CMPE 260

Laboratory Exercise 5: Binary Multiplier with Built in Self Test

Course section: CMPE26001L1.2165

Name: Hrishikesh Nitin Moholkar

Lecture instructor: Cliver, Richard

Teaching Assistant: Grigoryev, Iosif

Date: 04/19/2017

Abstract

The exercise deals with the development of (BIST) which is used to confirm the functionality of ip-core generated multiplier. The BIST consists of LFSR (Linear Feedback Shift Register) and MISR (Multiple Input Signature Register). The LFSR generates output signature whenever the test mode is enabled. This signature is fed to the multiplier and the output is fed to the MISR. The MISR generates the final signature which is used by the user for testing circuits. The output of the BIST will be a normal product of inputs if the test mode is disabled and will be the output signature from MISR if test mode is enabled. This lab requires us to generate a sequence of random numbers when test mode is set to 1 and should do until 100 clock cycles. Once it reaches 100^{th} cycle it should latch to a particular value. Before that, whenever the test mode is enabled, the sequence which is generated should be same whenever it is enabled after certain interval. The behavioral simulation, post-route simulation and the hardware simulation works as expected. The overall lab is a success.

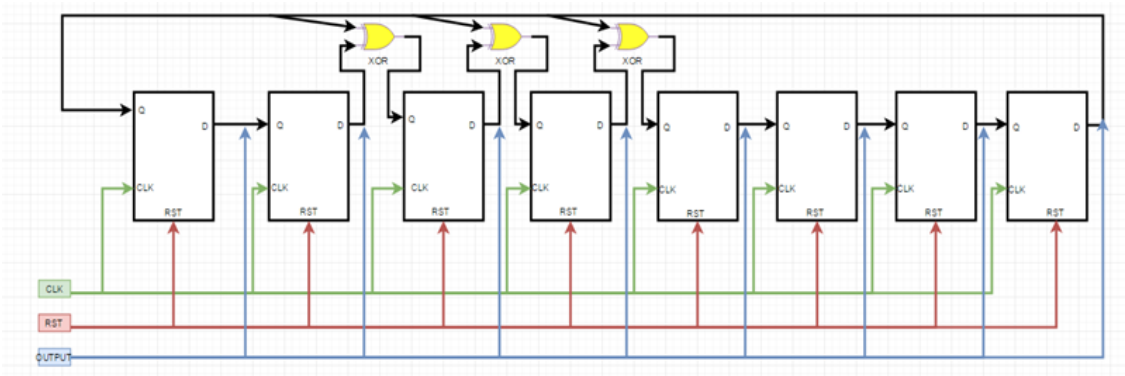
Design Methodology

Binary Multiplier with Built in Self-Test:

1) LFSR
2) MISR

LFSR:

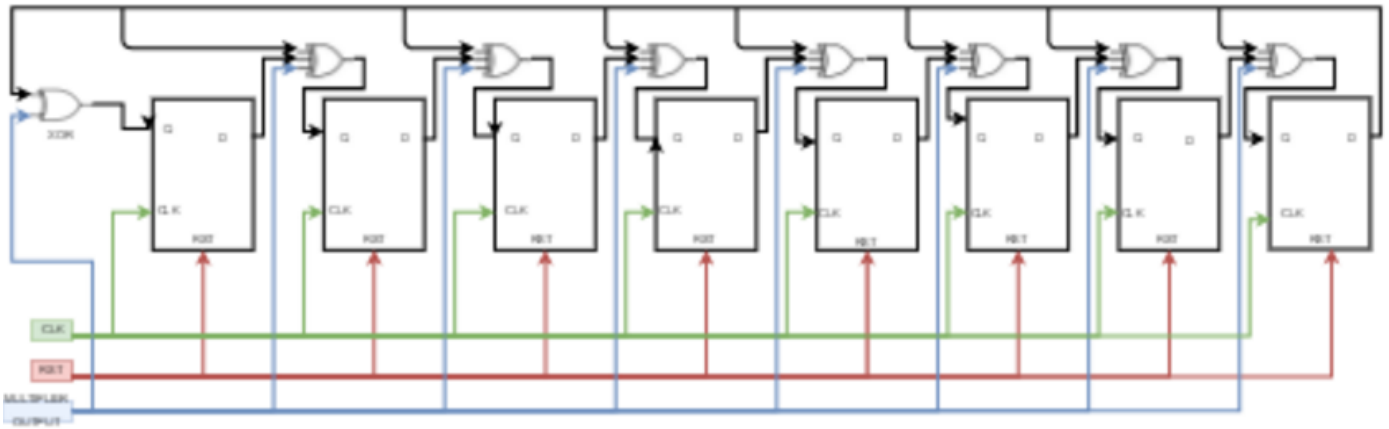
The LFSR consists of 8 d flip-flops, 3 ex-or gates. It has clock and enable as inputs. Output is taken from each output port of d flip-flop.



