EE/CSE 371
October 13, 2025
Lab 2 Report

Design Procedure

This lab designs and tests three memory modules and controls them via a virtual FPGA. The goal of this lab was to better understand how memory is created via a chip built M10K RAM 32x3 memory block and accessed inside an FPGA. The ram stores 32 words each of 3 bits. Task 1 of the lab uses the built in IP catalog to generate a pre-made RAM. Task 2 creates a custom memory system using arrays. Task 3 incorporates the design from Task 2 alongside a dual port ram, creating a memory system that allows reading and writing to different memories by controlling the FPGA switches and keys. Task 3 also initializes memory using a Memory Initialization File.

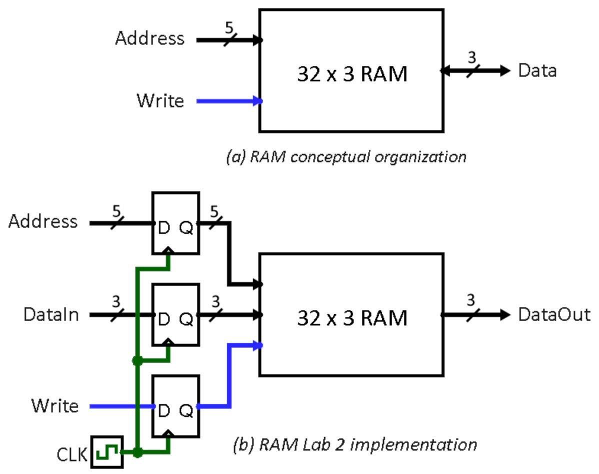


Figure 1: Diagram of 32x3 RAM with one and two ports designed in Tasks 2 and 3.

Overall System and Block Diagram

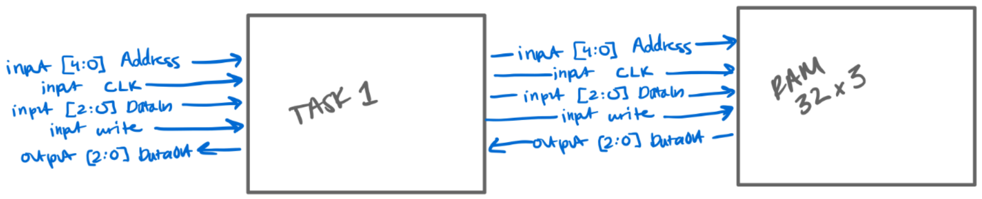


Figure 2: Diagram of Task1

Task 1

Task1.svwe created a M10K 32x3 single port RAM through the IP catalog in Quartus, matching the diagram provided. The module connects the external inputs: 5bit address (Address), 3bit data input (DataIn), clock (CLK), and write enable (Write) to RAM ports. The output from the memory (q) is connected to the 3-bit data output signal (DataOut). This design reflects the structure shown in the lab’s Figure 1.

