Digital Design Diploma

Assignment 5

FSM & Memories

Name	Kareem Hassan Atif
Group	G2

Submitted to: Eng. Kareem Waseem

1. Tesla Car:

```
module Tesla_Car (input clk,rst, input [7:0] speed_limit,car_speed, input [6:0] leading_distance,
output reg unlock_doors, accelerate_car);

parameter MIN_DISTANCE = 7'd40;
localparam STOP = 2'b00,  ACCELERATE = 2'b01,  DECELERATE = 2'b10;
(*fsm_encoding = "gray"*)
reg [1:0] cs,ns;

/*--State Memory--*/
always @(posedge clk or posedge rst) begin
    if (rst)         cs <= STOP;
    else             cs <= ns;
end

/*--Next State and Output Logic--*/
always @(*) begin
    case (cs)
        STOP: begin
            {unlock_doors,accelerate_car} = {1'b1,1'b0};
            if (leading_distance >= MIN_DISTANCE)   ns = ACCELERATE;
            else                                    ns = STOP;
        end

        ACCELERATE: begin
            {unlock_doors,accelerate_car} = {1'b0,1'b1};
            if (leading_distance >= MIN_DISTANCE && car_speed < speed_limit)   ns = ACCELERATE;
            else                                                               ns = DECELERATE;
        end

        DECELERATE: begin
            {unlock_doors,accelerate_car} = {1'b0,1'b0};
            if (!car_speed)                                                        ns = STOP;
            else if (leading_distance >= MIN_DISTANCE && car_speed < speed_limit)  ns = ACCELERATE;
            else                                                                   ns = DECELERATE;
        end

        default: {unlock_doors,accelerate_car, ns} = {1'b1,1'b0, STOP};
    endcase
end
endmodule
```

Figure 1: Q1 Code

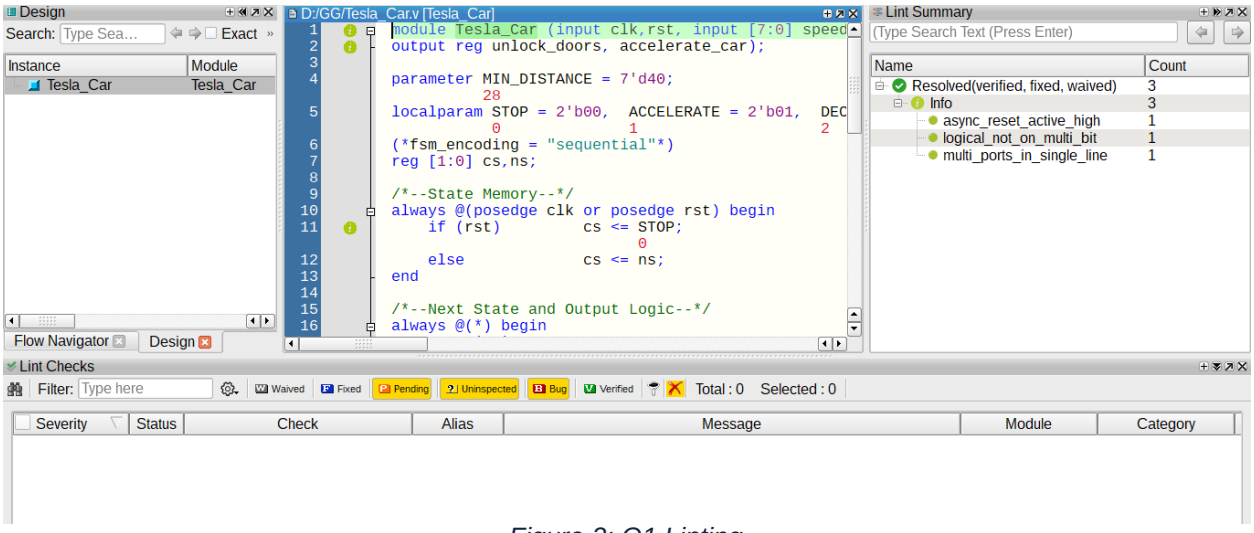

Figure 2: Q1 Linting

```
module Tesla_Car_tb ();
reg clk,rst;
reg [7:0] speed_limit,car_speed;
reg [6:0] leading_distance;
wire unlock_doors, accelerate_car;

/*module Tesla_Car (input clk,rst, input [7:0] speed_limit,car_speed, input [6:0] leading_distance,
output reg unlock_doors, accelerate_car);*/
Tesla_Car DUT(clk,rst,speed_limit,car_speed,leading_distance,unlock_doors, accelerate_car);

initial begin
    clk=0;
    forever #5 clk=~clk;
end

initial begin
    rst=1; speed_limit=$random; car_speed=$random; leading_distance=$random;
    #2;
    if (DUT.cs != DUT.STOP || !unlock_doors || accelerate_car) begin
        $display ("Error in reset!!");
        $stop;
    end

    rst=0; speed_limit=$random; car_speed=$random; leading_distance=41;
    @(negedge clk);
    if (DUT.cs != DUT.ACCELERATE || unlock_doors || !accelerate_car) begin
        $display ("Error in ACCELERATE!!");
        $stop;
    end

    speed_limit=80; car_speed=70; leading_distance=41;
    @(negedge clk);
    if (DUT.cs != DUT.ACCELERATE || unlock_doors || !accelerate_car) begin
        $display ("Error in ACCELERATE!!");
        $stop;
    end

    speed_limit=80; car_speed=90; leading_distance=41;
    @(negedge clk);
    if (DUT.cs != DUT.DECELERATE || unlock_doors || accelerate_car) begin
        $display ("Error in DECELERATE!!");
        $stop;
    end

    car_speed=0;
    @(negedge clk);
    if (DUT.cs != DUT.STOP || !unlock_doors || accelerate_car) begin
        $display ("Error in STOP!!");
        $stop;
    end

    repeat (10) begin
        speed_limit=$random; car_speed=$random; leading_distance=$random;
        @(negedge clk);
    end
    $stop;
end
endmodule
```

Figure 3: Q1 Testbench

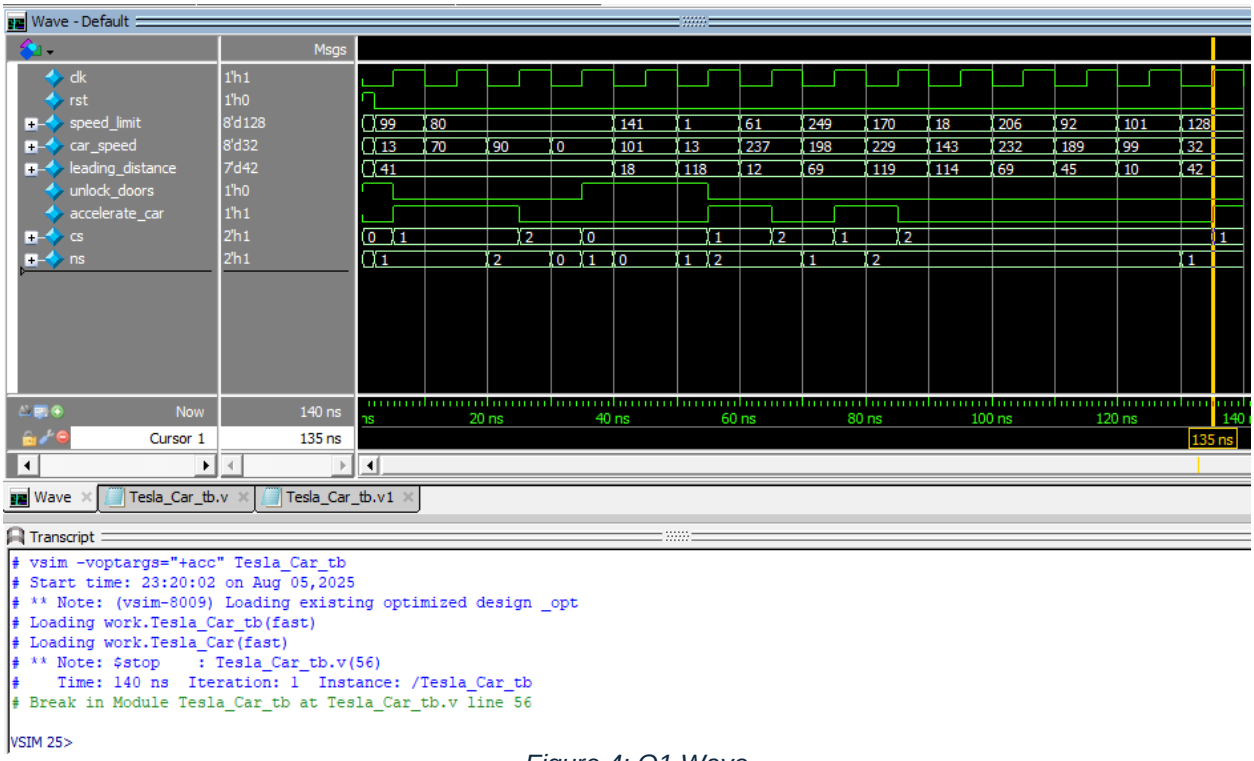

Figure 4: Q1 Wave

```
vlib work
vlog Tesla_Car.v Tesla_Car_tb.v
vsim -voptargs=+acc Tesla_Car_tb
add wave *
add wave -position insertpoint  \
sim:/Tesla_Car_tb/DUT/cs \
sim:/Tesla_Car_tb/DUT/ns
run -all
#quit -sim
```

Figure 5: Q1 DO File

```
## This file is a general .xdc for the Basys3 rev B board
## To use it in a project:
## - uncomment the lines corresponding to used pins
## - rename the used ports (in each line, after get_ports) according to the top level signal names in the project

## Clock signal
set_property -dict { PACKAGE_PIN W5   IOSTANDARD LVCMOS33 } [get_ports clk]
create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports clk]

## Configuration options, can be used for all designs
set_property CONFIG_VOLTAGE 3.3 [current_design]
set_property CFGBVS VCCO [current_design]

## SPI configuration mode options for QSPI boot, can be used for all designs
set_property BITSTREAM.GENERAL.COMPRESS TRUE [current_design]
set_property BITSTREAM.CONFIG.CONFIGRATE 33 [current_design]
set_property CONFIG_MODE SPIx4 [current_design]
```

Figure 6: Q1 Constraint File

1.1 Gray Encoding:

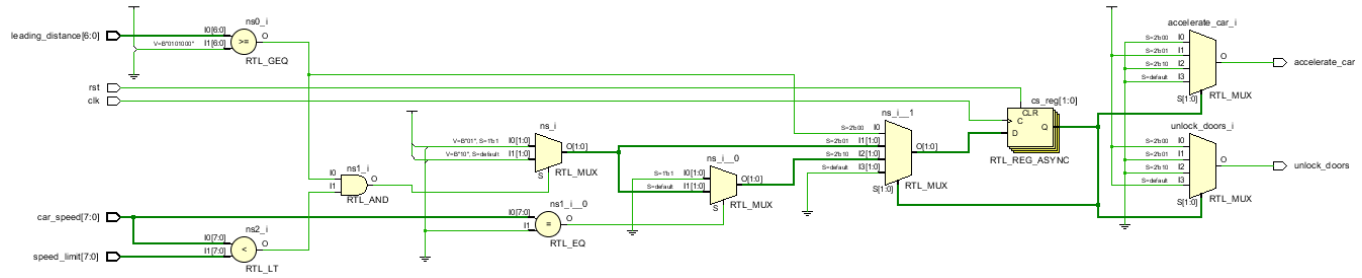

Figure 7: Q1 RTL (Gray)

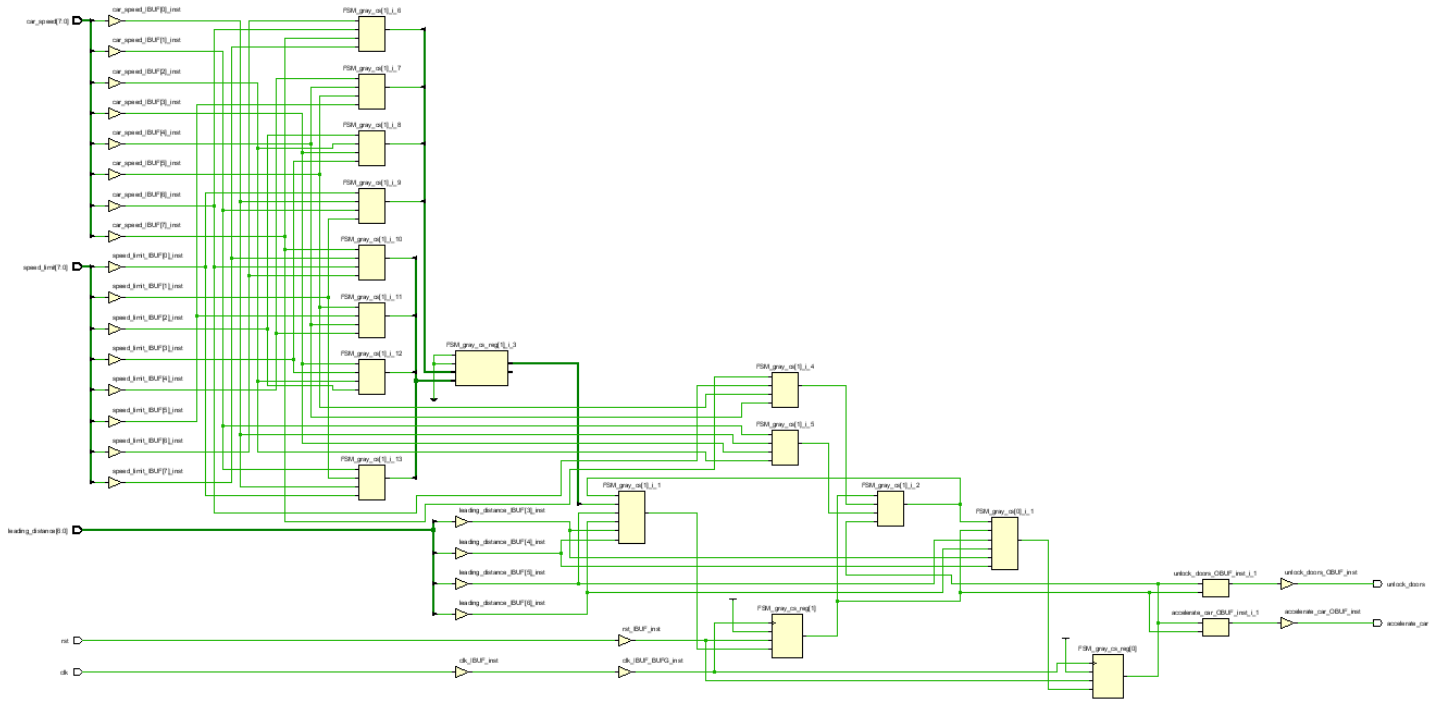

Figure 8: Q1 Synthesis (Gray)

State	New Encoding	Previous Encoding
STOP	00	00
ACCELERATE	01	01
DECELERATE	11	10

INFO: [Synth 8-3354] encoded FSM with state register 'cs_reg' using encoding 'gray' in module 'Tesla_Car'

Figure 9: Q1 Encoding Report (Gray)

```
// Copyright 1986-2018 Xilinx, Inc. All Rights Reserved.
// ----------------------------------------------------------------------------
// Tool Version: Vivado v.2018.2 (win64) Build 2258646 Thu Jun 14 20:03:12 MDT 2018
// Date         : Wed Aug  6 00:06:56 2025
// Host         : DESKTOP-DTNSAHB running 64-bit major release  (build 9200)
// Command      : write_verilog {D:/College Projects/Digital_Course_Waseem/Vv/Q1/Tesla_Car_Gray.v}
// Design       : Tesla_Car
// Purpose      : This is a Verilog netlist of the current design or from a specific cell of the design. The output is an
//                IEEE 1364-2001 compliant Verilog HDL file that contains netlist information obtained from the input
//                design files.
// Device       : xc7a35ticpg236-1L
// ----------------------------------------------------------------------------
`timescale 1 ps / 1 ps

(* ACCELERATE = "2'b01" *) (* DECELERATE = "2'b10" *) (* MIN_DISTANCE = "7'b0101000" *)
(* STOP = "2'b00" *)
(* STRUCTURAL_NETLIST = "yes" *)
module Tesla_Car
   (clk,
    rst,
    speed_limit,
    car_speed,
    leading_distance,
    unlock_doors,
    accelerate_car);
  input clk;
  input rst;
  input [7:0]speed_limit;
  input [7:0]car_speed;
  input [6:0]leading_distance;
  output unlock_doors;
  output accelerate_car;
```

Figure 10: Q1 Netlist (Gray)

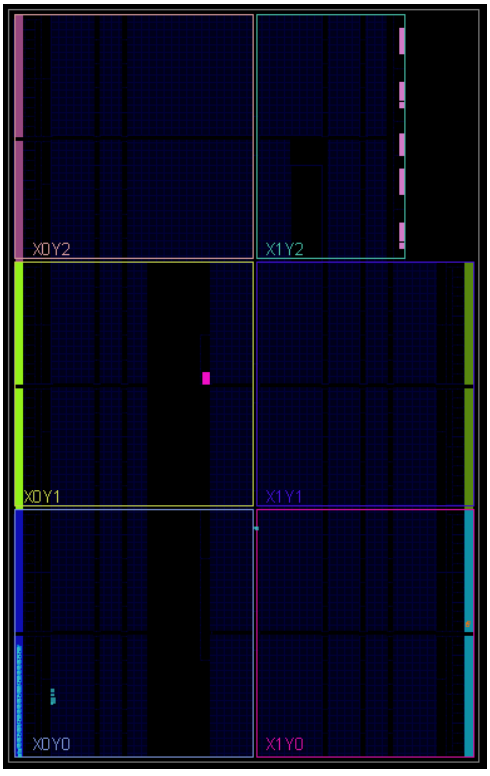

Figure 11: Q1 Device (Gray)

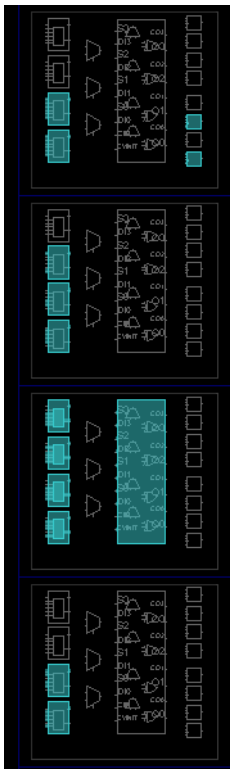

Figure 12: Q1 Zoomed Device (Gray)

Design Timing Summary

Setup		Hold		Pulse Width	
Worst Negative Slack (WNS):	7.755 ns	Worst Hold Slack (WHS):	0.420 ns	Worst Pulse Width Slack (WPWS):	4.500 ns
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	0.000 ns	Total Pulse Width Negative Slack (TPWS):	0.000 ns
Number of Failing Endpoints:	0	Number of Failing Endpoints:	0	Number of Failing Endpoints:	0
Total Number of Endpoints:	2	Total Number of Endpoints:	2	Total Number of Endpoints:	3

All user specified timing constraints are met.

Figure 13: Q1 Synthesis Timing Report (Gray)

Design Timing Summary

Setup		Hold		Pulse Width	
Worst Negative Slack (WNS):	8.181 ns	Worst Hold Slack (WHS):	0.521 ns	Worst Pulse Width Slack (WPWS):	4.500 ns
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	0.000 ns	Total Pulse Width Negative Slack (TPWS):	0.000 ns
Number of Failing Endpoints:	0	Number of Failing Endpoints:	0	Number of Failing Endpoints:	0
Total Number of Endpoints:	2	Total Number of Endpoints:	2	Total Number of Endpoints:	3

All user specified timing constraints are met.

