ECE 3544: Digital Design I
Project 1 (Part B): Simulation in the ModelSim Environment; Continuous Assignment Models

Student Name: Sefunmi Ashiru

Honor Code Pledge: I have neither given nor received unauthorized assistance on this assignment.

Sefunmi Ashiru

Grading: The design project will be graded on a 100 point basis, as shown below:

Manner of Presentation (25 points)

_____ Completed cover sheet included with report (5 points)

_____ Organization: Clear, concise presentation of content; Use of appropriate, well-organized sections (10 points)

_____ Mechanics: Spelling and grammar (10 points)

Technical Merit (75 points)

_____ General discussion: *Did you describe the objectives in your own words? Did you address the questions posed in the project specification?* (10 points)

_____ Design discussion: *What was your approach to deriving the circuit you had to design? How does the design process that resulted in your continuous assignment model compare to the one you used to obtain a structural model that used primitive gates? How applicable is the design process you followed in this assignment to larger and more general designs?* (10 points)

_____ Testing discussion: *What was your approach to formulating your test benches, and how did you verify the correctness of your design and implementation?* (10 points)

_____ Conclusions: *Did you discuss the lessons you learned from the assignment? Did you discuss the applicability of your design process to larger and more general designs? Did you assess your implementation?* (10 points)

_____ Supporting figures (15 points total)
- *Correct waveforms showing simulation of both decoder modules.* (3 points)
- *Correct waveforms showing correct operation of your design.* (6 points)
- *Quartus reports and schematics* (6 points)

_____ Supporting files: *Do the submitted files produce the correct response?* (20 points)

_____ **Project Grade**

Purpose

In this project, I will be working with Modalism to run simulations. I will be simulating two pre-designed circuits. A 74138 3-to-8 decoder, which was written using a continuous assignment, and a 4-to-16 decoder that uses two of the 3-to-8 decoders. We must use a test bench to simulate each decoder and produce output waveforms to compare with its structural counterpart. We must then design a simple digital logic circuit that can be modelled and simulated using the Modalism tools. Through doing this we will be able to compare the performance and work of structural models to continuous assignment models, gain experience with timing models and test benches for simulating digital circuits in Verilog.

Answers to Noted Questions

- Comment on whether the change in the modelling technique affects the behaviour of the two decoders.

Both decoders function the same in the test bench even though they utilize different methods.

- Comment on whether the change in the modelling technique affects the behaviour of this model, as compared to the one you derived in Project 1A. b

Both Magnitude compares behave the same but use different methods to get to the result. However, I would have been likely for my structural module to have static hazards.

- Discuss the features of the RTL schematics that stand out to you.

The RTL seems to optimize the given Verilog for a more efficient hardware execution.

- Compare the resource usage of the two implementations.is very similar for both modules.

For the continuous Assignment and structural module, the resource usage is similar for both devices. The device required access to the input 12 pins as well as the 5 LEDs which output the result of the comparison

- What conclusions about the approach to modelling can you make from your comparison of the resource usage?

After Modelling and comparing both devices I can conclude that continuous assignment, while less efficient for transistor count, adds optimization and performance difference to design circuits compared to structural, while structural design can provide experienced users with more control over the circuit layout and connection at the risk of design hazards into your system.

Technical Approach

For my Design process utilized the continuous assignment prebuilt functions. Before reaching this decision, I considered optimizing gate count by making some output the negation of other developed gates, but this might lead to static 1 or 0 hazards. So, I used a unique predefined continuous assignment function that

represents each desired output's result mapping. At first, I felt like this was a bit of lazy designing on my part compared to the structural assignment used for project 1a, but then I realized by utilizing prebuilt functionality in Verilog I would be allowing the system more freedom to optimize in such a way that it would consider static hazards but also implement the design in a way that optimizes gate count. My Hypothesis was proven correct after I synthesized the design and showed the Continuous - RTL Viewer layout of the circuit.

Results

Testbench

The schematics from testbench can be found in the Appendix 1

When designing the testbench for the Comparator structural module I assigned the input registered for valA statically and set it to central value (4 =3'b0100) to ensure that the results should be out but all cases have all base cases (>,<, and ==) output. ValB was assigned dynamically using a for loop to test all outputs for a 3-bit value. It should be noted that the counter had to have 4 bits so that it could reach a value greater than a 3-bit number and trigger the for loop to stop.