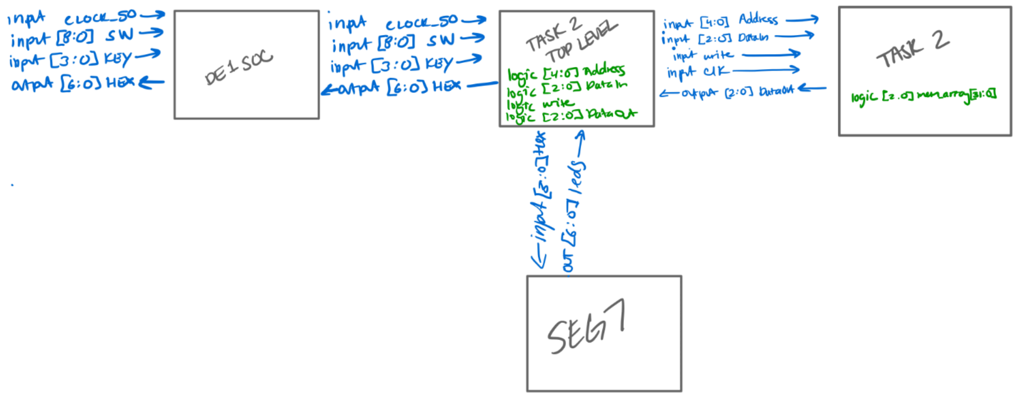


Figure 3: Diagram of Task 2

Task 2

Instead of using a prebuilt 32x3 RAM from the catalog, Task 2 uses a multidimensional array logic [2:0] memory_array [31:0], a 32 element array with each element 3 bits. The Task 2 module takes inputs for address, data, write enable, and clock, along with an output for the data read from memory. When write enable (Write) is high, the data input (DataIn) is stored at the given address on the rising edge of the clock. Otherwise, the output (DataOut) reads the value previously stored at the address.

A top level module, task2_toplevel, connects the task2 memory module to the DE1 switches, HEXs and keys. Switches SW8 to SW4 function as address selectors for the user, SW3 through SW1 serve as data input. SW0 serves as write enable and KEY0 is used as a clock signal. Memory addresses are displayed on HEX5–HEX4, input data on HEX1, and the output data on HEX0 utilizing the given 7 segment display module.

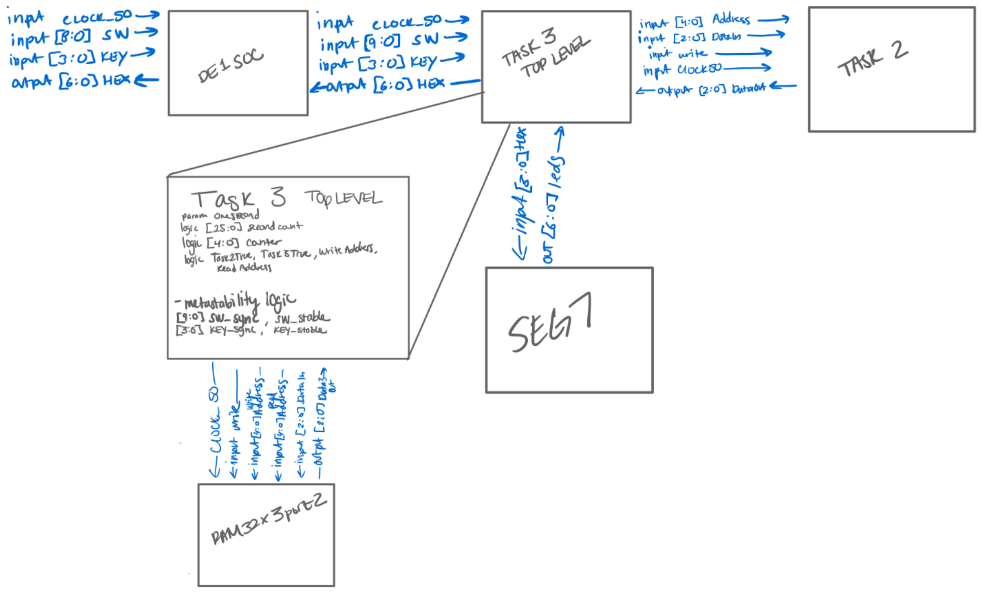


Figure 4: Diagram of Task 3.

Task 3

Task 3 incorporates an IP catalog designed M10K 32x3 dual port RAM with separate addresses for reading and writing alongside the single-port RAM from Task 2. A Memory Initialization File, ram32x3.mif, is created and initialized with integer values. A new top level module, task3_toplevel includes Task 2 single port logic alongside the logic for the dual port RAM. A new switch, SW9, selects between the two memory modules for operation and chooses one. When SW9 = 0, the design utilizes task 2 memory, writing and reading data from the single-port RAM. When SW9= 1, it utilizes Task 3 memory where the dual-port RAM (ram32x3port2) allows simultaneous read and write using separate address ports. The output data between the two modules (DataTask2Out or DataTask3Out) is selected using a MUX and displayed on a seven segment display.