Figure 14: Q1 Implementation Timing Report (Gray)

Name	Slice LUTs (20800)	Slice Registers (41600)	Bonded IOB (106)	BUFGCTRL (32)
N Tesla_Car	11	2	24	1

Figure 15: Q1 Synthesis Utilization Report (Gray)

Name	Slice LUTs (20800)	Slice Registers (41600)	Slice (8150)	LUT as Logic (20800)	LUT Flip Flop Pairs (20800)	Bonded IOB (106)	BUFGCTRL (32)
N Tesla_Car	11	2	4	11	2	24	1

Figure 16: Q1 Implementation Utilization Report (Gray)

1.2 Sequential Encoding:

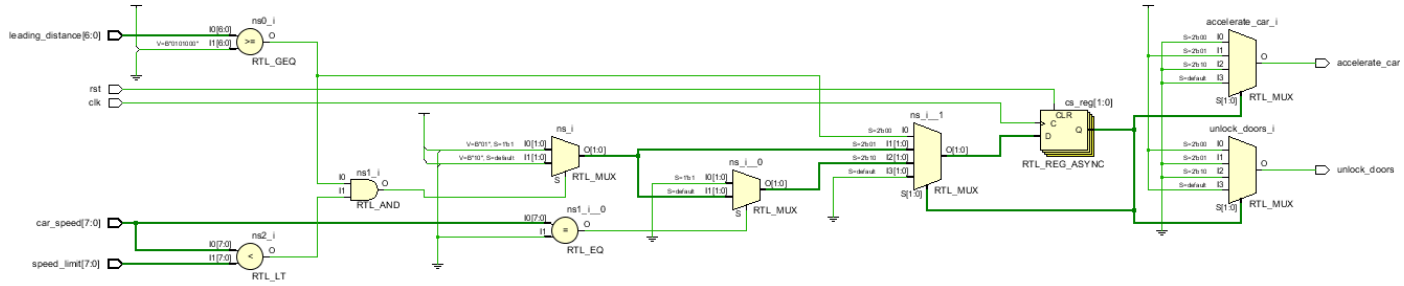

Figure 17: Q1 RTL (Sequential)

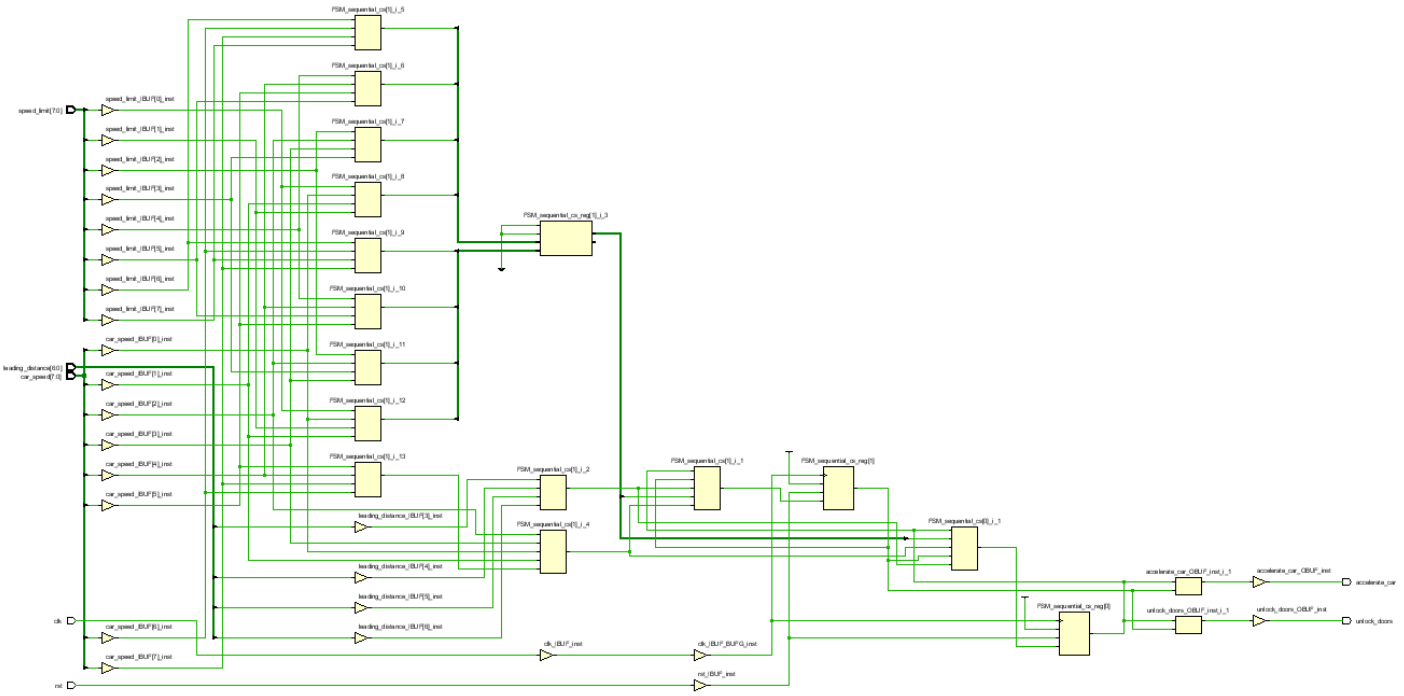

Figure 18: Q1 Synthesis (Sequential)

State	New Encoding	Previous Encoding
STOP	00	00
ACCELERATE	01	01
DECELERATE	10	10

INFO: [Synth 8-3354] encoded FSM with state register 'cs_reg' using encoding 'sequential' in module 'Tesla_Car'

Figure 19: Q1 Encoding Report (Sequential)

```
1   // Copyright 1986-2018 Xilinx, Inc. All Rights Reserved.
2   // ----------------------------------------------------------------------------
3   // Tool Version: Vivado v.2018.2 (win64) Build 2258646 Thu Jun 14 20:03:12 MDT 2018
4   // Date        : Wed Aug  6 00:57:56 2025
5   // Host        : DESKTOP-DTNSAHB running 64-bit major release  (build 9200)
6   // Command     : write_verilog {D:/College Projects/Digital_Course_Waseem/Vv/Q1/Tesla_Car_Seq.v}
7   // Design      : Tesla_Car
8   // Purpose     : This is a Verilog netlist of the current design or from a specific cell of the design. The output is a
9   //               IEEE 1364-2001 compliant Verilog HDL file that contains netlist information obtained from the input
10  //               design files.
11  // Device      : xc7a35ticpg236-1L
12  // ----------------------------------------------------------------------------
13  `timescale 1 ps / 1 ps
14
15  (* ACCELERATE = "2'b01" *) (* DECELERATE = "2'b10" *) (* MIN_DISTANCE = "7'b0101000" *)
16  (* STOP = "2'b00" *)
17  (* STRUCTURAL_NETLIST = "yes" *)
18  module Tesla_Car
19     (clk,
20      rst,
21      speed_limit,
22      car_speed,
23      leading_distance,
24      unlock_doors,
25      accelerate_car);
26    input clk;
27    input rst;
28    input [7:0]speed_limit;
29    input [7:0]car_speed;
30    input [6:0]leading_distance;
31    output unlock_doors;
32    output accelerate_car;
```

Figure 20: Q1 Netlist (Sequential)

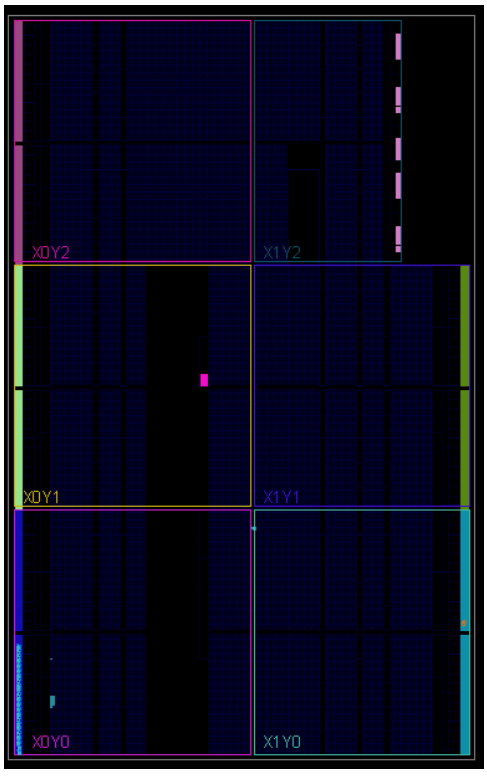

Figure 21: Q1 Device (Sequential)

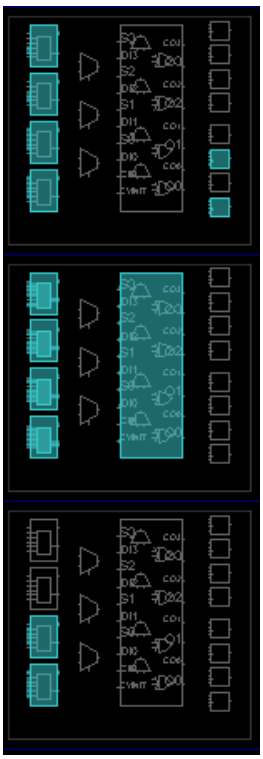

Figure 22: Q1 Zoomed Device (Sequential)

Design Timing Summary

Setup		Hold		Pulse Width	
Worst Negative Slack (WNS):	8.268 ns	Worst Hold Slack (WHS):	0.294 ns	Worst Pulse Width Slack (WPWS):	4.500 ns
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	0.000 ns	Total Pulse Width Negative Slack (TPWS):	0.000 ns
Number of Failing Endpoints:	0	Number of Failing Endpoints:	0	Number of Failing Endpoints:	0
Total Number of Endpoints:	2	Total Number of Endpoints:	2	Total Number of Endpoints:	3

All user specified timing constraints are met.

Figure 23: Q1 Synthesis Timing Report (Sequential)

Design Timing Summary

Setup		Hold		Pulse Width	
Worst Negative Slack (WNS):	8.745 ns	Worst Hold Slack (WHS):	0.248 ns	Worst Pulse Width Slack (WPWS):	4.500 ns
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	0.000 ns	Total Pulse Width Negative Slack (TPWS):	0.000 ns
Number of Failing Endpoints:	0	Number of Failing Endpoints:	0	Number of Failing Endpoints:	0
Total Number of Endpoints:	2	Total Number of Endpoints:	2	Total Number of Endpoints:	3

All user specified timing constraints are met.

Figure 24: Q1 Implementation Timing Report (Sequential)

Name 1	Slice LUTs (20800)	Slice Registers (41600)	Bonded IOB (106)	BUFGCTRL (32)
N Tesla_Car	11	2	24	1

Figure 25: Q1 Synthesis Utilization Report (Sequential)

Name 1	Slice LUTs (20800)	Slice Registers (41600)	Slice (8150)	LUT as Logic (20800)	LUT Flip Flop Pairs (20800)	Bonded IOB (106)	BUFGCTRL (32)
N Tesla_Car	11	2	4	11	2	24	1

Figure 26: Q1 Implementation Utilization Report (Sequential)

1.3 One-Hot Encoding:

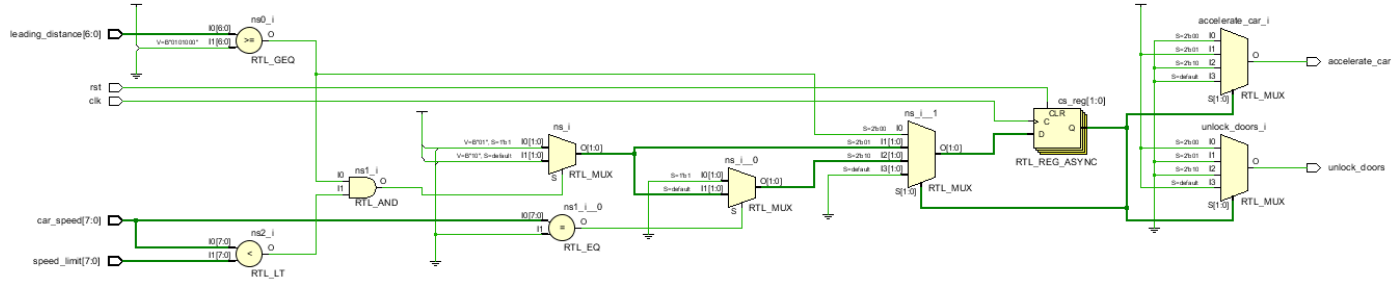

Figure 27: Q1 RTL (One-Hot)

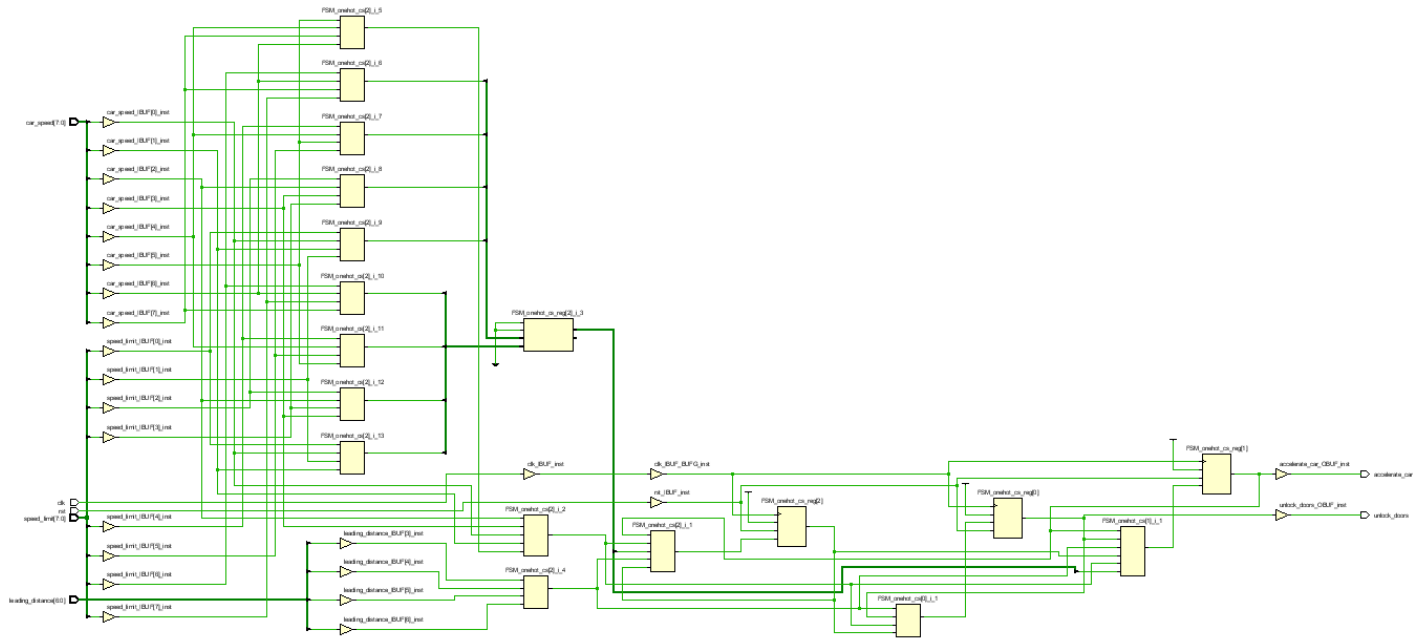

Figure 28: Q1 Synthesis (One-Hot)

State	New Encoding	Previous Encoding
STOP	001	00
ACCELERATE	010	01
DECELERATE	100	10

```
INFO: [Synth 8-3354] encoded FSM with state register 'cs_reg' using encoding 'one-hot' in module 'Tesla_Car'
```

Figure 29: Q1 Encoding Report (One-Hot)

```
// Copyright 1986-2018 Xilinx, Inc. All Rights Reserved.
// ----------------------------------------------------------------------------
// Tool Version: Vivado v.2018.2 (win64) Build 2258646 Thu Jun 14 20:03:12 MDT 2018
// Date        : Wed Aug  6 00:35:30 2025
// Host        : DESKTOP-DTNSAHB running 64-bit major release  (build 9200)
// Command     : write_verilog {D:/College Projects/Digital_Course_Waseem/Vv/Q1/Tesla_Car_One_Hot.v}
// Design      : Tesla_Car
// Purpose     : This is a Verilog netlist of the current design or from a specific cell of the design. The output is an
//               IEEE 1364-2001 compliant Verilog HDL file that contains netlist information obtained from the input
//               design files.
// Device      : xc7a35ticpg236-1L
// ----------------------------------------------------------------------------
`timescale 1 ps / 1 ps

(* ACCELERATE = "2'b01" *) (* DECELERATE = "2'b10" *) (* MIN_DISTANCE = "7'b0101000" *)
(* STOP = "2'b00" *)
(* STRUCTURAL_NETLIST = "yes" *)
module Tesla_Car
   (clk,
    rst,
    speed_limit,
    car_speed,
    leading_distance,
    unlock_doors,
    accelerate_car);
  input clk;
  input rst;
  input [7:0]speed_limit;
  input [7:0]car_speed;
  input [6:0]leading_distance;
  output unlock_doors;
  output accelerate_car;
```

Figure 30: Q1 Netlist (One-Hot)

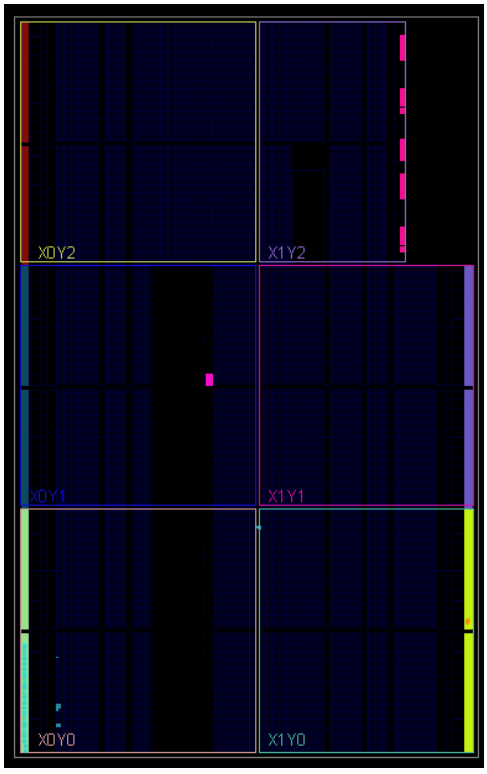

Figure 31: Q1 Device (One-Hot)

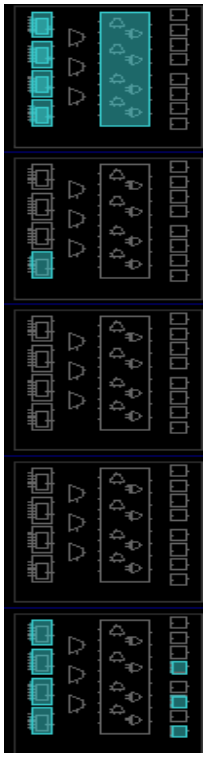

Figure 32: Q1 Zoomed Device (One-Hot)

Design Timing Summary

Setup		Hold		Pulse Width	
Worst Negative Slack (WNS):	8.115 ns	Worst Hold Slack (WHS):	0.291 ns	Worst Pulse Width Slack (WPWS):	4.500 ns
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	0.000 ns	Total Pulse Width Negative Slack (TPWS):	0.000 ns
Number of Failing Endpoints:	0	Number of Failing Endpoints:	0	Number of Failing Endpoints:	0
Total Number of Endpoints:	3	Total Number of Endpoints:	3	Total Number of Endpoints:	4

All user specified timing constraints are met.

Figure 33: Q1 Synthesis Timing Report (One-Hot)

Design Timing Summary

Setup		Hold		Pulse Width	
Worst Negative Slack (WNS):	8.697 ns	Worst Hold Slack (WHS):	0.281 ns	Worst Pulse Width Slack (WPWS):	4.500 ns
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	0.000 ns	Total Pulse Width Negative Slack (TPWS):	0.000 ns
Number of Failing Endpoints:	0	Number of Failing Endpoints:	0	Number of Failing Endpoints:	0
Total Number of Endpoints:	3	Total Number of Endpoints:	3	Total Number of Endpoints:	4

All user specified timing constraints are met.

Figure 34: Q1 Implementation Timing Report (One-Hot)

Name 1	Slice LUTs (20800)	Slice Registers (41600)	Bonded IOB (106)	BUFGCTRL (32)
N Tesla_Car	10	3	24	1

Figure 35: Q1 Synthesis Utilization Report (One-Hot)

Name 1	Slice LUTs (20800)	Slice Registers (41600)	Slice (8150)	LUT as Logic (20800)	LUT Flip Flop Pairs (20800)	Bonded IOB (106)	BUFGCTRL (32)
N Tesla_Car	10	3	4	10	3	24	1

Figure 36: Q1 Implementation Utilization Report (One-Hot)

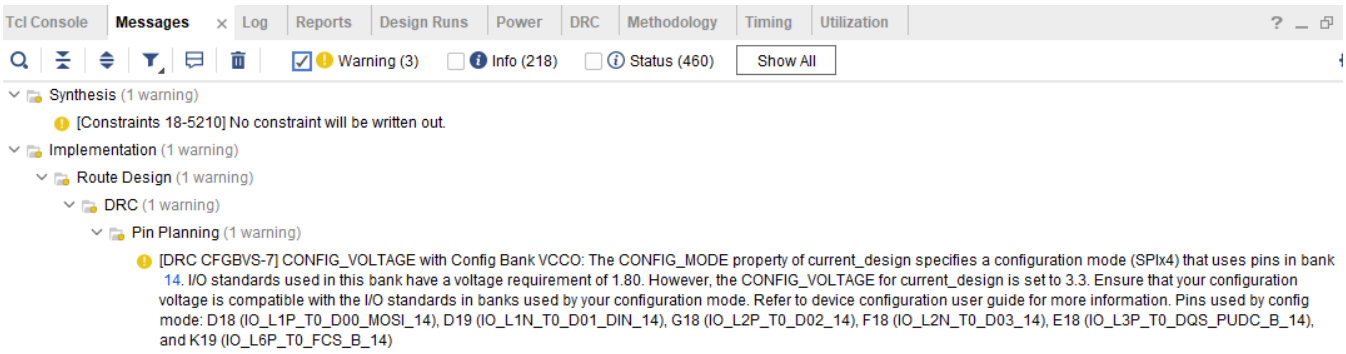

Figure 37: Q1 Messages

Sequential and Gray encodings use two flip flops, while one-hot encoding uses three flip flops. Sequential encoding has the highest setup slack. So, Sequential is the best encoding in this case, as it has the highest slack which means higher frequency clock, with the least number of registers used.