Quartus Schematics

The schematics and details from Quartus can be found in Appendix 2

Structural

The structural schematic at first, I believed to be unchanged from by design, but it seems that the optimizer used an add operator labelled add0. In my design I used the functionality of one of my structural gates, under the assumption that a wire from its output would be connected in parallel with each other. This would have caused a static 0 or 1 but was averted as the optimizer/compiler translated it to 3 copies of the element. This allows for more regulated gate levels but increases the transistor count from my original design. I believe this optimization was due to me using continuous assignment primitive logic gates. If I had implemented my design using structural model functions with predefined wire connections.

Continuous

The continuous schematic appeared like what I hypothesized above. The optimizer/compiler to my design requirements for each continuous assignment operation and simplified the logic to reduce transistor count and redundancy, while also preventing static hazards from occurring. The schematic however only has one primitive logic gate while the other operating gates are relational. This hints at there being more complexing and gate operations under a layer of abstraction, making the continuous assignment, likely more expensive in transistor count but more reliable

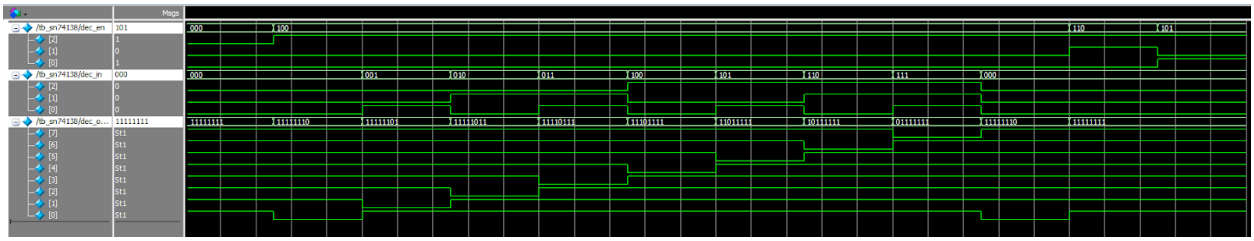
Conclusions

In Conclusion I feel that I have learned a lot about Verilog, structural & continuous design, as well as how to simulate and synthesize my designs. If I were to do this project again, I would attempt to write a bad design so that I can see the static 0 and 1 appear during simulation.

