SAN JOSE STATE UNIVERSITY

Charles W. Davidson College of Engineering

DEPARTMENT OF ELECTRICAL ENGINEERING

**EE 271 – Advanced Digital System Design and Synthesis**

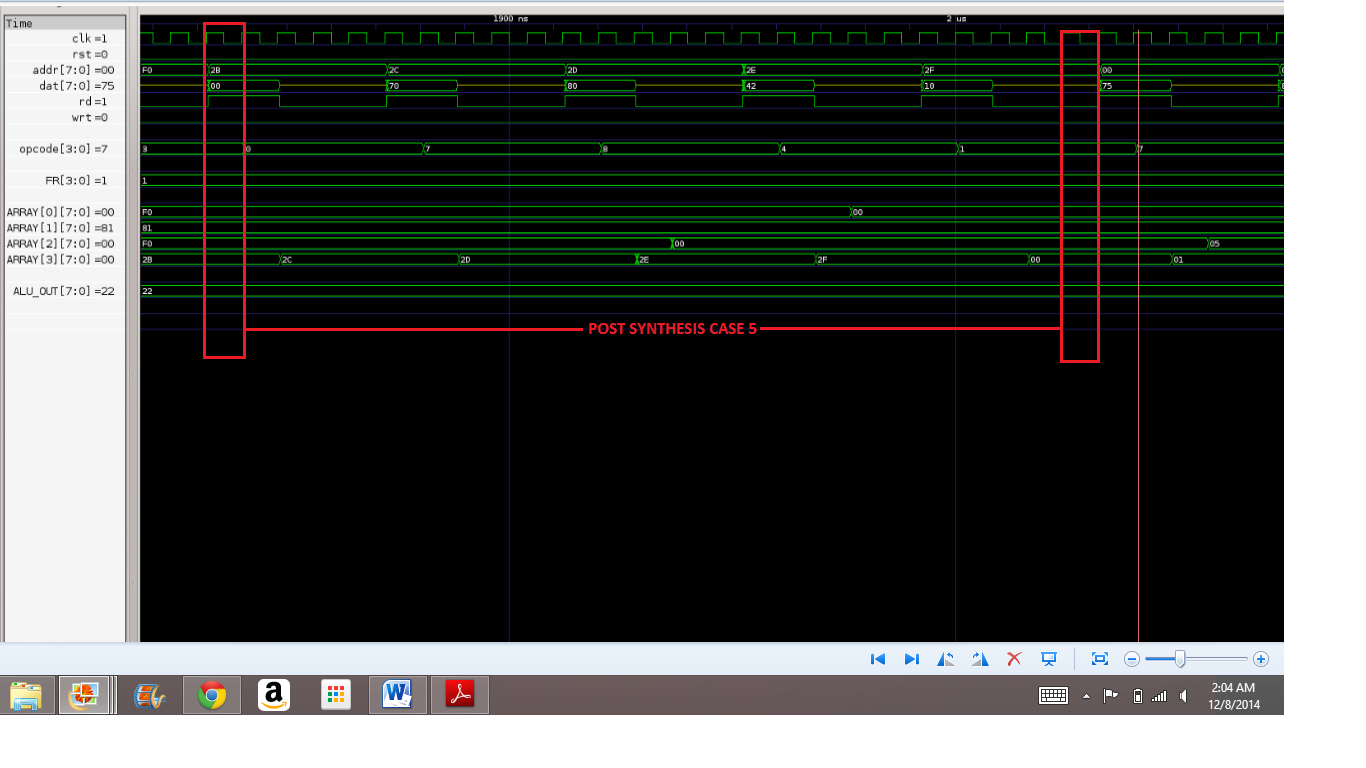
Fall 2014 Final Project Report

**A Simple 8-bit Scalar Processor**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | MOHITKUMAR PATEL | **Email** | mohitkumar.patel@sjsu.edu |
| **SJSUID** | 009288552 | **Phone** | +1 213-300-7130 |

**Executive Summary**

I have designed a Simple 8-bit Scalar Processor which supports subset of MIPS instructions. This Scalar Processor is designed with a Controller, an 8-bit ALU and 256 byte Memory. The machine codes of 10 different instructions are stored in the Instruction Memory ranging from 0 to 127 bytes and Data is stored in Data Memory ranging from 128 to 256 bytes. Controller is the heart of this system which controls all other modules and performs instruction fetch, decode, execution, and write back operations. Moreover, there are four different Registers named General Register (GR), Program Counter Register (PR), Address Register (AR) and Data Register (DR) which are used to transfer the data to or from the Memory. ALU is designed to perform two operations i.e. Additions and Subtractions. The Controller sets all internal Register to zero whenever RESET is asserted. The Controller Fetches the Instruction from the Instruction Memory, Decodes the Instructions and defines Instruction Machine code, Updates the Program Counter Register (PR) for the next instruction to be fetched, and Executes the instruction and stores the data to Data Memory if instructed.

****

## I. General Project Information

## Table I.1: List of EDA Tools Used

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Company** | **Used for** | **Free? (Y/N)** | **Software Documents** |
| VCS | Synopsys | Simulation & test | No |  |
| ModelSim PE Student Edition | Mentor Graphics | Simulation & test | Yes |  |
| Design Vision | Synopsys | Synthesis | No |  |

## Table I.2: List of Libraries Used

|  |  |  |  |
| --- | --- | --- | --- |
| **Library file name** | **Company** | **Used with**  (EDA tool) | **The libraries are at**  (directories on eecad systems) |
| Tc240c.db\_WCCO  M25 | Toshiba | Synopsys | /apps/toshiba/sjsu/synopsys/tc240c/tc240c.db\_  WCCOM25 |
| Tc240c.db\_NOMI  N25 | Toshiba | Synopsys | /apps/toshiba/sjsu/synopsys/tc240c/tc240c.db\_  NOMIN25 |
| Tc240c.db\_BCCO  M25 | Toshiba | Synopsys | /apps/toshiba/sjsu/synopsys/tc240c/tc240c.db\_  BCCOM25 |

## Table I.3: List of Verilog Modules (both design and test modules)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Module Name** | **Input Ports** | **Output Ports** | **Inout Ports** | **Short Description** |
| SCALAR\_PROCESSOR | clk, rst | [7:0]addr, rd, wrt | [7:0]dat | This module is basically the controller unit. Whenever rst signal is asserted, controller sets all internal register to zero and when the clk signal is preset and reset is not asserted, controller provides addr signal to point the instruction address of Instruction Memory which needs to be fetched. Then it reads the Instruction or Data through dat port executes it and writes the result back to Memory according to instruction operations through dat port. |
| SCALAR\_TEST | [7:0]addr, rd, wrt | clk,rst | Dat | This is the test bench for a Scalar Processor which contains 256 byte Memory. Different Memory instructions and Data are stored in Memory for operation. This test bench module is to verify the functions implemented in the controller. |
| ALU | [7:0]a, cin, [7:0]b | [7:0]sum,cout,ov |  | This module is used to perform Addition and Subtraction. Input to this module is provided bu controller unit. Whenever cin is low, ALU performs addition and when cin signal is high ALU performs subtraction between port ‘a’ and port ‘b’ data. The output signal ov is used to define the overflow detection of ALU result |
| RCA\_8 | [7:0]a,  cin,  [7:0]b | [7:0]sum, cout |  | This is 8 bit ripple carry adder used to perform 8 bit number addition between ‘a’ port data and ‘b’ port data and output is achieved at sum port. If the result is above 8 bit then cout bit will go high which indicates carry out. This is the sub module of ALU module. |
| add\_full | a,b,cin | sum,cout |  | This is one bit full adder used to implement 8 bit ripple carry adder. This is the sub module of RCA\_8 module. |
| add\_half | a, b | sum, cout |  | This is one bit half adder used to implement full adder. This is the sub module of add\_full module. |

## II. The Design Overview

An 8-bit Scalar Processor block diagram is shown in figure II.1. The block diagram contains a 4x8 register array which defines AR, GR, DR and PR register for this project. This array is inside the Control unit and operated by controller only. Plus, an 8-bit ALU is connected to Controller unit to perform arithmetic operations such as addition and subtraction. The combination of both Control and ALU unit defines the 8-bit Scalar Processor. There is a Memory of 256 bytes connected to Control Unit to provide instruction machine codes and data. This Memory is basically inside the testbench of Scalar Processor. The Memory has 0 to 127 bytes for machine code and 128 to 256 bytes for data. Control Unit accesses this Memory by asserting rd, wrt and addr signals.

**4 x 8 ARRAY**

**CONTROL UNIT**

**ALU**



PR

DR

GR

AR







[3:0]FR

D



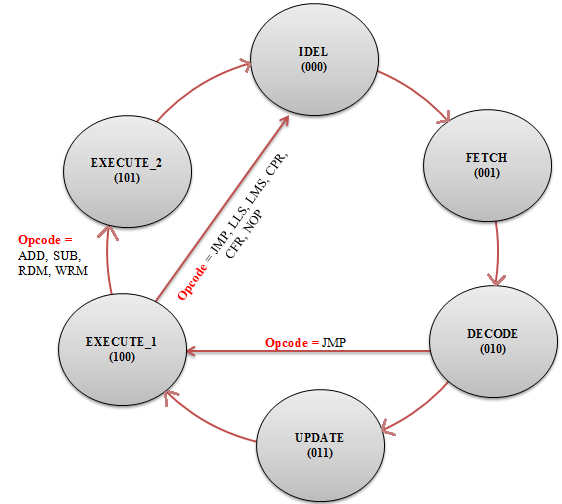
 **SCALAR PROCESSOR**



**256 BYTE MEMORY**

**TEST BENCH**

**Figure II.1**: Overall Processor Block Diagram

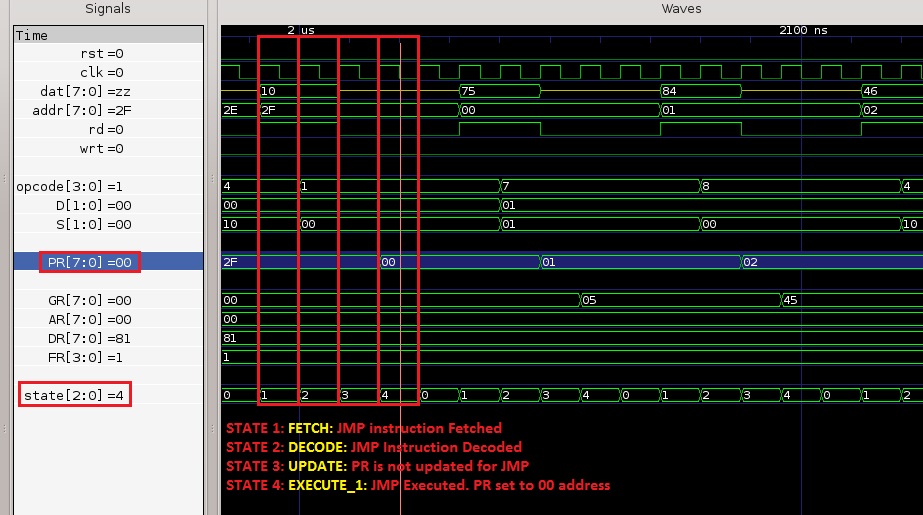


**Figure II.2**: Overall State Transition Diagram(s)

## Table II.1: Number of Clock Cycles Required for Each Instruction

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Instruction** | **# of Clock Cycles** | **Instruction** | **# of Clock Cycles** | **Instruction** | **# of Clock Cycles** |
| NOP | 1 | LMS | 1 | ADD | 2 |
| JMP | 1 | CFR | 1 | SUB | 2 |
| CPR | 1 | RDM | 2 |  |  |
| LLS | 1 | WRM | 2 |  |  |

During simulation, I analyzed that for ADD and SUB instructions, it takes one clock cycle to give the input to ALU by Control Unit. Since ALU is a combinational block, it gives the output in the same clock cycle. However, storing the ALU output to destination register takes one more clock cycle for both ALU and SUB instructions. Moreover, for RDM and WRM instructions ,the processor takes one clock cycle to set the memory address to read or write and one clock cycle to read from or write to Memory Unit. Therefore, the optimum number of clock cycles for ADD, SUB, RDM and WRM instruction are two that I was able to achieve.

**III. RTL-Level Simulations/Tests for Each Instruction**

**Figure III.1**: Test Result for **JMP** Instruction

**Simulation Result for JMP:**

1992 ns FETCH STATE: Instruction is Fetched

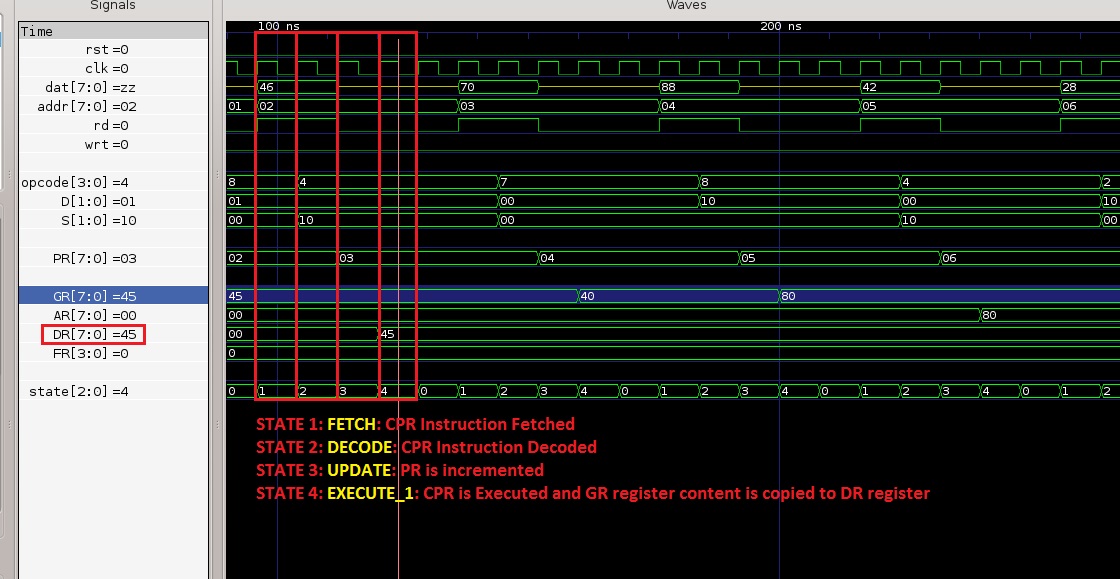
2000 ns DECODE STATE: Instruction is Decoded

2008 ns Instruction : JMP

UPDATE STATE:PR Register is not Incremented to Fetch the next Instruction.