A basic counter cycles through the various addresses of the separate modules, displayed on HEX 2 & 3, and displays the data stored at those locations on HEX0. The counter uses a custom made clock divider to slow down counting: a 26bit register (secondCount) increments at every clock edge, and when it reaches the set value (ONE_SECOND = 100), it resets to zero and increments the counter by one. This makes the counter increase much slower, 1HZ, than the actual clock and allows for address output to be displayed every second. When reset is high, both values go to zero. Write enable, address and data input are controlled using the same switches as task 2, KEY 3 is used as a reset. A synchronous 50 MHz clock was used for both memories and replaced KEY[0] logic from Task2.

To ensure stable input signals, all asynchronous switch and key inputs are passed through a two stage synchronizer (SW_Sync) followed by a stability register (SW_Stable). The double flip-flop synchronization protects against metastability and ensures asynchronous transitions from mechanical inputs settle before entering the system.

Results and Testing

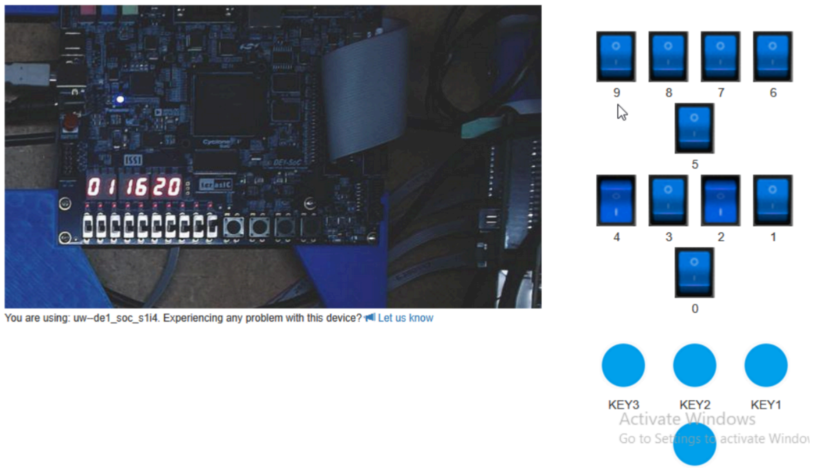

You are using: uw--de1_soc_s1i4. Experiencing any problem with this device? Let us know

Figure 4: Task 3 deployed on LabsLand FPGA

Task 1:
Task 1 test bench verifies memory is properly read and written, 3 bit values are stored into specific addresses and then later read from the memory location to check if it updated properly. Write enable is turned off to make sure writing does not occur without the appropriate enable signal by confirming the q output of selected addresses. This confirms the memory module works as expected. A clock is generated with a 10 ns period, and test sequences are applied. First, the write operation writes the value 4 to address 1, then disables the write signal. Next the read operation reads from address 1 to confirm the stored data. Next a write disabled test tries to change the data at address 1 while Write = 0 to confirm memory contents remain unchanged. Similarly, the value 5 writes to address 2, write is disabled, and then the value is confirmed to verify correct operation. This test bench confirms task 1 correctly stores and outputs data according to the write enable and address signals.