2. Gray Counter FSM:

```
module Gray_Counter_FSM (input clk,rst, output reg [1:0] y);
localparam A=2'b00,  B=2'b01,  C=2'b10,  D=2'b11;
reg [1:0] cs,ns;

/*--State Memory--*/
always @(posedge clk or posedge rst) begin
    if (rst)      cs <= A;
    else          cs <= ns;
end

/*--Next State and Output Logic--*/
always @(*) begin
    case (cs)
        A:       {ns, y} = {B, 2'b00};
        B:       {ns, y} = {C, 2'b01};
        C:       {ns, y} = {D, 2'b11};
        D:       {ns, y} = {A, 2'b10};
        default: {ns, y} = {A, 2'b00};
    endcase
end
endmodule
```

Figure 38: Q2 Code

```
module Gray_Counter (input clk,rst, output [1:0] gray_out);
reg [1:0] binary_out;

always @(posedge clk or posedge rst) begin
    if (rst)   binary_out <= 0;
    else       binary_out <= binary_out + 1;
end

assign gray_out = {binary_out[1] , ^binary_out};

endmodule
```

Figure 39: Q2 Golden Model Code

```
module Gray_Counter_FSM_tb ();
reg clk,rst;
wire [1:0] y,y_Golden;

//module Gray_Counter_FSM (input clk,rst, output reg [1:0] y);
Gray_Counter_FSM DUT(clk,rst,y);
//module Gray_Counter (input clk,rst, output reg [1:0] gray_out);
Gray_Counter DUT_GOLDEN(clk,rst,y_Golden);

initial begin
    clk=0;
    forever #5 clk=~clk;
end

initial begin
    rst=1;
    #2;
    if (DUT.cs != DUT.A || y !=y_Golden) begin
        $display("Error in Reset!!");
        $stop;
    end
    rst=0;
    @(negedge clk);
    if (DUT.cs != DUT.B || y !=y_Golden) begin
        $display("Error!!");
        $stop;
    end
    @(negedge clk);
    if (DUT.cs != DUT.C || y !=y_Golden) begin
        $display("Error!!");
        $stop;
    end
    @(negedge clk);
    if (DUT.cs != DUT.D || y !=y_Golden) begin
        $display("Error!!");
        $stop;
    end
    @(negedge clk);
    if (DUT.cs != DUT.A || y !=y_Golden) begin
        $display("Error!!");
        $stop;
    end
    repeat (5) begin
        @(negedge clk);
        if (y !=y_Golden) begin
            $display("Error!!");
            $stop;
        end
    end
    $stop;
end
endmodule
```

Figure 40: Q2 Testbench

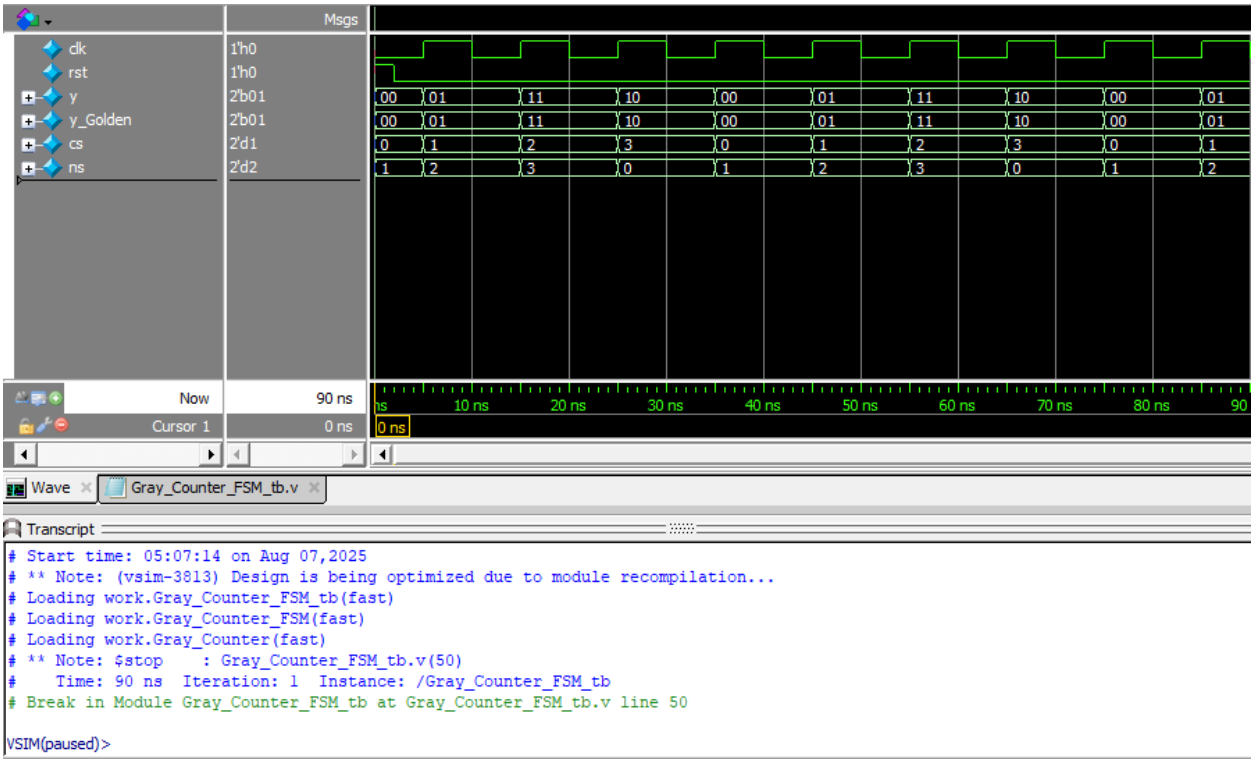

Figure 41: Q2 Wave

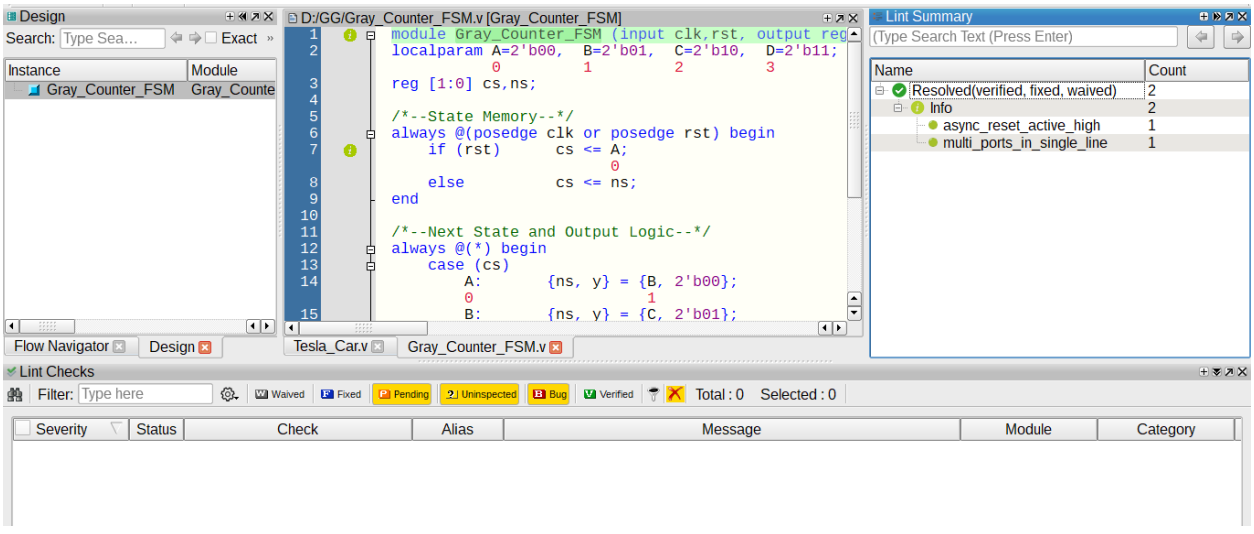

Figure 42: Q2 Linting

```
vlib work
vlog Gray_Counter_FSM.v Gray_Counter_FSM_tb.v
vsim -voptargs=+acc Gray_Counter_FSM_tb
add wave *
add wave -position insertpoint  \
sim:/Gray_Counter_FSM_tb/DUT/cs \
sim:/Gray_Counter_FSM_tb/DUT/ns
run -all
#quit -sim
```

Figure 43: Q2 DO File

```
## This file is a general .xdc for the Basys3 rev B board
## To use it in a project:
## - uncomment the lines corresponding to used pins
## - rename the used ports (in each line, after get_ports) according to the top level signal names in the project

## Clock signal
set_property -dict { PACKAGE_PIN W5   IOSTANDARD LVCMOS33 } [get_ports clk]
create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports clk]

## LEDs
set_property -dict { PACKAGE_PIN U16   IOSTANDARD LVCMOS33 } [get_ports {y[0]}]
set_property -dict { PACKAGE_PIN E19   IOSTANDARD LVCMOS33 } [get_ports {y[1]}]

##Buttons
set_property -dict { PACKAGE_PIN U18   IOSTANDARD LVCMOS33 } [get_ports rst]

## Configuration options, can be used for all designs
set_property CONFIG_VOLTAGE 3.3 [current_design]
set_property CFGBVS VCCO [current_design]

## SPI configuration mode options for QSPI boot, can be used for all designs
set_property BITSTREAM.GENERAL.COMPRESS TRUE [current_design]
set_property BITSTREAM.CONFIG.CONFIGRATE 33 [current_design]
set_property CONFIG_MODE SPIx4 [current_design]
```

Figure 44: Q2 Constraint File

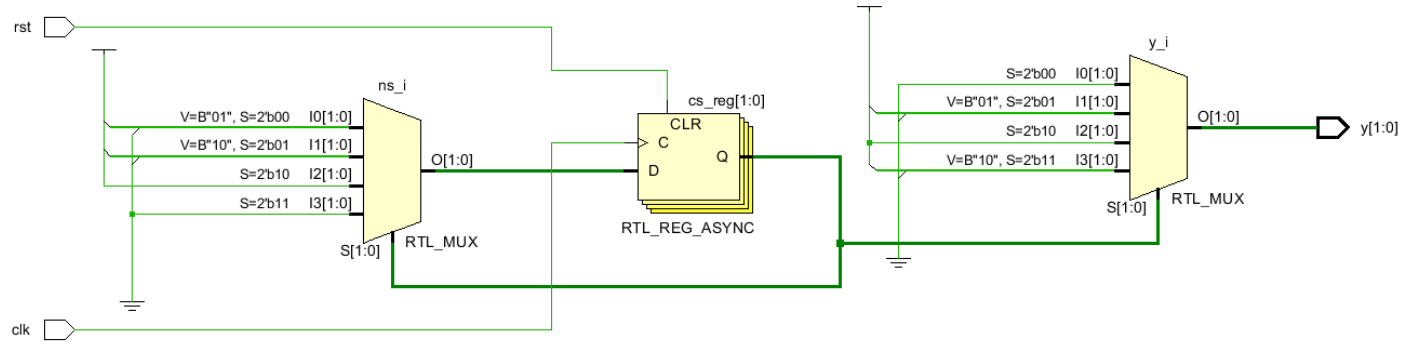

Figure 45: Q2 RTL

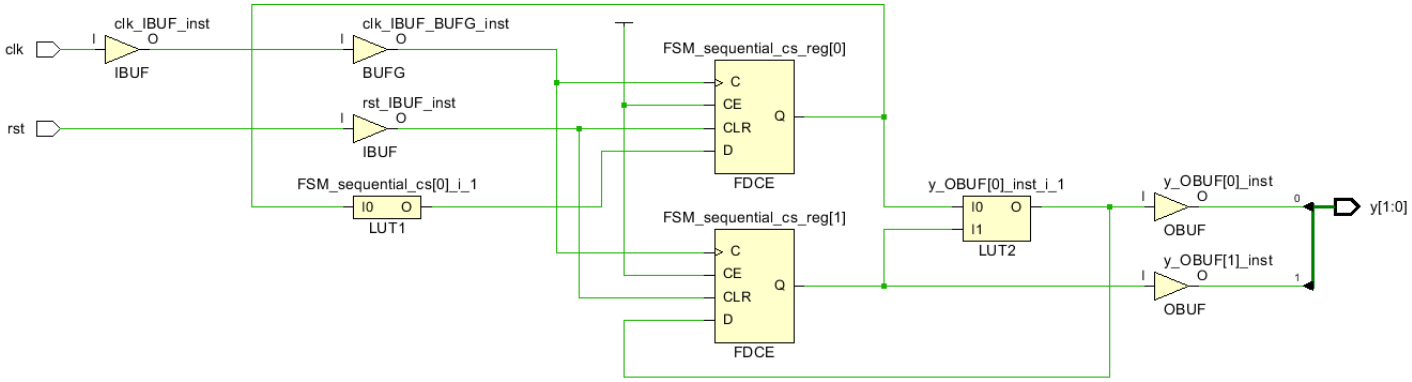

Figure 46: Q2 Synthesis

```
// Copyright 1986-2018 Xilinx, Inc. All Rights Reserved.
// ----------------------------------------------------------------------------
// Tool Version: Vivado v.2018.2 (win64) Build 2258646 Thu Jun 14 20:03:12 MDT 2018
// Date        : Wed Aug  6 03:22:40 2025
// Host        : DESKTOP-DTNSAHB running 64-bit major release  (build 9200)
// Command     : write_verilog {D:/College Projects/Digital_Course_Waseem/Vv/Q2/Gray_Counter_FSM_Netlist.v}
// Design      : Gray_Counter_FSM
// Purpose     : This is a Verilog netlist of the current design or from a specific cell of the design. The output is an
//               IEEE 1364-2001 compliant Verilog HDL file that contains netlist information obtained from the input
//               design files.
// Device      : xc7a35ticpg236-1L
// ----------------------------------------------------------------------------
`timescale 1 ps / 1 ps

(* A = "2'b00" *) (* B = "2'b01" *) (* C = "2'b10" *)
(* D = "2'b11" *)
(* STRUCTURAL_NETLIST = "yes" *)
module Gray_Counter_FSM
   (clk,
    rst,
    y);
  input clk;
  input rst;
  output [1:0]y;

  wire \<const1> ;
  wire clk;
  wire clk_IBUF;
  wire clk_IBUF_BUFG;
  (* RTL_KEEP = "yes" *) wire [0:0]cs;
  wire [0:0]ns;
  wire rst;
```

Figure 47: Q2 Netlist

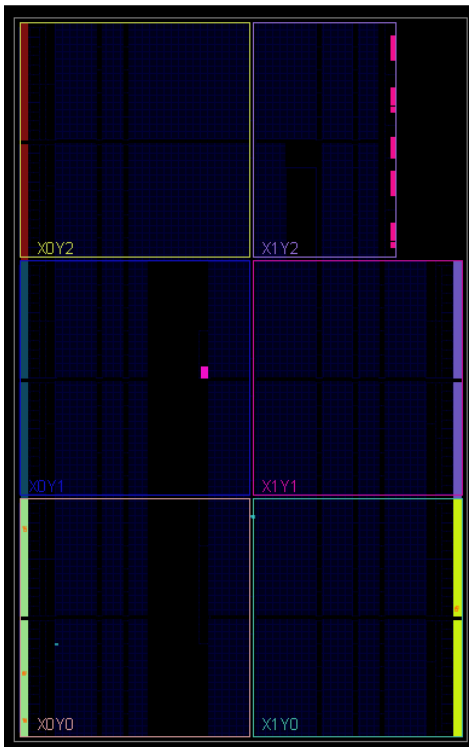

Figure 48: Q2 Device

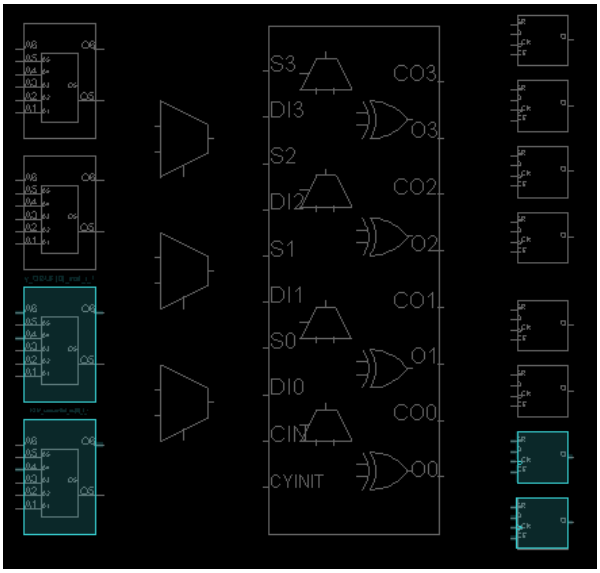

Figure 49: Q2 Zoomed Device

Design Timing Summary

Setup		Hold		Pulse Width	
Worst Negative Slack (WNS):	8.617 ns	Worst Hold Slack (WHS):	0.139 ns	Worst Pulse Width Slack (WPWS):	4.500 ns
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	0.000 ns	Total Pulse Width Negative Slack (TPWS):	0.000 ns
Number of Failing Endpoints:	0	Number of Failing Endpoints:	0	Number of Failing Endpoints:	0
Total Number of Endpoints:	2	Total Number of Endpoints:	2	Total Number of Endpoints:	3

All user specified timing constraints are met.

Figure 50: Q2 Synthesis Timing Report

Design Timing Summary

Setup		Hold		Pulse Width	
Worst Negative Slack (WNS):	8.019 ns	Worst Hold Slack (WHS):	0.367 ns	Worst Pulse Width Slack (WPWS):	4.500 ns
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	0.000 ns	Total Pulse Width Negative Slack (TPWS):	0.000 ns
Number of Failing Endpoints:	0	Number of Failing Endpoints:	0	Number of Failing Endpoints:	0
Total Number of Endpoints:	2	Total Number of Endpoints:	2	Total Number of Endpoints:	3

All user specified timing constraints are met.

