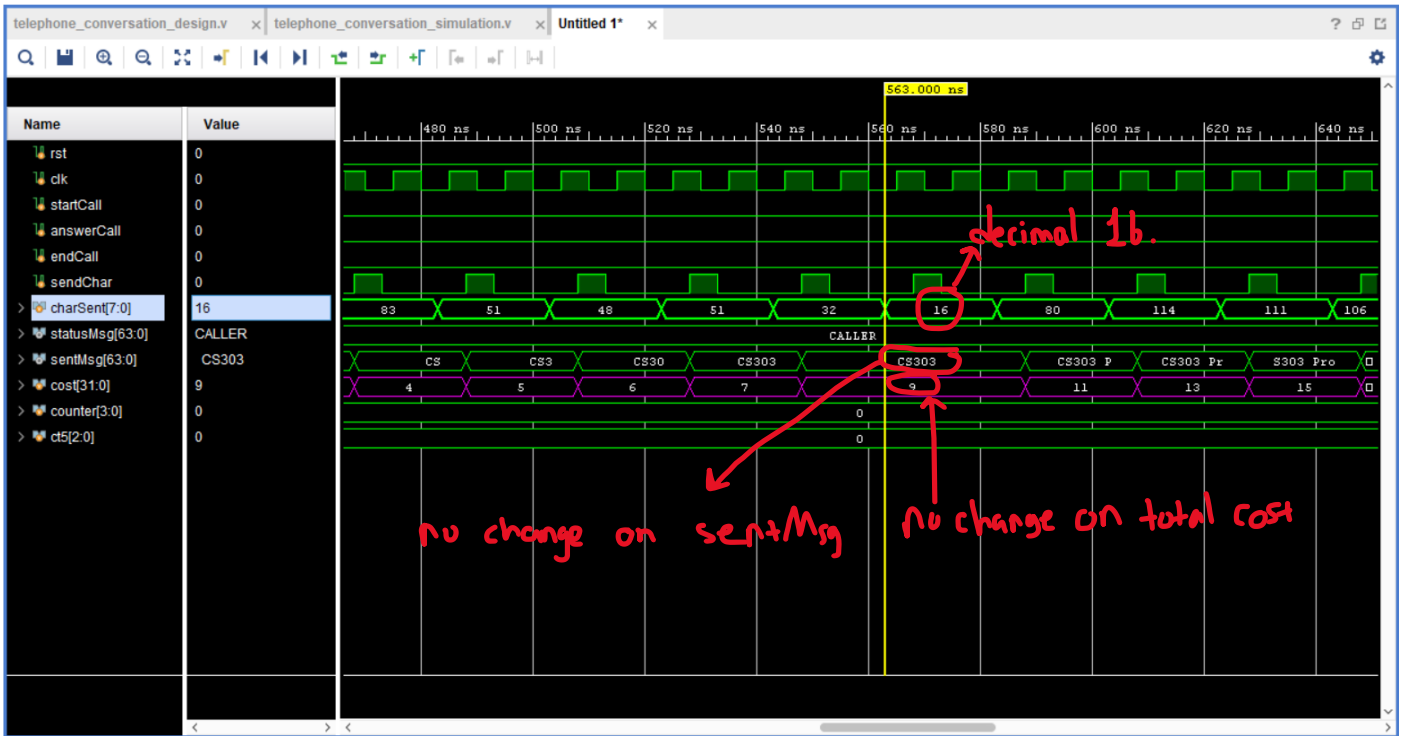


Figure 13: dummy character (8'd16) is tried to be send to the callee but failed. Neither sentMsg nor total cost is affected by dummy character.

11) At some point caller may want to end the call. And the way for caller to do that is to send 8'd127 "DEL" code to the callee. In this ss, caller sent 8'd127 to the callee and ended the call. As shown below, circuit moved from CALLER state to COST state exactly at the first posedge of clk after 8'd127 is sent.