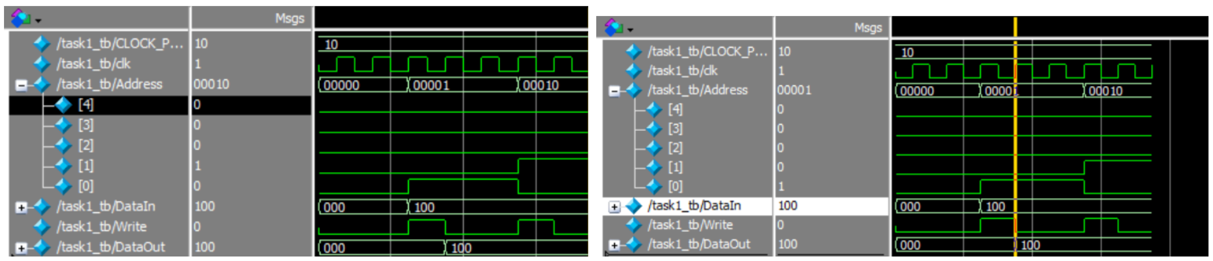
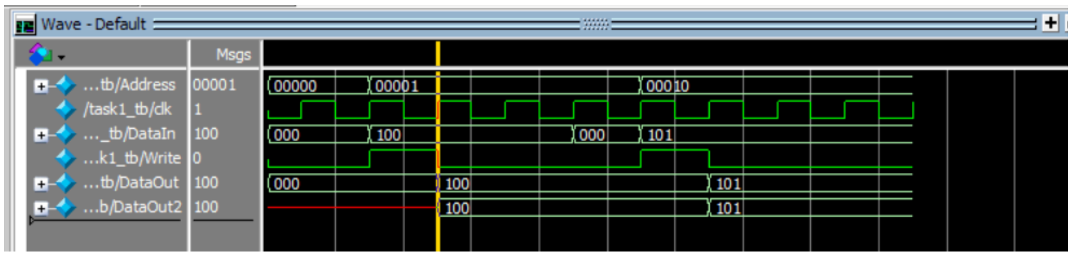


Figure 5: ModelSim Waveforms Task 1

Task2:
The test bench from Task is updated and used to verify that reading and writing works the same way. The simulation confirms that the memory behaves correctly and that data is stored and retrieved from the right addresses. The design was tested on LabsLand and confirmed that data could be written to any address, then after we turn off the write switch we can read the same data back correctly. The updated test bench (task1_tb) verifies that the RAM correctly performs read and write operations. It instantiates both the Task 1 (library-based) and Task 2 (array-based) RAM modules for comparison.



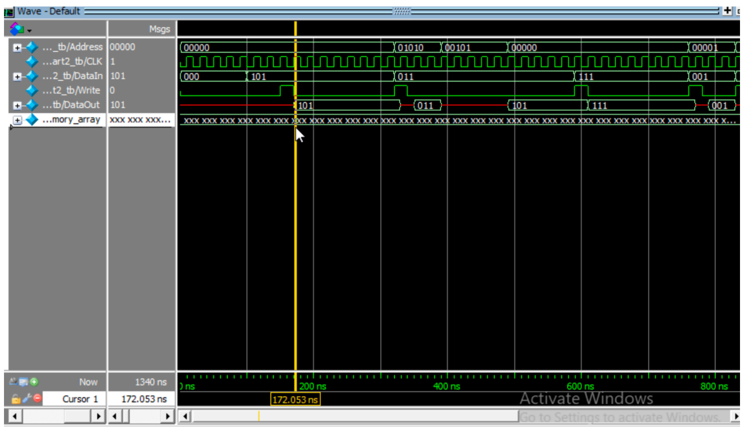


Figure 6: ModelSim Waveforms Task 6

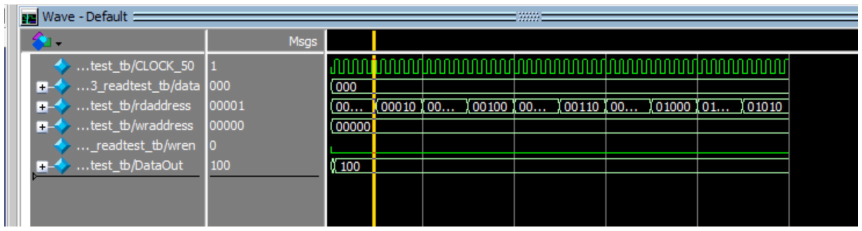
Task3:

In the counter testbench, counter_tb, a 50 MHz clock is generated by toggling the clock signal every 10 ns. The simulation holds reset for one clock cycle, then releases and lets the counter run for 50 clock cycles before stopping, therefore verifying that the counter increments correctly.