Figure 51: Q2 Implementation Timing Report

Name	Slice LUTs (20800)	Slice Registers (41600)	Bonded IOB (106)	BUFGCTRL (32)
Gray_Counter_FSM	2	2	4	1

Figure 52: Q2 Synthesis Utilization Report

Name	Slice LUTs (20800)	Slice Registers (41600)	Slice (8150)	LUT as Logic (20800)	LUT Flip Flop Pairs (20800)	Bonded IOB (106)	BUFGCTRL (32)
Gray_Counter_FSM	2	2	1	2	1	4	1

Figure 53: Q2 Implementation Utilization Report

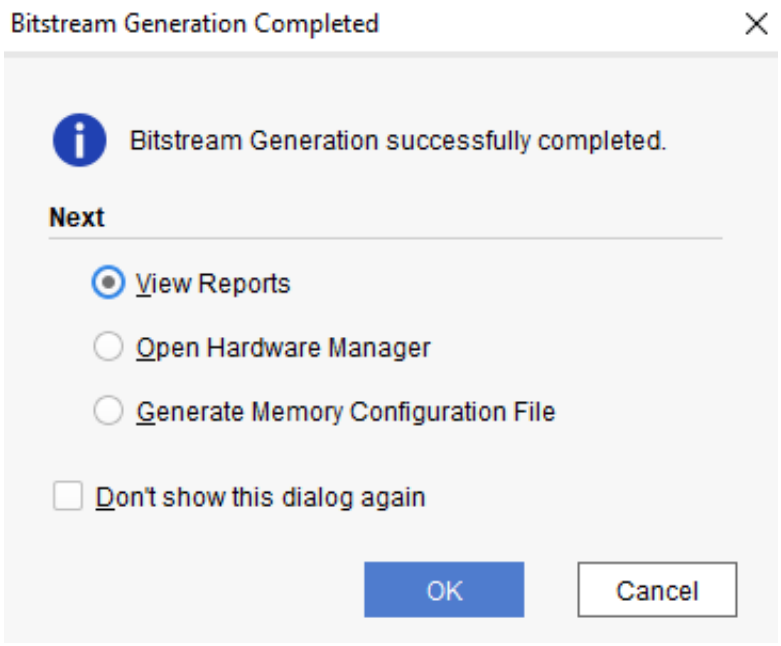

Figure 54: Q2 Bitstream

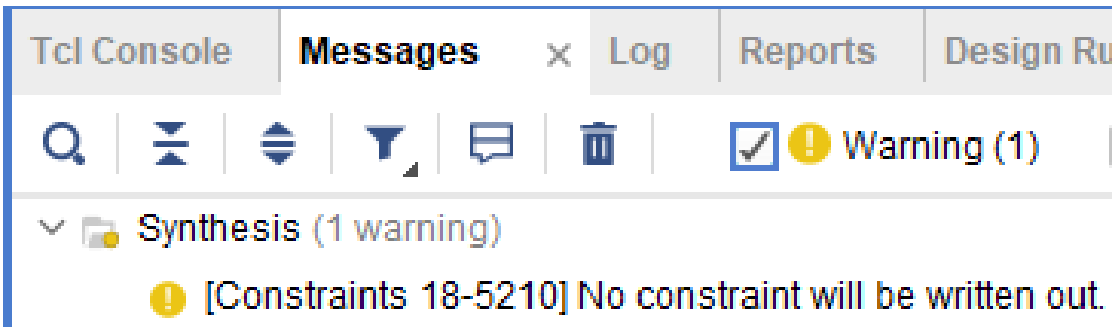

Figure 55: Q2 Messages

3. Sequence Detector (111):

```
1  module Sequence_Detector_111 (input clk,rst,Din, output ERR);
2  localparam START = 3'b000,  D0_IS_1 = 3'b001,   D1_IS_1 = 3'b010,  D0_NOT_1 = 3'b011,  D1_NOT_1 = 3'b100;
3  (*fsm_encoding = "gray"*)
4  reg [2:0] cs,ns;
5
6  /*--State Memory--*/
7  always @(posedge clk or posedge rst) begin
8      if (rst)   cs <= START;
9      else       cs <= ns;
10 end
11
12 /*--Next State Logic--*/
13 always @(*) begin
14     case (cs)
15         START: begin
16             if (Din)   ns = D0_IS_1;
17             else       ns = D0_NOT_1;
18         end
19         D0_IS_1: begin
20             if (Din)   ns = D1_IS_1;
21             else       ns = D1_NOT_1;
22         end
23         D1_IS_1:       ns = START;
24         D0_NOT_1:      ns = D1_NOT_1;
25         D1_NOT_1:      ns = START;
26         default:       ns = START;
27     endcase
28 end
29
30 assign ERR = (cs == D1_IS_1 && Din)? 1: 0;
31
32 endmodule
```

Figure 56: Q3 Code

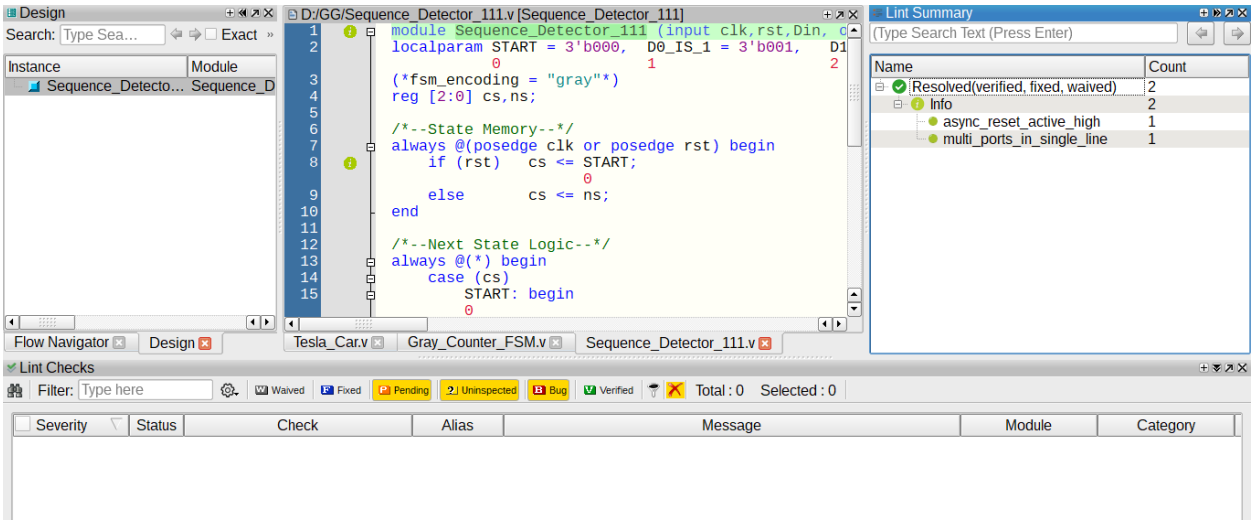

Figure 57: Q3 Linting

```
module Sequence_Detector_111_tb ();
reg clk,rst,Din;
wire ERR;

//module Sequence_Detector_111 (input clk,rst,Din, output reg ERR);
Sequence_Detector_111 DUT(clk,rst,Din,ERR);

initial begin
    clk=0;
    forever #5 clk=~clk;
end

initial begin
    rst=1; Din=$random;
    @(negedge clk);
    if (DUT.cs != DUT.START || ERR) begin
        $display("Eroor in Reset!!");
        $stop;
    end
    rst=0;
    repeat (30) begin
        Din=$random;
        @(negedge clk);
    end
    $stop;
end
endmodule
```

Figure 58: Q3 Testbench

```
vlib work
vlog Sequence_Detector_111.v Sequence_Detector_111_tb.v
vsim -voptargs=+acc Sequence_Detector_111_tb
add wave *
add wave -position insertpoint  \
sim:/Sequence_Detector_111_tb/DUT/cs \
sim:/Sequence_Detector_111_tb/DUT/ns
run -all
#quit -sim
```

Figure 59: Q3 DO File

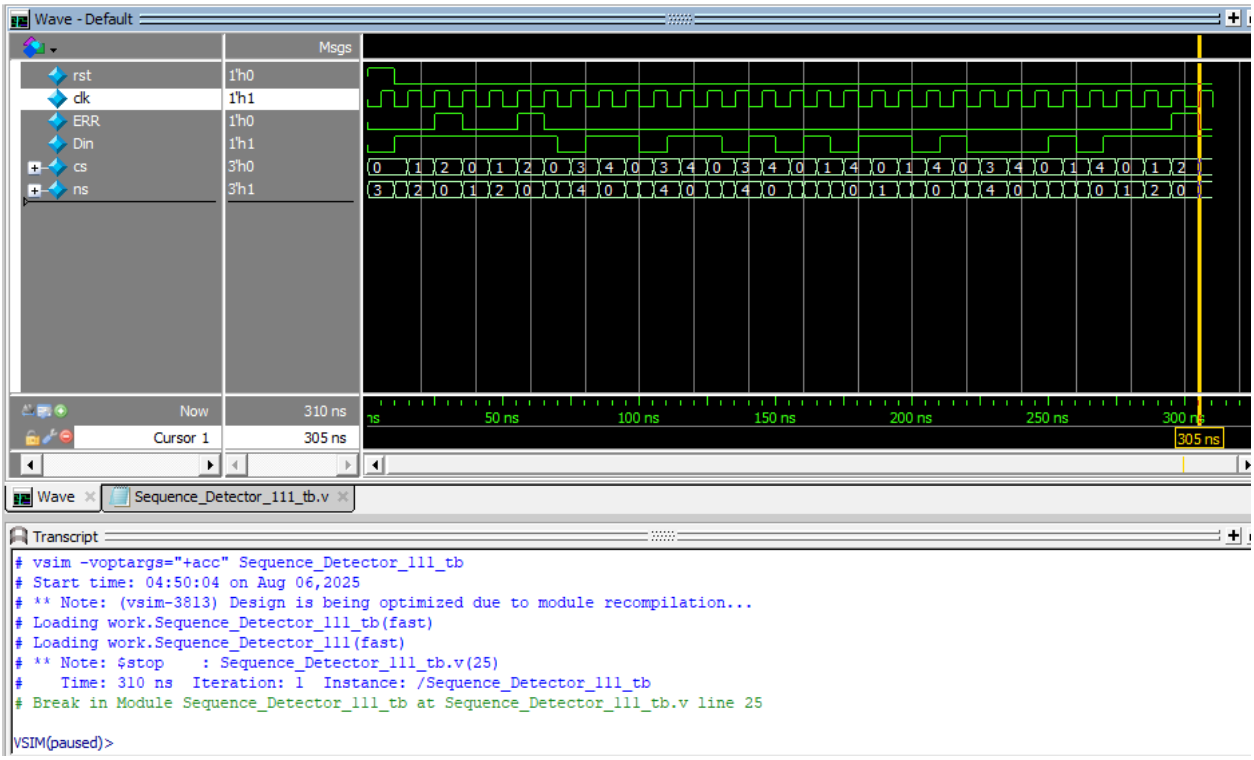

Figure 60: Q3 Wave

```
## This file is a general .xdc for the Basys3 rev B board
## To use it in a project:
## - uncomment the lines corresponding to used pins
## - rename the used ports (in each line, after get_ports) according to the top level signal names in the project

## Clock signal
set_property -dict { PACKAGE_PIN W5   IOSTANDARD LVCMOS33 } [get_ports clk]
create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports clk]


## Switches
set_property -dict { PACKAGE_PIN V17   IOSTANDARD LVCMOS33 } [get_ports {Din}]

## LEDs
set_property -dict { PACKAGE_PIN U16   IOSTANDARD LVCMOS33 } [get_ports {ERR}]

##Buttons
set_property -dict { PACKAGE_PIN U18   IOSTANDARD LVCMOS33 } [get_ports rst]


## Configuration options, can be used for all designs
set_property CONFIG_VOLTAGE 3.3 [current_design]
set_property CFGBVS VCCO [current_design]

## SPI configuration mode options for QSPI boot, can be used for all designs
set_property BITSTREAM.GENERAL.COMPRESS TRUE [current_design]
set_property BITSTREAM.CONFIG.CONFIGRATE 33 [current_design]
set_property CONFIG_MODE SPIx4 [current_design]
```

Figure 61: Q3 Constraint File

3.1 Gray Encoding:

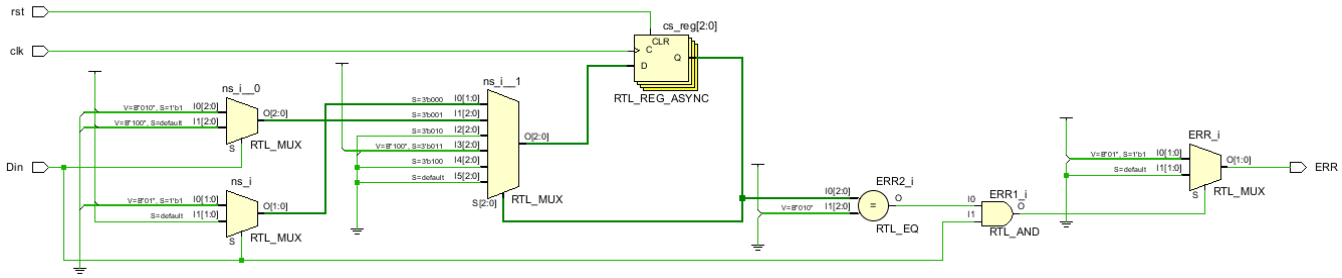

Figure 62: Q3 RTL (Gray)

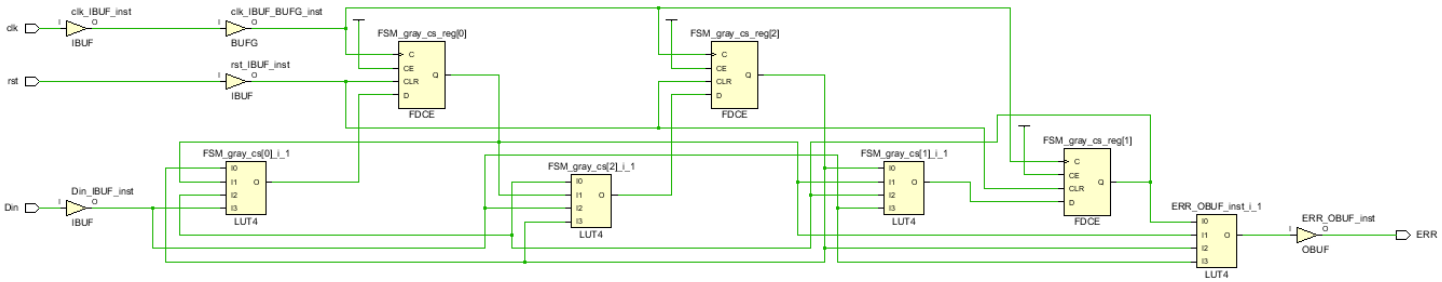

Figure 63: Q3 Synthesis (Gray)

```
-----------------------------------------------------------------------------------------------------
                   State |                     New Encoding |                Previous Encoding
-----------------------------------------------------------------------------------------------------
                   START |                              000 |                             000
                  D0_IS_1 |                              001 |                             001
                  D1_IS_1 |                              011 |                             010
                 D0_NOT_1 |                              010 |                             011
                 D1_NOT_1 |                              111 |                             100
-----------------------------------------------------------------------------------------------------
INFO: [Synth 8-3354] encoded FSM with state register 'cs_reg' using encoding 'gray' in module 'Sequence_Detector_111'
```

Figure 64: Q3 Encoding Report (Gray)

```
1  // Copyright 1986-2018 Xilinx, Inc. All Rights Reserved.
2  // ----------------------------------------------------------------------------
3  // Tool Version: Vivado v.2018.2 (win64) Build 2258646 Thu Jun 14 20:03:12 MDT 2018
4  // Date        : Wed Aug  6 05:22:32 2025
5  // Host        : DESKTOP-DTNSAHB running 64-bit major release  (build 9200)
6  // Command     : write_verilog {D:/College Projects/Digital_Course_Waseem/Vv/Q3/Sequence_Detector_111_Gray.v}
7  // Design      : Sequence_Detector_111
8  // Purpose     : This is a Verilog netlist of the current design or from a specific cell of the design. The output is an
9  //               IEEE 1364-2001 compliant Verilog HDL file that contains netlist information obtained from the input
10 //               design files.
11 // Device      : xc7a35ticpg236-1L
12 // ----------------------------------------------------------------------------
13 `timescale 1 ps / 1 ps
14
15 (* D0_IS_1 = "3'b001" *) (* D0_NOT_1 = "3'b011" *) (* D1_IS_1 = "3'b010" *)
16 (* D1_NOT_1 = "3'b100" *) (* START = "3'b000" *)
17 (* STRUCTURAL_NETLIST = "yes" *)
18 module Sequence_Detector_111
19    (clk,
20     rst,
21     Din,
22     ERR);
23   input clk;
24   input rst;
25   input Din;
26   output ERR;
27
28   wire \<const1> ;
29   wire Din;
30   wire Din_IBUF;
31   wire ERR;
32   wire ERR_OBUF;
```

Figure 65: Q3 Netlist (Gray)

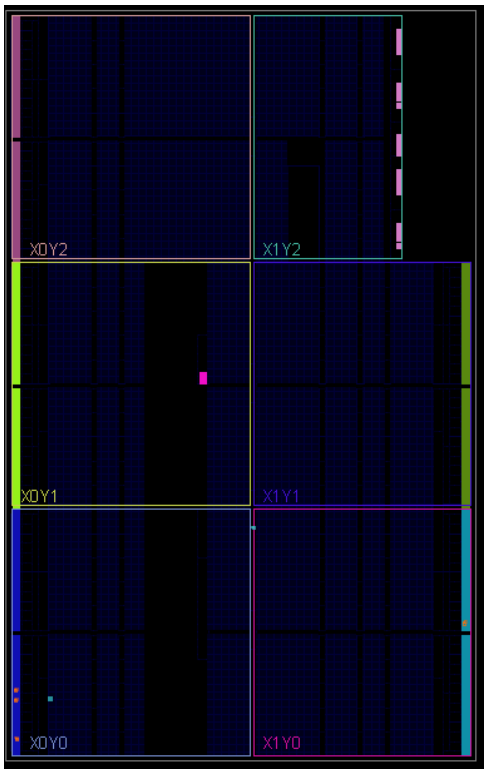

Figure 66: Q3 Device (Gray)

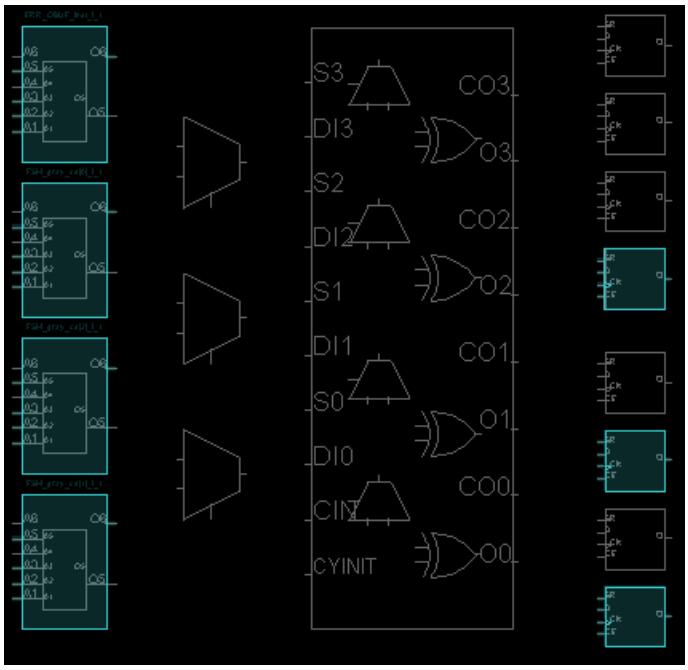

Figure 67: Q3 Zoomed Device (Gray)

Design Timing Summary

Setup		Hold		Pulse Width	
Worst Negative Slack (WNS):	8.333 ns	Worst Hold Slack (WHS):	0.144 ns	Worst Pulse Width Slack (WPWS):	4.500 ns
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	0.000 ns	Total Pulse Width Negative Slack (TPWS):	0.000 ns
Number of Failing Endpoints:	0	Number of Failing Endpoints:	0	Number of Failing Endpoints:	0
Total Number of Endpoints:	3	Total Number of Endpoints:	3	Total Number of Endpoints:	4

All user specified timing constraints are met.

Figure 68: Q3 Synthesis Timing Report (Gray)

Design Timing Summary

Setup		Hold		Pulse Width	
Worst Negative Slack (WNS):	8.513 ns	Worst Hold Slack (WHS):	0.279 ns	Worst Pulse Width Slack (WPWS):	4.500 ns
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	0.000 ns	Total Pulse Width Negative Slack (TPWS):	0.000 ns
Number of Failing Endpoints:	0	Number of Failing Endpoints:	0	Number of Failing Endpoints:	0
Total Number of Endpoints:	3	Total Number of Endpoints:	3	Total Number of Endpoints:	4

All user specified timing constraints are met.

