CDA 4203L
Computer System Design Lab

Lab 4 Report
Vending Machine FSM

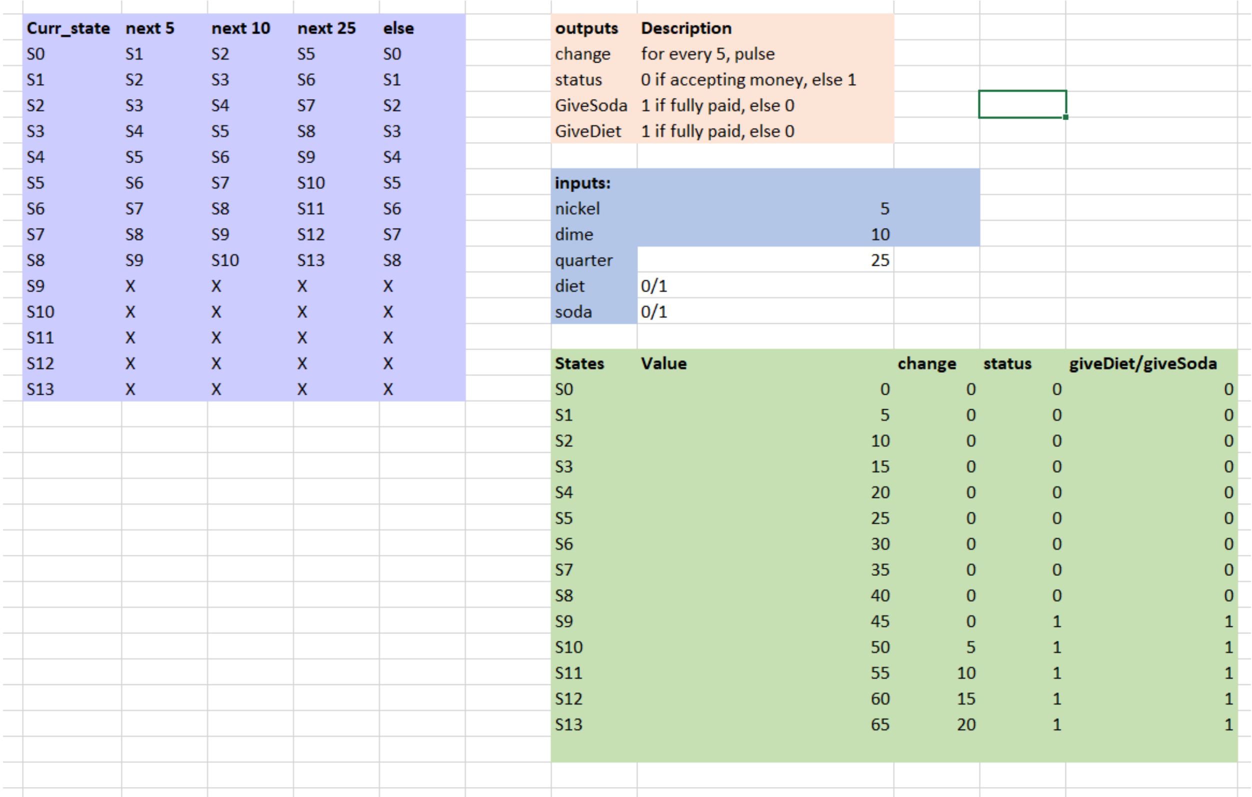
Today's Date:	2/20/2022
Team Members:	Ahmed El Maliki - U18019330
	Stella Kariuki – U19511653
	Ivan Gonzalez -
Work Distribution:	Briefly explain the tasks completed by each team member Ahmed – completed code and ensured proper simulation Stella – Board testing and video presentation Ivan – Report
No. of Hours Spent:	Way too many
Exercise Difficulty: (Easy, Average, Hard)	Hard – Debugger and clock divider came without instructions. Many warnings initially generated stating creation of latches during synthesis, combinatorial implementation of clock (gate clock). Debugging took a huge chunk of time. Link to video presentation: https://youtu.be/NenAHn2YNMs

Any Other Feedback:	Some more information on how to use the provided files should be included, and why they should be used. Websites or links for guidance should be enough.

Problem 1: Show the state diagram of your vending machine controller. Briefly explain how the design works. *Use as many pages as needed.*

Using the Moore model, States were decided upon using the following logic:

- Paying using nickels only would give us 10 total states, S0 through S9, where each state is as a result of an increment of 5c. So S0 would be 0c (base case, no money inserted), S1 would be 5c (1 nickel inserted), … S9 would be 45c (9 nickels inserted, no more needed)
- The max amount of money you can insert in the machine is 65c, given the case where you have already inserted 40c and only have a quarter in your pocket. Analyzing the intermediary states arrived at show that there are states between S9 (45c) and S13 (65c), and that we have a total of 14 states. The **state table** is shown in the snippet of the excel file below in purple. Note that the next state is as a result of the input in the current state, so each state will point to 4 different next states, due to the three different inputs (nickel, dime, quarter), and 1 next state is the default (0 input) which keeps the current state until a coin is inserted.
- The **Outputs** are zero for all states that are below 45c and are all shown below in **green,** described in **orange.** The **Inputs** are also specified in **blue** below. Note that the clock and reset inputs are not included here. They will be included in the simulation and code.



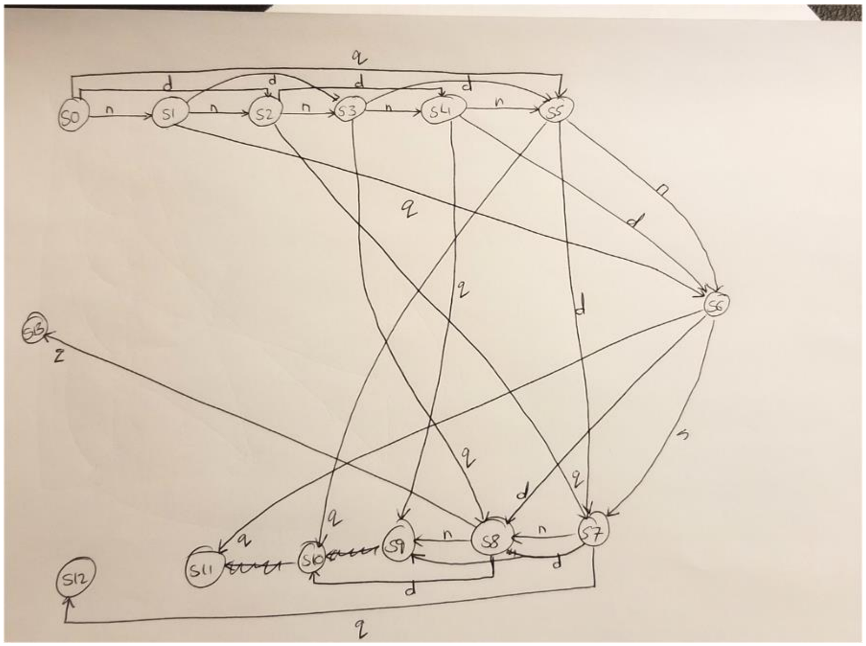
Curr_state	next 5	next 10	next 25	else
S0	S1	S2	S5	S0
S1	S2	S3	S6	S1
S2	S3	S4	S7	S2
S3	S4	S5	S8	S3
S4	S5	S6	S9	S4
S5	S6	S7	S10	S5
S6	S7	S8	S11	S6
S7	S8	S9	S12	S7
S8	S9	S10	S13	S8
S9	X	X	X	X
S10	X	X	X	X
S11	X	X	X	X
S12	X	X	X	X
S13	X	X	X	X

outputs	Description
change	for every 5, pulse
status	0 if accepting money, else 1
GiveSoda	1 if fully paid, else 0
GiveDiet	1 if fully paid, else 0

inputs:		
nickel		5
dime		10
quarter		25
diet	0/1	
soda	0/1	

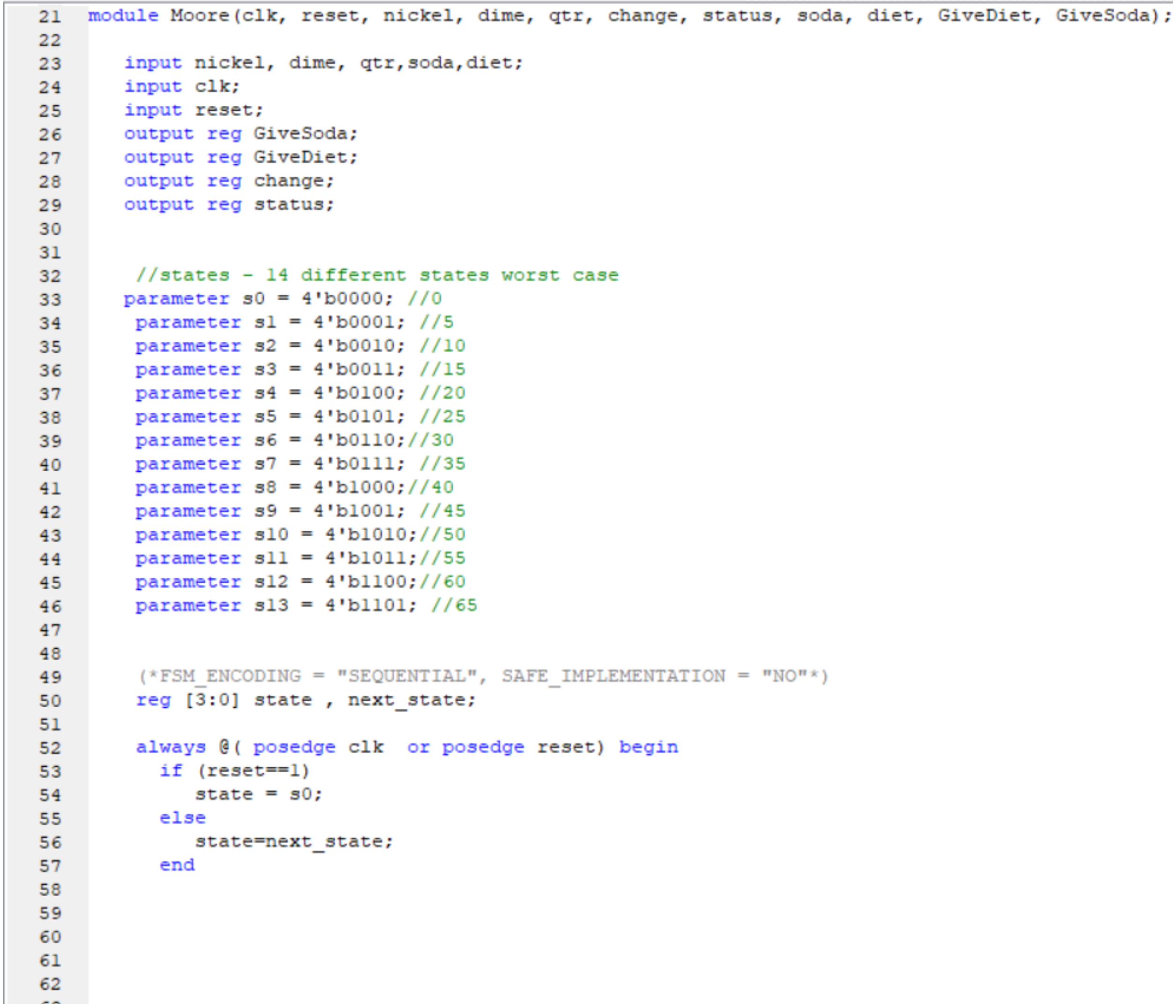
States	Value	change	status	giveDiet/giveSoda
S0	0	0	0	0
S1	5	0	0	0
S2	10	0	0	0
S3	15	0	0	0
S4	20	0	0	0
S5	25	0	0	0
S6	30	0	0	0
S7	35	0	0	0
S8	40	0	0	0
S9	45	0	1	1
S10	50	5	1	1
S11	55	10	1	1
S12	60	15	1	1
S13	65	20	1	1

State diagram

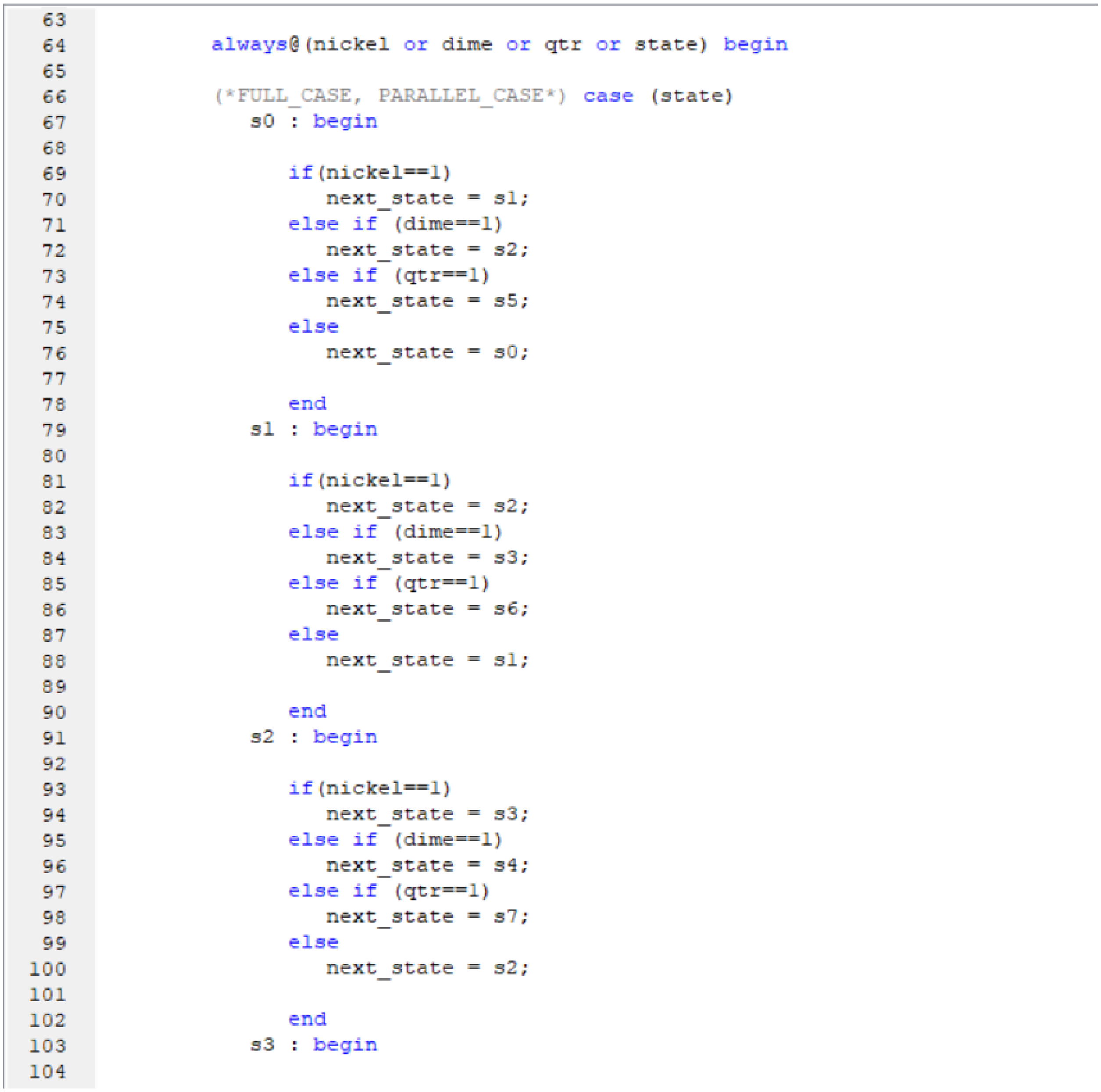

q
d
d
d
d
S0
n
S1
n
S2
n
S3
n
S4
n
S5
n
q
d
q
d
S6
S13
z
n
q
d
q
q
q
S11
S10
S9
n
S8
n
S7
d
d
S12
q

Problem 1: Include the Verilog code of (a) FSM; (b) Testbench; and (c) Simulation waveforms. *Use as many pages as needed.*

