

CS 342000 | CS343000
Instructor: Professor Izidor Gertner
Spring 2022
Azwad Shameem, 4/10/2022
Laboratory Project: Design Arithmetic Logic Unit

Please hand write and sign statements affirming that you will not cheat here and in your submission:

"I will neither give nor receive unauthorized assistance on this LAB. I will use only one computing device to perform this LAB".

I will neither give nor receive unauthorized assistance on this exam. I will only use one computing device to perform this test

Azwad Shameem

Table of Contents

Objective.....	3
Description of Functionality and Simulation	4
Functionality:.....	4
RS Register.....	4
RT Register	5
RD Register.....	6
N-Bit AddSub with Flags Unit.....	7
Bitwise Operations Unit.....	8
IMM16 Register	9
Sign Extender Unit	10
MAR – Memory Address Register.....	11
MDR – Memory Data Register	12
ALU – Arithmetic Logic Unit.....	13
3 Ported RAM	18
Instruction Register.....	23
Data Memory.....	28
Simulation:.....	30
Part I.A	30
Part I.B	44
Part II.....	50
Part III.....	58
Part IV	60
Conclusion:.....	62

Objective

The objective of the Arithmetic Logic Unit lab was to implement MIPS instructions in VHDL. MIPS instructions such as ADD, ADDU, SUB, SUBU, AND, NOR, OR, SLL, SRA, ADDI, ADDIU, ANDI, ORI, LW, SW was all intended to be implemented by VHDL programming. The VHDL program was then to be tested by utilizing ModelSim simulations to create waveforms for select cases that would show the results of the MIPS instructions that were implemented in VHDL. Furthermore, in order to ascertain correctness of the VHDL implementation of MIPS instructions we had to compare the MIPS instructions that ran on the MARS environment with the ModelSim simulation waveforms to confirm the correctness of the VHDL implementation of the MIPS instructions.

Description of Functionality and Simulation

Functionality:

RS Register

The screenshot shows the Quartus Prime software interface. The top menu bar includes File, Edit, View, Project, Assignments, Processing, Tools, Window, and Help. The main window displays the VHDL code for the RS Register. The code defines an entity and an architecture. The entity has generic parameters for content and result width, and a port with a clock input, a write enable input, and two outputs: memory and result. The architecture contains two processes: P1 handles writes, and P2 handles reads. Both processes update the memory and calculate the result based on the current memory value and the write operation. The compilation report at the bottom indicates a successful compilation with 0 errors and 81 warnings.

```

library IEEE;
use IEEE.std_logic_1164.all;

entity Shameem_04_10_22_RS is
  generic (Shameem_04_10_22_N: integer := 32);
  port (
    Shameem_04_10_22_clk, Shameem_04_10_22_wren, Shameem_04_10_22_rden, Shameem_04_10_22_chen: in STD_LOGIC;
    Shameem_04_10_22_content: in STD_LOGIC_VECTOR(Shameem_04_10_22_N-1 downto 0);
    Shameem_04_10_22_result: out STD_LOGIC_VECTOR(Shameem_04_10_22_N-1 downto 0)
  );
end Shameem_04_10_22_RS;

architecture arch of Shameem_04_10_22_RS is
begin
  Shameem_04_10_22_P1: process(Shameem_04_10_22_clk, Shameem_04_10_22_wren)
  begin
    if(rising edge(Shameem_04_10_22_clk) AND Shameem_04_10_22_wren = '1')
      then Shameem_04_10_22_memory <= Shameem_04_10_22_content;
    end if;
  end process Shameem_04_10_22_P1;
  Shameem_04_10_22_P2: process(Shameem_04_10_22_rden, Shameem_04_10_22_chen, Shameem_04_10_22_memory)
  begin
    if(Shameem_04_10_22_rden = '1' AND Shameem_04_10_22_chen = '1')
      then Shameem_04_10_22_result <= Shameem_04_10_22_memory;
    elsif(Shameem_04_10_22_chen = '0')
      then Shameem_04_10_22_result <= (others => '0');
    end if;
  end process Shameem_04_10_22_P2;
end arch;

```

Compilation Report:

- Command: quartus_edt --read_settings_files=off --write_settings_files=off Shameem_04_10_22_ALU -c Shameem_04_10_22_ALU
- 18236 Number of processors has not been specified which may cause overloading on shared machines. Set the global assignment NUM_PARALLEL_PROCESSORS in your QSF to an appropriate value for best performance.
- 204019 Generated file Shameem_04_10_22_ALU.vo in folder "/home/azwad/Desktop/Shameem_04_10_22_ALU/simulation/modelsim/" for EDA simulation tool
- Quartus Prime EDA Netlist Writer was successful. 0 errors, 1 warning
- 293000 Quartus Prime Full Compilation was successful. 0 errors, 81 warnings

Figure 1: RS Register VHDL code compiled successfully in Quartus.

This screenshot shows the Quartus Prime software interface with the same project open. The main window displays the flow summary of the compilation process. The flow summary table provides detailed statistics for the compilation, including flow settings, timing models, and resource utilization. The compilation report at the bottom is identical to Figure 1, showing a successful compilation with 0 errors and 81 warnings.

Flow	Summary	Value
Flow Settings	Successful - Sat Apr 2 14:46:30 2022	
Flow Non-Default Global Settings	20.1.1 Build 720 11/11/2020 SJ Lite Edition	
Flow Elapsed Time	Shameem_04_10_22_ALU	
Flow OS Summary	Revision Name	
Flow Log	Shameem_04_10_22_RS	
Analysis & Synthesis	Family	Cyclone V
Fitter	Device	5CGXF7C7F23C8
Assembler	Timing Models	Final
Timing Analyzer	Logic utilization (in ALMs)	33 / 56,480 (< 1 %)
EDA Netlist Writer	Total registers	32
Flow Messages	Total pins	68 / 268 (25 %)
Flow Suppressed Messages	Total virtual pins	0
	Total block memory bits	0 / 7024,640 (0 %)
	Total DSP Blocks	0 / 156 (0 %)
	Total HSI RX PCSS	0 / 6 (0 %)
	Total HSI PMA RX Deserializers	0 / 6 (0 %)
	Total HSI TX PCSS	0 / 6 (0 %)
	Total HSI PMA TX Serializers	0 / 6 (0 %)
	Total PLLs	0 / 13 (0 %)
	Total DLLs	0 / 4 (0 %)

Compilation Report:

- Command: quartus_edt --read_settings_files=off --write_settings_files=off Shameem_04_10_22_ALU -c Shameem_04_10_22_ALU
- 18236 Number of processors has not been specified which may cause overloading on shared machines. Set the global assignment NUM_PARALLEL_PROCESSORS in your QSF to an appropriate value for best performance.
- 204019 Generated file Shameem_04_10_22_ALU.vo in folder "/home/azwad/Desktop/Shameem_04_10_22_ALU/simulation/modelsim/" for EDA simulation tool
- Quartus Prime EDA Netlist Writer was successful. 0 errors, 1 warning
- 293000 Quartus Prime Full Compilation was successful. 0 errors, 81 warnings

Figure 2: RS Register compilation report in Quartus.

RT Register

The screenshot shows the Quartus Prime software interface. The top menu bar includes File, Edit, View, Project, Assignments, Processing, Tools, Window, and Help. The main window displays the VHDL code for the RT Register:

```

library IEEE;
use IEEE.std_logic_1164.all;

entity Shameem_04_10_22_RT is
    generic (Shameem_04_10_22_N: integer := 32);
    port (
        Shameem_04_10_22_clk, Shameem_04_10_22_wren, Shameem_04_10_22_rden, Shameem_04_10_22_chen: in STD_LOGIC;
        Shameem_04_10_22_content: in STD_LOGIC_VECTOR(Shameem_04_10_22_N-1 downto 0);
        Shameem_04_10_22_result: out STD_LOGIC_VECTOR(Shameem_04_10_22_N-1 downto 0)
    );
end Shameem_04_10_22_RT;

architecture arch of Shameem_04_10_22_RT is
begin
    Shameem_04_10_22_P1: process(Shameem_04_10_22_clk, Shameem_04_10_22_wren)
    begin
        if(rising_edge(Shameem_04_10_22_clk) AND Shameem_04_10_22_wren = '1')
        then Shameem_04_10_22_memory <= Shameem_04_10_22_content;
        end if;
    end process Shameen_04_10_22_P1;
    Shameem_04_10_22_P2: process(Shameem_04_10_22_rden, Shameem_04_10_22_chen, Shameem_04_10_22_memory)
    begin
        if(Shameem_04_10_22_rden = '1' AND Shameem_04_10_22_chen = '1')
        then Shameem_04_10_22_result <= Shameem_04_10_22_memory;
        elsif(Shameem_04_10_22_chen = '0')
        then Shameem_04_10_22_result <= (others => '0');
        end if;
    end process Shameen_04_10_22_P2;
end arch;

```

The Project Navigator shows two files: Shameem_04_10_22_RS.vhd and Shameem_04_10_22_RT.vhd. The Tasks panel indicates successful compilation. The IP Catalog is open, showing the installed IP catalog. The Messages panel shows compilation logs, including warnings about processor specification and success messages for EDA Netlist Writer and full compilation.

Figure 3: RT Register VHDL code compiled successfully in Quartus.

This screenshot shows the Quartus Prime software interface with the same layout as Figure 3. The main window displays the compilation report for the RT Register project:

Flow Summary	Value
Flow Status	Successful – Sat Apr 2 15:02:26 2022
Quartus Prime Version	20.1 Build 720 11/11/2020 SJ Lite Edition
Revision Name	Shameem_04_10_22_ALU
Top-level Entity Name	Shameem_04_10_22_RT
Family	Cyclone V
Device	5CGXF7C7F2C8
Timing Models	Final
Logic utilization (in ALMs)	33 / 56,480 (< 1 %)
Total registers	32
Total pins	68 / 268 (25 %)
Total virtual pins	0
Total block memory bits	0 / 7,024,640 (0 %)
Total DSP Blocks	0 / 156 (0 %)
Total HSSI RX PCGs	0 / 6 (0 %)
Total HSSI PMA RX Deserializers	0 / 6 (0 %)
Total HSSI TX PCGs	0 / 6 (0 %)
Total HSSI PMA TX Serializers	0 / 6 (0 %)
Total PLLs	0 / 7 (0 %)
Total DLLs	0 / 4 (0 %)

The Project Navigator, Tasks panel, and IP Catalog are also visible, along with the same compilation logs as in Figure 3.

Figure 4: RT Register compilation report in Quartus.

RD Register

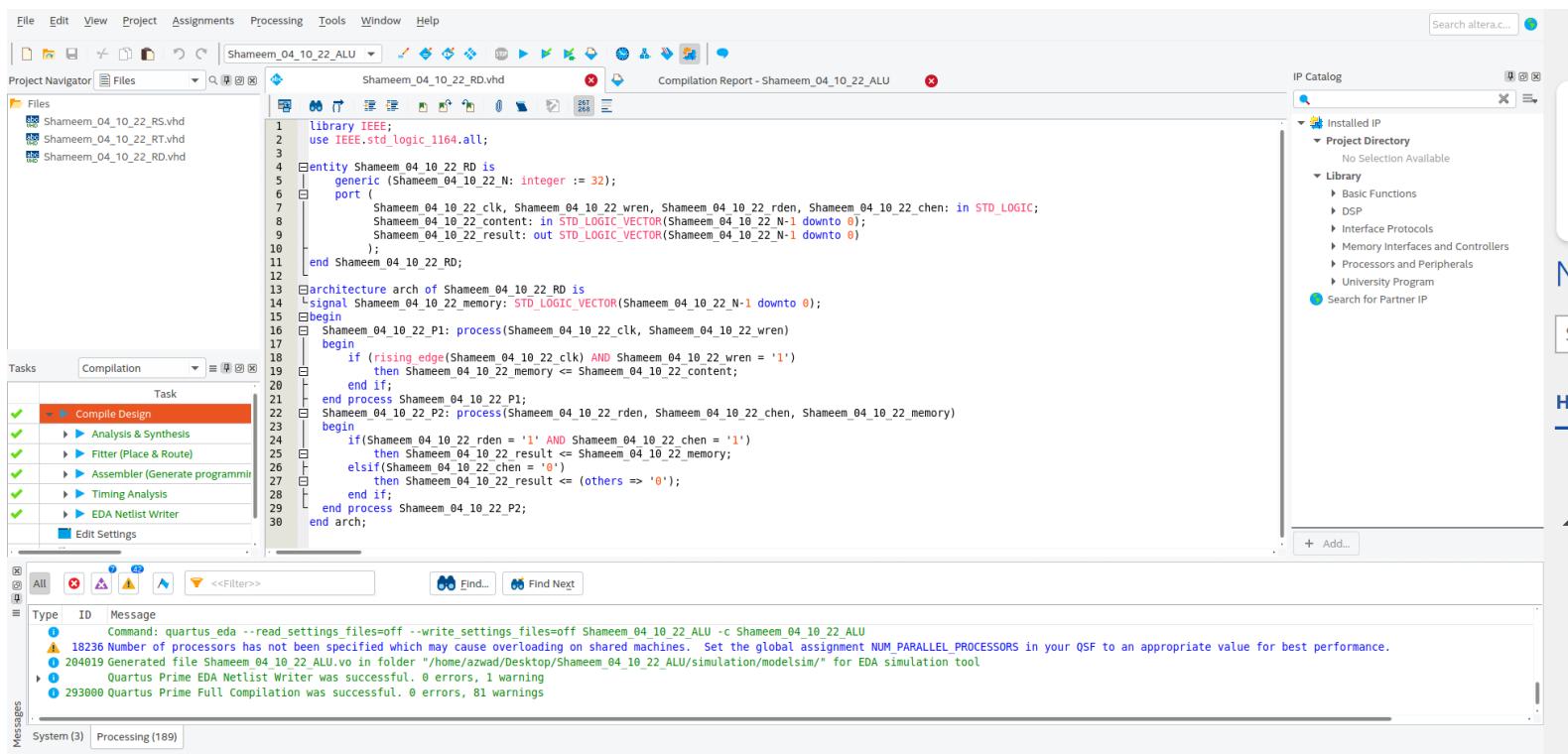


Figure 5: RD Register VHDL code compiled successfully in Quartus.

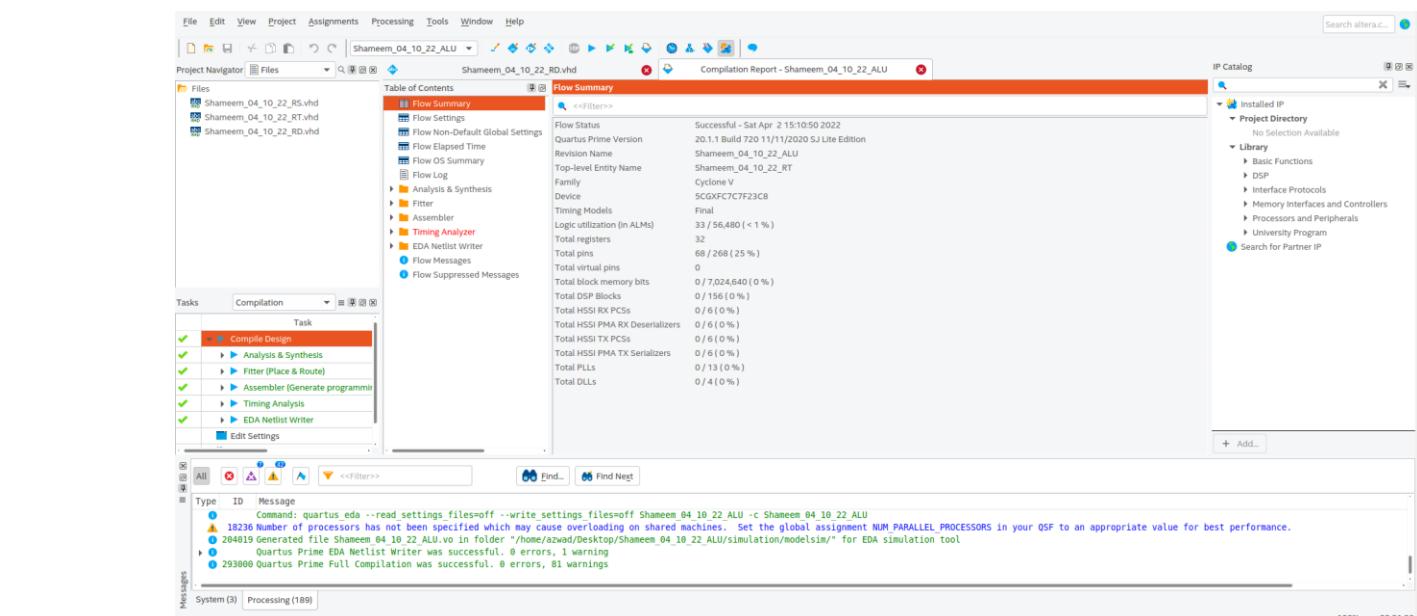


Figure 6: RT Register compilation report in Quartus.

N-Bit AddSub with Flags Unit

The screenshot shows the Quartus Prime software interface with the following details:

- Project Navigator:** Shows files: Shameem_04_10_22_RS.vhd, Shameem_04_10_22_RT.vhd, Shameem_04_10_22_RD.vhd, and Shameem_04_10_22_NBitAddSubFlags.vhd.
- Code Editor:** Displays the VHDL code for the N-Bit AddSub with Flags unit.
- Tasks:** Shows the compilation process with 'Compile Design' selected.
- Messages:** Displays compilation logs with 0 errors and 15 warnings.
- IP Catalog:** Shows the installed IP directory.

```

use ieee.std_logic_unsigned.all;
entity Shameem_04_10_22_NBitAddSubFlags is
    generic (Shameem_04_10_22_X: natural := 16);
    port (
        Shameem_04_10_22_a, Shameem_04_10_22_b: in STD_LOGIC_VECTOR(Shameem_04_10_22_X-1 downto 0);
        Shameem_04_10_22_result: out STD_LOGIC_VECTOR(Shameem_04_10_22_X-1 downto 0);
        Shameem_04_10_22_operation: in STD_LOGIC;
        Shameem_04_10_22_o, Shameem_04_10_22_n, Shameem_04_10_22_z: out STD_LOGIC
    );
end Shameem_04_10_22_NBitAddSubFlags;

architecture arch of Shameem_04_10_22_NBitAddSubFlags is
begin
    Shameem_04_10_22_ip: process(Shameem_04_10_22_b, Shameem_04_10_22_operation)
    begin
        for i in 0 to Shameem_04_10_22_X-1 loop
            Shameem_04_10_22_tmp(i) <= Shameem_04_10_22_b(i) xor Shameem_04_10_22_operation;
        end loop;
        process;
            Shameem_04_10_22_cin(0) <= Shameem_04_10_22_operation;
            Shameem_04_10_22_result <= Shameem_04_10_22_answer;
            Shameem_04_10_22_answer <= Shameem_04_10_22_a XOR Shameem_04_10_22_tmp XOR Shameem_04_10_22_cin(Shameem_04_10_22_X-1 downto 0);
            Shameem_04_10_22_cin(Shameem_04_10_22_X downto 1) <= (Shameem_04_10_22_a AND Shameem_04_10_22_tmp) OR (Shameem_04_10_22_cin(Shameem_04_10_22_X-1 downto 0) AND (Shameem_04_10_22_a XOR Shameem_04_10_22_tmp));
            Shameem_04_10_22_o <= Shameem_04_10_22_cin(Shameem_04_10_22_X-1) XOR Shameem_04_10_22_cin(Shameem_04_10_22_X);
            Shameem_04_10_22_n <= Shameem_04_10_22_answer(Shameem_04_10_22_cin(Shameem_04_10_22_X-1));
            Shameem_04_10_22_z <= NOT(OR_reduce(Shameem_04_10_22_answer));
        end process;
    end;

```

Figure 7: N-Bit AddSub with Flags VHDL code compiled successfully in Quartus.

The screenshot shows the Quartus Prime software interface with the following details:

- Project Navigator:** Shows files: Shameem_04_10_22_RS.vhd, Shameem_04_10_22_RT.vhd, Shameem_04_10_22_RD.vhd, and Shameem_04_10_22_NBitAddSubFlags.vhd.
- Flow Summary:** Provides a detailed summary of the compilation process, including flow settings, analysis, synthesis, and timing analysis results.
- Tasks:** Shows the compilation process with 'Compile Design' selected.
- Messages:** Displays compilation logs with 0 errors and 15 warnings.
- IP Catalog:** Shows the installed IP directory.

Flow	Value
Flow Status	Successful - Sat Apr. 2 15:22:02 2022
Quartus Prime Version	20.1 Build 720 11/11/2020 SJ Lite Edition
Revision Name	Shameem_04_10_22_ALU
Top-level Entity Name	Shameem_04_10_22_NBitAddSubFlags
Family	Cyclone V
Device	5CGXF7C7F23C8
Timing Models	Final
Logic utilization (in ALMs)	29 / 56,480 (< 1 %)
Total registers	0
Total pins	52 / 268 (19 %)
Total virtual pins	0
Total block memory bits	0 / 7,024,640 (0 %)
Total DSP Blocks	0 / 156 (0 %)
Total HSSI RX PCAs	0 / 6 (0 %)
Total HSSI RX Deserializers	0 / 6 (0 %)
Total HSSI TX PCAs	0 / 6 (0 %)
Total HSSI PMA RX Serializers	0 / 6 (0 %)
Total PLLs	0 / 13 (0 %)
Total DLLs	0 / 4 (0 %)

Figure 8: N-Bit AddSub with Flags compilation report in Quartus.

Bitwise Operations Unit

The screenshot shows the Quartus Prime interface with the following details:

- File Menu:** File, Edit, View, Project, Assignments, Processing, Tools, Window, Help.
- Project Navigator:** Shows files: Shameem_04_10_22_RS.vhd, Shameem_04_10_22_RT.vhd, Shameem_04_10_22_RD.vhd, Shameem_04_10_22_NBAddSubFlags.vhd, and Shameem_04_10_22_BitwiseOperations.vhd.
- Code Editor:** Displays the VHDL code for the Bitwise Operations Unit. The code defines a generic port, architecture, and various logic operations like AND, OR, NOR, etc., using STD_LOGIC_VECTOR and STD_INTEGER.
- Tasks:** Shows the compilation process completed successfully with 0 errors and 1 warning.
- Messages:** Displays compilation messages including warnings about the number of processors and success of the EDA Netlist Writer.
- IP Catalog:** Shows installed IP components and a search bar for partner IP.

Figure 9: Bitwise Operations VHDL code compiled successfully in Quartus.

The screenshot shows the Quartus Prime interface with the following details:

- File Menu:** File, Edit, View, Project, Assignments, Processing, Tools, Window, Help.
- Project Navigator:** Shows files: Shameem_04_10_22_RS.vhd, Shameem_04_10_22_RT.vhd, Shameem_04_10_22_RD.vhd, Shameem_04_10_22_NBAddSubFlags.vhd, and Shameem_04_10_22_BitwiseOperations.vhd.
- Flow Summary:** Provides a detailed report of the compilation process, including flow status, revision information, and device utilization.
- Tasks:** Shows the compilation process completed successfully with 0 errors and 1 warning.
- Messages:** Displays compilation messages including warnings about the number of processors and success of the EDA Netlist Writer.
- IP Catalog:** Shows installed IP components and a search bar for partner IP.

Figure 10: Bitwise Operations compilation report in Quartus.

The screenshot shows the Quartus Prime software interface. The main window displays the VHDL code for the `Shameem_04_10_22_IMM16.vhd` file. The code defines an entity `Shameem_04_10_22_IMM16` with a generic parameter `N` set to 16. It contains an architecture `arch` with two processes: `P1` and `P2`. Process `P1` handles the write operation based on the rising edge of the clock and the enable signal. Process `P2` handles the read operation, comparing the input memory address with the read address. The code uses standard IEEE libraries and STD logic vectors.

The left sidebar shows the Project Navigator with files like `Shameem_04_10_22_RS.vhd`, `Shameem_04_10_22_RT.vhd`, etc., and the Tasks panel showing successful compilation steps. The bottom Messages panel displays build logs with warnings about the number of processors and success messages for the netlist writer and full compilation.

```

library IEEE;
use IEEE.STD_LOGIC_1164.all;

entity Shameem_04_10_22_IMM16 is
    generic (Shameem_04_10_22_N: integer := 16);
    port (
        Shameem_04_10_22_clock, Shameem_04_10_22_write, Shameem_04_10_22_read, Shameem_04_10_22_enable: in STD_LOGIC;
        Shameem_04_10_22_input: in STD_LOGIC_VECTOR(Shameem_04_10_22_N-1 downto 0);
        Shameem_04_10_22_out: out STD_LOGIC_VECTOR(Shameem_04_10_22_N-1 downto 0)
    );
end Shameem_04_10_22_IMM16;

architecture arch of Shameem_04_10_22_IMM16 is
begin
    process(Shameem_04_10_22_clock, Shameem_04_10_22_write)
    begin
        if(rising edge(Shameem_04_10_22_clock) AND Shameem_04_10_22_write = '1')
        then Shameem_04_10_22_memory <= Shameem_04_10_22_input;
        end if;
    end process Shameen_04_10_22_P1;
    process(Shameem_04_10_22_read, Shameem_04_10_22_enable, Shameem_04_10_22_memory)
    begin
        if(Shameem_04_10_22_read = '1' AND Shameem_04_10_22_enable = '1')
        then Shameem_04_10_22_out <= Shameem_04_10_22_memory;
        elsif(Shameem_04_10_22_enable = '0')
        then Shameem_04_10_22_out <= (others => '0');
        end if;
    end process Shameen_04_10_22_P2;
end arch;

```

Type ID Message
 Command: quartus eda --read_settings_files=off --write_settings_files=off Shameem_04_10_22_ALU -c Shameem_04_10_22_ALU
 18236 Number of processors has not been specified which may cause overloading on shared machines. Set the global assignment NUM_PARALLEL_PROCESSORS in your QSF to an appropriate value for best performance.
 284019 Generated file Shameem_04_10_22_ALU.vo in folder "/home/azwad/Desktop/Shameem_04_10_22_ALU/simulation/modelsim/" for EDA simulation tool
 Quartus Prime EDA Netlist Writer was successful. 0 errors, 1 warning
 293000 Quartus Prime Full Compilation was successful. 0 errors, 49 warnings

IMM16 Register
Figure 11: IMM16 Register VHDL code compiled successfully in Quartus.

This screenshot shows the Quartus Prime software interface with the compilation report for the `Shameem_04_10_22_ALU` project. The central window displays the `Flow Summary` report, which provides detailed statistics about the compilation process. Key metrics include:
 - Flow Status: Successful - Sat Apr 2 16:11:58 2022
 - Quartus Prime Version: 20.1 Build 720 11/11/2020 SJ Edition
 - Revision Name: Shameem_04_10_22_ALU
 - Top-level Entity Name: Shameem_04_10_22_IMM16
 - Family: Cyclone V
 - Device: 5CQXFC7CF23CB
 - Timing Models: Final
 - Logic utilization (in ALMs): 17 / 56,480 (< 1 %)
 - Total registers: 16
 - Total pins: 36 / 268 (13 %)
 - Total virtual pins: 0
 - Total block memory bits: 0 / 7,024,640 (0 %)
 - Total DSP Blocks: 0 / 156 (0 %)
 - Total HSSI RX PCGs: 0 / 6 (0 %)
 - Total HSSI PMA RX Deserializers: 0 / 6 (0 %)
 - Total HSSI TX PCGs: 0 / 6 (0 %)
 - Total HSSI PMA TX Serializers: 0 / 6 (0 %)
 - Total PLLs: 0 / 13 (0 %)
 - Total DLLs: 0 / 4 (0 %)

The left sidebar shows the Project Navigator and Tasks panel, both indicating successful compilation steps. The bottom Messages panel shows the same log entries as Figure 11.

Figure 12: IMM16 Register compilation report in Quartus.

Sign Extender Unit

The screenshot shows the Quartus Prime software interface with the following details:

- File Menu:** File, Edit, View, Project, Assignments, Processing, Tools, Window, Help.
- Project Navigator:** Shows files: Shameem_04_10_22_RS.vhd, Shameem_04_10_22_RT.vhd, Shameem_04_10_22_RD.vhd, Shameem_04_10_22_NB1AddSubFlags.vhd, Shameem_04_10_22_BitwiseOperations.vhd, Shameem_04_10_22_IMM16.vhd, and Shameem_04_10_22_SignExtender.vhd.
- Code Editor:** Displays the VHDL code for the Sign Extender unit.
- Compilation Report:** Shows successful compilation results.
- IP Catalog:** Shows installed IP components.
- Tasks:** Task list includes: Compile Design, Analysis & Synthesis, Fitter (Place & Route), Assembler (Generate program), Timing Analysis, EDA Netlist Writer, and Edit Settings. The 'Compile Design' task is highlighted.
- Messages:** Shows compilation messages, including warnings about shared machines and success messages for Quartus Prime tools.
- Bottom Status:** System (6) Processing (161), Line 16 Col 123, VHDL File, 100% 00:00:51.

