***Digital Design Project Report***

Chen Jiyuan, Cheng Shuaihan, Fan Shun

Group 10

***Notice: We have upgraded our function on setting the number of people to participate!***

**Developing Report**

1. **Task division**

Before the specific development process, we initially divided the scope of the module that everyone is responsible for. However, during the development process, new requirements constantly appeared, so every one of us accepted extra tasks. Due to the limited time in normal times, our main development time is concentrated on the weekend. The three team members work together to complete the task and communicate with each other.

The tasks were assigned as follows:

Chen Jiyuan: design top module and manipulate the mark which players get

Cheng Shuaihan: design some displaying module and help with some other part

Fan Shun: design something to deal with countdown and answering the question

**2 . Schedule**

On the evening of December 7, 2019, the group members met for the first time and assigned the modules and tasks each person was responsible for.

On December 8, 2019, Chen Jiyuan completed several different frequency divider demo prototypes. These models played a great role in the later judgment and selection modules.

From December 13 to December 15, 2019 This is the second weekend of the project. We completed the main framework of the project, which included:

a . Fan Shun together completed the countdown module used by the answering device. At this time, we can complete a single 30 second countdown and display it on the seven-segment digital tube.

b. Fan Shun and Chen Jiyuan completed the answering part. The judgment module used includes functions such as accumulating scores, judging right and wrong, adding and subtracting scores, and judging winning or losing. At this point, the general framework of the project has been set up, and the follow-up task requirements have completely exceeded the scope of the original envisioned task. We re-discussed further task planning.

From December 17, 2019 to December 19, 2019

a .Chen Jiyuan implemented the keyboard driver on the development board and changed the judgment logic to enable us to use the keypad to select the specific score value for addition and subtraction;

b .Fan Shun connected the buzzer module and applied the top-level combination module. And the judgment module has been improved, so that the project can play the pre-designed buzzer music during the countdown;

c. Countdown module has been improved by Chen Shuaihan so that we can choose the countdown time for each round of response;

d .Fan Shun and Chen Jiyuan further improved the initial The display module can now display the welcome characters at startup;

e .Chen Shuaihan made an improvement so that between the rounds of the game we display the current player's score, scroll after each game when a player get 10 marks to show the score of each player, and finally the winner's score. At this point, project development is largely complete.

**3 .Responsibility**

Everyone is responsible for the modules they develop, and the responsibilities are assigned as follows:

CountDown: Fan Shun,Cheng Shuaihan

Answer\_Parts: Chen Jiyuan ,Fan Shun

Beeper: Fan Shun

Final\_Output: Cheng Shuaihan

KeyBoard: Chen Jiyuan,Cheng Shuaihan

ShowHello: Chen Jiyuan,Fan Shun

Display\_Mark: Fan Shun,Chen Jiyuan

FreqencyDivider: Chen Jiyuan

LiangZhu: Fan Shun

ProceedMark: Chen Jiyuan

**Designing Report**

1. **Demand Analysis**

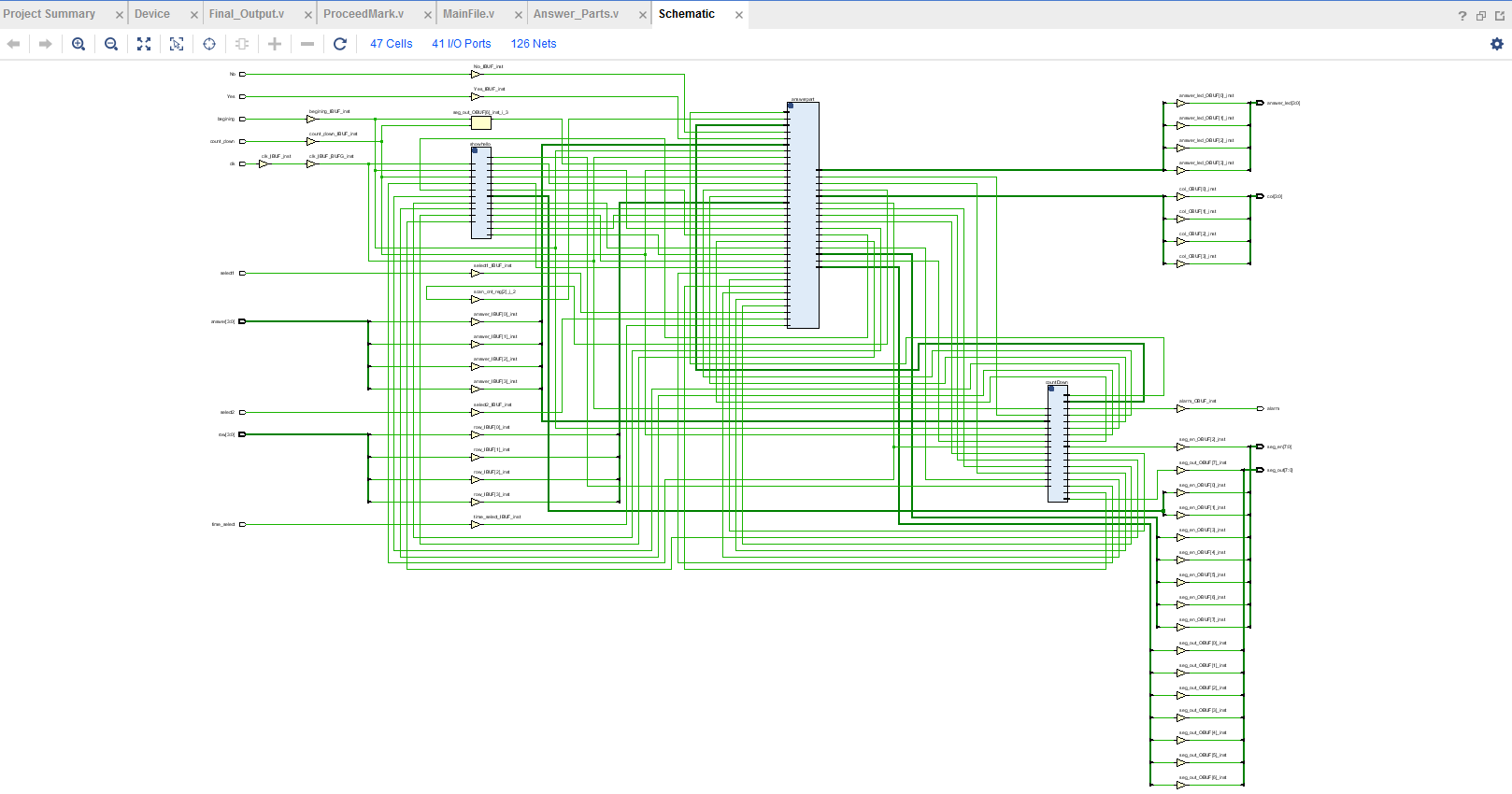
Our task requirement is to implement a multi-person answering machine:The host can give out questions and we can do answers within limited time.After the game we can see the final winner with his score. Our needs are as follows:

Input: (a)The system clock is used to implement countdown, buzzer sound and other functions; (b)One DIP switch is used to indicate the reset of the whole game;(c) 1 DIP switch is used to indicate the reset of each round answer; (d)2 DIP switches are used to indicate whether the answer is right or wrong;(e)1 DIP switch is used to select the countdown time (20s and 30s) of this round; (f)4 DIP switches are used to indicate the player's answer;(g) 9 numbers on the keyboard are used to indicate the scores added and subtracted in this round.

Output: (a)The seven-segment digital tube is used to display the “hello” page, the current time left for this round, the scores of the current rounds player after each round is completed,(b)Beeper is used to play Background music;(c) 4 led lights are used to indicate a player in the current answer round.

1. **System structure diagram**

This picture is the RTL analysis structure diagram of the top-level module, including ten sub-modules. The interface and functions of each module are explained in great detail in the code comments. Please open the source code file to view it.



Inclusion relationship between modules

MainFile.v(beginning, count\_down,clk,time\_select,Yes,No,row,answer,select1,select2,col,seg\_out,seg\_en,answer\_led,alarm)

1.CountDown.v (rst,clk,time\_select,DIG,Y,alarm,ifOver)

1)FrequencyDivider.v(clk,rst,newclk)

2)LiangZhu.v(clk\_50M,rst,speaker)

3)Beeper.v(clk\_in,rst\_n\_in,tone\_en,piano\_out) 2.Answer\_Parts.v(clk,rst,reload,beginNew,Answer,Yes,No,row,select1,select2,col,DIG,,seg\_out,getter,led,alarm)

1)FrequencyDivider.v(clk,rst,newclk)

2)Beeper.v(clk\_in,rst\_n\_in,tone\_en,piano\_out)

3)ProceedMark.v(clk,rst,answer,Yes,No,getter,row,reload,select1,select2,col,DIG,seg\_out)

[1]FrequencyDivider.v(clk,rst,newclk)

[2]Display\_Mark.v(rst,clk,Answer,mark\_one,mark\_two,mark\_three,mark\_four,DIG,Y)

[3]KeyBoard.v(clk,rst\_n,row\_data,key\_flag,key\_value,col\_data) [4]Final\_Output.v(select1,select2,winner,score1,score2,score3,score4,seg\_en,seg\_out)

3.ShowHello.v(rst,clk,DIG,Y)

4.FrequencyDivider.v(clk,rst,newclk)

1. **System execution process**

The text description of the system execution process is as follows:

1. The entire system is started by a reset signal, corresponding to a DIP switch. After startup, the system clk signal is connected to the entire project, and at this time enters the top-level module Mainfile.v. The top module passes the reset signal to the ShowHello.v module, which is used to display the word “hello” on the seven-segment digital tube.
2. The host first uses a DIP switch to select the current counting time. This signal is passed to the CountDown.v module to select whether the duration of a round answer is 20s or 30s. After that, the host used two DIP switches to select the number of players. The signal for selecting the number of players was passed to the Answer\_Parts.v module to indicate the number of players in the competition. The following section explains this module in more detail.
3. The host dials the beginning signal to start the call. This signal is asserted. This signal is passed to ShowHello.v to terminate the word “hello” that is displayed. At the same

time, the beginning signal is also passed to the CountDown.v module, which is used to count down in the system, and also displays the real-time countdown on the seven-segment digital tube at a millisecond level refresh rate.The beginning signal is also passed to the Beeper.v module. This module outputs a signal to make the buzzer sound. The effect is a traditional Chinese song "Liang Zhu".

1. A player pressed his answer button. The four pick-up signals correspond to four dip switches. These four signals are combined and transmitted to the CountDown.v module to indicate the end of the countdown. At the same time, the countdown is locked in this module, and all other players' push-buttons are disabled. An LED lights up to indicate a player's response. This signal indicating that a question has been rushed into the Answer\_Parts.v module, this module is used to calculate the player's total score and plus and minus points. The getter signal is also passed to the ProceedMark.v module. This module reads the player score signal passed in Answer\_Parst.v and whether the current question is answered. It is used to display the signal on the seven-segment digital tube after the end of a response.
2. At the same time, the getter signal is also transmitted to the keyboard drive module KeyBoard.v as an enable signal of the keyboard. At this time, the keyboard needs to be selected by a number from the Answer\_Parts.v module to indicate the scores added and subtracted in this round of competition. . Two judgment signals (corresponding to two DIP switches) yes and no are passed to Answer\_Parts.v and then passed to the score addition and subtraction module (ProceedMark.v) to modify the stored player score.When someone answers, the getter signal takes effect, and the getter signal is also passed into the Beeper.v signal to terminate the currently playing music.
3. After the host judges the right or wrong and the score is over, the beginning signal is reset and then set again. A new round of rush answering is started, and then the above operation is repeated.
4. When the judgment module in Answer\_Part.v used to record the scores of the current players in the current round has detected that the score is greater than or equal to ten, the game ends. A begg signal indicating whether the game is over is asserted and output to the final\_output.v module. The Final\_output.v module starts.
5. final\_output.v module obtains the scores of 4 players from the Answer\_Part.v module and the winner of the game. After that, the seven-segment digital tube cyclically outputs the scores of the 4 players and finally stop on the winner's score.