2016 ns EXECUTE\_1 STATE: Executed JMP and Jumped to Instruction stored at 00

Memory Location



**Figure III.2**: Test Result for **CPR** Instruction

**Simulation Result for CPR:**

96 ns FETCH STATE: Instruction is Fetched

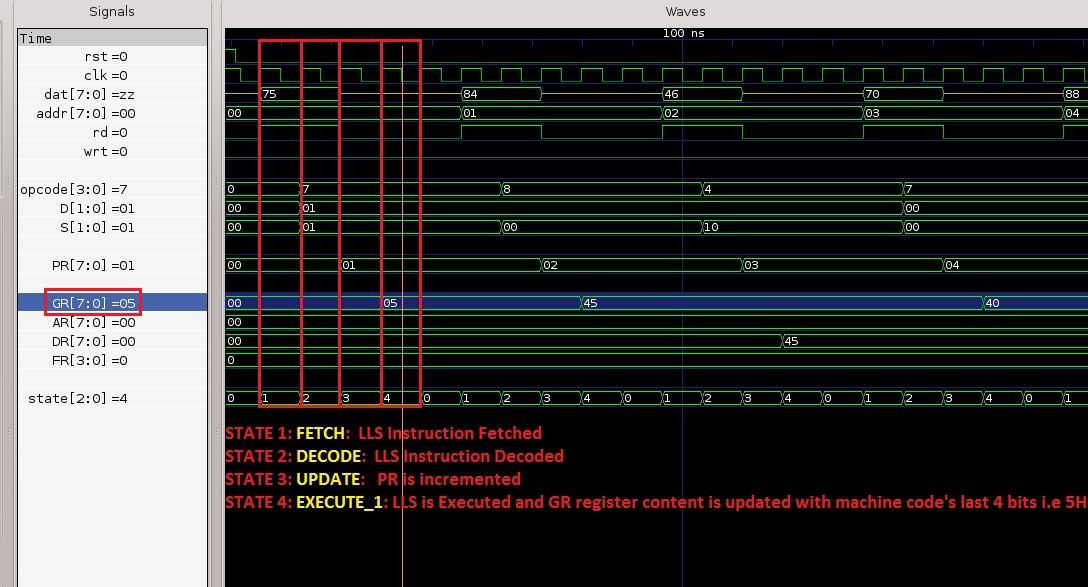
104 ns DECODE STATE: Instruction is Decoded

112 ns Instruction : CPR, 01, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

120 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 01 reg.

AR[00]= 00, DR[01]= 45, GR[10]= 45, PR[11]= 03



**Figure III.3**: Test Result for **LLS** Instruction

**Simulation Result for LLS:**

16 ns FETCH STATE: Instruction is Fetched

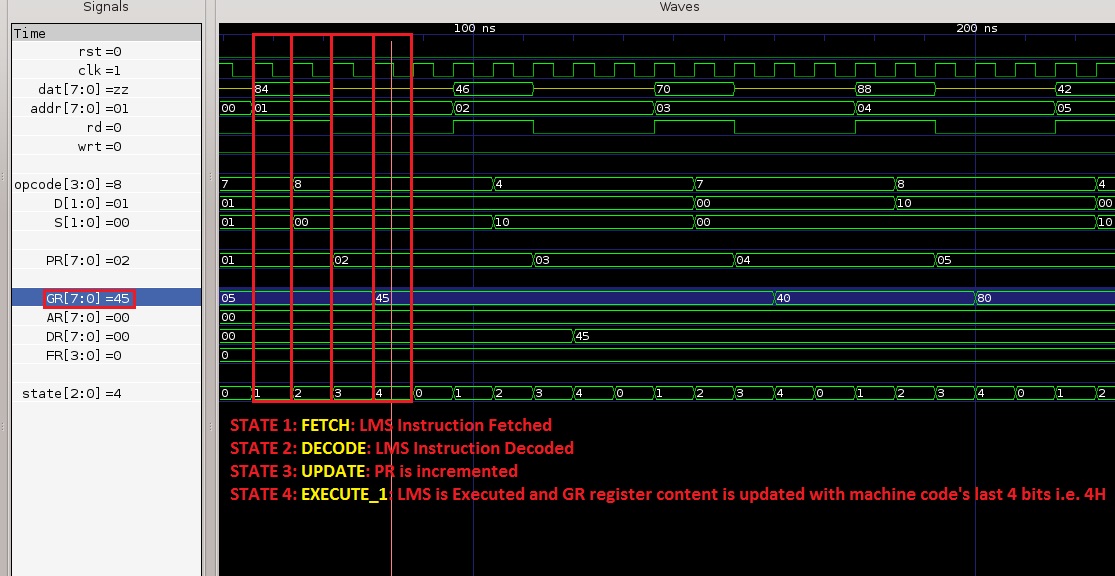
24 ns DECODE STATE: Instruction is Decoded

32 ns Instruction : LLS, 5

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

40 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 5

AR[00]= 00, DR[01]= 00, GR[10]= 05, PR[11]= 01



**Figure III.4**: Test Result for **LMS** Instruction

**Simulation Result for LMS:**

56 ns FETCH STATE: Instruction is Fetched

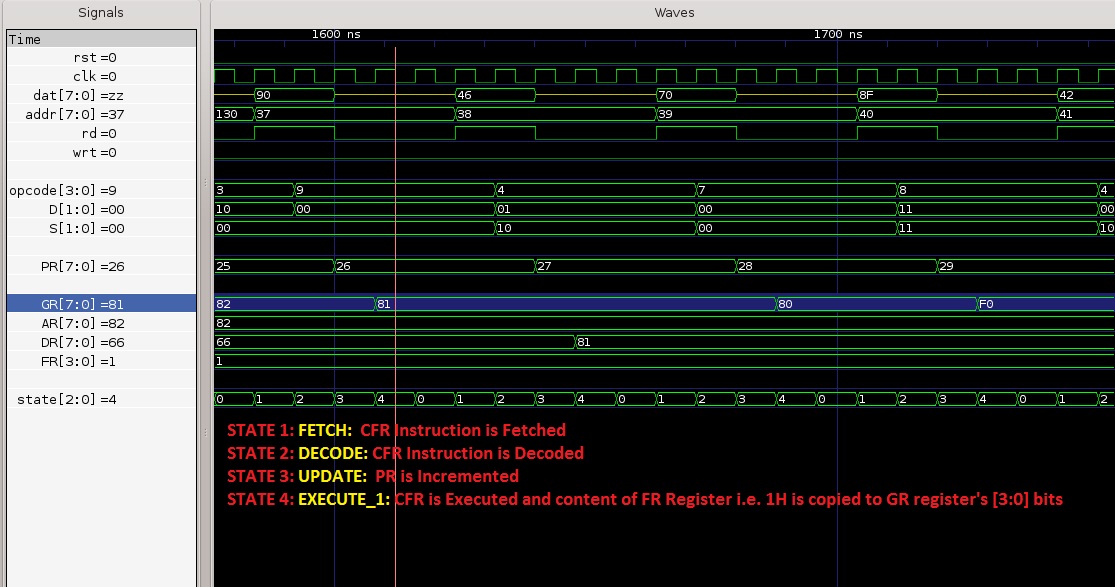
64 ns DECODE STATE: Instruction is Decoded

72 ns Instruction : LMS, 4

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

80 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 4

AR[00]= 00, DR[01]= 00, GR[10]= 45, PR[11]= 02



**Figure III.5**: Test Result for **CFR** Instruction

**Simulation Result for CFR:**

1584 ns FETCH STATE: Instruction is Fetched

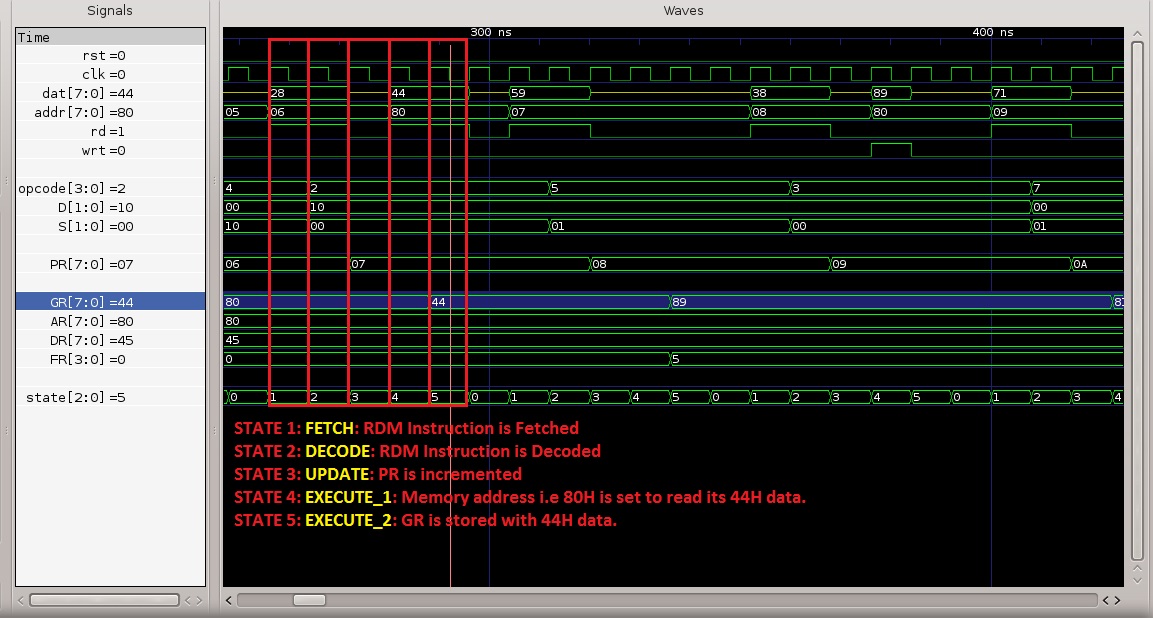
1592 ns DECODE STATE: Instruction is Decoded

1600 ns Instruction : CFR;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1608 ns EXECUTE\_1 STATE: Executed CFR and Stored data at LSBs of GR = 1

AR[00]= 82, DR[01]= 66, GR[10]= 81, PR[11]= 26



**Figure III.6**: Test Result for **RDM** Instruction

**Simulation Result for RDM:**

256 ns FETCH STATE: Instruction is Fetched

264 ns DECODE STATE: Instruction is Decoded

272 ns Instruction : RDM, 10;

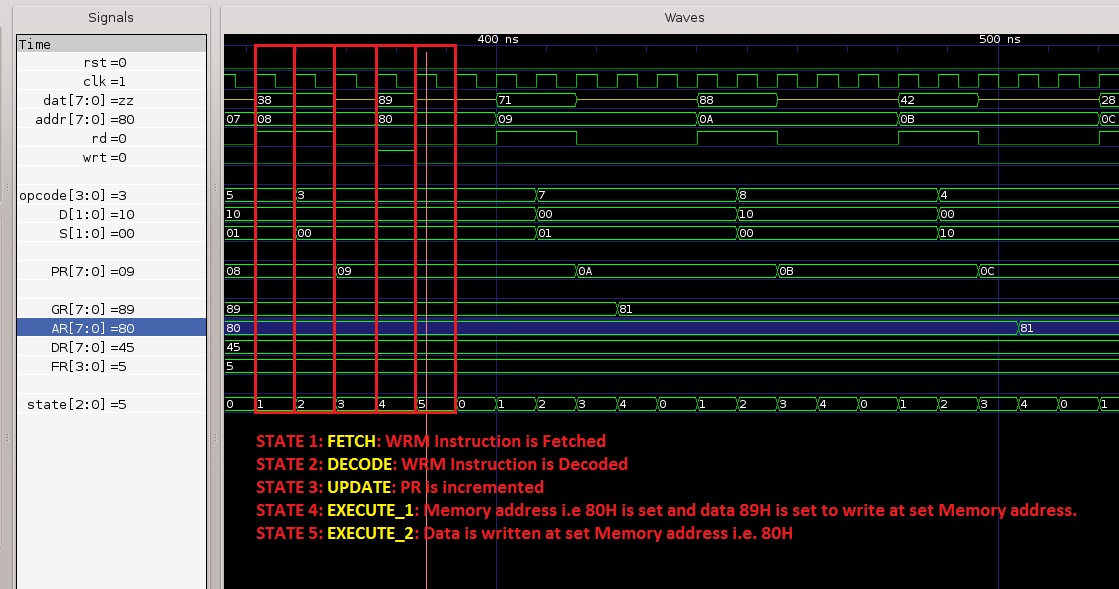
UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

280 ns EXECUTE\_1 STATE: Executing RDM...Setting AR = 80 to Memory Address...