Figure 13: Sign Extender VHDL code compiled successfully in Quartus.

The screenshot shows the Quartus Prime software interface with the following details:

- File Menu:** File, Edit, View, Project, Assignments, Processing, Tools, Window, Help.
- Project Navigator:** Shows files: Shameem_04_10_22_RS.vhd, Shameem_04_10_22_RT.vhd, Shameem_04_10_22_RD.vhd, Shameem_04_10_22_NB1AddSubFlags.vhd, Shameem_04_10_22_BitwiseOperations.vhd, Shameem_04_10_22_IMM16.vhd, and Shameem_04_10_22_SignExtender.vhd.
- Table of Contents:** Flow Summary, Flow Settings, Flow Non-Default Global Settings, Flow Elapsed Time, Flow OS Summary, Flow Log, Analysis & Synthesis, Fitter, Assembler, Timing Analyzer, EDA Netlist Writer, Flow Messages, and Flow Suppressed Messages.
- Compilation Report:** Shows successful compilation results.
- IP Catalog:** Shows installed IP components.
- Tasks:** Task list includes: Compile Design, Analysis & Synthesis, Fitter (Place & Route), Assembler (Generate program), Timing Analysis, EDA Netlist Writer, and Edit Settings. The 'Compile Design' task is highlighted.
- Messages:** Shows compilation messages, including warnings about shared machines and success messages for Quartus Prime tools.
- Bottom Status:** System (6) Processing (161), 100% 00:00:51.

Figure 14: Sign Extender compilation report in Quartus.

MAR – Memory Address Register

```

library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.STD_LOGIC_UNSIGNED.all;
use IEEE.numeric_std.all;

entity Shameem_04_10_22_MAR is
    generic (Shameem_04_10_22_N: integer := 32);
    port
        Shemeen_04_10_22_clk, Shemeen_04_10_22_wren, Shemeen_04_10_22_rden, Shemeen_04_10_22_chen: in STD_LOGIC;
        Shemeen_04_10_22_content: in STD_LOGIC_VECTOR(Shameem_04_10_22_N-1 downto 0);
        Shemeen_04_10_22_ext: in STD_LOGIC_VECTOR(Shameem_04_10_22_N-1 downto 0);
        Shemeen_04_10_22_result: out STD_LOGIC_VECTOR(Shameem_04_10_22_N-1 downto 0);
end Shameen_04_10_22_MAR;

architecture arch of Shameen_04_10_22_MAR is
begin
    Shemeen_04_10_22_P1: process(Shameen_04_10_22_clk, Shemeen_04_10_22_wren)
    begin
        if(rising_edge(Shameen_04_10_22_clk) AND Shemeen_04_10_22_wren = '1')
            then Shemeen_04_10_22_memory <= Shemeen_04_10_22_content + Shemeen_04_10_22_ext;
        end if;
    end process Shemeen_04_10_22_P1;
    Shemeen_04_10_22_P2: process(Shameen_04_10_22_rden, Shemeen_04_10_22_chen, Shemeen_04_10_22_memory)
    begin
        if(Shemeen_04_10_22_rden = '1' AND Shemeen_04_10_22_chen = '1')
            then Shemeen_04_10_22_result <= Shemeen_04_10_22_memory;
        elsif(Shemeen_04_10_22_chen = '0')
            then Shemeen_04_10_22_result <= (others => '0');
        end if;
    end process Shemeen_04_10_22_P2;
end arch;

```

Messages

- 18236 Number of processors has not been specified which may cause overloading on shared machines. Set the global assignment NUM_PARALLEL_PROCESSORS in your QSF to an appropriate value for best performance.
- 204019 Generated file Shameem_04_10_22_ALU.vo was successful. 0 errors, 1 warning
- 293000 Quartus Prime EDA Netlist Writer was successful. 0 errors, 81 warnings

System (7) Processing (194)

Figure 15: Memory Address Register VHDL code compiled successfully in Quartus.

Flow Summary

Flow Settings	Successful - Sat Apr 2 16:28:40 2022
Quartus Prime Version	20.1 Build 720 11/11/2020 SJ Lite Edition
Revision Name	Shameem_04_10_22_ALU
Top-level Entity Name	Shameem_04_10_22_MAR
Family	Cyclone V
Device	SGCXFC7CFC23CB
Timing Models	Final
Logic utilization (In ALMs)	49 / 56,480 (< 1 %)
Total registers	32
Total pins	100 / 268 (37 %)
Total virtual pins	0
Total block memory bits	0 / 7,024,640 (0 %)
Total DSP Blocks	0 / 156 (0 %)
Total HSSI RX PC5s	0 / 6 (0 %)
Total HSSI TX PC5s	0 / 6 (0 %)
Total HSSI PMA TX Serializers	0 / 6 (0 %)
Total PLLs	0 / 13 (0 %)
Total DLLs	0 / 4 (0 %)

Messages

- 18236 Number of processors has not been specified which may cause overloading on shared machines. Set the global assignment NUM_PARALLEL_PROCESSORS in your QSF to an appropriate value for best performance.
- 204019 Generated file Shameem_04_10_22_ALU.vo was successful. 0 errors, 1 warning
- 293000 Quartus Prime Full Compilation was successful. 0 errors, 81 warnings

System (7) Processing (194)

Figure 16: Memory Address Register compilation report in Quartus.

MDR – Memory Data Register

The screenshot shows the Quartus Prime interface with the project "Shameem_04_10_22_ALU" open. The main window displays the VHDL code for the Memory Data Register (MDR). The code includes declarations for IEEE packages and the entity/architecture pair for the MDR. The architecture contains two processes: P1 handles writes (wren = '1'), and P2 handles reads (rden = '1'). Both processes update memory and calculate result based on chen and ext inputs. The compilation report at the bottom indicates a successful build with 0 errors and 81 warnings.

```

library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.STD_LOGIC_UNSIGNED.all;
use IEEE.numeric_std.all;

entity Shameem_04_10_22_MDR is
    generic (Shameem_04_10_22_N: integer := 32);
    port (
        Shameem_04_10_22_clk, Shameem_04_10_22_wren, Shameem_04_10_22_rden, Shameem_04_10_22_chen: in STD_LOGIC;
        Shameem_04_10_22_content: in STD_LOGIC_VECTOR(Shameem_04_10_22_N-1 downto 0);
        Shameem_04_10_22_ext: in STD_LOGIC_VECTOR(Shameem_04_10_22_N-1 downto 0);
        Shameem_04_10_22_result: out STD_LOGIC_VECTOR(Shameem_04_10_22_N-1 downto 0));
    end Shameem_04_10_22_MDR;

Architecture arch of Shameem_04_10_22_MDR is
begin
    Shameem_04_10_22_P1: process(Shameem_04_10_22_clk, Shameem_04_10_22_wren)
    begin
        if(rising edge(Shameem_04_10_22_clk) AND Shameem_04_10_22_wren = '1')
        then Shameem_04_10_22_memory <= Shameem_04_10_22_content + Shameem_04_10_22_ext;
        end if;
    end process Shameem_04_10_22_P1;
    Shameem_04_10_22_P2: process(Shameem_04_10_22_rden, Shameem_04_10_22_chen, Shameem_04_10_22_memory)
    begin
        if(Shameem_04_10_22_rden = '1' AND Shameem_04_10_22_chen = '1')
        then Shameem_04_10_22_result <= Shameem_04_10_22_memory;
        elsif(Shameem_04_10_22_chen = '0')
        then Shameem_04_10_22_result <= (others => '0');
        end if;
    end process Shameem_04_10_22_P2;
end arch;

```

Compilation Report:

- Command: quartus eda --read settings_files=off --write settings_files=off Shameem_04_10_22_ALU -c Shameem_04_10_22_ALU
- 18236 Number of processors has not been specified which may cause overloading on shared machines. Set the global assignment NUM_PARALLEL_PROCESSORS in your QSF to an appropriate value for best performance.
- 204019 Generated file Shameem_04_10_22_ALU.vo in folder "/home/azwad/Desktop/Shameem_04_10_22_ALU/simulation/modelsim/" for EDA simulation tool
- Quartus Prime EDA Netlist Writer was successful. 0 errors, 1 warning
- 293000 Quartus Prime Full Compilation was successful. 0 errors, 81 warnings

Figure 17: Memory Data Register VHDL code compiled successfully in Quartus.

The screenshot shows the Quartus Prime interface with the project "Shameem_04_10_22_ALU" open. The main window displays the Flow Summary report. The report details the flow status (Successful - Sat Apr 2 16:28:40 2022), Quartus Prime Version (20.1 Build 720 11/1/2020 SJ Lite Edition), Revision Name (Shameem_04_10_22_ALU), and various hardware and timing metrics. The compilation tasks listed in the tasks panel include Compile Design, Analysis & Synthesis, Fitter, Assembler, Timing Analysis, and EDA Netlist Writer, all of which were successful.

Flow Status	Success - Sat Apr 2 16:28:40 2022
Quartus Prime Version	20.1 Build 720 11/1/2020 SJ Lite Edition
Revision Name	Shameem_04_10_22_ALU
Top-level Entity Name	Shameem_04_10_22_MAR
Family	Cyclone V
Device	5CGXF7C7F23C8
Timing Models	Final
Logic utilization (In ALMs)	49 / 56,480 (< 1 %)
Total registers	32
Total pins	100 / 268 (37 %)
Total virtual pins	0
Total block memory bits	0 / 7,024,640 (0 %)
Total DSP Blocks	0 / 156 (0 %)
Total HSSI RX PC5s	0 / 6 (0 %)
Total HSSI PMA-RX Deserializers	0 / 6 (0 %)
Total HSSI TX PC5s	0 / 6 (0 %)
Total HSSI PMA-TX Serializers	0 / 6 (0 %)
Total PLLs	0 / 13 (0 %)
Total DILs	0 / 4 (0 %)

Compilation Report:

- Command: quartus eda --read settings_files=off --write settings_files=off Shameem_04_10_22_ALU -c Shameem_04_10_22_ALU
- 18236 Number of processors has not been specified which may cause overloading on shared machines. Set the global assignment NUM_PARALLEL_PROCESSORS in your QSF to an appropriate value for best performance.
- 204019 Generated file Shameem_04_10_22_ALU.vo in folder "/home/azwad/Desktop/Shameem_04_10_22_ALU/simulation/modelsim/" for EDA simulation tool
- Quartus Prime EDA Netlist Writer was successful. 0 errors, 1 warning
- 293000 Quartus Prime Full Compilation was successful. 0 errors, 81 warnings

Figure 18: Memory Data Register compilation report in Quartus.

ALU – Arithmetic Logic Unit

The screenshot shows the Quartus Prime interface with the project "Shameem_04_10_22_ALU" open. The main window displays the VHDL code for the ALU. The "Compilation" tab in the tasks bar is selected, showing a green checkmark next to "Compile Design". The status bar at the bottom right indicates "100% 00:03:05". The IP Catalog on the right side shows various IP components available for the project.

```

library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_signed.all;
use IEEE.numeric_std.all;

entity Shameem_04_10_22_ALU is
    generic(N: integer := 32);
    port (
        Shameem_04_10_22_clk: in std_logic;
        Shameem_04_10_22_RInput, Shameem_04_10_22_MDRinput: in std_logic_vector (N-1 downto 0);
        Shameem_04_10_22_Imm: in std_logic_vector (15 downto 0);
        Shameem_04_10_22_operation: in std_logic_vector(3 downto 0);
        Shameem_04_10_22_RDinput: out std_logic_vector (N-1 downto 0);
        Shameem_04_10_22_ROutput: out std_logic_vector (N-1 downto 0);
        Shameem_04_10_22_Z, Shameem_04_10_22_N: out std_logic := '0';
    );
end Shameem_04_10_22_ALU;

architecture arch of Shameem_04_10_22_ALU is
    signal Shameem_04_10_22_cout, Shameem_04_10_22_tmp1, Shameem_04_10_22_tmp2, Shameem_04_10_22_tmp3, Shameem_04_10_22_tmp4, Shameem_04_10_22_tmp5,
        Shameem_04_10_22_tmp6, Shameem_04_10_22_tmp7, Shameem_04_10_22_tmp8, Shameem_04_10_22_tmp9, z_tmp3, Shameem_04_10_22_tmp10,
        Shameem_04_10_22_tmp11, Shameem_04_10_22_tmp12, Shameem_04_10_22_tmp13, Shameem_04_10_22_tmp14, Shameem_04_10_22_tmp15, Shameem_04_10_22_tmp16,
        Shameem_04_10_22_TMP17, Shameem_04_10_22_TMP18, Shameem_04_10_22_TMP19, Shameem_04_10_22_TMP20,
        Shameem_04_10_22_MRwren, Shameem_04_10_22_MRwren: std_logic := '0';
    signal Shameem_04_10_22_tmp21, Shameem_04_10_22_tmp22, Shameem_04_10_22_tmp23, Shameem_04_10_22_tmp24, Shameem_04_10_22_tmp25,
        Shameem_04_10_22_tmp26, Shameem_04_10_22_tmp27: std_logic_vector (N-1 downto 0);
    signal Shameem_04_10_22_RSout, Shameem_04_10_22_RTout, Shameem_04_10_22_RDin, Shameem_04_10_22_EXTout,
        Shameem_04_10_22_MROutput, Shameem_04_10_22_MROutput: std_logic_vector (N-1 downto 0);
    signal imm_0: std_logic_vector (15 downto 0);

```

Figure 19: Arithmetic Logic Unit VHDL code compiled successfully in Quartus.

This screenshot continues the VHDL code for the ALU. It defines two components: "Shameem_04_10_22_RS" and "Shameem_04_10_22_RT". Both components have a generic parameter "N" set to 32. They each have a single port with a generic "Shameem_04_10_22_N" set to 32. The port contains three signals: "Shameem_04_10_22_clk", "Shameem_04_10_22_content" (type STD_LOGIC_VECTOR), and "Shameem_04_10_22_result" (type STD_LOGIC_VECTOR).

```

component Shameem_04_10_22_RS is
    generic (Shameem_04_10_22_N: integer := 32);
    port (
        Shameem_04_10_22_clk, Shameem_04_10_22_wren, Shameem_04_10_22_rden, Shameem_04_10_22_chen: in STD_LOGIC;
        Shameem_04_10_22_content: in STD_LOGIC_VECTOR(Shameem_04_10_22_N-1 downto 0);
        Shameem_04_10_22_result: out STD_LOGIC_VECTOR(Shameem_04_10_22_N-1 downto 0)
    );
end component;

component Shameem_04_10_22_RT is
    generic (Shameem_04_10_22_N: integer := 32);
    port (
        Shameem_04_10_22_clk, Shameem_04_10_22_wren, Shameem_04_10_22_rden, Shameem_04_10_22_chen: in STD_LOGIC;
        Shameem_04_10_22_content: in STD_LOGIC_VECTOR(Shameem_04_10_22_N-1 downto 0);
        Shameem_04_10_22_result: out STD_LOGIC_VECTOR(Shameem_04_10_22_N-1 downto 0)
    );
end component;

```

Figure 20: Arithmetic Logic Unit VHDL code continued.

```

File Edit View Project Assignments Processing Tools Window Help
Project Navigator Files Compilation Report - Shameem_04_10_22_ALU
Shameem_04_10_22_RS.vhd
Shameem_04_10_22_RT.vhd
Shameem_04_10_22_NBitAddSubFlags.vhd
Shameem_04_10_22_BitwiseOperations.vhd
Shameem_04_10_22_IMMI16.vhd
Shameem_04_10_22_SignExtender.vhd
Shameem_04_10_22_MAR.vhd
Shameem_04_10_22_MDR.vhd
Shameem_04_10_22_ALU.vhd

Shameem_04_10_22_ALU.vhd
component Shameem_04_10_22_RT is
  generic (Shameem_04_10_22_N: integer := 32);
  port (
    Shameem_04_10_22_clk, Shameem_04_10_22_wren, Shameem_04_10_22_rden, Shameem_04_10_22_chen: in STD_LOGIC;
    Shameem_04_10_22_content: in STD_LOGIC_VECTOR(Shameem_04_10_22_N-1 downto 0);
    Shameem_04_10_22_result: out STD_LOGIC_VECTOR(Shameem_04_10_22_N-1 downto 0)
  );
end component;

component Shameem_04_10_22_RD is
  generic (Shameem_04_10_22_N: integer := 32);
  port (
    Shameem_04_10_22_clk, Shameem_04_10_22_wren, Shameem_04_10_22_rden, Shameem_04_10_22_chen: in STD_LOGIC;
    Shameem_04_10_22_content: in STD_LOGIC_VECTOR(Shameem_04_10_22_N-1 downto 0);
    Shameem_04_10_22_result: out STD_LOGIC_VECTOR(Shameem_04_10_22_N-1 downto 0)
  );
end component;

component Shameem_04_10_22_NBitAddSubFlags is
  generic (Shameem_04_10_22_X: natural := 16);
  port (
    Shameem_04_10_22_a, Shameem_04_10_22_b: in STD_LOGIC_VECTOR(Shameem_04_10_22_X-1 downto 0);
    Shameem_04_10_22_result: out STD_LOGIC_VECTOR(Shameem_04_10_22_X-1 downto 0);
    Shameem_04_10_22_operation: in STD_LOGIC;
    Shameem_04_10_22_o, Shameem_04_10_22_n, Shameem_04_10_22_z: out STD_LOGIC
  );
end component;

component Shameem_04_10_22_BitwiseOperations is
  generic(Shameem_04_10_22_N: integer := 32);
  port (
    Shameem_04_10_22_input1, Shameem_04_10_22_input2, Shameem_04_10_22_ext: in STD_LOGIC_VECTOR(Shameem_04_10_22_N-1 downto 0);
    Shameem_04_10_22_imm: in STD_LOGIC_VECTOR(15 downto 0);
    Shameem_04_10_22_operation: in STD_LOGIC_VECTOR(3 downto 0);
    Shameem_04_10_22_output: out STD_LOGIC_VECTOR(Shameem_04_10_22_N-1 downto 0)
  );
end component;

component Shameem_04_10_22_IMMI16 is
  generic (Shameem_04_10_22_N: integer := 16);
  port (
    Shameem_04_10_22_clock, Shameem_04_10_22_write, Shameem_04_10_22_read, Shameem_04_10_22_enable: in STD_LOGIC;
    Shameem_04_10_22_input: in STD_LOGIC_VECTOR(Shameem_04_10_22_N-1 downto 0);
    Shameem_04_10_22_out: out STD_LOGIC_VECTOR(Shameem_04_10_22_N-1 downto 0)
  );
end component;

```

Find what: tmp2 Replace with: 0_22_IMMOutput Look in: Current File Search: Down Find Next Replace Replace All Mark All

Messages

Type	ID	Message
1		Command: quartus_edt --read_settings_files=off --write_settings_files=off Shameem_04_10_22_ALU -c Shameem_04_10_22_ALU
18236		Number of processors has not been specified which may cause overloading on shared machines. Set the global assignment NUM_PARALLEL_PROCESSORS in your QSF to an appropriate value for best performance.
204019		Generated file Shameem_04_10_22_ALU.vo in folder '/home/azwad/Desktop/Shameem_04_10_22_ALU/simulation/modelsim/' for EDA simulation tool
1		Quartus Prime EDA Netlist Writer was successful. 0 errors, 1 warning
1		293000 Quartus Prime Full Compilation was successful. 0 errors, 310 warnings

System (14) Processing (555)

Figure 21: Arithmetic Logic Unit VHDL code continued.

```

File Edit View Project Assignments Processing Tools Window Help
Project Navigator Files Compilation Report - Shameem_04_10_22_ALU
Shameem_04_10_22_RS.vhd
Shameem_04_10_22_RT.vhd
Shameem_04_10_22_NBitAddSubFlags.vhd
Shameem_04_10_22_BitwiseOperations.vhd
Shameem_04_10_22_IMMI16.vhd
Shameem_04_10_22_SignExtender.vhd
Shameem_04_10_22_MAR.vhd
Shameem_04_10_22_MDR.vhd
Shameem_04_10_22_ALU.vhd

Shameem_04_10_22_ALU.vhd
component Shameem_04_10_22_NBitAddSubFlags is
  generic (Shameem_04_10_22_X: natural := 16);
  port (
    Shameem_04_10_22_a, Shameem_04_10_22_b: in STD_LOGIC_VECTOR(Shameem_04_10_22_X-1 downto 0);
    Shameem_04_10_22_result: out STD_LOGIC_VECTOR(Shameem_04_10_22_X-1 downto 0);
    Shameem_04_10_22_operation: in STD_LOGIC;
    Shameem_04_10_22_o, Shameem_04_10_22_n, Shameem_04_10_22_z: out STD_LOGIC
  );
end component;

component Shameem_04_10_22_BitwiseOperations is
  generic(Shameem_04_10_22_N: integer := 32);
  port (
    Shameem_04_10_22_input1, Shameem_04_10_22_input2, Shameem_04_10_22_ext: in STD_LOGIC_VECTOR(Shameem_04_10_22_N-1 downto 0);
    Shameem_04_10_22_imm: in STD_LOGIC_VECTOR(15 downto 0);
    Shameem_04_10_22_operation: in STD_LOGIC_VECTOR(3 downto 0);
    Shameem_04_10_22_output: out STD_LOGIC_VECTOR(Shameem_04_10_22_N-1 downto 0)
  );
end component;

component Shameem_04_10_22_IMMI16 is
  generic (Shameem_04_10_22_N: integer := 16);
  port (
    Shameem_04_10_22_clock, Shameem_04_10_22_write, Shameem_04_10_22_read, Shameem_04_10_22_enable: in STD_LOGIC;
    Shameem_04_10_22_input: in STD_LOGIC_VECTOR(Shameem_04_10_22_N-1 downto 0);
    Shameem_04_10_22_out: out STD_LOGIC_VECTOR(Shameem_04_10_22_N-1 downto 0)
  );
end component;

```

Find what: tmp2 Replace with: 0_22_IMMOutput Look in: Current File Search: Down Find Next Replace Replace All Mark All

Messages

Type	ID	Message
1		Command: quartus_edt --read_settings_files=off --write_settings_files=off Shameem_04_10_22_ALU -c Shameem_04_10_22_ALU
18236		Number of processors has not been specified which may cause overloading on shared machines. Set the global assignment NUM_PARALLEL_PROCESSORS in your QSF to an appropriate value for best performance.
204019		Generated file Shameem_04_10_22_ALU.vo in folder '/home/azwad/Desktop/Shameem_04_10_22_ALU/simulation/modelsim/' for EDA simulation tool
1		Quartus Prime EDA Netlist Writer was successful. 0 errors, 1 warning
1		293000 Quartus Prime Full Compilation was successful. 0 errors, 310 warnings

System (14) Processing (555)

Figure 22: Arithmetic Logic Unit VHDL code continued

```

component Shameem_04_10_22_IMM16 is
generic(Shameem_04_10_22_N: integer := 16);
port(
    Shameem_04_10_22_clock, Shameem_04_10_22_write, Shameem_04_10_22_read, Shameem_04_10_22_enable: in STD_LOGIC;
    Shameem_04_10_22_input: in STD_LOGIC_VECTOR(Shameem_04_10_22_N-1 downto 0);
    Shameem_04_10_22_out: out STD_LOGIC_VECTOR(Shameem_04_10_22_N-1 downto 0)
);
end component;

component Shameem_04_10_22_SignExtender is
port(
    Shameem_04_10_22_input: in STD_LOGIC_VECTOR(15 downto 0);
    Shameem_04_10_22_output: out STD_LOGIC_VECTOR(31 downto 0)
);
end component;

component Shameem_04_10_22_MAR is
generic(Shameem_04_10_22_N: integer := 32);
port(
    Shameem_04_10_22_clk, Shameem_04_10_22_wren, Shameem_04_10_22_rden, Shameem_04_10_22_chen: in STD_LOGIC;
    Shameem_04_10_22_content: in STD_LOGIC_VECTOR(Shameem_04_10_22_N-1 downto 0);
    Shameem_04_10_22_ext: in STD_LOGIC_VECTOR(Shameem_04_10_22_N-1 downto 0);
    Shameem_04_10_22_result: out STD_LOGIC_VECTOR(Shameem_04_10_22_N-1 downto 0)
);
end component;

component Shameem_04_10_22_MDR is
generic(Shameem_04_10_22_N: integer := 32);
port(
    Shameem_04_10_22_clk, Shameem_04_10_22_wren, Shameem_04_10_22_rden, Shameem_04_10_22_chen: in STD_LOGIC;
    Shameem_04_10_22_content: in STD_LOGIC_VECTOR(Shameem_04_10_22_N-1 downto 0);
    Shameem_04_10_22_ext: in STD_LOGIC_VECTOR(Shameem_04_10_22_N-1 downto 0);
    Shameem_04_10_22_result: out STD_LOGIC_VECTOR(Shameem_04_10_22_N-1 downto 0)
);
end component;

```

Messages:

- Command: quartus_edt --read_settings_files=off --write_settings_files=off Shameem_04_10_22_ALU -c Shameem_04_10_22_ALU
- 18236 Number of processors has not been specified which may cause overloading on shared machines. Set the global assignment NUM_PARALLEL_PROCESSORS in your QSF to an appropriate value for best performance.
- 204019 Generated file Shameem_04_10_22_ALU.vo in folder "/home/azwad/Desktop/Shameem_04_10_22_ALU/simulation/modelsim/" for EDA simulation tool
- Quartus Prime EDA Netlist Writer was successful. 0 errors, 1 warning
- 293000 Quartus Prime Full Compilation was successful. 0 errors, 310 warnings

Figure 23: Arithmetic Logic Unit VHDL code continued.

```

component Shameem_04_10_22_MDR is
generic(Shameem_04_10_22_N: integer := 32);
port(
    Shameem_04_10_22_clk, Shameem_04_10_22_wren, Shameem_04_10_22_rden, Shameem_04_10_22_chen: in STD_LOGIC;
    Shameem_04_10_22_content: in STD_LOGIC_VECTOR(Shameem_04_10_22_N-1 downto 0);
    Shameem_04_10_22_ext: in STD_LOGIC_VECTOR(Shameem_04_10_22_N-1 downto 0);
    Shameem_04_10_22_result: out STD_LOGIC_VECTOR(Shameem_04_10_22_N-1 downto 0)
);
end component;

begin
RD: Shameem_04_10_22_RD generic map (32) port map (Shameem_04_10_22_clk, '1', '1', '1', '1', Shameem_04_10_22_RDin, Shameem_04_10_22_RDinput);
RS: Shameem_04_10_22_RS generic map (32) port map (Shameem_04_10_22_clk, '1', '1', '1', '1', Shameem_04_10_22_RSin, Shameem_04_10_22_RSout);
IMM: Shameem_04_10_22_IMM generic map (16) port map (Shameem_04_10_22_clk, '1', '1', '1', '1', Shameem_04_10_22_imm, imm_o);
Extension: Shameem_04_10_22_SignExtender generic map (Shameem_04_10_22_imm, Shameem_04_10_22_EXTout);
BitOperation: Shameem_04_10_22_BitwiseOperations generic map (32) port map (Shameem_04_10_22_RSout, Shameem_04_10_22_RTout, Shameem_04_10_22_EXTout);
add: Shameem_04_10_22_NBItAddSubFlags generic map (32) port map (Shameem_04_10_22_RSout, Shameem_04_10_22_RTout, Shameem_04_10_22_tmpl, '0', Sha
addi: Shameem_04_10_22_NBItAddSubFlags generic map (32) port map (Shameem_04_10_22_RSout, Shameem_04_10_22_EXTout, Shameem_04_10_22_tmpl2, '0', Sha
addiu: Shameem_04_10_22_NBItAddSubFlags generic map (32) port map (Shameem_04_10_22_RSout, Shameem_04_10_22_EXTout, Shameem_04_10_22_tmpl3, '0', Sha
addu: Shameem_04_10_22_NBItAddSubFlags generic map (32) port map (Shameem_04_10_22_RSout, Shameem_04_10_22_RTout, Shameem_04_10_22_tmpl4, '0', Sha
sub: Shameem_04_10_22_NBItAddSubFlags generic map (32) port map (Shameem_04_10_22_RSout, Shameem_04_10_22_RTout, Shameem_04_10_22_tmpl5, '1', Sha
subu: Shameem_04_10_22_NBItAddSubFlags generic map (32) port map (Shameem_04_10_22_RSout, Shameem_04_10_22_RTout, Shameem_04_10_22_tmpl6, '1', Sha
MAR: Shameem_04_10_22_MAR generic map (32) port map (Shameem_04_10_22_clk, '1', '1', '1', '1', Shameem_04_10_22_RSout, Shameem_04_10_22_EXTout, Shameem_
MDR: Shameem_04_10_22_MDR generic map (32) port map (Shameem_04_10_22_clk, '1', '1', '1', '1', Shameem_04_10_22_RSout, Shameem_04_10_22_MDRinput, Shameem_04_10_22_MDRoutput);
RT: Shameem_04_10_22_RT generic map (32) port map (Shameem_04_10_22_clk, '1', '1', '1', '1', Shameem_04_10_22_RTin, Shameem_04_10_22_RTout, Shameem_04_10_22_RTout);

Shameem_04_10_22_P1: process(Shameem_04_10_22_tmpl1, Shameem_04_10_22_tmpl2, Shameem_04_10_22_tmpl3, Shameem_04_10_22_tmpl4, Shameem_04_10_22_tmpl5
begin
    case Shameem_04_10_22_operation is