LFSR

The input to the ex-or gate in LFSR should not be zero. If zero is fed then it will latch to one particular value hence defeating its purpose as a random number generator.

MISR:
MISR consists of 8 d flip-flop and 8 ex-or gates. The inputs are as follows:
Multiplier output and clock.

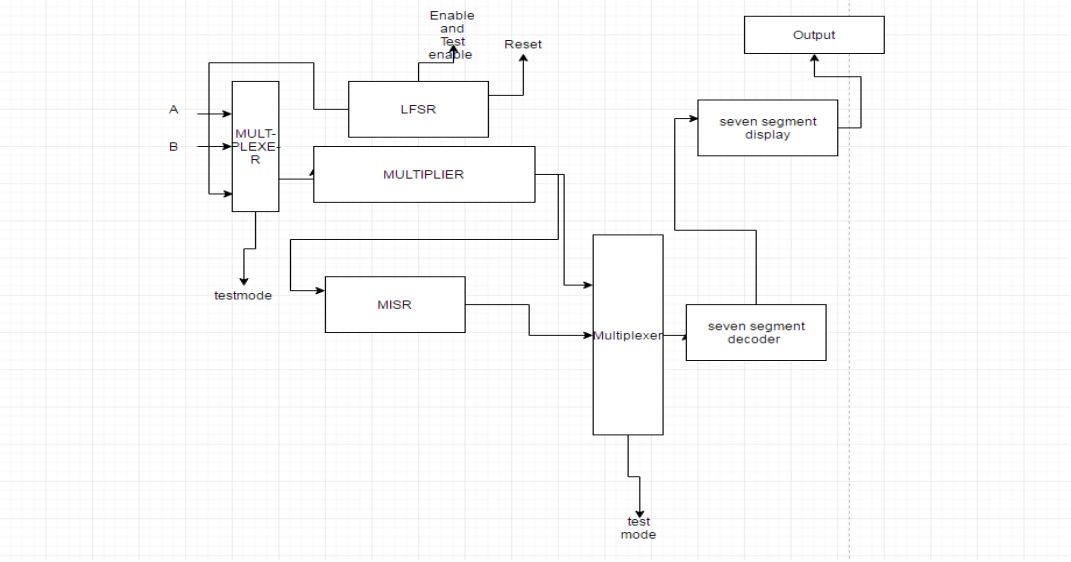


MISR

IP-core multiplier:
This is a built in multiplier. Two inputs which are 4 bits long and one output which is 8 bits long.

Wrapper:
The wrapper encloses the LFSR, MISR, Counter, Multiplier, Multiplexer, seven-segment decoder and seven segment display.

The inputs to wrapper are as follows:
A: input data which is 3 bits long
B: input data which is 3 bits long
Enable: 1 bit long
Test enable: 1 bit long
Outputs are as follows:
Hundred, tens, ones: 1 bit long