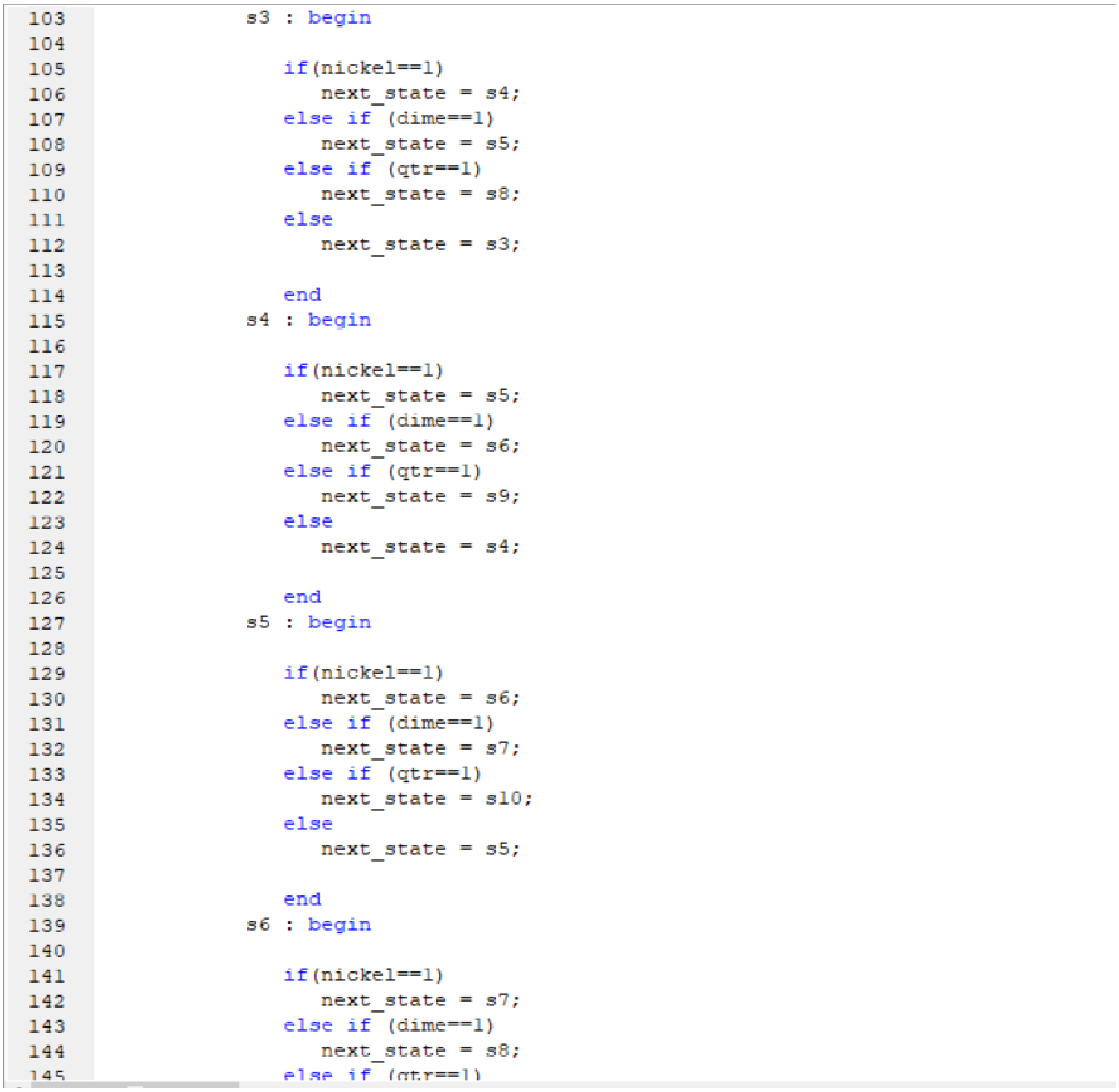
a. FSM Verilog Code (Moore.v)



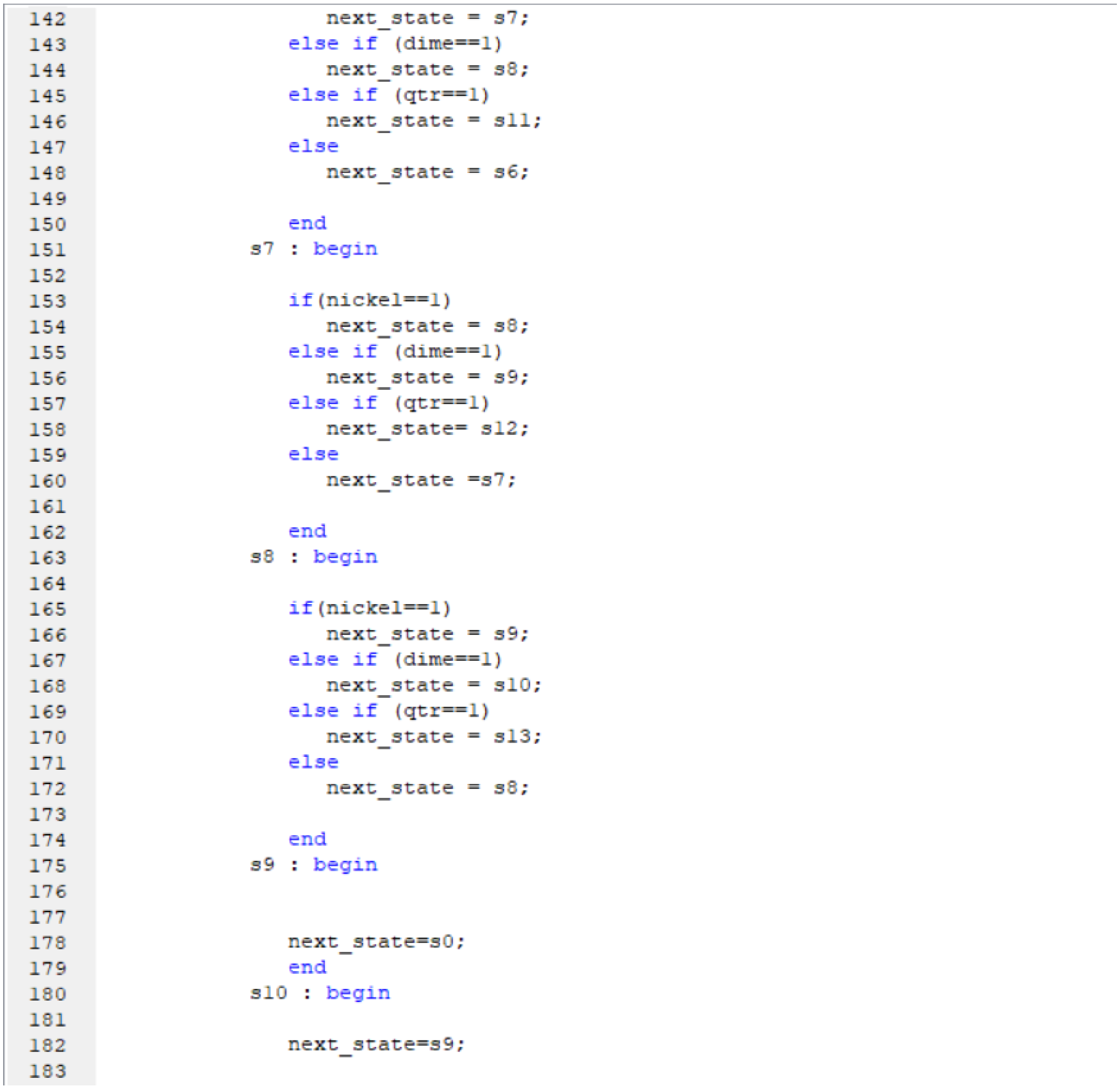
```
module Moore(clk, reset, nickel, dime, qtr, change, status, soda, diet, GiveDiet, GiveSoda);

   input nickel, dime, qtr,soda,diet;
   input clk;
   input reset;
   output reg GiveSoda;
   output reg GiveDiet;
   output reg change;
   output reg status;


    //states - 14 different states worst case
   parameter s0 = 4'b0000; //0
    parameter s1 = 4'b0001; //5
    parameter s2 = 4'b0010; //10
    parameter s3 = 4'b0011; //15
    parameter s4 = 4'b0100; //20
    parameter s5 = 4'b0101; //25
    parameter s6 = 4'b0110;//30
    parameter s7 = 4'b0111; //35
    parameter s8 = 4'b1000;//40
    parameter s9 = 4'b1001; //45
    parameter s10 = 4'b1010;//50
    parameter s11 = 4'b1011;//55
    parameter s12 = 4'b1100;//60
    parameter s13 = 4'b1101; //65


    (*FSM_ENCODING = "SEQUENTIAL", SAFE_IMPLEMENTATION = "NO"*)
    reg [3:0] state , next_state;

    always @( posedge clk  or posedge reset) begin
      if (reset==1)
         state = s0;
      else
         state=next_state;
      end
```



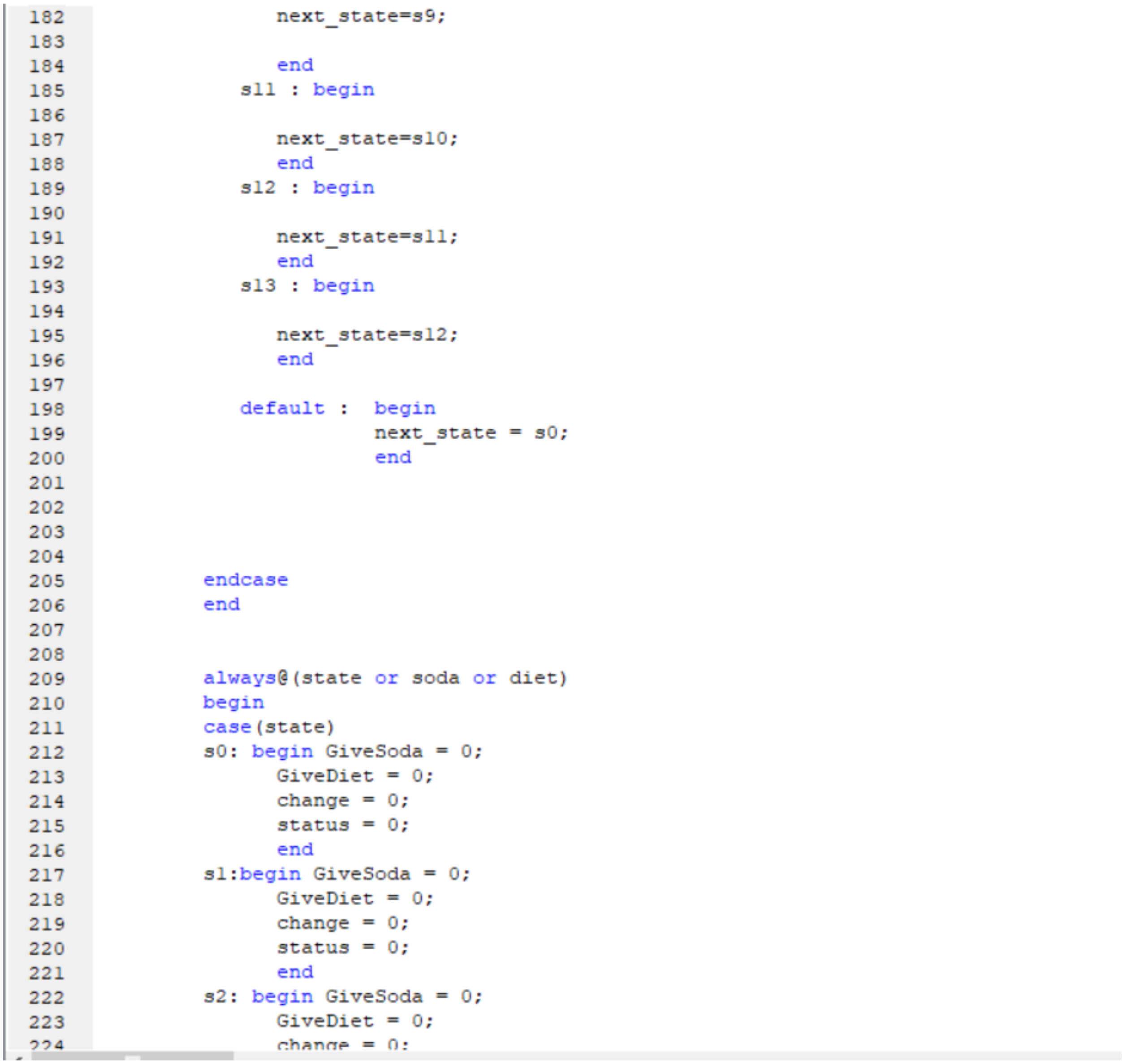
```

        always@(nickel or dime or qtr or state) begin

        (*FULL_CASE, PARALLEL_CASE*) case (state)
           s0 : begin

              if(nickel==1)
                 next_state = s1;
              else if (dime==1)
                 next_state = s2;
              else if (qtr==1)
                 next_state = s5;
              else
                 next_state = s0;

              end
           s1 : begin

              if(nickel==1)
                 next_state = s2;
              else if (dime==1)
                 next_state = s3;
              else if (qtr==1)
                 next_state = s6;
              else
                 next_state = s1;

              end
           s2 : begin

              if(nickel==1)
                 next_state = s3;
              else if (dime==1)
                 next_state = s4;
              else if (qtr==1)
                 next_state = s7;
              else
                 next_state = s2;

              end
           s3 : begin

```



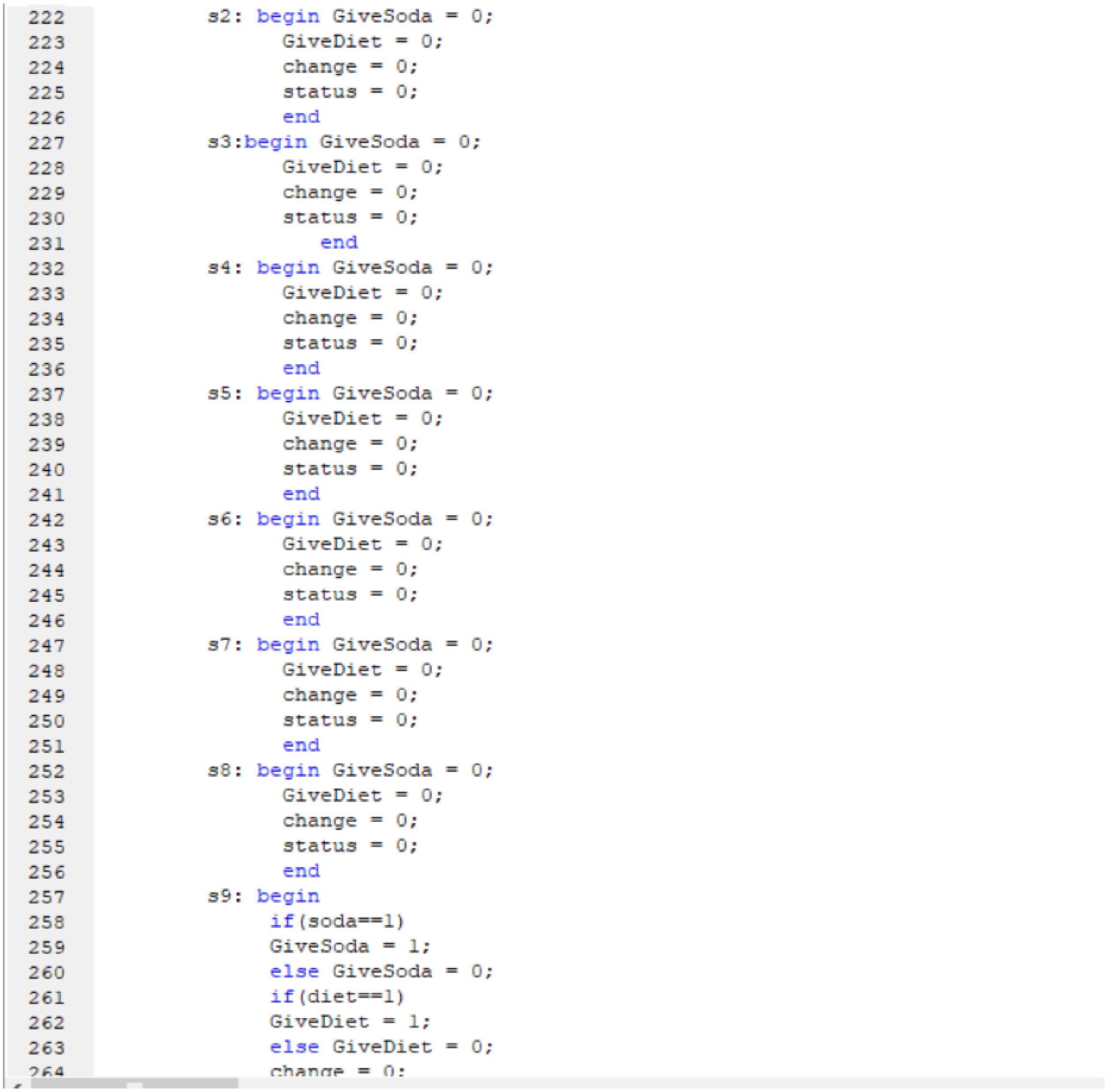
```
            s3 : begin

               if(nickel==1)
                  next_state = s4;
               else if (dime==1)
                  next_state = s5;
               else if (qtr==1)
                  next_state = s8;
               else
                  next_state = s3;

               end
            s4 : begin

               if(nickel==1)
                  next_state = s5;
               else if (dime==1)
                  next_state = s6;
               else if (qtr==1)
                  next_state = s9;
               else
                  next_state = s4;

               end
            s5 : begin

               if(nickel==1)
                  next_state = s6;
               else if (dime==1)
                  next_state = s7;
               else if (qtr==1)
                  next_state = s10;
               else
                  next_state = s5;

               end
            s6 : begin

               if(nickel==1)
                  next_state = s7;
               else if (dime==1)
                  next_state = s8;
               else if (qtr==1)
```