```

Messages:

- Command: quartus_edt --read_settings_files=off --write_settings_files=off Shameem_04_10_22_ALU -c Shameem_04_10_22_ALU
- 18236 Number of processors has not been specified which may cause overloading on shared machines. Set the global assignment NUM_PARALLEL_PROCESSORS in your QSF to an appropriate value for best performance.
- 204019 Generated file Shameem_04_10_22_ALU.vo in folder "/home/azwad/Desktop/Shameem_04_10_22_ALU/simulation/modelsim/" for EDA simulation tool
- Quartus Prime EDA Netlist Writer was successful. 0 errors, 1 warning
- 293000 Quartus Prime Full Compilation was successful. 0 errors, 310 warnings

Figure 24: Arithmetic Logic Unit VHDL code continued.

```

Shameem_04_10_22_P1: process(Shameem_04_10_22_tmp21, Shameem_04_10_22_tmp22, Shameem_04_10_22_tmp23, Shameem_04_10_22_tmp24, Shameem_04_10_22_tmp25)
begin
    case Shameem_04_10_22_operation is
        when "0000"=>
            Shameem_04_10_22_RDoutput <= Shameem_04_10_22_tmp21;
            Shameem_04_10_22_0 <= Shameem_04_10_22_tmp3;
            Shameem_04_10_22_N <= Shameem_04_10_22_tmp2;
            Shameem_04_10_22_Z <= Shameem_04_10_22_tmp1;
        when "0001"=>
            Shameem_04_10_22_RToutput <= Shameem_04_10_22_tmp22;
            Shameem_04_10_22_0 <= Shameem_04_10_22_tmp6;
            Shameem_04_10_22_N <= Shameem_04_10_22_tmp5;
            Shameem_04_10_22_Z <= Shameem_04_10_22_tmp4;
        when "0010"=>
            Shameem_04_10_22_RToutput <= Shameem_04_10_22_tmp23;
            Shameem_04_10_22_0 <= "0";
            Shameem_04_10_22_N <= Shameem_04_10_22_tmp8;
            Shameem_04_10_22_Z <= Shameem_04_10_22_tmp7;
        when "0011"=>
            Shameem_04_10_22_RDoutput <= Shameem_04_10_22_tmp24;
            Shameem_04_10_22_0 <= "0";
            Shameem_04_10_22_N <= Shameem_04_10_22_tmp10;
            Shameem_04_10_22_Z <= Z_tmp3;
        when "0100"=>
            Shameem_04_10_22_RDoutput <= Shameem_04_10_22_tmp25;
            Shameem_04_10_22_0 <= Shameem_04_10_22_tmp14;
            Shameem_04_10_22_N <= Shameem_04_10_22_tmp13;
            Shameem_04_10_22_Z <= Shameem_04_10_22_tmp12;
    end case;
end process;

```

Find what: tmp2 Replace with: 0_22_IMMoutput Look in: Current File Search: Down Find Next Replace Replace All Mark All

Type ID Message

- Command: quartus_edt --read_settings_files=off --write_settings_files=off Shameem_04_10_22_ALU -c Shameem_04_10_22_ALU
- 18236 Number of processors has not been specified which may cause overloading on shared machines. Set the global assignment NUM_PARALLEL_PROCESSORS in your QSF to an appropriate value for best performance.
- 204019 Generated file Shameem_04_10_22_ALU.vo in folder "/home/azwad/Desktop/Shameem_04_10_22_ALU/simulation/modelsim/" for EDA simulation tool
- Quartus Prime EDA Netlist Writer was successful. 0 errors, 1 warning
- 293000 Quartus Prime Full Compilation was successful. 0 errors, 310 warnings

Figure 25: Arithmetic Logic Unit VHDL code continued.

```

when "0101"=>
    Shameem_04_10_22_RDoutput <= Shameem_04_10_22_tmp26;
    Shameem_04_10_22_0 <= "0";
    Shameem_04_10_22_N <= Shameem_04_10_22_tmp16;
    Shameem_04_10_22_Z <= Shameem_04_10_22_tmp15;
when "0110" | "1000" | "1001" | "1011" | "1100" | "1101" =>
    Shameem_04_10_22_RToutput <= Shameem_04_10_22_tmp27;
    Shameem_04_10_22_0 <= Shameem_04_10_22_tmp20;
    Shameem_04_10_22_N <= Shameem_04_10_22_tmp19;
    Shameem_04_10_22_Z <= Shameem_04_10_22_tmp18;
when "0111" | "1010" =>
    Shameem_04_10_22_RToutput <= Shameem_04_10_22_tmp27;
    Shameem_04_10_22_0 <= Shameem_04_10_22_tmp20;
    Shameem_04_10_22_N <= Shameem_04_10_22_tmp19;
    Shameem_04_10_22_Z <= Shameem_04_10_22_tmp18;
when "1110" =>
    Shameem_04_10_22_RToutput <= Shameem_04_10_22_MARoutput;
when "1111" =>
    Shameem_04_10_22_RToutput <= Shameem_04_10_22_MDRoutput;
when others =>
    Shameem_04_10_22_RDoutput <= x"00000000";
end case;
end process;

```

Find what: tmp2 Replace with: 0_22_IMMoutput Look in: Current File Search: Down Find Next Replace Replace All Mark All

Type ID Message

- Command: quartus_edt --read_settings_files=off --write_settings_files=off Shameem_04_10_22_ALU -c Shameem_04_10_22_ALU
- 18236 Number of processors has not been specified which may cause overloading on shared machines. Set the global assignment NUM_PARALLEL_PROCESSORS in your QSF to an appropriate value for best performance.
- 204019 Generated file Shameem_04_10_22_ALU.vo in folder "/home/azwad/Desktop/Shameem_04_10_22_ALU/simulation/modelsim/" for EDA simulation tool
- Quartus Prime EDA Netlist Writer was successful. 0 errors, 1 warning

Figure 26: Arithmetic Logic Unit VHDL code continued.

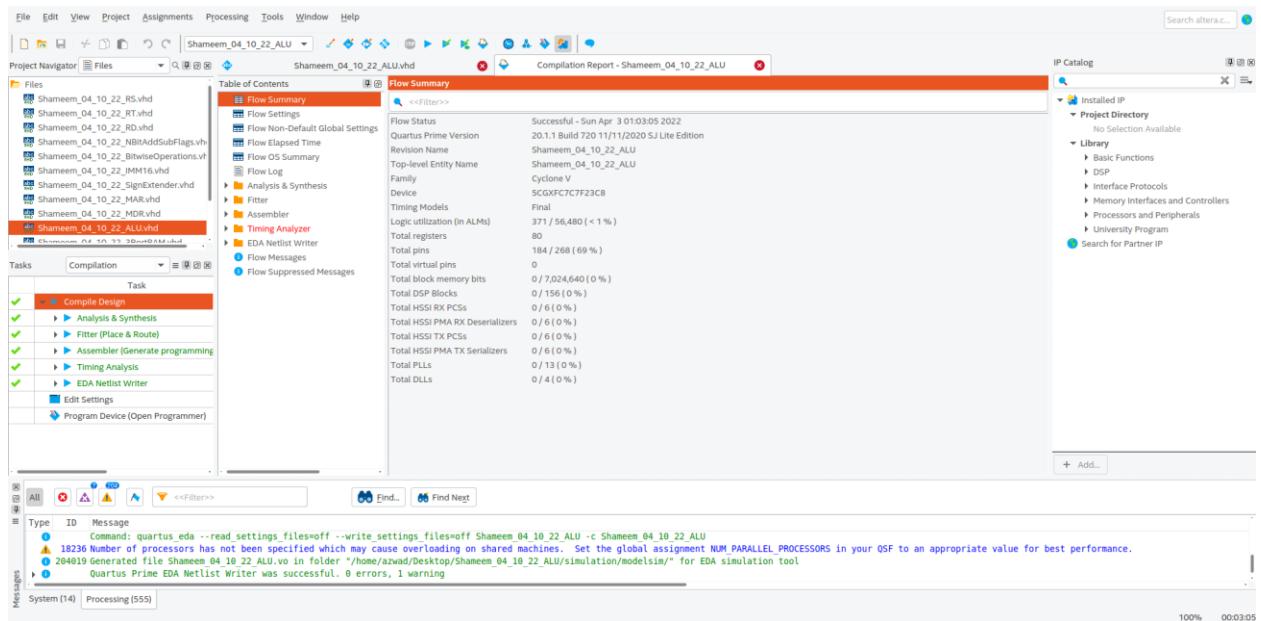


Figure 27: Arithmetic Logic Unit compilation report.

3 Ported RAM

```

use ieee.STD_LOGIC_1164.all;
use ieee.numeric_std.all;

entity Shameem_04_10_22_3PortRAM is
    port (
        signal Shameem_04_10_22_REGWR, Shameem_04_10_22_CLK: in STD_LOGIC;
        signal Shameem_04_10_22_Operation: in STD_LOGIC_VECTOR(3 downto 0);
        signal Shameem_04_10_22_RS, Shameem_04_10_22_RT: in STD_LOGIC_VECTOR(31 downto 0);
        signal Shameem_04_10_22_RDATA1, Shameem_04_10_22_RDATA2: out STD_LOGIC_VECTOR(31 downto 0)
    );
end entity Shameem_04_10_22_3PortRAM;

architecture arch of Shameem_04_10_22_3PortRAM is
begin
    process(Shameem_04_10_22_CLK, Shameem_04_10_22_Operation, Shameem_04_10_22_REGWR)
    begin
        if rising_edge(Shameem_04_10_22_CLK) and Shameem_04_10_22_REGWR = '1' then
            case Shameem_04_10_22_Operation is
                when "0000" =>
                    Shameem_04_10_22_R(0) <= Shameem_04_10_22_RS;
                    Shameem_04_10_22_R(1) <= Shameem_04_10_22_RT;
                when "0001" =>
                    Shameem_04_10_22_R(2) <= Shameem_04_10_22_RS;
                    Shameem_04_10_22_R(3) <= Shameem_04_10_22_RT;
                when "0010" =>
                    Shameem_04_10_22_R(4) <= Shameem_04_10_22_RS;
                    Shameem_04_10_22_R(5) <= Shameem_04_10_22_RT;
                when "0011" =>
                    Shameem_04_10_22_R(6) <= Shameem_04_10_22_RS;
                    Shameem_04_10_22_R(7) <= Shameem_04_10_22_RT;
            end case;
        end if;
    end process;
end;

```

Messages

Type	ID	Message
Info	1	Command: quartus eda --read_settings_files=off --write_settings_files=off Shameem_04_10_22_ALU -c Shameem_04_10_22_ALU
Warning	18236	Number of processors has not been specified which may cause overloading on shared machines. Set the global assignment NUM_PARALLEL_PROCESSORS in your QSF to an appropriate value for best performance.
Info	204019	Generated file Shameem_04_10_22_ALU.vo in folder '/home/azwad/Desktop/Shameem_04_10_22_ALU/simulation/modelsim/' for EDA simulation tool
Info	293000	Quartus Prime EDA Netlist Writer was successful. 0 errors, 1 warning
Info	293000	Quartus Prime Full Compilation was successful. 0 errors, 81 warnings

System (8) Processing (197)

Figure 28: 3 Ported RAM VHDL code compiled successfully on Quartus.

```

        when "0100" =>
            Shameem_04_10_22_R(8) <= Shameem_04_10_22_RS;
            Shameem_04_10_22_R(9) <= Shameem_04_10_22_RT;
        when "0101" =>
            Shameem_04_10_22_R(10) <= Shameem_04_10_22_RS;
            Shameem_04_10_22_R(11) <= Shameem_04_10_22_RT;
        when "0110" =>
            Shameem_04_10_22_R(12) <= Shameem_04_10_22_RS;
            Shameem_04_10_22_R(13) <= Shameem_04_10_22_RT;
        when "0111" =>
            Shameem_04_10_22_R(14) <= Shameem_04_10_22_RS;
            Shameem_04_10_22_R(15) <= Shameem_04_10_22_RT;
        when "1000" =>
            Shameem_04_10_22_R(16) <= Shameem_04_10_22_RS;
            Shameem_04_10_22_R(17) <= Shameem_04_10_22_RT;
        when "1001" =>
            Shameem_04_10_22_R(18) <= Shameem_04_10_22_RS;
            Shameem_04_10_22_R(19) <= Shameem_04_10_22_RT;
    end case;
end if;
end process;
end;

```

Messages

Type	ID	Message
Info	1	Command: quartus eda --read_settings_files=off --write_settings_files=off Shameem_04_10_22_ALU -c Shameem_04_10_22_ALU
Warning	18236	Number of processors has not been specified which may cause overloading on shared machines. Set the global assignment NUM_PARALLEL_PROCESSORS in your QSF to an appropriate value for best performance.
Info	204019	Generated file Shameem_04_10_22_ALU.vo in folder '/home/azwad/Desktop/Shameem_04_10_22_ALU/simulation/modelsim/' for EDA simulation tool
Info	293000	Quartus Prime EDA Netlist Writer was successful. 0 errors, 1 warning
Info	293000	Quartus Prime Full Compilation was successful. 0 errors, 81 warnings

System (8) Processing (197)

Figure 29: 3 Ported RAM VHDL code continued.

The screenshot shows the Quartus Prime software interface with the following details:

- File Menu:** File, Edit, View, Project, Assignments, Processing, Tools, Window, Help.
- Project Navigator:** Shows files: Shameem_04_10_22_RS.vhd, Shameem_04_10_22_RT.vhd, Shameem_04_10_22_RD.vhd, Shameem_04_10_22_NBItAddSubFlags.vhd, Shameem_04_10_22_BitwiseOperations.vhd, Shameem_04_10_22_IMM16.vhd, Shameem_04_10_22_SignExtender.vhd, Shameem_04_10_22_MAR.vhd, Shameem_04_10_22_MDR.vhd, Shameem_04_10_22_ALU.vhd, Shameem_04_10_22_3PortRAM.vhd.
- Code Editor:** Displays VHDL code for a 3-ported RAM. The code defines a process `Shameem_04_10_22_ReadReg1` that takes `Shameem_04_10_22_CLK` and `Shameem_04_10_22_Operation` as inputs. It uses a case statement based on the operation to assign values from `Shameem_04_10_22_R(0)` to `Shameem_04_10_22_R(26)` to `Shameem_04_10_22_RDATA1`.
- Tasks:** Shows the task `Compile Design` is selected.
- IP Catalog:** Shows sections for Installed IP, Project Directory (No Selection Available), Library (Basic Functions, DSP, Interface Protocols, Memory Interfaces and Controllers, Processors and Peripherals, University Program), and Search for Partner IP.
- Messages:** Shows compilation messages:
 - Command: quartus_edt --read_settings_files=off --write_settings_files=off Shameem_04_10_22_ALU -c Shameem_04_10_22_ALU
 - 18236 Number of processors has not been specified which may cause overloading on shared machines. Set the global assignment NUM_PARALLEL_PROCESSORS in your QSF to an appropriate value for best performance.
 - 204019 Generated file Shameem_04_10_22_ALU.vo in folder "/home/azwad/Desktop/Shameem_04_10_22_ALU/simulation/modelsim/" for EDA simulation tool
 - Quartus Prime EDA Netlist Writer was successful. 0 errors, 1 warning
 - 293800 Quartus Prime Full Compilation was successful. 0 errors, 81 warnings

Figure 30: 3 Ported RAM VHDL code continued.

The screenshot shows the Quartus Prime software interface with the following details:

- File Menu:** File, Edit, View, Project, Assignments, Processing, Tools, Window, Help.
- Project Navigator:** Shows files: Shameem_04_10_22_RS.vhd, Shameem_04_10_22_RT.vhd, Shameem_04_10_22_RD.vhd, Shameem_04_10_22_NBItAddSubFlags.vhd, Shameem_04_10_22_BitwiseOperations.vhd, Shameem_04_10_22_IMM16.vhd, Shameem_04_10_22_SignExtender.vhd, Shameem_04_10_22_MAR.vhd, Shameem_04_10_22_MDR.vhd, Shameem_04_10_22_ALU.vhd, Shameem_04_10_22_3PortRAM.vhd.
- Code Editor:** Displays VHDL code for a 3-ported RAM. The code defines a process `Shameem_04_10_22_ReadReg1` that takes `Shameem_04_10_22_CLK` and `Shameem_04_10_22_Operation` as inputs. It uses a case statement based on the operation to assign values from `Shameem_04_10_22_R(0)` to `Shameem_04_10_22_R(26)` to `Shameem_04_10_22_RDATA1`.
- Tasks:** Shows the task `Compile Design` is selected.
- IP Catalog:** Shows sections for Installed IP, Project Directory (No Selection Available), Library (Basic Functions, DSP, Interface Protocols, Memory Interfaces and Controllers, Processors and Peripherals, University Program), and Search for Partner IP.
- Messages:** Shows compilation messages:
 - Command: quartus_edt --read_settings_files=off --write_settings_files=off Shameem_04_10_22_ALU -c Shameem_04_10_22_ALU
 - 18236 Number of processors has not been specified which may cause overloading on shared machines. Set the global assignment NUM_PARALLEL_PROCESSORS in your QSF to an appropriate value for best performance.
 - 204019 Generated file Shameem_04_10_22_ALU.vo in folder "/home/azwad/Desktop/Shameem_04_10_22_ALU/simulation/modelsim/" for EDA simulation tool
 - Quartus Prime EDA Netlist Writer was successful. 0 errors, 1 warning
 - 293800 Quartus Prime Full Compilation was successful. 0 errors, 81 warnings

Figure 31: 3 Ported RAM VHDL code continued.

```

Shameem_04_10_22_RS.vhd
Shameem_04_10_22_RT.vhd
Shameem_04_10_22_RD.vhd
Shameem_04_10_22_NBitAddSubFlags.vhd
Shameem_04_10_22_BitwiseOperations.vhd
Shameem_04_10_22_IMM16.vhd
Shameem_04_10_22_SignExtender.vhd
Shameem_04_10_22_MAR.vhd
Shameem_04_10_22_MDR.vhd
Shameem_04_10_22_ALU.vhd
Shameem_04_10_22_3PortRAM.vhd

Shameem_04_10_22_RS.vhd
Shameem_04_10_22_RT.vhd
Shameem_04_10_22_RD.vhd
Shameem_04_10_22_NBitAddSubFlags.vhd
Shameem_04_10_22_BitwiseOperations.vhd
Shameem_04_10_22_IMM16.vhd
Shameem_04_10_22_SignExtender.vhd
Shameem_04_10_22_MAR.vhd
Shameem_04_10_22_MDR.vhd
Shameem_04_10_22_ALU.vhd
Shameem_04_10_22_3PortRAM.vhd

Shameem_04_10_22_3PortRAM.vhd
Compilation Report - Shameem_04_10_22_ALU
IP Catalog

File Edit View Project Assignments Processing Tools Window Help
Search altera.c

Project Navigator Files
Shameem_04_10_22_3PortRAM.vhd
Compilation Report - Shameem_04_10_22_ALU
IP Catalog

Files
Shameem_04_10_22_RS.vhd
Shameem_04_10_22_RT.vhd
Shameem_04_10_22_RD.vhd
Shameem_04_10_22_NBitAddSubFlags.vhd
Shameem_04_10_22_BitwiseOperations.vhd
Shameem_04_10_22_IMM16.vhd
Shameem_04_10_22_SignExtender.vhd
Shameem_04_10_22_MAR.vhd
Shameem_04_10_22_MDR.vhd
Shameem_04_10_22_ALU.vhd
Shameem_04_10_22_3PortRAM.vhd

Tasks Compilation
Task
Compile Design
Analysis & Synthesis
Fitter (Place & Route)
Assembler (Generate program)
Timing Analysis
EDA Netlist Writer
Edit Settings

Messages
All Find... Find Next
Type ID Message
1 Command: quartus eda --read settings files=off --write settings files=off Shameem_04_10_22_ALU -c Shameem_04_10_22_ALU
A 18236 Number of processors has not been specified which may cause overloading on shared machines. Set the global assignment NUM_PARALLEL_PROCESSORS in your QSF to an appropriate value for best performance.
1 204019 Generated file Shameem_04_10_22_ALU.vo in folder "/home/azwad/Desktop/Shameem_04_10_22_ALU/simulation/modelsim/" for EDA simulation tool
1 Quartus Prime EDA Netlist Writer was successful. 0 errors, 1 warning
1 293000 Quartus Prime Full Compilation was successful. 0 errors, 81 warnings

System (8) Processing (197)
100% 00:01:05

```

Figure 32: 3 Ported RAM VHDL code continued.

```

Shameem_04_10_22_RS.vhd
Shameem_04_10_22_RT.vhd
Shameem_04_10_22_RD.vhd
Shameem_04_10_22_NBitAddSubFlags.vhd
Shameem_04_10_22_BitwiseOperations.vhd
Shameem_04_10_22_IMM16.vhd
Shameem_04_10_22_SignExtender.vhd
Shameem_04_10_22_MAR.vhd
Shameem_04_10_22_MDR.vhd
Shameem_04_10_22_ALU.vhd
Shameem_04_10_22_3PortRAM.vhd

Shameem_04_10_22_RS.vhd
Shameem_04_10_22_RT.vhd
Shameem_04_10_22_RD.vhd
Shameem_04_10_22_NBitAddSubFlags.vhd
Shameem_04_10_22_BitwiseOperations.vhd
Shameem_04_10_22_IMM16.vhd
Shameem_04_10_22_SignExtender.vhd
Shameem_04_10_22_MAR.vhd
Shameem_04_10_22_MDR.vhd
Shameem_04_10_22_ALU.vhd
Shameem_04_10_22_3PortRAM.vhd

Shameem_04_10_22_3PortRAM.vhd
Compilation Report - Shameem_04_10_22_ALU
IP Catalog

File Edit View Project Assignments Processing Tools Window Help
Search altera.c

Project Navigator Files
Shameem_04_10_22_3PortRAM.vhd
Compilation Report - Shameem_04_10_22_ALU
IP Catalog

Files
Shameem_04_10_22_RS.vhd
Shameem_04_10_22_RT.vhd
Shameem_04_10_22_RD.vhd
Shameem_04_10_22_NBitAddSubFlags.vhd
Shameem_04_10_22_BitwiseOperations.vhd
Shameem_04_10_22_IMM16.vhd
Shameem_04_10_22_SignExtender.vhd
Shameem_04_10_22_MAR.vhd
Shameem_04_10_22_MDR.vhd
Shameem_04_10_22_ALU.vhd
Shameem_04_10_22_3PortRAM.vhd

Tasks Compilation
Task
Compile Design
Analysis & Synthesis
Fitter (Place & Route)
Assembler (Generate program)
Timing Analysis
EDA Netlist Writer
Edit Settings

Messages
All Find... Find Next
Type ID Message
1 Command: quartus eda --read settings files=off --write settings files=off Shameem_04_10_22_ALU -c Shameem_04_10_22_ALU
A 18236 Number of processors has not been specified which may cause overloading on shared machines. Set the global assignment NUM_PARALLEL_PROCESSORS in your QSF to an appropriate value for best performance.
1 204019 Generated file Shameem_04_10_22_ALU.vo in folder "/home/azwad/Desktop/Shameem_04_10_22_ALU/simulation/modelsim/" for EDA simulation tool
1 Quartus Prime EDA Netlist Writer was successful. 0 errors, 1 warning
1 293000 Quartus Prime Full Compilation was successful. 0 errors, 81 warnings

System (8) Processing (197)
100% 00:01:05

```

Figure 33: 3 Ported RAM VHDL code continued.

```

187      when "1111" =>
188          Shameem_04_10_22_RDATA1 <= Shameem_04_10_22_R(30);
189      when others => null;
190  end case;
191 end process;
192
193 Shameem_04_10_22_ReadReg2 : process(Shameem_04_10_22_CLK, Shameem_04_10_22_Operation)
194 begin
195     case Shameem_04_10_22_Operation is
196         when "0000" =>
197             Shameem_04_10_22_RDATA2 <= Shameem_04_10_22_R(1);
198         when "0001" =>
199             Shameem_04_10_22_RDATA2 <= Shameem_04_10_22_R(3);
200         when "0010" =>
201             Shameem_04_10_22_RDATA2 <= Shameem_04_10_22_R(5);
202         when "0011" =>
203             Shameem_04_10_22_RDATA2 <= Shameem_04_10_22_R(7);
204         when "1000" =>
205             Shameem_04_10_22_RDATA2 <= Shameem_04_10_22_R(9);
206         when "1001" =>
207             Shameem_04_10_22_RDATA2 <= Shameem_04_10_22_R(11);
208         when "1010" =>
209             Shameem_04_10_22_RDATA2 <= Shameem_04_10_22_R(13);
210         when "1011" =>
211             Shameem_04_10_22_RDATA2 <= Shameem_04_10_22_R(15);
212         when "1100" =>
213             Shameem_04_10_22_RDATA2 <= Shameem_04_10_22_R(17);
214         when "1101" =>
215             Shameem_04_10_22_RDATA2 <= Shameem_04_10_22_R(19);
216         when "1110" =>
217             Shameem_04_10_22_RDATA2 <= Shameem_04_10_22_R(21);
218         when "1111" =>
219             Shameem_04_10_22_RDATA2 <= Shameem_04_10_22_R(23);
220     end case;
221 end process;
222
223 Shameem_04_10_22_ALU: entity work.Shameem_04_10_22_ALU port map(
224     Address1,
225     Address2,
226     Address3,
227     RDATAREAD1,
228     RDATAREAD2,
229     RDATAREAD3,
230     RDATAWRITE1,
231     RDATAWRITE2,
232     RDATAWRITE3,
233     RDATAREAD,
234     RDATAWRITE
235 );

```

Messages

- Command: quartus eda --read settings_files=off --write settings_files=off Shameem_04_10_22_ALU -c Shameem_04_10_22_ALU
- 18236 Number of processors has not been specified which may cause overloading on shared machines. Set the global assignment NUM_PARALLEL_PROCESSORS in your QSF to an appropriate value for best performance.
- 204019 Generated file Shameem_04_10_22_ALU.vo in folder "/home/azwad/Desktop/Shameem_04_10_22_ALU/simulation/modelsim/" for EDA simulation tool
- Quartus Prime EDA Netlist Writer was successful. 0 errors, 1 warning
- 293000 Quartus Prime Full Compilation was successful. 0 errors, 81 warnings

Figure 34: 3 Ported RAM VHDL code continued.

```

116      Shameem_04_10_22_RDATA2 <= Shameem_04_10_22_R(1);
117      when "0001" =>
118          Shameem_04_10_22_RDATA2 <= Shameem_04_10_22_R(3);
119      when "0010" =>
120          Shameem_04_10_22_RDATA2 <= Shameem_04_10_22_R(5);
121      when "0011" =>
122          Shameem_04_10_22_RDATA2 <= Shameem_04_10_22_R(7);
123      when "0100" =>
124          Shameem_04_10_22_RDATA2 <= Shameem_04_10_22_R(9);
125      when "0101" =>
126          Shameem_04_10_22_RDATA2 <= Shameem_04_10_22_R(11);
127      when "0110" =>
128          Shameem_04_10_22_RDATA2 <= Shameem_04_10_22_R(13);
129      when "0111" =>
130          Shameem_04_10_22_RDATA2 <= Shameem_04_10_22_R(15);
131      when "1000" =>
132          Shameem_04_10_22_RDATA2 <= Shameem_04_10_22_R(17);
133      when "1001" =>
134          Shameem_04_10_22_RDATA2 <= Shameem_04_10_22_R(19);
135      when "1010" =>
136          Shameem_04_10_22_RDATA2 <= Shameem_04_10_22_R(21);
137      when "1011" =>
138          Shameem_04_10_22_RDATA2 <= Shameem_04_10_22_R(23);
139      when "1100" =>
140          Shameem_04_10_22_RDATA2 <= Shameem_04_10_22_R(25);
141      when "1101" =>
142          Shameem_04_10_22_RDATA2 <= Shameem_04_10_22_R(27);
143      when "1110" =>
144          Shameem_04_10_22_RDATA2 <= Shameem_04_10_22_R(29);
145      when "1111" =>
146          Shameem_04_10_22_RDATA2 <= Shameem_04_10_22_R(31);
147      when others => null;

```

Messages

- Command: quartus eda --read settings_files=off --write settings_files=off Shameem_04_10_22_ALU -c Shameem_04_10_22_ALU
- 18236 Number of processors has not been specified which may cause overloading on shared machines. Set the global assignment NUM_PARALLEL_PROCESSORS in your QSF to an appropriate value for best performance.
- 204019 Generated file Shameem_04_10_22_ALU.vo in folder "/home/azwad/Desktop/Shameem_04_10_22_ALU/simulation/modelsim/" for EDA simulation tool
- Quartus Prime EDA Netlist Writer was successful. 0 errors, 1 warning
- 293000 Quartus Prime Full Compilation was successful. 0 errors, 81 warnings

Figure 35: 3 Ported RAM VHDL code continued.