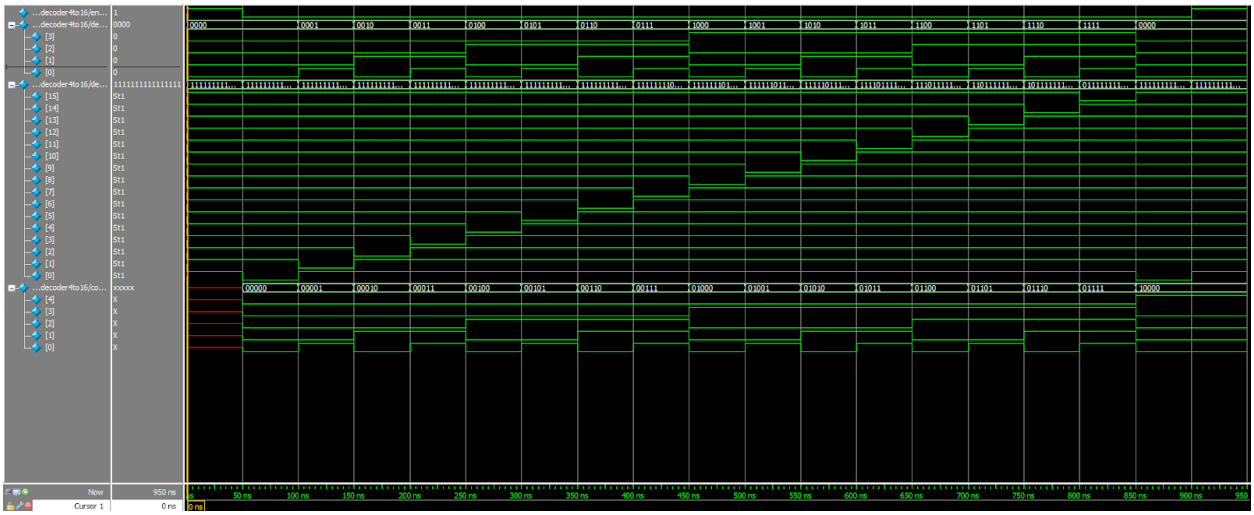
Appendices

Appendix 0 - Part 1A results

Simulating 74138 3-to-8 decoder Chip



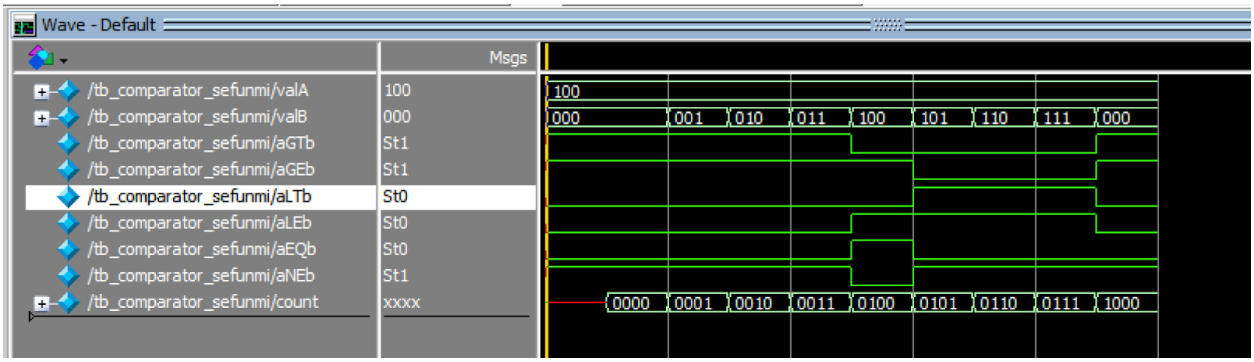
Simulating 4-to-16 decoder



Simulating Magnitude comparator

Comparator structural module key:

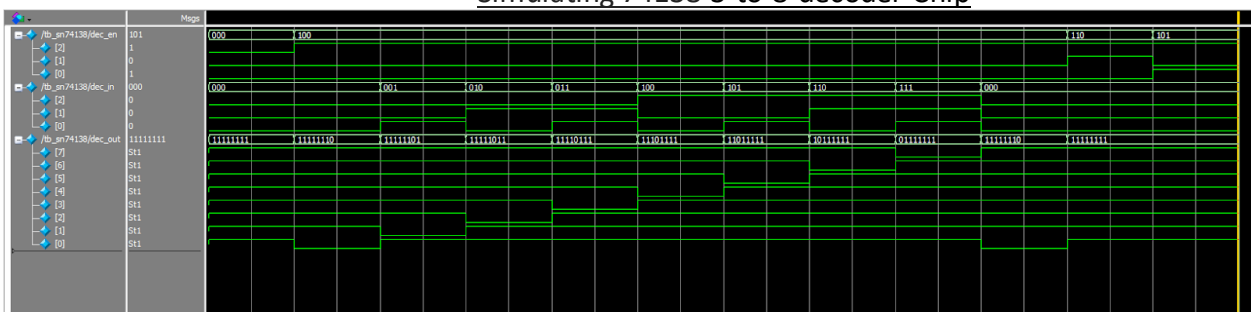
- aGTb: 1 if A is greater than B, 0 otherwise.
- aGEb: 1 if A is greater than or equal to B, 0 otherwise.
- aLTb: 1 if A is less than B, 0 otherwise.
- aLEb: 1 if A is less than or equal to B, 0 otherwise.
- aEQb: 1 if A is equal to B, 0 otherwise.
- aNEb: 1 if A is not equal to B, 0 otherwise.