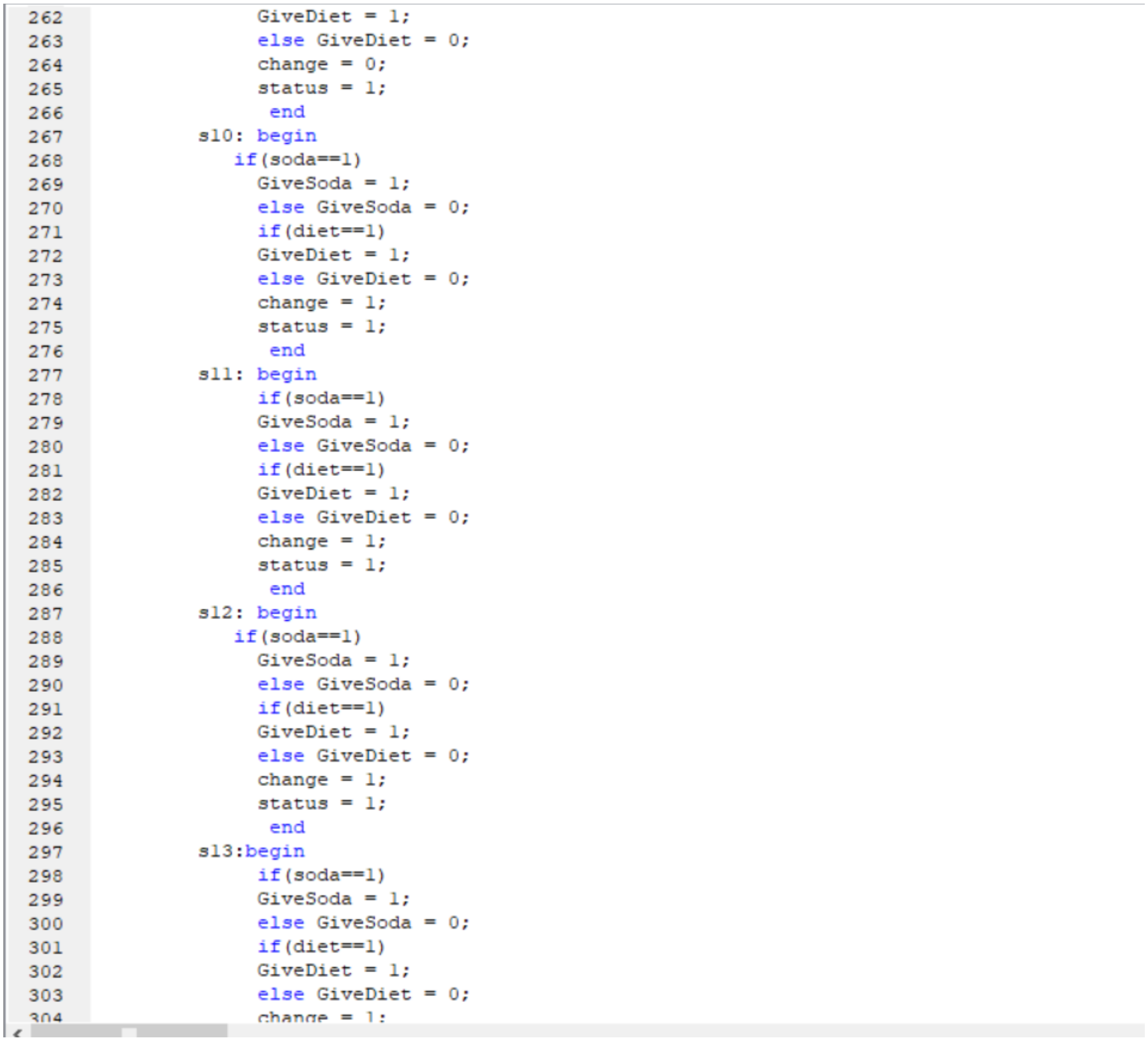
```
            next_state = s7;
         else if (dime==1)
            next_state = s8;
         else if (qtr==1)
            next_state = s11;
         else
            next_state = s6;

         end
      s7 : begin

         if(nickel==1)
            next_state = s8;
         else if (dime==1)
            next_state = s9;
         else if (qtr==1)
            next_state= s12;
         else
            next_state =s7;

         end
      s8 : begin

         if(nickel==1)
            next_state = s9;
         else if (dime==1)
            next_state = s10;
         else if (qtr==1)
            next_state = s13;
         else
            next_state = s8;

         end
      s9 : begin


         next_state=s0;
         end
      s10 : begin

         next_state=s9;

```



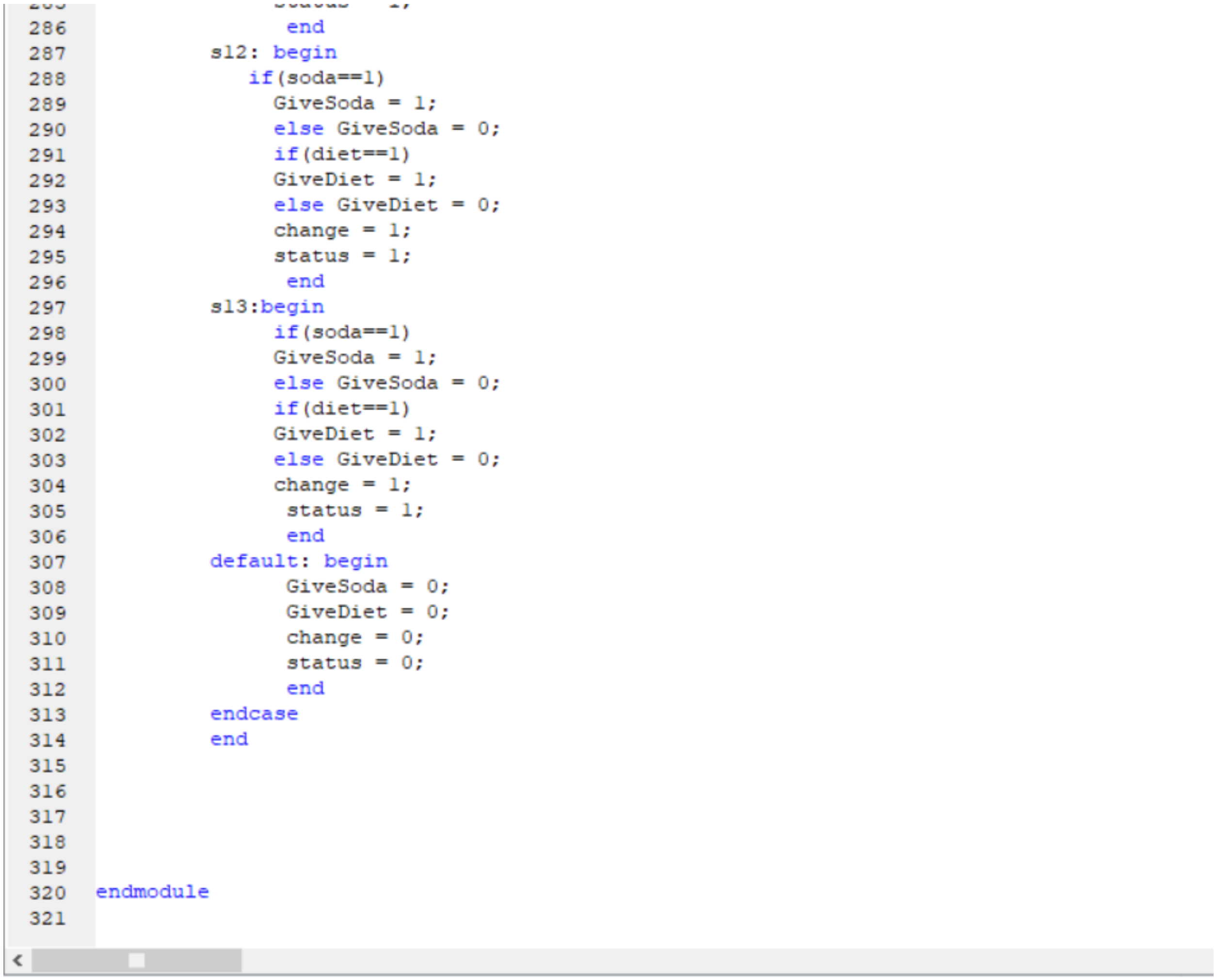
```
              next_state=s9;

              end
           s11 : begin

              next_state=s10;
              end
           s12 : begin

              next_state=s11;
              end
           s13 : begin

              next_state=s12;
              end

           default :  begin
                      next_state = s0;
                      end




        endcase
        end


        always@(state or soda or diet)
        begin
        case(state)
        s0: begin GiveSoda = 0;
              GiveDiet = 0;
              change = 0;
              status = 0;
              end
        s1:begin GiveSoda = 0;
              GiveDiet = 0;
              change = 0;
              status = 0;
              end
        s2: begin GiveSoda = 0;
              GiveDiet = 0;
              change = 0;
```



```
            s2: begin GiveSoda = 0;
                  GiveDiet = 0;
                  change = 0;
                  status = 0;
                  end
            s3:begin GiveSoda = 0;
                  GiveDiet = 0;
                  change = 0;
                  status = 0;
                     end
            s4: begin GiveSoda = 0;
                  GiveDiet = 0;
                  change = 0;
                  status = 0;
                  end
            s5: begin GiveSoda = 0;
                  GiveDiet = 0;
                  change = 0;
                  status = 0;
                  end
            s6: begin GiveSoda = 0;
                  GiveDiet = 0;
                  change = 0;
                  status = 0;
                  end
            s7: begin GiveSoda = 0;
                  GiveDiet = 0;
                  change = 0;
                  status = 0;
                  end
            s8: begin GiveSoda = 0;
                  GiveDiet = 0;
                  change = 0;
                  status = 0;
                  end
            s9: begin
                 if(soda==1)
                 GiveSoda = 1;
                 else GiveSoda = 0;
                 if(diet==1)
                 GiveDiet = 1;
                 else GiveDiet = 0;
                 change = 0;
```

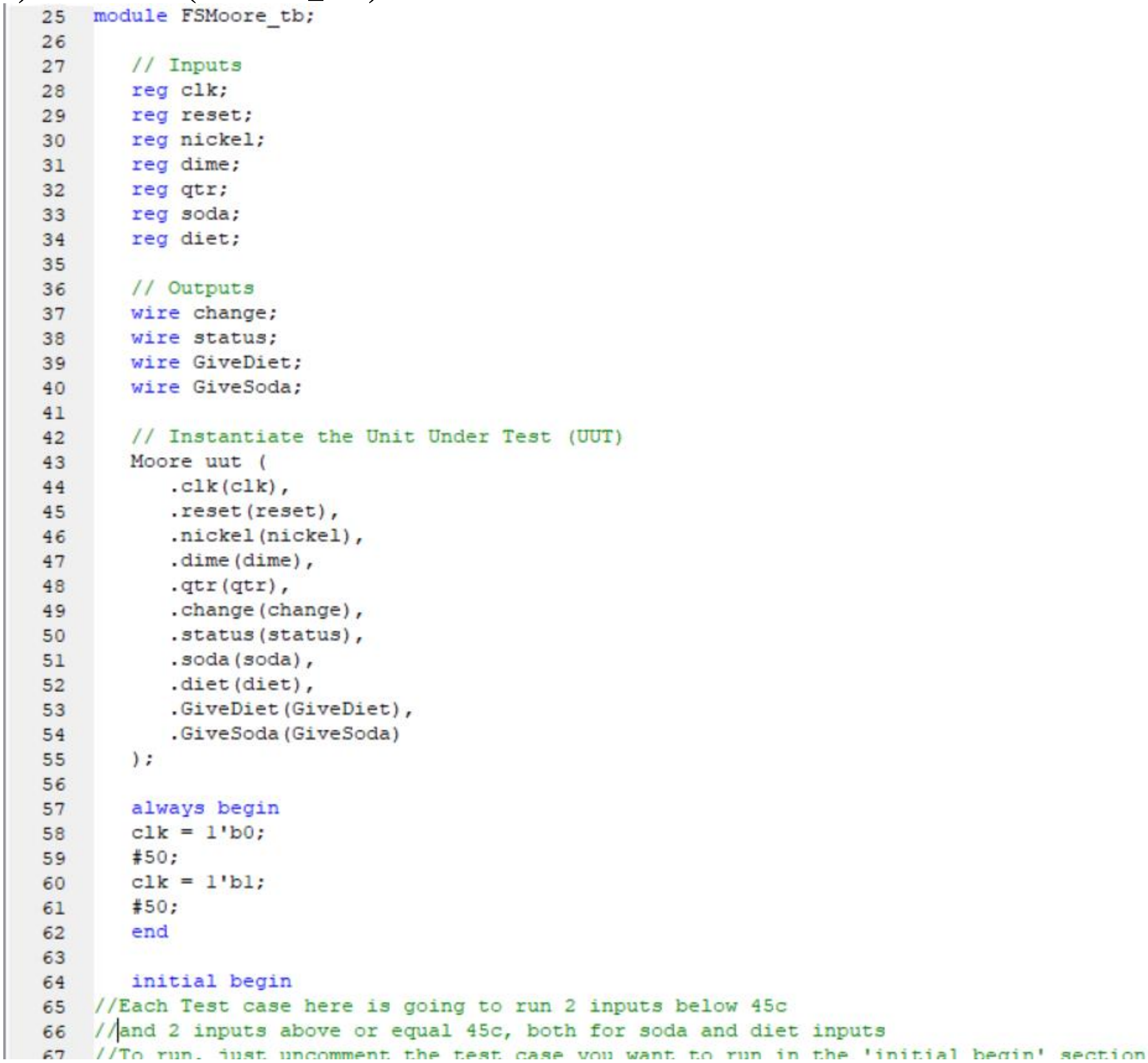


```
             GiveDiet = 1;
             else GiveDiet = 0;
             change = 0;
             status = 1;
              end
        s10: begin
           if(soda==1)
             GiveSoda = 1;
             else GiveSoda = 0;
             if(diet==1)
             GiveDiet = 1;
             else GiveDiet = 0;
             change = 1;
             status = 1;
              end
        s11: begin
             if(soda==1)
             GiveSoda = 1;
             else GiveSoda = 0;
             if(diet==1)
             GiveDiet = 1;
             else GiveDiet = 0;
             change = 1;
             status = 1;
              end
        s12: begin
           if(soda==1)
             GiveSoda = 1;
             else GiveSoda = 0;
             if(diet==1)
             GiveDiet = 1;
             else GiveDiet = 0;
             change = 1;
             status = 1;
              end
        s13:begin
             if(soda==1)
             GiveSoda = 1;
             else GiveSoda = 0;
             if(diet==1)
             GiveDiet = 1;
             else GiveDiet = 0;
             change = 1;
```

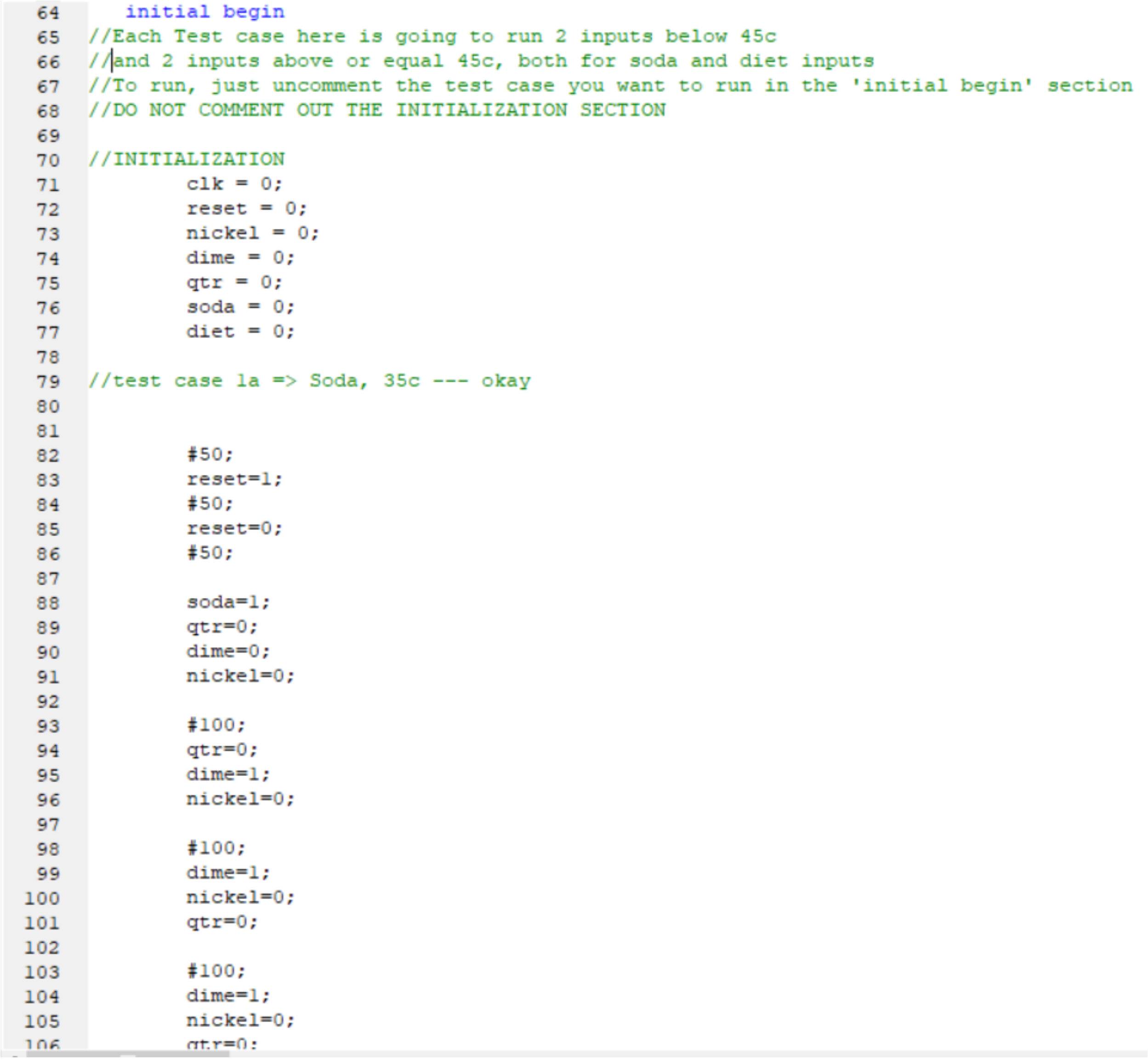


```
              status = 1;
               end
         s12: begin
            if(soda==1)
              GiveSoda = 1;
              else GiveSoda = 0;
              if(diet==1)
              GiveDiet = 1;
              else GiveDiet = 0;
              change = 1;
              status = 1;
               end
         s13:begin
              if(soda==1)
              GiveSoda = 1;
              else GiveSoda = 0;
              if(diet==1)
              GiveDiet = 1;
              else GiveDiet = 0;
              change = 1;
               status = 1;
               end
         default: begin
               GiveSoda = 0;
               GiveDiet = 0;
               change = 0;
               status = 0;
               end
         endcase
         end





endmodule