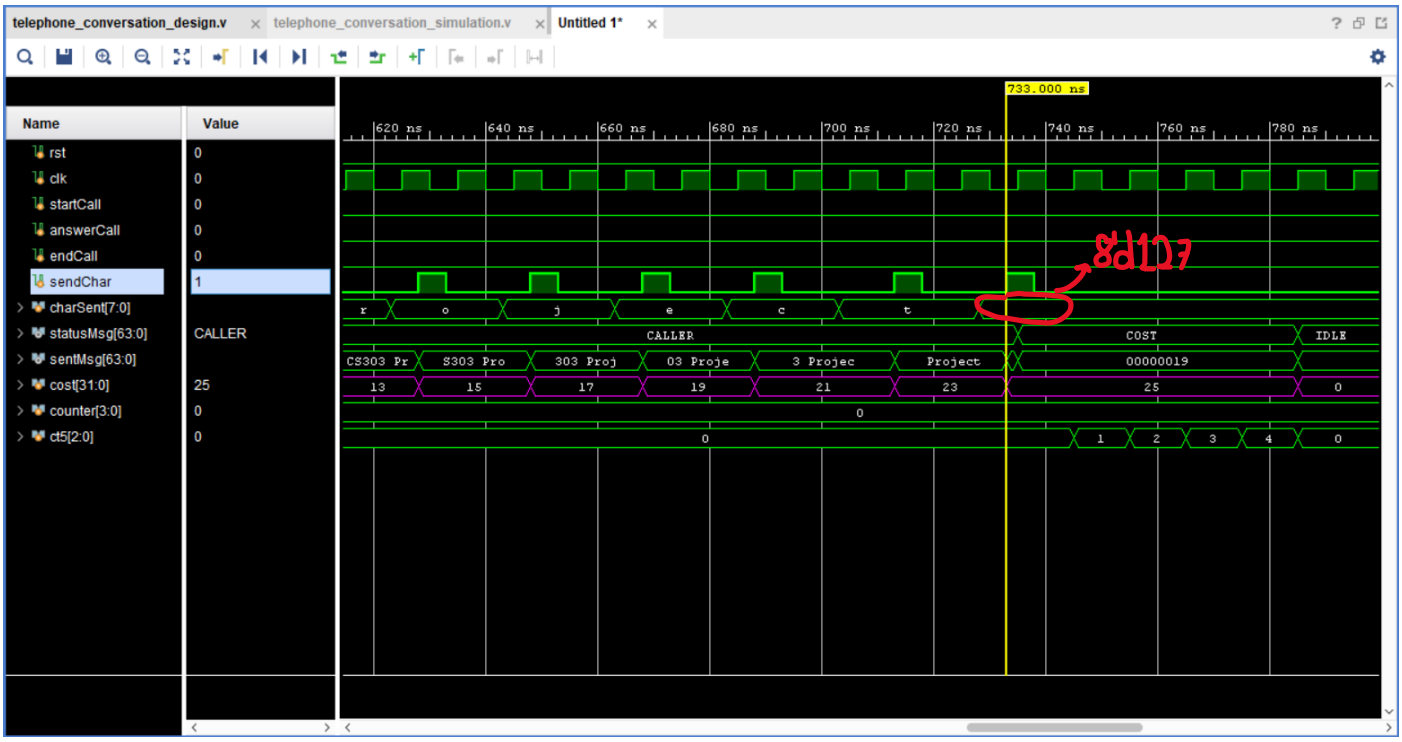
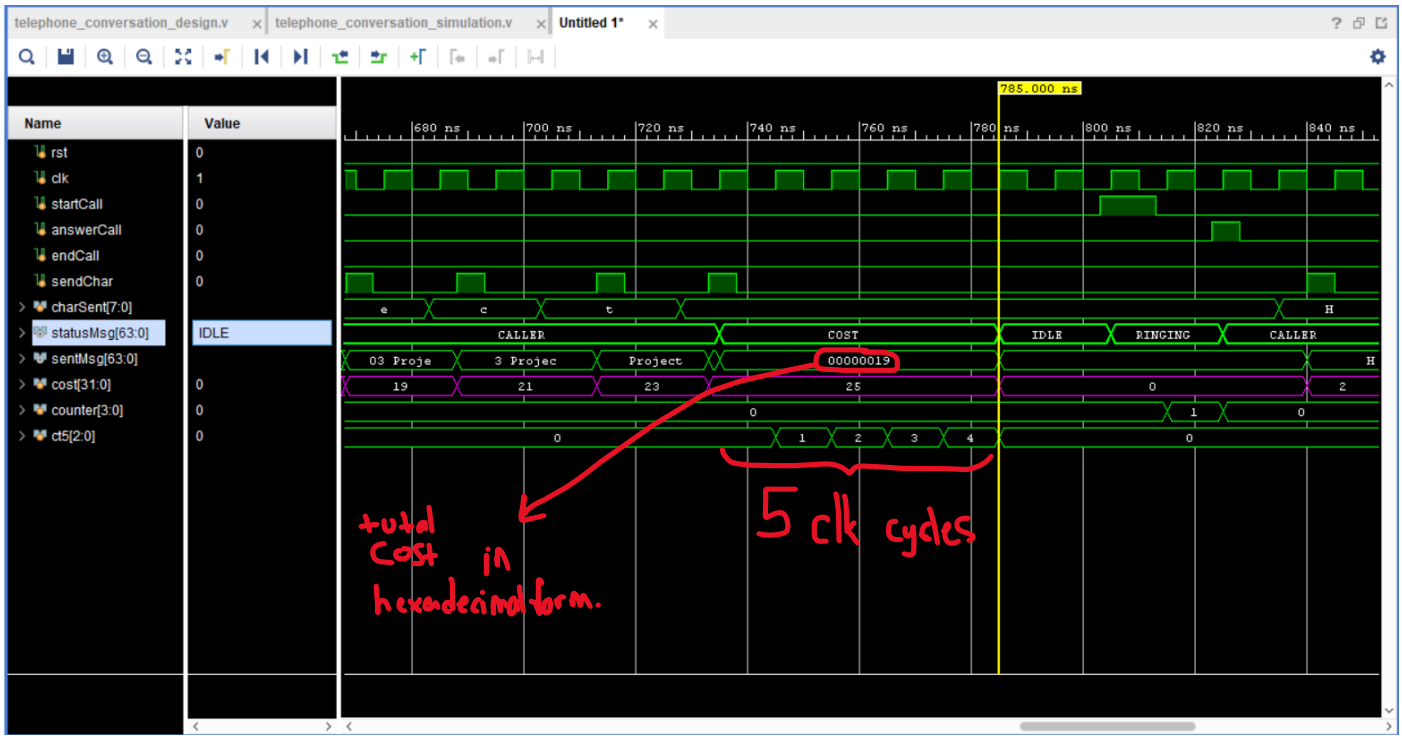