Appendix 1.1 - Continuous assignment Results

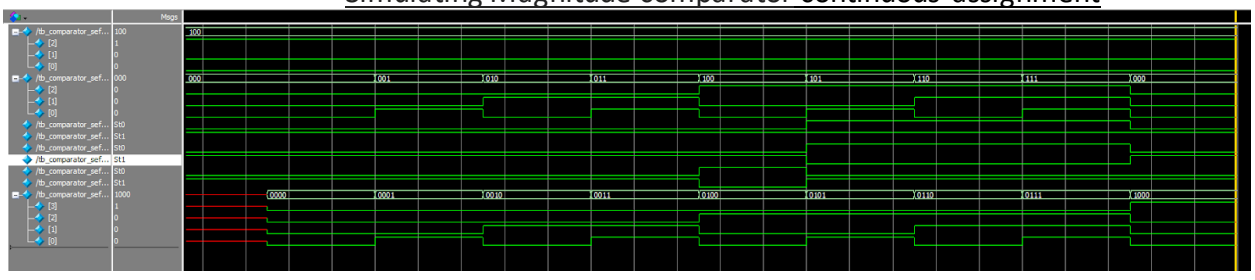
Simulating 4-to-16 decoder

Simulating 74138 3-to-8 decoder Chip



Appendix 1.2 - Continuous assignment Magnitude comparator Results

Simulating Magnitude comparator continuous assignment



- **A section that contains the reports and schematics from Quartus.**

Appendix 2 - Quartus Results

Appendix 2.1 - Structural - Synthesis and analysis

	Resource	Usage
1	Estimate of Logic utilization (ALMs needed)	0
2		
3	Combinational ALUT usage for logic	0
1	-- 7 input functions	0
2	-- 6 input functions	0
3	-- 5 input functions	0
4	-- 4 input functions	0
5	-- <=3 input functions	0
4		
5	Dedicated logic registers	0
6		
7	I/O pins	12
8		
9	Total DSP Blocks	0
10		
11	Maximum fan-out node	SW[0]~output
12	Maximum fan-out	1
13	Total fan-out	12
14	Average fan-out	0.50

Appendix 2.2 - Continuous - Synthesis and analysis

	Resource	Usage
1	Estimate of Logic utilization (ALMs needed)	0
2		
3	Combinational ALUT usage for logic	0
1	-- 7 input functions	0
2	-- 6 input functions	0
3	-- 5 input functions	0
4	-- 4 input functions	0
5	-- <=3 input functions	0
4		
5	Dedicated logic registers	0
6		
7	I/O pins	12
8		
9	Total DSP Blocks	0
10		
11	Maximum fan-out node	SW[0]~output
12	Maximum fan-out	1
13	Total fan-out	12
14	Average fan-out	0.50

Appendix 2.3 - Structural - Fitter

	Resource	Usage	%
1	Logic utilization (ALMs needed / total ALMs on device)	1 / 32,070	< 1 %
2	ALMs needed [=A-B+C]	1	
1	[A] ALMs used in final placement [=a+b+c+d]	1 / 32,070	< 1 %
1	[a] ALMs used for LUT logic and registers	0	
2	[b] ALMs used for LUT logic	1	
3	[c] ALMs used for registers	0	
4	[d] ALMs used for memory (up to half of total ALMs)	0	
2	[B] Estimate of ALMs recoverable by dense packing	0 / 32,070	0 %
3	[C] Estimate of ALMs unavailable [=a+b+c+d]	0 / 32,070	0 %
1	[a] Due to location constrained logic	0	
2	[b] Due to LAB-wide signal conflicts	0	
3	[c] Due to LAB input limits	0	
4	[d] Due to virtual I/Os	0	
3			
4	Difficulty packing design	Low	
5			
6	Total LABs: partially or completely used	1 / 3,207	< 1 %
1	-- Logic LABs	1	
2	-- Memory LABs (up to half of total LABs)	0	
7			
8	Combinational ALUT usage for logic	1	
1	-- 7 input functions	0	
2	-- 6 input functions	0	
3	-- 5 input functions	0	
4	-- 4 input functions	0	
5	-- <=3 input functions	1	
9	Combinational ALUT usage for route-throughs	0	
10			

11	Dedicated logic registers	0	
1	-- By type:		
1	-- Primary logic registers	0 / 64,140	0 %
2	-- Secondary logic registers	0 / 64,140	0 %
2	-- By function:		
1	-- Design implementation registers	0	
2	-- Routing optimization registers	0	
12			
13	Virtual pins	0	
14	I/O pins	12 / 457	3 %
1	-- Clock pins	0 / 8	0 %
2	-- Dedicated input pins	0 / 21	0 %
15			
16	Hard processor system peripheral utilization		
1	-- Boot from FPGA	0 / 1 (0 %)	
2	-- Clock resets	0 / 1 (0 %)	
3	-- Cross trigger	0 / 1 (0 %)	
4	-- S2F AXI	0 / 1 (0 %)	
5	-- F2S AXI	0 / 1 (0 %)	
6	-- AXI Lightweight	0 / 1 (0 %)	
7	-- SDRAM	0 / 1 (0 %)	
8	-- Interrupts	0 / 1 (0 %)	
9	-- JTAG	0 / 1 (0 %)	
10	-- Loan I/O	0 / 1 (0 %)	
11	-- MPU event standby	0 / 1 (0 %)	
12	-- MPU general purpose	0 / 1 (0 %)	
13	-- STM event	0 / 1 (0 %)	
14	-- TPIU trace	0 / 1 (0 %)	
15	-- DMA	0 / 1 (0 %)	