```

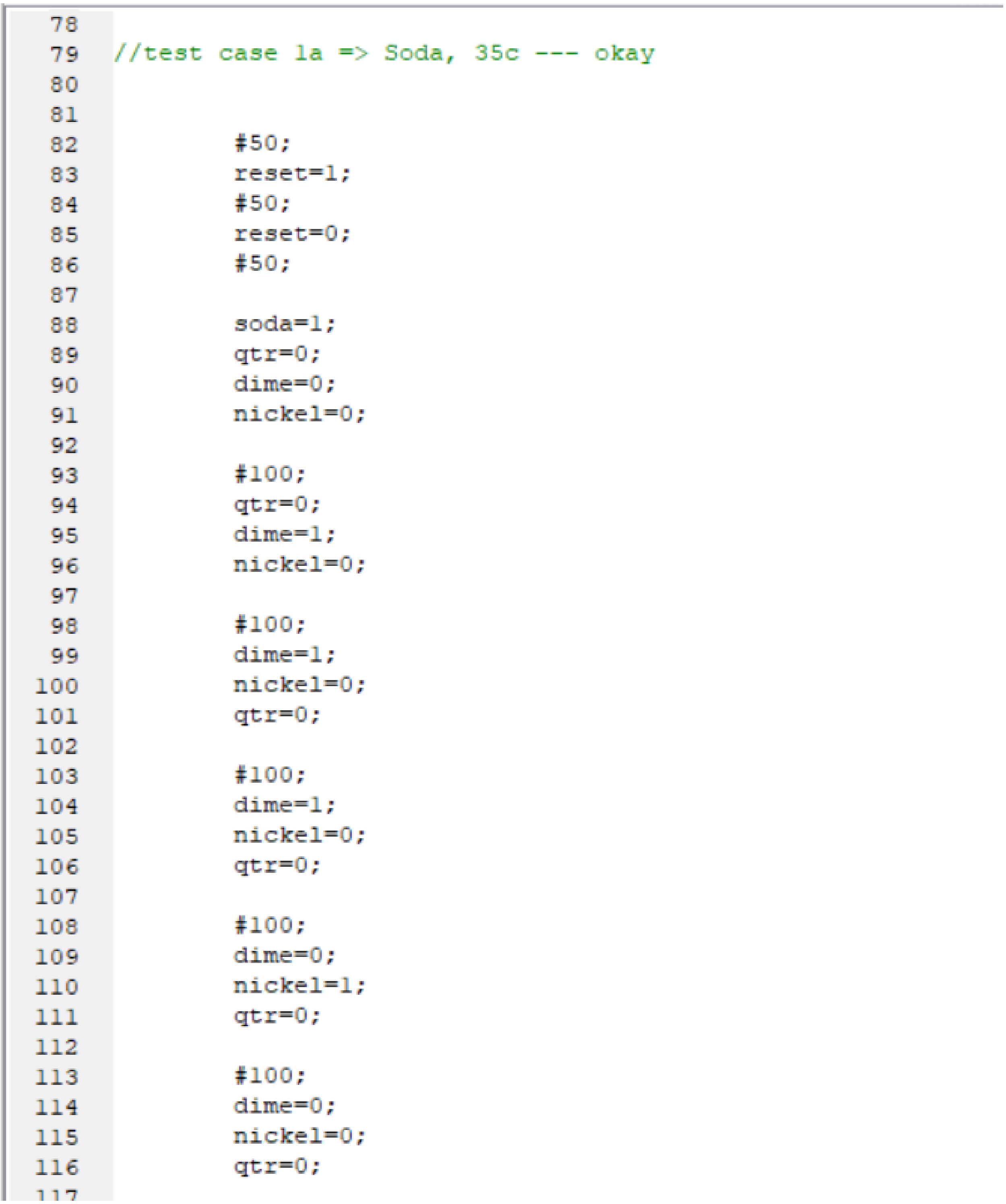
b) Testbench – (FSMoore_tb.v)



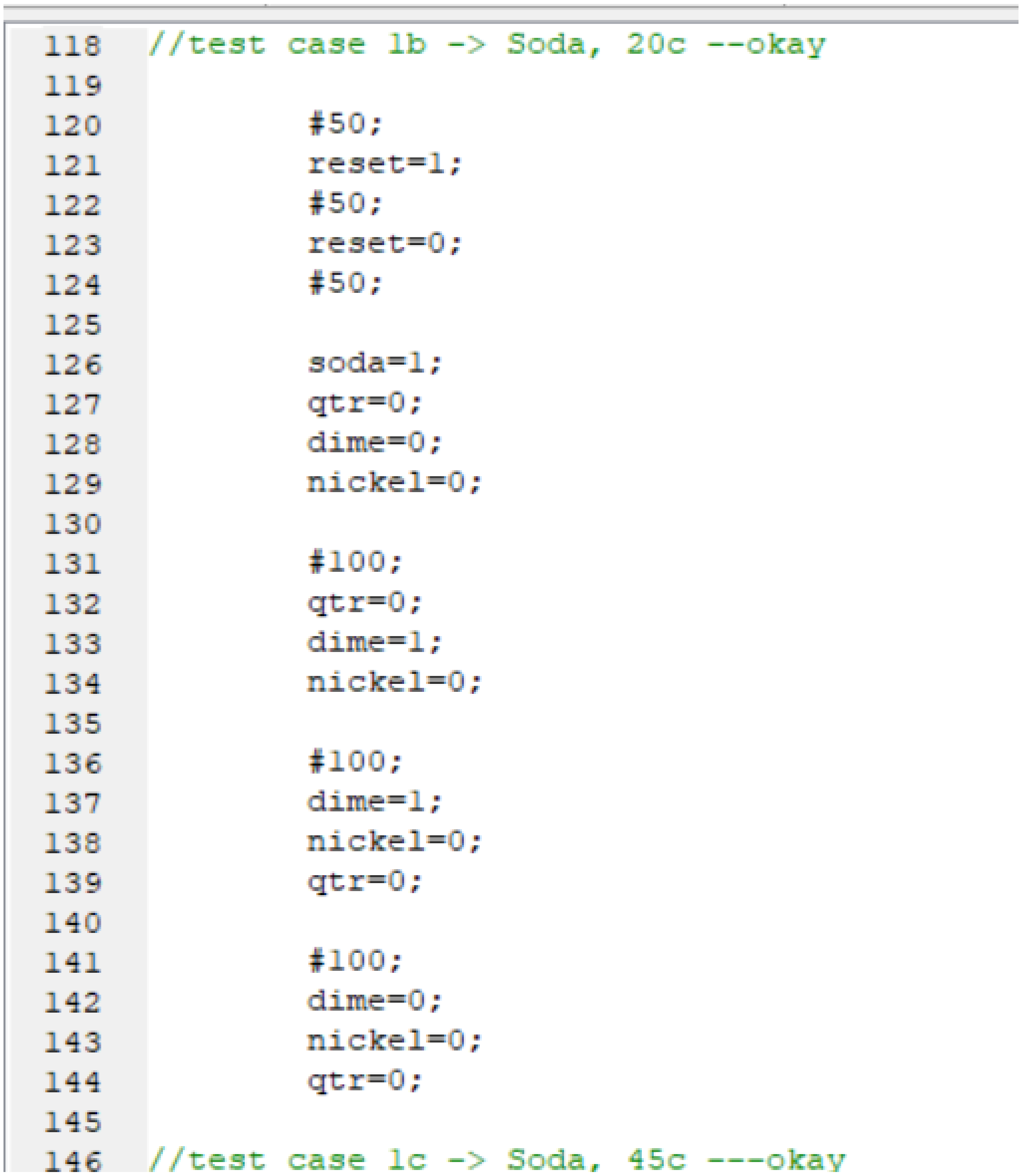
```
module FSMoore_tb;

   // Inputs
   reg clk;
   reg reset;
   reg nickel;
   reg dime;
   reg qtr;
   reg soda;
   reg diet;

   // Outputs
   wire change;
   wire status;
   wire GiveDiet;
   wire GiveSoda;

   // Instantiate the Unit Under Test (UUT)
   Moore uut (
      .clk(clk),
      .reset(reset),
      .nickel(nickel),
      .dime(dime),
      .qtr(qtr),
      .change(change),
      .status(status),
      .soda(soda),
      .diet(diet),
      .GiveDiet(GiveDiet),
      .GiveSoda(GiveSoda)
   );

   always begin
   clk = 1'b0;
   #50;
   clk = 1'b1;
   #50;
   end

   initial begin
//Each Test case here is going to run 2 inputs below 45c
//and 2 inputs above or equal 45c, both for soda and diet inputs
//To run, just uncomment the test case you want to run in the 'initial begin' section
```



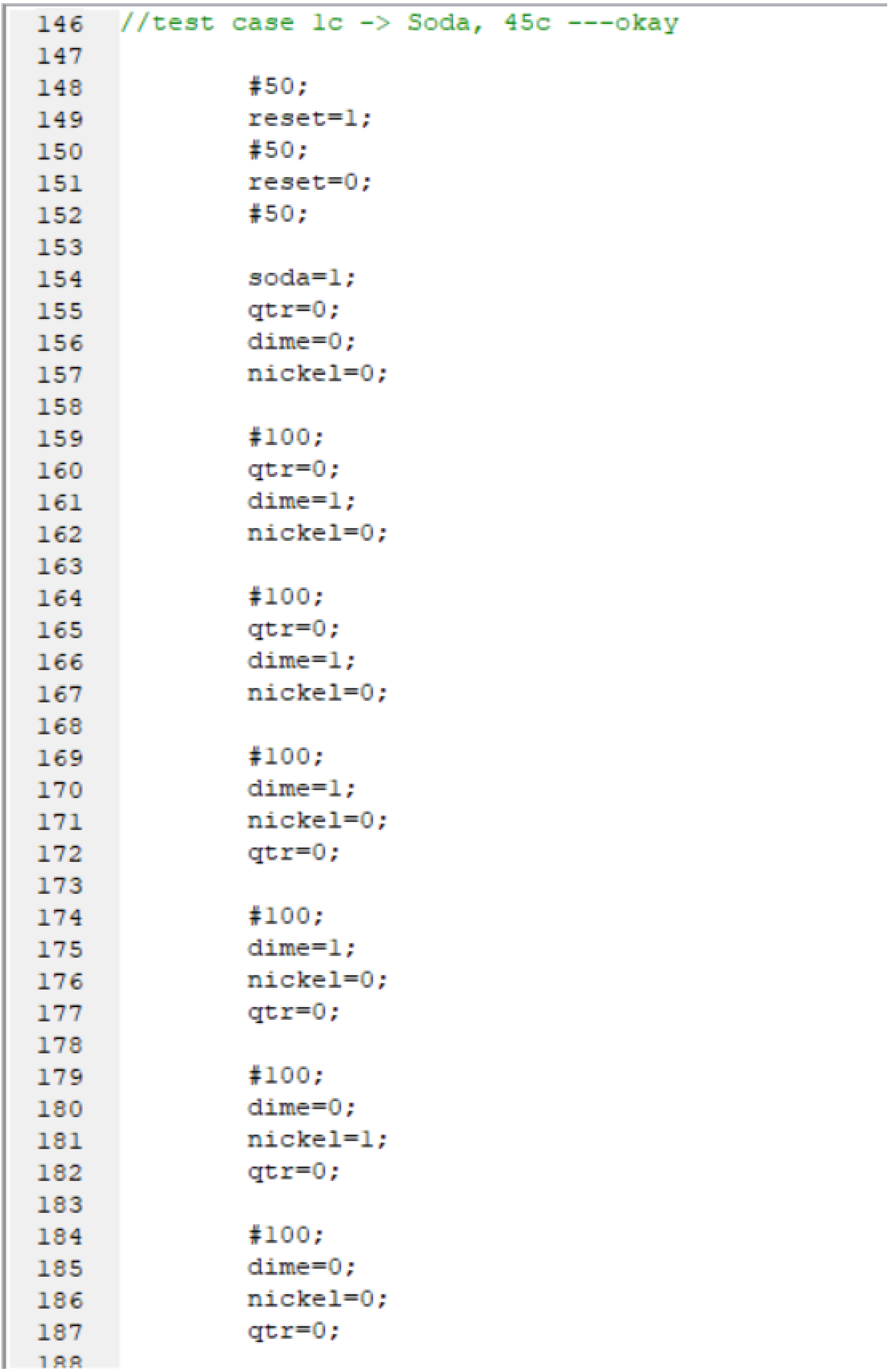
```
   initial begin
//Each Test case here is going to run 2 inputs below 45c
//and 2 inputs above or equal 45c, both for soda and diet inputs
//To run, just uncomment the test case you want to run in the 'initial begin' section
//DO NOT COMMENT OUT THE INITIALIZATION SECTION

//INITIALIZATION
        clk = 0;
        reset = 0;
        nickel = 0;
        dime = 0;
        qtr = 0;
        soda = 0;
        diet = 0;

//test case 1a => Soda, 35c --- okay


        #50;
        reset=1;
        #50;
        reset=0;
        #50;

        soda=1;
        qtr=0;
        dime=0;
        nickel=0;

        #100;
        qtr=0;
        dime=1;
        nickel=0;

        #100;
        dime=1;
        nickel=0;
        qtr=0;

        #100;
        dime=1;
        nickel=0;
        qtr=0;
```