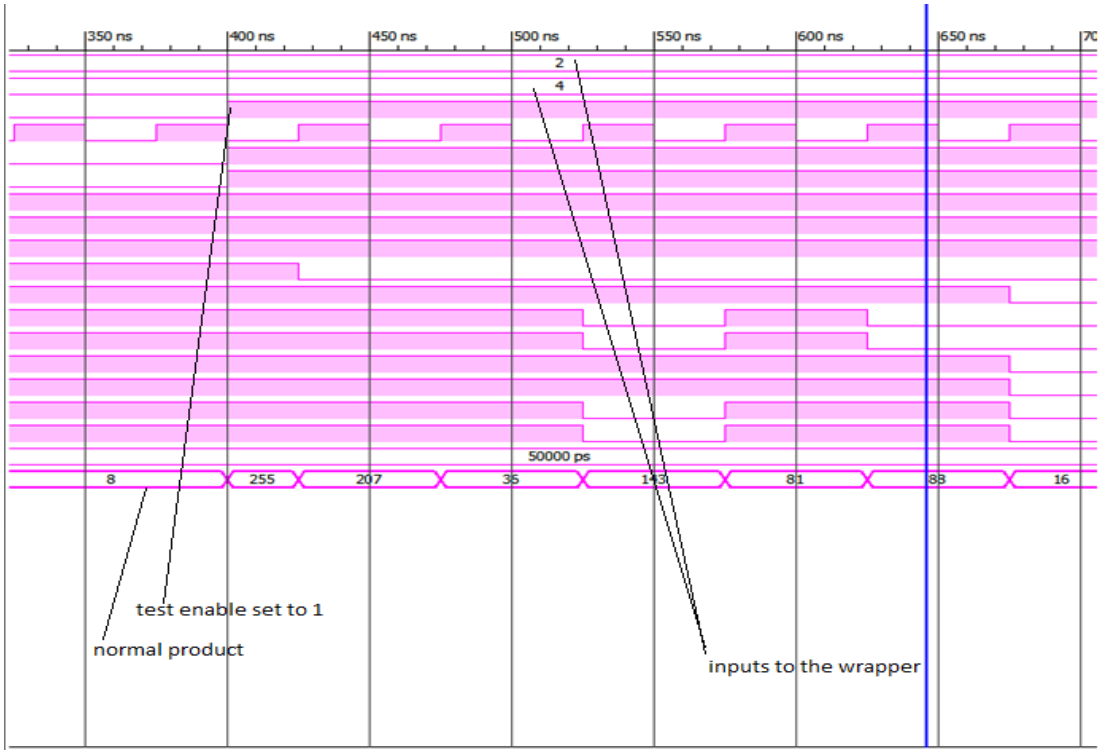


Wrapper

Results

Behavioral Simulation:

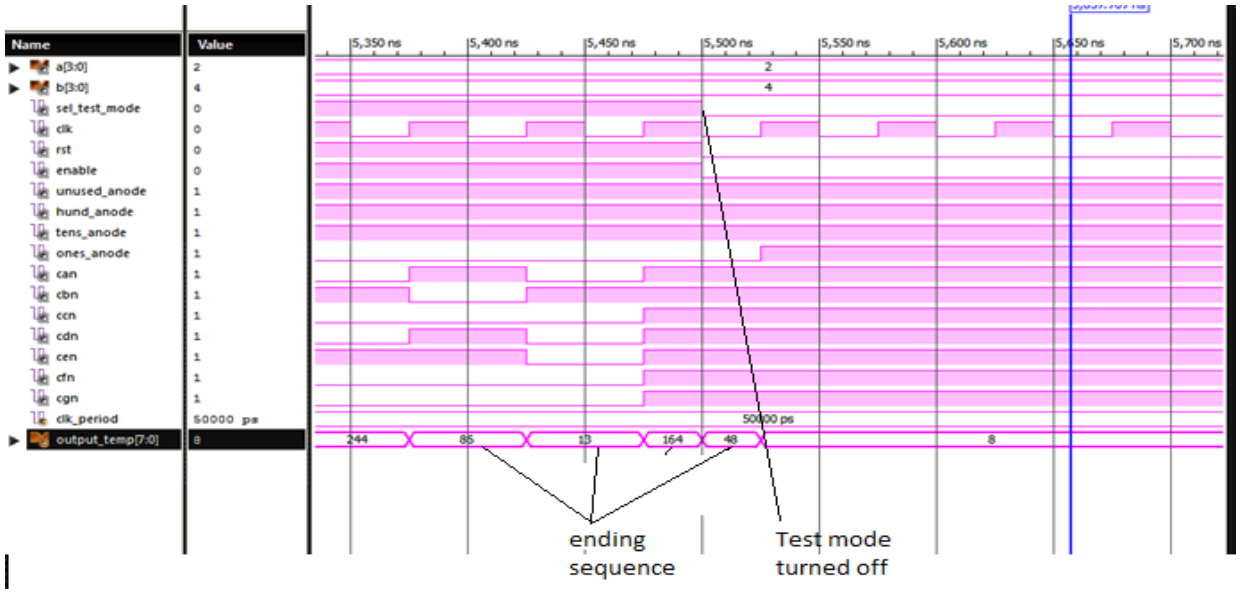
Part 1:



Test-mode on for wrapper

As per the above simulation, the test mode is turned on after 400 ns. Before that the output of the wrapper is normal product of multiplication. The random numbers are generated when the test mode is enabled. The starting number is 255.