File Edit View Project Assignments Processing Tools Window Help

Project Navigator Files Compilation Report - Shameem_04_10_22_ALU

IP Catalog

Files

- Shameem_04_10_22_RS.vhd
- Shameem_04_10_22_RT.vhd
- Shameem_04_10_22_RD.vhd
- Shameem_04_10_22_NBAddSubFlags.vhd
- Shameem_04_10_22_BitwiseOperations.vhd
- Shameem_04_10_22_IMM16.vhd
- Shameem_04_10_22_SignExtender.vhd
- Shameem_04_10_22_MAR.vhd
- Shameem_04_10_22_MDR.vhd
- Shameem_04_10_22_ALU.vhd
- Shameem_04_10_22_3PortRAM.vhd

Tasks Compilation

Task

Compile Design

Analysis & Synthesis

Fitter (Place & Route)

Assembler (Generate programming)

Timing Analysis

EDA Netlist Writer

Edit Settings

119 when "0010" =>
Shameem_04_10_22_RDATA2 <= Shameem_04_10_22_R(5);
when "0011" =>
Shameem_04_10_22_RDATA2 <= Shameem_04_10_22_R(7);
when "0100" =>
Shameem_04_10_22_RDATA2 <= Shameem_04_10_22_R(9);
when "0101" =>
Shameem_04_10_22_RDATA2 <= Shameem_04_10_22_R(11);
when "0110" =>
Shameem_04_10_22_RDATA2 <= Shameem_04_10_22_R(13);
when "0111" =>
Shameem_04_10_22_RDATA2 <= Shameem_04_10_22_R(15);
when "1000" =>
Shameem_04_10_22_RDATA2 <= Shameem_04_10_22_R(17);
when "1001" =>
Shameem_04_10_22_RDATA2 <= Shameem_04_10_22_R(19);
when "1010" =>
Shameem_04_10_22_RDATA2 <= Shameem_04_10_22_R(21);
when "1011" =>
Shameem_04_10_22_RDATA2 <= Shameem_04_10_22_R(23);
when "1100" =>
Shameem_04_10_22_RDATA2 <= Shameem_04_10_22_R(25);
when "1101" =>
Shameem_04_10_22_RDATA2 <= Shameem_04_10_22_R(27);
when "1110" =>
Shameem_04_10_22_RDATA2 <= Shameem_04_10_22_R(29);
when "1111" =>
Shameem_04_10_22_RDATA2 <= Shameem_04_10_22_R(31);
when others => null;
end case;
end process;
end architecture ARCH;

All Find... Find Next

Type ID Message

- Command: quartus_edt --read_settings_files=off --write_settings_files=off Shameem_04_10_22_ALU -c Shameem_04_10_22_ALU
- 18236 Number of processors has not been specified which may cause overloading on shared machines. Set the global assignment NUM_PARALLEL_PROCESSORS in your QSF to an appropriate value for best performance.
- 204019 Generated file Shameem_04_10_22_ALU.vo in folder "/home/azwad/Desktop/Shameem_04_10_22_ALU/simulation/modelsim/" for EDA simulation tool
- 293000 Quartus Prime EDA Netlist Writer was successful. 0 errors, 1 warning
- 293000 Quartus Prime Full Compilation was successful. 0 errors, 81 warnings

System (8) Processing (197)

Ln 106 Col 63 VHDL File

100% 00:01:05

Figure 36: 3 Ported RAM VHDL code continued.

File Edit View Project Assignments Processing Tools Window Help

Project Navigator Files Compilation Report - Shameem_04_10_22_ALU

IP Catalog

Files

- Shameem_04_10_22_RS.vhd
- Shameem_04_10_22_RT.vhd
- Shameem_04_10_22_RD.vhd
- Shameem_04_10_22_NBAddSubFlags.vhd
- Shameem_04_10_22_BitwiseOperations.vhd
- Shameem_04_10_22_IMM16.vhd
- Shameem_04_10_22_SignExtender.vhd
- Shameem_04_10_22_MAR.vhd
- Shameem_04_10_22_MDR.vhd
- Shameem_04_10_22_ALU.vhd
- Shameem_04_10_22_3PortRAM.vhd

Tasks Compilation

Task

Compile Design

Analysis & Synthesis

Fitter (Place & Route)

Assembler (Generate programming)

Timing Analysis

EDA Netlist Writer

Edit Settings

Table of Contents Flow Summary

Flow Summary

Flow Status: Successful - Sat Apr 2 17:35:29 2022

Quartus Prime Version: 20.1 Build 720 11/1/2020 SJ Lite Edition

Revision Name: Shameem_04_10_22_ALU

Top-level Entity Name: Shameem_04_10_22_3PortRAM

Family: Cyclone V

Device: 5CGXF7C7F23C8

Timing Models: Final

Logic utilization (in ALMs): 469 / 56,480 (< 1 %)

Total registers: 1024

Total pins: 166 / 268 (62 %)

Total virtual pins: 0

Total block memory bits: 0 / 7,023,640 (0 %)

Total DSP Blocks: 0 / 156 (0 %)

Total HSSI RX PCGs: 0 / 6 (0 %)

Total HSSI PMA RX Deserializers: 0 / 6 (0 %)

Total HSSI TX PCGs: 0 / 6 (0 %)

Total HSSI PMA TX Serializers: 0 / 6 (0 %)

Total PLLs: 0 / 13 (0 %)

Total DLLs: 0 / 4 (0 %)

All Find... Find Next

Type ID Message

- Command: quartus_edt --read_settings_files=off --write_settings_files=off Shameem_04_10_22_ALU -c Shameem_04_10_22_ALU
- 18236 Number of processors has not been specified which may cause overloading on shared machines. Set the global assignment NUM_PARALLEL_PROCESSORS in your QSF to an appropriate value for best performance.
- 204019 Generated file Shameem_04_10_22_ALU.vo in folder "/home/azwad/Desktop/Shameem_04_10_22_ALU/simulation/modelsim/" for EDA simulation tool
- 293000 Quartus Prime EDA Netlist Writer was successful. 0 errors, 1 warning
- 293000 Quartus Prime Full Compilation was successful. 0 errors, 79 warnings

System (9) Processing (165)

100% 00:02:42

Figure 37: 3 Ported RAM compilation report

Instruction Register

The screenshot shows the Quartus Prime interface with the following details:

- File Menu:** File, Edit, View, Project, Assignments, Processing, Tools, Window, Help.
- Project Navigator:** Shows files including `Shameem_04_10_22_RS.vhd`, `Shameem_04_10_22_RT.vhd`, `Shameem_04_10_22_RD.vhd`, `Shameem_04_10_22_NBitAddSubFlags.vhd`, `Shameem_04_10_22_BitwiseOperations.vhd`, `Shameem_04_10_22_IMM16.vhd`, `Shameem_04_10_22_SignExtender.vhd`, `Shameem_04_10_22_MAR.vhd`, `Shameem_04_10_22_MDR.vhd`, `Shameem_04_10_22_ALU.vhd`, `Shameem_04_10_22_3PortRAM.vhd`, and `Shameem_04_10_22_InstructionRegister.vhd`.
- Tasks:** Compilation, Task, Compile Design (highlighted), Analysis & Synthesis, Fitter (Place & Route), Assembler (Generate program), Timing Analysis, EDA Netlist Writer, Edit Settings.
- Code Editor:** Displays the VHDL code for the `Shameem_04_10_22_InstructionRegister` entity. The code defines a memory array for instructions and handles writes based on operation codes.
- Compilation Report:** Shows success messages including "Quartus Prime EDA Netlist Writer was successful. 0 errors, 1 warning" and "293000 Quartus Prime Full Compilation was successful. 0 errors, 233 warnings".
- IP Catalog:** Shows installed IP and project directory.
- Messages:** Shows system and processing messages.
- Bottom Status:** 100% 00:03:33.

Figure 38: Instruction Register VHDL code compiled successfully in Quartus.

The screenshot shows the Quartus Prime interface with the following details:

- File Menu:** File, Edit, View, Project, Assignments, Processing, Tools, Window, Help.
- Project Navigator:** Shows files including `Shameem_04_10_22_RS.vhd`, `Shameem_04_10_22_RT.vhd`, `Shameem_04_10_22_RD.vhd`, `Shameem_04_10_22_NBitAddSubFlags.vhd`, `Shameem_04_10_22_BitwiseOperations.vhd`, `Shameem_04_10_22_IMM16.vhd`, `Shameem_04_10_22_SignExtender.vhd`, `Shameem_04_10_22_MAR.vhd`, `Shameem_04_10_22_MDR.vhd`, `Shameem_04_10_22_ALU.vhd`, `Shameem_04_10_22_3PortRAM.vhd`, and `Shameem_04_10_22_InstructionRegister.vhd`.
- Tasks:** Compilation, Task, Compile Design (highlighted), Analysis & Synthesis, Fitter (Place & Route), Assembler (Generate program), Timing Analysis, EDA Netlist Writer, Edit Settings.
- Code Editor:** Displays the continuation of the VHDL code for the `Shameem_04_10_22_InstructionRegister` entity, specifically handling cases for operation codes 0010 through 0101.
- Compilation Report:** Shows success messages including "Quartus Prime EDA Netlist Writer was successful. 0 errors, 1 warning" and "293000 Quartus Prime Full Compilation was successful. 0 errors, 233 warnings".
- IP Catalog:** Shows installed IP and project directory.
- Messages:** Shows system and processing messages.
- Bottom Status:** 100% 00:03:33.

Figure 39: Instruction Register VHDL code continued.

```

File Edit View Project Assignments Processing Tools Window Help
Project Navigator Files Search altera...
Shameem_04_10_22_ALU
Shameem_04_10_22_InstructionRegister.vhd Compilation Report - Shameem_04_10_22_ALU
IP Catalog
Installed IP
Project Directory
No Selection Available
Library
Basic Functions
DSP
Interface Protocols
Memory Interfaces and Controllers
Processors and Peripherals
University Program
Search for Partner IP

Files
Shameem_04_10_22_RS.vhd
Shameem_04_10_22_RT.vhd
Shameem_04_10_22_RD.vhd
Shameem_04_10_22_NBitAddSubFlags.vhd
Shameem_04_10_22_BitwiseOperations.vhd
Shameem_04_10_22_IMM16.vhd
Shameem_04_10_22_SignExtender.vhd
Shameem_04_10_22_MAR.vhd
Shameem_04_10_22_MDR.vhd
Shameem_04_10_22_ALU.vhd
Shameem_04_10_22_3PortRAM.vhd
Shameem_04_10_22_InstructionRegister.vhd

Tasks Compilation
Task
Compile Design (✓)
Analysis & Synthesis (✓)
Fitter (Place & Route) (✓)
Assembler (Generate programmr) (✓)
Timing Analysis (✓)
EDA Netlist Writer (✓)
Edit Settings (✓)

Type ID Message
1 Command: quartus_edt --read_settings_files=off --write_settings_files=off Shameem_04_10_22_ALU -c Shameem_04_10_22_ALU
2 18236 Number of processors has not been specified which may cause overloading on shared machines. Set the global assignment NUM_PARALLEL_PROCESSORS in your QSF to an appropriate value for best performance.
3 204019 Generated file Shameem_04_10_22_ALU.vo in folder "/home/azwad/Desktop/Shameem_04_10_22_ALU/simulation/modelsim/" for EDA simulation tool
4 Quartus Prime EDA Netlist Writer was successful. 0 errors, 1 warning
5 293000 Quartus Prime Full Compilation was successful. 0 errors, 233 warnings

System (11) Processing (316)

```

Figure 40 shows the VHDL code for the Instruction Register. The code defines a process for reading instruction bits from IR(8) to IR(0). The process uses a case statement based on the value of IR(8). The case branches are labeled with binary values: "0011", "0100", "0101", "0110", "0111", "1000", "1001", and "1010". Each branch contains assignments for RS, RD, RT, and IMM based on the value of IR(9) through IR(0). The code also includes comments and some unused assignments.

Figure 40: Instruction Register VHDL code continued.

```

File Edit View Project Assignments Processing Tools Window Help
Project Navigator Files Search altera...
Shameem_04_10_22_ALU
Shameem_04_10_22_InstructionRegister.vhd Compilation Report - Shameem_04_10_22_ALU
IP Catalog
Installed IP
Project Directory
No Selection Available
Library
Basic Functions
DSP
Interface Protocols
Memory Interfaces and Controllers
Processors and Peripherals
University Program
Search for Partner IP

Files
Shameem_04_10_22_RS.vhd
Shameem_04_10_22_RT.vhd
Shameem_04_10_22_RD.vhd
Shameem_04_10_22_NBitAddSubFlags.vhd
Shameem_04_10_22_BitwiseOperations.vhd
Shameem_04_10_22_IMM16.vhd
Shameem_04_10_22_SignExtender.vhd
Shameem_04_10_22_MAR.vhd
Shameem_04_10_22_MDR.vhd
Shameem_04_10_22_ALU.vhd
Shameem_04_10_22_3PortRAM.vhd
Shameem_04_10_22_InstructionRegister.vhd

Tasks Compilation
Task
Compile Design (✓)
Analysis & Synthesis (✓)
Fitter (Place & Route) (✓)
Assembler (Generate programmr) (✓)
Timing Analysis (✓)
EDA Netlist Writer (✓)
Edit Settings (✓)

Type ID Message
1 Command: quartus_edt --read_settings_files=off --write_settings_files=off Shameem_04_10_22_ALU -c Shameem_04_10_22_ALU
2 18236 Number of processors has not been specified which may cause overloading on shared machines. Set the global assignment NUM_PARALLEL_PROCESSORS in your QSF to an appropriate value for best performance.
3 204019 Generated file Shameem_04_10_22_ALU.vo in folder "/home/azwad/Desktop/Shameem_04_10_22_ALU/simulation/modelsim/" for EDA simulation tool
4 Quartus Prime EDA Netlist Writer was successful. 0 errors, 1 warning
5 293000 Quartus Prime Full Compilation was successful. 0 errors, 233 warnings

System (11) Processing (316)

```

Figure 41 shows the VHDL code for the Instruction Register. The code defines a process for reading instruction bits from IR(24) to IR(0). The process uses a case statement based on the value of IR(24). The case branches are labeled with binary values: "1001", "1010", "1011", "1100", "1101", "1110", and "1111". Each branch contains assignments for RS, RD, RT, and IMM based on the value of IR(23) through IR(0). The code also includes comments and some unused assignments.

Figure 41: Instruction Register VHDL code continued.

```

Shameem_04_10_22_RS.vhd
Shameem_04_10_22_RT.vhd
Shameem_04_10_22_RD.vhd
Shameem_04_10_22_NBitAddSubFlags.vhd
Shameem_04_10_22_BitwiseOperations.vhd
Shameem_04_10_22_IMM16.vhd
Shameem_04_10_22_SignExtender.vhd
Shameem_04_10_22_MAR.vhd
Shameem_04_10_22_MDR.vhd
Shameem_04_10_22_ALU.vhd
Shameem_04_10_22_3PortRAM.vhd
Shameem_04_10_22_InstructionRegister.vhd

Shameem_04_10_22_IR(33)(3 downto 0) <= Shameem_04_10_22_Operation(3 downto 0);
Shameem_04_10_22_IR(33)(31 downto 0) <= (others => '0');
Shameem_04_10_22_IR(34) <= Shameem_04_10_22_RS(31 downto 0);
Shameem_04_10_22_IR(35) <= Shameem_04_10_22_IMM(31 downto 0);

when "1100" =>
  Shameem_04_10_22_IR(36)(3 downto 0) <= Shameem_04_10_22_Operation(3 downto 0);
  Shameem_04_10_22_IR(36)(31 downto 4) <= (others => '0');
  Shameem_04_10_22_IR(37) <= Shameem_04_10_22_RS(31 downto 0);
  Shameem_04_10_22_IR(38) <= Shameem_04_10_22_IMM(31 downto 0);

when "1101" =>
  Shameem_04_10_22_IR(39)(3 downto 0) <= Shameem_04_10_22_Operation(3 downto 0);
  Shameem_04_10_22_IR(39)(31 downto 4) <= (others => '0');
  Shameem_04_10_22_IR(40) <= Shameem_04_10_22_RS(31 downto 0);
  Shameem_04_10_22_IR(41) <= Shameem_04_10_22_IMM(31 downto 0);

when "1110" =>
  Shameem_04_10_22_IR(42)(3 downto 0) <= Shameem_04_10_22_Operation(3 downto 0);
  Shameem_04_10_22_IR(42)(31 downto 4) <= (others => '0');
  Shameem_04_10_22_IR(43) <= Shameem_04_10_22_RS(31 downto 0);
  Shameem_04_10_22_IR(44) <= Shameem_04_10_22_IMM(31 downto 0);

when "1111" =>
  Shameem_04_10_22_IR(45)(3 downto 0) <= Shameem_04_10_22_Operation(3 downto 0);
  Shameem_04_10_22_IR(45)(31 downto 4) <= (others => '0');
  Shameem_04_10_22_IR(46) <= Shameem_04_10_22_RS(31 downto 0);
  Shameem_04_10_22_IR(47) <= Shameem_04_10_22_IMM(31 downto 0);

when others => null;
end case;
end if;
end process;

Shameem_04_10_22_readMem : process(Shameem_04_10_22_clk, Shameem_04_10_22_Operation)
begin
  case Shameem_04_10_22_Operation is
    when "0000" =>

```

Messages

- Command: quartus_edt --read_settings_files=off --write_settings_files=off Shameem_04_10_22_ALU -c Shameem_04_10_22_ALU
- 18236 Number of processors has not been specified which may cause overloading on shared machines. Set the global assignment NUM_PARALLEL_PROCESSORS in your QSF to an appropriate value for best performance.
- 204019 Generated file Shameem_04_10_22_ALU.vo in folder "/home/azwad/Desktop/Shameem_04_10_22_ALU/simulation/modelsim/" for EDA simulation tool
- Quartus Prime EDA Netlist Writer was successful. 0 errors, 1 warning
- 293000 Quartus Prime Full Compilation was successful. 0 errors, 233 warnings

Figure 42: Instruction Register VHDL code continued.

```

      end case;
    when "0001" =>
      Shameem_04_10_22_INSTRUCTION(67 downto 64) <= Shameem_04_10_22_IR(0)(3 downto 0);
      Shameem_04_10_22_INSTRUCTION(63 downto 32) <= Shameem_04_10_22_IR(1);
      Shameem_04_10_22_INSTRUCTION(31 downto 0) <= Shameem_04_10_22_IR(2);
    when "0001" =>
      Shameem_04_10_22_INSTRUCTION(67 downto 64) <= Shameem_04_10_22_IR(3)(3 downto 0);
      Shameem_04_10_22_INSTRUCTION(63 downto 32) <= Shameem_04_10_22_IR(4);
      Shameem_04_10_22_INSTRUCTION(31 downto 0) <= Shameem_04_10_22_IR(5);
    when "0010" =>
      Shameem_04_10_22_INSTRUCTION(67 downto 64) <= Shameem_04_10_22_IR(6)(3 downto 0);
      Shameem_04_10_22_INSTRUCTION(63 downto 32) <= Shameem_04_10_22_IR(7);
      Shameem_04_10_22_INSTRUCTION(31 downto 0) <= Shameem_04_10_22_IR(8);
    when "0011" =>
      Shameem_04_10_22_INSTRUCTION(67 downto 64) <= Shameem_04_10_22_IR(9)(3 downto 0);
      Shameem_04_10_22_INSTRUCTION(63 downto 32) <= Shameem_04_10_22_IR(10);
      Shameem_04_10_22_INSTRUCTION(31 downto 0) <= Shameem_04_10_22_IR(11);
    when "0100" =>
      Shameem_04_10_22_INSTRUCTION(67 downto 64) <= Shameem_04_10_22_IR(12)(3 downto 0);
      Shameem_04_10_22_INSTRUCTION(63 downto 32) <= Shameem_04_10_22_IR(13);
      Shameem_04_10_22_INSTRUCTION(31 downto 0) <= Shameem_04_10_22_IR(14);
    when "0101" =>
      Shameem_04_10_22_INSTRUCTION(67 downto 64) <= Shameem_04_10_22_IR(15)(3 downto 0);
      Shameem_04_10_22_INSTRUCTION(63 downto 32) <= Shameem_04_10_22_IR(16);
      Shameem_04_10_22_INSTRUCTION(31 downto 0) <= Shameem_04_10_22_IR(17);
    when "0110" =>
      Shameem_04_10_22_INSTRUCTION(67 downto 64) <= Shameem_04_10_22_IR(18)(3 downto 0);
      Shameem_04_10_22_INSTRUCTION(63 downto 32) <= Shameem_04_10_22_IR(19);
      Shameem_04_10_22_INSTRUCTION(31 downto 0) <= Shameem_04_10_22_IR(20);

```

Messages

- Command: quartus_edt --read_settings_files=off --write_settings_files=off Shameem_04_10_22_ALU -c Shameem_04_10_22_ALU
- 18236 Number of processors has not been specified which may cause overloading on shared machines. Set the global assignment NUM_PARALLEL_PROCESSORS in your QSF to an appropriate value for best performance.
- 204019 Generated file Shameem_04_10_22_ALU.vo in folder "/home/azwad/Desktop/Shameem_04_10_22_ALU/simulation/modelsim/" for EDA simulation tool
- Quartus Prime EDA Netlist Writer was successful. 0 errors, 1 warning
- 293000 Quartus Prime Full Compilation was successful. 0 errors, 233 warnings

Figure 43: Instruction Register VHDL code continued.

The screenshot shows the Quartus Prime software interface with the following details:

- File Menu:** File, Edit, View, Project, Assignments, Processing, Tools, Window, Help.
- Project Navigator:** Shows files including `Shameem_04_10_22_ALU.vhd`, `Shameem_04_10_22_InstructionRegister.vhd`, and a compilation report for `Shameem_04_10_22_ALU`.
- Tasks:** Task list includes **Compile Design** (highlighted in red), Analysis & Synthesis, Fitter (Place & Route), Assembler (Generate program), Timing Analysis, EDA Netlist Writer, and Edit Settings.
- Code Editor:** Displays VHDL code for the instruction register, specifically handling cases for `"0111"`, `"0110"`, `"0111"`, `"1010"`, `"1011"`, and `"1100"`.
- IP Catalog:** Shows installed IP components like Basic Functions, DSP, Interface Protocols, Memory Interfaces and Controllers, Processors and Peripherals, and University Program.
- Messages:** Log window showing compilation messages, including warnings about the number of processors and success of the compilation.
- Bottom Status:** Shows the line number (Ln 88), column number (Col 27), file type (VHDL File), and timestamp (100% 00:03:33).

Figure 44: Instruction Register VHDL code continued.

The screenshot shows the Quartus Prime software interface with the following details:

- File Menu:** File, Edit, View, Project, Assignments, Processing, Tools, Window, Help.
- Project Navigator:** Shows files including `Shameem_04_10_22_ALU.vhd`, `Shameem_04_10_22_InstructionRegister.vhd`, and a compilation report for `Shameem_04_10_22_ALU`.
- Tasks:** Task list includes **Compile Design** (highlighted in red), Analysis & Synthesis, Fitter (Place & Route), Assembler (Generate program), Timing Analysis, EDA Netlist Writer, and Edit Settings.
- Code Editor:** Displays VHDL code for the instruction register, specifically handling cases for `"0101"`, `"1010"`, `"1011"`, and `"1100"`.
- IP Catalog:** Shows installed IP components like Basic Functions, DSP, Interface Protocols, Memory Interfaces and Controllers, Processors and Peripherals, and University Program.
- Messages:** Log window showing compilation messages, including warnings about the number of processors and success of the compilation.
- Bottom Status:** Shows the line number (Ln 88), column number (Col 27), file type (VHDL File), and timestamp (100% 00:03:33).

Figure 45: Instruction Register VHDL code continued.

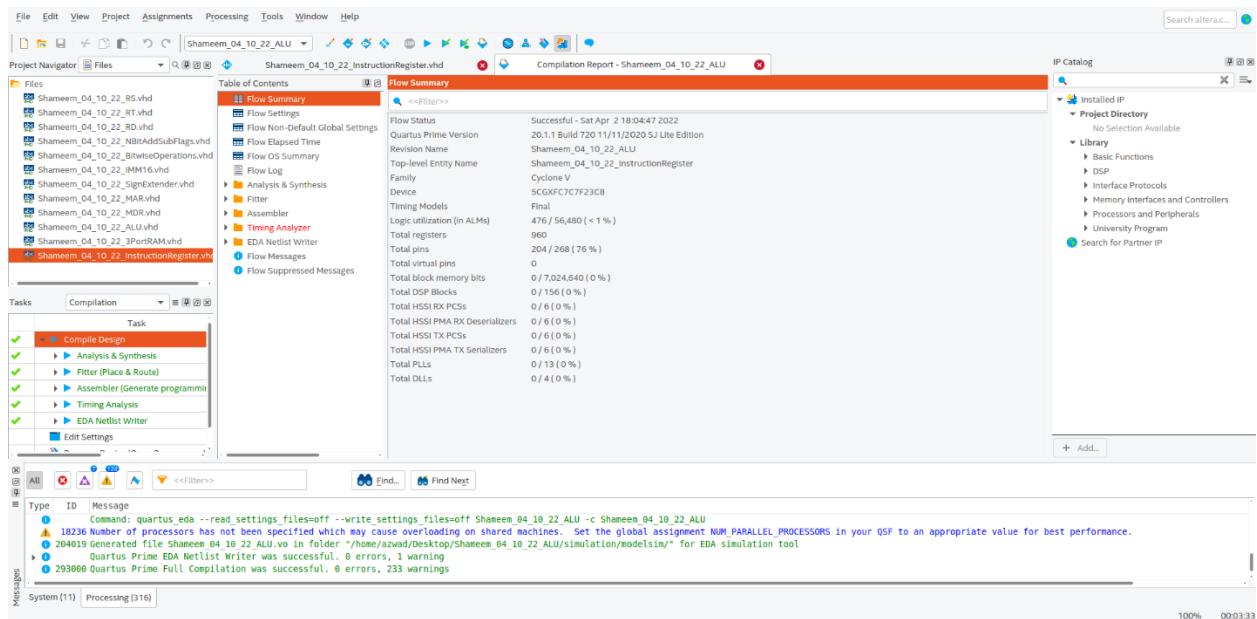


Figure 46: Instruction Register compilation report.