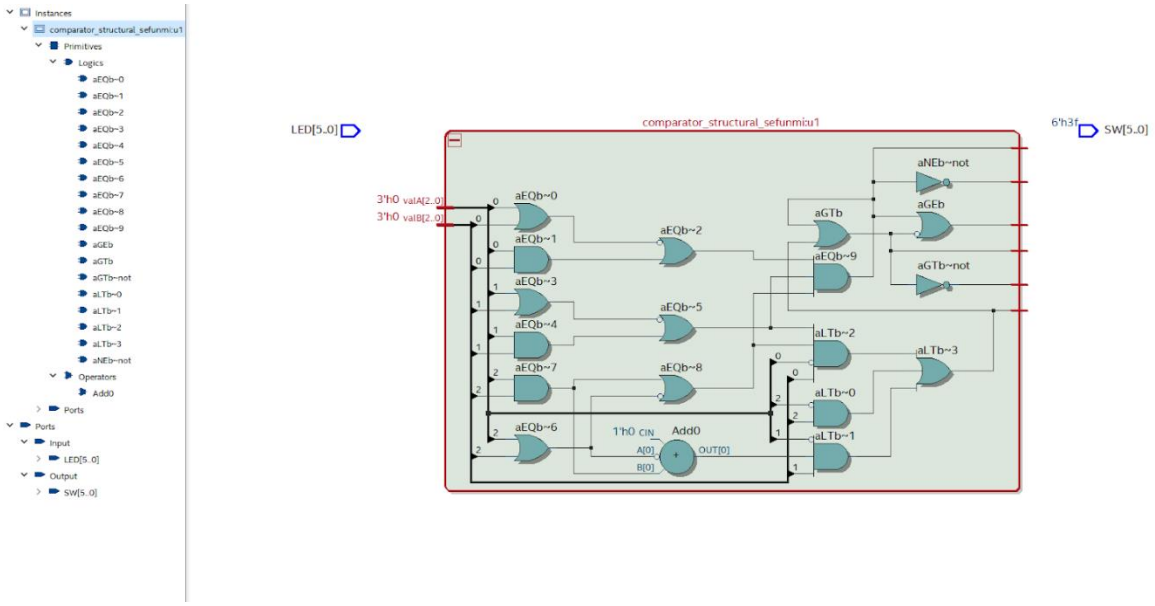
16	-- CAN	0 / 2 (0 %)	
17	-- EMAC	0 / 2 (0 %)	
18	-- I2C	0 / 4 (0 %)	
19	-- NAND Flash	0 / 1 (0 %)	
20	-- QSPI	0 / 1 (0 %)	
21	-- SDMMC	0 / 1 (0 %)	
22	-- SPI Master	0 / 2 (0 %)	
23	-- SPI Slave	0 / 2 (0 %)	
24	-- UART	0 / 2 (0 %)	
25	-- USB	0 / 2 (0 %)	
17			
18	M10K blocks	0 / 397	0 %
19	Total MLAB memory bits	0	
20	Total block memory bits	0 / 4,065,280	0 %
21	Total block memory implementation bits	0 / 4,065,280	0 %
22			
23	Total DSP Blocks	0 / 87	0 %
24			
25	Fractional PLLs	0 / 6	0 %
26	Global signals	0	
1	-- Global clocks	0 / 16	0 %
2	-- Quadrant clocks	0 / 66	0 %
3	-- Horizontal periphery clocks	0 / 18	0 %
27	SERDES Transmitters	0 / 100	0 %
28	SERDES Receivers	0 / 100	0 %
29	JTAGs	0 / 1	0 %
30	ASMI blocks	0 / 1	0 %
31	CRC blocks	0 / 1	0 %
32	Remote update blocks	0 / 1	0 %
33	Oscillator blocks	0 / 1	0 %

34	Impedance control blocks	0 / 4	0 %
35	Hard Memory Controllers	0 / 2	0 %
36	Average interconnect usage (total/H/V)	0.0% / 0.0% / 0.0%	
37	Peak interconnect usage (total/H/V)	0.0% / 0.0% / 0.0%	
38	Maximum fan-out	1	
39	Highest non-global fan-out	1	
40	Total fan-out	12	
41	Average fan-out	0.48	