288 ns EXECUTE\_2 STATE: Read the Data 44 from Memory Location 80 and Stored

into 10 Register

AR[00]= 80, DR[01]= 45, GR[10]= 44, PR[11]= 07



**Figure III.7**: Test Result for **WRM** Instruction

**Simulation Result for WRM:**

352 ns FETCH STATE: Instruction is Fetched

360 ns DECODE STATE: Instruction is Decoded

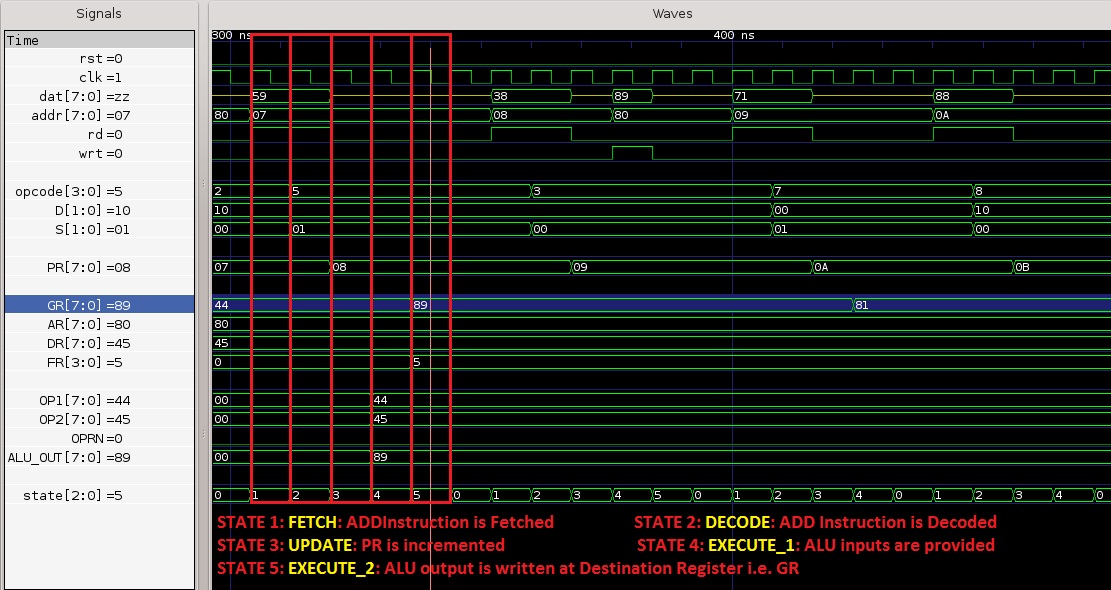
368 ns Instruction : WRM, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

376 ns EXECUTE\_1 STATE: Executing WRM...Setting AR = 80 to Memory Address...

384 ns EXECUTE\_2 STATE: 10 Register Data 89 is Written at Memory Address 80

WRITEEN DATA : RAM[80h]= 89, RAM[81h]= 55, RAM[82h]= 66, RAM[FO]= xx



**Figure III.8**: Test Result for **ADD** Instruction

**Simulation Result for ADD:**

304 ns FETCH STATE: Instruction is Fetched

312 ns DECODE STATE: Instruction is Decoded

320 ns Instruction : ADD, 10, 01;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

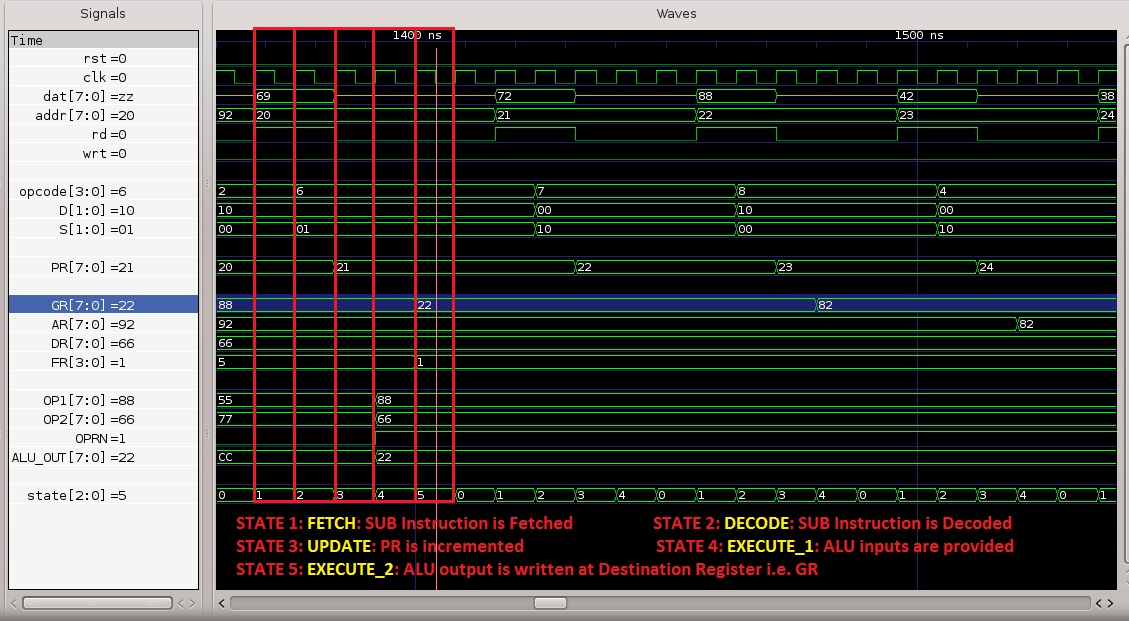
328 ns EXECUTE\_1 STATE: Executing ADD...Providing ALU Opcodes 44 and 45 and

Adding them...

336 ns EXECUTE\_2 STATE: Added the 10 Register Data 44 with 01 Register Data

45 and the Result is Stored into 10 Register

AR[00]= 80, DR[01]= 45, GR[10]= 89, PR[11]= 08



**Figure III.9**: Test Result for **SUB** Instruction

**Simulation Result for SUB:**

1368 ns FETCH STATE: Instruction is Fetched

1376 ns DECODE STATE: Instruction is Decoded

1384 ns Instruction : SUB, 10, 01;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1392 ns EXECUTE\_1 STATE: Executing SUB...Providing ALU Opcodes 88 and 66 and

Adding them...

1400 ns EXECUTE\_2 STATE: Subtracted the 01 Register Data 66 from 10 Register

Data 88 and the Result is Stored into 10 Register

AR[00]= 92, DR[01]= 66, GR[10]= 22, PR[11]= 21

**IV. RTL-Level Simulations/Tests of Whole Design (processor)**

IV.1 Test Program, Machine Codes, and Test Data

## Table IV.1: Memory Code Address (in Hex), Test Instruction, and Machine Code (in Hex)

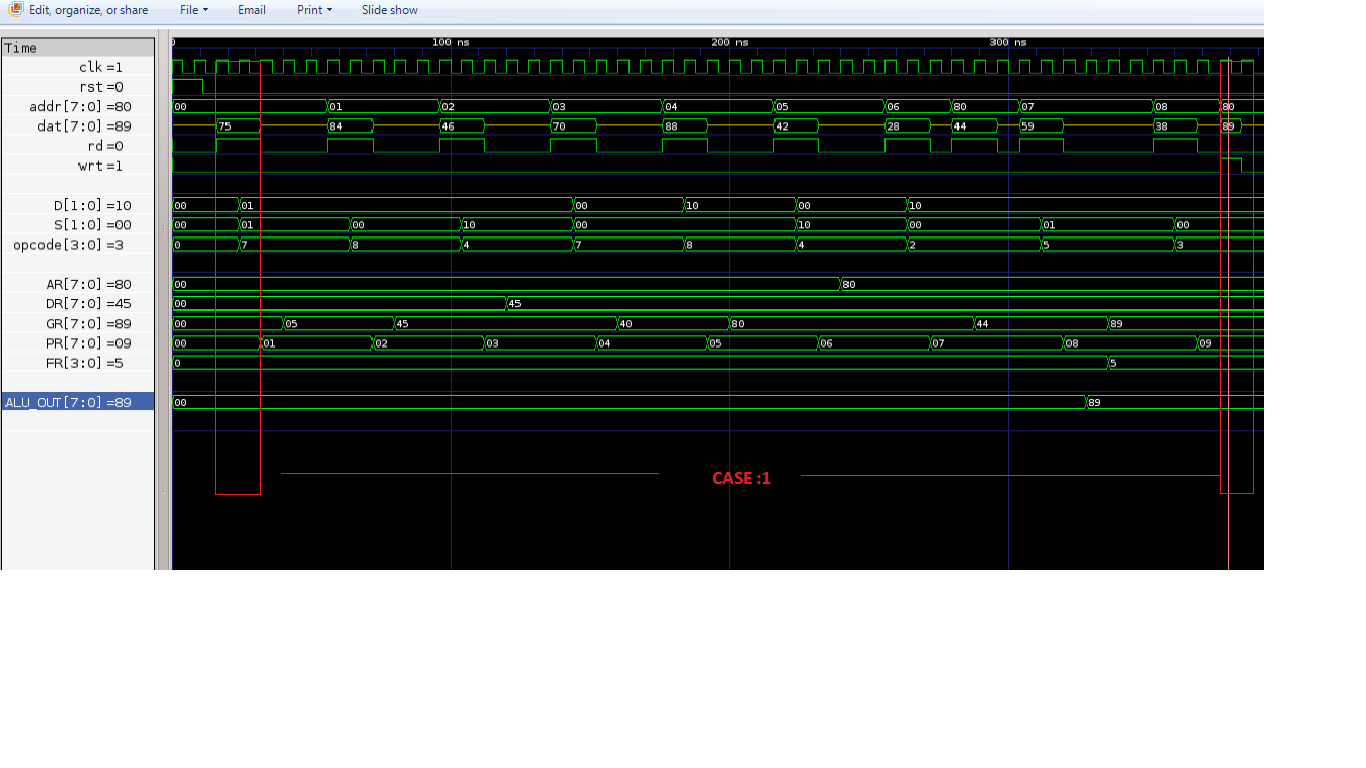
|  |  |  |
| --- | --- | --- |
| **Memory Address** | **Instruction** | **Machine Code** |
| 0x00 | LLS | 0x75 |
| 0x01 | LMS | 0x84 |
| 0x02 | CPR DR, GR | 0x46 |
| 0x03 | LLS | 0x70 |
| 0x04 | LMS | 0x88 |
| 0x05 | CPR AR, GR | 0x42 |
| 0x06 | RDM GR | 0x28 |
| 0x07 | ADD GR, DR | 0x59 |
| 0x08 | WRM GR | 0x38 |
| 0x09 | LLS | 0x71 |
| 0x0a | LMS | 0x88 |
| 0x0b | CPR AR, GR | 0x42 |
| 0x0c | RDM GR | 0x28 |
| 0x0d | CPR DR, GR | 0x46 |
| 0x0e | LLS | 0x71 |
| 0x0f | LMS | 0x89 |
| 0x10 | CPR AR, GR | 0x42 |
| 0x11 | RDM GR | 0x28 |
| 0x12 | ADD DR, GR | 0x56 |
| 0x13 | LLS | 0x71 |
| 0x14 | LMS | 0x88 |
| 0x15 | CPR AR, GR | 0x42 |
| 0x16 | WRM DR | 0x34 |
| 0x17 | LLS | 0x72 |
| 0x18 | LMS | 0x88 |
| 0x19 | CPR AR, GR | 0x42 |
| 0x1a | RDM GR | 0x28 |
| 0x1b | CPR DR,GR | 0x46 |
| 0x1c | LLS | 0x72 |
| 0x1d | LMS | 0x89 |
| 0x1e | CPR AR, GR | 0x42 |
| 0x1f | RDM GR | 0x28 |
| 0x20 | SUB GR, DR | 0x66 |
| 0x21 | LLS | 0x72 |
| 0x22 | LMS | 0x88 |
| 0x23 | CPR AR, GR | 0x42 |
| 0x24 | WRM GR | 0x34 |
| 0x25 | CFR | 0x90 |
| 0x26 | CPR DR, GR | 0x46 |
| 0x27 | LLS | 0x70 |
| 0x28 | LMS | 0x8f |
| 0x29 | CPR AR, GR | 0x42 |
| 0x2a | WRM DR | 0x34 |
| 0x2b | NOP | 0x00 |
| 0x2c | LLS | 0x70 |
| 0x2d | LMS | 0x80 |
| 0x2e | CPR AR, GR | 0x42 |
| 0x2f | JMP | 0x10 |

## Table IV.2: Initial Test Data Stored at Memory Addresses (in Hex)

|  |  |  |  |
| --- | --- | --- | --- |
| **Memory Address** | **Data** | **Memory Address** | **Data** |
| 0x80 | 0x44 | 0x91 | 0x77 |
| 0x81 | 0x55 | 0x92 | 0x88 |
| 0x82 | 0x66 |  |  |

IV.2 Test Results

* **Case 1:**

****

WRITEEN DATA : RAM[80h]= 44, RAM[81h]= 55, RAM[82h]= 66, RAM[FO]= xx

16 ns FETCH STATE: Instruction is Fetched

24 ns DECODE STATE: Instruction is Decoded

32 ns Instruction : LLS, 5

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

40 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 5

AR[00]= 00, DR[01]= 00, GR[10]= 05, PR[11]= 01

56 ns FETCH STATE: Instruction is Fetched

64 ns DECODE STATE: Instruction is Decoded

72 ns Instruction : LMS, 4

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

80 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 4

AR[00]= 00, DR[01]= 00, GR[10]= 45, PR[11]= 02

96 ns FETCH STATE: Instruction is Fetched

104 ns DECODE STATE: Instruction is Decoded

112 ns Instruction : CPR, 01, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

120 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 01 reg.