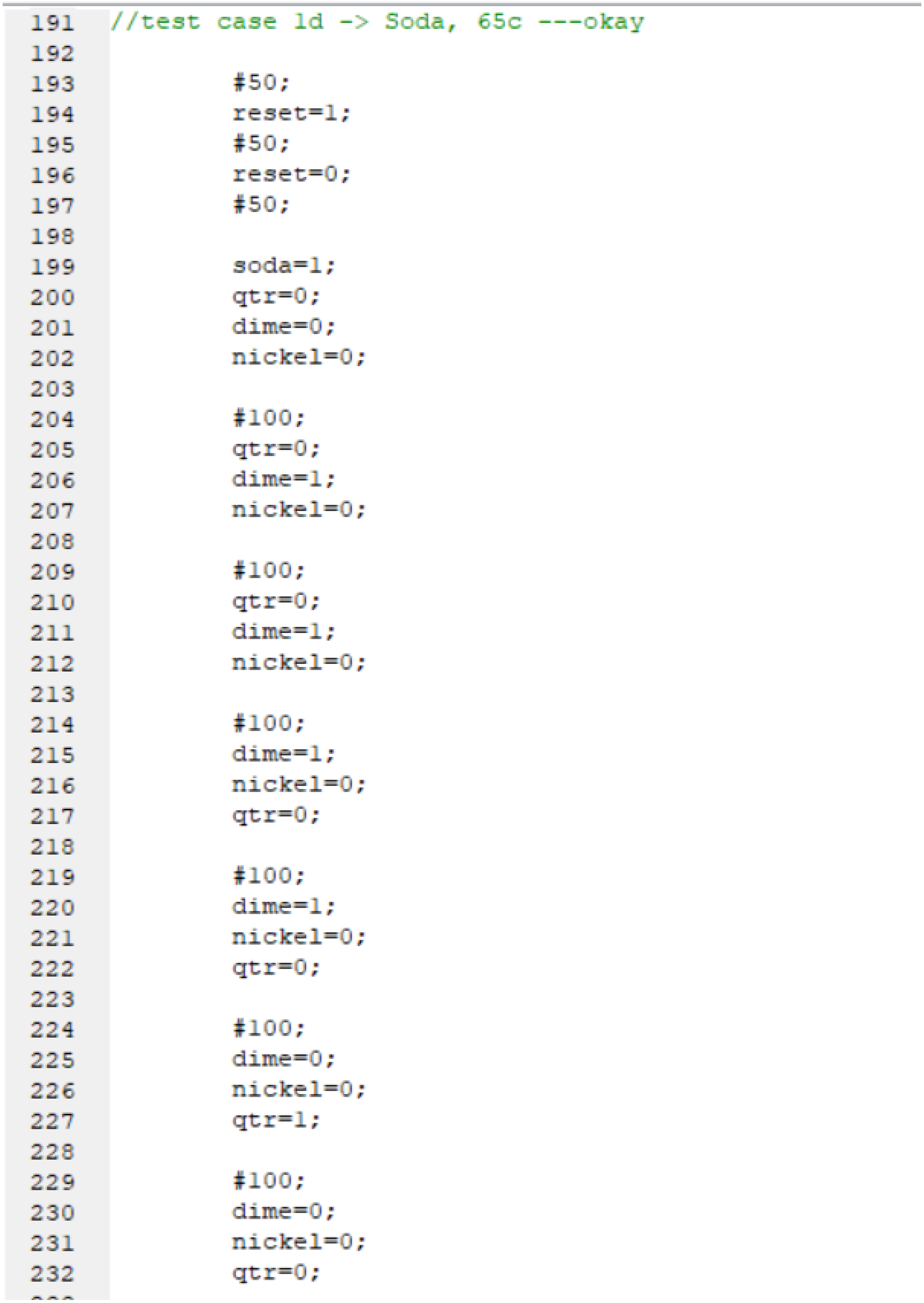
```

//test case 1a => Soda, 35c --- okay


        #50;
        reset=1;
        #50;
        reset=0;
        #50;

        soda=1;
        qtr=0;
        dime=0;
        nickel=0;

        #100;
        qtr=0;
        dime=1;
        nickel=0;

        #100;
        dime=1;
        nickel=0;
        qtr=0;

        #100;
        dime=1;
        nickel=0;
        qtr=0;

        #100;
        dime=0;
        nickel=1;
        qtr=0;

        #100;
        dime=0;
        nickel=0;
        qtr=0;
```



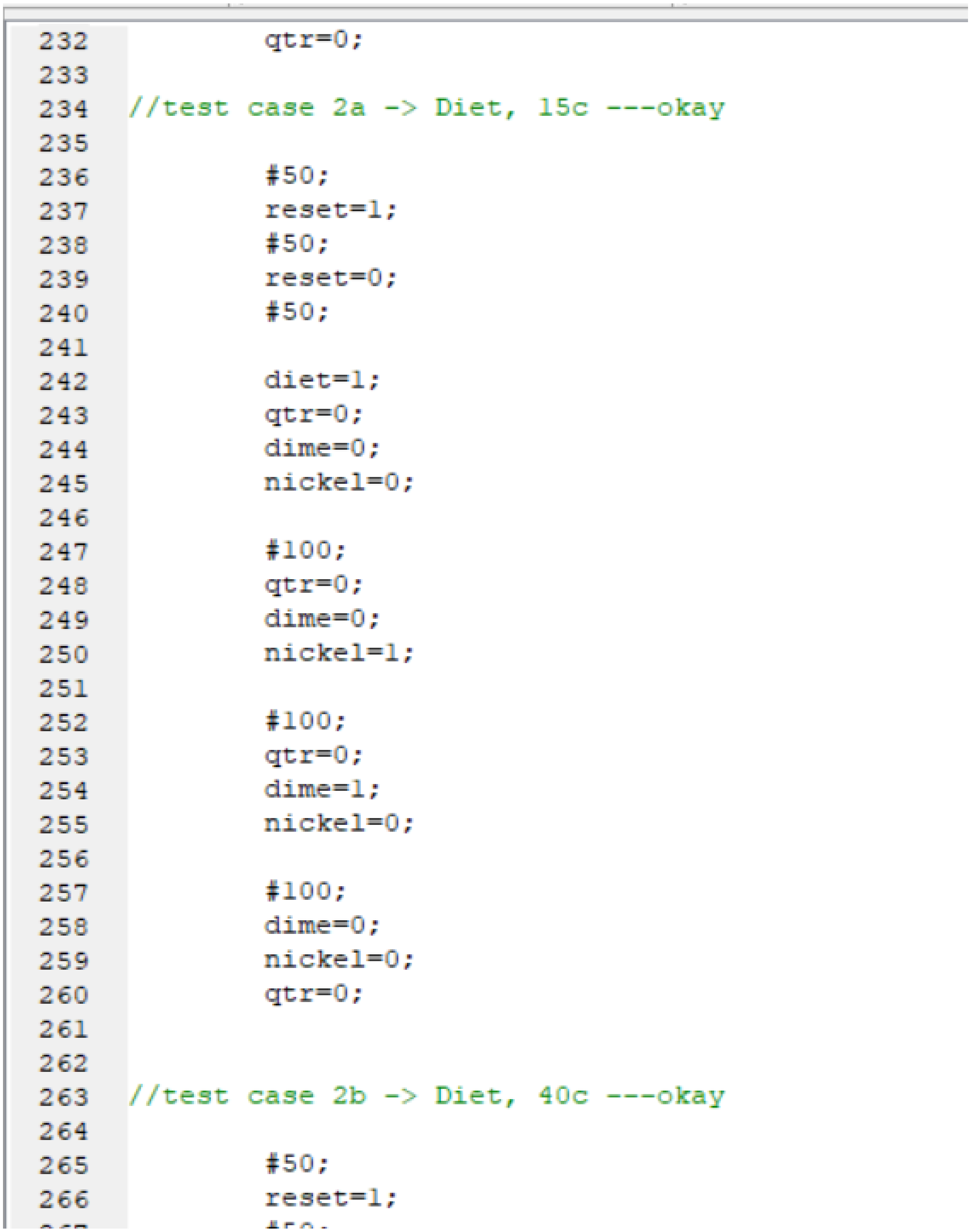
```
//test case 1b -> Soda, 20c --okay

        #50;
        reset=1;
        #50;
        reset=0;
        #50;

        soda=1;
        qtr=0;
        dime=0;
        nickel=0;

        #100;
        qtr=0;
        dime=1;
        nickel=0;

        #100;
        dime=1;
        nickel=0;
        qtr=0;

        #100;
        dime=0;
        nickel=0;
        qtr=0;

//test case 1c -> Soda, 45c ---okay
```



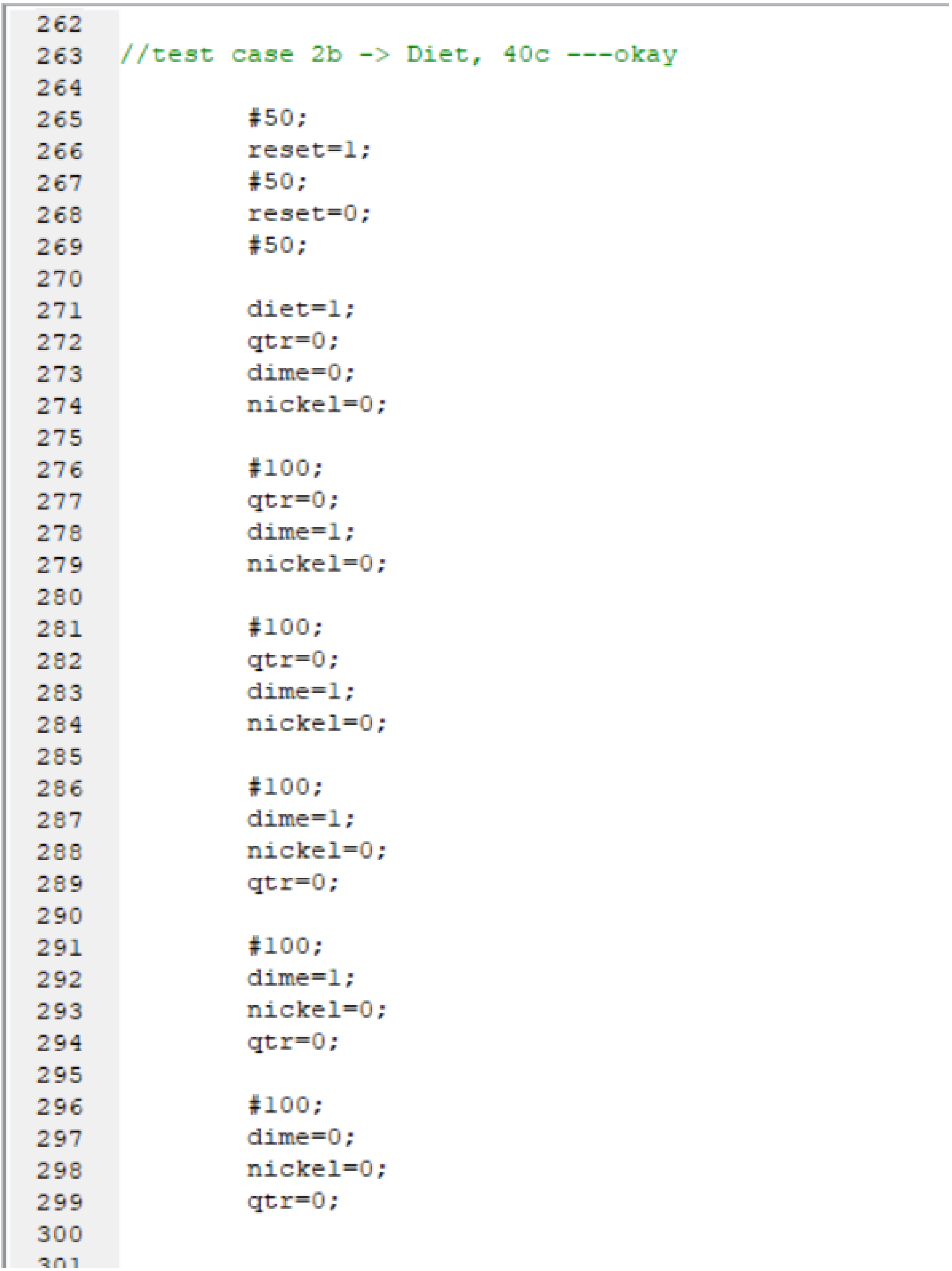
```
//test case 1c -> Soda, 45c ---okay

        #50;
        reset=1;
        #50;
        reset=0;
        #50;

        soda=1;
        qtr=0;
        dime=0;
        nickel=0;

        #100;
        qtr=0;
        dime=1;
        nickel=0;

        #100;
        qtr=0;
        dime=1;
        nickel=0;

        #100;
        dime=1;
        nickel=0;
        qtr=0;

        #100;
        dime=1;
        nickel=0;
        qtr=0;

        #100;
        dime=0;
        nickel=1;
        qtr=0;

        #100;
        dime=0;
        nickel=0;
        qtr=0;
```



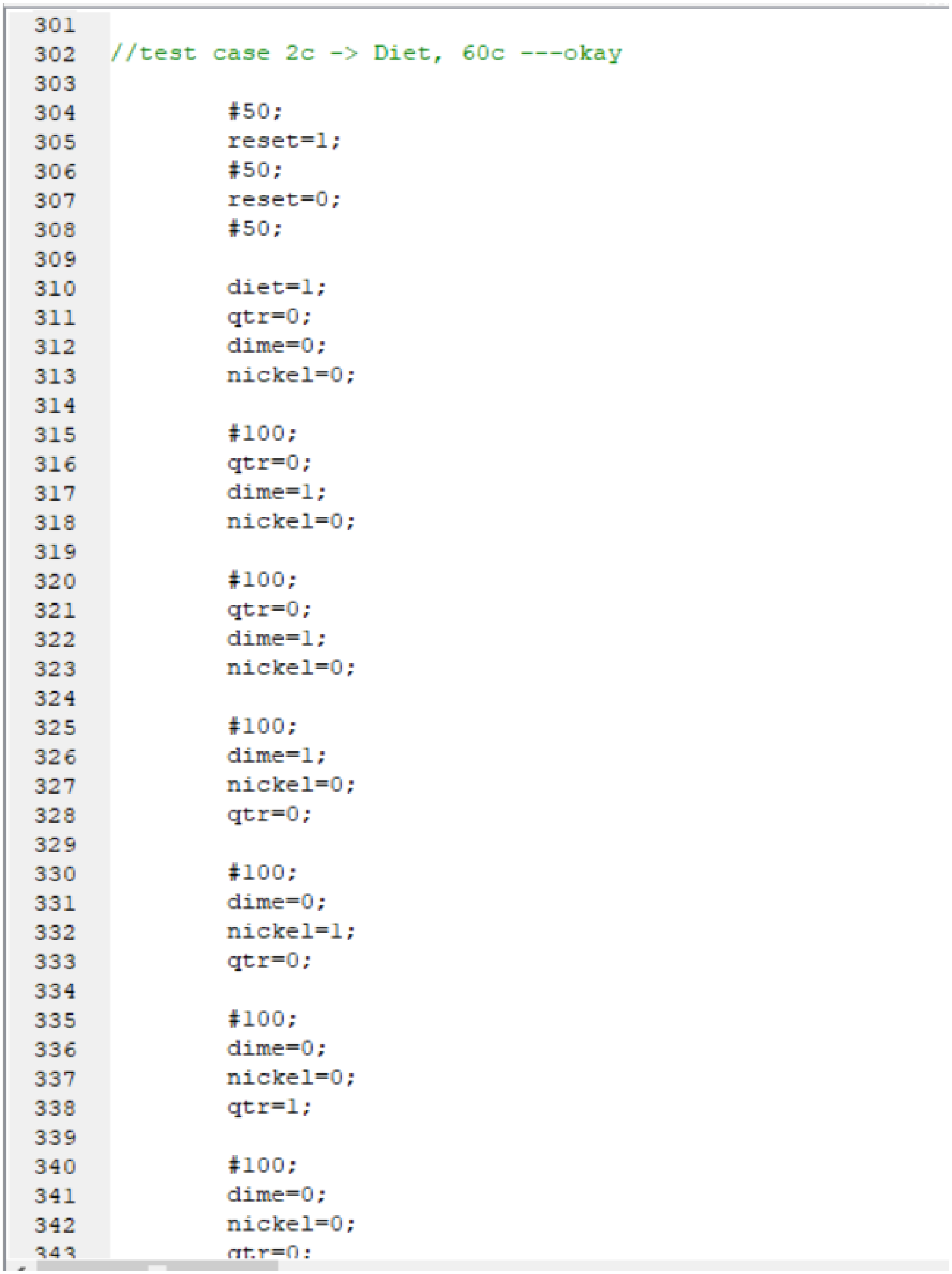
```
//test case 1d -> Soda, 65c ---okay

        #50;
        reset=1;
        #50;
        reset=0;
        #50;

        soda=1;
        qtr=0;
        dime=0;
        nickel=0;

        #100;
        qtr=0;
        dime=1;
        nickel=0;

        #100;
        qtr=0;
        dime=1;
        nickel=0;

        #100;
        dime=1;
        nickel=0;
        qtr=0;

        #100;
        dime=1;
        nickel=0;
        qtr=0;

        #100;
        dime=0;
        nickel=0;
        qtr=1;

        #100;
        dime=0;
        nickel=0;
        qtr=0;
```