Data Memory

The screenshot shows the Quartus Prime interface with the following details:

- Project Navigator:** Shows files including `Shameem_04_10_22_ALU.vhd`, `Shameem_04_10_22_DataMemory.vhd`, and a compilation report for `Shameem_04_10_22_ALU`.
- Code Editor:** Displays the VHDL code for `Shameem_04_10_22_DataMemory`. The code defines an entity `Shameem_04_10_22_DataMemory` with a port containing three signals: `Shameem_04_10_22_clk`, `Shameem_04_10_22_MEMORY`, and `Shameem_04_10_22_MDR`. It includes an architecture `arch` with a `WriteMemory` process and a `ReadMemory` process.
- Tasks:** A list of build steps: Compile Design, Analysis & Synthesis, Fitter (Place & Route), Assembler (Generate program), Timing Analysis, and EDA Netlist Writer. The 'Compile Design' step is highlighted.
- Messages:** A panel showing compilation logs. It includes messages from Quartus Prime EDA Netlist Writer and Quartus Prime Full Compilation, both indicating success with 0 errors and 43 warnings.
- IP Catalog:** On the right, it shows 'Installed IP' and 'Project Directory' sections.

Figure 47: Data Memory VHDL code compiled successfully in Quartus

This screenshot shows the continuation of the VHDL code for the Data Memory component. The code is identical to Figure 47 but continues further down the page, ending at line 40. The Quartus Prime interface remains the same, with the code editor, tasks, messages, and IP catalog visible.

Figure 48: Data Memory VHDL code continued.

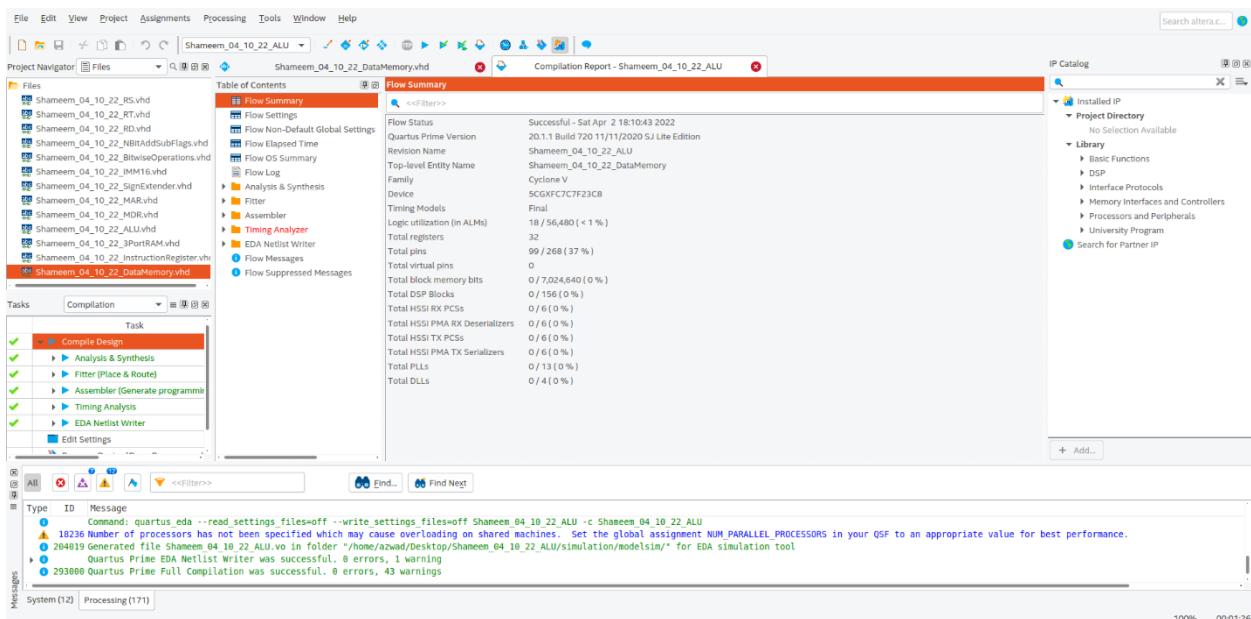


Figure 49: Data Memory compilation report.

Simulation: Part I.A *Add*

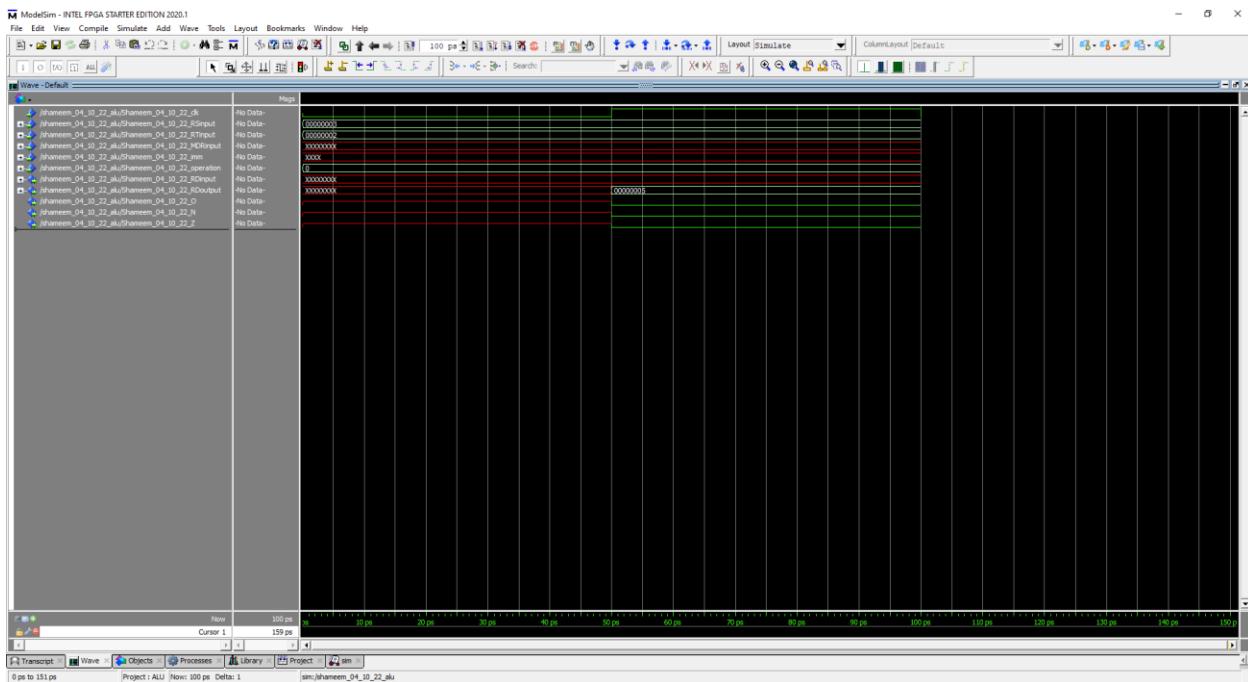


Figure 50: Simulation to represent ADD (radix is hexadecimal for everything)

In this simulation the operation is 0000 which stands for add so it takes RS and RT and adds them together and outputs them in the RD. Furthermore, 0x00000003 plus 0x00000002 equals 0x00000005, which means the flags zero and negative should be 0 as shown in the waveform because the output is clearly not negative or zero. The overflow flag is also 0 because the output does not trigger the rules of overflow since a positive plus a positive can equal a positive without any overflow.

$$R[rd] (0x00000005) = R[rs] (0x00000003) + R[rt] (0x00000002)$$

Shameem_04_10_22_Add.asm

```

1 .data
2 rd: .word 0x00000000
3 rs: .word 0x00000003
4 rt: .word 0x00000002
5
6 .text
7 lw $s0, rd
8 lw $s1, rs
9 lw $s2, rt
10
11 # rd = rs + rt
12 add $s0, $s1, $s2
13 sw $s0 rd
14

```

Figure 51: MIPS code for Add

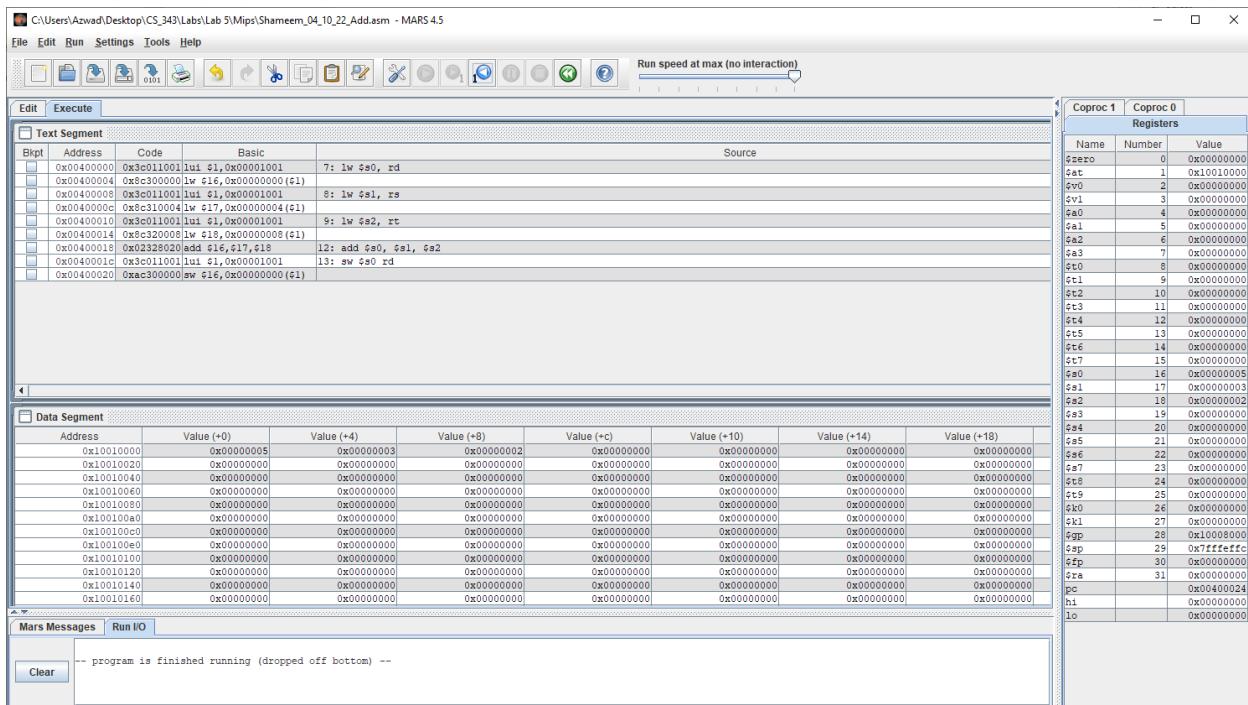


Figure 52: Executed Successfully.

Values RD, RS, RT shown clearly in data segment in the order of 0x00000005, 0x00000003, 0x00000002. The results of the MIPS code on MARS resulted in the same result as on the waveform simulation in figure 50. Therefore, add is the same on MARS and ModelSim waveform, so the VHDL code for add is working correctly.

Addu

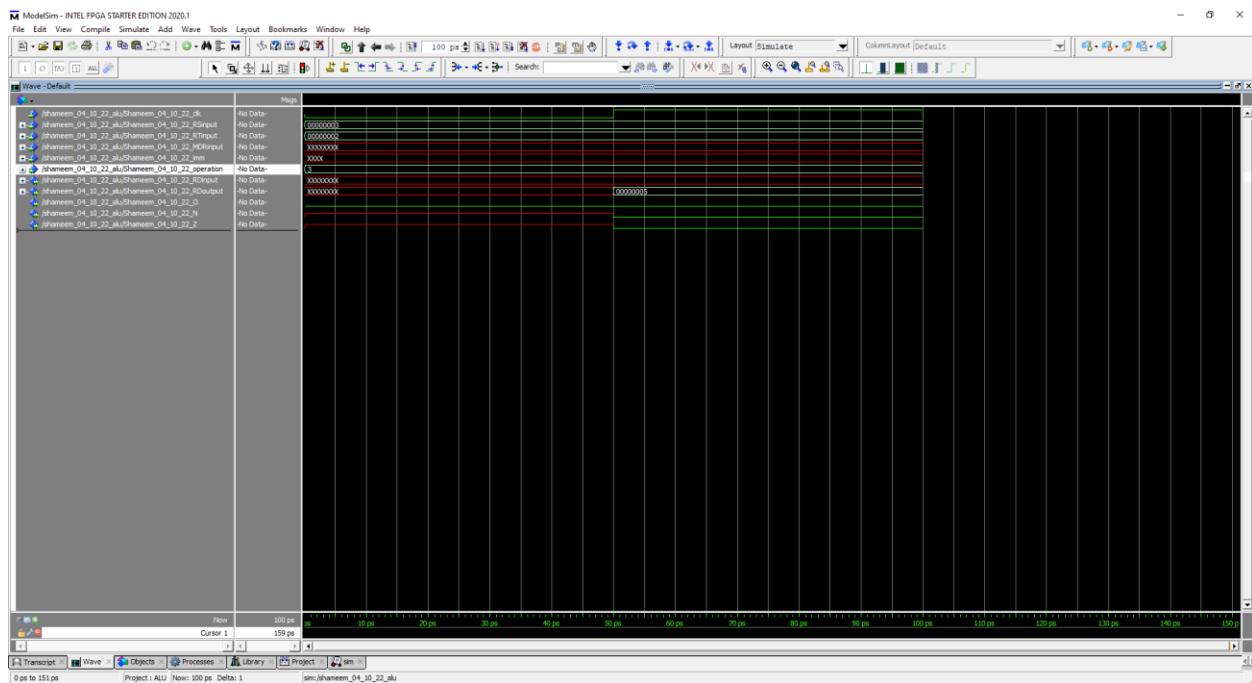


Figure 53: Simulation to represent Addu (radix is hexadecimal for everything)

In this simulation the operation is 0011, which stands for addu, so it takes RS and RT and adds them together and outputs them in the RD. Furthermore, since the operation is doing addu the overflow flag is set to '0' and will not be triggered.

```

Edit Execute
Shameem_04_10_22_Addu.asm

1 .data
2 rd: .word 0x00000000
3 rs: .word 0x00000003
4 rt: .word 0x00000002
5
6 .text
7 lw $s0, rd
8 lw $s1, rs
9 lw $s2, rt
10
11 # rd = rs + rt
12 addu $s0, $s1, $s2
13 sw $s0, rd
14

```

Figure 54: MIPS Addu

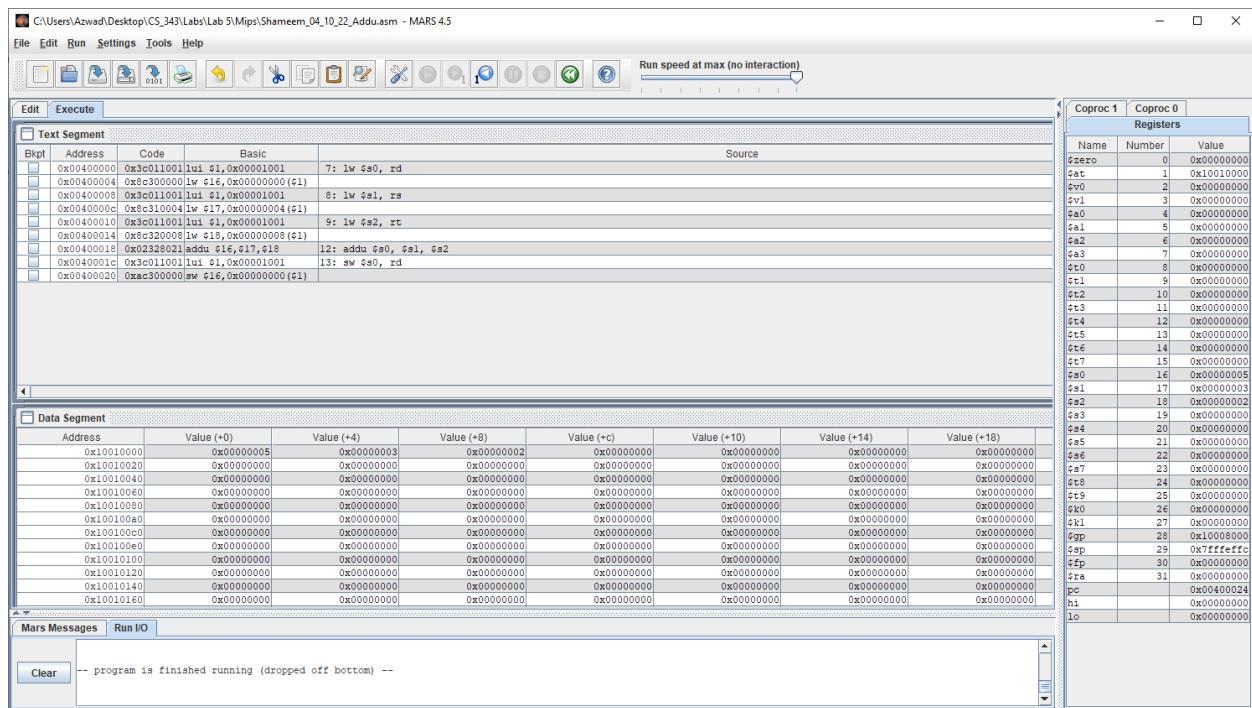


Figure 55: Executed Successfully in MARs.

Values RD, RS, RT from the equation $RD = RS + RT$ shown clearly in data segment in the order of 0x00000005, 0x00000003, 0x00000002. The results of the MIPS code on MARS resulted in the same result as on the waveform simulation in figure 53. Therefore, addu is the same on MARS and ModelSim waveform, so the VHDL code for addu is working correctly.

Sub

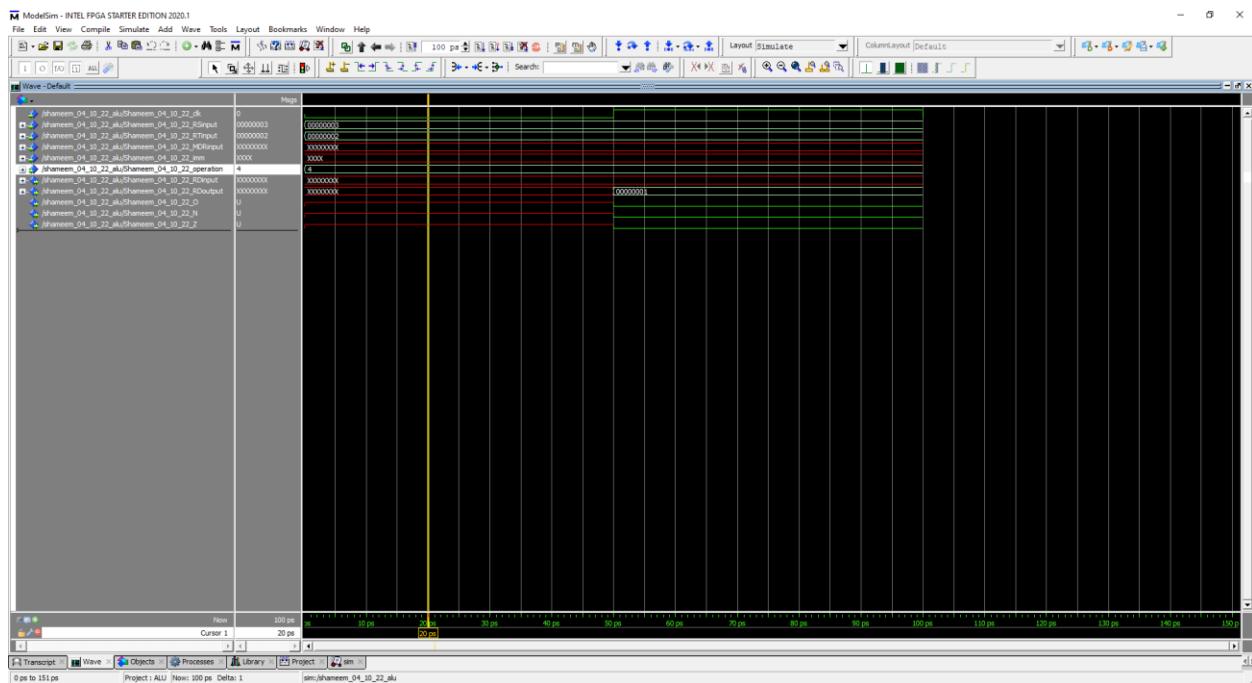


Figure 56: Simulation to represent Sub (radix is hexadecimal for everything)

In this simulation the operation is 0100 which stands for sub so it takes RS and RT and subtracts them and outputs them in the RD. Furthermore, 0x00000003 minus 0x00000002 equals 0x00000001, which means the flags zero and negative should be 0 as shown in the waveform because the output is clearly not negative or zero. The overflow flag is also 0 because the output does not trigger the rules of overflow since a positive minus a positive can equal a positive without any overflow.

$$R[rd] (0x00000001) = R[rs] (0x00000003) - R[rt] (0x00000002)$$

```

Shameem_04_10_22_Sub.asm

1 .data
2 rd: .word 0x00000000
3 rs: .word 0x00000003
4 rt: .word 0x00000002
5
6 .text
7 lw $s0, rd
8 lw $s1, rs
9 lw $s2, rt
10
11 # rd = rs - rt
12 sub $s0, $s1, $s2
13 sw $s0, rd
14

```

Figure 57: MIPS sub

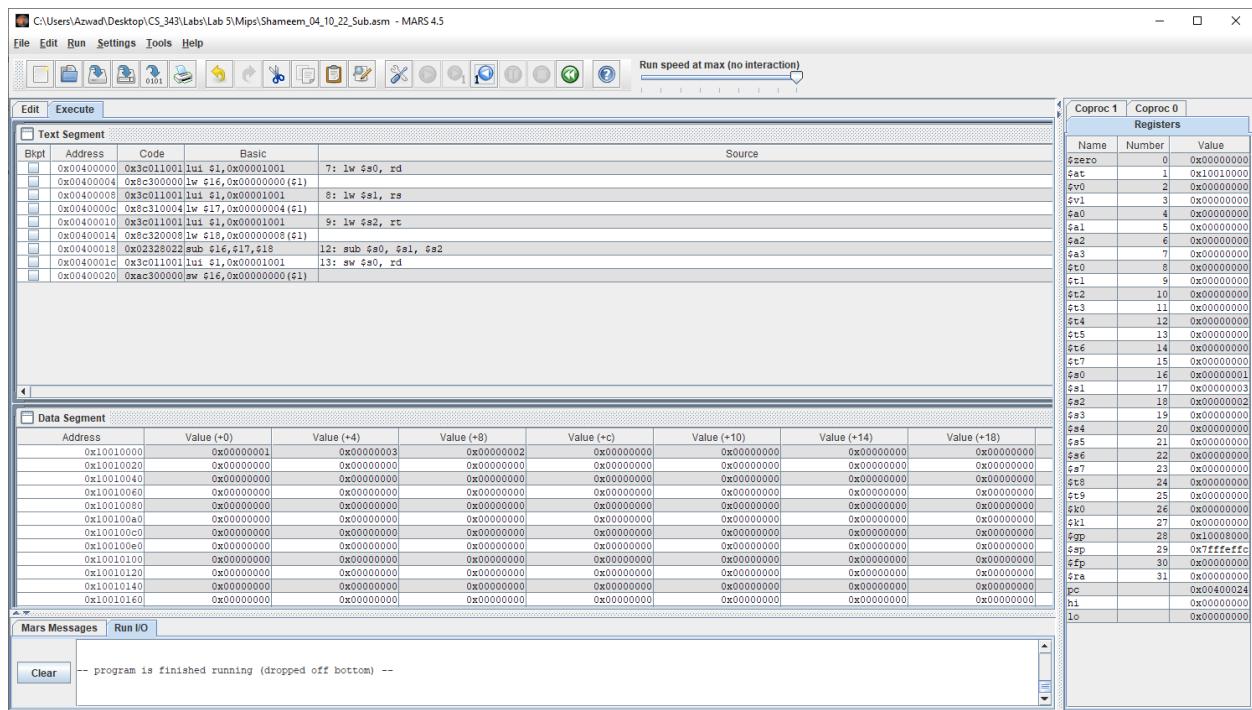


Figure 58: Executed Successfully on Mars.

Values RD, RS, RT shown clearly in data segment in the order of 0x00000001, 0x00000003, 0x00000002. The results of the MIPS code on MARS resulted in the same result as on the waveform simulation in figure 50. Therefore, sub is the same on MARS and ModelSim waveform, so the VHDL code for sub is working correctly.

Subu

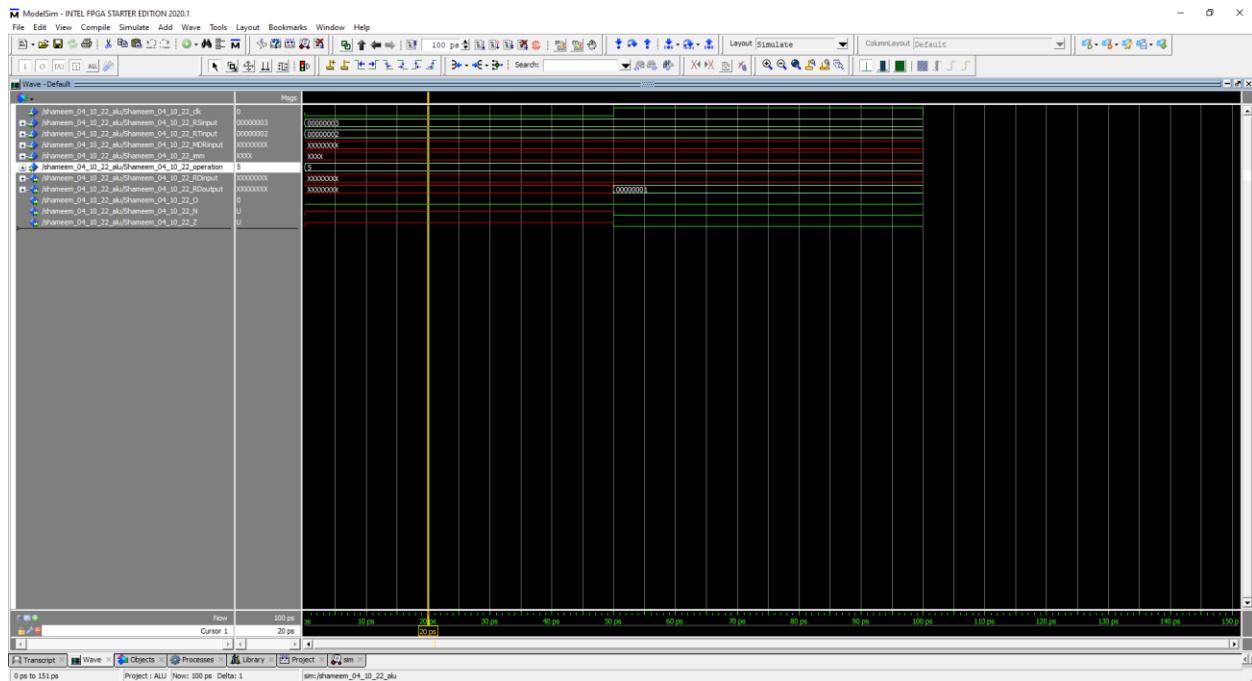


Figure 59: Simulation to represent Sub (radix is hexadecimal for everything)

In this simulation the operation is 0101 which stands for subu, so it takes RS and RT and subtracts them and outputs them in the RD. Furthermore, 0x00000003 minus 0x00000002 equals 0x00000001, which means the flags zero and negative should be 0 as shown in the waveform because the output is clearly not negative or zero. The overflow flag is also 0 because the output does not trigger the rules of overflow since a positive minus a positive can equal a positive without any overflow.

$$R[rd] (0x00000001) = R[rs] (0x00000003) - R[rt] (0x00000002)$$

```

Shameem_04_10_22_Shubu.asm

1 .data
2 rd: .word 0x00000000
3 rs: .word 0x00000003
4 rt: .word 0x00000002
5
6 .text
7 lw $s0, rd
8 lw $s1, rs
9 lw $s2, rt
10
11 # rd = rs - rt
12 subu $s0, $s1, $s2
13 sw $s0, rd
14

```

Figure 60: MIPS subu

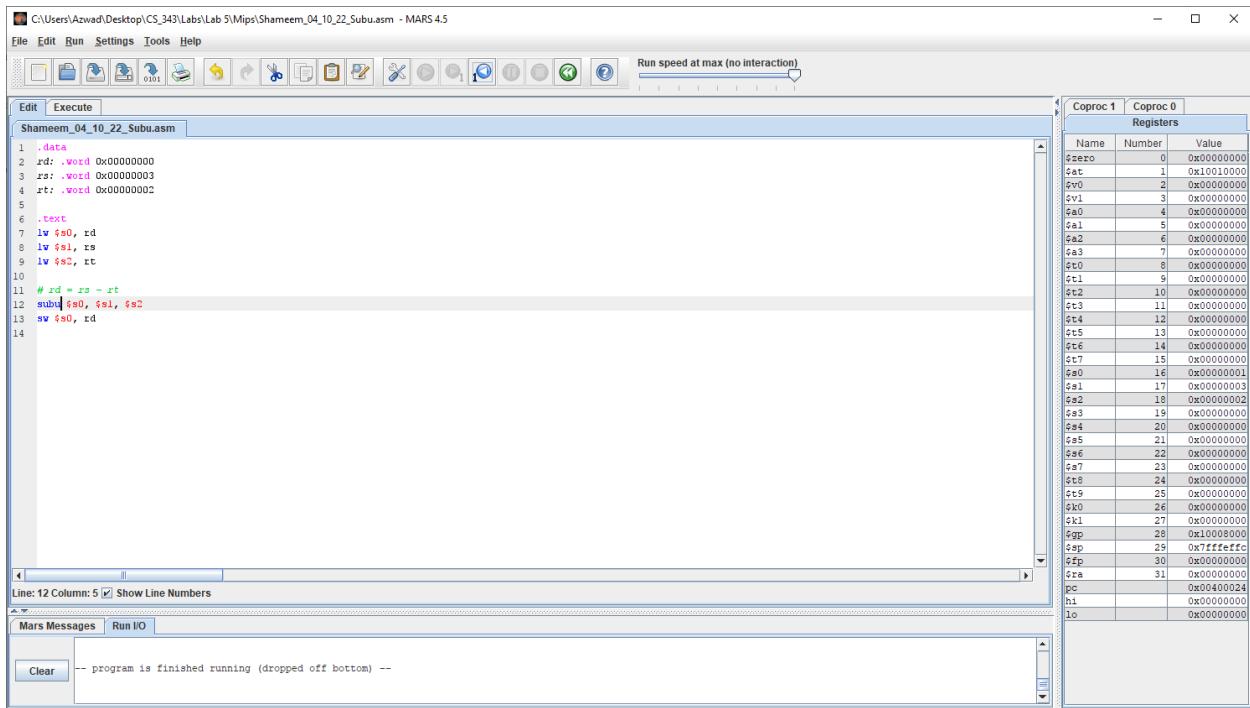


Figure 61: Executed Successfully on Mars.