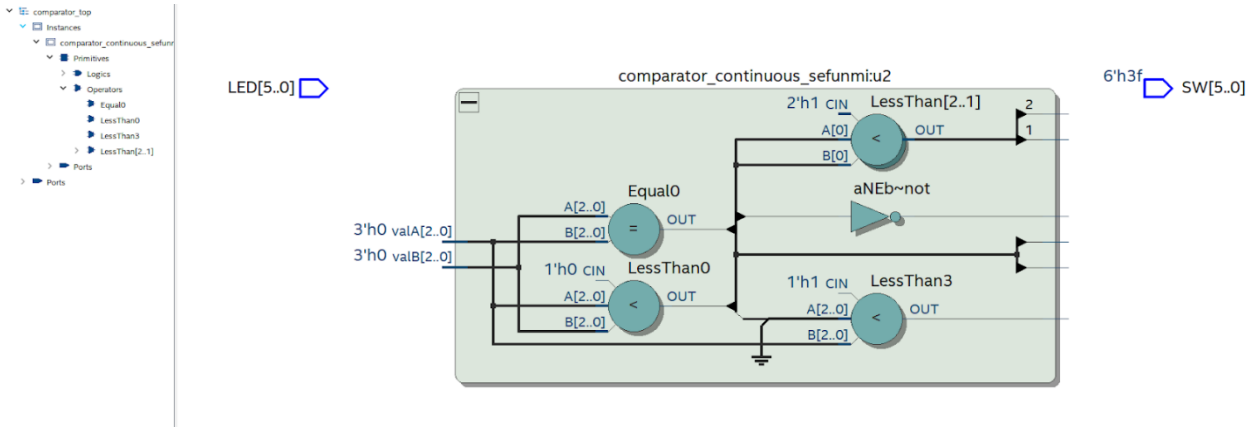
	Resource	Usage	%
1	Logic utilization (ALMs needed / total ALMs on device)	1 / 32,070	< 1 %
2	ALMs needed [=A-B+C]	1	
1	[A] ALMs used in final placement [=a+b+c+d]	1 / 32,070	< 1 %
1	[a] ALMs used for LUT logic and registers	0	
2	[b] ALMs used for LUT logic	1	
3	[c] ALMs used for registers	0	
4	[d] ALMs used for memory (up to half of total ALMs)	0	
2	[B] Estimate of ALMs recoverable by dense packing	0 / 32,070	0 %
3	[C] Estimate of ALMs unavailable [=a+b+c+d]	0 / 32,070	0 %
1	[a] Due to location constrained logic	0	
2	[b] Due to LAB-wide signal conflicts	0	
3	[c] Due to LAB input limits	0	
4	[d] Due to virtual I/Os	0	
3			
4	Difficulty packing design	Low	
5			
6	Total LABs: partially or completely used	1 / 3,207	< 1 %
1	-- Logic LABs	1	
2	-- Memory LABs (up to half of total LABs)	0	
7			
8	Combinational ALUT usage for logic	1	
1	-- 7 input functions	0	
2	-- 6 input functions	0	
3	-- 5 input functions	0	
4	-- 4 input functions	0	
5	-- <=3 input functions	1	
9	Combinational ALUT usage for route-throughs	0	
10			
11	Dedicated logic registers	0	
1	-- By type:		

11	Dedicated logic registers	0	
1	-- By type:		
1	-- Primary logic registers	0 / 64,140	0 %
2	-- Secondary logic registers	0 / 64,140	0 %
2	-- By function:		
1	-- Design implementation registers	0	
2	-- Routing optimization registers	0	
12			
13	Virtual pins	0	
14	I/O pins	12 / 457	3 %
1	-- Clock pins	0 / 8	0 %
2	-- Dedicated input pins	0 / 21	0 %
15			
16	Hard processor system peripheral utilization		
1	-- Boot from FPGA	0 / 1 (0 %)	
2	-- Clock resets	0 / 1 (0 %)	
3	-- Cross trigger	0 / 1 (0 %)	
4	-- S2F AXI	0 / 1 (0 %)	
5	-- F2S AXI	0 / 1 (0 %)	
6	-- AXI Lightweight	0 / 1 (0 %)	
7	-- SDRAM	0 / 1 (0 %)	
8	-- Interrupts	0 / 1 (0 %)	
9	-- JTAG	0 / 1 (0 %)	
10	-- Loan I/O	0 / 1 (0 %)	
11	-- MPU event standby	0 / 1 (0 %)	
12	-- MPU general purpose	0 / 1 (0 %)	
13	-- STM event	0 / 1 (0 %)	
14	-- TPIU trace	0 / 1 (0 %)	
15	-- DMA	0 / 1 (0 %)	