Part 2:



Test-mode disabled

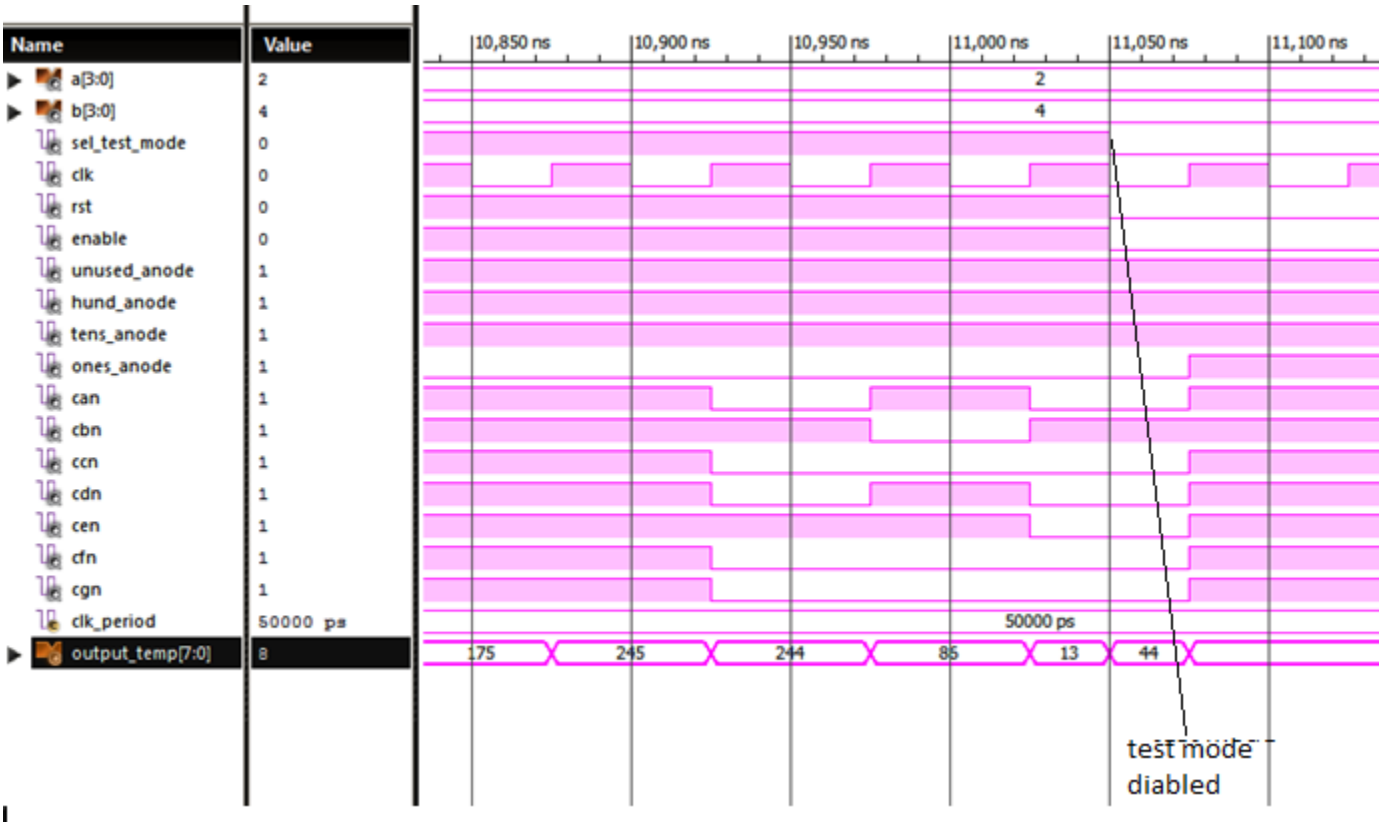
The test mode is disabled at 5500 ns. The ending sequence generated is as follows: 244, 85, 13, 44.

Part 3:

Test mode enabled at 6000 ns

The test mode is enabled at 6000 ns. The starting sequence is 255, 207 which is same as the previous sequence.

Part 4:



Test mode disabled at 11, 050 ns

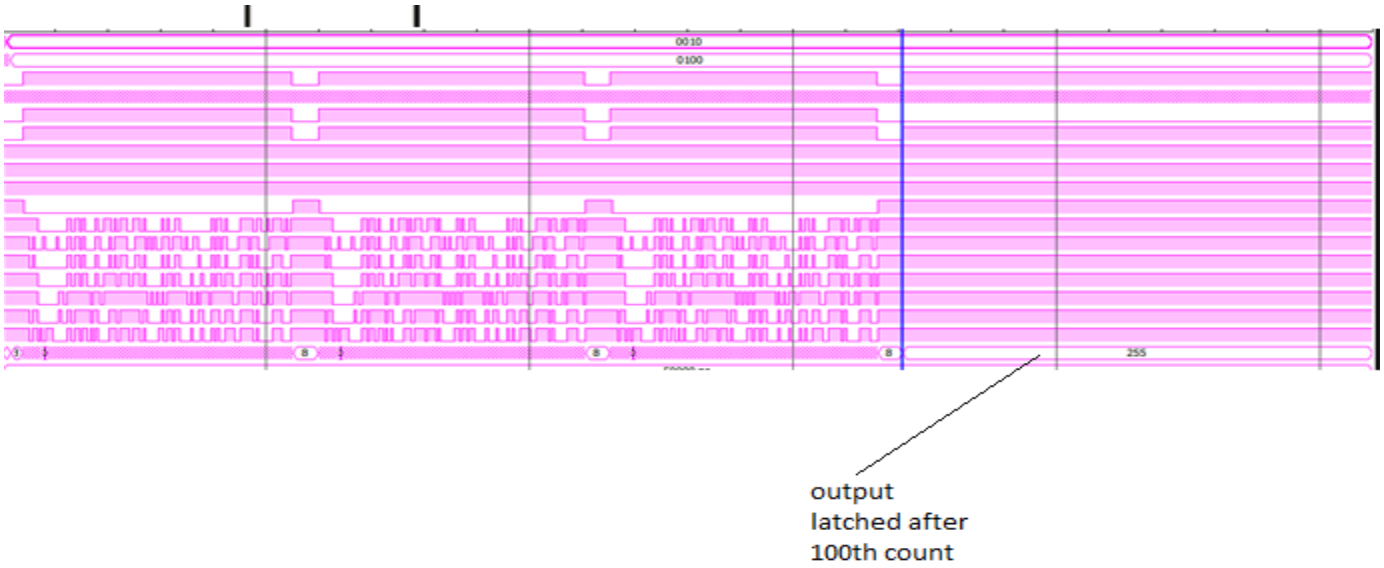
The test mode is disabled at 11, 050 ns. The ending sequence is same as the previous ending sequence. The sequence is as follows: 244, 85, 13, and 44.

Part 5:

Test mode turned off at 16, 500 ns

The ending sequence is as follows: 244, 85, 13, and 44.

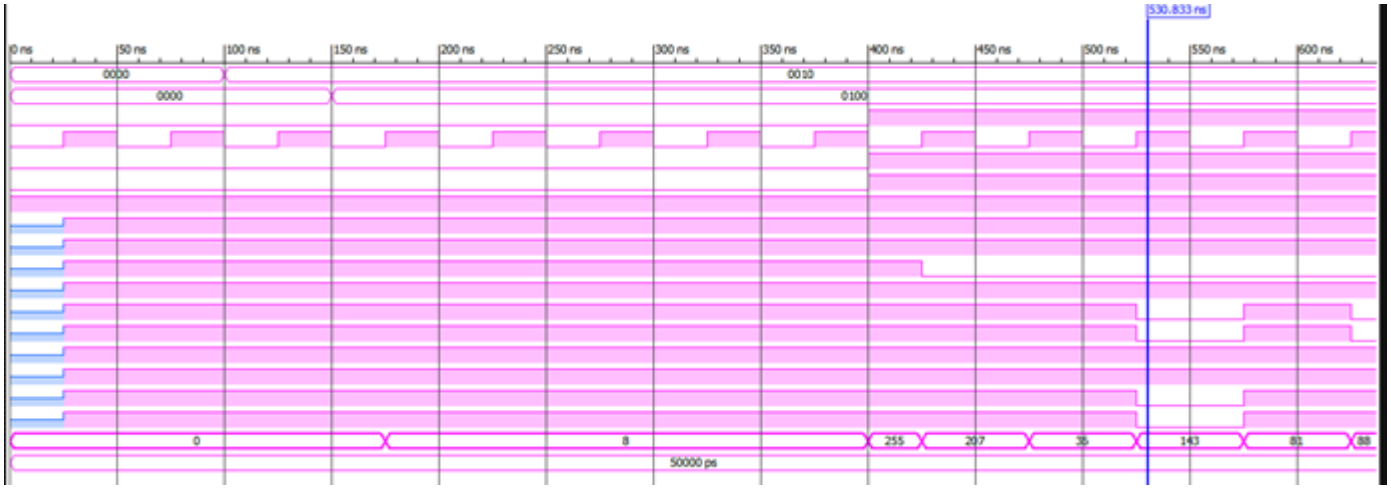
Part 6:



After 100th count, the output gets latched to 255.