Figure 14: 8'd127 is entered as sentMsg, when sendChar is pressed, it is sent to the callee. After the first clock cycle encountered, circuit moved from CALLER state to COST state.

12) When in the COST state, total cost is calculated and converted into the hexadecimal form and displayed at 64-bits sendMsg output. After circuit waits 5 clock cycles, everything is reset, and circuit moved to the IDLE state again. And ready to use the circuit al over from the beginning.



13) After this point, circuit can be used like nothing has happened before.

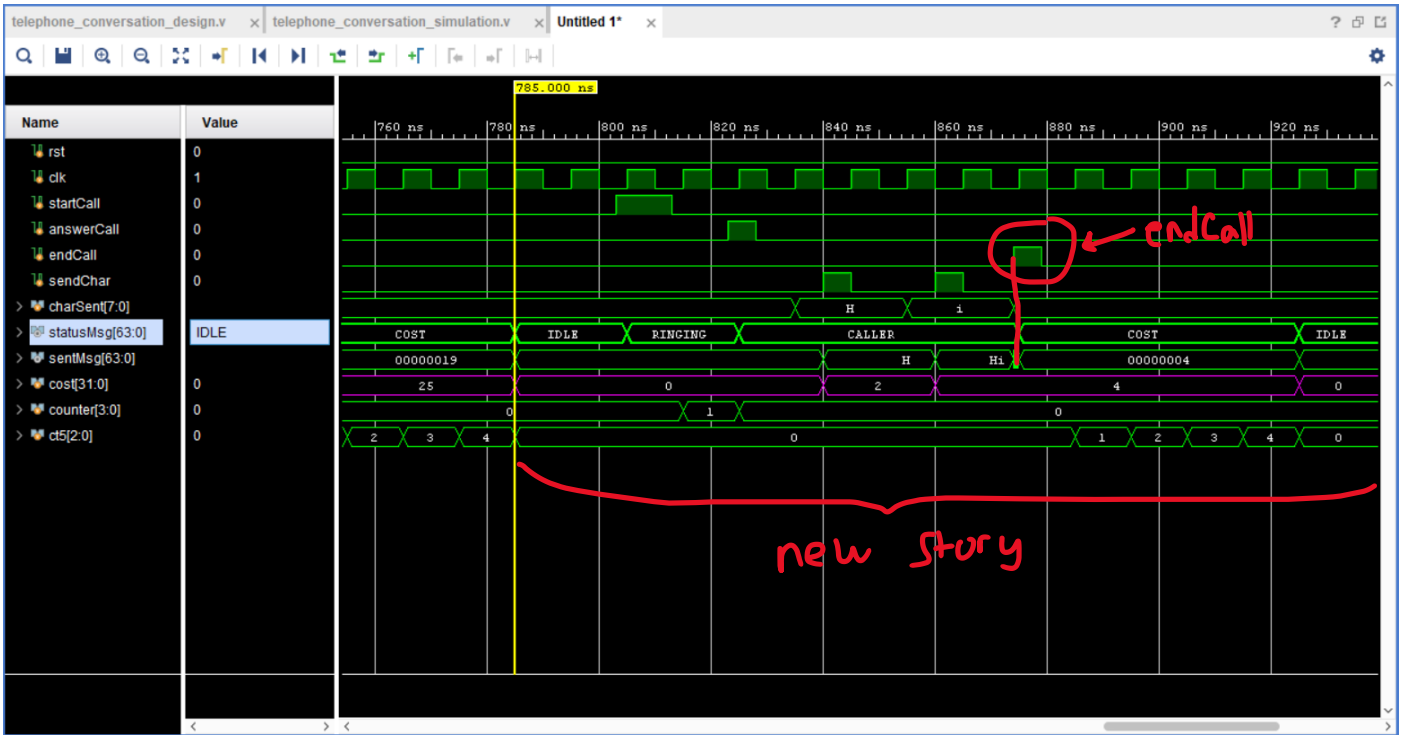


Figure 15: circuit is ready to be used from scratch. I decided to send Hi message to the callee and it costed 00000004 in hexadecimal as shown. I used endCall input in order to end the call this time as shown above. Notice that endCall didn't cost anything to the caller unlike 8'd127 code ending.