Figure 69: Q3 Implementation Timing Report (Gray)

Name 1	Slice LUTs (20800)	Slice Registers (41600)	Bonded IOB (106)	BUFGCTRL (32)
N Sequence_Detector_111	4	3	4	1

Figure 70: Q3 Synthesis Utilization Report (Gray)

Name 1	Slice LUTs (20800)	Slice Registers (41600)	Slice (8150)	LUT as Logic (20800)	LUT Flip Flop Pairs (20800)	Bonded IOB (106)	BUFGCTRL (32)
N Sequence_Detector_111	4	3	1	4	3	4	1

Figure 71: Q3 Implementation Utilization Report (Gray)

3.2 Sequential Encoding:

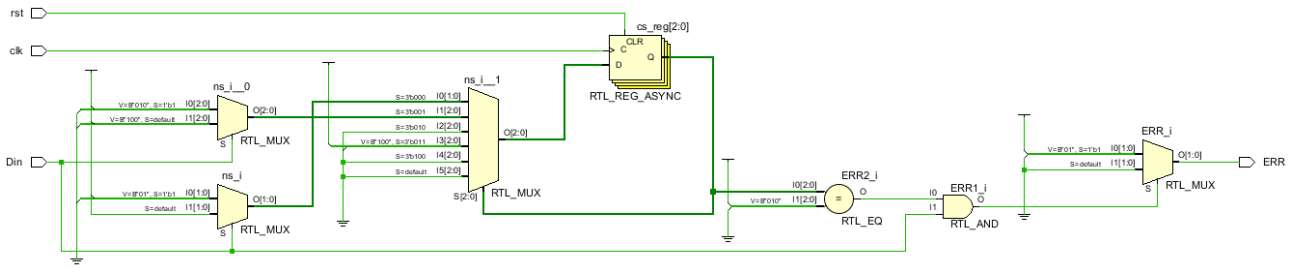

Figure 72: Q3 RTL (Sequential)

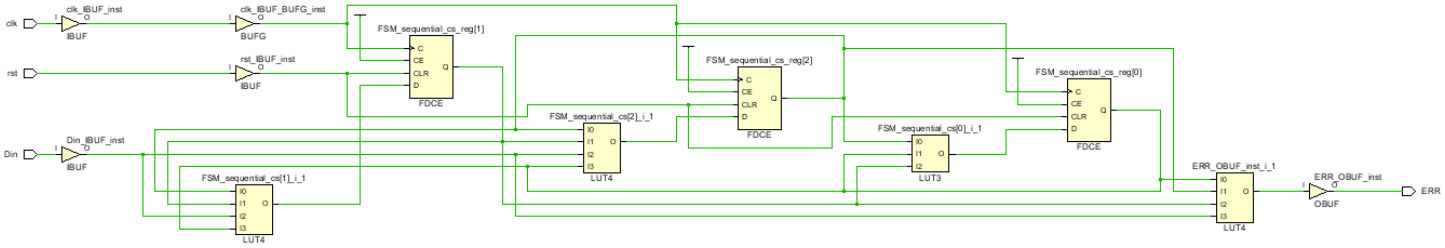

Figure 73: Q3 Synthesis (Sequential)

```
-------------------------------------------------------------------------------------------------
                State |                     New Encoding |                Previous Encoding
-------------------------------------------------------------------------------------------------
                START |                              000 |                               000
               D0_IS_1 |                              001 |                               001
               D1_IS_1 |                              010 |                               010
              D0_NOT_1 |                              011 |                               011
              D1_NOT_1 |                              100 |                               100
-------------------------------------------------------------------------------------------------
INFO: [Synth 8-3354] encoded FSM with state register 'cs_reg' using encoding 'sequential' in module 'Sequence_Detector_111'
```

Figure 74: Q3 Encoding Report (Sequential)

```
1  // Copyright 1986-2018 Xilinx, Inc. All Rights Reserved.
2  // ----------------------------------------------------------------------------
3  // Tool Version: Vivado v.2018.2 (win64) Build 2258646 Thu Jun 14 20:03:12 MDT 2018
4  // Date        : Wed Aug  6 06:07:30 2025
5  // Host        : DESKTOP-DTNSAHB running 64-bit major release  (build 9200)
6  // Command     : write_verilog {D:/College Projects/Digital_Course_Waseem/Vv/Sequence_Detector_111_Seq.v}
7  // Design      : Sequence_Detector_111
8  // Purpose     : This is a Verilog netlist of the current design or from a specific cell of the design. The output is an
9  //               IEEE 1364-2001 compliant Verilog HDL file that contains netlist information obtained from the input
10 //               design files.
11 // Device      : xc7a35ticpg236-1L
12 // ----------------------------------------------------------------------------
13 `timescale 1 ps / 1 ps
14
15 (* D0_IS_1 = "3'b001" *) (* D0_NOT_1 = "3'b011" *) (* D1_IS_1 = "3'b010" *)
16 (* D1_NOT_1 = "3'b100" *) (* START = "3'b000" *)
17 (* STRUCTURAL_NETLIST = "yes" *)
18 module Sequence_Detector_111
19    (clk,
20     rst,
21     Din,
22     ERR);
23   input clk;
24   input rst;
25   input Din;
26   output ERR;
27
28   wire \<const1> ;
29   wire Din;
30   wire Din_IBUF;
31   wire ERR;
32   wire ERR_OBUF;
```

Figure 75: Q3 Netlist (Sequential)

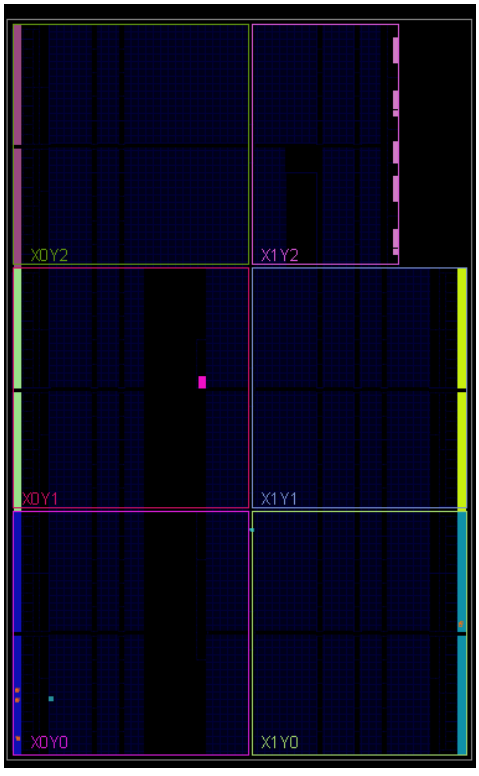

Figure 76: Q3 Device (Sequential)

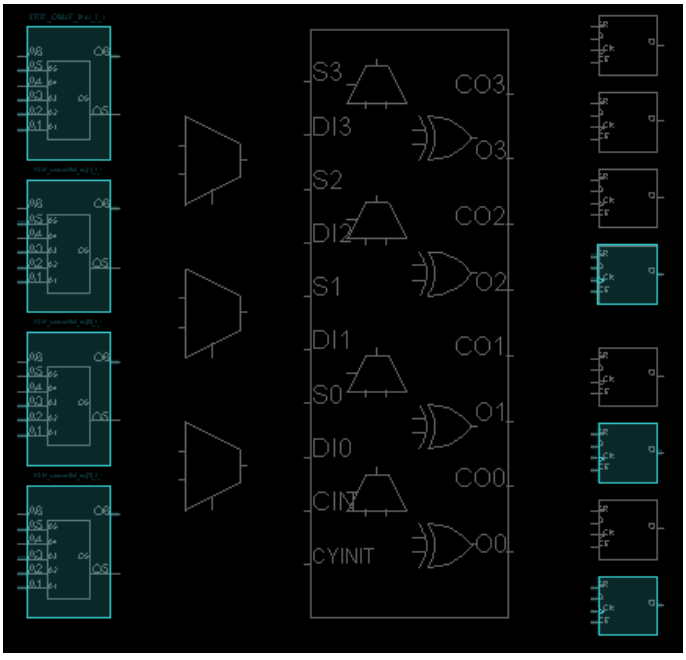

Figure 77: Q3 Zoomed Device (Sequential)

Design Timing Summary

Setup		Hold		Pulse Width	
Worst Negative Slack (WNS):	8.333 ns	Worst Hold Slack (WHS):	0.144 ns	Worst Pulse Width Slack (WPWS):	4.500 ns
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	0.000 ns	Total Pulse Width Negative Slack (TPWS):	0.000 ns
Number of Failing Endpoints:	0	Number of Failing Endpoints:	0	Number of Failing Endpoints:	0
Total Number of Endpoints:	3	Total Number of Endpoints:	3	Total Number of Endpoints:	4

All user specified timing constraints are met.

Figure 78: Q3 Synthesis Timing Report (Sequential)

Design Timing Summary

Setup		Hold		Pulse Width	
Worst Negative Slack (WNS):	8.513 ns	Worst Hold Slack (WHS):	0.279 ns	Worst Pulse Width Slack (WPWS):	4.500 ns
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	0.000 ns	Total Pulse Width Negative Slack (TPWS):	0.000 ns
Number of Failing Endpoints:	0	Number of Failing Endpoints:	0	Number of Failing Endpoints:	0
Total Number of Endpoints:	3	Total Number of Endpoints:	3	Total Number of Endpoints:	4

All user specified timing constraints are met.

Figure 79: Q3 Implementation Timing Report (Sequential)

Name 1	Slice LUTs (20800)	Slice Registers (41600)	Bonded IOB (106)	BUFGCTRL (32)
N Sequence_Detector_111	4	3	4	1

Figure 80: Q3 Synthesis Utilization Report (Sequential)

Name 1	Slice LUTs (20800)	Slice Registers (41600)	Slice (8150)	LUT as Logic (20800)	LUT Flip Flop Pairs (20800)	Bonded IOB (106)	BUFGCTRL (32)
N Sequence_Detector_111	4	3	1	4	3	4	1

Figure 81: Q3 Implementation Utilization Report (Sequential)

3.3 One-Hot Encoding:

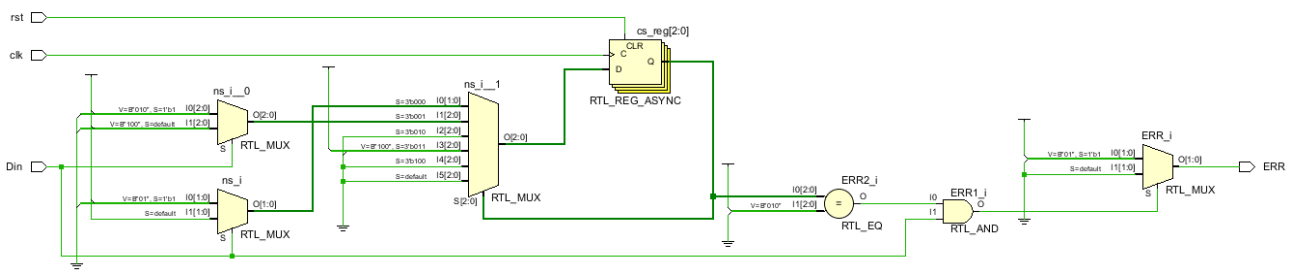

Figure 82: Q3 RTL (One-Hot)

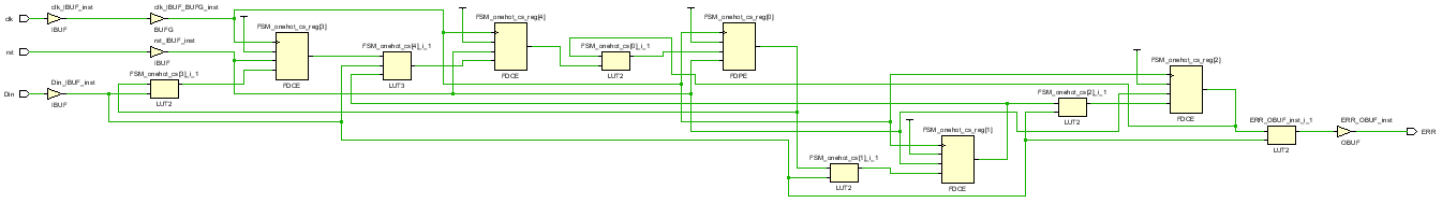

Figure 83: Q3 Synthesis (One-Hot)

State	New Encoding	Previous Encoding
START	00001	000
D0_IS_1	00010	001
D1_IS_1	00100	010
D0_NOT_1	01000	011
D1_NOT_1	10000	100

INFO: [Synth 8-3354] encoded FSM with state register 'cs_reg' using encoding 'one-hot' in module 'Sequence_Detector_111'

Figure 84: Q3 Encoding Report (One-Hot)

```
// Copyright 1986-2018 Xilinx, Inc. All Rights Reserved.
// ----------------------------------------------------------------------------
// Tool Version: Vivado v.2018.2 (win64) Build 2258646 Thu Jun 14 20:03:12 MDT 2018
// Date        : Wed Aug  6 05:43:27 2025
// Host        : DESKTOP-DTNSAHB running 64-bit major release  (build 9200)
// Command     : write_verilog {D:/College Projects/Digital_Course_Waseem/Vv/Q3/Sequence_Detector_111_One_Hot.v}
// Design      : Sequence_Detector_111
// Purpose     : This is a Verilog netlist of the current design or from a specific cell of the design. The output is an
//               IEEE 1364-2001 compliant Verilog HDL file that contains netlist information obtained from the input
//               design files.
// Device      : xc7a35ticpg236-1L
// ----------------------------------------------------------------------------
`timescale 1 ps / 1 ps

(* D0_IS_1 = "3'b001" *) (* D0_NOT_1 = "3'b011" *) (* D1_IS_1 = "3'b010" *)
(* D1_NOT_1 = "3'b100" *) (* START = "3'b000" *)
(* STRUCTURAL_NETLIST = "yes" *)
module Sequence_Detector_111
   (clk,
    rst,
    Din,
    ERR);
  input clk;
  input rst;
  input Din;
  output ERR;

  wire \<const1> ;
  wire Din;
  wire Din_IBUF;
  wire ERR;
  wire ERR_OBUF;
```

Figure 85: Q3 Netlist (One-Hot)

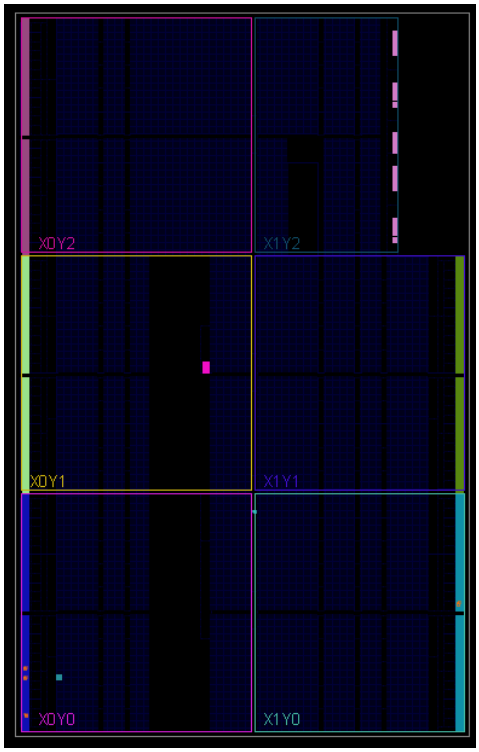

Figure 86: Q3 Device (One-Hot)

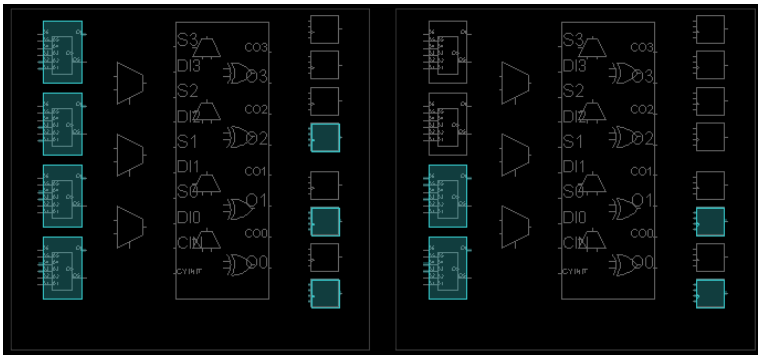

Figure 87: Q3 Zoomed Device (One-Hot)

Design Timing Summary

Setup		Hold		Pulse Width	
Worst Negative Slack (WNS):	8.357 ns	Worst Hold Slack (WHS):	0.134 ns	Worst Pulse Width Slack (WPWS):	4.500 ns
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	0.000 ns	Total Pulse Width Negative Slack (TPWS):	0.000 ns
Number of Failing Endpoints:	0	Number of Failing Endpoints:	0	Number of Failing Endpoints:	0
Total Number of Endpoints:	5	Total Number of Endpoints:	5	Total Number of Endpoints:	6

All user specified timing constraints are met.

Figure 88: Q3 Synthesis Timing Report (One-Hot)

Design Timing Summary

Setup		Hold		Pulse Width	
Worst Negative Slack (WNS):	8.711 ns	Worst Hold Slack (WHS):	0.145 ns	Worst Pulse Width Slack (WPWS):	4.500 ns
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	0.000 ns	Total Pulse Width Negative Slack (TPWS):	0.000 ns
Number of Failing Endpoints:	0	Number of Failing Endpoints:	0	Number of Failing Endpoints:	0
Total Number of Endpoints:	5	Total Number of Endpoints:	5	Total Number of Endpoints:	6

All user specified timing constraints are met.

Figure 89: Q3 Implementation Timing Report (One-Hot)

Name	Slice LUTs (20800)	Slice Registers (41600)	Bonded IOB (106)	BUFGCTRL (32)
N Sequence_Detector_111	6	5	4	1

Figure 90: Q3 Synthesis Utilization Report (One-Hot)

Name	Slice LUTs (20800)	Slice Registers (41600)	Slice (8150)	LUT as Logic (20800)	LUT Flip Flop Pairs (20800)	Bonded IOB (106)	BUFGCTRL (32)
N Sequence_Detector_111	6	5	2	6	5	4	1

Figure 91: Q3 Implementation Utilization Report (One-Hot)

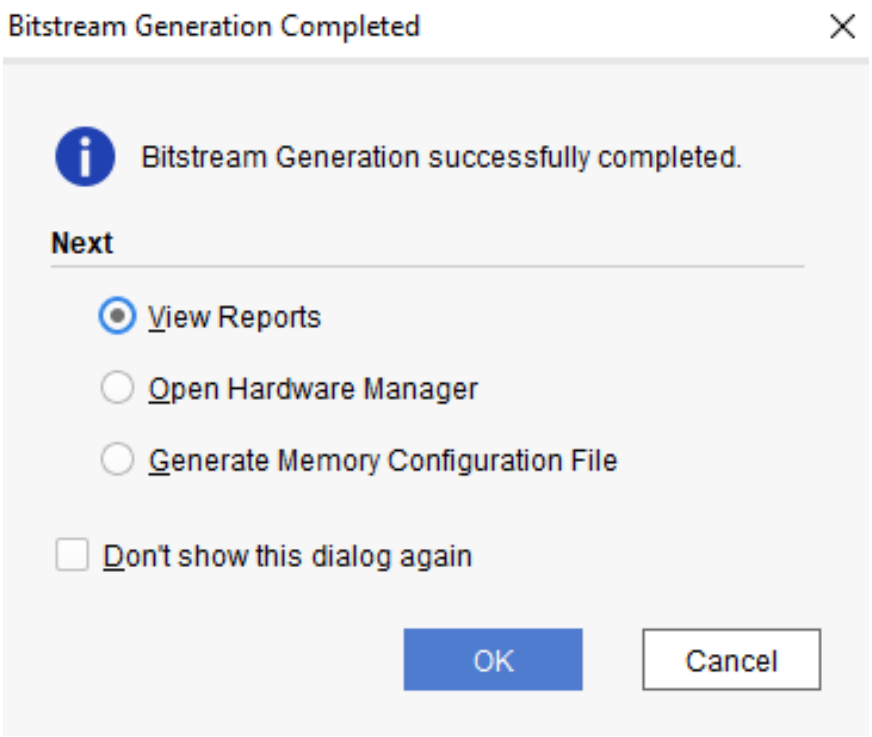

Figure 92: Q3 Bitstream

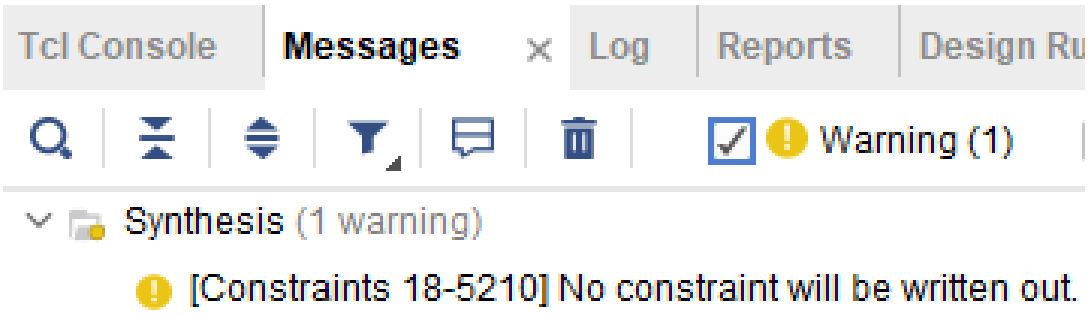

Figure 93: Q3 Messages

- If we want to use the highest frequency, then we choose **One Hot** encoding, as it has the highest setup time slack.
- If we want minimum area, we choose **Sequential** or **Gray** encoding, as they use less components than One Hot encoding.