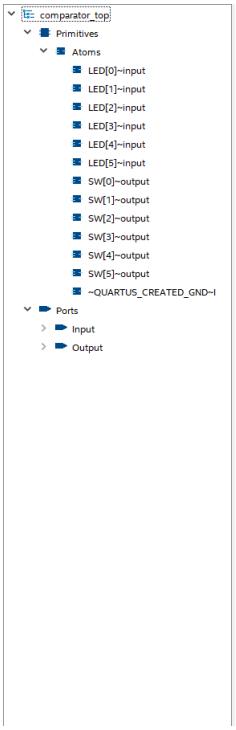
16	-- CAN	0 / 2 (0 %)	
17	-- EMAC	0 / 2 (0 %)	
18	-- I2C	0 / 4 (0 %)	
19	-- NAND Flash	0 / 1 (0 %)	
20	-- QSPI	0 / 1 (0 %)	
21	-- SDMMC	0 / 1 (0 %)	
22	-- SPI Master	0 / 2 (0 %)	
23	-- SPI Slave	0 / 2 (0 %)	
24	-- UART	0 / 2 (0 %)	
25	-- USB	0 / 2 (0 %)	
17			
18	M10K blocks	0 / 397	0 %
19	Total MLAB memory bits	0	
20	Total block memory bits	0 / 4,065,280	0 %
21	Total block memory implementation bits	0 / 4,065,280	0 %
22			
23	Total DSP Blocks	0 / 87	0 %
24			
25	Fractional PLLs	0 / 6	0 %
26	Global signals	0	
1	-- Global clocks	0 / 16	0 %
2	-- Quadrant clocks	0 / 66	0 %
3	-- Horizontal periphery clocks	0 / 18	0 %
27	SERDES Transmitters	0 / 100	0 %
28	SERDES Receivers	0 / 100	0 %
29	JTAGs	0 / 1	0 %
30	ASMI blocks	0 / 1	0 %
31	CRC blocks	0 / 1	0 %
32	Remote update blocks	0 / 1	0 %
33	Oscillator blocks	0 / 1	0 %
34	Impedance control blocks	0 / 4	0 %
35	Hard Memory Controllers	0 / 2	0 %
36	Average interconnect usage (total/H/V)	0.0% / 0.0% / 0.0%	
37	Peak interconnect usage (total/H/V)	0.0% / 0.0% / 0.0%	
38	Maximum fan-out	1	
39	Highest non-global fan-out	1	
40	Total fan-out	12	
41	Average fan-out	0.48	

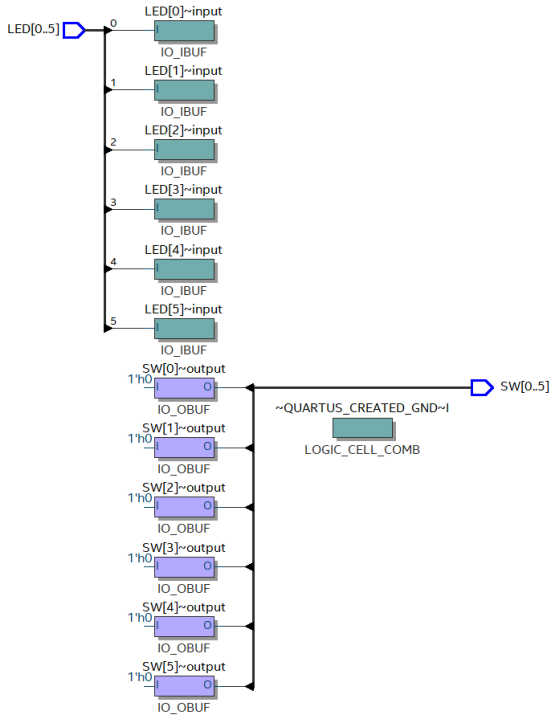


Appendix 2.6 - Continuous - RTL Viewer



Appendix 2.7 - Structural - Technology Map Viewer





Appendix 2.8 - Continuous - Technology Map Viewer