9. All the above modules can be reset by the reset signal of the top-level (MainFile.v) module.

1. **Submodule code**

Our main functional modules are as follows. Very detailed comments have been written in the code file to indicate the interface and functions of the module. Please open our design file to view and encode it as UTF-8.

Top-level module: MainFile.v

Keyboard input module: KeyBoard.v

Answer input module: Answer\_Parts.v

Score addition and subtraction judgment module: ProceedMark.v

Buzzer music module: LiangZhu.v

Buzzer decoding module: Beeper.v

Divider module: FrequencyDivider.v

Score display module: Display\_Mark.v

Output display module after the game is over: Final\_Output.v

Countdown display module: CountDown.v

Hello interface at the beginning of the game: showHello.v

1. **Constraint file**
2. set\_property IOSTANDARD LVCMOS33 [get\_ports {answer\_led[3]}]
3. set\_property IOSTANDARD LVCMOS33 [get\_ports {answer\_led[2]}]
4. set\_property IOSTANDARD LVCMOS33 [get\_ports {answer\_led[1]}]
5. set\_property IOSTANDARD LVCMOS33 [get\_ports {answer\_led[0]}]
6. set\_property IOSTANDARD LVCMOS33 [get\_ports {seg\_en[7]}]
7. set\_property IOSTANDARD LVCMOS33 [get\_ports {seg\_en[6]}]
8. set\_property IOSTANDARD LVCMOS33 [get\_ports {seg\_en[5]}]
9. set\_property IOSTANDARD LVCMOS33 [get\_ports {seg\_en[4]}]
10. set\_property IOSTANDARD LVCMOS33 [get\_ports {seg\_en[3]}]
11. set\_property IOSTANDARD LVCMOS33 [get\_ports {seg\_en[2]}]
12. set\_property IOSTANDARD LVCMOS33 [get\_ports {seg\_en[1]}]
13. set\_property IOSTANDARD LVCMOS33 [get\_ports {seg\_en[0]}]
14. set\_property IOSTANDARD LVCMOS33 [get\_ports {seg\_out[7]}]
15. set\_property IOSTANDARD LVCMOS33 [get\_ports {seg\_out[6]}]
16. set\_property IOSTANDARD LVCMOS33 [get\_ports {seg\_out[5]}]
17. set\_property IOSTANDARD LVCMOS33 [get\_ports {seg\_out[4]}]
18. set\_property IOSTANDARD LVCMOS33 [get\_ports {seg\_out[3]}]
19. set\_property IOSTANDARD LVCMOS33 [get\_ports {seg\_out[2]}]
20. set\_property IOSTANDARD LVCMOS33 [get\_ports {seg\_out[1]}]
21. set\_property IOSTANDARD LVCMOS33 [get\_ports {seg\_out[0]}]
22. set\_property PACKAGE\_PIN A21 [get\_ports {answer\_led[3]}]
23. set\_property PACKAGE\_PIN E22 [get\_ports {answer\_led[2]}]
24. set\_property PACKAGE\_PIN D22 [get\_ports {answer\_led[1]}]
25. set\_property PACKAGE\_PIN E21 [get\_ports {answer\_led[0]}]
26. set\_property IOSTANDARD LVCMOS33 [get\_ports {answer[0]}]
27. set\_property IOSTANDARD LVCMOS33 [get\_ports {answer[1]}]
28. set\_property IOSTANDARD LVCMOS33 [get\_ports {answer[2]}]
29. set\_property IOSTANDARD LVCMOS33 [get\_ports {answer[3]}]
30. set\_property IOSTANDARD LVCMOS33 [get\_ports count\_down]
31. set\_property IOSTANDARD LVCMOS33 [get\_ports No]
32. set\_property IOSTANDARD LVCMOS33 [get\_ports Yes]
33. set\_property PACKAGE\_PIN T5 [get\_ports {answer[0]}]
34. set\_property PACKAGE\_PIN T4 [get\_ports {answer[1]}]
35. set\_property PACKAGE\_PIN R4 [get\_ports {answer[2]}]
36. set\_property PACKAGE\_PIN W4 [get\_ports {answer[3]}]
37. set\_property PACKAGE\_PIN A18 [get\_ports {seg\_en[7]}]
38. set\_property PACKAGE\_PIN A20 [get\_ports {seg\_en[6]}]
39. set\_property PACKAGE\_PIN B20 [get\_ports {seg\_en[5]}]
40. set\_property PACKAGE\_PIN E18 [get\_ports {seg\_en[4]}]
41. set\_property PACKAGE\_PIN F18 [get\_ports {seg\_en[3]}]
42. set\_property PACKAGE\_PIN D19 [get\_ports {seg\_en[2]}]
43. set\_property PACKAGE\_PIN E19 [get\_ports {seg\_en[1]}]
44. set\_property PACKAGE\_PIN C19 [get\_ports {seg\_en[0]}]
45. set\_property PACKAGE\_PIN E13 [get\_ports {seg\_out[7]}]
46. set\_property PACKAGE\_PIN C15 [get\_ports {seg\_out[6]}]
47. set\_property PACKAGE\_PIN C14 [get\_ports {seg\_out[5]}]
48. set\_property PACKAGE\_PIN E17 [get\_ports {seg\_out[4]}]
49. set\_property PACKAGE\_PIN F16 [get\_ports {seg\_out[3]}]
50. set\_property PACKAGE\_PIN F14 [get\_ports {seg\_out[2]}]
51. set\_property PACKAGE\_PIN F13 [get\_ports {seg\_out[1]}]
52. set\_property PACKAGE\_PIN F15 [get\_ports {seg\_out[0]}]
53. set\_property IOSTANDARD LVCMOS33 [get\_ports begining]
54. set\_property PACKAGE\_PIN Y9 [get\_ports begining]
55. set\_property PACKAGE\_PIN W9 [get\_ports count\_down]
56. set\_property PACKAGE\_PIN Y8 [get\_ports No]
57. set\_property PACKAGE\_PIN Y7 [get\_ports Yes]
58. set\_property CLOCK\_DEDICATED\_ROUTE FALSE [get\_nets Yes\_IBUF]
59. set\_property CLOCK\_DEDICATED\_ROUTE FALSE [get\_nets No\_IBUF]
60. set\_property IOSTANDARD LVCMOS33 [get\_ports time\_select]
61. set\_property PACKAGE\_PIN AB8 [get\_ports time\_select]
62. set\_property IOSTANDARD LVCMOS33 [get\_ports select1]
63. set\_property PACKAGE\_PIN W6 [get\_ports select1]
64. set\_property IOSTANDARD LVCMOS33 [get\_ports select2]
65. set\_property PACKAGE\_PIN U5 [get\_ports select2]