AR[00]= 00, DR[01]= 45, GR[10]= 45, PR[11]= 03

136 ns FETCH STATE: Instruction is Fetched

144 ns DECODE STATE: Instruction is Decoded

152 ns Instruction : LLS, 0

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

160 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 0

AR[00]= 00, DR[01]= 45, GR[10]= 40, PR[11]= 04

176 ns FETCH STATE: Instruction is Fetched

184 ns DECODE STATE: Instruction is Decoded

192 ns Instruction : LMS, 8

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

200 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 8

AR[00]= 00, DR[01]= 45, GR[10]= 80, PR[11]= 05

216 ns FETCH STATE: Instruction is Fetched

224 ns DECODE STATE: Instruction is Decoded

232 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

240 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00 reg.

AR[00]= 80, DR[01]= 45, GR[10]= 80, PR[11]= 06

256 ns FETCH STATE: Instruction is Fetched

264 ns DECODE STATE: Instruction is Decoded

272 ns Instruction : RDM, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

280 ns EXECUTE\_1 STATE: Executing RDM...Setting AR = 80 to Memory Address...

288 ns EXECUTE\_2 STATE: Read the Data 44 from Memory Location 80 and Stored

into 10 Register

AR[00]= 80, DR[01]= 45, GR[10]= 44, PR[11]= 07

304 ns FETCH STATE: Instruction is Fetched

312 ns DECODE STATE: Instruction is Decoded

320 ns Instruction : ADD, 10, 01;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

328 ns EXECUTE\_1 STATE: Executing ADD...Providing ALU Opcodes 44 and 45 and

Adding them...

336 ns EXECUTE\_2 STATE: Added the 10 Register Data 44 with 01 Register Data

45 and the Result is Stored into 10 Register

AR[00]= 80, DR[01]= 45, GR[10]= 89, PR[11]= 08

352 ns FETCH STATE: Instruction is Fetched

360 ns DECODE STATE: Instruction is Decoded

368 ns Instruction : WRM, 10;

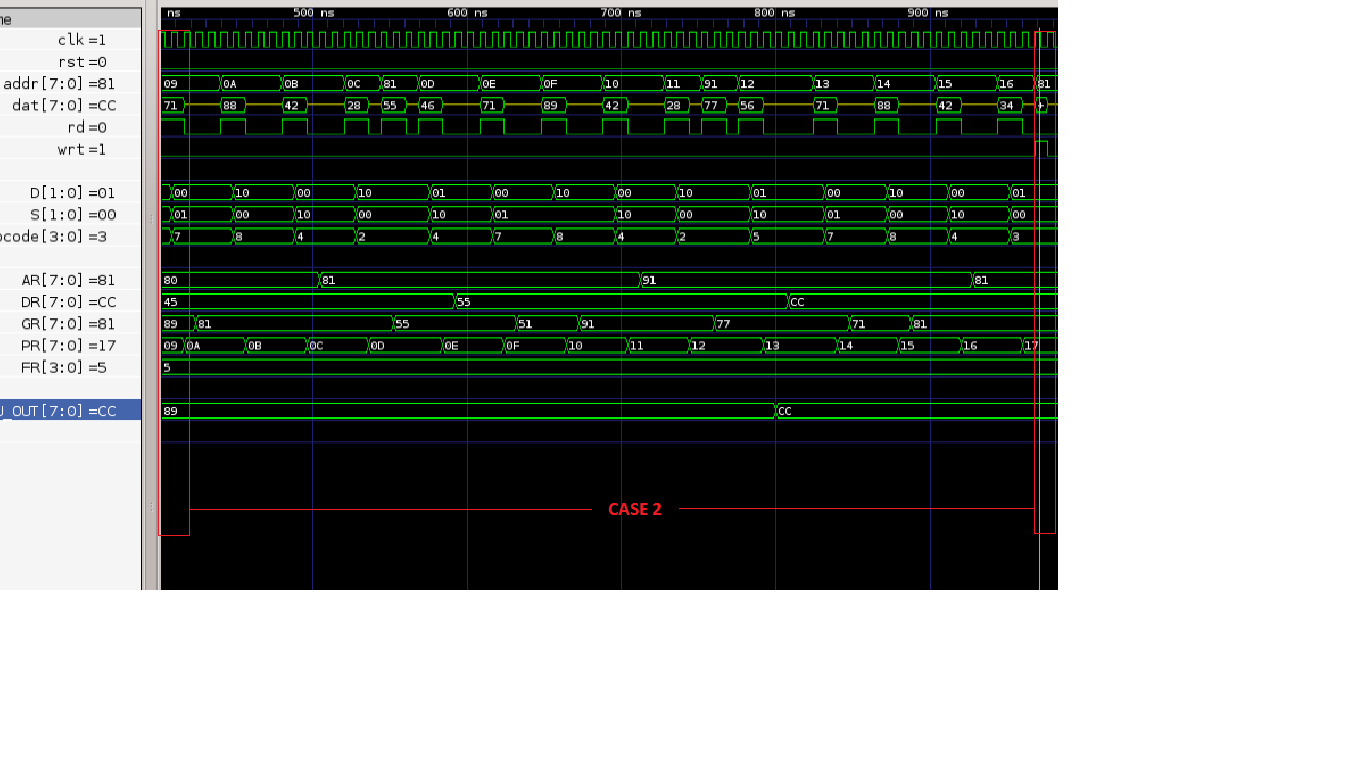
UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

376 ns EXECUTE\_1 STATE: Executing WRM...Setting AR = 80 to Memory Address...

384 ns EXECUTE\_2 STATE: 10 Register Data 89 is Written at Memory Address 80

WRITEEN DATA : RAM[80h]= 89, RAM[81h]= 55, RAM[82h]= 66, RAM[FO]= xx

* **Case 2:**

****

400 ns FETCH STATE: Instruction is Fetched

408 ns DECODE STATE: Instruction is Decoded

416 ns Instruction : LLS, 1

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

424 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 1

AR[00]= 80, DR[01]= 45, GR[10]= 81, PR[11]= 0a

440 ns FETCH STATE: Instruction is Fetched

448 ns DECODE STATE: Instruction is Decoded

456 ns Instruction : LMS, 8

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

464 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 8

AR[00]= 80, DR[01]= 45, GR[10]= 81, PR[11]= 0b

480 ns FETCH STATE: Instruction is Fetched

488 ns DECODE STATE: Instruction is Decoded

496 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

504 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00

register

AR[00]= 81, DR[01]= 45, GR[10]= 81, PR[11]= 0c

520 ns FETCH STATE: Instruction is Fetched

528 ns DECODE STATE: Instruction is Decoded

536 ns Instruction : RDM, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

544 ns EXECUTE\_1 STATE: Executing RDM...Setting AR = 81 to Memory Address...

552 ns EXECUTE\_2 STATE: Read the Data 55 from Memory Location 81 and Stored

into 10 Register

AR[00]= 81, DR[01]= 45, GR[10]= 55, PR[11]= 0d

568 ns FETCH STATE: Instruction is Fetched

576 ns DECODE STATE: Instruction is Decoded

584 ns Instruction : CPR, 01, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

592 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 01

register

AR[00]= 81, DR[01]= 55, GR[10]= 55, PR[11]= 0e

608 ns FETCH STATE: Instruction is Fetched

616 ns DECODE STATE: Instruction is Decoded

624 ns Instruction : LLS, 1

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

632 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 1

AR[00]= 81, DR[01]= 55, GR[10]= 51, PR[11]= 0f

648 ns FETCH STATE: Instruction is Fetched

656 ns DECODE STATE: Instruction is Decoded

664 ns Instruction : LMS, 9

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

672 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 9

AR[00]= 81, DR[01]= 55, GR[10]= 91, PR[11]= 10

688 ns FETCH STATE: Instruction is Fetched

696 ns DECODE STATE: Instruction is Decoded

704 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

712 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00

register

AR[00]= 91, DR[01]= 55, GR[10]= 91, PR[11]= 11

728 ns FETCH STATE: Instruction is Fetched

736 ns DECODE STATE: Instruction is Decoded

744 ns Instruction : RDM, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

752 ns EXECUTE\_1 STATE: Executing RDM...Setting AR = 91 to Memory Address...

760 ns EXECUTE\_2 STATE: Read the Data 77 from Memory Location 91 and Stored

into 10 Register

AR[00]= 91, DR[01]= 55, GR[10]= 77, PR[11]= 12

776 ns FETCH STATE: Instruction is Fetched

784 ns DECODE STATE: Instruction is Decoded

792 ns Instruction : ADD, 01, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

800 ns EXECUTE\_1 STATE: Executing ADD...Providing ALU Opcodes 55 and 77 and

Adding them...

808 ns EXECUTE\_2 STATE: Added the 01 Register Data 55 with 10 Register Data

77 and the Result is Stored into 01 Register

AR[00]= 91, DR[01]= cc, GR[10]= 77, PR[11]= 13

824 ns FETCH STATE: Instruction is Fetched

832 ns DECODE STATE: Instruction is Decoded

840 ns Instruction : LLS, 1

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

848 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 1

AR[00]= 91, DR[01]= cc, GR[10]= 71, PR[11]= 14

864 ns FETCH STATE: Instruction is Fetched

872 ns DECODE STATE: Instruction is Decoded

880 ns Instruction : LMS, 8

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

888 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 8

AR[00]= 91, DR[01]= cc, GR[10]= 81, PR[11]= 15

904 ns FETCH STATE: Instruction is Fetched

912 ns DECODE STATE: Instruction is Decoded

920 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

928 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00

register

AR[00]= 81, DR[01]= cc, GR[10]= 81, PR[11]= 16

944 ns FETCH STATE: Instruction is Fetched

952 ns DECODE STATE: Instruction is Decoded

960 ns Instruction : WRM, 01;

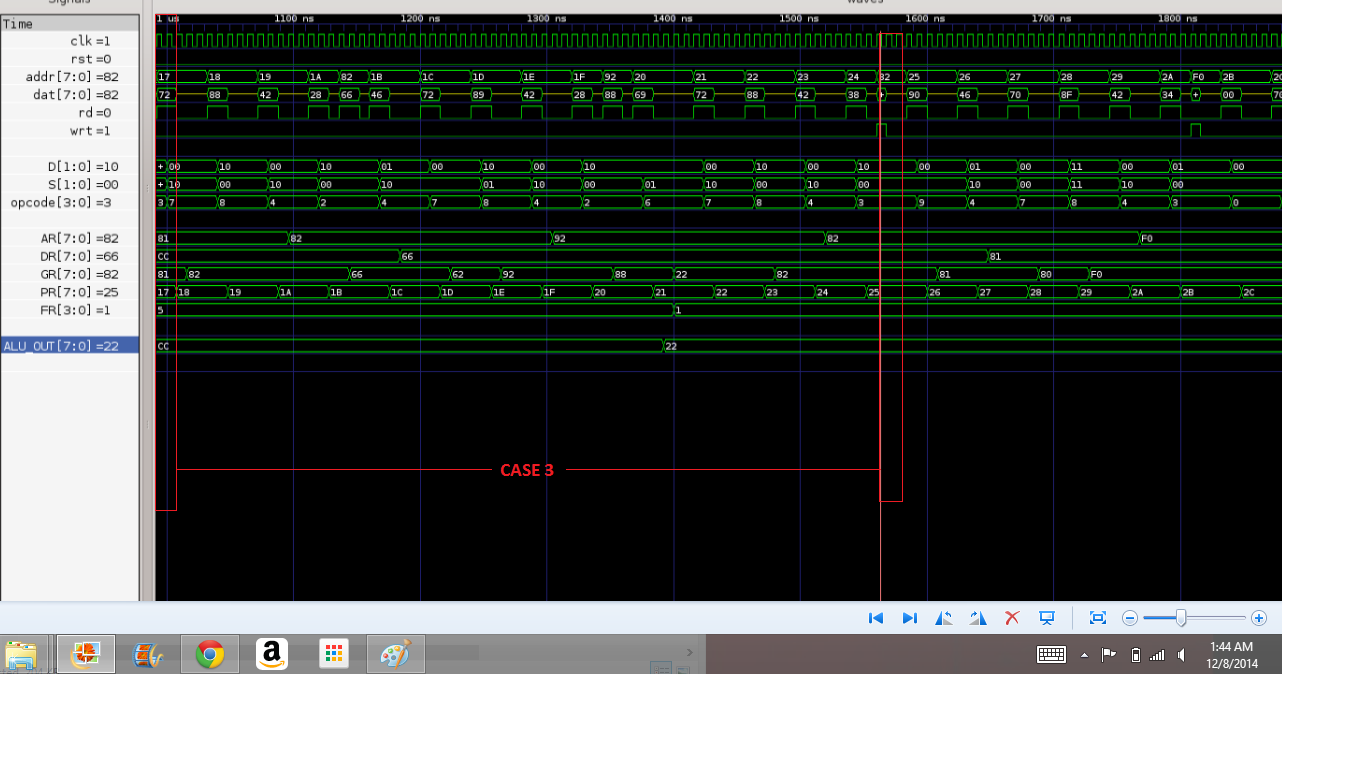
UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

968 ns EXECUTE\_1 STATE: Executing WRM...Setting AR = 81 to Memory Address...