Values RD, RS, RT shown clearly in data segment in the order of 0x00000001, 0x00000003, 0x00000002. The results of the MIPS code on MARS resulted in the same result as on the waveform simulation in figure 59. Therefore, subu is the same on MARS and ModelSim waveform, so the VHDL code for subu is working correctly.

And

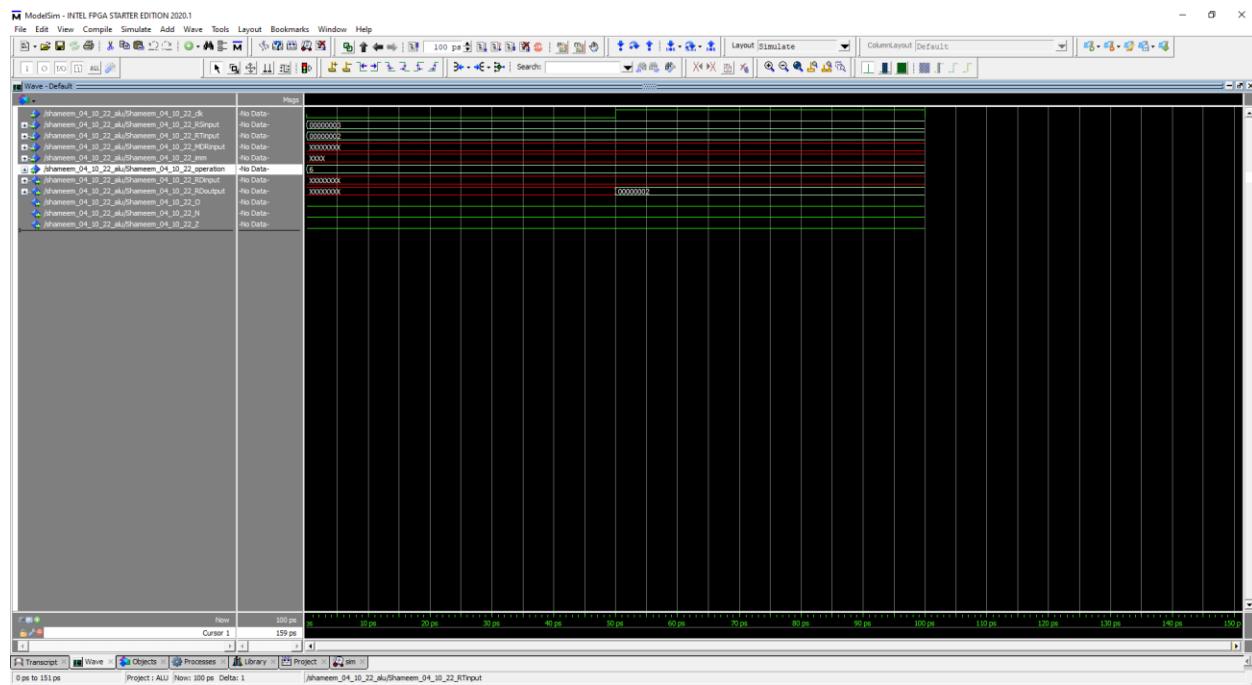


Figure 62: Simulation to represent AND (radix is hexadecimal for everything)

In this simulation the operation is 0110 which stands for add so it takes RS and RT and ands them, which outputs them in the RD. Furthermore, 0x00000003 and 0x00000002 equals 0x00000002, which means the flags zero and negative should be 0 as shown in the waveform because the output is clearly not negative or zero. The overflow flag is also 0 because and does not trigger an overflow.

$R[rd] (0x00000002) = R[rs] (0x00000003) \text{ and } R[rt] (0x00000002)$

```
Shameem_04_10_22_And.asm
1 .data
2 rd: .word 0x00000000
3 rs: .word 0x00000003
4 rt: .word 0x00000002
5
6 .text
7 lw $s0, rd
8 lw $s1, rs
9 lw $s2, rt
10
11 # rd = rs & rt
12 and $s0, $s1, $s2
13 sw $s0, rd
14
```

Figure 63: MIPS and

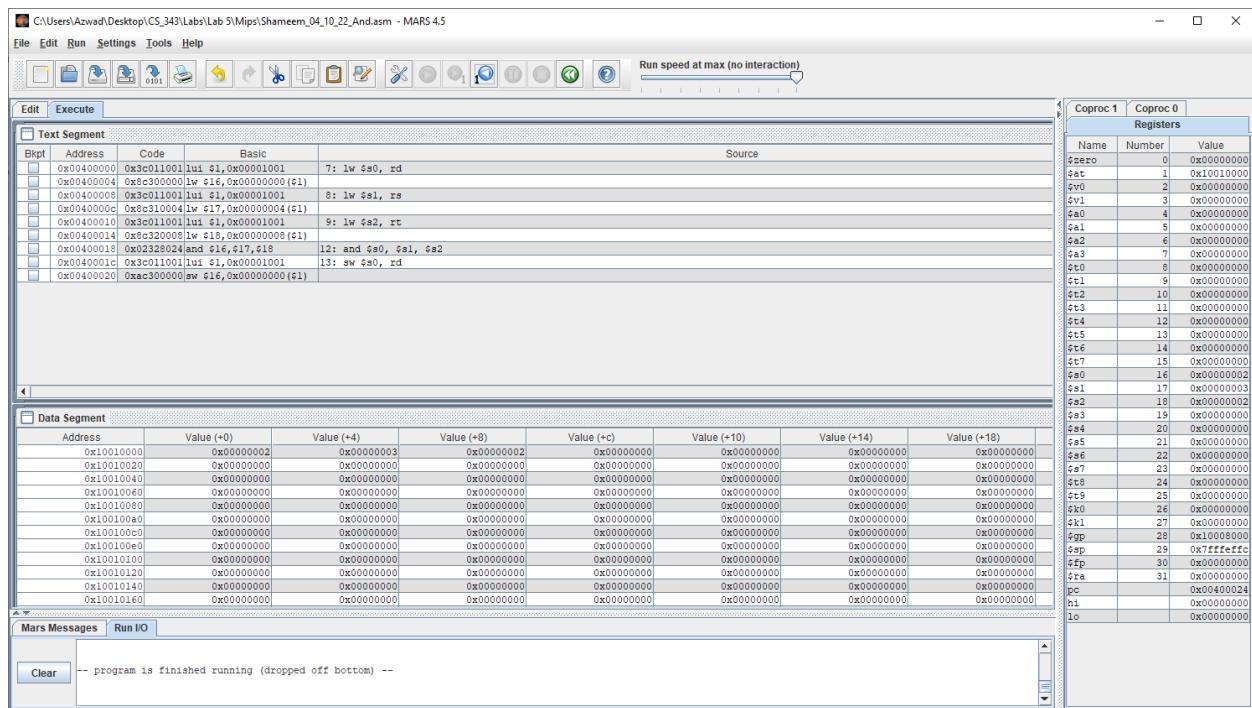


Figure 64: Successfully executed in MARs.

Values RD, RS, RT shown clearly in data segment in the order of 0x00000001, 0x00000003, 0x00000002. The results of the MIPS code on MARS resulted in the same result as on the waveform simulation in figure 62. Therefore, and is the same on MARS and ModelSim waveform, so the VHDL code for and is working correctly.

Nor

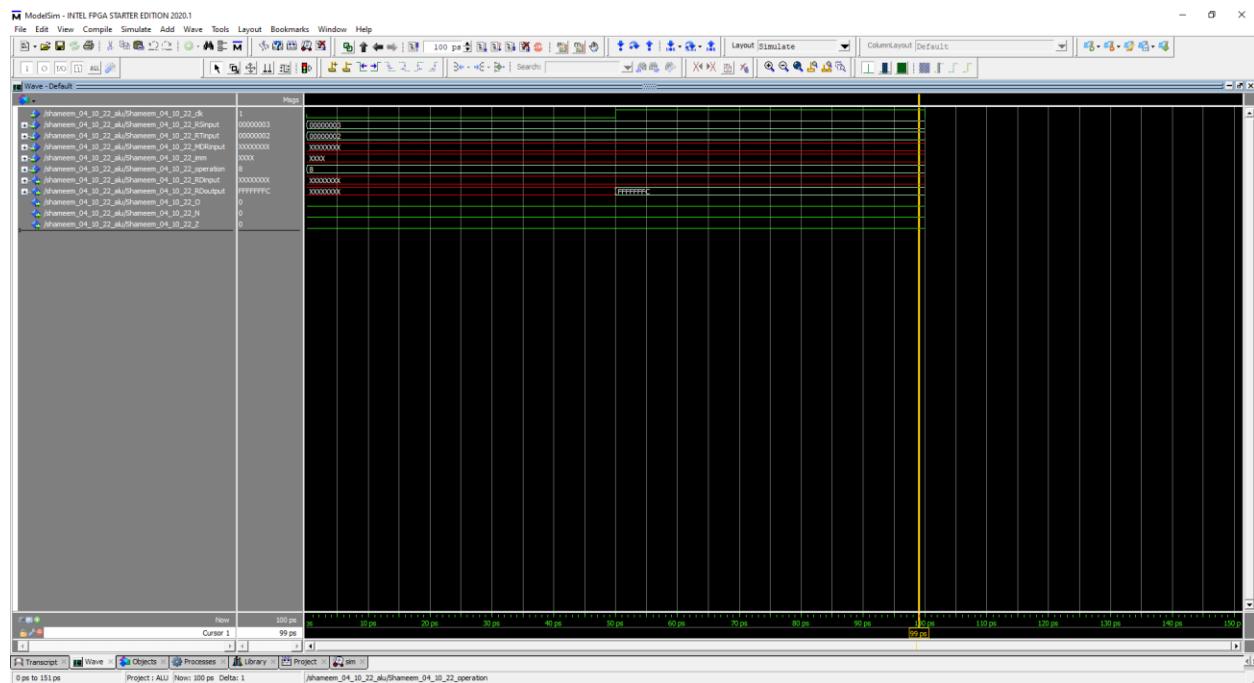


Figure 65: Simulation to represent AND (radix is hexadecimal for everything)

In this simulation the operation is 1000 which stands for add so it takes RS and RT and ands them, which outputs them in the RD. Furthermore, 0x00000003 and 0x00000002 equals 0xFFFFFFFFC, which means the flags zero and negative should be 0 as shown in the waveform because the output is clearly not negative or zero. The overflow flag is also 0 because nor does not trigger an overflow.

$$R[rd] (0xFFFFFFFFC) = R[rs] (0x00000003) \text{ nor } R[rt] (0x00000002)$$

```

Edit Execute
Shameem_04_10_22_Nor.asm
1 .data
2 rd: .word 0x00000000
3 rs: .word 0x00000003
4 rt: .word 0x00000002
5
6
7 .text
8 lw $s0, rd
9 lw $s1, rs
10 lw $s2, rt
11
12 # rd = ~ (rs | rt)
13 nor $s0, $s1, $s2
14 sw $s0, rd
15

```

Figure 66: MIPS nor

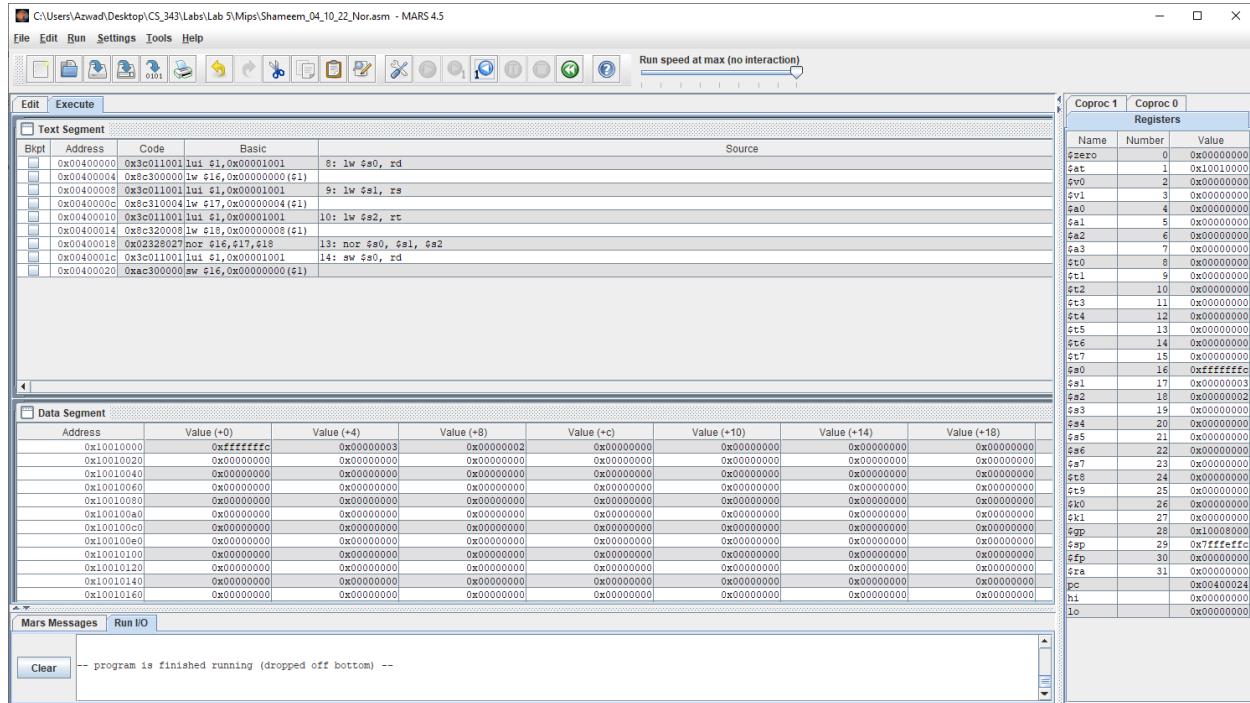


Figure 67: Successfully executed in MARs.

Values RD, RS, RT shown clearly in data segment in the order of 0xFFFFFFF, 0x00000003, 0x00000002. The results of the MIPS code on MARS resulted in the same result as on the waveform simulation in figure 65. Therefore, nor is the same on MARS and ModelSim waveform, so the VHDL code for nor is working correctly.

OR

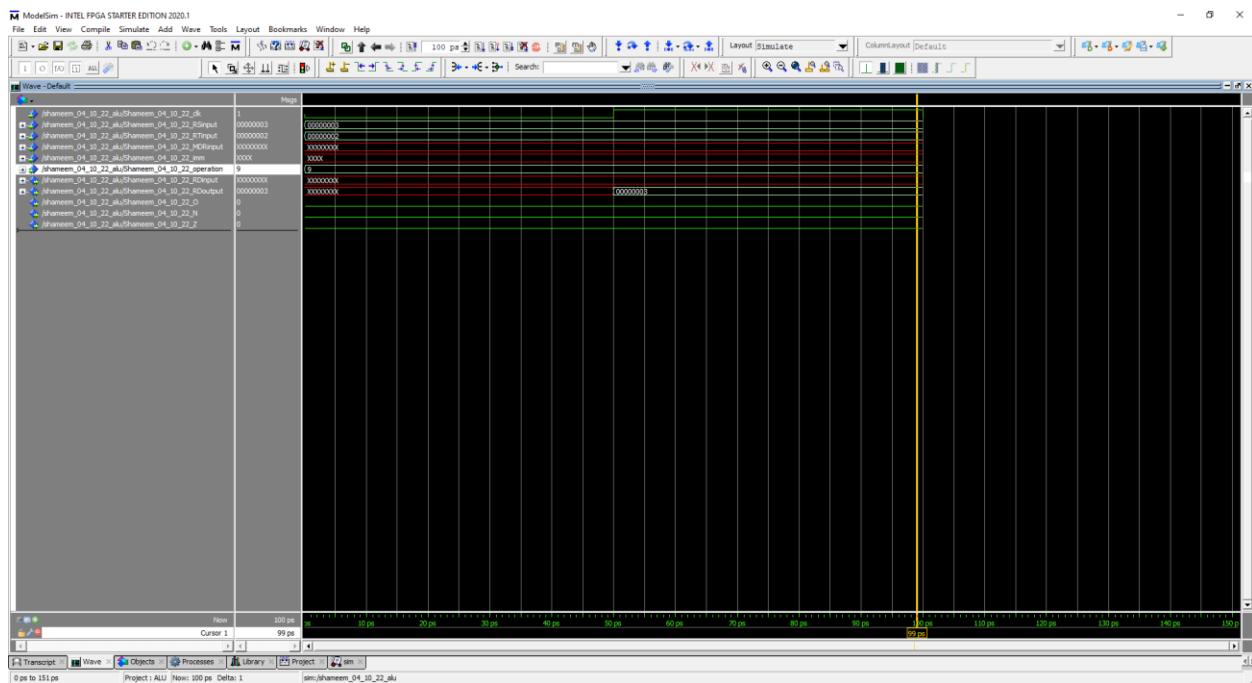


Figure 68: Simulation to represent AND (radix is hexadecimal for everything)

In this simulation the operation is 1001 which stands for add so it takes RS and RT and ands them, which outputs them in the RD. Furthermore, 0x00000003 and 0x00000002 equals 0x00000003, which means the flags zero and negative should be 0 as shown in the waveform because the output is clearly not negative or zero. The overflow flag is also 0 because or does not trigger an overflow.

$R[rd] (0x00000003) = R[rs] (0x00000003) \text{ nor } R[rt] (0x00000002)$

```
Shameem_04_10_22_Or.asm
1 .data
2 rd: .word 0x00000000
3 rs: .word 0x00000003
4 rt: .word 0x00000002
5
6 .text
7 lw $s0, rd
8 lw $s1, rs
9 lw $s2, rt
10
11 # rd = (rs | rt)
12 or $s0, $s1, $s2
13 sw $s0, rd
14
```

Figure 69: MIPS or

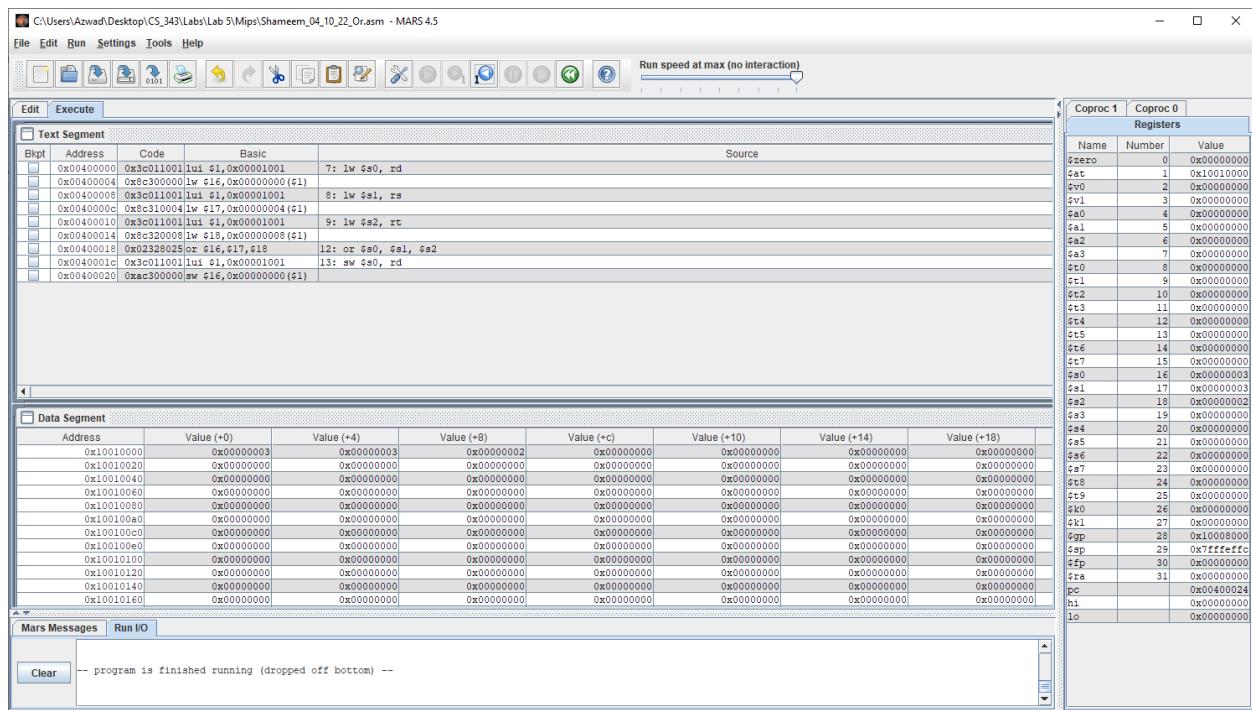


Figure 70: Executed successfully on MARs.

Values RD, RS, RT shown clearly in data segment in the order of 0x00000003, 0x00000002, 0x00000000. The results of the MIPS code on MARS resulted in the same result as on the waveform simulation in figure 62. Therefore, OR is the same on MARS and ModelSim waveform, so the VHDL code for OR is working correctly.

Part I.B

Shift Left Logical – SLL

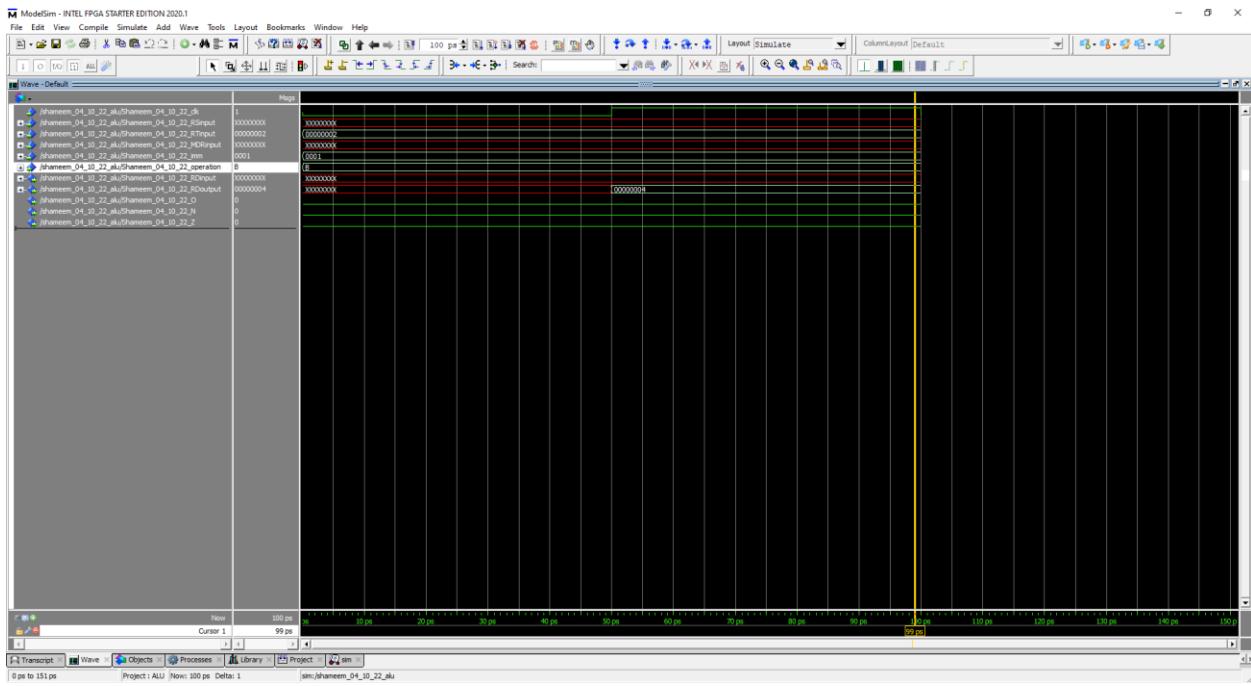


Figure 71: Simulation to represent SLL (radix is hexadecimal for everything)

$$R[rd] (0x00000004) = R[rt] (0x00000002) \ll shamt$$

In this simulation the operation is 1011 which stands for add so it takes RT and shifts it left by shamt, which then outputs in the RD. Furthermore, 0x00000002 shifted by shamt equals 0x00000004, which means the flags zero and negative should be 0 as shown in the waveform because the output is clearly not negative or zero. The overflow flag is also 0 because shifting does not trigger an overflow.

```

Edit Execute
Shameem_04_10_22_SLL.asm
1 .data
2 rd: .word 0x00000002
3 rt: .word 0x00000002
4
5 .text
6 lw $s0, rd
7 lw $s1, rt
8
9 # rd = rt << shamt
10 sll $s0, $s1, 1
11 sw $s0, rd
12

```

Figure 72: MIPS sll

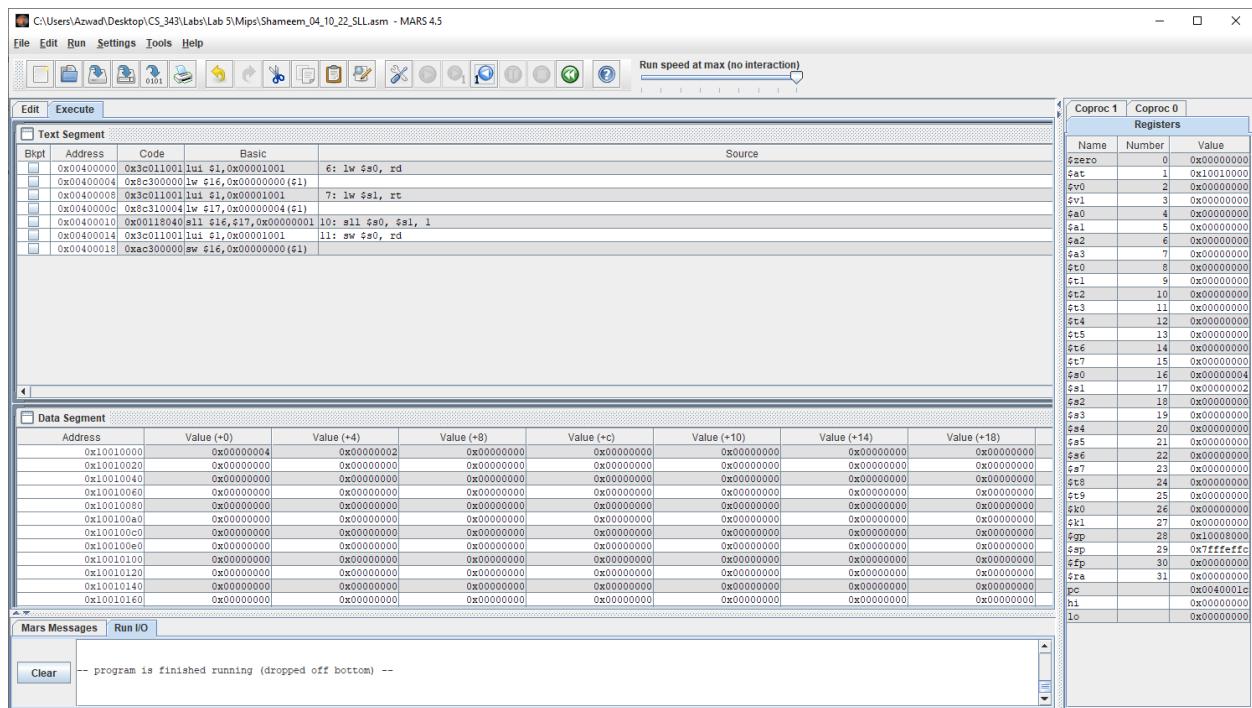


Figure 73: Executed successfully in MARs.

$$R[rd] (0x00000004) = R[rt] (0x00000002) \ll \text{shamt}$$

Values RD, RS shown clearly in data segment in the order of 0x00000004, 0x00000002. The results of the MIPS code on MARS resulted in the same result as on the waveform simulation in figure 71. Therefore, SLL is the same on MARS and ModelSim waveform, so the VHDL code for SLL is working correctly.