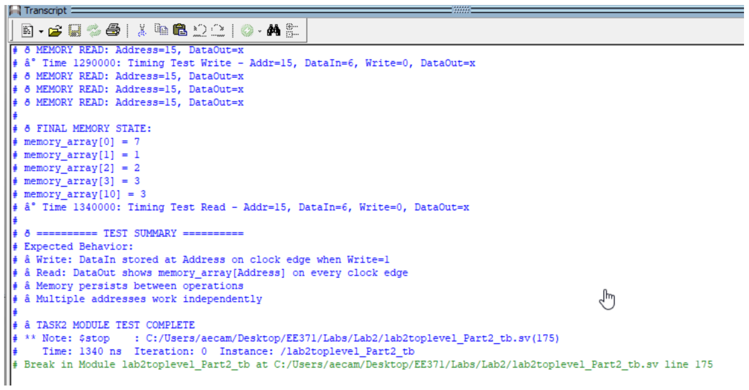
The task3_readtest_tb verifies the read operations from the Memory Initialization File. It generates a continuous 50 MHz clock and does not perform any writes (wren = 0), only read functionality is tested. It sequentially reads from memory addresses 1 through 10, waits between reads, and prints address's output data to the console. We first wanted to confirm the RAM correctly loading from the MIF file when selected for reading.

The lab2toplevel_task3_tb testbench simulates the top-level design by toggling switch and key inputs to verify Task 2 and Task 3 behavior. It starts by resetting the system, then tests Task 2 by writing and reading data combinations to confirm correct memory and display functionality. Then it switches to Task 3 mode by toggling switch 9, and checks the counter operates and checks for the correct HEX outputs. Finally, it switches back to Task 2 to confirm the transitions.

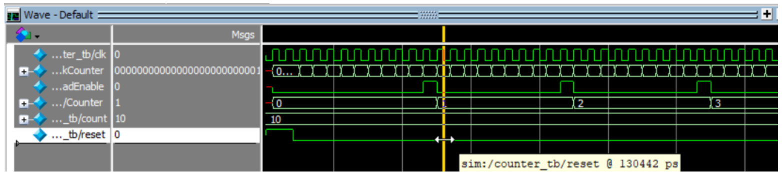
After deploying the module on the FPGA, we verified the dual-port RAM started with the values from the MIF file and that writing new data worked correctly. Switching between the two memories using SW9 confirmed that they were completely independent. Overall the testbenches were kept pretty light and simple, this was due to the difficulty reading data in ModelSim described in the experience report. Display statements were to have a more accurate understanding of module behavior than the waveform alone. In parallel, deployed testing was done on FPGA via LabsLand to further confirm proper functionality.



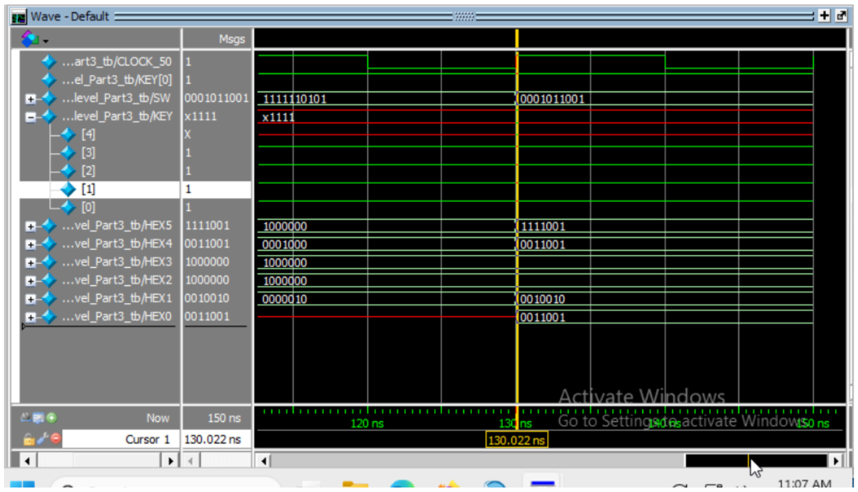
```
Wave - Default
                         Msgs
...test_tb/CLOCK_50      1
...3_readtest_tb/data    000
...test_tb/rdaddress     00001
...test_tb/wraddress     00000
..._readtest_tb/wren     0
...test_tb/DataOut       100
```