Post-Route simulation:

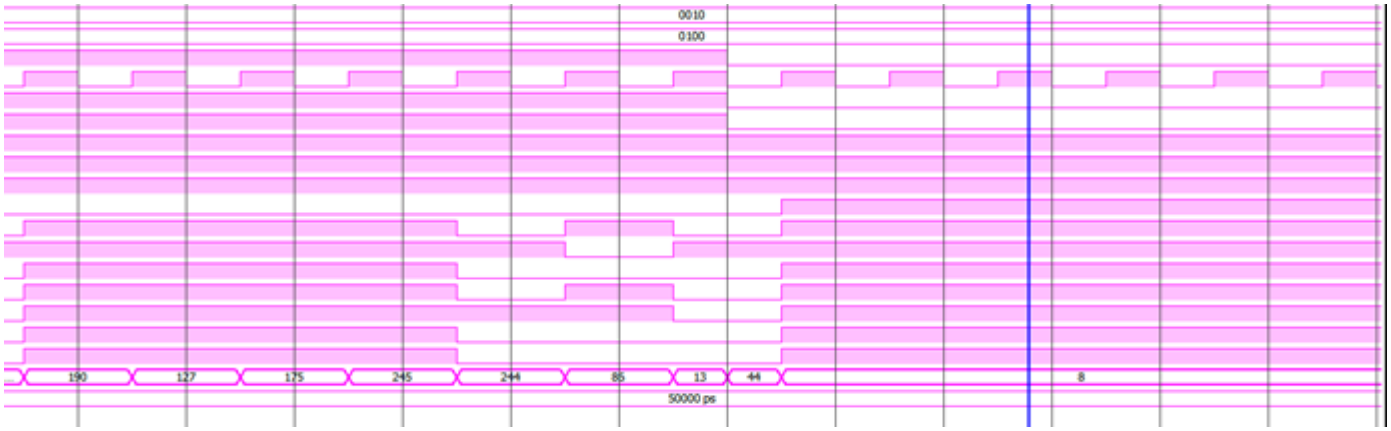
Part1:



Post route test mode enabled

When test mode is enabled the sequence starts with the values 255, 207, 36 etc.

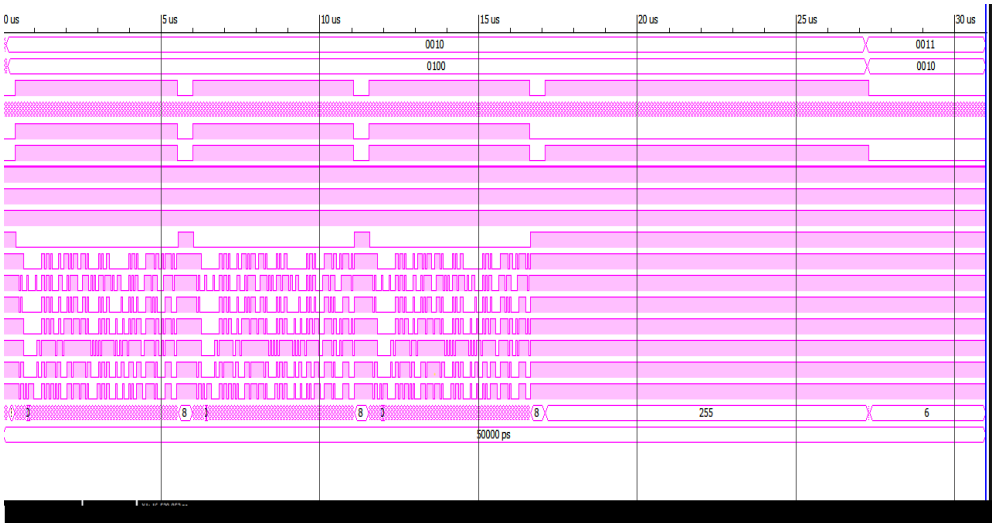
Part2:



Post route test mode disabled

The sequence ends at 86, 13 and 44. The same sequence happens every time test-enable is set to 1.

Part 3:



Post route simulation

The sequence for post-route starts with 255 and ends with 44. The same sequence repeats whenever the test mode is enabled. After 100th count the value get latched to 255. At last, the test mode is turned off. The numbers input are 3 and 2. As the test mode is off, the output is 6. Hence this **shows** that the **multiplier works**.

Area used:

```
Device Utilization Summary:

Slice Logic Utilization:
  Number of Slice Registers:                    53 out of  18,224     1%
    Number used as Flip Flops:                  53
    Number used as Latches:                      0
    Number used as Latch-thrus:                  0
    Number used as AND/OR logics:                0
  Number of Slice LUTs:                        107 out of   9,112     1%
    Number used as logic:                      104 out of   9,112     1%
      Number using O6 output only:              51
      Number using O5 output only:              36
      Number using O5 and O6:                   17
      Number used as ROM:                        0
    Number used as Memory:                       0 out of   2,176     0%
    Number used exclusively as route-thrus:      3
      Number with same-slice register load:      0
      Number with same-slice carry load:         3
      Number with other load:                    0

Slice Logic Distribution:
  Number of occupied Slices:                    42 out of   2,278     1%
  Number of MUXCYs used:                        64 out of   4,556     1%
  Number of LUT Flip Flop pairs used:          111
    Number with an unused Flip Flop:            61 out of     111    54%
    Number with an unused LUT:                   4 out of     111     3%
    Number of fully used LUT-FF pairs:          46 out of     111    41%
    Number of slice register sites lost
      to control set restrictions:               0 out of  18,224     0%

  A LUT Flip Flop pair for this architecture represents one LUT paired with
  one Flip Flop within a slice.  A control set is a unique combination of
  clock, reset, set, and enable signals for a registered element.
  The Slice Logic Distribution report is not meaningful if the design is
  over-mapped for a non-slice resource or if Placement fails.
```