SYNTHESIS RESULT

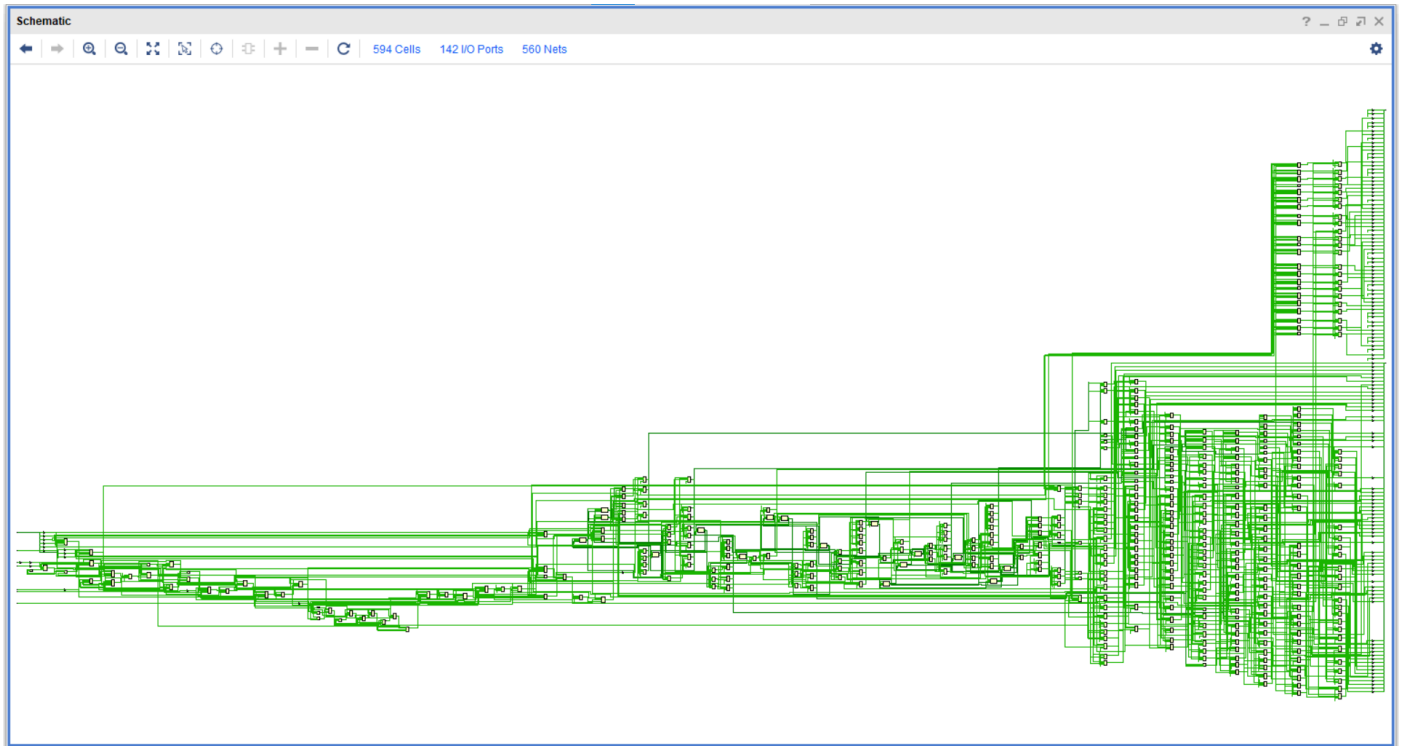