976 ns EXECUTE\_2 STATE: 01 Register Data cc is Written at Memory Address 81

WRITTEN DATA : RAM[80h]= 89, RAM[81h]= cc, RAM[82h]= 66, RAM[FO]= xx

* **Case 3:**

****

992 ns FETCH STATE: Instruction is Fetched

1000 ns DECODE STATE: Instruction is Decoded

1008 ns Instruction : LLS, 2

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1016 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 2

AR[00]= 81, DR[01]= cc, GR[10]= 82, PR[11]= 18

1032 ns FETCH STATE: Instruction is Fetched

1040 ns DECODE STATE: Instruction is Decoded

1048 ns Instruction : LMS, 8

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1056 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 8

AR[00]= 81, DR[01]= cc, GR[10]= 82, PR[11]= 19

1072 ns FETCH STATE: Instruction is Fetched

1080 ns DECODE STATE: Instruction is Decoded

1088 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1096 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00 register

AR[00]= 82, DR[01]= cc, GR[10]= 82, PR[11]= 1a

1112 ns FETCH STATE: Instruction is Fetched

1120 ns DECODE STATE: Instruction is Decoded

1128 ns Instruction : RDM, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1136 ns EXECUTE\_1 STATE: Executing RDM...Setting AR = 82 to Memory Address...

1144 ns EXECUTE\_2 STATE: Read the Data 66 from Memory Location 82 and Stored into 10 Register

AR[00]= 82, DR[01]= cc, GR[10]= 66, PR[11]= 1b

1160 ns FETCH STATE: Instruction is Fetched

1168 ns DECODE STATE: Instruction is Decoded

1176 ns Instruction : CPR, 01, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1184 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 01

register

AR[00]= 82, DR[01]= 66, GR[10]= 66, PR[11]= 1c

1200 ns FETCH STATE: Instruction is Fetched

1208 ns DECODE STATE: Instruction is Decoded

1216 ns Instruction : LLS, 2

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1224 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 2

AR[00]= 82, DR[01]= 66, GR[10]= 62, PR[11]= 1d

1240 ns FETCH STATE: Instruction is Fetched

1248 ns DECODE STATE: Instruction is Decoded

1256 ns Instruction : LMS, 9

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1264 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 9

AR[00]= 82, DR[01]= 66, GR[10]= 92, PR[11]= 1e

1280 ns FETCH STATE: Instruction is Fetched

1288 ns DECODE STATE: Instruction is Decoded

1296 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1304 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00

register

AR[00]= 92, DR[01]= 66, GR[10]= 92, PR[11]= 1f

1320 ns FETCH STATE: Instruction is Fetched

1328 ns DECODE STATE: Instruction is Decoded

1336 ns Instruction : RDM, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1344 ns EXECUTE\_1 STATE: Executing RDM...Setting AR = 92 to Memory Address...

1352 ns EXECUTE\_2 STATE: Read the Data 88 from Memory Location 92 and Stored into 10 Register

AR[00]= 92, DR[01]= 66, GR[10]= 88, PR[11]= 20

1368 ns FETCH STATE: Instruction is Fetched

1376 ns DECODE STATE: Instruction is Decoded

1384 ns Instruction : SUB, 10, 01;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1392 ns EXECUTE\_1 STATE: Executing SUB...Providing ALU Opcodes 88 and 66 and Adding them...

1400 ns EXECUTE\_2 STATE: Subtracted the 01 Register Data 66 from 10 Register

Data 88 and the Result is Stored into 10 Register

AR[00]= 92, DR[01]= 66, GR[10]= 22, PR[11]= 21

1416 ns FETCH STATE: Instruction is Fetched

1424 ns DECODE STATE: Instruction is Decoded

1432 ns Instruction : LLS, 2

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1440 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 2

AR[00]= 92, DR[01]= 66, GR[10]= 22, PR[11]= 22

1456 ns FETCH STATE: Instruction is Fetched

1464 ns DECODE STATE: Instruction is Decoded

1472 ns Instruction : LMS, 8

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1480 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 8

AR[00]= 92, DR[01]= 66, GR[10]= 82, PR[11]= 23

1496 ns FETCH STATE: Instruction is Fetched

1504 ns DECODE STATE: Instruction is Decoded

1512 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1520 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00

register

AR[00]= 82, DR[01]= 66, GR[10]= 82, PR[11]= 24

1536 ns FETCH STATE: Instruction is Fetched

1544 ns DECODE STATE: Instruction is Decoded

1552 ns Instruction : WRM, 10;

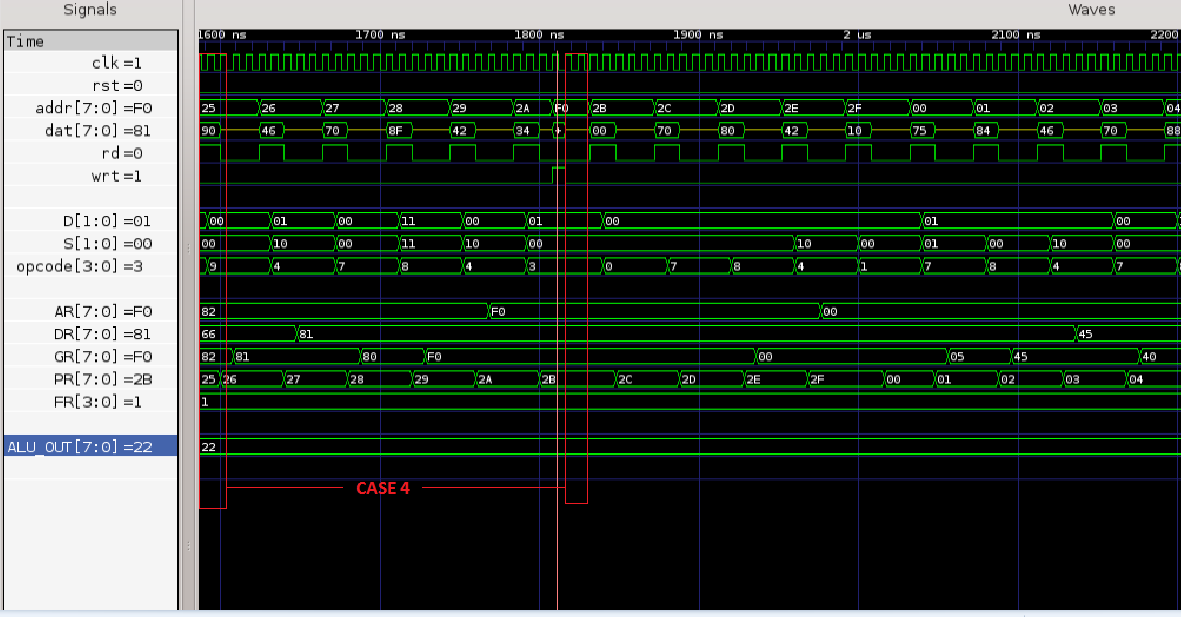
UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1560 ns EXECUTE\_1 STATE: Executing WRM...Setting AR = 82 to Memory Address...

1568 ns EXECUTE\_2 STATE: 10 Register Data 82 is Written at Memory Address 82

WRITTEN DATA : RAM[80h]= 89, RAM[81h]= cc, RAM[82h]= 82, RAM[FO]= xx

* **Case 4:**



1584 ns FETCH STATE: Instruction is Fetched

1592 ns DECODE STATE: Instruction is Decoded

1600 ns Instruction : CFR;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1608 ns EXECUTE\_1 STATE: Executed CFR and Stored data at LSBs of GR = 1

AR[00]= 82, DR[01]= 66, GR[10]= 81, PR[11]= 26

1624 ns FETCH STATE: Instruction is Fetched

1632 ns DECODE STATE: Instruction is Decoded

1640 ns Instruction : CPR, 01, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1648 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 01 register

AR[00]= 82, DR[01]= 81, GR[10]= 81, PR[11]= 27

1664 ns FETCH STATE: Instruction is Fetched

1672 ns DECODE STATE: Instruction is Decoded

1680 ns Instruction : LLS, 0

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1688 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 0

AR[00]= 82, DR[01]= 81, GR[10]= 80, PR[11]= 28

1704 ns FETCH STATE: Instruction is Fetched

1712 ns DECODE STATE: Instruction is Decoded

1720 ns Instruction : LMS, f

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1728 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = f

AR[00]= 82, DR[01]= 81, GR[10]= f0, PR[11]= 29

1744 ns FETCH STATE: Instruction is Fetched

1752 ns DECODE STATE: Instruction is Decoded

1760 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1768 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00 register

AR[00]= f0, DR[01]= 81, GR[10]= f0, PR[11]= 2a

1784 ns FETCH STATE: Instruction is Fetched

1792 ns DECODE STATE: Instruction is Decoded

1800 ns Instruction : WRM, 01;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1808 ns EXECUTE\_1 STATE: Executing WRM...Setting AR = f0 to Memory Address...

1816 ns EXECUTE\_2 STATE: 01 Register Data 81 is Written at Memory Address f0

WRITTEN DATA : RAM[80h]= 89, RAM[81h]= cc, RAM[82h]= 82, RAM[FO]= 81

1832 ns FETCH STATE: Instruction is Fetched

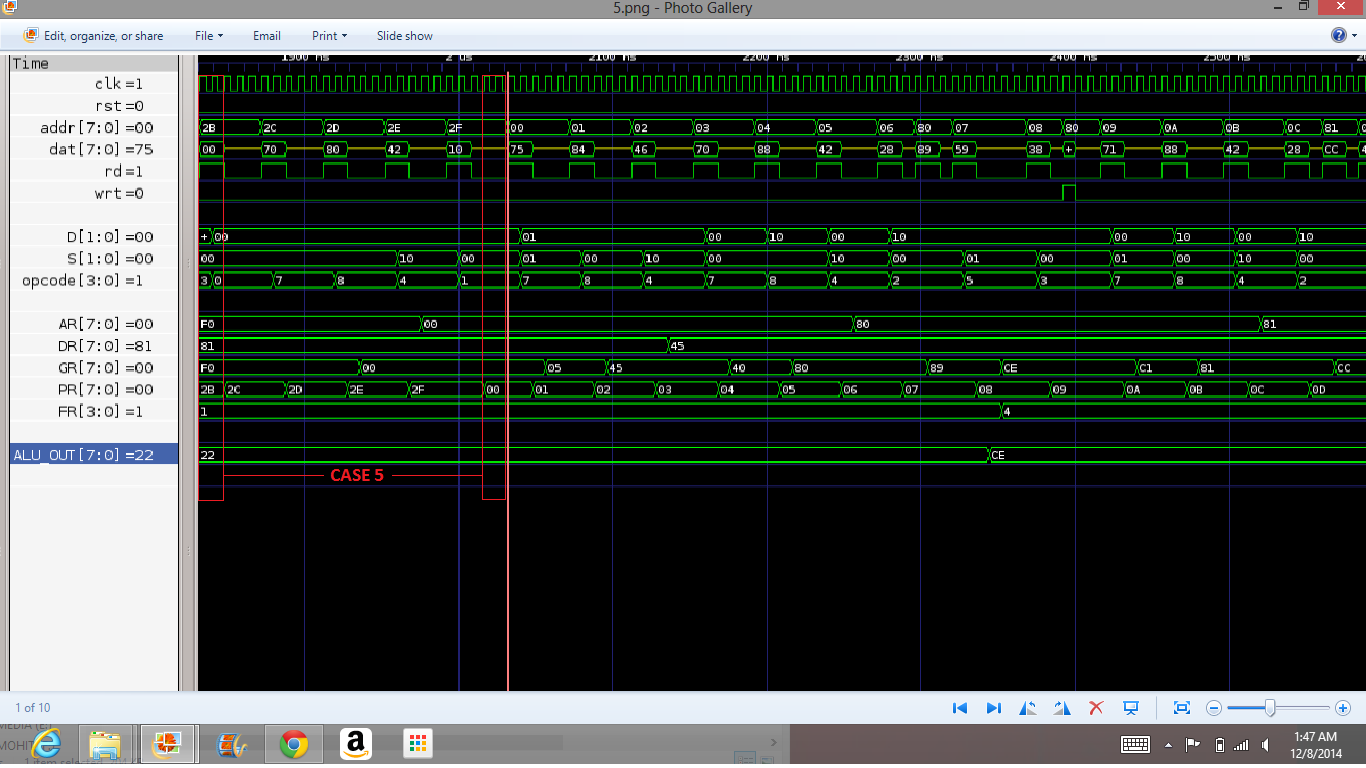
1840 ns DECODE STATE: Instruction is Decoded

1848 ns Instruction : NOP

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1856 ns EXECUTE\_1 STATE: Executed NOP and No Operation is Perfomed

* **Case 5:**

****

1872 ns FETCH STATE: Instruction is Fetched

1880 ns DECODE STATE: Instruction is Decoded

1888 ns Instruction : LLS, 0

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1896 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 0

AR[00]= f0, DR[01]= 81, GR[10]= f0, PR[11]= 2d

1912 ns FETCH STATE: Instruction is Fetched

1920 ns DECODE STATE: Instruction is Decoded

1928 ns Instruction : LMS, 0

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1936 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 0

AR[00]= f0, DR[01]= 81, GR[10]= 00, PR[11]= 2e

1952 ns FETCH STATE: Instruction is Fetched

1960 ns DECODE STATE: Instruction is Decoded

1968 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1976 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00

register

AR[00]= 00, DR[01]= 81, GR[10]= 00, PR[11]= 2f

1992 ns FETCH STATE: Instruction is Fetched

2000 ns DECODE STATE: Instruction is Decoded

2008 ns Instruction : JMP

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2016 ns EXECUTE\_1 STATE: Executed JMP and Jumped to Instruction stored at 00

Memory Location

**V. Synthesis and Optimizations of Whole Design (processor)**

In the table below, I have shown 6 different trials that I have gone through in synthesizing and optimizing my design. I have set different constrains like Area, Clock etc. and observer the optimized cell area and optimized power for my design.