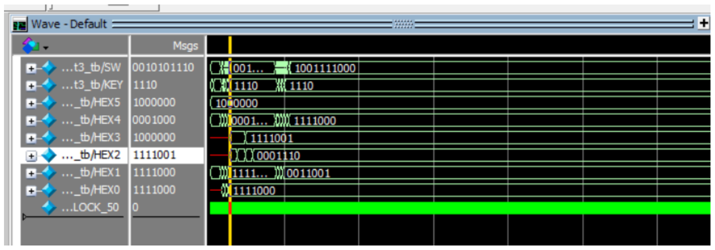
```
# ð MEMORY READ: Address=15, DataOut=x
# â° Time 1290000: Timing Test Write - Addr=15, DataIn=6, Write=0, DataOut=x
# ð MEMORY READ: Address=15, DataOut=x
# ð MEMORY READ: Address=15, DataOut=x
# ð MEMORY READ: Address=15, DataOut=x
#
# ð FINAL MEMORY STATE:
# memory_array[0] = 7
# memory_array[1] = 1
# memory_array[2] = 2
# memory_array[3] = 3
# memory_array[10] = 3
# â° Time 1340000: Timing Test Read - Addr=15, DataIn=6, Write=0, DataOut=x
#
# ð ========== TEST SUMMARY ==========
# Expected Behavior:
# â Write: DataIn stored at Address on clock edge when Write=1
# â Read: DataOut shows memory_array[Address] on every clock edge
# â Memory persists between operations
# â Multiple addresses work independently
#
# â TASK2 MODULE TEST COMPLETE
# ** Note: $stop    : C:/Users/aecam/Desktop/EE371/Labs/Lab2/lab2toplevel_Part2_tb.sv(175)
#    Time: 1340 ns  Iteration: 0  Instance: /lab2toplevel_Part2_tb
# Break in Module lab2toplevel_Part2_tb at C:/Users/aecam/Desktop/EE371/Labs/Lab2/lab2toplevel_Part2_tb.sv line 175
```



```
Wave - Default
                  Msgs
...ter_tb/clk     0
...kCounter       00000000000000000000000001
...adEnable       0
.../Counter       1
..._tb/count      10
..._tb/reset      0
sim:/counter_tb/reset @ 130442 ps
```



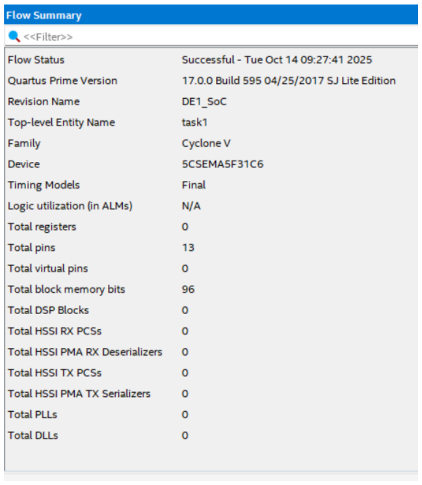
```
Wave - Default
                          Msgs
...art3_tb/CLOCK_50       1
...el_Part3_tb/KEY[0]     1
...level_Part3_tb/SW      0001011001
...level_Part3_tb/KEY     x1111
   [4]                    X
   [3]                    1
   [2]                    1
   [1]                    1
   [0]                    1
...vel_Part3_tb/HEX5      1111001
...vel_Part3_tb/HEX4      0011001
...vel_Part3_tb/HEX3      1000000
...vel_Part3_tb/HEX2      1000000
...vel_Part3_tb/HEX1      0010010
...vel_Part3_tb/HEX0      0011001
Now                       150 ns
Cursor 1                  130.022 ns
```



```
Wave - Default
                  Msgs
...t3_tb/SW       0010101110
...t3_tb/KEY      1110
..._tb/HEX5       1000000
..._tb/HEX4       0001000
..._tb/HEX3       1000000
..._tb/HEX2       1111001
..._tb/HEX1       1111000
..._tb/HEX0       1111000
...LOCK_50        0
```