Explanation of the constraint file:

1. The answer\_led signal corresponds to 4 led output signals, which are used to indicate the current successful player.

b .seg\_en signal is a seven-segment digital tube enable signal

c .seg\_out signal is a seven-segment digital tube output signal

d .The answer signal is the input signal of the answering player, and the corresponding 4 dial switches indicate the answering action of the answering player.

e .count\_down is a reset signal for a round of pickup, indicating that the host announces the start of a new round of pickup

f .No and Yes signals are signals that the host judges whether they are right or wrong, corresponding to two DIP switches

g. Beginning is the reset signal for the entire game, indicating whether the game is started, corresponding to a DIP switch

h .time\_select signal indicates the countdown time selection input signal, corresponding to a DIP switch

i .alarm is the output signal of the buzzer

j .row is the input signal of the keyboard

k. col is the output signal of the keyboard

l .clk is the clock signal of the system

m . select1 and select2 are used to select the number of players

**Testing Report**

**Testbench file**

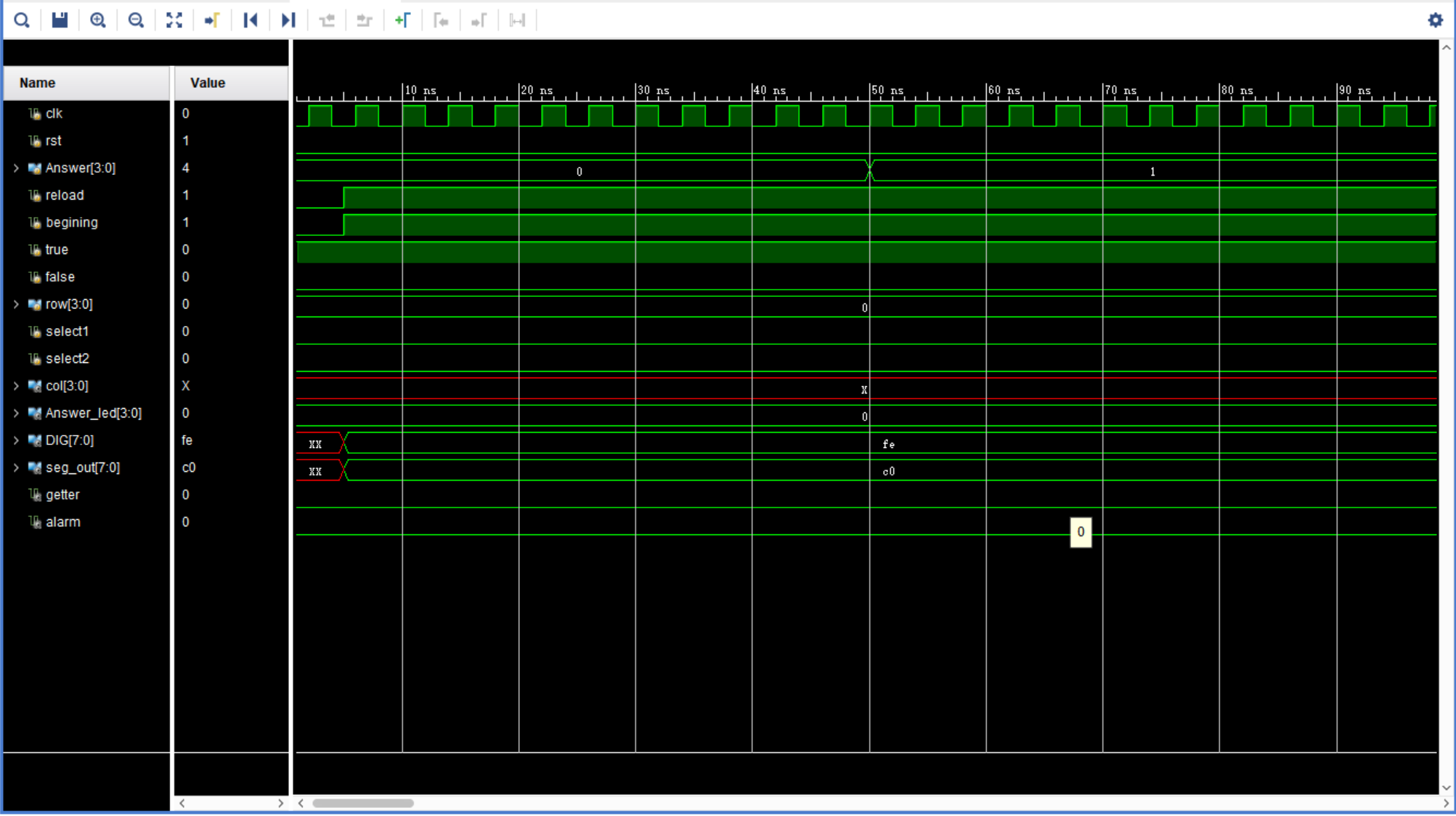
The testbench file is attached to the file package submitted, please ask the teacher to open the sim folder to view it.