Area used by logic

As per the above picture, the number of flip flops used is 53. The number of slices occupied is 107. The number of LUT flip flop pair used is 111.

Timing Results:

Constraint	Check	Worst Case Slack	Best Case Achievable	Timing Errors	Timing Score
Autotimespec constraint for clock net clk_BUFGP	SETUP	N/A	4.120ns	N/A	0
	HOLD	0.450ns		0	0

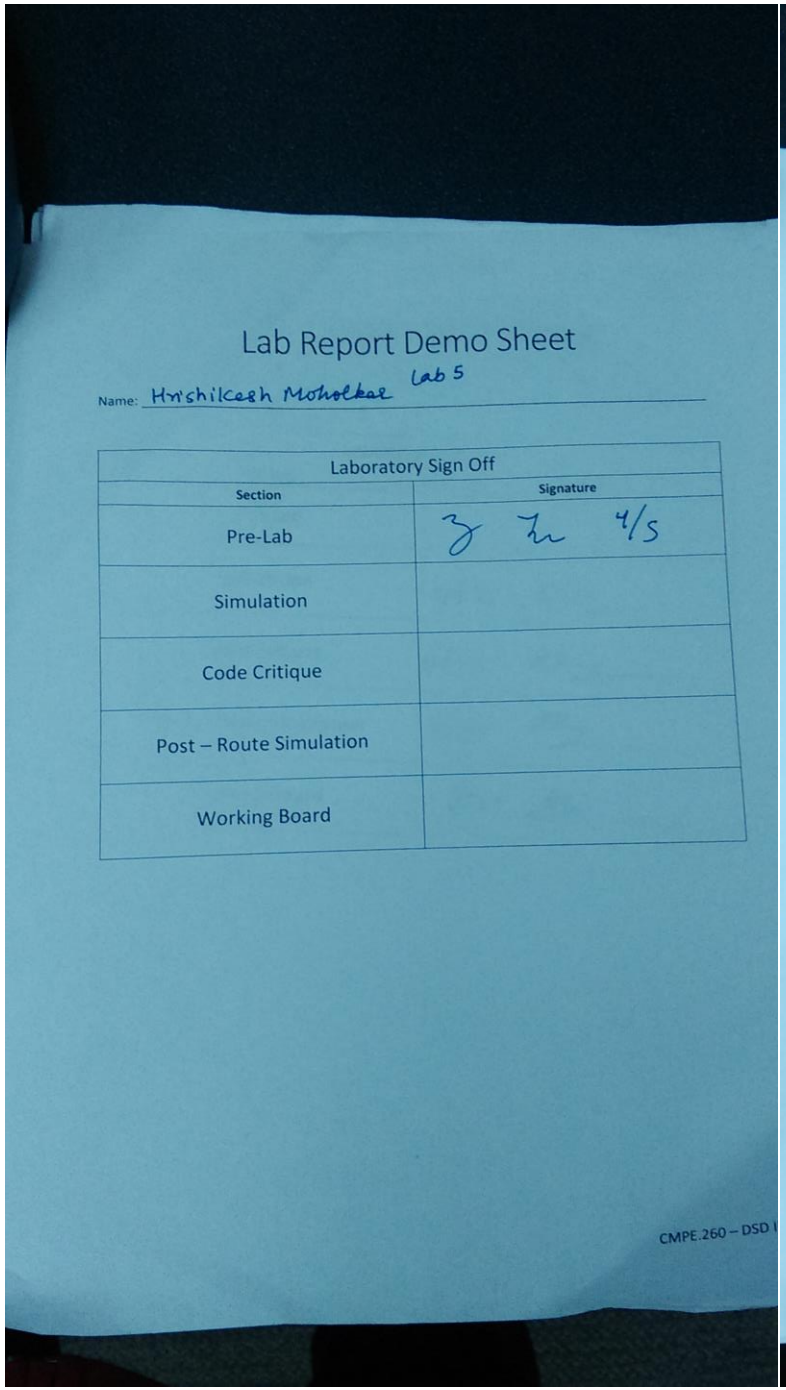
Timing result

The best case achievable for the BIST is 4.120 ns.