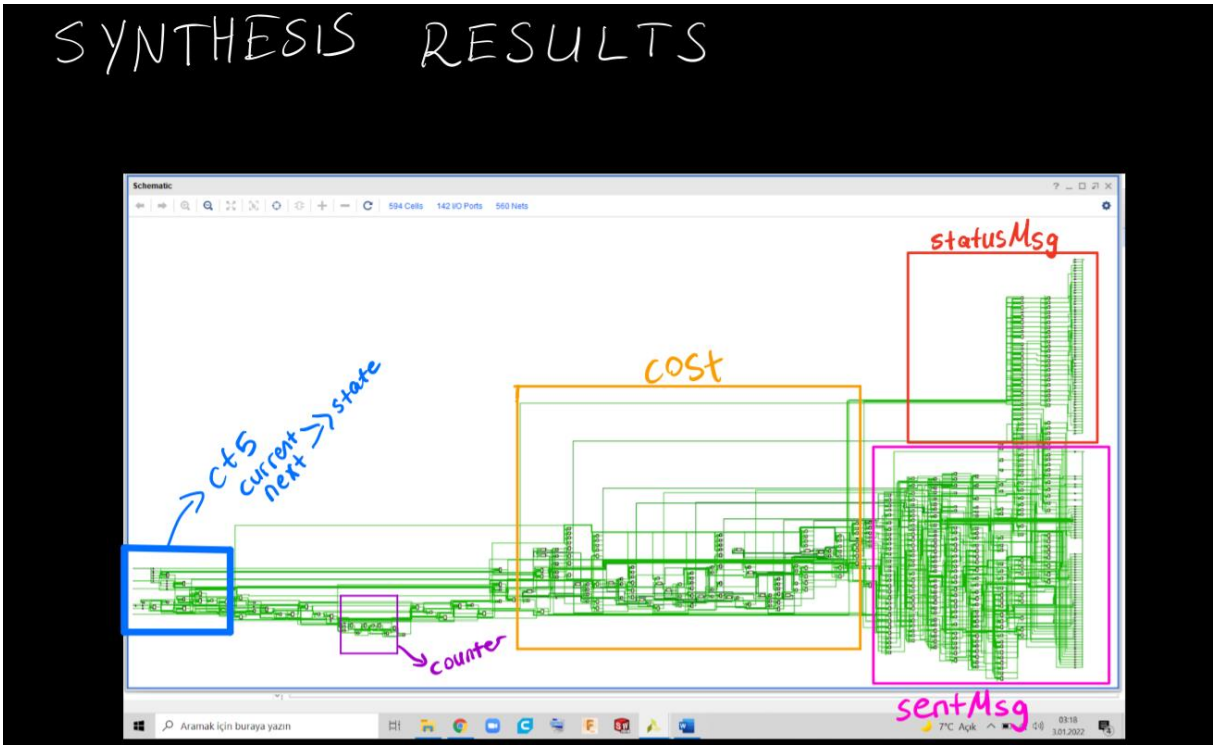
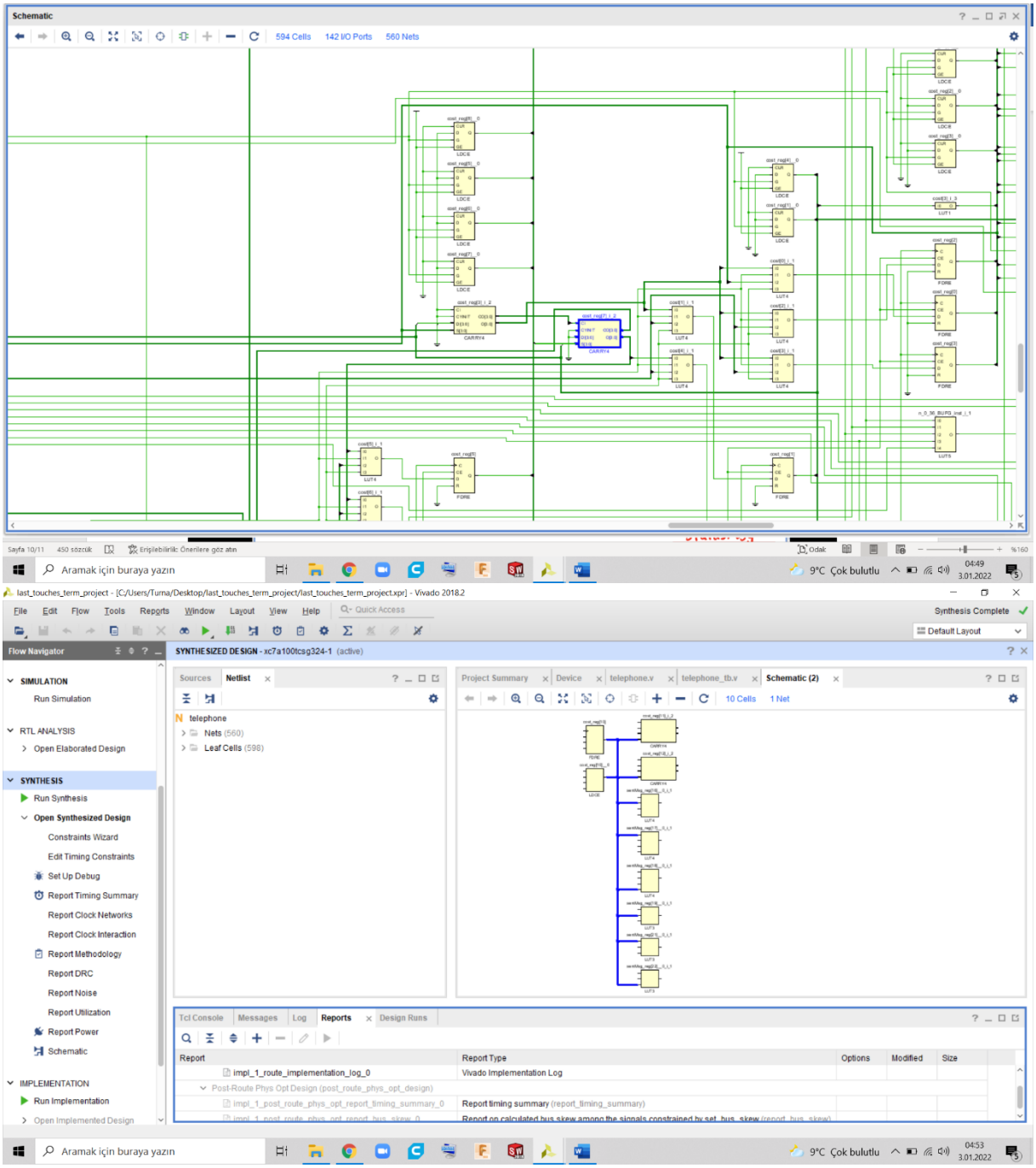