## Table V.1: Synthesis Constraints and Results

|  |  |  |
| --- | --- | --- |
| **Trial #** | **Constraint settings** (area, clock, delay, etc.) | **Synthesis results**  (time slack, area, power, etc.) |
| 1 | Clock :2.5ns  Area 1500 | SLACK: -0.02 (Violated) ; Cell Area:1389  Power: 2.8892 mW |
| 2 | Clock : 3ns  Area 1300 | SLACK: 0.00 (Met) ; Cell Area:1163  Power: 2.6742 mW |
| 3 | Clock :4ns  Area 1200 | SLACK: 0.00 (Met) ; Cell Area:1108  Power: 1.8995 mW |
| 4 | Clock : 5ns  Area 1100 | SLACK: 0.03 (Met) ; Cell Area:1089  Power: 1.8995 mW |
| 5 | Clock : 6ns  Area 1000 | SLACK: 0.05 (Met) ; Cell Area:1089  Power: 1.6743 mW |
| 6 | Clock : 7ns  Area 900 | SLACK: 0.05 (Met) ; Cell Area:1089  Power: 1.6159 mW |

**Figure V.1**: Area versus Clock of the Final Design

PR (11)

DR (10)

GR (01)

AR (00)

**Figure V.1**: Power versus Clock of the Final Design

PR (11)

DR (10)

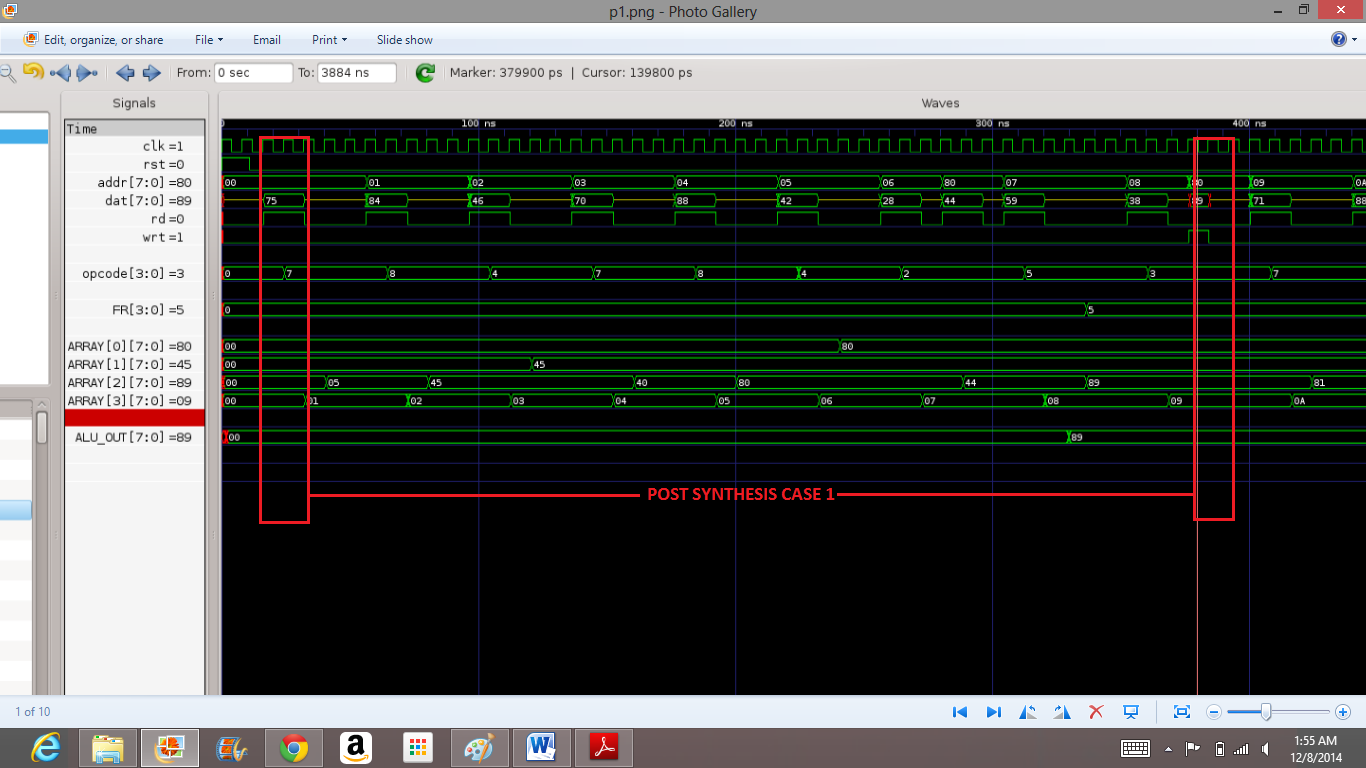
GR (01)

AR (00)

By, setting different constrains in synthesis script, I observed that when we increase the clock period, the cell area reduces and therefor the dynamic power consumption is also reduced. Moreover, when we increase the clock period the SLACK increases and if we decrease the clock period, the SLACK decreases. The Slack becomes negative when I set the clock period below 3 ns. Plus, the optimize number of cell area I observed was 1108 and the maximum dynamic power of the circuit is 2.4745 mW.

**VI. Gate-Level Simulations/Tests of Whole Design (processor)**

* **Case 1:**

****

WRITEEN DATA : RAM[80h]= 44, RAM[81h]= 55, RAM[82h]= 66, RAM[FO]= xx

16 ns FETCH STATE: Instruction is Fetched

24 ns DECODE STATE: Instruction is Decoded

32 ns Instruction : LLS, 5

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

40 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 5

AR[00]= 00, DR[01]= 00, GR[10]= 05, PR[11]= 01

56 ns FETCH STATE: Instruction is Fetched

64 ns DECODE STATE: Instruction is Decoded

72 ns Instruction : LMS, 4

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

80 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 4

AR[00]= 00, DR[01]= 00, GR[10]= 45, PR[11]= 02

96 ns FETCH STATE: Instruction is Fetched

104 ns DECODE STATE: Instruction is Decoded

112 ns Instruction : CPR, 01, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

120 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 01 reg.

AR[00]= 00, DR[01]= 45, GR[10]= 45, PR[11]= 03

136 ns FETCH STATE: Instruction is Fetched

144 ns DECODE STATE: Instruction is Decoded

152 ns Instruction : LLS, 0

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

160 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 0

AR[00]= 00, DR[01]= 45, GR[10]= 40, PR[11]= 04

176 ns FETCH STATE: Instruction is Fetched

184 ns DECODE STATE: Instruction is Decoded

192 ns Instruction : LMS, 8

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

200 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 8

AR[00]= 00, DR[01]= 45, GR[10]= 80, PR[11]= 05

216 ns FETCH STATE: Instruction is Fetched

224 ns DECODE STATE: Instruction is Decoded

232 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

240 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00 reg.

AR[00]= 80, DR[01]= 45, GR[10]= 80, PR[11]= 06

256 ns FETCH STATE: Instruction is Fetched

264 ns DECODE STATE: Instruction is Decoded

272 ns Instruction : RDM, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

280 ns EXECUTE\_1 STATE: Executing RDM...Setting AR = 80 to Memory Address...

288 ns EXECUTE\_2 STATE: Read the Data 44 from Memory Location 80 and Stored

into 10 Register

AR[00]= 80, DR[01]= 45, GR[10]= 44, PR[11]= 07

304 ns FETCH STATE: Instruction is Fetched

312 ns DECODE STATE: Instruction is Decoded

320 ns Instruction : ADD, 10, 01;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

328 ns EXECUTE\_1 STATE: Executing ADD...Providing ALU Opcodes 44 and 45 and

Adding them...

336 ns EXECUTE\_2 STATE: Added the 10 Register Data 44 with 01 Register Data

45 and the Result is Stored into 10 Register

AR[00]= 80, DR[01]= 45, GR[10]= 89, PR[11]= 08

352 ns FETCH STATE: Instruction is Fetched

360 ns DECODE STATE: Instruction is Decoded

368 ns Instruction : WRM, 10;

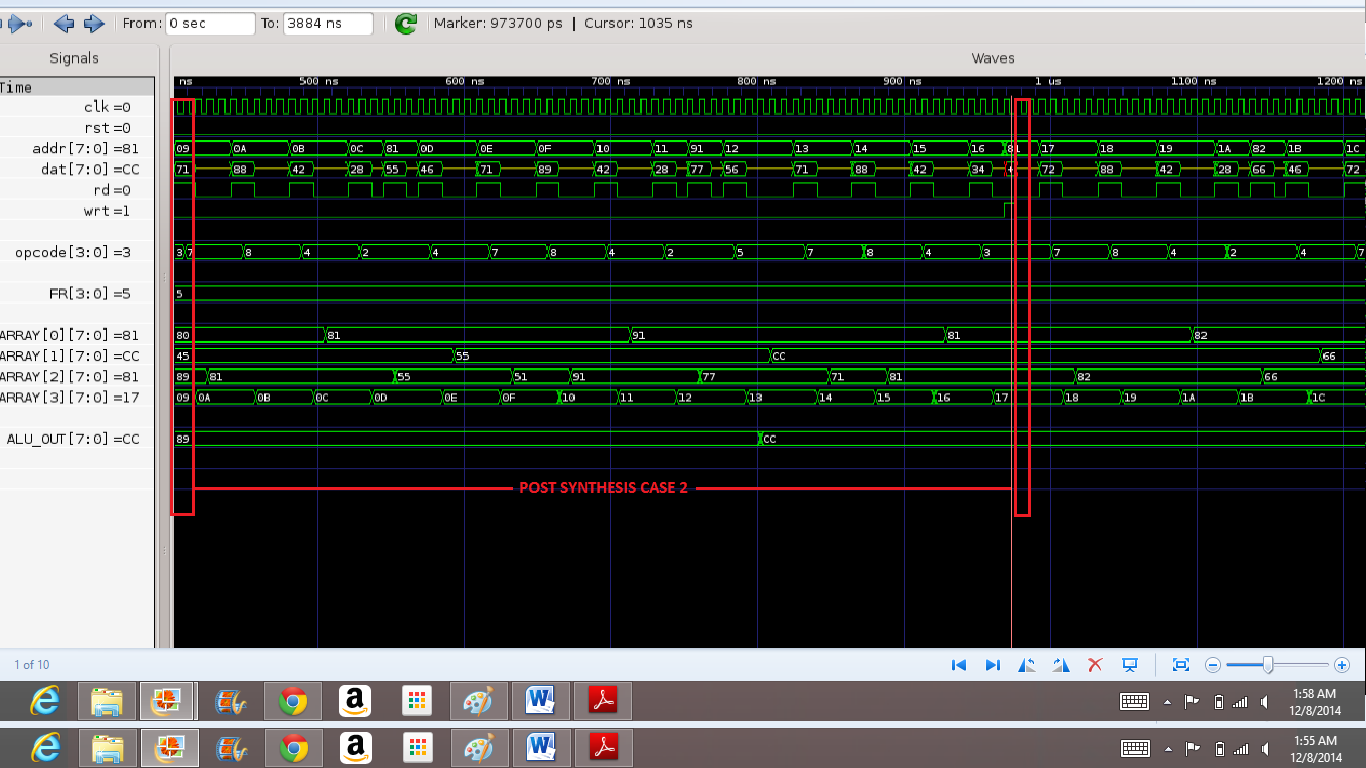
UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

376 ns EXECUTE\_1 STATE: Executing WRM...Setting AR = 80 to Memory Address...

384 ns EXECUTE\_2 STATE: 10 Register Data 89 is Written at Memory Address 80

WRITEEN DATA : RAM[80h]= 89, RAM[81h]= 55, RAM[82h]= 66, RAM[FO]= xx

* **Case 2:**

****

400 ns FETCH STATE: Instruction is Fetched

408 ns DECODE STATE: Instruction is Decoded

416 ns Instruction : LLS, 1

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

424 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 1

AR[00]= 80, DR[01]= 45, GR[10]= 81, PR[11]= 0a

440 ns FETCH STATE: Instruction is Fetched

448 ns DECODE STATE: Instruction is Decoded

456 ns Instruction : LMS, 8

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

464 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 8

AR[00]= 80, DR[01]= 45, GR[10]= 81, PR[11]= 0b

480 ns FETCH STATE: Instruction is Fetched

488 ns DECODE STATE: Instruction is Decoded

496 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

504 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00

register

AR[00]= 81, DR[01]= 45, GR[10]= 81, PR[11]= 0c

520 ns FETCH STATE: Instruction is Fetched

528 ns DECODE STATE: Instruction is Decoded

536 ns Instruction : RDM, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

544 ns EXECUTE\_1 STATE: Executing RDM...Setting AR = 81 to Memory Address...