Conclusion

The BIST works as expected and the ip-core multiplier works perfectly. The MISR generates a signature whenever test mode is enabled. The output of the multiplier is the regular multiplication product whenever test mode is disabled. The BIST functions correctly when simulated on hardware. The overall lab is a success.

Submission:

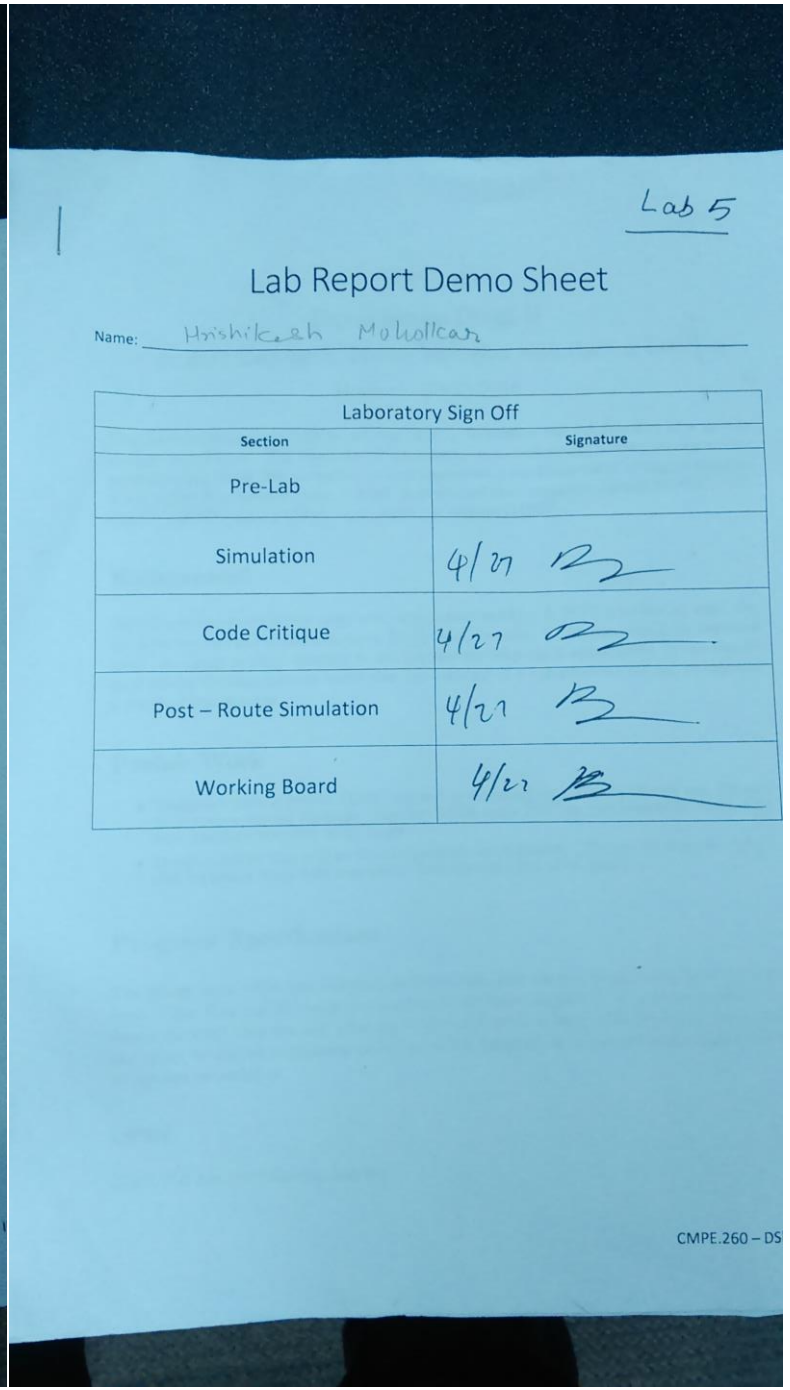


Lab Report Demo Sheet

Name: Hrishikesh Moholkar Lab 5

Laboratory Sign Off	
Section	Signature
Pre-Lab	4/5
Simulation	
Code Critique	
Post – Route Simulation	
Working Board	

CMPE.260 – DSD



Lab 5

Lab Report Demo Sheet

Name: Hrishikesh Moholkar

Laboratory Sign Off	
Section	Signature
Pre-Lab	
Simulation	4/27
Code Critique	4/27
Post – Route Simulation	4/27
Working Board	4/27

CMPE.260 – DS