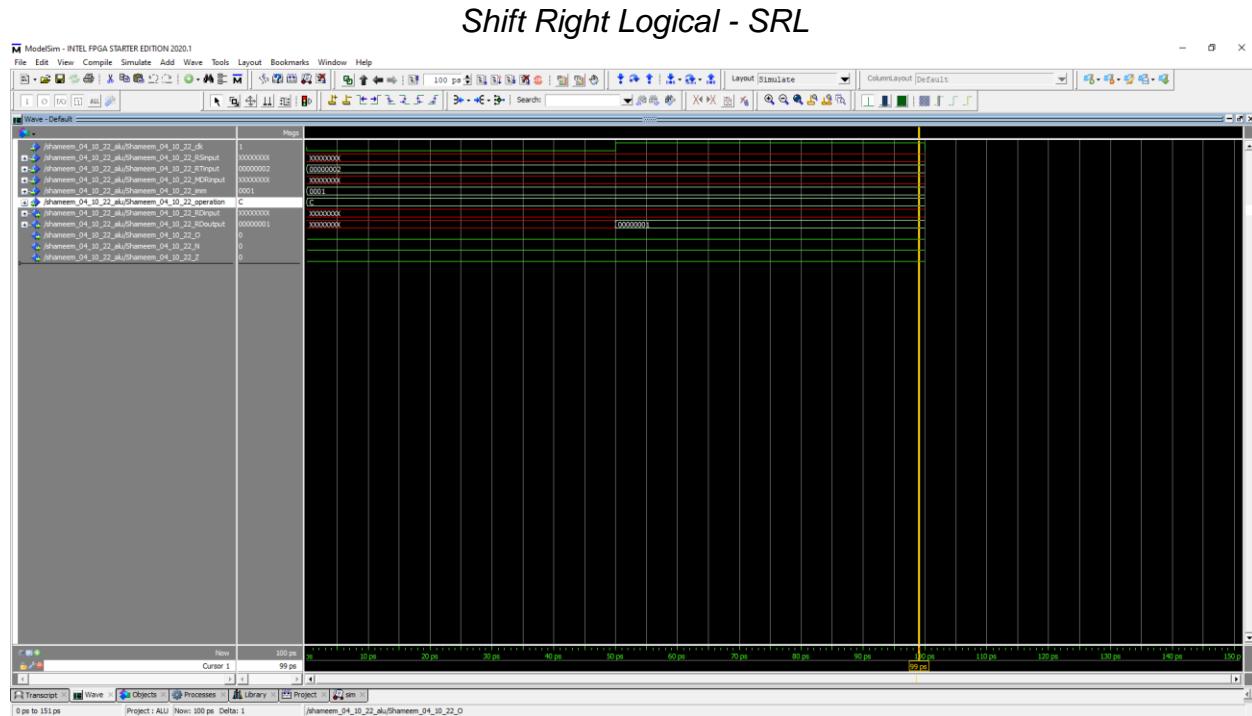


Figure 74: Simulation to represent SRL (radix is hexadecimal for everything)

$R [rd] (0x00000001) = R[rt] (0x00000002) \gg shamt$

In this simulation the operation is 1100 which stands for add so it takes RT and shifts it left by shamt, which then outputs in the RD. Furthermore, 0x00000002 shifted by shamt equals 0x00000001, which means the flags zero and negative should be 0 as shown in the waveform because the output is clearly not negative or zero. The overflow flag is also 0 because shifting does not trigger an overflow.

```
Shameem_04_10_22_SRL.asm
1 .data
2 rd: .word 0x00000000
3 rt: .word 0x00000002
4
5 .text
6 lw $s0, rd
7 lw $s1, rt
8
9 # rd = rt << shamt
10 srl $s0, $s1, 1
11 sw $s0, rd
12
```

Figure 75: MIPS srl

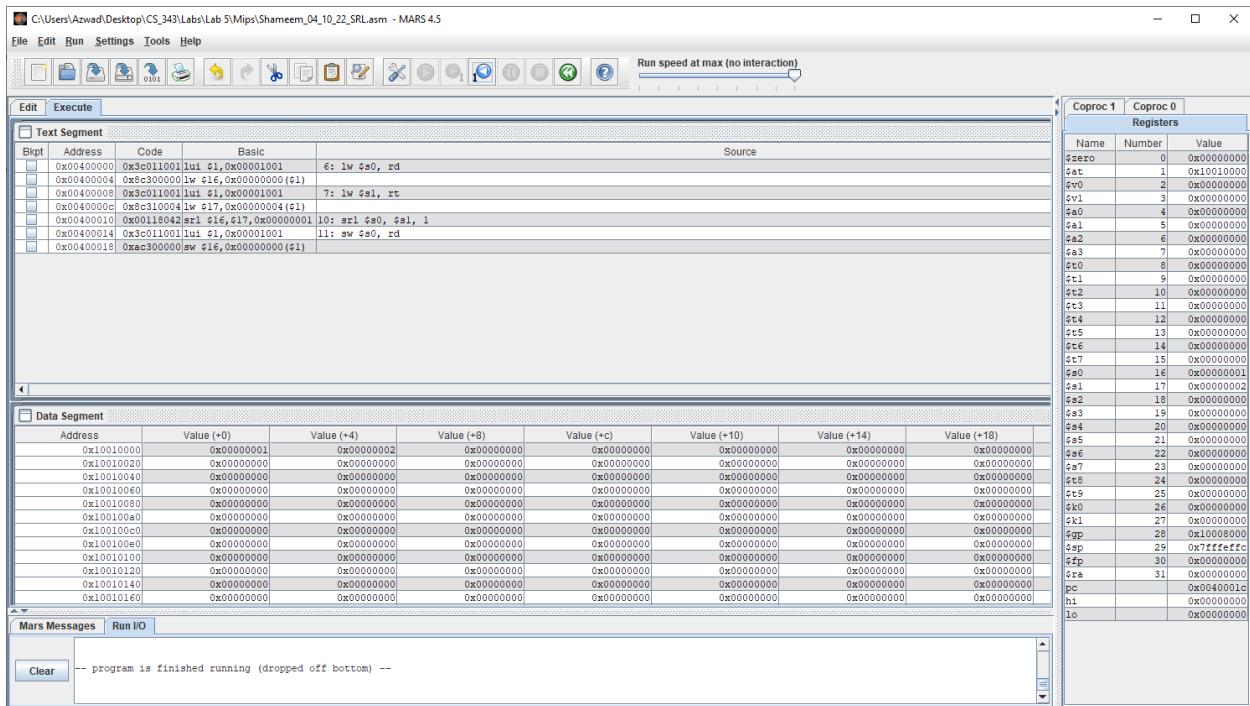


Figure 76: Executed successfully on MARs.

R [rd] (0x00000001) = R[rt] (0x00000002) >> shamt

Values RD, RS shown clearly in data segment in the order of 0x00000001, 0x00000002. The results of the MIPS code on MARs resulted in the same result as on the waveform simulation in figure 74. Therefore, SRL is the same on MARs and ModelSim waveform, so the VHDL code for SRL is working correctly.

Shift Left Arithmetic - SRA

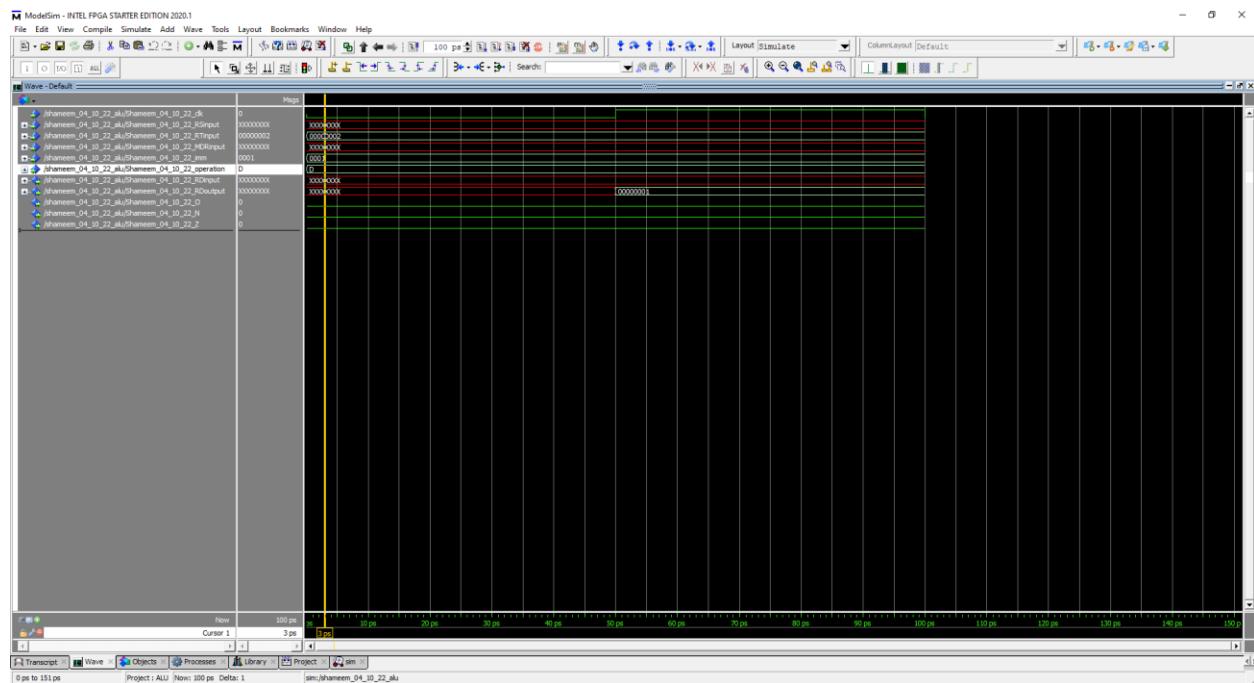


Figure 77: Simulation to represent SLL (radix is hexadecimal for everything)

R [rd] (0x00000001) = R[rt] (0x00000002) >>> shamt

In this simulation the operation is 1101 which stands for add so it takes RT and shifts it left by shamt, which then outputs in the RD. Furthermore, 0x00000002 shifted by shamt equals 0x00000001, which means the flags zero and negative should be 0 as shown in the waveform because the output is clearly not negative or zero. The overflow flag is also 0 because shifting does not trigger an overflow.

```
Shameem_04_10_22_SRA.asm
1 .data
2 rt: .word 0x00000002
3 rd: .word 0x00000000
4
5 .text
6 lw $s1, rt
7 lw $s2, rd
8
9 # rd = rt << shamt
10 sta $s2, $s1, 1
11 sw $s2, rd
12
```

Figure 78: MIPS sra

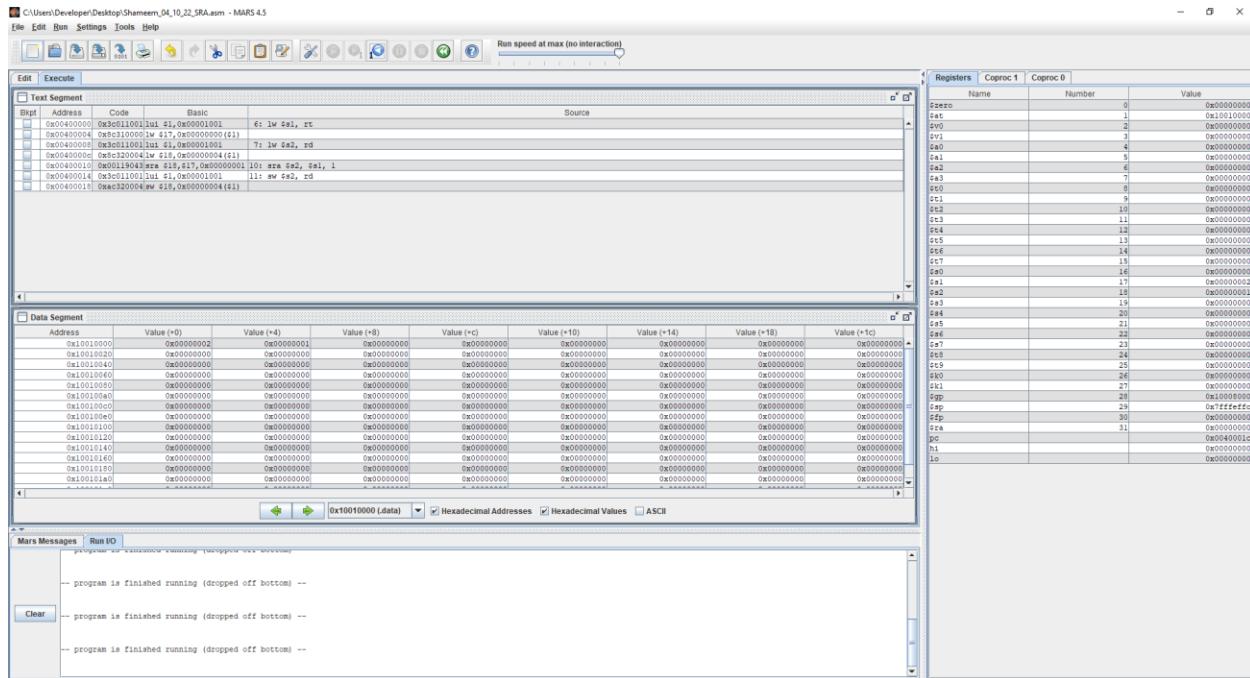


Figure 78: Executed successfully on MARs.

$R[rd] (0x00000001) = R[rt] (0x00000002) \ggg shamt$

Values RS, RD shown clearly in data segment in the order of `0x00000002, 0x00000001`. The results of the MIPS code on MARs resulted in the same result as on the waveform simulation in figure 76. Therefore, SRA is the same on MARs and ModelSim waveform, so the VHDL code for SRA is working correctly.

Part II Addi

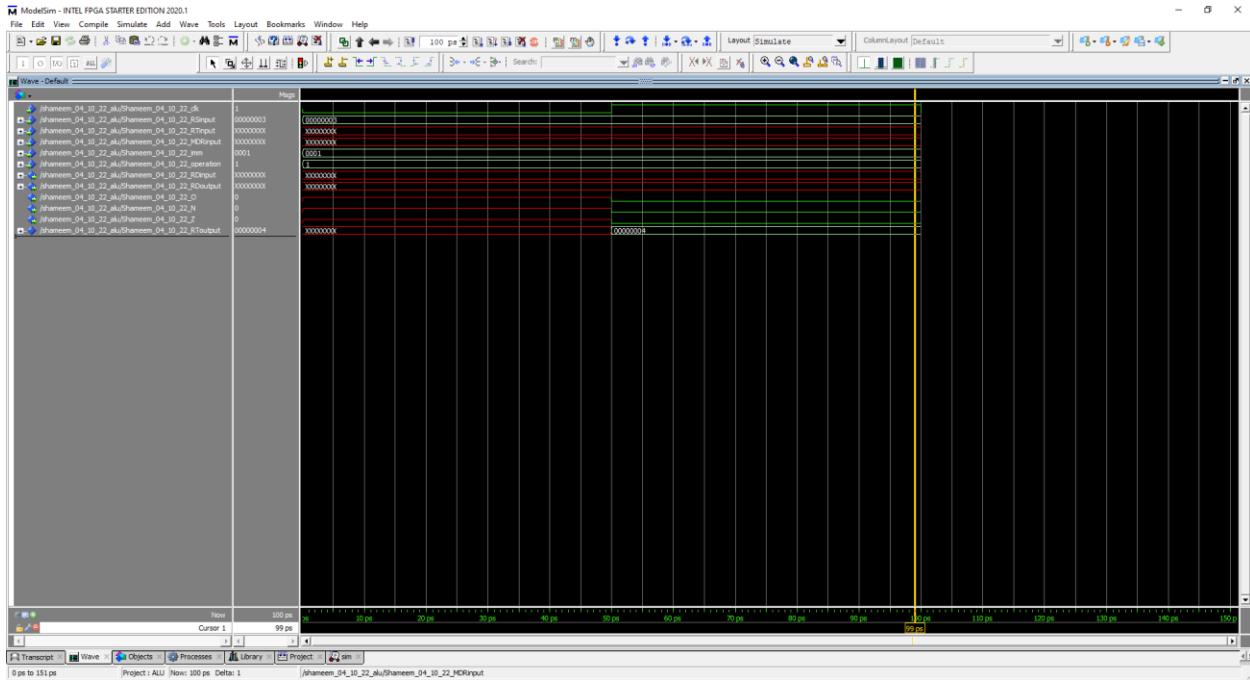


Figure 79: Simulation to represent Addi (radix is hexadecimal for everything)

$$R[rt] (0x00000004) = R[rs] (0x00000003) + \text{SignExtImm}$$

In this simulation the operation is 0001 which stands for addi so it takes RS and ZeroExtImm and adds them together and outputs them in the RD. Furthermore, 0x00000003 plus 0x00000001 equals 0x00000004, which means the flags zero and negative should be 0 as shown in the waveform because the output is clearly not negative or zero. The overflow flag is also 0 because positive plus positive equals a positive does not trigger the overflow.

```

Edit Execute
Shameem_04_10_22_Addi.asm
1 .data
2 rt: .word 0x00000000
3 rs: .word 0x00000003
4
5 .text
6 lw $s0, rt
7 lw $s1, rs
8
9 # rd = rt + signExtImm
10 addi $s0, $s1, 1
11 sw $s0, rt
12

```

Figure 80: MIPS addi

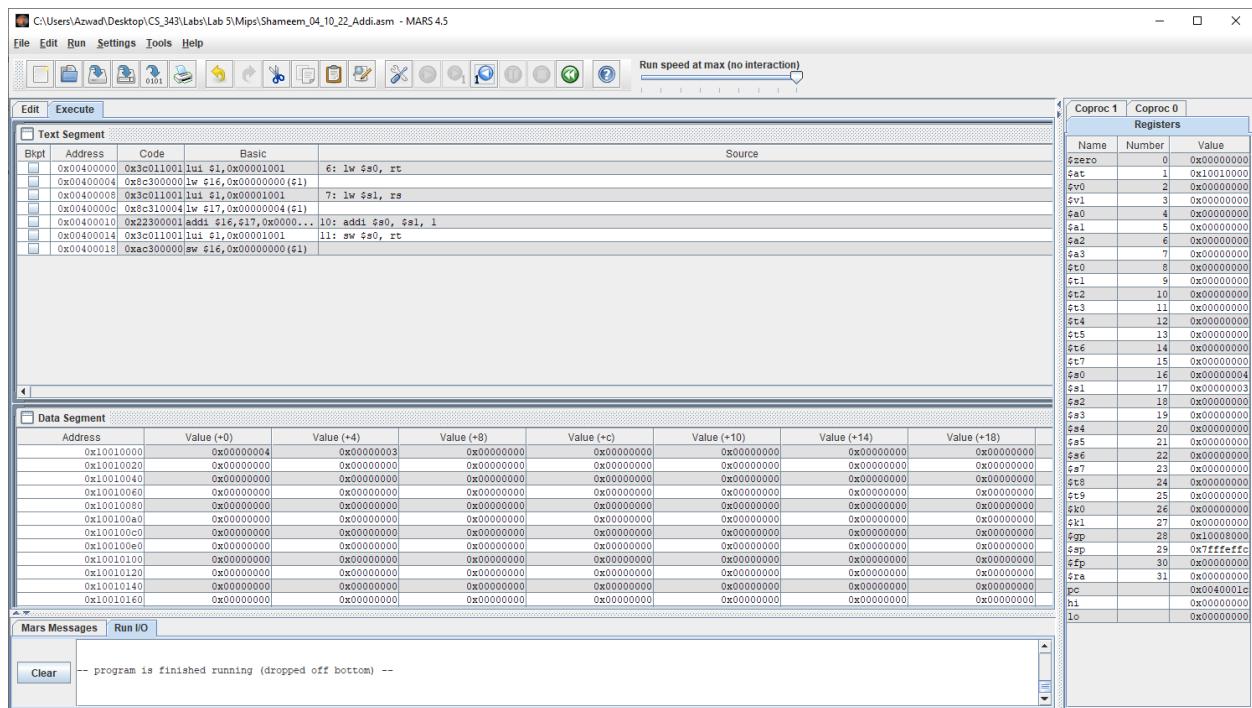


Figure 81: Executed successfully on MARs.

$$R[rt] (0x00000004) = R[rs] (0x00000003) + \text{SignExtImm}$$

Values RT, Rs shown clearly in data segment in the order of 0x00000004, 0x00000003. The results of the MIPS code on MARs resulted in the same result as on the waveform simulation in figure 79. Therefore, Addi is the same on MARS and ModelSim waveform, so the VHDL code for Addi is working correctly.

Addiu

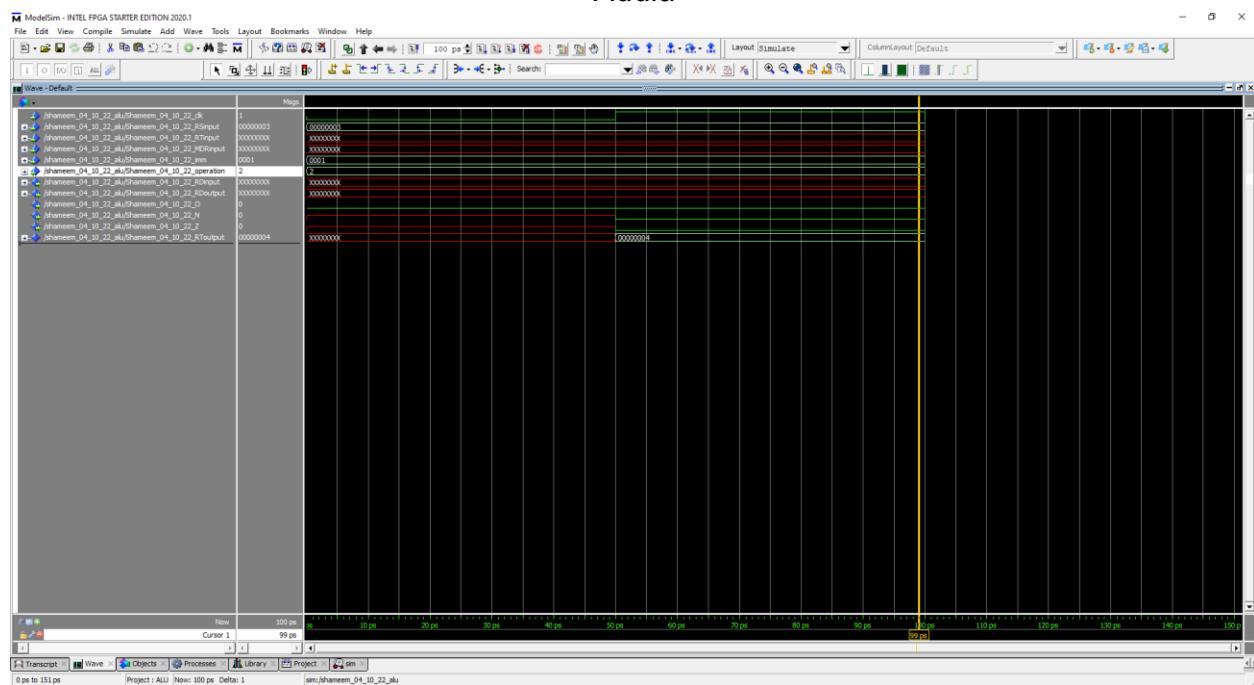


Figure 83: Simulation to represent Addiu (radix is hexadecimal for everything)

$$R[rt] (0x00000004) = R[rs] (0x00000003) + \text{SignExtImm}$$

In this simulation the operation is 0010 which stands for addiu, so it takes RS and ZeroExtImm and adds them together and outputs them in the RD. Furthermore, 0x00000003 plus 0x00000001 equals 0x00000004, which means the flags zero and negative should be 0 as shown in the waveform because the output is clearly not negative or zero. The overflow flag is also 0 because addiu does not detect overflows.

```

Edit Execute
Shameem_04_10_22_Addiu.asm
1 .data
2 rt: .word 0x00000000
3 rs: .word 0x00000003
4
5 .text
6 lw $s0, rt
7 lw $s1, rs
8
9 # rd = rt + signExtImm
10 addiu $s0, $s1, 1
11 sw $s0, rt
12

```

Figure 83: MIPS addiu

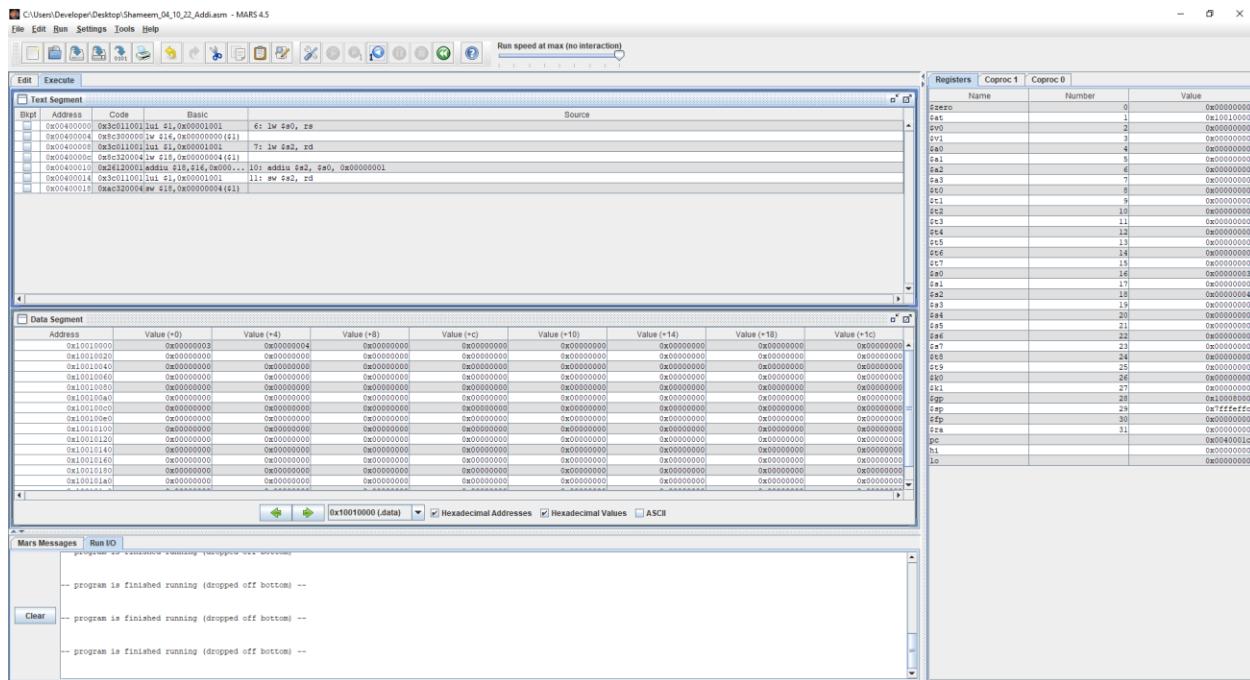


Figure 84: Executed successfully on MARs.

$$R[rt] (0x00000004) = R[rs] (0x00000003) + \text{SignExt}Iimm$$

Values RT, Rs shown clearly in data segment in the order of 0x00000004, 0x00000003. The results of the MIPS code on MARS resulted in the same result as on the waveform simulation in figure 79. Therefore, Addiu is the same on MARS and ModelSim waveform, so the VHDL code for Addiu is working correctly.