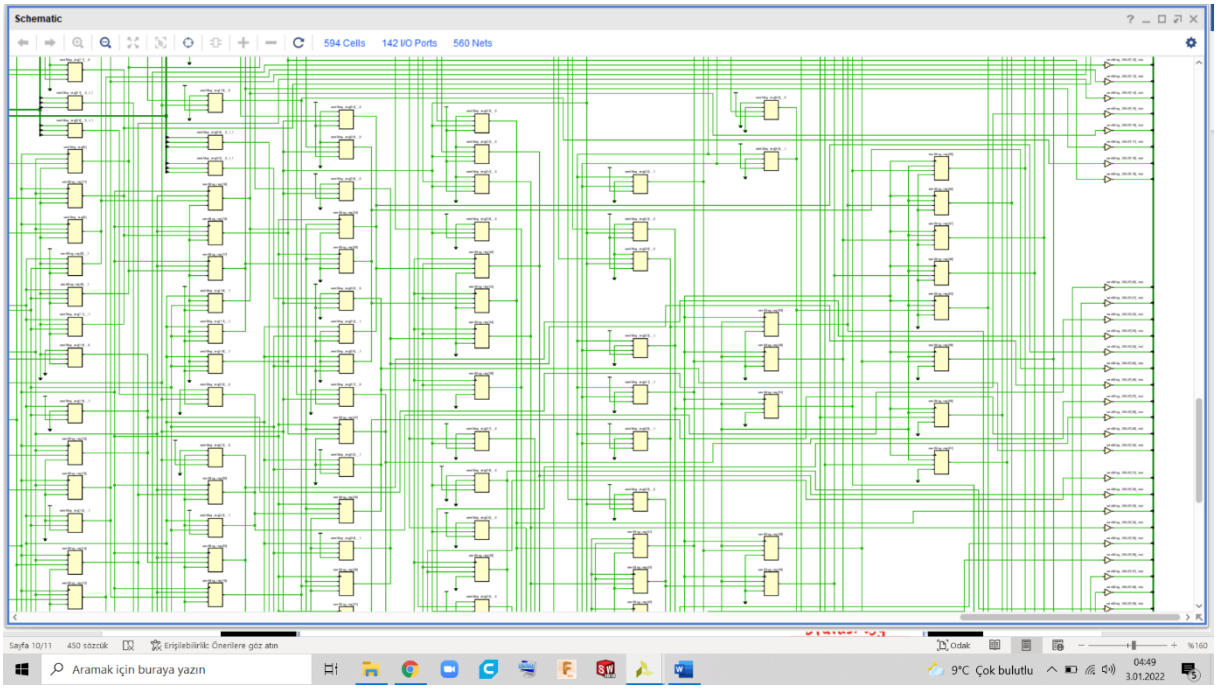