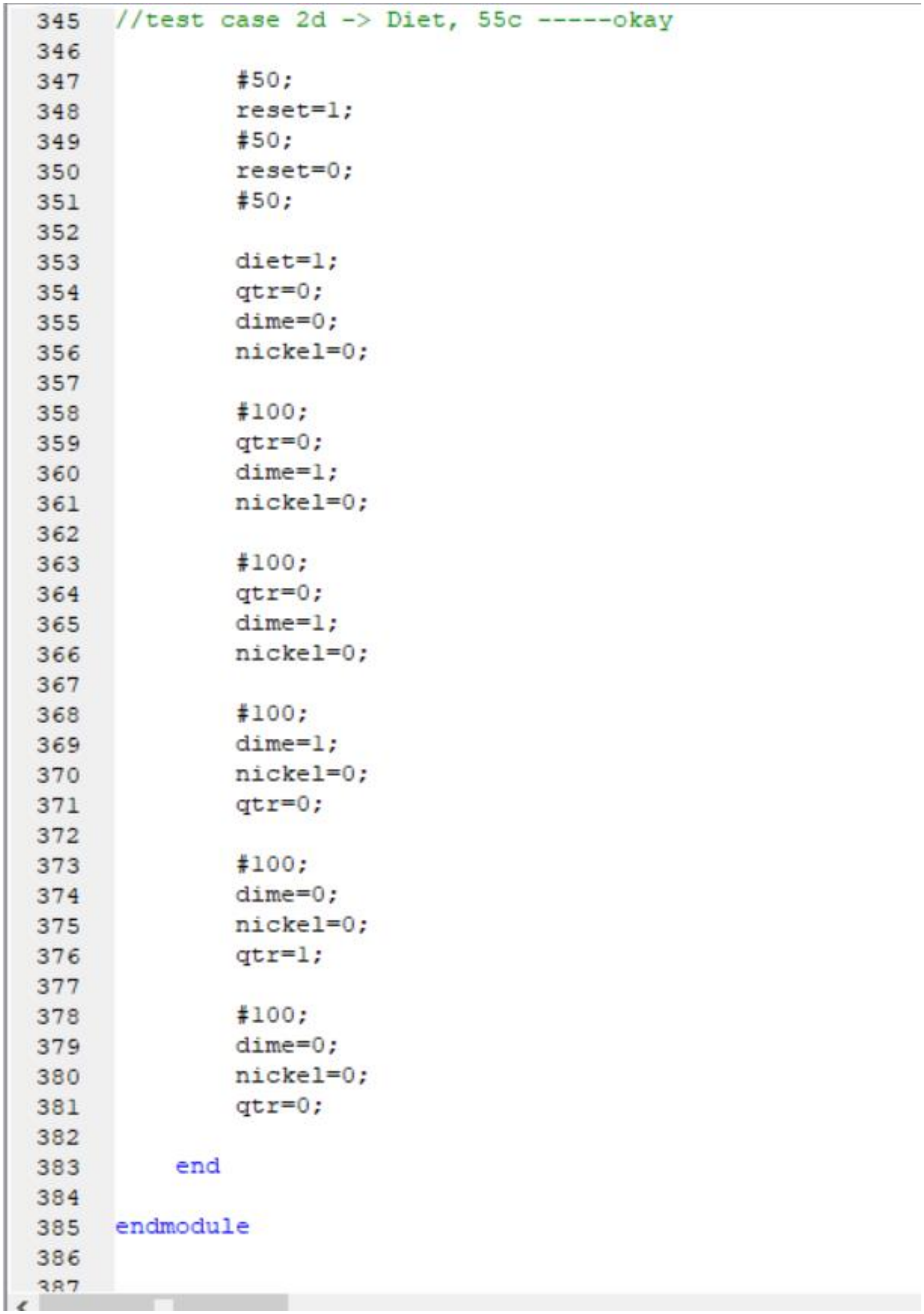
```
        qtr=0;

//test case 2a -> Diet, 15c ---okay

        #50;
        reset=1;
        #50;
        reset=0;
        #50;

        diet=1;
        qtr=0;
        dime=0;
        nickel=0;

        #100;
        qtr=0;
        dime=0;
        nickel=1;

        #100;
        qtr=0;
        dime=1;
        nickel=0;

        #100;
        dime=0;
        nickel=0;
        qtr=0;


//test case 2b -> Diet, 40c ---okay

        #50;
        reset=1;
```



```
//test case 2b -> Diet, 40c ---okay

        #50;
        reset=1;
        #50;
        reset=0;
        #50;

        diet=1;
        qtr=0;
        dime=0;
        nickel=0;

        #100;
        qtr=0;
        dime=1;
        nickel=0;

        #100;
        qtr=0;
        dime=1;
        nickel=0;

        #100;
        dime=1;
        nickel=0;
        qtr=0;

        #100;
        dime=1;
        nickel=0;
        qtr=0;

        #100;
        dime=0;
        nickel=0;
        qtr=0;

```



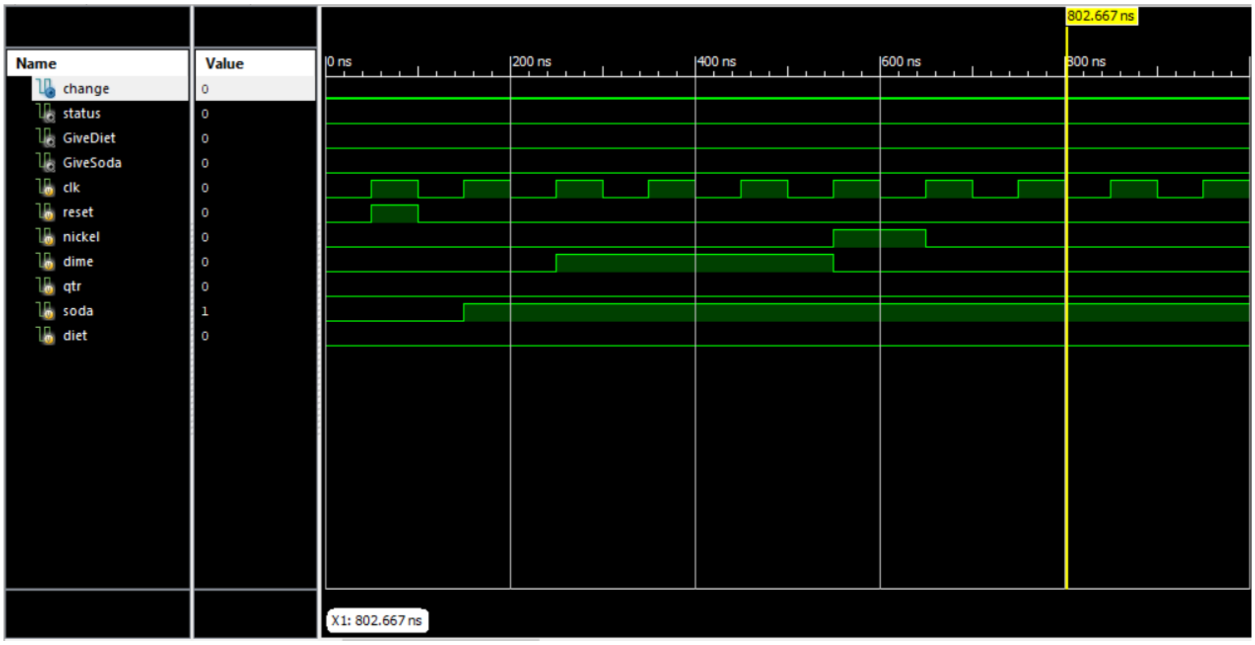
```
//test case 2c -> Diet, 60c ---okay

        #50;
        reset=1;
        #50;
        reset=0;
        #50;

        diet=1;
        qtr=0;
        dime=0;
        nickel=0;

        #100;
        qtr=0;
        dime=1;
        nickel=0;

        #100;
        qtr=0;
        dime=1;
        nickel=0;

        #100;
        dime=1;
        nickel=0;
        qtr=0;

        #100;
        dime=0;
        nickel=1;
        qtr=0;

        #100;
        dime=0;
        nickel=0;
        qtr=1;

        #100;
        dime=0;
        nickel=0;
        qtr=0;
```



```
//test case 2d -> Diet, 55c -----okay

        #50;
        reset=1;
        #50;
        reset=0;
        #50;

        diet=1;
        qtr=0;
        dime=0;
        nickel=0;

        #100;
        qtr=0;
        dime=1;
        nickel=0;

        #100;
        qtr=0;
        dime=1;
        nickel=0;

        #100;
        dime=1;
        nickel=0;
        qtr=0;

        #100;
        dime=0;
        nickel=0;
        qtr=1;

        #100;
        dime=0;
        nickel=0;
        qtr=0;

    end

endmodule

```

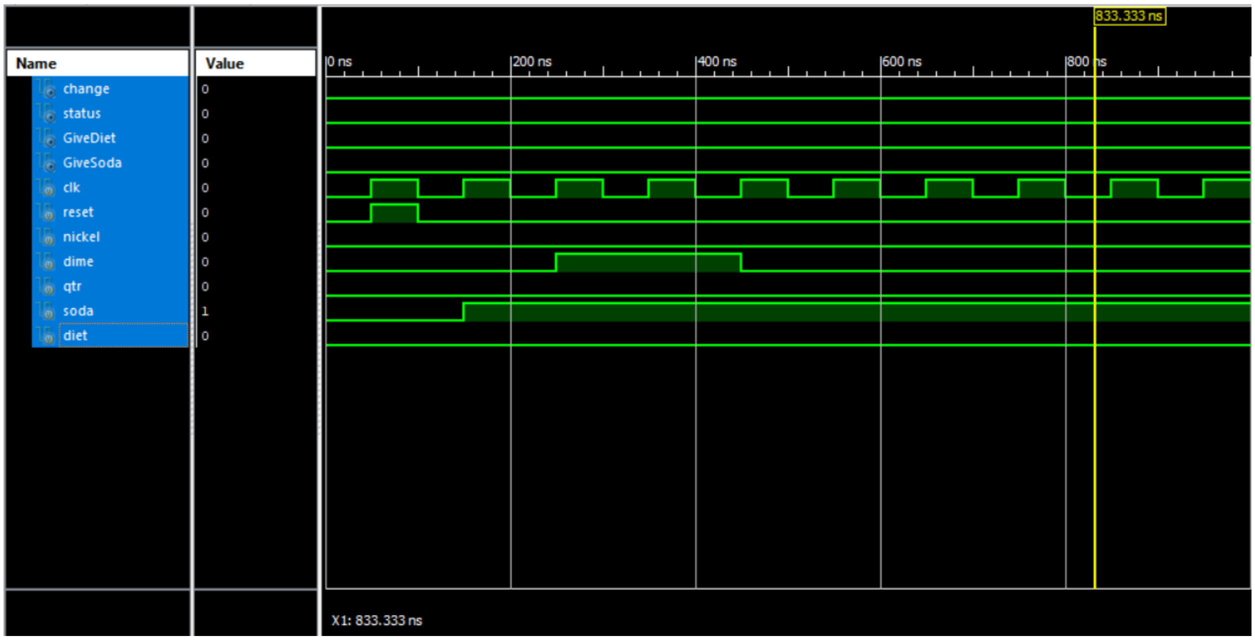
c) Test cases – simulations

I. Test case 1a : Soda, 35c



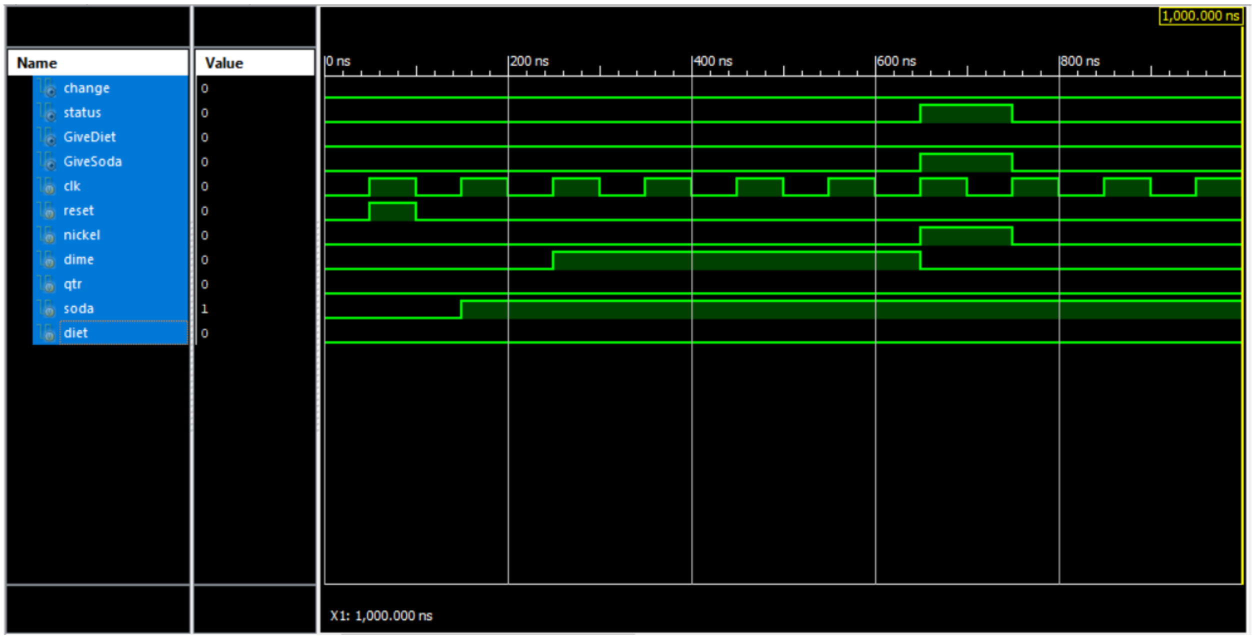
Name	Value
change	0
status	0
GiveDiet	0
GiveSoda	0
clk	0
reset	0
nickel	0
dime	0
qtr	0
soda	1
diet	0

II. Case 1b Soda, 20c



Name	Value
change	0
status	0
GiveDiet	0
GiveSoda	0
clk	0
reset	0
nickel	0
dime	0
qtr	0
soda	1
diet	0

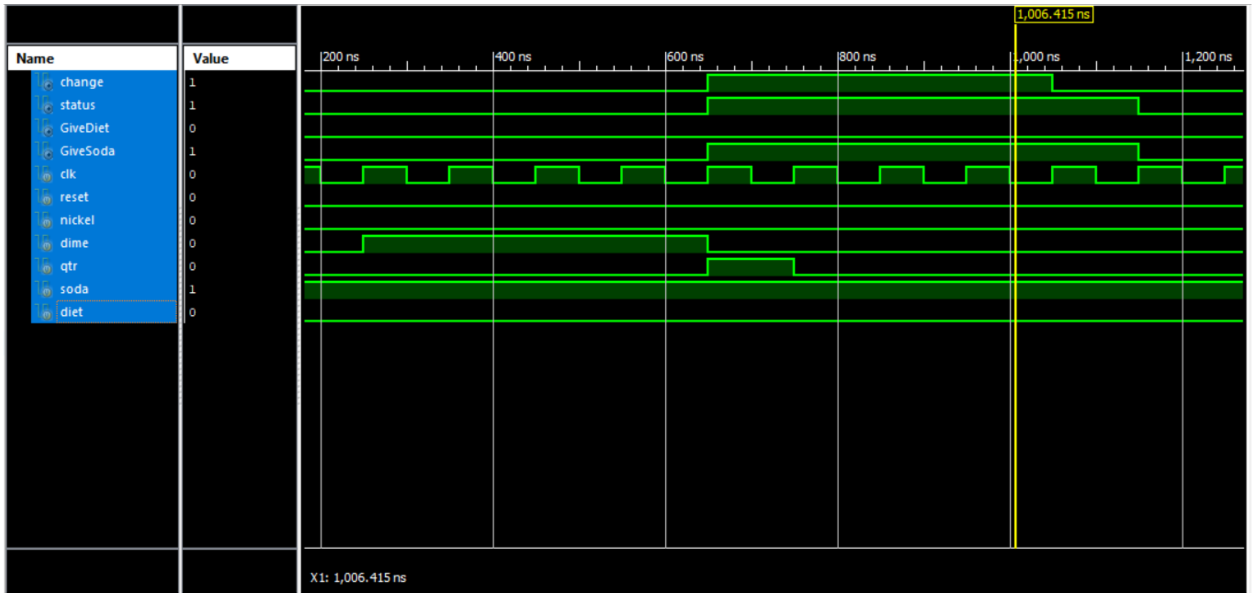
III. Case 1c Soda, 45c



Name	Value
change	0
status	0
GiveDiet	0
GiveSoda	0
clk	0
reset	0
nickel	0
dime	0
qtr	0
soda	1
diet	0

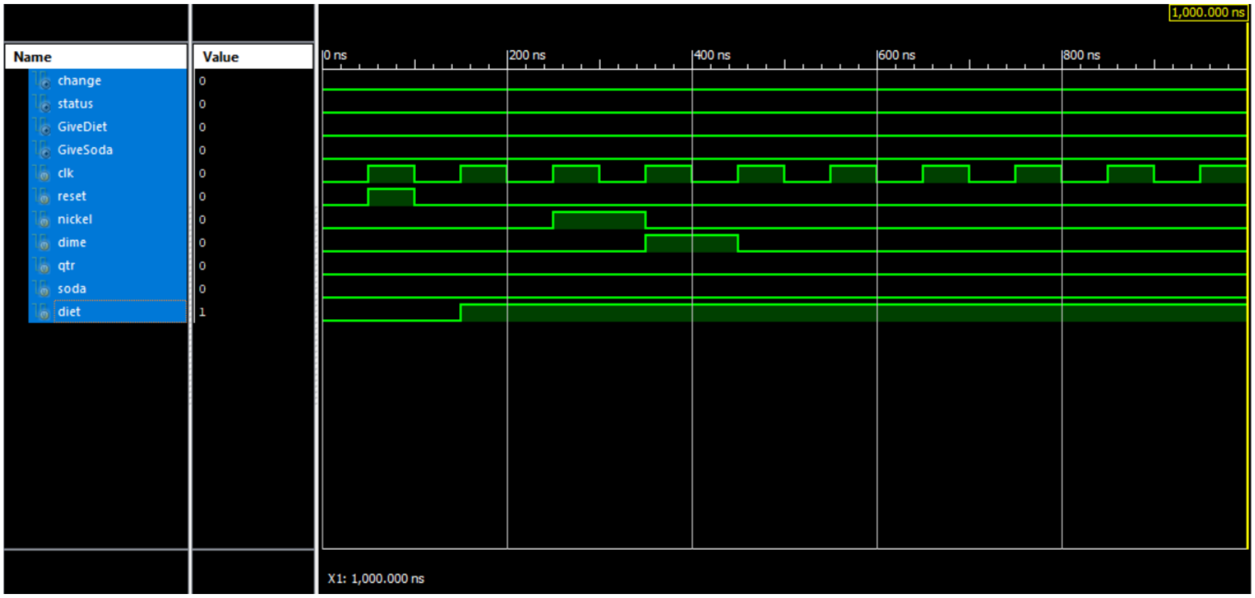
IV. Case 1d Soda, 65c

Note that change has a full clock cycle pulse for each nickel; Change of 20 is 4 cc’s long.