4. Pipelined RAM:

```
1   module Reg_MUX #(parameter WIDTH=18, parameter REG= "TRUE")
2   (input clk,rst,enable, input [WIDTH-1:0] D, output [WIDTH-1:0] out);
3   reg [WIDTH-1:0] D_reg;
4
5   always @(posedge clk) begin
6       if (rst)            D_reg <= 0;
7       else if (enable)    D_reg <= D;
8   end
9
10  assign out = (REG == "TRUE")? D_reg : (REG == "FALSE")? D : 0;
11
12  endmodule
```

Figure 94: Q4 Reg with MUX

```
1   module Pipelined_RAM #(parameter MEM_WIDTH = 16, MEM_DEPTH = 1024, ADDR_SIZE = 10,
2   ADDR_PIPELINE = "FALSE", DOUT_PIPELINE = "TRUE", PARITY_ENABLE = 1)
3   (input clk,rst,blk_select,wr_en,rd_en,addr_en,dout_en, input [ADDR_SIZE-1:0] addr, input [MEM_WIDTH-1:0] din,
4   output [MEM_WIDTH-1:0] dout, output parity_out);
5
6   reg [MEM_WIDTH-1:0] dout_mux;
7   wire [ADDR_SIZE-1:0] addr_mux;
8   reg [MEM_WIDTH-1:0] mem [MEM_DEPTH-1:0];
9
10  /*module Reg_MUX #(parameter WIDTH=18, parameter REG= "TRUE")
11  (input clk,rst,enable, input [WIDTH-1:0] D, output [WIDTH-1:0] out);*/
12  Reg_MUX #(ADDR_SIZE,ADDR_PIPELINE) addr_REG(clk,rst,addr_en,addr, addr_mux);
13  Reg_MUX #(MEM_WIDTH,DOUT_PIPELINE) dout_REG(clk,rst,dout_en,dout_mux, dout);
14
15  always @(posedge clk) begin
16      if (rst)                    dout_mux <= 0;
17      else if (blk_select) begin
18          if (wr_en)              mem [addr_mux] <= din;
19          if (rd_en)              dout_mux <= mem [addr_mux];
20      end
21  end
22
23  assign parity_out = PARITY_ENABLE? ^dout : 0;
24
25  endmodule
```

Figure 95: Q4 Code

```
module Pipelined_RAM_tb ();
parameter MEM_WIDTH = 16, MEM_DEPTH = 1024, ADDR_SIZE = 10,
ADDR_PIPELINE = "FALSE", DOUT_PIPELINE = "TRUE", PARITY_ENABLE = 1;
reg clk,rst,blk_select,wr_en,rd_en,addr_en,dout_en;
reg [ADDR_SIZE-1:0] addr;
reg [MEM_WIDTH-1:0] din;
wire [MEM_WIDTH-1:0] dout;
wire parity_out;

/*module Pipelined_RAM #(parameter MEM_WIDTH = 16, MEM_DEPTH = 1024, ADDR_SIZE = 10,
ADDR_PIPELINE = "FALSE", DOUT_PIPELINE = "TRUE", PARITY_ENABLE = 1)
(input clk,rst,blk_select,wr_en,rd_en,addr_en,dout_en, input [ADDR_SIZE-1:0] addr, input [MEM_WIDTH-1:0] din,
output [MEM_WIDTH-1:0] dout, output parity_out);*/
Pipelined_RAM #(MEM_WIDTH,MEM_DEPTH,ADDR_SIZE,ADDR_PIPELINE,DOUT_PIPELINE,PARITY_ENABLE)
DUT(clk,rst,blk_select,wr_en,rd_en,addr_en,dout_en,addr,din,dout,parity_out);

initial begin
    clk=0;
    forever #5 clk=~clk;
end

initial begin
    $readmemh ("mem.dat",DUT.mem);
    rst = 1; blk_select=$random; wr_en=$random; rd_en=$random; addr_en=$random; dout_en=$random; addr=$random; din=$random;
    @(negedge clk);
    if (dout || parity_out) begin
        $display("Error in Reset");
        $stop;
    end

    rst = 0; blk_select=1; addr_en=0; dout_en=1;
    repeat (20) begin
        wr_en=$random; rd_en=$random; addr=$urandom_range(1008,1023); din=$random;
        @(negedge clk);
        @(negedge clk);
    end

    dout_en=0; wr_en=$random; rd_en=1; addr=$urandom_range(1008,1023); din=$random;
    @(negedge clk);

    blk_select=0; wr_en=1; rd_en=1; addr=$urandom_range(1008,1023); din=5;
    @(negedge clk);
    $stop;
end
endmodule
```

Figure 96: Q4 Testbench

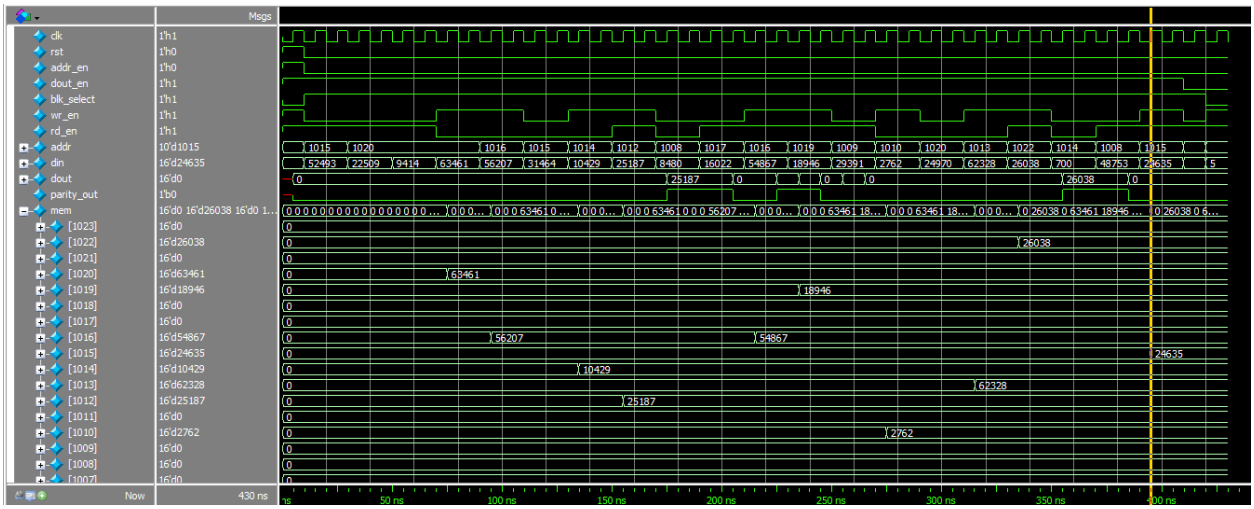

Figure 97: Q4 Wave

```
vlib work
vlog Pipelined_RAM.v Pipelined_RAM_tb.v
vsim -voptargs=+acc Pipelined_RAM_tb
add wave *
add wave -position insertpoint  \
sim:/Pipelined_RAM_tb/DUT/mem
run -all
#quit -sim
```

Figure 98: Q4 DO File

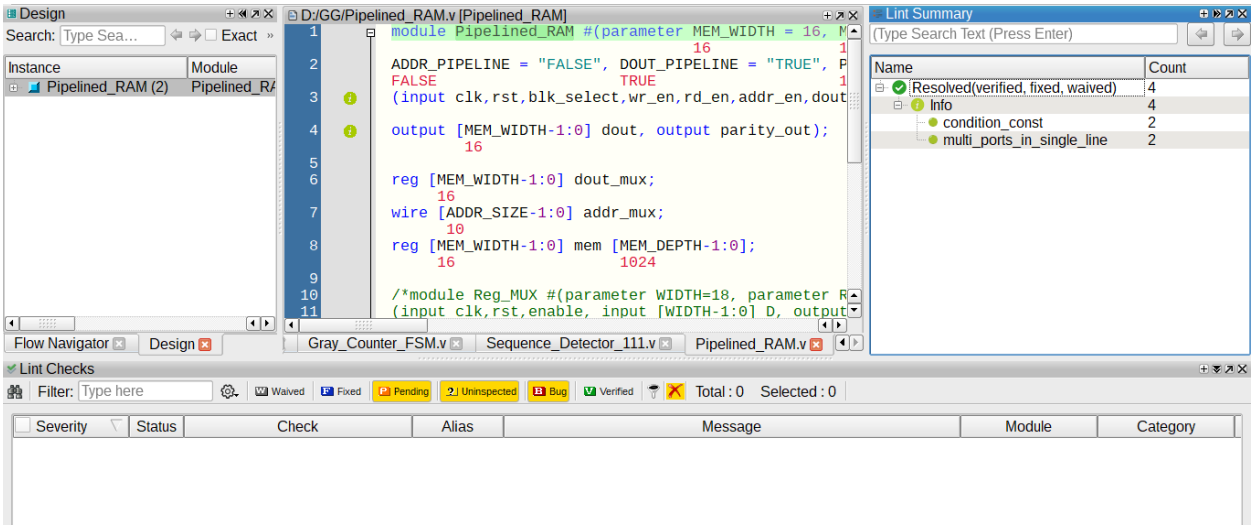

Figure 99: Q4 Linting

```
## This file is a general .xdc for the Basys3 rev B board
## To use it in a project:
## - uncomment the lines corresponding to used pins
## - rename the used ports (in each line, after get_ports) according to the top level signal names in the project

## Clock signal
set_property -dict { PACKAGE_PIN W5   IOSTANDARD LVCMOS33 } [get_ports clk]
create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports clk]


## Configuration options, can be used for all designs
set_property CONFIG_VOLTAGE 3.3 [current_design]
set_property CFGBVS VCCO [current_design]

## SPI configuration mode options for QSPI boot, can be used for all designs
set_property BITSTREAM.GENERAL.COMPRESS TRUE [current_design]
set_property BITSTREAM.CONFIG.CONFIGRATE 33 [current_design]
set_property CONFIG_MODE SPIx4 [current_design]
```

Figure 100: Q4 Constraint File

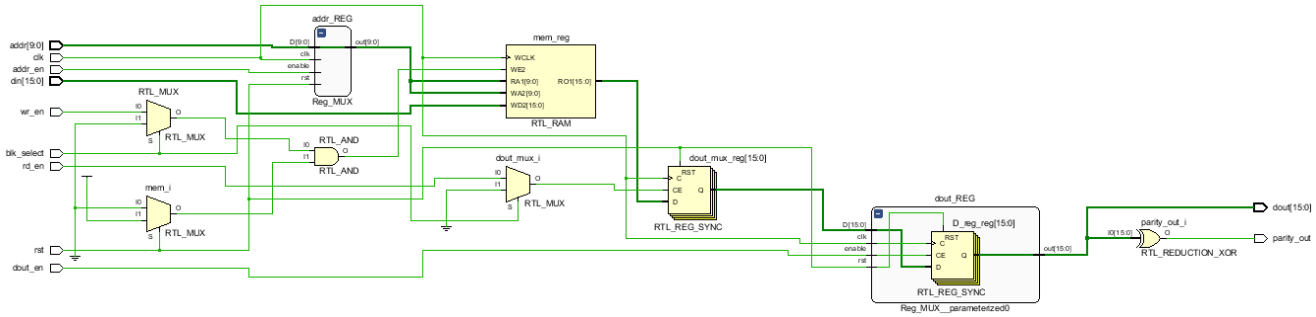

Figure 101: Q4 RTL

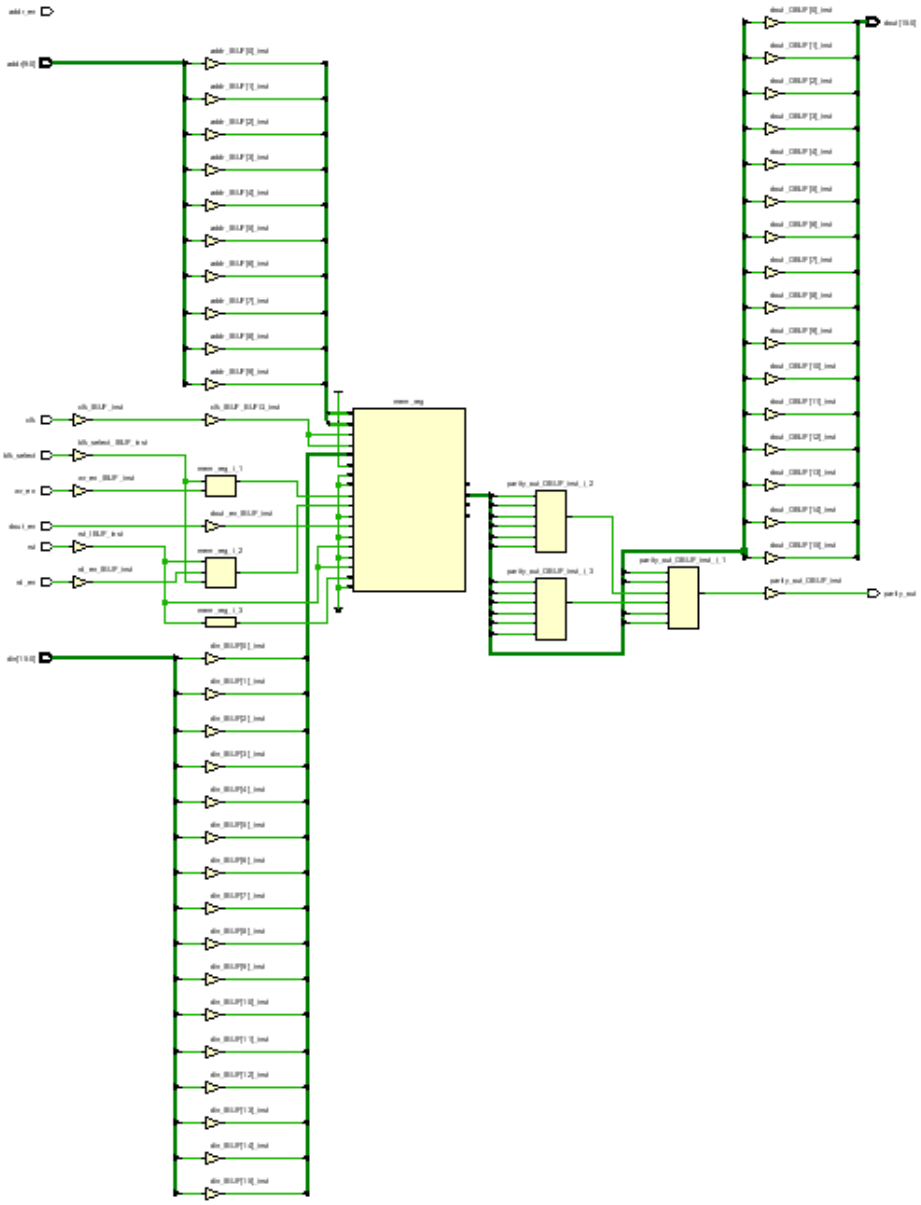

Figure 102: Q4 Synthesis

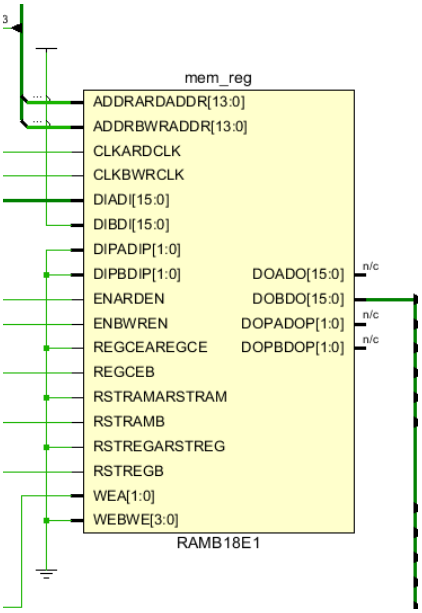

Figure 103: Q4 RAM Synthesis

```
// Copyright 1986-2018 Xilinx, Inc. All Rights Reserved.
// ----------------------------------------------------------------------------
// Tool Version: Vivado v.2018.2 (win64) Build 2258646 Thu Jun 14 20:03:12 MDT 2018
// Date        : Wed Aug  6 10:10:45 2025
// Host        : DESKTOP-DTNSAHB running 64-bit major release  (build 9200)
// Command     : write_verilog {D:/College Projects/Digital_Course_Waseem/Vv/Q4/Pipelined_RAM_Netlist.v}
// Design      : Pipelined_RAM
// Purpose     : This is a Verilog netlist of the current design or from a specific cell of the design. The output is an
//               IEEE 1364-2001 compliant Verilog HDL file that contains netlist information obtained from the input
//               design files.
// Device      : xc7a35ticpg236-1L
// ----------------------------------------------------------------------------
`timescale 1 ps / 1 ps

(* ADDR_PIPELINE = "FALSE" *) (* ADDR_SIZE = "10" *) (* DOUT_PIPELINE = "TRUE" *)
(* MEM_DEPTH = "1024" *) (* MEM_WIDTH = "16" *) (* PARITY_ENABLE = "1" *)
(* STRUCTURAL_NETLIST = "yes" *)
module Pipelined_RAM
   (clk,
    rst,
    blk_select,
    wr_en,
    rd_en,
    addr_en,
    dout_en,
    addr,
    din,
    dout,
    parity_out);
  input clk;
  input rst;
  input blk_select;
```

Figure 104: Q4 Netlist

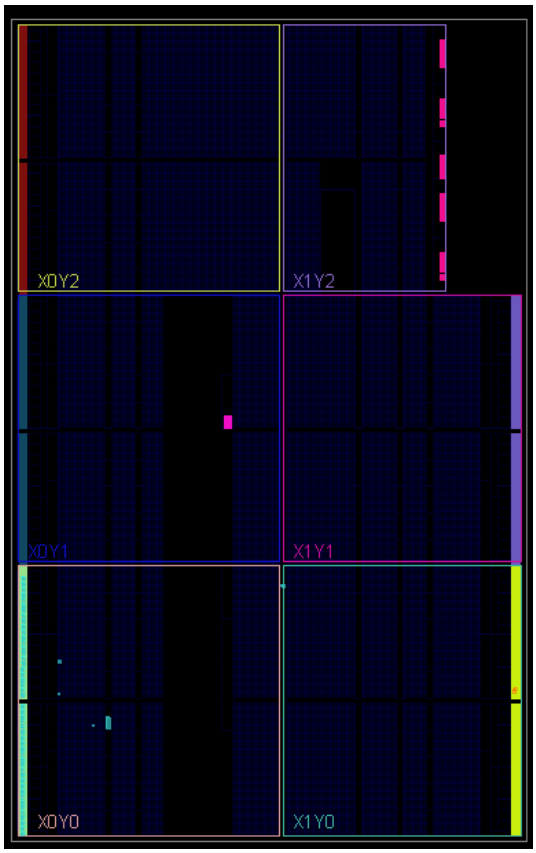

Figure 105: Q4 Device

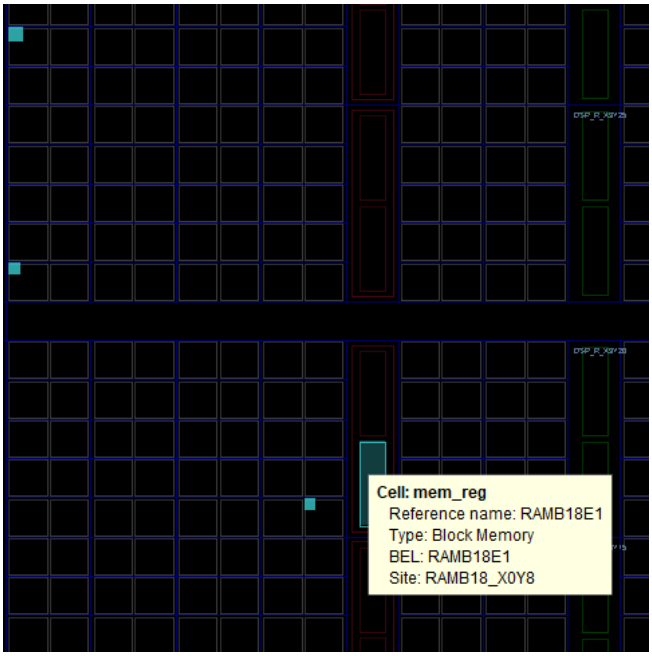

Figure 106: Q4 Zoomed Device

Name	1	Slice LUTs (20800)	Block RAM Tile (50)	Bonded IOB (106)	BUFGCTRL (32)
N Pipelined_RAM		6	0.5	49	1

Figure 107: Q4 Synthesis Utilization Report

Name	1	Slice LUTs (20800)	Slice (8150)	LUT as Logic (20800)	Block RAM Tile (50)	Bonded IOB (106)	BUFGCTRL (32)
N Pipelined_RAM		7	3	7	0.5	49	1