Figure 7: ModelSim Waveforms and Display for Task 3

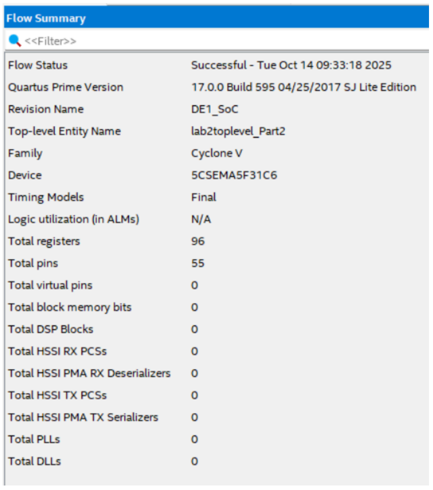
Flow Summary



Flow Summary

<<Filter>>

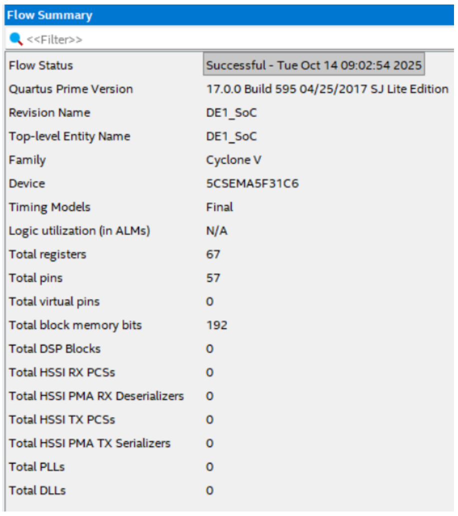
Flow Status	Successful - Tue Oct 14 09:27:41 2025
Quartus Prime Version	17.0.0 Build 595 04/25/2017 SJ Lite Edition
Revision Name	DE1_SoC
Top-level Entity Name	task1
Family	Cyclone V
Device	5CSEMA5F31C6
Timing Models	Final
Logic utilization (in ALMs)	N/A
Total registers	0
Total pins	13
Total virtual pins	0
Total block memory bits	96
Total DSP Blocks	0
Total HSSI RX PCSs	0
Total HSSI PMA RX Deserializers	0
Total HSSI TX PCSs	0
Total HSSI PMA TX Serializers	0
Total PLLs	0
Total DLLs	0



Flow Summary

<<Filter>>

Flow Status	Successful - Tue Oct 14 09:33:18 2025
Quartus Prime Version	17.0.0 Build 595 04/25/2017 SJ Lite Edition
Revision Name	DE1_SoC
Top-level Entity Name	lab2toplevel_Part2
Family	Cyclone V
Device	5CSEMA5F31C6
Timing Models	Final
Logic utilization (in ALMs)	N/A
Total registers	96
Total pins	55
Total virtual pins	0
Total block memory bits	0
Total DSP Blocks	0
Total HSSI RX PCSs	0
Total HSSI PMA RX Deserializers	0
Total HSSI TX PCSs	0
Total HSSI PMA TX Serializers	0
Total PLLs	0
Total DLLs	0



Flow Summary

<<Filter>>

Flow Status	Successful - Tue Oct 14 09:02:54 2025
Quartus Prime Version	17.0.0 Build 595 04/25/2017 SJ Lite Edition
Revision Name	DE1_SoC
Top-level Entity Name	DE1_SoC
Family	Cyclone V
Device	5CSEMA5F31C6
Timing Models	Final
Logic utilization (in ALMs)	N/A
Total registers	67
Total pins	57
Total virtual pins	0
Total block memory bits	192
Total DSP Blocks	0
Total HSSI RX PCSs	0
Total HSSI PMA RX Deserializers	0
Total HSSI TX PCSs	0
Total HSSI PMA TX Serializers	0
Total PLLs	0
Total DLLs	0

Flow Summaries for Task 1, 2, and 3