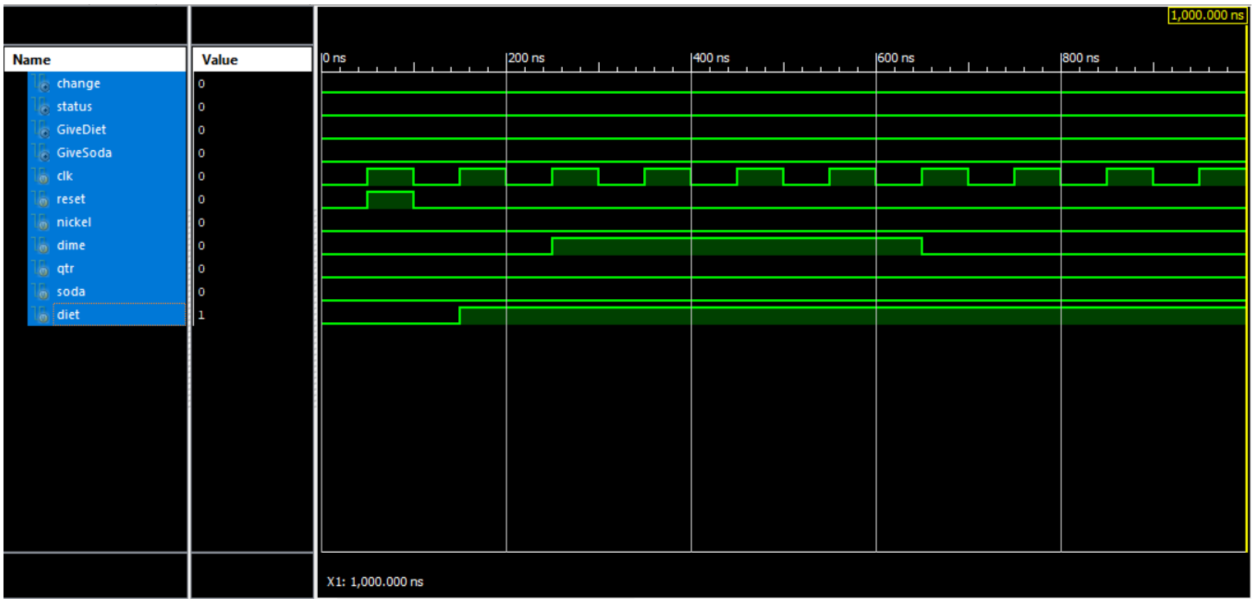
Name	Value
change	1
status	1
GiveDiet	0
GiveSoda	1
clk	0
reset	0
nickel	0
dime	0
qtr	0
soda	1
diet	0

V. Case 2a Diet, 15c



Name	Value
change	0
status	0
GiveDiet	0
GiveSoda	0
clk	0
reset	0
nickel	0
dime	0
qtr	0
soda	0
diet	1

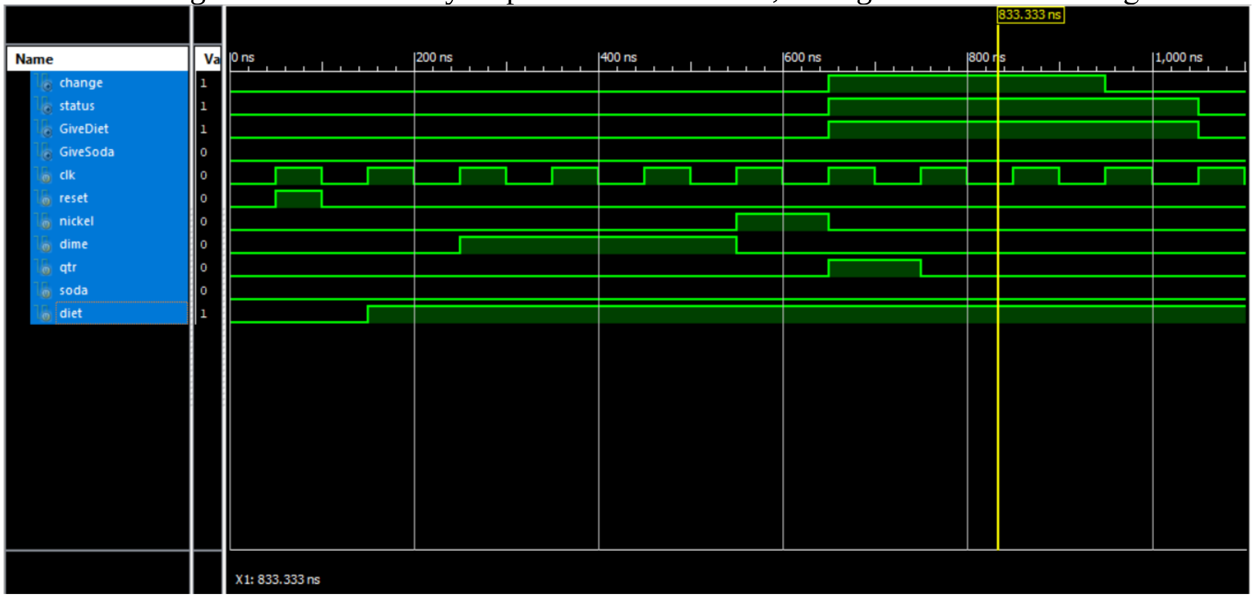
VI. Case 2b Diet, 40c



Name	Value
change	0
status	0
GiveDiet	0
GiveSoda	0
clk	0
reset	0
nickel	0
dime	0
qtr	0
soda	0
diet	1

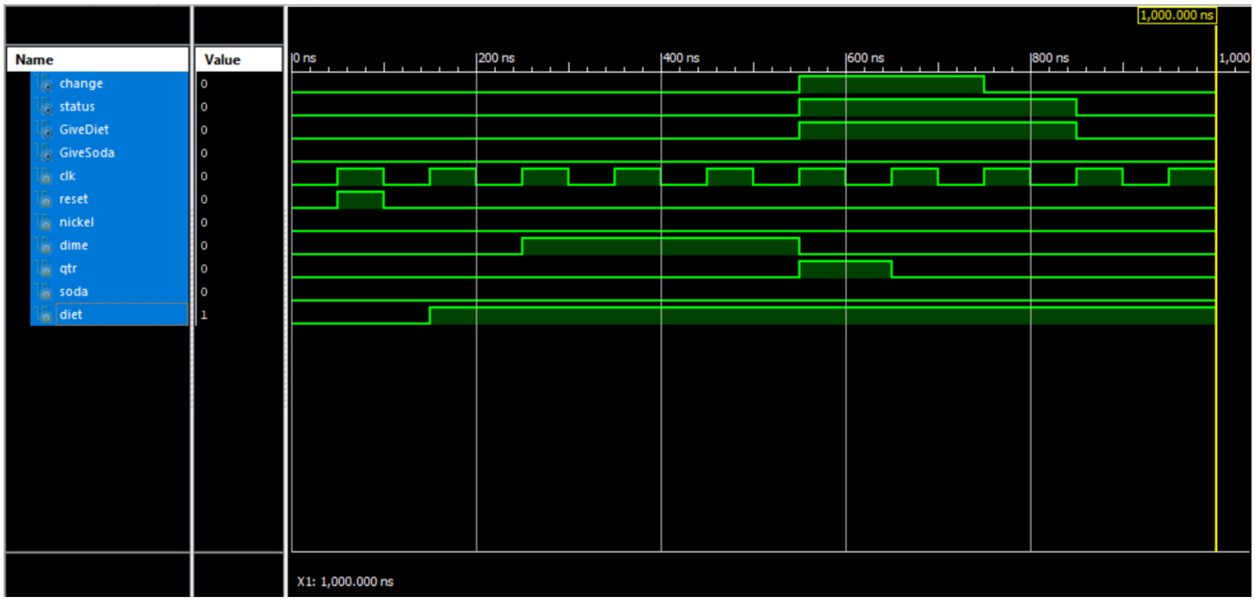
VII. Case 2c Diet, 60c

Note that change has a full clock cycle pulse for each nickel; Change of 15 is 3 cc's long



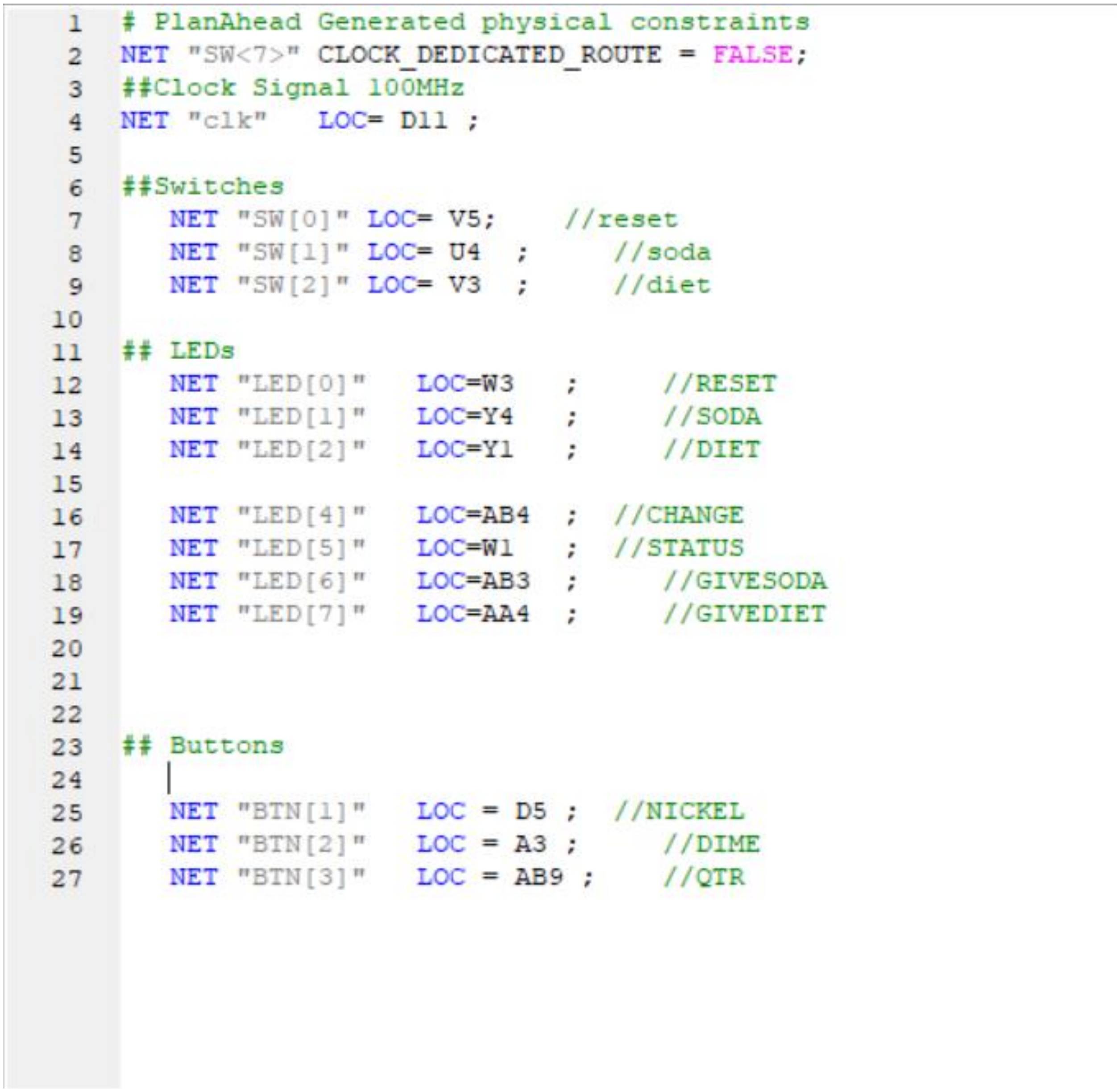
VIII. Case 2d Diet, 55c

Note that change has a full clock cycle pulse for each nickel; Change of 10 is 2 cc's long



Problem 2: Include (a) the **.ucf** file; and (b) synthesis report. *Use as many pages as needed.*

a. L4PINMAP.ucf



```
# PlanAhead Generated physical constraints
NET "SW<7>" CLOCK_DEDICATED_ROUTE = FALSE;
##Clock Signal 100MHz
NET "clk"   LOC= D11 ;

##Switches
   NET "SW[0]" LOC= V5;    //reset
   NET "SW[1]" LOC= U4  ;     //soda
   NET "SW[2]" LOC= V3  ;     //diet

## LEDs
   NET "LED[0]"   LOC=W3   ;     //RESET
   NET "LED[1]"   LOC=Y4   ;     //SODA
   NET "LED[2]"   LOC=Y1   ;     //DIET

   NET "LED[4]"   LOC=AB4  ;  //CHANGE
   NET "LED[5]"   LOC=W1   ;  //STATUS
   NET "LED[6]"   LOC=AB3  ;     //GIVESODA
   NET "LED[7]"   LOC=AA4  ;     //GIVEDIET



## Buttons

   NET "BTN[1]"   LOC = D5 ;  //NICKEL
   NET "BTN[2]"   LOC = A3 ;     //DIME
   NET "BTN[3]"   LOC = AB9 ;    //QTR
```