Andi

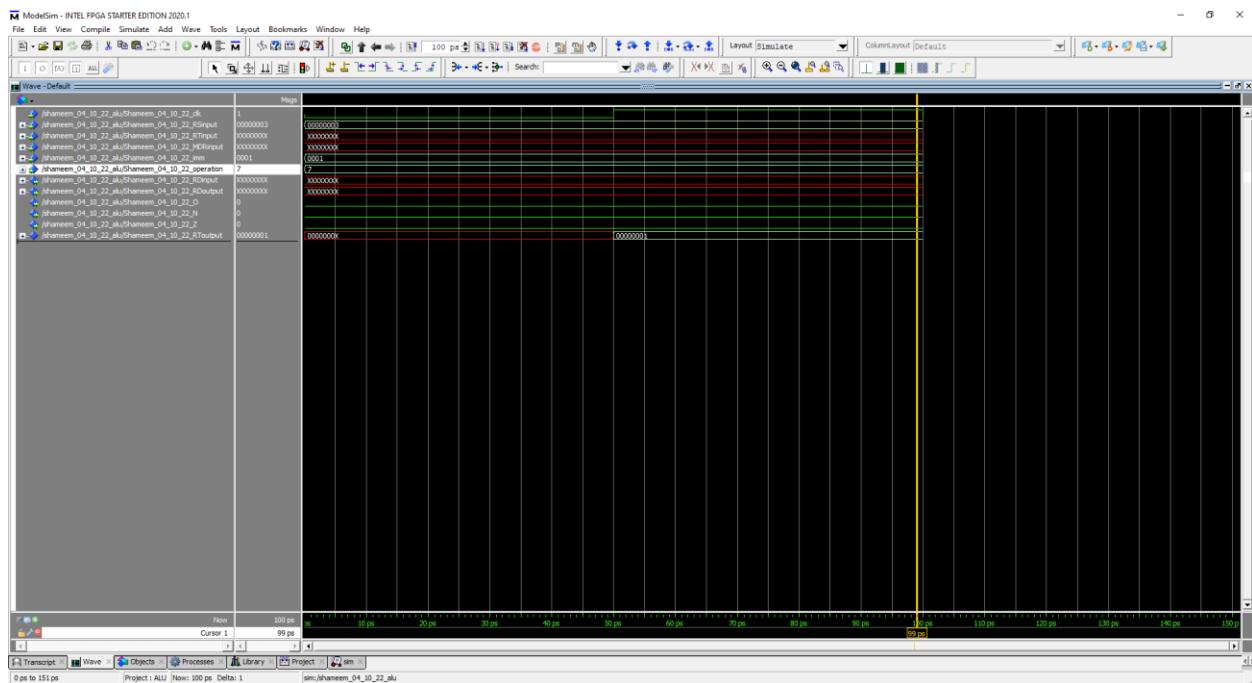


Figure 85: Simulation to represent Andi (radix is hexadecimal for everything)

$$R[rt] (0x00000001) = R[rs] (0x00000003) \& ZeroExtImm$$

In this simulation the operation is 0111 which stands for andi, so it takes RS and RT and adds them together and outputs them in the RD. Furthermore, 0x00000003 & 0x00000001 equals 0x00000001, which means the flags zero and negative should be 0 as shown in the waveform because the output is clearly not negative or zero. The overflow flag is also 0 because andi does not trigger overflows.

```

Edit Execute
Shameem_04_10_22_Aandi.asm

1 .data
2 rt: .word 0x00000000
3 rs: .word 0x00000003
4
5 .text
6 lw $s0, rt
7 lw $s1, rs
8
9 # rd = rt & zeroExtImm
10 andi $s0, $s1, 0x00000001
11 sw $s0, rt
12

```

Figure 86: MIPS andi

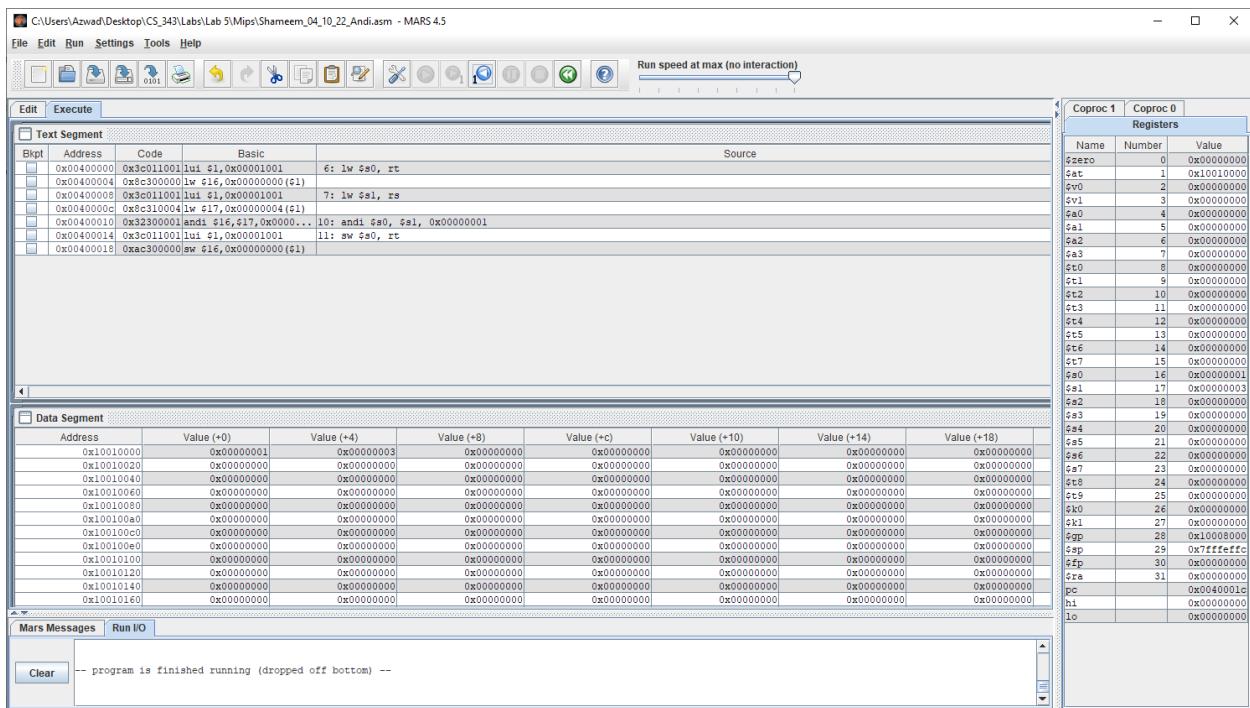


Figure 87: Executed successfully on MARs.

R [rt] (0x00000001) = R[rs] (0x00000003) & ZeroExtImm

Values RT, RS shown clearly in data segment in the order of 0x00000001, 0x00000003. The results of the MIPS code on MARs resulted in the same result as on the waveform simulation in figure 82. Therefore, Andi is the same on MARS and ModelSim waveform, so the VHDL code for Andi is working correctly.

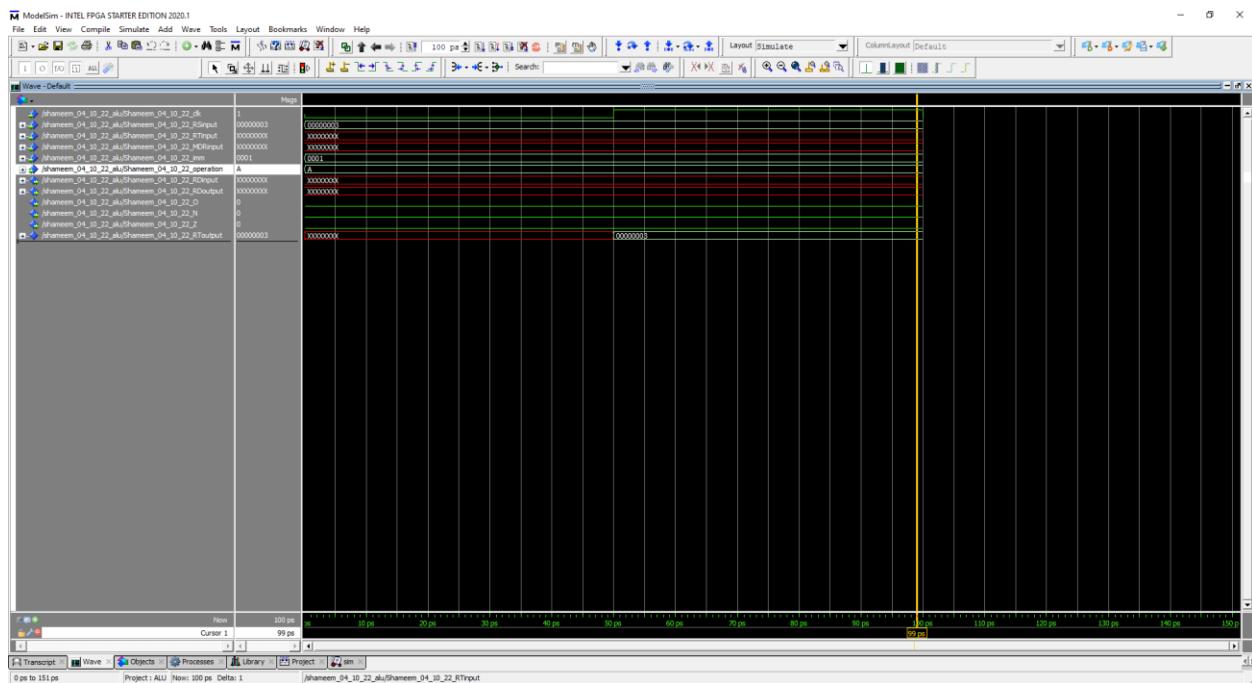
Ori

Figure 88: Simulation to represent Andi (radix is hexadecimal for everything)

$$R[rt] (0x00000003) = R[rs] (0x00000003) \mid \text{ZeroExtImm}$$

In this simulation the operation is 1010 which stands for Ori, so it takes RS and RT and adds them together and outputs them in the RD. Furthermore, 0x00000003 or 0x00000001 equals 0x00000003, which means the flags zero and negative should be 0 as shown in the waveform because the output is clearly not negative or zero. The overflow flag is also 0 because Ori does not trigger overflows.

```

Edit Execute
Shameem_04_10_22_Ori.asm

1 .data
2 rt: .word 0x00000000
3 rs: .word 0x00000003
4
5 .text
6 lw $s0, rt
7 lw $s1, rs
8
9 # rt = rs | ZeroExtImm
10 ori $s0, $s1, 1
11 sw $s0, rt
12
13

```

Figure 89: MIPS ori

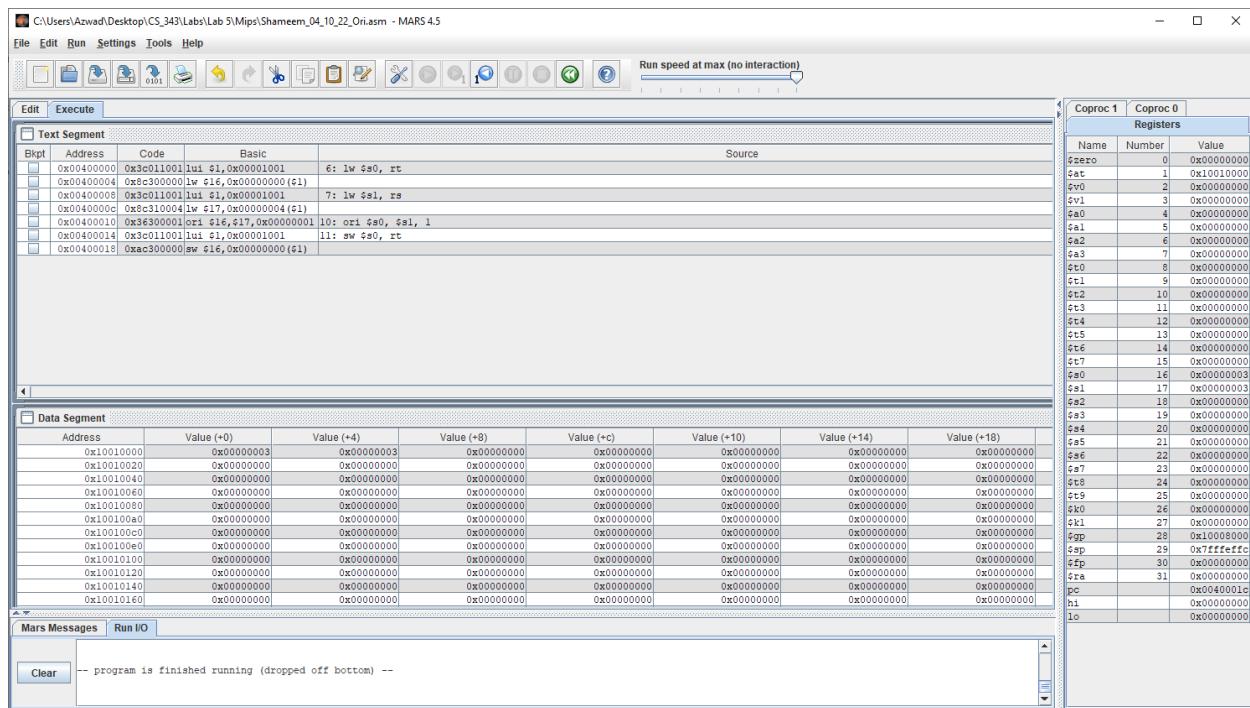


Figure 90: Executed successfully on MARs.

R [rt] (0x00000003) = R[rs] (0x00000003) | ZeroExtImm

Values RT, RS shown clearly in data segment in the order of 0x00000001, 0x00000003. The results of the MIPS code on MARS resulted in the same result as on the waveform simulation in figure 87. Therefore, Ori is the same on MARS and ModelSim waveform, so the VHDL code for Ori is working correctly.

Part III LW

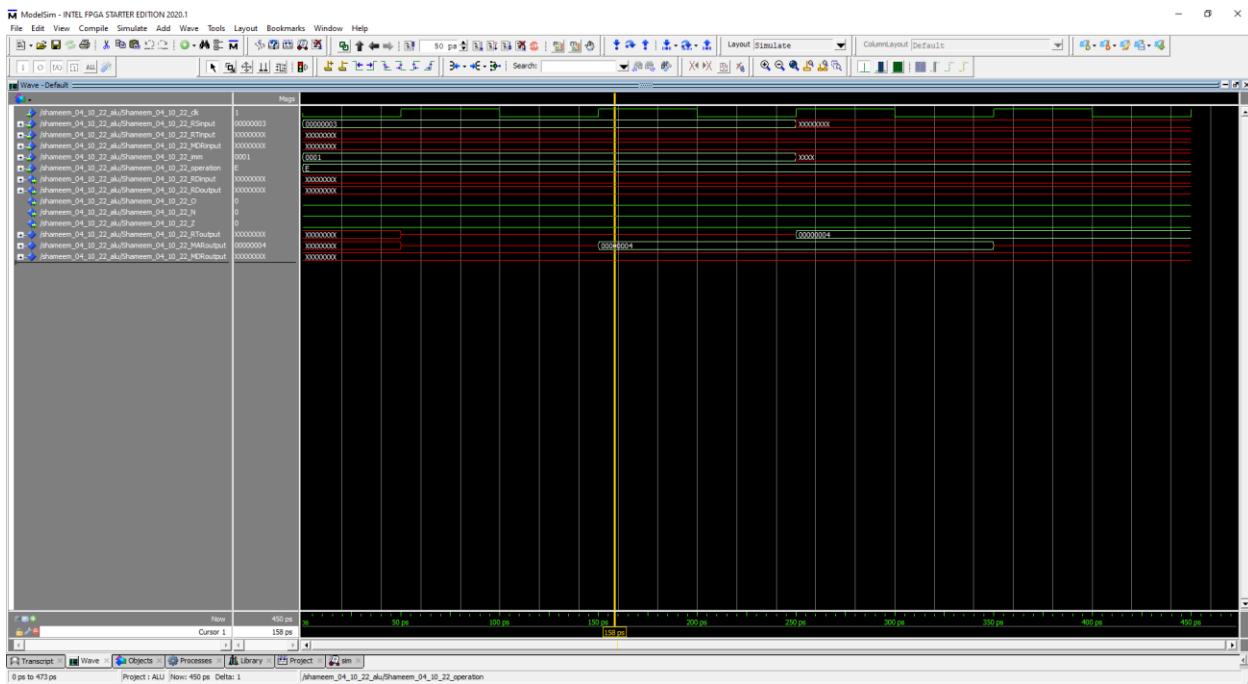


Figure 91: Simulation to represent LW (radix is hexadecimal for everything)

$$R[rt] (0x00000004) = M[R[rs] (0x00000003) + SignExtImm]$$

In this simulation the operation is 1111 which stands for LW, so it takes RS and SignExtImm and adds them together and outputs them into MAR which then goes into RT. Furthermore, 0x00000003 plus 0x00000001 equals 0x00000004, which means the flags zero and negative should be 0 as shown in the waveform because the output is clearly not negative or zero. The overflow flag is also 0 because the output does not trigger the rules of overflow since a positive plus a positive can equal a positive without any overflow.

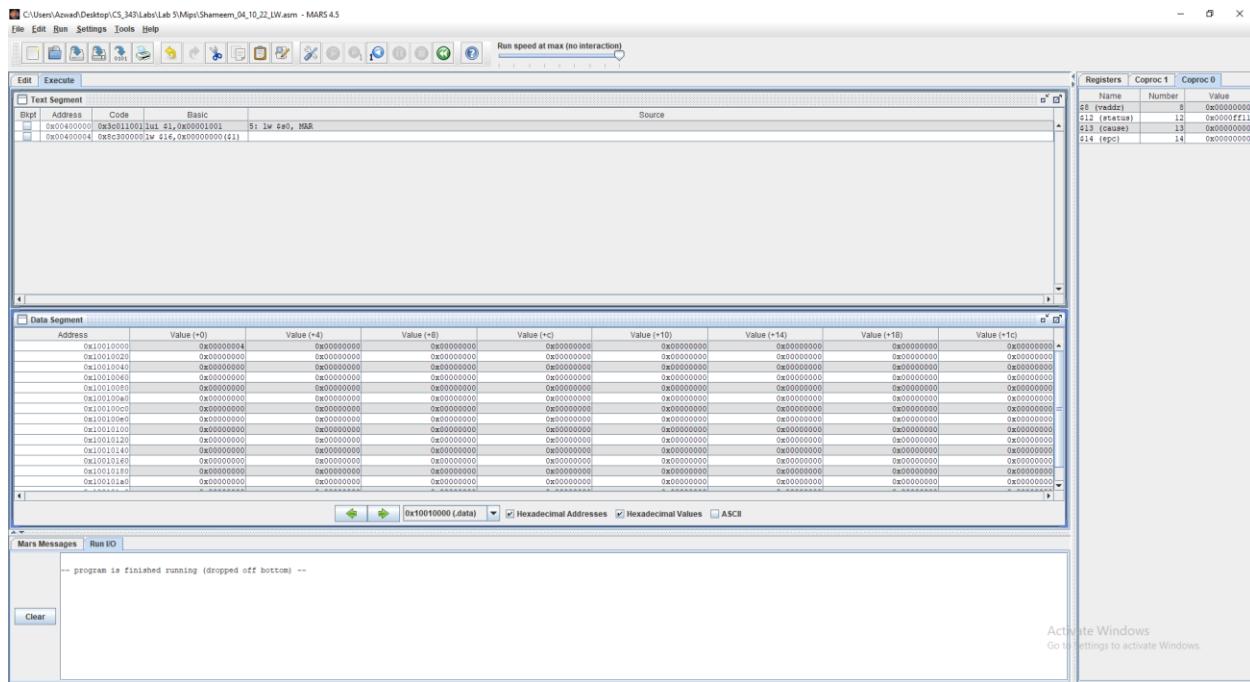


Figure 92: Executed successfully on MARs.

$$R[rt] (0x00000004) = M[R[rs] (0x00000003) + SignExtImm]$$

Values MAR shown clearly in data segment, which is 0x00000004. The results of the MIPS code on MARS resulted in the same result as on the waveform simulation in figure 91. Therefore, lw is the same on MARS and ModelSim waveform, so the VHDL code for lw is working correctly.

Part IV SW

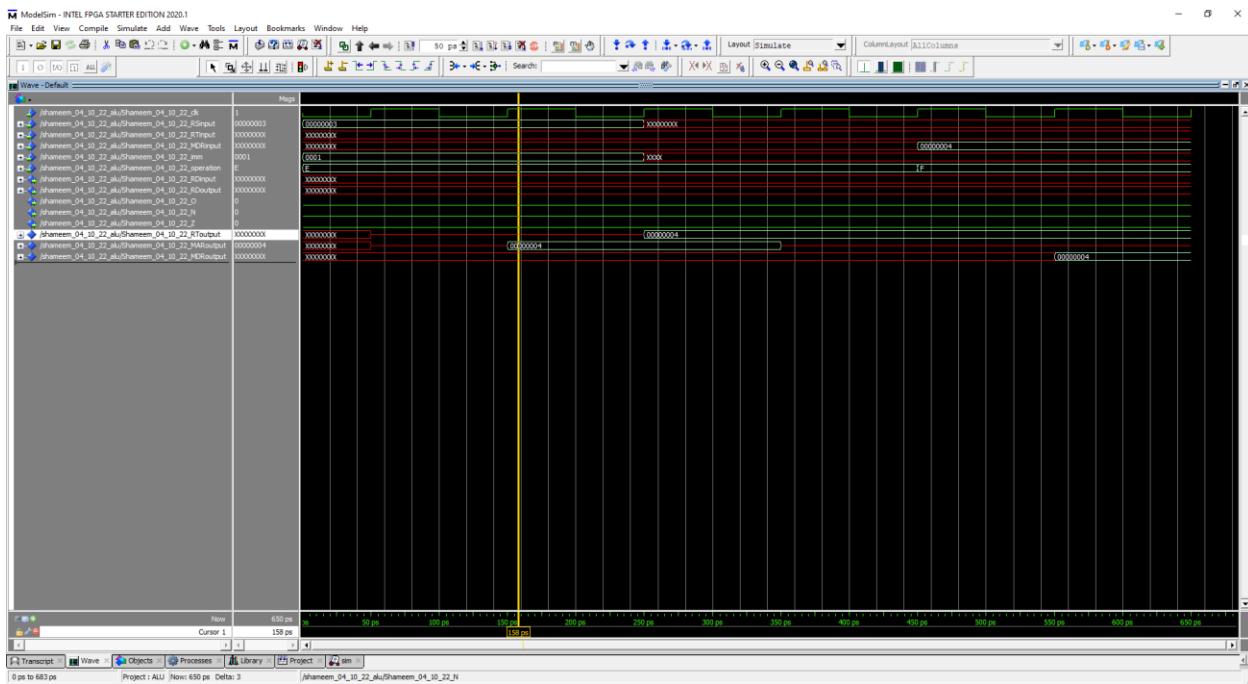


Figure 93: Simulation to represent SW (radix is hexadecimal for everything)

$$R[rt] (0x00000004) = M[R[rs]] (0x00000003) + \text{SignExtImm}$$

In this simulation the operation is 1110 at 450 ps which stands for SW, so it takes RS and SignExtImm and adds them together and outputs them into MDR which then goes into RT. Furthermore, 0x00000003 plus 0x00000001 equals 0x00000004, which means the flags zero and negative should be 0 as shown in the waveform because the output is clearly not negative or zero. The overflow flag is also 0 because the output does not trigger the rules of overflow since a positive plus a positive can equal a positive without any overflow.

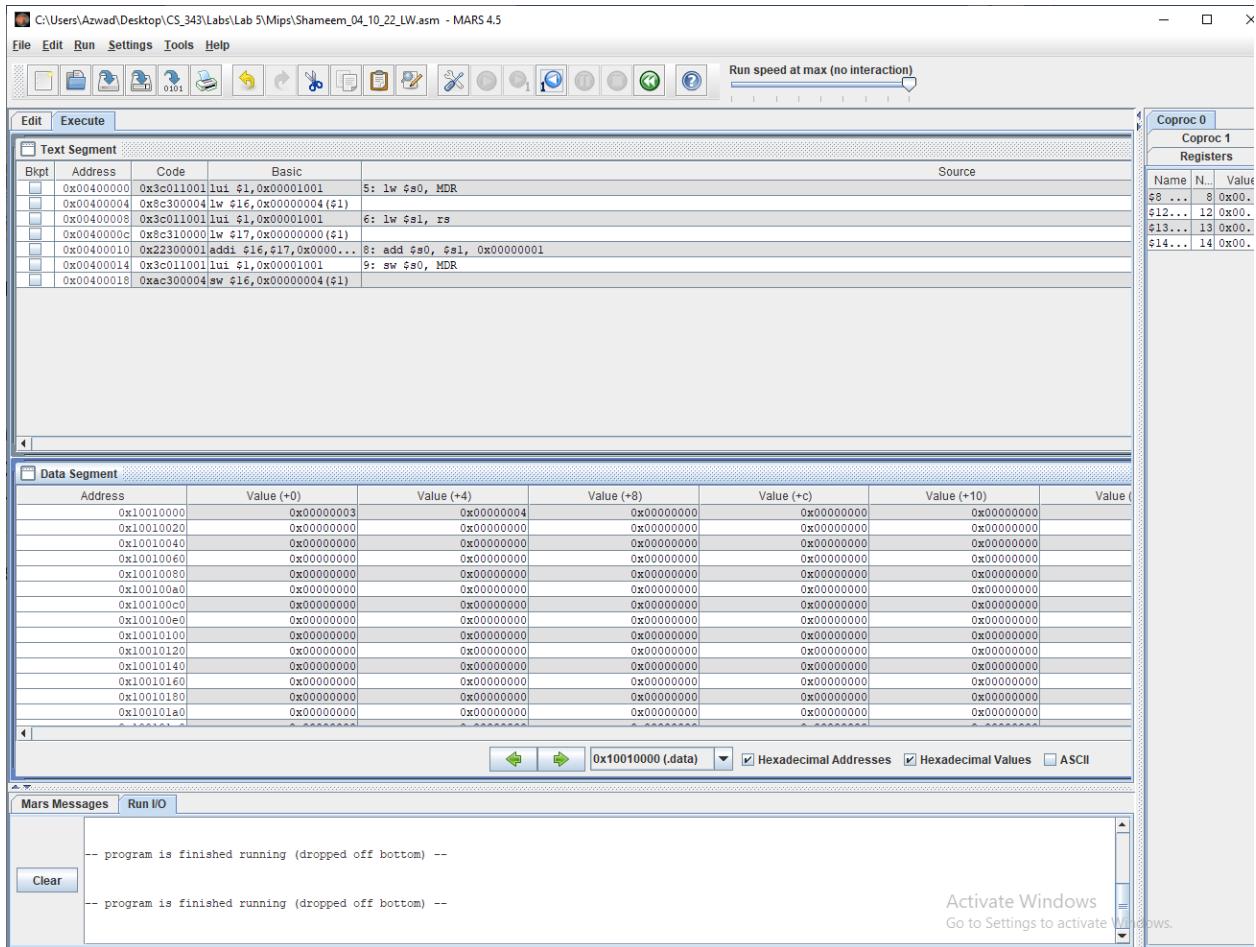


Figure 94: Executed successfully on MARs.

$$R[rt] (0x00000004) = M[R[rs]] (0x00000003) + \text{SignExt}Iimm$$

Values RS, MDR shown clearly in data segment, which is 0x00000003 and 0x00000004. The results of the MIPS code on MARS resulted in the same result as on the waveform simulation in figure 91. Therefore, SW is the same on MARS and ModelSim waveform, so the VHDL code for SW is working correctly.

Conclusion:

The Arithmetic Logic Unit lab was an important step because it allowed us to implement MIPS instructions in VHDL. MIPS instructions such as ADD, ADDU, SUB, SUBU, AND, NOR, OR, SLL, SRA, ADDI, ADDIU, ANDI, ORI, LW, SW was to be implemented by VHDL programming. Furthermore, the VHDL program that implemented MIPS instructions was then to be tested by utilizing ModelSim simulations and MIPS instructions on MARS in order to verify correctness. After completing the Arithmetic Logic Unit lab we have taken a step forward closer to making a CPU because we have proved that we can create instructions that a CPU may have taken through VHDL programming.