552 ns EXECUTE\_2 STATE: Read the Data 55 from Memory Location 81 and Stored

into 10 Register

AR[00]= 81, DR[01]= 45, GR[10]= 55, PR[11]= 0d

568 ns FETCH STATE: Instruction is Fetched

576 ns DECODE STATE: Instruction is Decoded

584 ns Instruction : CPR, 01, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

592 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 01

register

AR[00]= 81, DR[01]= 55, GR[10]= 55, PR[11]= 0e

608 ns FETCH STATE: Instruction is Fetched

616 ns DECODE STATE: Instruction is Decoded

624 ns Instruction : LLS, 1

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

632 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 1

AR[00]= 81, DR[01]= 55, GR[10]= 51, PR[11]= 0f

648 ns FETCH STATE: Instruction is Fetched

656 ns DECODE STATE: Instruction is Decoded

664 ns Instruction : LMS, 9

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

672 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 9

AR[00]= 81, DR[01]= 55, GR[10]= 91, PR[11]= 10

688 ns FETCH STATE: Instruction is Fetched

696 ns DECODE STATE: Instruction is Decoded

704 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

712 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00

register

AR[00]= 91, DR[01]= 55, GR[10]= 91, PR[11]= 11

728 ns FETCH STATE: Instruction is Fetched

736 ns DECODE STATE: Instruction is Decoded

744 ns Instruction : RDM, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

752 ns EXECUTE\_1 STATE: Executing RDM...Setting AR = 91 to Memory Address...

760 ns EXECUTE\_2 STATE: Read the Data 77 from Memory Location 91 and Stored

into 10 Register

AR[00]= 91, DR[01]= 55, GR[10]= 77, PR[11]= 12

776 ns FETCH STATE: Instruction is Fetched

784 ns DECODE STATE: Instruction is Decoded

792 ns Instruction : ADD, 01, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

800 ns EXECUTE\_1 STATE: Executing ADD...Providing ALU Opcodes 55 and 77 and

Adding them...

808 ns EXECUTE\_2 STATE: Added the 01 Register Data 55 with 10 Register Data

77 and the Result is Stored into 01 Register

AR[00]= 91, DR[01]= cc, GR[10]= 77, PR[11]= 13

824 ns FETCH STATE: Instruction is Fetched

832 ns DECODE STATE: Instruction is Decoded

840 ns Instruction : LLS, 1

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

848 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 1

AR[00]= 91, DR[01]= cc, GR[10]= 71, PR[11]= 14

864 ns FETCH STATE: Instruction is Fetched

872 ns DECODE STATE: Instruction is Decoded

880 ns Instruction : LMS, 8

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

888 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 8

AR[00]= 91, DR[01]= cc, GR[10]= 81, PR[11]= 15

904 ns FETCH STATE: Instruction is Fetched

912 ns DECODE STATE: Instruction is Decoded

920 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

928 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00

register

AR[00]= 81, DR[01]= cc, GR[10]= 81, PR[11]= 16

944 ns FETCH STATE: Instruction is Fetched

952 ns DECODE STATE: Instruction is Decoded

960 ns Instruction : WRM, 01;

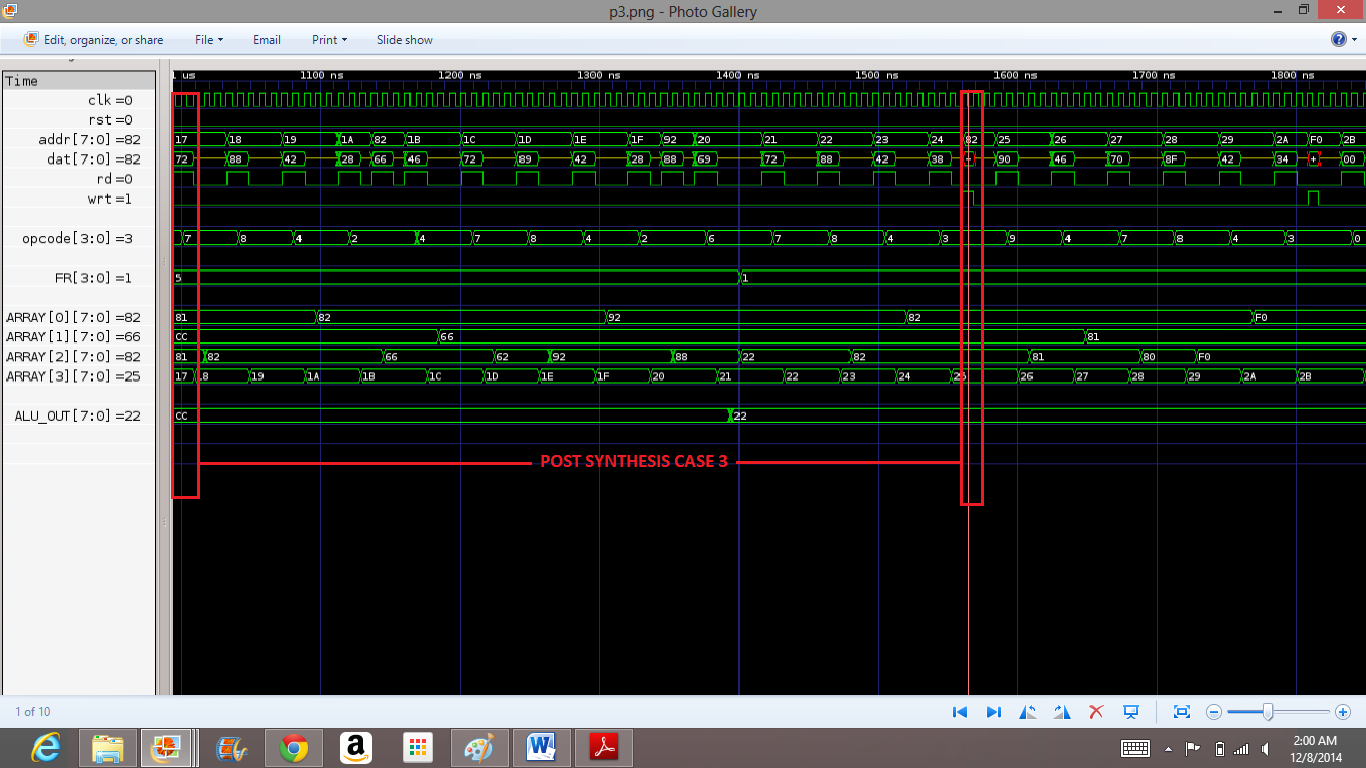
UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

968 ns EXECUTE\_1 STATE: Executing WRM...Setting AR = 81 to Memory Address...

976 ns EXECUTE\_2 STATE: 01 Register Data cc is Written at Memory Address 81

WRITTEN DATA : RAM[80h]= 89, RAM[81h]= cc, RAM[82h]= 66, RAM[FO]= xx

* **Case 3:**

****

992 ns FETCH STATE: Instruction is Fetched

1000 ns DECODE STATE: Instruction is Decoded

1008 ns Instruction : LLS, 2

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1016 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 2

AR[00]= 81, DR[01]= cc, GR[10]= 82, PR[11]= 18

1032 ns FETCH STATE: Instruction is Fetched

1040 ns DECODE STATE: Instruction is Decoded

1048 ns Instruction : LMS, 8

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1056 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 8

AR[00]= 81, DR[01]= cc, GR[10]= 82, PR[11]= 19

1072 ns FETCH STATE: Instruction is Fetched

1080 ns DECODE STATE: Instruction is Decoded

1088 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1096 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00 register

AR[00]= 82, DR[01]= cc, GR[10]= 82, PR[11]= 1a

1112 ns FETCH STATE: Instruction is Fetched

1120 ns DECODE STATE: Instruction is Decoded

1128 ns Instruction : RDM, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1136 ns EXECUTE\_1 STATE: Executing RDM...Setting AR = 82 to Memory Address...

1144 ns EXECUTE\_2 STATE: Read the Data 66 from Memory Location 82 and Stored into 10 Register

AR[00]= 82, DR[01]= cc, GR[10]= 66, PR[11]= 1b

1160 ns FETCH STATE: Instruction is Fetched

1168 ns DECODE STATE: Instruction is Decoded

1176 ns Instruction : CPR, 01, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1184 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 01

register

AR[00]= 82, DR[01]= 66, GR[10]= 66, PR[11]= 1c

1200 ns FETCH STATE: Instruction is Fetched

1208 ns DECODE STATE: Instruction is Decoded

1216 ns Instruction : LLS, 2

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1224 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 2

AR[00]= 82, DR[01]= 66, GR[10]= 62, PR[11]= 1d

1240 ns FETCH STATE: Instruction is Fetched

1248 ns DECODE STATE: Instruction is Decoded

1256 ns Instruction : LMS, 9

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1264 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 9

AR[00]= 82, DR[01]= 66, GR[10]= 92, PR[11]= 1e

1280 ns FETCH STATE: Instruction is Fetched

1288 ns DECODE STATE: Instruction is Decoded

1296 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1304 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00

register

AR[00]= 92, DR[01]= 66, GR[10]= 92, PR[11]= 1f

1320 ns FETCH STATE: Instruction is Fetched

1328 ns DECODE STATE: Instruction is Decoded

1336 ns Instruction : RDM, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1344 ns EXECUTE\_1 STATE: Executing RDM...Setting AR = 92 to Memory Address...

1352 ns EXECUTE\_2 STATE: Read the Data 88 from Memory Location 92 and Stored into 10 Register

AR[00]= 92, DR[01]= 66, GR[10]= 88, PR[11]= 20

1368 ns FETCH STATE: Instruction is Fetched

1376 ns DECODE STATE: Instruction is Decoded

1384 ns Instruction : SUB, 10, 01;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1392 ns EXECUTE\_1 STATE: Executing SUB...Providing ALU Opcodes 88 and 66 and Adding them...

1400 ns EXECUTE\_2 STATE: Subtracted the 01 Register Data 66 from 10 Register

Data 88 and the Result is Stored into 10 Register

AR[00]= 92, DR[01]= 66, GR[10]= 22, PR[11]= 21

1416 ns FETCH STATE: Instruction is Fetched

1424 ns DECODE STATE: Instruction is Decoded

1432 ns Instruction : LLS, 2

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1440 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 2

AR[00]= 92, DR[01]= 66, GR[10]= 22, PR[11]= 22

1456 ns FETCH STATE: Instruction is Fetched

1464 ns DECODE STATE: Instruction is Decoded

1472 ns Instruction : LMS, 8

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1480 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 8

AR[00]= 92, DR[01]= 66, GR[10]= 82, PR[11]= 23

1496 ns FETCH STATE: Instruction is Fetched

1504 ns DECODE STATE: Instruction is Decoded

1512 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1520 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00

register

AR[00]= 82, DR[01]= 66, GR[10]= 82, PR[11]= 24

1536 ns FETCH STATE: Instruction is Fetched

1544 ns DECODE STATE: Instruction is Decoded

1552 ns Instruction : WRM, 10;

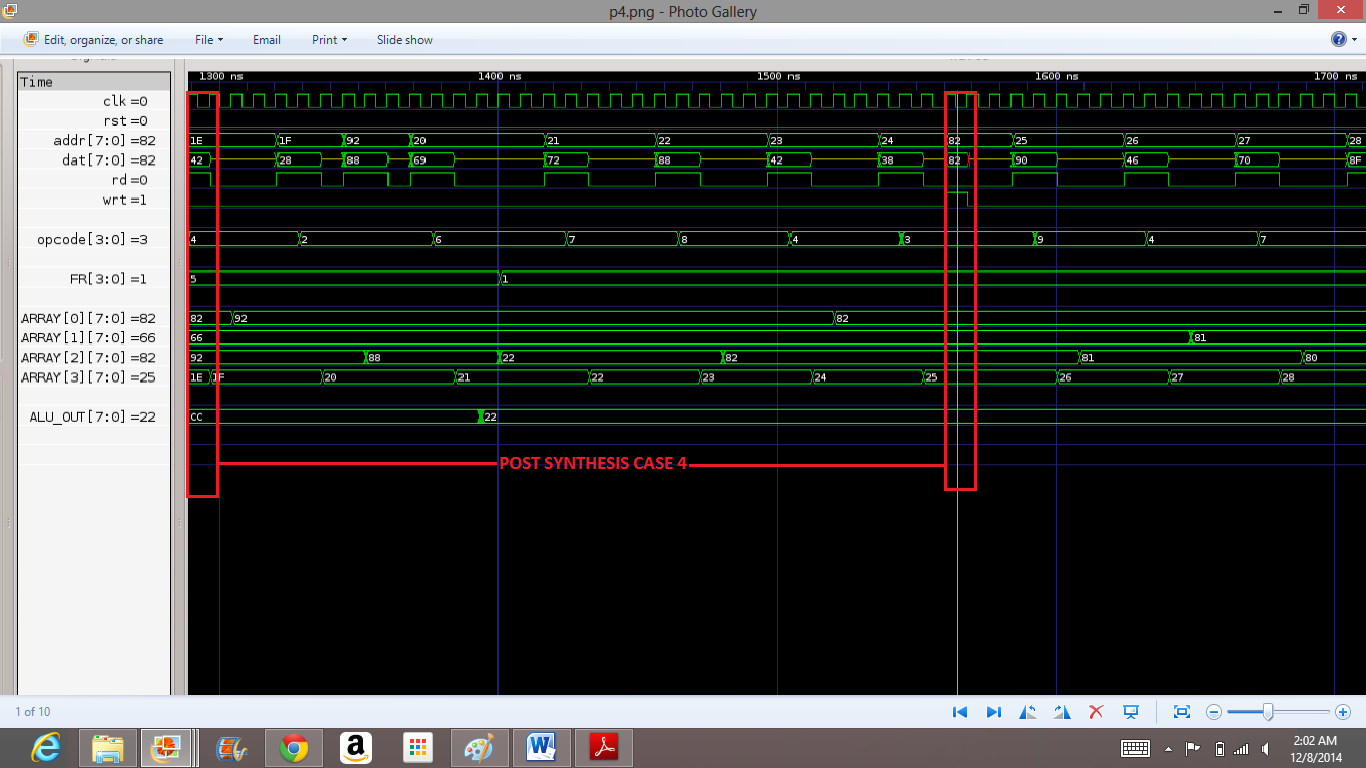
UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1560 ns EXECUTE\_1 STATE: Executing WRM...Setting AR = 82 to Memory Address...