**Waveform test results of the submodule**

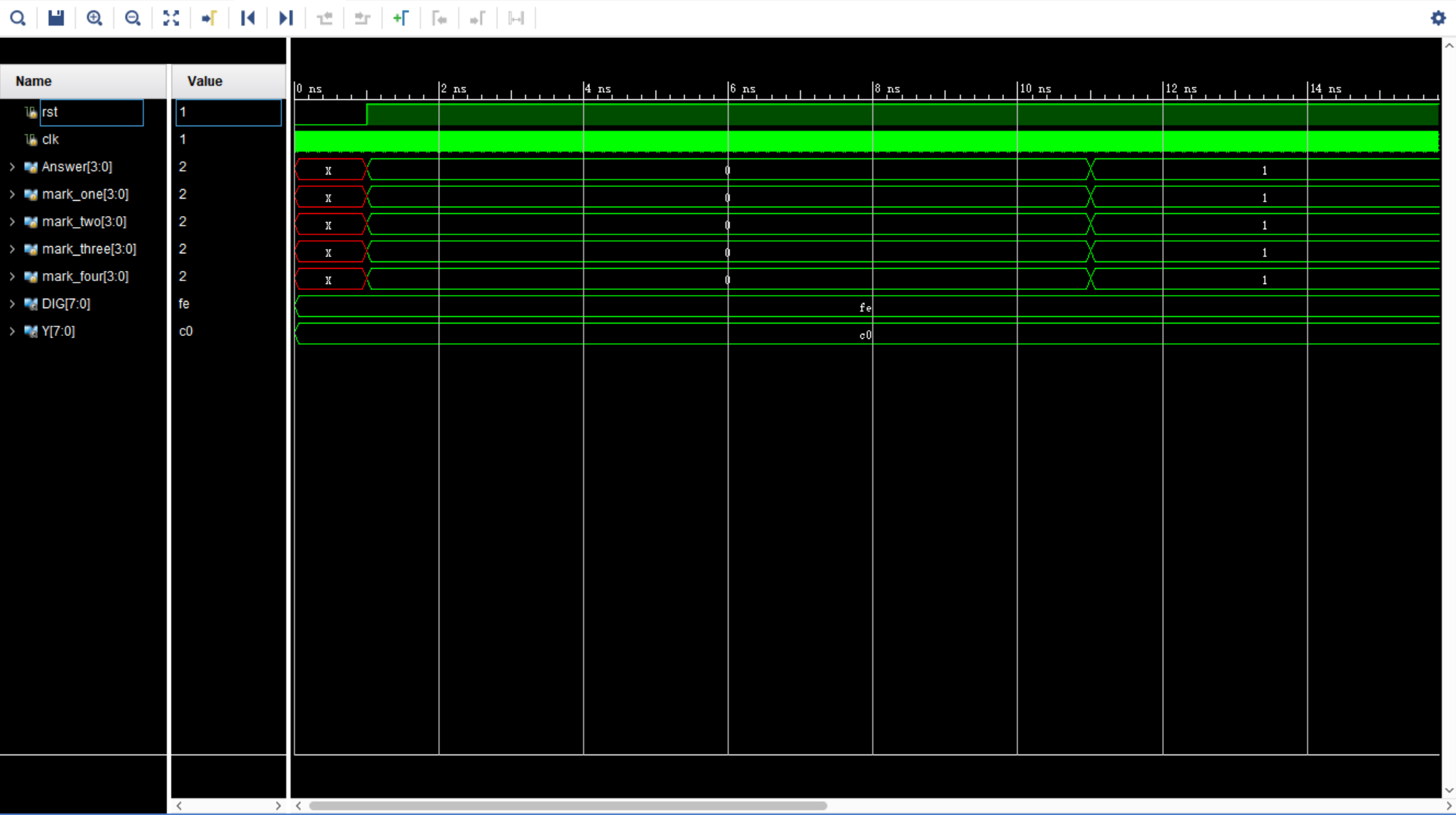
Testing result for KeyBoard.v



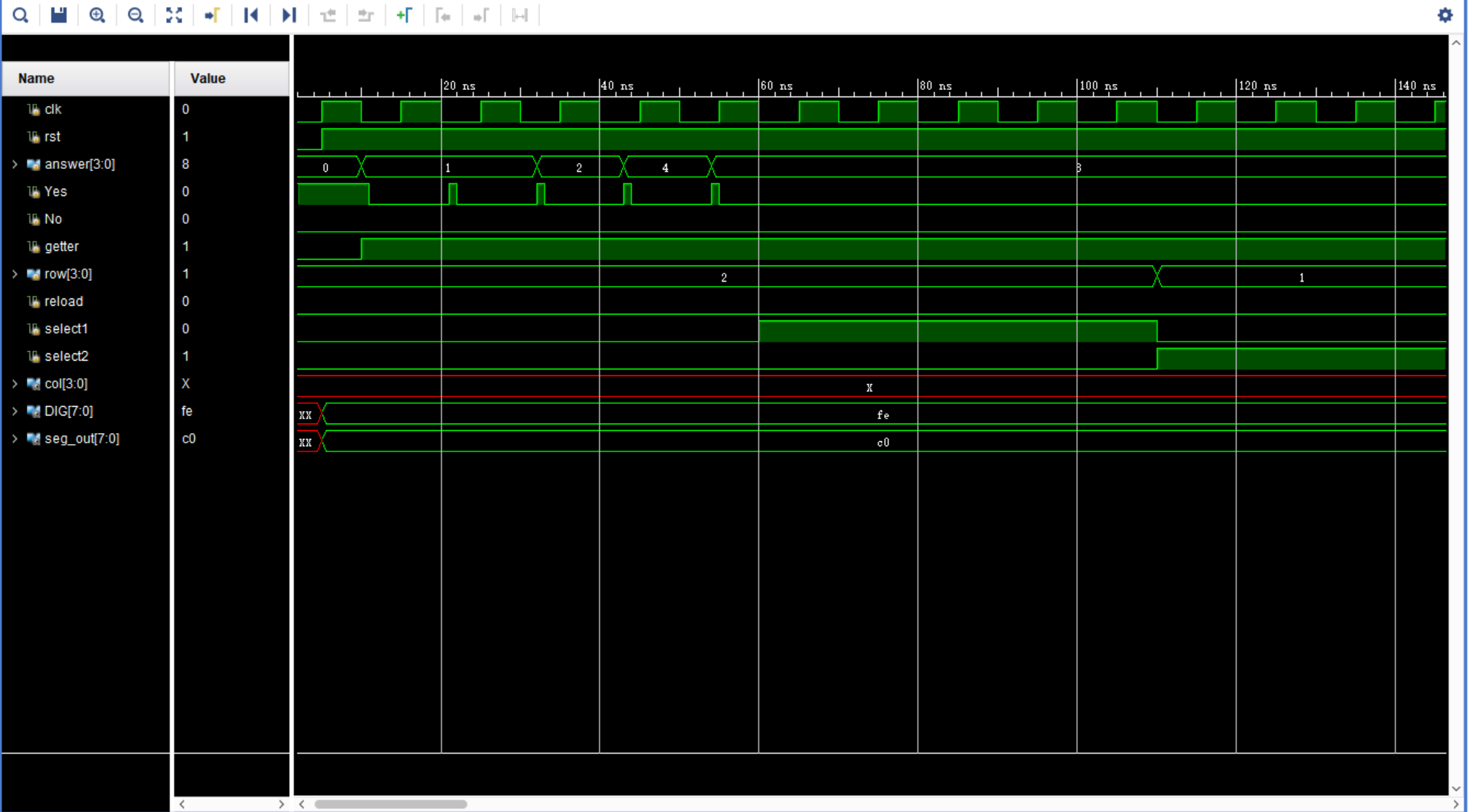
Test result for Answer\_Parts



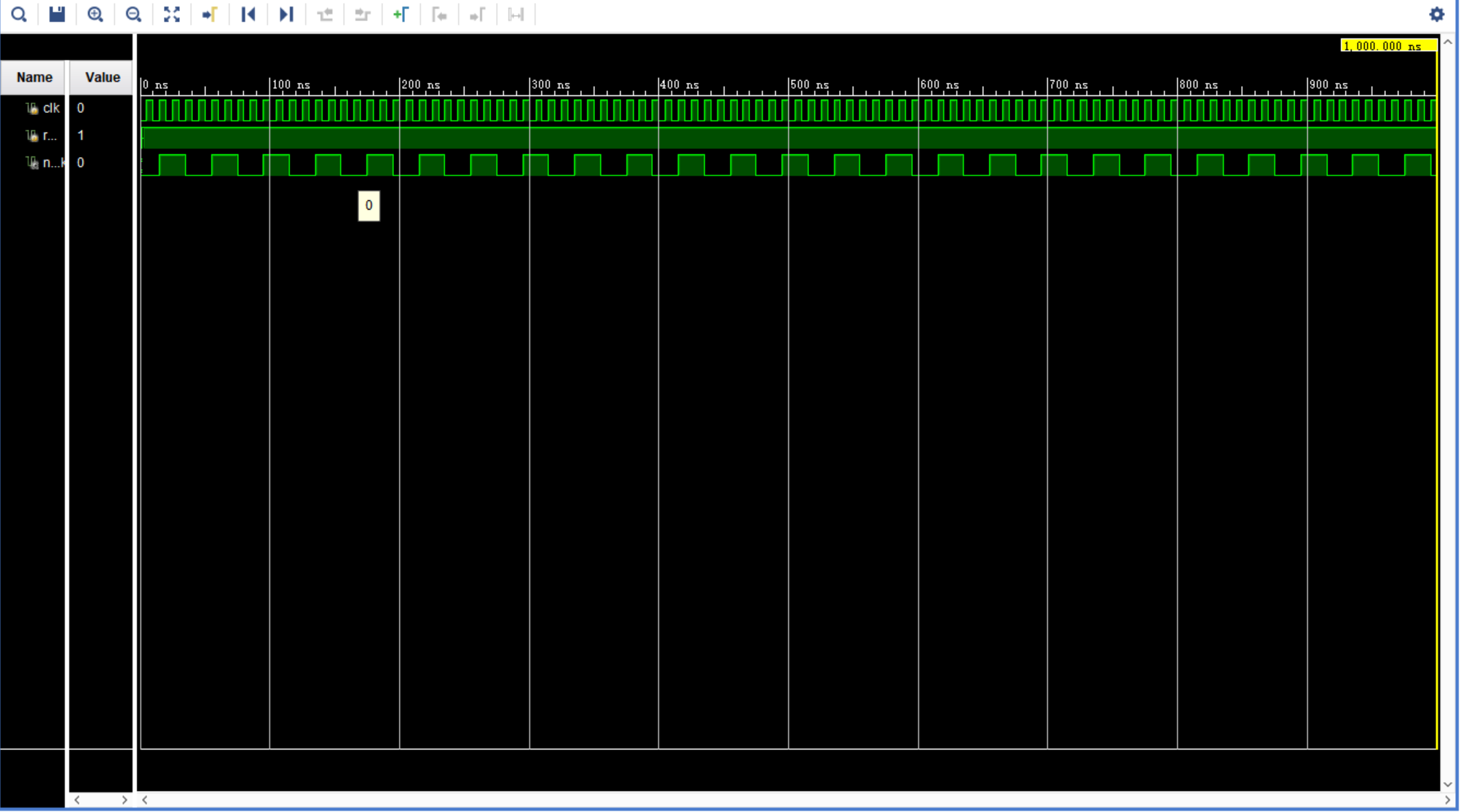
Test result for Display\_Mark



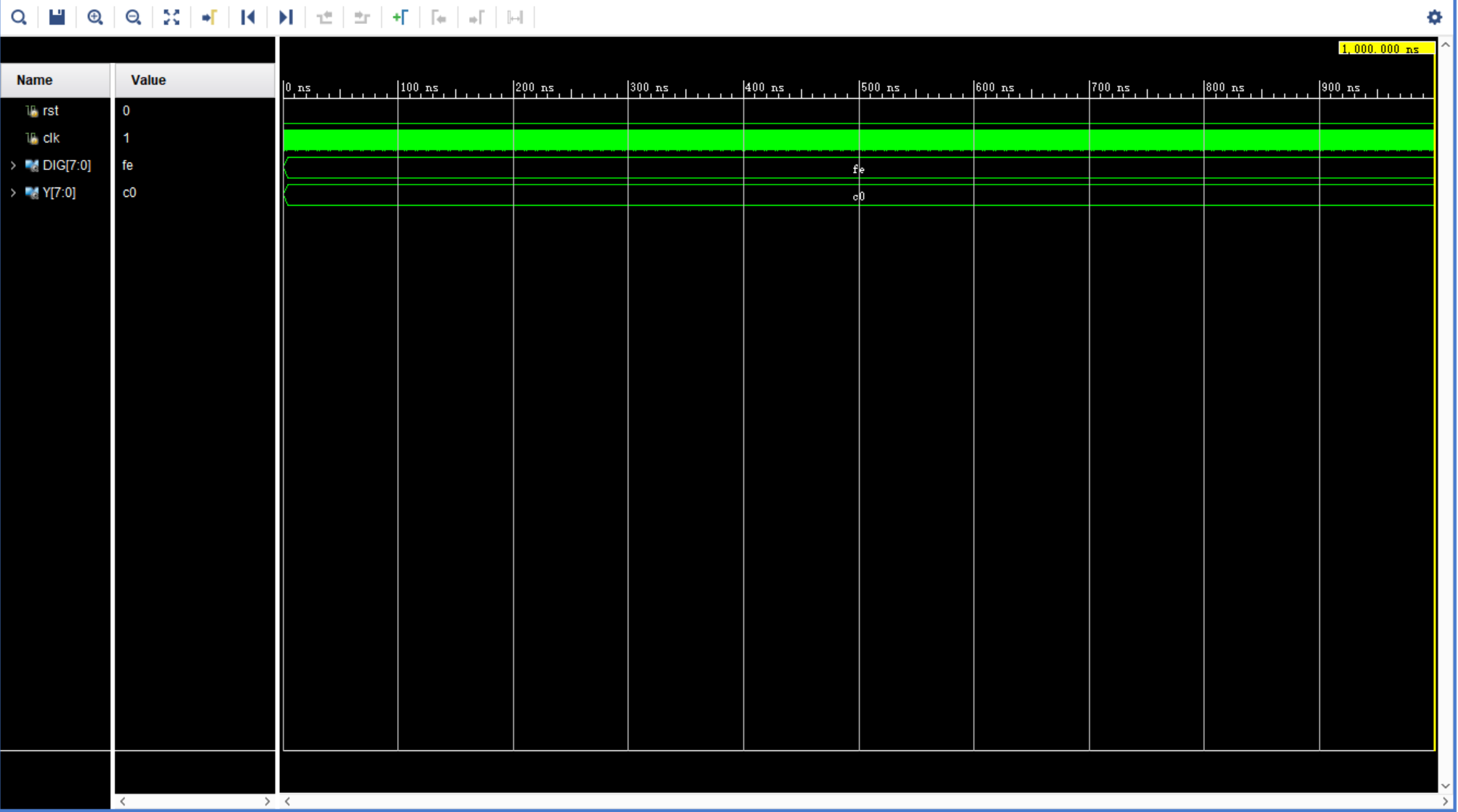
Test result for ProceedMark



Test result for FrequencyDivider

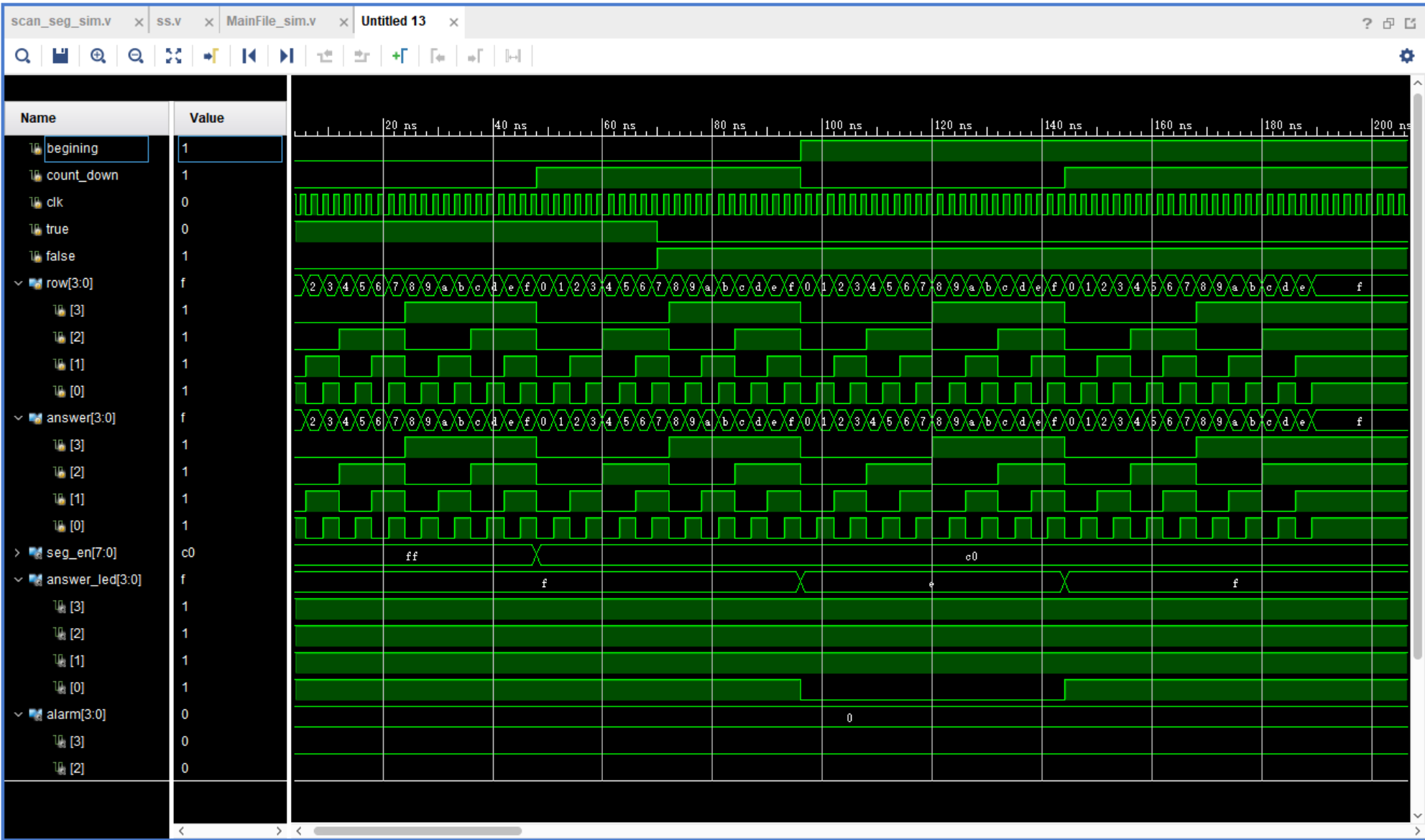


Test result for ShowHello

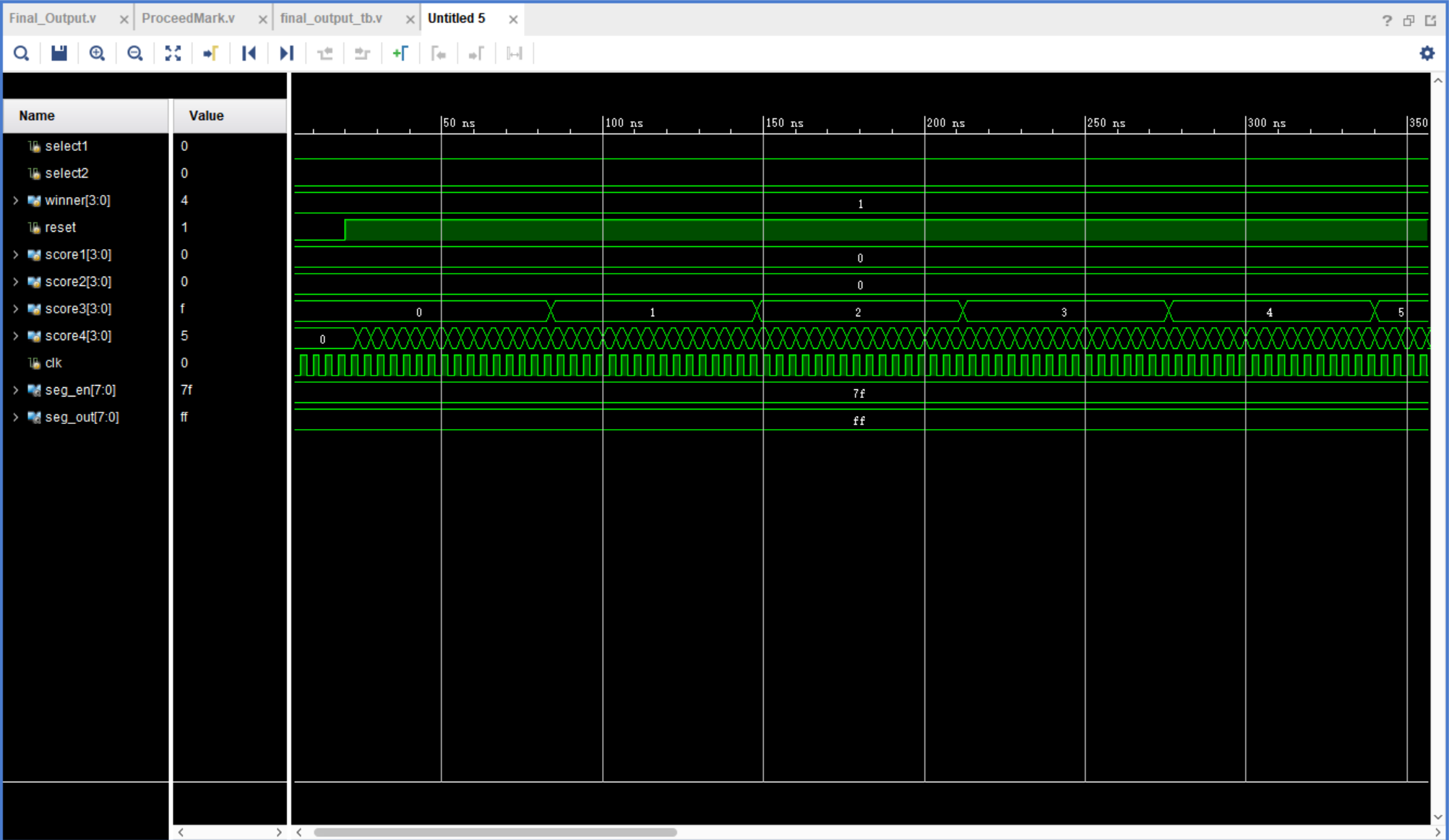


The test result of the top-level module (MainFile.v) is the test result of the final overall project. In order to facilitate testing, the buzzer module and keyboard module have been temporarily removed

Test result for MainFile



Test result for Final\_Output



**Summary**

a) Summary of problems and solutions encountered during development

There are undoubtedly many difficulties encountered in the development process, which can be summarized as follows:

1. When the function modules were docked after each, it was found that the module responsible by the teammates did not meet expectations.

**Solution:** At the beginning of the discussion, the content and interface of the functionmodule is decided, and the developer strictly follows the given interface to program.

1. For modules with similar functions, different people have different ideas

**Solution:** Give the dispute-prone module to one person to complete, and ensure that

other team members do not intervene, and each person is only responsible for his

own module, so that the development efficiency can be greatly improved.

b) Describe the characteristics and optimization direction of the current system.

**Characteristics:** We can use the keyboard freely while giving scores,and we use the

Beeper to play beautiful music while countingdown. Also,we have a warming welcome

for players who start the game.We can choose the countdown time and set the number of players,which are very useful.

**Optimization:**We can increase the richness of the music produced by the buzzer. In

order to achieve this purpose, we can plan to modify the design file of the buzzer to

change its frequency and sounding time.

We also have plans to make the page that finally displays scores more beautiful, such

as adding a display process that congratulates the player on winning.

***Finally, we have attached a video to demonstrate our project.***