Figure 108: Q4 Implementation Utilization Report

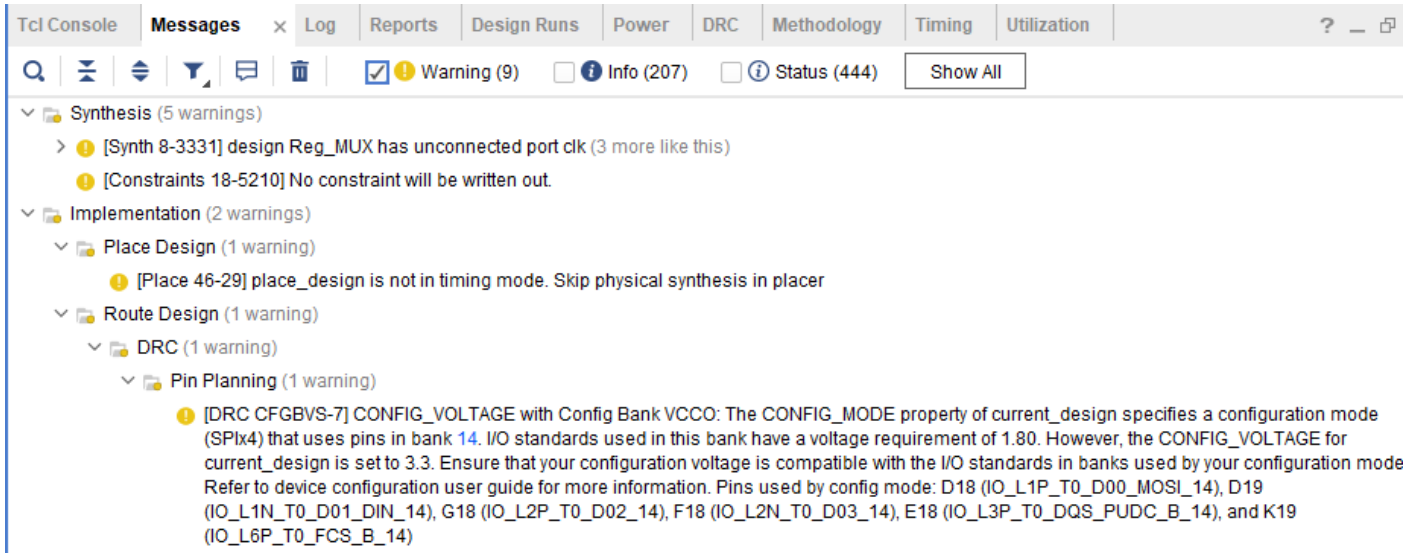

Figure 109: Q4 Messages

5. FIFO:

```
module FIFO #(parameter FIFO_WIDTH = 16, FIFO_DEPTH = 512 /* 2^9 */)
(input clk_a,clk_b,rst,wen_a,ren_b, input [FIFO_WIDTH-1:0] din_a, output reg [FIFO_WIDTH-1:0] dout_b, output full,empty);
reg [8:0] addr_wr,          //9 bit Write Counter (Pointer) (It's the address in the memory where we'll be writing)
          addr_rd,          //9 bit Read Counter (Pointer) (It's the address in the memory where we'll be reading)
          size;             //9 bit counter to Keep track of the number of data in memory to know if it's full or empty

reg [FIFO_WIDTH-1:0] mem [FIFO_DEPTH-1:0];

//Write Logic
always @(posedge clk_a) begin
    if (rst)                    {addr_wr, size} <= 0;
    else if (!full && wen_a) begin
        mem [addr_wr] <= din_a;
        addr_wr <=  addr_wr + 1;
        size <=  size + 1;
    end
end

//Read Logic
always @(posedge clk_b) begin
    if (rst)                    {dout_b, addr_rd, size} <= 0;
    else if (!empty && ren_b) begin
        dout_b <= mem [addr_rd];
        addr_rd <= addr_rd + 1;
        size <= size - 1;
    end
end

assign full = size == FIFO_DEPTH? 1 : 0;
assign empty = !size? 1 : 0;

endmodule
```

Figure 110: Q5 Code

```
module FIFO_tb ();
parameter FIFO_WIDTH = 16, FIFO_DEPTH = 512;
reg clk_a,clk_b,rst,wen_a,ren_b;
reg [FIFO_WIDTH-1:0] din_a;
wire [FIFO_WIDTH-1:0] dout_b;
wire full,empty;

/*module FIFO #(parameter FIFO_WIDTH = 16, FIFO_DEPTH = 512)
(input clk_a,clk_b,rst,wen_a,ren_b, input [FIFO_WIDTH-1:0] din_a, output reg [FIFO_WIDTH-1:0] dout_b, output full,empty);*/
FIFO #(FIFO_WIDTH,FIFO_DEPTH) DUT(clk_a,clk_b,rst,wen_a,ren_b,din_a,dout_b,full,empty);

initial begin
    clk_a = 0;
    clk_b = 0;
    forever begin
        #5 clk_a = ~clk_a;
        #7 clk_b = ~clk_b;
    end
end

initial begin
    $readmemh ("mem.dat", DUT.mem);
    rst = 1; wen_a=$random; ren_b=$random; din_a=$random;
    @(negedge clk_b);
    if (dout_b || full || !empty) begin
        $display("Error in Reset!!");
        $stop;
    end

    rst = 0;
    repeat (30) begin
        wen_a=$random; ren_b=$random; din_a=$urandom_range(1,100);
        @(negedge clk_b);
    end
    $stop;
end
endmodule
```

Figure 111: Q5 Testbench

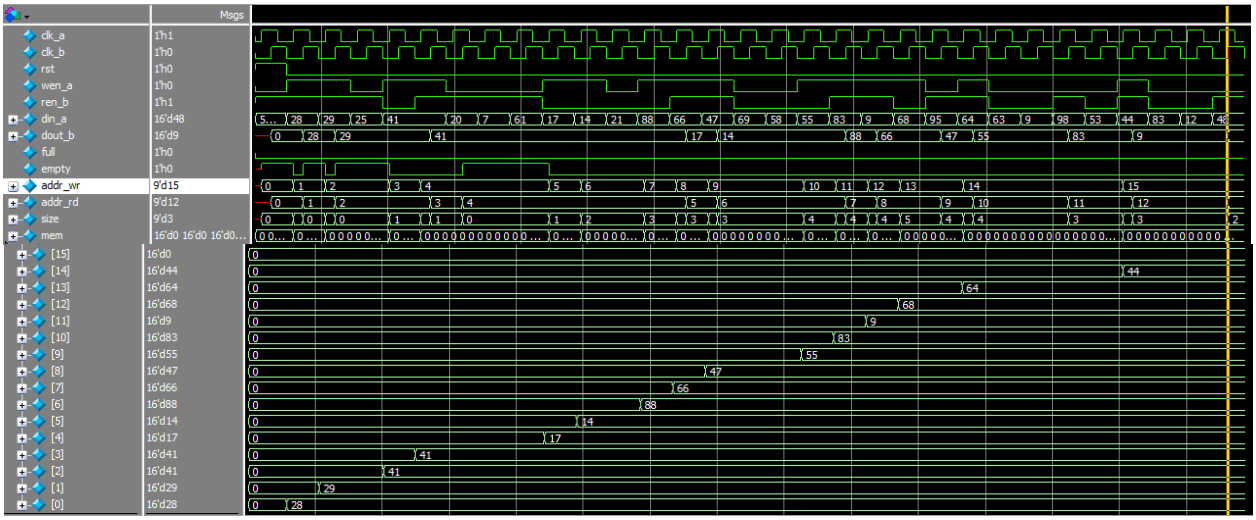

Figure 112: Q5 Wave

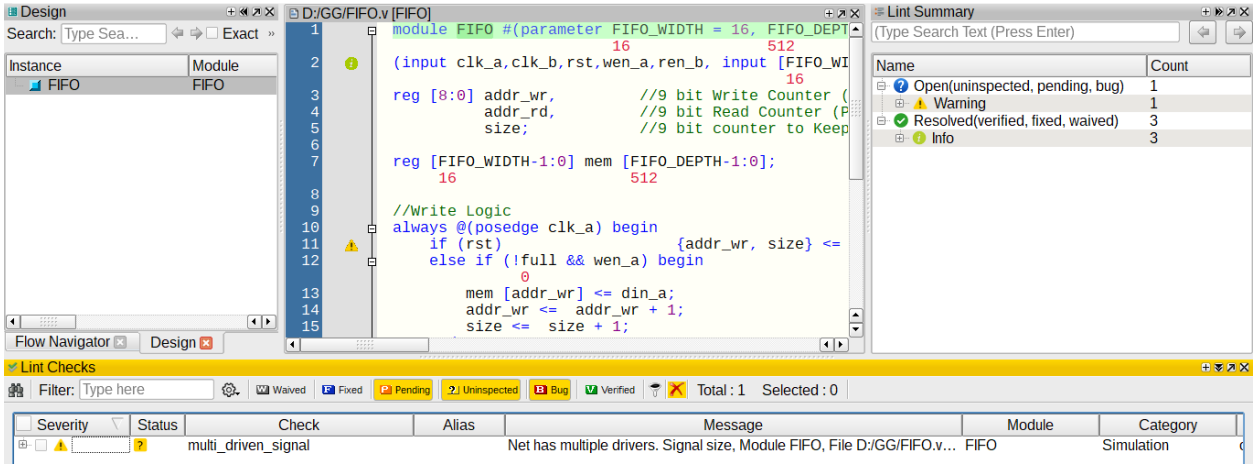

Figure 113: Q5 Linting

```
vlib work
vlog FIFO.v FIFO_tb.v
vsim -voptargs=+acc FIFO_tb
add wave *
add wave -position insertpoint  \
sim:/FIFO_tb/DUT/addr_wr  \
sim:/FIFO_tb/DUT/addr_rd  \
sim:/FIFO_tb/DUT/size  \
sim:/FIFO_tb/DUT/mem
run -all
#quit -sim
```

Figure 114: Q5 DO File

```
## This file is a general .xdc for the Basys3 rev B board
## To use it in a project:
## - uncomment the lines corresponding to used pins
## - rename the used ports (in each line, after get_ports) according to the top level signal names in the project

## Clock signal
set_property -dict { PACKAGE_PIN W5 IOSTANDARD LVCMOS33 } [get_ports clk_a]
create_clock -add -name sys_clk_a_pin -period 10.00 -waveform {0 5} [get_ports clk_a]
create_clock -add -name sys_clk_b_pin -period 10.00 -waveform {0 5} [get_ports clk_b]


## Configuration options, can be used for all designs
set_property CONFIG_VOLTAGE 3.3 [current_design]
set_property CFGBVS VCCO [current_design]

## SPI configuration mode options for QSPI boot, can be used for all designs
set_property BITSTREAM.GENERAL.COMPRESS TRUE [current_design]
set_property BITSTREAM.CONFIG.CONFIGRATE 33 [current_design]
set_property CONFIG_MODE SPIx4 [current_design]
```

Figure 115: Q5 Constraint File

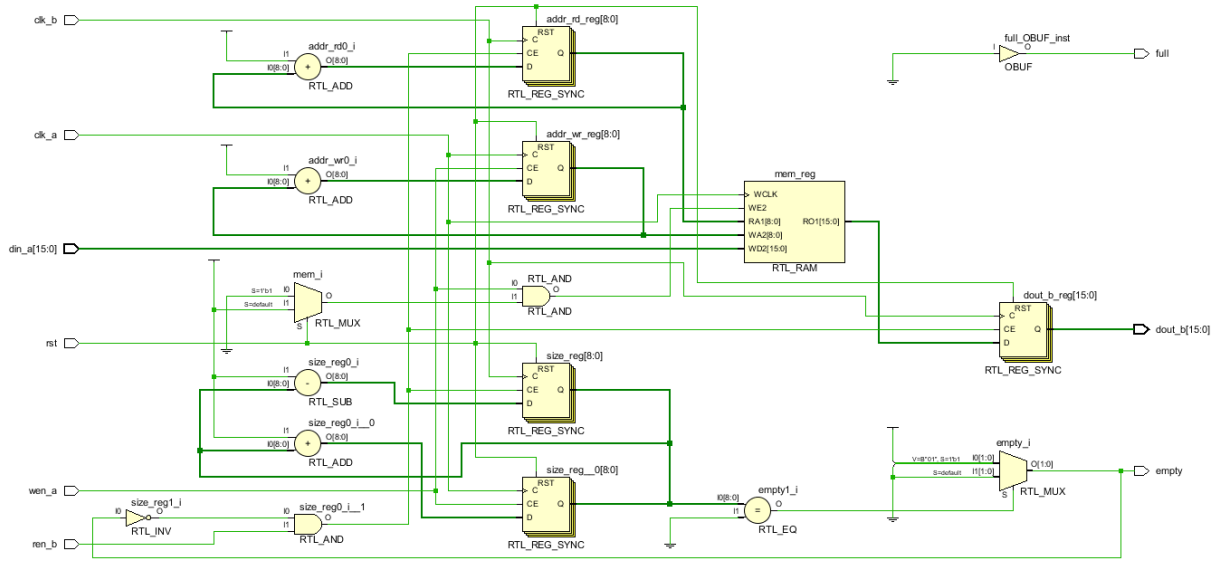

Figure 116: Q5 RTL

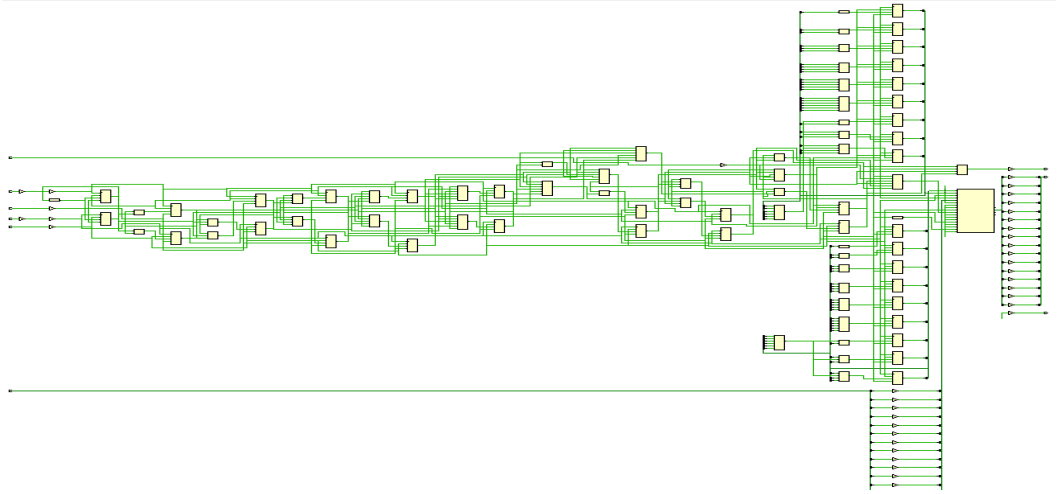

Figure 117: Q5 Synthesis

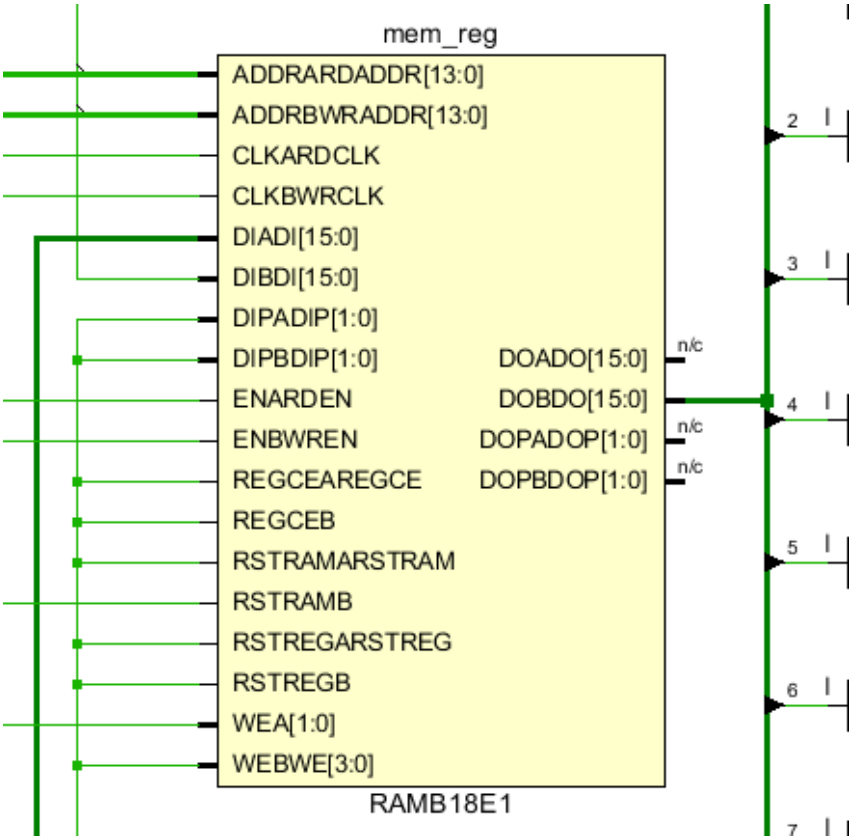

Figure 118: Q5 RAM Synthesis

```
1  // Copyright 1986-2018 Xilinx, Inc. All Rights Reserved.
2  // ----------------------------------------------------------------------------
3  // Tool Version: Vivado v.2018.2 (win64) Build 2258646 Thu Jun 14 20:03:12 MDT 2018
4  // Date        : Thu Aug  7 03:11:50 2025
5  // Host        : DESKTOP-DTNSAHB running 64-bit major release  (build 9200)
6  // Command     : write_verilog {D:/College Projects/Digital_Course_Waseem/Vv/Q5/FIFO_Netlist.v}
7  // Design      : FIFO
8  // Purpose     : This is a Verilog netlist of the current design or from a specific cell of the design. The output is an
9  //               IEEE 1364-2001 compliant Verilog HDL file that contains netlist information obtained from the input
10 //               design files.
11 // Device      : xc7a35ticpg236-1L
12 // ----------------------------------------------------------------------------
13 `timescale 1 ps / 1 ps
14
15 (* FIFO_DEPTH = "512" *) (* FIFO_WIDTH = "16" *)
16 (* STRUCTURAL_NETLIST = "yes" *)
17 module FIFO
18    (clk_a,
19     clk_b,
20     rst,
21     wen_a,
22     ren_b,
23     din_a,
24     dout_b,
25     full,
26     empty);
27   input clk_a;
28   input clk_b;
29   input rst;
30   input wen_a;
31   input ren_b;
32   input [15:0]din_a;
```

Figure 119: Q5 Netlist

Design Timing Summary

Setup		Hold		Pulse Width	
Worst Negative Slack (WNS):	5.504 ns	Worst Hold Slack (WHS):	0.232- ns	Worst Pulse Width Slack (WPWS):	4.500 ns
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	1.724- ns	Total Pulse Width Negative Slack (TPWS):	0.000 ns
Number of Failing Endpoints:	0	Number of Failing Endpoints:	15	Number of Failing Endpoints:	0
Total Number of Endpoints:	73	Total Number of Endpoints:	73	Total Number of Endpoints:	40

Timing constraints are not met.

Figure 120: Q5 Synthesis Timing Report

Name 1	Slice LUTs (20800)	Slice Registers (41600)	Block RAM Tile (50)	Bonded IOB (106)	BUFGCTRL (32)
N FIFO	38	36	0.5	39	2

Figure 121: Q5 Synthesis Utilization Report

Figure 122: Q5 Messages

The critical warning is due to "size" being assigned in two different always blocks.. I tried to figure out another way to get the logic of "full" and "empty" without "size", or try to manipulate "size" in another way that do the same functionality without any errors, but I couldn't think of anything.