Figure 16: schematic of the entire circuit design




Schematic
594 Cells
142 I/O Ports
560 Nets

last_touches_term_project - [C:/Users/Turna/Desktop/last_touches_term_project/last_touches_term_project.xpr] - Vivado 2018.2
File Edit Flow Tools Reports Window Layout View Help
Quick Access
Synthesis Complete
Default Layout
Flow Navigator
SYNTHESIZED DESIGN - xc7a100tcsg324-1 (active)
SIMULATION
Run Simulation
RTL ANALYSIS
Open Elaborated Design
SYNTHESIS
Run Synthesis
Open Synthesized Design
Constraints Wizard
Edit Timing Constraints
Set Up Debug
Report Timing Summary
Report Clock Networks
Report Clock Interaction
Report Methodology
Report DRC
Report Noise
Report Utilization
Report Power
Schematic
IMPLEMENTATION
Run Implementation
Open Implemented Design
Sources
Netlist
telephone
Nets (560)
Leaf Cells (598)
Project Summary
Device
telephone.v
telephone_tb.v
Schematic (2)
10 Cells
1 Net
Tcl Console
Messages
Log
Reports
Design Runs
Report
Report Type
Options
Modified
Size
impl_1_route_implementation_log_0
Vivado Implementation Log
Post-Route Phys Opt Design (post_route_phys_opt_design)
impl_1_post_route_phys_opt_report_timing_summary_0
Report timing summary (report_timing_summary)
Aramak için buraya yazın
9°C Çok bulutlu
04:53
3.01.2022


Schematic
594 Cells
142 I/O Ports
560 Nets