b. Synthesis Report

```
=========================================================================
==========
*                    Synthesis Options Summary                    *
=========================================================================
==========
---- Source Parameters
Input File Name                    : "L4_boardtest.prj"
Ignore Synthesis Constraint File   : NO

---- Target Parameters
Output File Name                   : "L4_boardtest"
Output Format                      : NGC
Target Device                      : xc6slx45-3-csg484

---- Source Options
Top Module Name                    : L4_boardtest
Automatic FSM Extraction           : YES
FSM Encoding Algorithm             : Auto
Safe Implementation                : No
FSM Style                          : LUT
RAM Extraction                     : Yes
RAM Style                          : Auto
ROM Extraction                     : Yes
Shift Register Extraction          : YES
ROM Style                          : Auto
Resource Sharing                   : YES
Asynchronous To Synchronous        : NO
Shift Register Minimum Size        : 2
Use DSP Block                      : Auto
Automatic Register Balancing       : No

---- Target Options
LUT Combining                      : Auto
Reduce Control Sets                : Auto
Add IO Buffers                     : YES
Global Maximum Fanout              : 100000
Add Generic Clock Buffer(BUFG)     : 16
Register Duplication               : YES
Optimize Instantiated Primitives   : NO
Use Clock Enable                   : Auto
Use Synchronous Set                : Auto
Use Synchronous Reset              : Auto
Pack IO Registers into IOBs        : Auto
```

```
Equivalent register Removal        : YES

---- General Options
Optimization Goal                  : Speed
Optimization Effort                : 1
Power Reduction                    : NO
Keep Hierarchy                     : No
Netlist Hierarchy                  : As_Optimized
RTL Output                         : Yes
Global Optimization                : AllClockNets
Read Cores                         : YES
Write Timing Constraints           : NO
Cross Clock Analysis               : NO
Hierarchy Separator                : /
Bus Delimiter                      : <>
Case Specifier                     : Maintain
Slice Utilization Ratio            : 100
BRAM Utilization Ratio             : 100
DSP48 Utilization Ratio            : 100
Auto BRAM Packing                  : NO
Slice Utilization Ratio Delta      : 5

===============================================================
==========


===============================================================
==========
*                       HDL Parsing                           *
===============================================================
==========
Analyzing Verilog file "C:\Users\kariu\Documents\Spring
2022\CDA4203L\Projects\Lab4\Moore.v" into library work
Parsing module <Moore>.
Analyzing Verilog file "C:\Users\kariu\Documents\Spring
2022\CDA4203L\Projects\Lab4\clock_divider.v" into library work
Parsing module <clock_divider>.
Analyzing Verilog file "C:\Users\kariu\Documents\Spring 2022\CDA4203L\Labs\Lab4
Material\Lab_4___Finite_State_Machine_on_FPGA_export\Sample divider and
debouncer\debouncer.v" into library work
Parsing module <debouncer>.
Analyzing Verilog file "C:\Users\kariu\Documents\Spring
2022\CDA4203L\Projects\Lab4\L4_boardtest.v" into library work
Parsing module <L4_boardtest>.
```

```
===============================================================
==========
*                          HDL Elaboration                           *
===============================================================
==========

Elaborating module <L4_boardtest>.

Elaborating module <clock_divider>.

Elaborating module <debouncer>.

Elaborating module <Moore>.

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

Synthesizing Unit <L4_boardtest>.
    Related source file is "C:\Users\kariu\Documents\Spring
2022\CDA4203L\Projects\Lab4\L4_boardtest.v".
WARNING:Xst:647 - Input <SW<7:3>> is never used. This port will be preserved and
left unconnected if it belongs to a top-level block or it belongs to a sub-block and the
hierarchy of this sub-block is preserved.
WARNING:Xst:647 - Input <BTN<0:0>> is never used. This port will be preserved and
left unconnected if it belongs to a top-level block or it belongs to a sub-block and the
hierarchy of this sub-block is preserved.
    Summary:
            no macro.
Unit <L4_boardtest> synthesized.

Synthesizing Unit <clock_divider>.
    Related source file is "C:\Users\kariu\Documents\Spring
2022\CDA4203L\Projects\Lab4\clock_divider.v".
    Found 26-bit register for signal <count>.
    Found 1-bit register for signal <clk_div>.
    Found 26-bit adder for signal <count[25]_GND_2_o_add_2_OUT> created at line 26.
    Summary:
            inferred   1 Adder/Subtractor(s).
            inferred  27 D-type flip-flop(s).
            inferred   1 Multiplexer(s).
Unit <clock_divider> synthesized.

Synthesizing Unit <debouncer>.
```

```
    Related source file is "C:\Users\kariu\Documents\Spring 2022\CDA4203L\Labs\Lab4
Material\Lab_4___Finite_State_Machine_on_FPGA_export\Sample divider and
debouncer\debouncer.v".
    Found 1-bit register for signal <sync2>.
    Found 1-bit register for signal <sync3>.
    Found 1-bit register for signal <clock_div_prev>.
    Found 1-bit register for signal <out_prev>.
    Found 1-bit register for signal <out>.
    Found 1-bit register for signal <sync>.
    Summary:
            inferred   6 D-type flip-flop(s).
Unit <debouncer> synthesized.

Synthesizing Unit <Moore>.
    Related source file is "C:\Users\kariu\Documents\Spring
2022\CDA4203L\Projects\Lab4\Moore.v".
        s0 = 4'b0000
        s1 = 4'b0001
        s2 = 4'b0010
        s3 = 4'b0011
        s4 = 4'b0100
        s5 = 4'b0101
        s6 = 4'b0110
        s7 = 4'b0111
        s8 = 4'b1000
        s9 = 4'b1001
        s10 = 4'b1010
        s11 = 4'b1011
        s12 = 4'b1100
        s13 = 4'b1101
    Set property "FSM_ENCODING = SEQUENTIAL" for signal <state>.
    Set property "SAFE_IMPLEMENTATION = NO" for signal <state>.
    Set property "FSM_ENCODING = SEQUENTIAL" for signal <next_state>.
    Set property "SAFE_IMPLEMENTATION = NO" for signal <next_state>.
    Found 4-bit register for signal <state>.
    Found finite state machine <FSM_0> for signal <state>.
    -----------------------------------------------------------------------
    | States             | 14                                             |
    | Transitions        | 41                                             |
    | Inputs             | 3                                              |
    | Outputs            | 6                                              |
    | Clock              | clk (rising_edge)                              |
    | Reset              | reset (positive)                               |
    | Reset type         | asynchronous                                   |
    | Reset State        | 0000                                           |
    | Encoding           | SEQUENTIAL                                     |
    | Implementation     | LUT                                  |
    -----------------------------------------------------------------------
    Summary:
           inferred   2 Multiplexer(s).
           inferred   1 Finite State Machine(s).
Unit <Moore> synthesized.

================================================================
==========
HDL Synthesis Report

Macro Statistics
# Adders/Subtractors                              : 1
 26-bit adder                                   : 1
# Registers                                     : 20
 1-bit register                                 : 19
 26-bit register                                : 1
# Multiplexers                                    : 3
 1-bit 2-to-1 multiplexer                         : 2
 26-bit 2-to-1 multiplexer                        : 1
# FSMs                                          : 1

================================================================
==========

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


Synthesizing (advanced) Unit <clock_divider>.
The following registers are absorbed into counter <count>: 1 register on signal <count>.
Unit <clock_divider> synthesized (advanced).

================================================================
==========
Advanced HDL Synthesis Report

Macro Statistics
# Counters                                      : 1
 26-bit up counter                                : 1
# Registers                                     : 19
 Flip-Flops                                     : 19
# Multiplexers                                    : 2
```

```
 1-bit 2-to-1 multiplexer                         : 2
# FSMs                                            : 1

=========================================================================
==========

=========================================================================
==========
*                       Low Level Synthesis                            *
=========================================================================
==========
Optimizing FSM <FSM_vending/FSM_0> on signal <state[1:4]> with SEQUENTIAL
encoding.
-------------------
 State | Encoding
-------------------
 0000  | 0000
 0101  | 0001
 0010  | 0010
 0001  | 0011
 0110  | 0100
 0011  | 0101
 0111  | 0110
 0100  | 0111
 1000  | 1000
 1001  | 1001
 1010  | 1010
 1011  | 1011
 1100  | 1100
 1101  | 1101
-------------------
INFO:Xst:2261 - The FF/Latch <d3/clock_div_prev> in Unit <L4_boardtest> is
equivalent to the following 2 FFs/Latches, which will be removed : <d2/clock_div_prev>
<d1/clock_div_prev>

Optimizing unit <L4_boardtest> ...

Optimizing unit <Moore> ...
WARNING:Xst:1293 - FF/Latch <cd/count_25> has a constant value of 0 in block
<L4_boardtest>. This FF/Latch will be trimmed during the optimization process.

Mapping all equations...
Building and optimizing final netlist ...
Found area constraint ratio of 100 (+ 5) on block L4_boardtest, actual ratio is 0.

Final Macro Processing ...
```

```
Processing Unit <L4_boardtest> :
        Found 2-bit shift register for signal <d3/sync2>.
        Found 2-bit shift register for signal <d2/sync2>.
        Found 2-bit shift register for signal <d1/sync2>.
Unit <L4_boardtest> processed.

=====================================================================
==========
Final Register Report

Macro Statistics
# Registers                                   : 40
 Flip-Flops                                   : 40
# Shift Registers                               : 3
 2-bit shift register                           : 3

=====================================================================
==========

=====================================================================
==========
*                    Partition Report                      *
=====================================================================
==========

Partition Implementation Status
-------------------------------

  No Partitions were found in this design.

-------------------------------

=====================================================================
==========
*                     Design Summary                        *
=====================================================================
==========

Top Level Output File Name         : L4_boardtest.ngc

Primitive and Black Box Usage:
------------------------------
# BELS                           : 126
#      GND                       : 1
#      INV                       : 1
#      LUT1                      : 24
#      LUT2                      : 26
#      LUT3                      : 2
#      LUT4                      : 1
#      LUT5                      : 4
#      LUT6                      : 14
#      MUXCY                     : 24
#      MUXF7                     : 3
#      VCC                       : 1
#      XORCY                     : 25
# FlipFlops/Latches              : 43
#      FD                        : 10
#      FDC                       : 30
#      FDE                       : 3
# Shift Registers                : 3
#      SRLC16E                   : 3
# Clock Buffers                  : 1
#      BUFGP                     : 1
# IO Buffers                     : 13
#      IBUF                      : 6
#      OBUF                      : 7

Device utilization summary:
---------------------------

Selected Device : 6slx45csg484-3


Slice Logic Utilization:
 Number of Slice Registers:              43  out of  54576     0%
 Number of Slice LUTs:                   75  out of  27288     0%
    Number used as Logic:                72  out of  27288     0%
    Number used as Memory:                3  out of   6408     0%
       Number used as SRL:                3

Slice Logic Distribution:
 Number of LUT Flip Flop pairs used:     79
   Number with an unused Flip Flop:      36  out of     79    45%
   Number with an unused LUT:             4  out of     79     5%
   Number of fully used LUT-FF pairs:    39  out of     79    49%
   Number of unique control sets:         4

IO Utilization:
 Number of IOs:                          21
 Number of bonded IOBs:                  14  out of    320     4%
```

```
Specific Feature Utilization:
 Number of BUFG/BUFGCTRLs:              1  out of     16     6%

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

  No Partitions were found in this design.

---------------------------


=========================================================================
Timing Report

NOTE: THESE TIMING NUMBERS ARE ONLY A SYNTHESIS ESTIMATE.
      FOR ACCURATE TIMING INFORMATION PLEASE REFER TO THE TRACE
REPORT
      GENERATED AFTER PLACE-and-ROUTE.

Clock Information:
------------------
-----------------------------------+------------------------------------+-------+
Clock Signal                       | Clock buffer(FF name)              | Load  |
-----------------------------------+------------------------------------+-------+
clk                                | BUFGP                              | 42    |
cd/clk_div                         | NONE(FSM_vending/state_FSM_FFd2)| 4     |
-----------------------------------+------------------------------------+-------+
INFO:Xst:2169 - HDL ADVISOR - Some clock signals were not automatically buffered
by XST with BUFG/BUFR resources. Please use the buffer_type constraint in order to
insert these buffers to the clock signals to help prevent skew problems.

Asynchronous Control Signals Information:
----------------------------------------
No asynchronous control signals found in this design

Timing Summary:
---------------
Speed Grade: -3

   Minimum period: 3.947ns (Maximum Frequency: 253.383MHz)
   Minimum input arrival time before clock: 2.929ns
   Maximum output required time after clock: 4.939ns
   Maximum combinational path delay: 5.536ns
```

```
Timing Details:
---------------
All values displayed in nanoseconds (ns)

=========================================================================
==========
Timing constraint: Default period analysis for Clock 'clk'
  Clock period: 3.947ns (frequency: 253.383MHz)
  Total number of paths / destination ports: 1004 / 39
-------------------------------------------------------------------------
Delay:               3.947ns (Levels of Logic = 3)
  Source:            cd/count_9 (FF)
  Destination:       cd/count_0 (FF)
  Source Clock:      clk rising
  Destination Clock: clk rising

  Data Path: cd/count_9 to cd/count_0
                                Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     FDC:C->Q              2   0.447   0.981  cd/count_9 (cd/count_9)
     LUT6:I0->O            2   0.203   0.845  cd/GND_2_o_GND_2_o_equal_7_o<25>4
(cd/GND_2_o_GND_2_o_equal_7_o<25>3)
     LUT6:I3->O           14   0.205   0.958  cd/GND_2_o_GND_2_o_equal_7_o<25>5
(cd/GND_2_o_GND_2_o_equal_7_o)
     LUT2:I1->O            1   0.205   0.000  cd/Mcount_count_eqn_01
(cd/Mcount_count_eqn_0)
     FDC:D                     0.102          cd/count_0
    ----------------------------------------
    Total                      3.947ns (1.162ns logic, 2.785ns route)
                                       (29.4% logic, 70.6% route)

=========================================================================
==========
Timing constraint: Default period analysis for Clock 'cd/clk_div'
  Clock period: 2.790ns (frequency: 358.468MHz)
  Total number of paths / destination ports: 29 / 4
-------------------------------------------------------------------------
Delay:               2.790ns (Levels of Logic = 2)
  Source:            FSM_vending/state_FSM_FFd2 (FF)
  Destination:       FSM_vending/state_FSM_FFd2 (FF)
  Source Clock:      cd/clk_div rising
  Destination Clock: cd/clk_div rising

  Data Path: FSM_vending/state_FSM_FFd2 to FSM_vending/state_FSM_FFd2
                                Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     FDC:C->Q            12   0.447   1.253  FSM_vending/state_FSM_FFd2
(FSM_vending/state_FSM_FFd2)
     LUT6:I1->O           1   0.203   0.580  FSM_vending/state_FSM_FFd2-In1
(FSM_vending/state_FSM_FFd2-In1)
     LUT6:I5->O           1   0.205   0.000  FSM_vending/state_FSM_FFd2-In2
(FSM_vending/state_FSM_FFd2-In)
     FDC:D                    0.102          FSM_vending/state_FSM_FFd2
    ----------------------------------------
    Total                      2.790ns (0.957ns logic, 1.833ns route)
                                       (34.3% logic, 65.7% route)

=========================================================================
==========
Timing constraint: Default OFFSET IN BEFORE for Clock 'clk'
  Total number of paths / destination ports: 29 / 29
-------------------------------------------------------------------------
Offset:              2.929ns (Levels of Logic = 1)
  Source:            SW<0> (PAD)
  Destination:       cd/count_0 (FF)
  Destination Clock: clk rising

  Data Path: SW<0> to cd/count_0
                                Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     IBUF:I->O           31   1.222   1.277  SW_0_IBUF (LED_0_OBUF)
     FDC:CLR                  0.430          cd/count_0
    ----------------------------------------
    Total                      2.929ns (1.652ns logic, 1.277ns route)
                                       (56.4% logic, 43.6% route)

=========================================================================
==========
Timing constraint: Default OFFSET IN BEFORE for Clock 'cd/clk_div'
  Total number of paths / destination ports: 4 / 4
-------------------------------------------------------------------------
Offset:              2.929ns (Levels of Logic = 1)
  Source:            SW<0> (PAD)
  Destination:       FSM_vending/state_FSM_FFd2 (FF)
  Destination Clock: cd/clk_div rising

  Data Path: SW<0> to FSM_vending/state_FSM_FFd2
                                Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     IBUF:I->O            31   1.222   1.277  SW_0_IBUF (LED_0_OBUF)
     FDC:CLR                    0.430         FSM_vending/state_FSM_FFd4
    ----------------------------------------
    Total                      2.929ns (1.652ns logic, 1.277ns route)
                                       (56.4% logic, 43.6% route)

=========================================================================
==========
Timing constraint: Default OFFSET OUT AFTER for Clock 'cd/clk_div'
  Total number of paths / destination ports: 15 / 4
-------------------------------------------------------------------------
Offset:              4.939ns (Levels of Logic = 2)
  Source:            FSM_vending/state_FSM_FFd2 (FF)
  Destination:       LED<7> (PAD)
  Source Clock:      cd/clk_div rising

  Data Path: FSM_vending/state_FSM_FFd2 to LED<7>
                                Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     FDC:C->Q             12   0.447   1.137  FSM_vending/state_FSM_FFd2
(FSM_vending/state_FSM_FFd2)
     LUT5:I2->O            1   0.205   0.579  FSM_vending/Mmux_GiveDiet11
(LED_7_OBUF)
     OBUF:I->O                 2.571          LED_7_OBUF (LED<7>)
    ----------------------------------------
    Total                      4.939ns (3.223ns logic, 1.716ns route)
                                       (65.3% logic, 34.7% route)

=========================================================================
==========
Timing constraint: Default path analysis
  Total number of paths / destination ports: 5 / 5
-------------------------------------------------------------------------
Delay:               5.536ns (Levels of Logic = 3)
  Source:            SW<2> (PAD)
  Destination:       LED<7> (PAD)

  Data Path: SW<2> to LED<7>
                                Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     IBUF:I->O             2   1.222   0.961  SW_2_IBUF (LED_2_OBUF)
     LUT5:I0->O            1   0.203   0.579  FSM_vending/Mmux_GiveDiet11
(LED_7_OBUF)
```

```
    OBUF:I->O              2.571        LED_7_OBUF (LED<7>)
    ----------------------------------------
    Total                      5.536ns (3.996ns logic, 1.540ns route)
                                        (72.2% logic, 27.8% route)

=========================================================================
==========

Cross Clock Domains Report:
--------------------------

Clock to Setup on destination clock cd/clk_div
---------------+---------+---------+---------+---------+
               | Src:Rise| Src:Fall| Src:Rise| Src:Fall|
Source Clock   |Dest:Rise|Dest:Rise|Dest:Fall|Dest:Fall|
---------------+---------+---------+---------+---------+
cd/clk_div     |    2.790|         |         |         |
clk            |    2.569|         |         |         |
---------------+---------+---------+---------+---------+

Clock to Setup on destination clock clk
---------------+---------+---------+---------+---------+
               | Src:Rise| Src:Fall| Src:Rise| Src:Fall|
Source Clock   |Dest:Rise|Dest:Rise|Dest:Fall|Dest:Fall|
---------------+---------+---------+---------+---------+
clk            |    3.947|         |         |         |
---------------+---------+---------+---------+---------+

=========================================================================
==========


Total REAL time to Xst completion: 6.00 secs
Total CPU time to Xst completion: 5.93 secs

-->

Total memory usage is 4486176 kilobytes

Number of errors   :    0 (   0 filtered)
Number of warnings :    3 (   0 filtered)
Number of infos    :    2 (   0 filtered)
```