6. IP ALSU:

What changed in the RTL code due to using IPs:

```
 8   wire [3 : 0] adder_out;
 9   generate
10       if (FULL_ADDER == "ON")
11           c_addsub_1 IP_Full_Adder(
12             .A(A_reg),         // input wire [2 : 0] A
13             .B(B_reg),         // input wire [2 : 0] B
14             .C_IN(cin_reg),    // input wire C_IN
15             .S(adder_out)              // output wire [3 : 0] S
16           );
17       else if (FULL_ADDER == "OFF")
18           c_addsub_1 IP_Half_Adder(
19             .A(A_reg),         // input wire [2 : 0] A
20             .B(B_reg),         // input wire [2 : 0] B
21             .C_IN(0),          // input wire C_IN
22             .S(adder_out)              // output wire [3 : 0] S
23           );
24   endgenerate
25
26   wire [5 : 0] multiply_out;
27   mult_gen_0 IP_multiplier (
28     .A(A_reg),  // input wire [2 : 0] A
29     .B(B_reg),  // input wire [2 : 0] B
30     .P(multiply_out)  // output wire [5 : 0] P
31   );
```

Figure 123: Q6 Code (1)

```
74                  3'b010: begin                                   //Addition
75                      if (red_op_A_reg || red_op_B_reg)       {out,leds} <= {6'b0 , ~leds};          //Invalid
76                      else                                    {out,leds} <= {2'b0, adder_out , 16'b0};
77                  end
78
79                  3'b011: begin                                  //Multiplication
80                      if (red_op_A_reg || red_op_B_reg)       {out,leds} <= {6'b0 , ~leds};          //Invalid
81                      else                                    {out,leds} <= {multiply_out , 16'b0};
82                  end
```

Figure 124: Q6 Code (2)

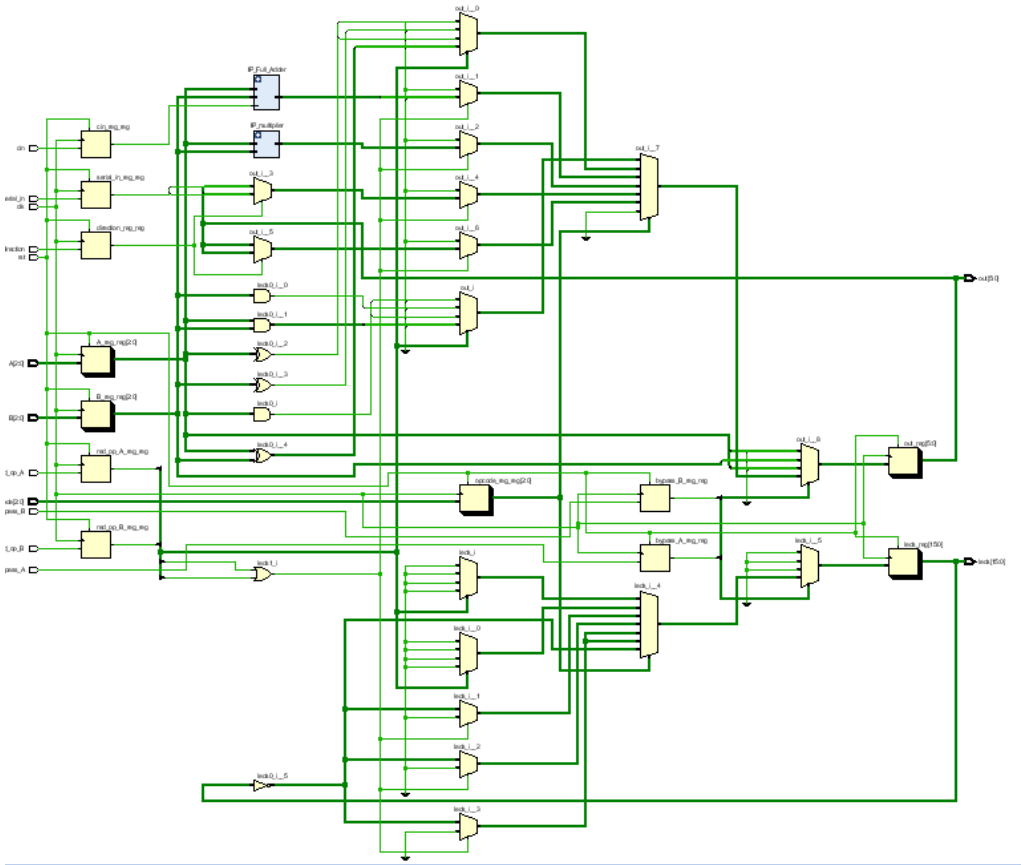

Figure 125: Q6 RTL

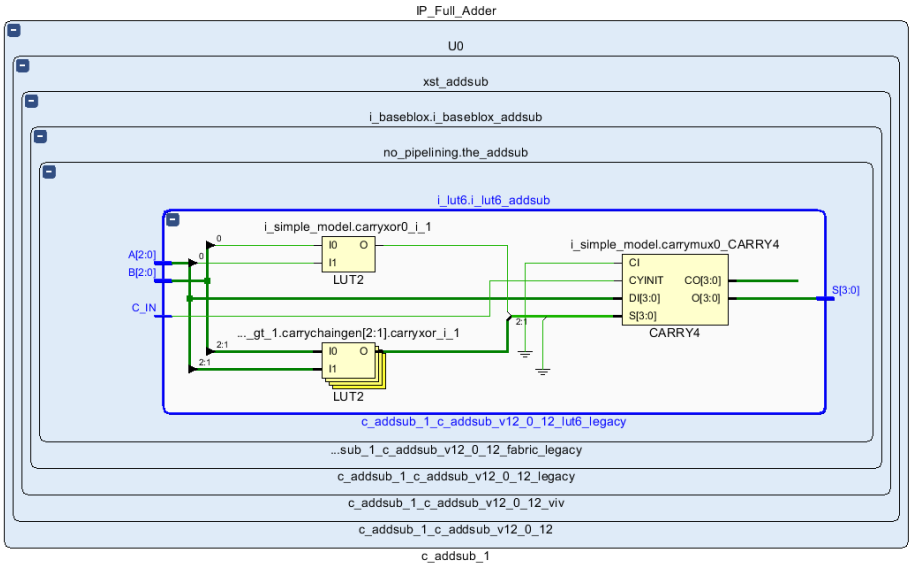

Figure 126: Q6 IP Adder RTL

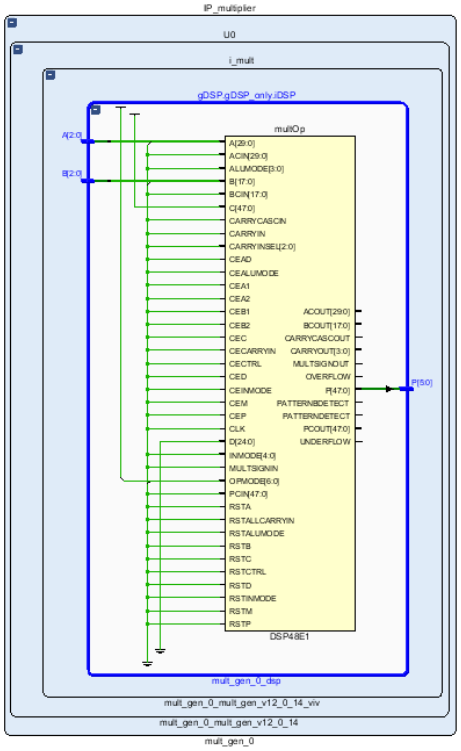

Figure 127: Q6 IP Multiplier RTL

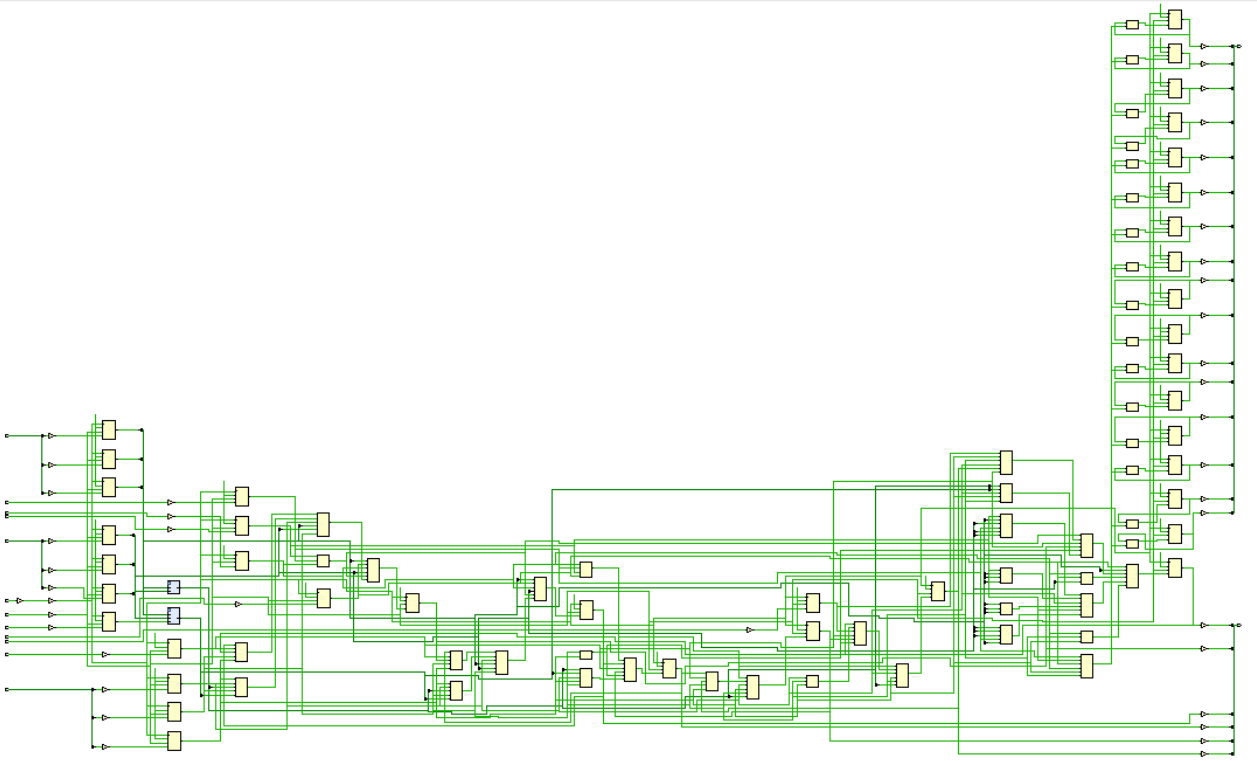

Figure 128: Q6 Synthesis

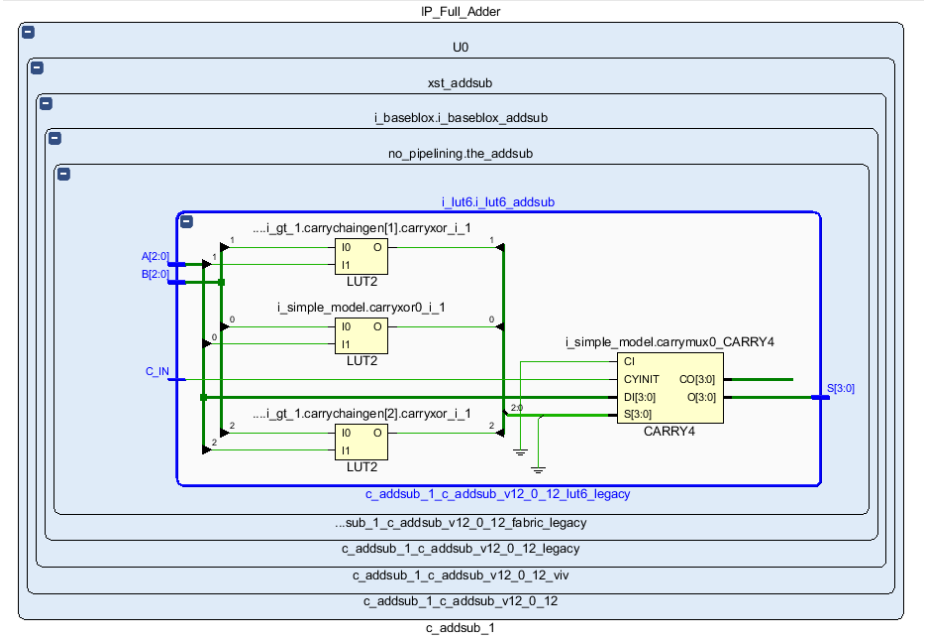

Figure 129: Q6 IP Adder Synthesis

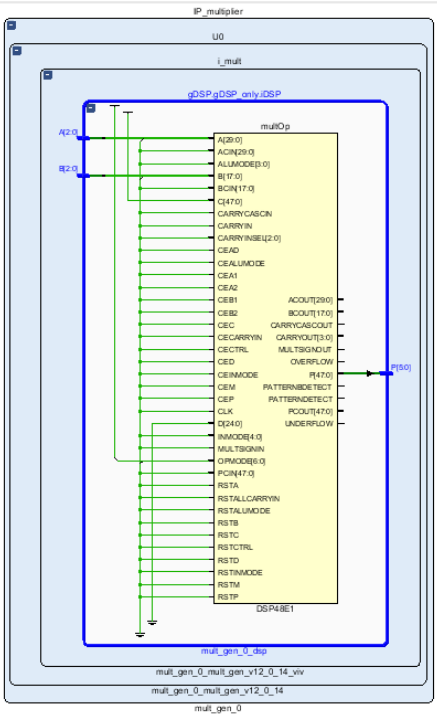

Figure 130: Q6 IP Multiplier Synthesis

```
// Copyright 1986-2018 Xilinx, Inc. All Rights Reserved.
// ----------------------------------------------------------------------------
// Tool Version: Vivado v.2018.2 (win64) Build 2258646 Thu Jun 14 20:03:12 MDT 2018
// Date        : Thu Aug  7 04:24:24 2025
// Host        : DESKTOP-DTNSAHB running 64-bit major release  (build 9200)
// Command     : write_verilog {D:/College Projects/Digital_Course_Waseem/Vv/Q6/ALSY_IP_Netlist.v}
// Design      : ALSU_IP
// Purpose     : This is a Verilog netlist of the current design or from a specific cell of the design. The output is an
//               IEEE 1364-2001 compliant Verilog HDL file that contains netlist information obtained from the input
//               design files.
// Device      : xc7a35ticpg236-1L
// ----------------------------------------------------------------------------
`timescale 1 ps / 1 ps

(* FULL_ADDER = "ON" *) (* INPUT_PRIORITY = "A" *)
(* STRUCTURAL_NETLIST = "yes" *)
module ALSU_IP
   (clk,
    rst,
    cin,
    serial_in,
    red_op_A,
    red_op_B,
    bypass_A,
    bypass_B,
    direction,
    A,
    B,
    opcode,
    out,
    leds);
  input clk;
```

Figure 131: Q6 Netlist

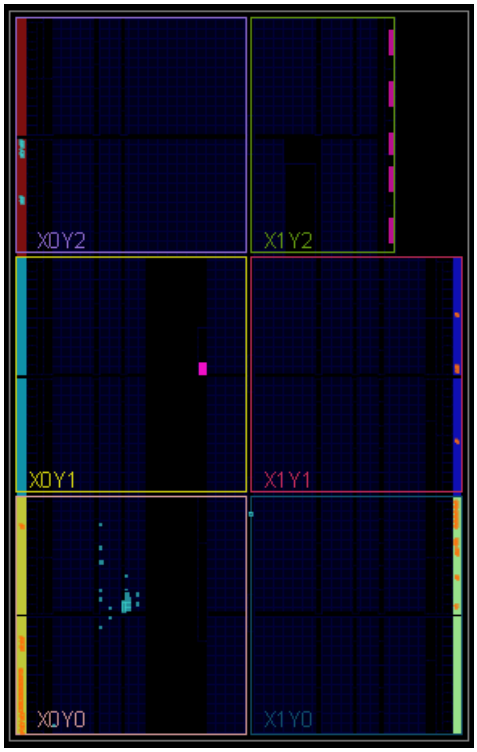

Figure 132: Q6 Device

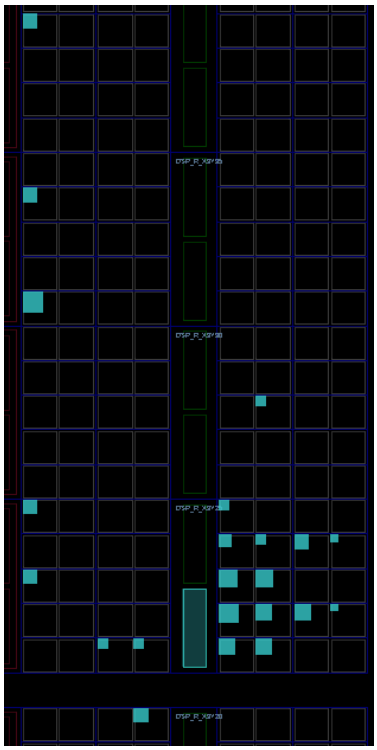

Figure 133: Q6 Zoomed Device

Design Timing Summary

Setup		Hold		Pulse Width	
Worst Negative Slack (WNS):	2.481 ns	Worst Hold Slack (WHS):	0.206 ns	Worst Pulse Width Slack (WPWS):	4.500 ns
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	0.000 ns	Total Pulse Width Negative Slack (TPWS):	0.000 ns
Number of Failing Endpoints:	0	Number of Failing Endpoints:	0	Number of Failing Endpoints:	0
Total Number of Endpoints:	22	Total Number of Endpoints:	22	Total Number of Endpoints:	39

All user specified timing constraints are met.

Figure 134: Q6 Synthesis Timing Report

Design Timing Summary

Setup		Hold		Pulse Width	
Worst Negative Slack (WNS):	2.123 ns	Worst Hold Slack (WHS):	0.283 ns	Worst Pulse Width Slack (WPWS):	4.500 ns
Total Negative Slack (TNS):	0.000 ns	Total Hold Slack (THS):	0.000 ns	Total Pulse Width Negative Slack (TPWS):	0.000 ns
Number of Failing Endpoints:	0	Number of Failing Endpoints:	0	Number of Failing Endpoints:	0
Total Number of Endpoints:	28	Total Number of Endpoints:	28	Total Number of Endpoints:	45

All user specified timing constraints are met.

Figure 135: Q6 Implementation Timing Report

Name 1	Slice LUTs (20800)	Slice Registers (41600)	DSPs (90)	Bonded IOB (106)	BUFGCTRL (32)
ALSU_IP	36	38	1	40	1
IP_Full_Adder (c_adds...	3	0	0	0	0
IP_multiplier (mult_ge...	0	0	1	0	0

Figure 136: Q6 Synthesis Utilization Report

Name 1	Slice LUTs (20800)	Slice Registers (41600)	Slice (8150)	LUT as Logic (20800)	LUT Flip Flop Pairs (20800)	DSPs (90)	Bonded IOB (106)	BUFGCTRL (32)
ALSU_IP	36	44	24	36	12	1	40	1
IP_Full_Adder (c_adds...	3	0	1	3	0	0	0	0
IP_multiplier (mult_ge...	0	0	0	0	0	1	0	0

Figure 137: Q6 Implementation Utilization Report

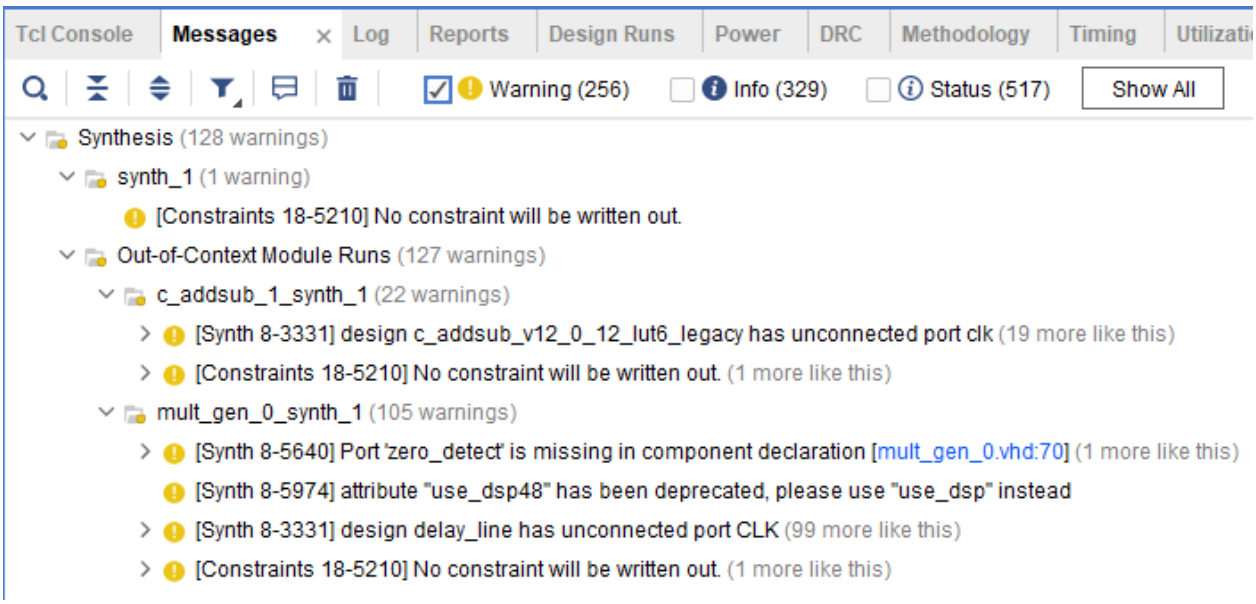

Figure 138: Q6 Messages