1568 ns EXECUTE\_2 STATE: 10 Register Data 82 is Written at Memory Address 82

WRITTEN DATA : RAM[80h]= 89, RAM[81h]= cc, RAM[82h]= 82, RAM[FO]= xx

* **Case 4:**



1584 ns FETCH STATE: Instruction is Fetched

1592 ns DECODE STATE: Instruction is Decoded

1600 ns Instruction : CFR;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1608 ns EXECUTE\_1 STATE: Executed CFR and Stored data at LSBs of GR = 1

AR[00]= 82, DR[01]= 66, GR[10]= 81, PR[11]= 26

1624 ns FETCH STATE: Instruction is Fetched

1632 ns DECODE STATE: Instruction is Decoded

1640 ns Instruction : CPR, 01, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1648 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 01 register

AR[00]= 82, DR[01]= 81, GR[10]= 81, PR[11]= 27

1664 ns FETCH STATE: Instruction is Fetched

1672 ns DECODE STATE: Instruction is Decoded

1680 ns Instruction : LLS, 0

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1688 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 0

AR[00]= 82, DR[01]= 81, GR[10]= 80, PR[11]= 28

1704 ns FETCH STATE: Instruction is Fetched

1712 ns DECODE STATE: Instruction is Decoded

1720 ns Instruction : LMS, f

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1728 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = f

AR[00]= 82, DR[01]= 81, GR[10]= f0, PR[11]= 29

1744 ns FETCH STATE: Instruction is Fetched

1752 ns DECODE STATE: Instruction is Decoded

1760 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1768 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00 register

AR[00]= f0, DR[01]= 81, GR[10]= f0, PR[11]= 2a

1784 ns FETCH STATE: Instruction is Fetched

1792 ns DECODE STATE: Instruction is Decoded

1800 ns Instruction : WRM, 01;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1808 ns EXECUTE\_1 STATE: Executing WRM...Setting AR = f0 to Memory Address...

1816 ns EXECUTE\_2 STATE: 01 Register Data 81 is Written at Memory Address f0

WRITTEN DATA : RAM[80h]= 89, RAM[81h]= cc, RAM[82h]= 82, RAM[FO]= 81

1832 ns FETCH STATE: Instruction is Fetched

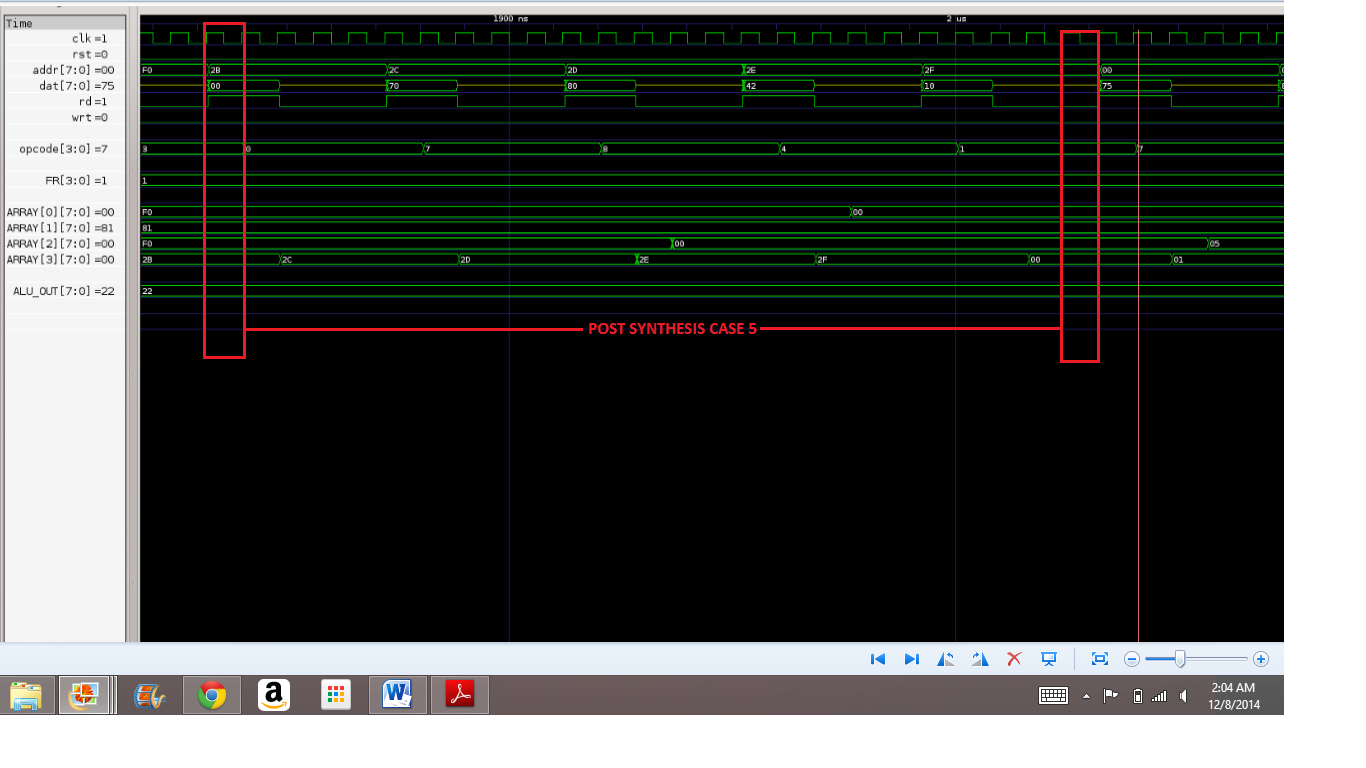
1840 ns DECODE STATE: Instruction is Decoded

1848 ns Instruction : NOP

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1856 ns EXECUTE\_1 STATE: Executed NOP and No Operation is Perfomed

* **Case 5:**

****

1872 ns FETCH STATE: Instruction is Fetched

1880 ns DECODE STATE: Instruction is Decoded

1888 ns Instruction : LLS, 0

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1896 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 0

AR[00]= f0, DR[01]= 81, GR[10]= f0, PR[11]= 2d

1912 ns FETCH STATE: Instruction is Fetched

1920 ns DECODE STATE: Instruction is Decoded

1928 ns Instruction : LMS, 0

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1936 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 0

AR[00]= f0, DR[01]= 81, GR[10]= 00, PR[11]= 2e

1952 ns FETCH STATE: Instruction is Fetched

1960 ns DECODE STATE: Instruction is Decoded

1968 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1976 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00

register

AR[00]= 00, DR[01]= 81, GR[10]= 00, PR[11]= 2f

1992 ns FETCH STATE: Instruction is Fetched

2000 ns DECODE STATE: Instruction is Decoded

2008 ns Instruction : JMP

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2016 ns EXECUTE\_1 STATE: Executed JMP and Jumped to Instruction stored at 00

Memory Location

**VII. Conclusion**

The Scalar Processor was successfully completed with all the functionalities implemented and verified. This project was very helpful to me in understanding the basic concept of Verilog design. I also studied tradeoff between speed, area and power consumption. The area and timing constraints were met to design a highly optimized circuit with low power consumption. Also, the speed of operation of the circuit is satisfactory. However, the pipelining was not implemented in this project but the same can be implemented to make the Scalar Processor much faster. Moreover, in this project, many logical and immediate instructions could be implemented. To encapsulate, the project involved designing the basic microprocessor with memory, and ALU and Control Unit which supports subset of MIPS instructions.

**Appendix A**

**A.1 Contents from EDA Tool Configurations and Setup Files**

**1. .cshrc**

# @(#)cshrc

set filec

set history=100

set prompt="$cwd>[\!] "

limit coredumpsize 0

set path=(~ . /bin /usr/bin /usr/local/bin /usr/kerberos/bin

/usr/sbin /sbin /etc /usr/etc /usr/X11R6/bin /apps/silvaco/bin

/apps/vnc /apps/synopsys /apps/synopsys/installer

/apps/synopsys/CORE/bin /apps/synopsys/SYNTH/bin

/apps/synopsys/SYNTH/linux/bin

/apps/synopsys/SYNTH/linux/syn/bin /apps/synopsys/TCAD/bin

/apps/synopsys/VCSMX\_NEW/bin)

setenv MANPATH "/usr/share/man:"

umask 002

setenv MYPATH $PATH

setenv SYNOPSYS /apps/synopsys

setenv SYNOPSYS\_SIM $SYNOPSYS/VCSMX\_NEW

setenv VCS\_HOME $SYNOPSYS\_SIM

setenv CLS\_CSD\_COMPATIBILITY\_LOCKING NO

setenv SKIP\_CDS\_DIALOG

setenv VCS\_ARCH\_OVERRIDE linux

setenv SNPSLMD $SYNOPSYS/SYNTH

set path=($path $SNPSLMD/linux/bin )

set path=($path $SNPSLMD/linux/syn/bin )

source $SYNOPSYS\_SIM/bin/environ.csh

if ( ! $?DISPLAY ) then

set tty = `tty|sed "s,/dev/,,"`

set who = `who | /bin/grep $tty`

set where = `echo $who | awk '{print $6}'`

set loc = `echo $where | sed "s/[(,),:]//g"`

setenv DISPLAY ${loc}:0.0

unset tty who where loc

if ( -f ${locker\_desc} ) ${cat} ${locker\_desc}

endif

unset base\_dir info\_dir locker\_desc

cat uname

**2. .login**

if ( $?ENVIRONMENT == 0 ) then

umask 22 EE 271 Final Project Report Summer 2014

Page 25 of 254

set history=100

set noclobber ignoreeof

set ignoreeof

setenv VISUAL /bin/vi

setenv EDITOR /bin/ed

stty erase '^H' kill '^U' intr '^C' echoe tostop

endif

echo "Your HOME is $HOME"

**3. .synopsys dc .setup**

designer="EE271 Students"

company="San Jose State University, EE Dept."

search\_path="/apps/synopsys/SYNTH/libraries/syn"

search\_path=search\_path + "/apps/synopsys/CORE/libraries/syn"

search\_path=search\_path + "./src" + "./db"

link\_library={"\*","class.db","and\_or.db","dw\_foundation.sldb"}

target\_library={"class.db","and\_or.db"}

EE271 Final

symbol\_library={"class.sdb","generic.sdb"}

synthetic\_library={"dw\_foundation.sldb","standard.sldb"}

define\_design\_lib WORK -path ./work

alias rc "report\_constraint -all\_violators"

view\_script\_submenu\_items={"Clean Sweep","remove\_design

\_designs"}

edifin\_lib\_in\_port\_symbol = "ipin"

edifin\_lib\_out\_port\_symbol = "opin"

edifin\_lib\_inout\_port\_symbol = "iopin"

edifin\_lib\_in\_osc\_symbol = "iooff"

edifin\_lib\_out\_osc\_symbol = "ooff"

edifin\_lib\_inout\_osc\_symbol = "ioff"

edifin\_lib\_logic\_1\_symbol = "vdd"

edifin\_lib\_logic\_0\_symbol = "gnd"

edifin\_lib\_ripper\_bus = "bus\_end"

edifin\_lib\_route\_grid = 1024

edifin\_lib\_templates =

{{A,landscape,Asize},{A,portrait,Asize.book},{B,landscape,Bsize}

,{C,landscape,Csize},{D,landscape,Dsize},{E,landscape,Esize},{F,

landscape,Fsize}}

edifin\_ground\_net\_name = "gnd!"

edifin\_ground\_net\_property\_name = ""

edifin\_ground\_net\_property\_value = ""

edifout\_ground\_name = "gnd"

edifout\_ground\_net\_name = "gnd!"

edifout\_ground\_net\_property\_name = ""

edifout\_ground\_net\_property\_value = "" EE 271 Final Project Report Summer 2014

Page 26 of 254

edifout\_ground\_pin\_name = "gnd!"

edifin\_power\_net\_name = "vdd!"

edifin\_power\_net\_property\_name = ""

edifin\_power\_net\_property\_value = ""

edifout\_power\_name = "vdd"

edifout\_power\_net\_name = "vdd!"

edifout\_power\_net\_property\_name = ""

edifout\_power\_net\_property\_value = ""

edifout\_power\_pin\_name = "vdd!"

edifout\_power\_and\_ground\_representation = "net"

edifin\_autoconnect\_ports = "true"

single\_group\_per\_sheet = "true"

use\_port\_name\_for\_oscs = "false"

write\_name\_nets\_same\_as\_ports = "true"

edifout\_netlist\_only = "false"

edifout\_target\_system = "cadence"

edifout\_instantiate\_ports = "true"

**A.2 Scripts and/or Commands Used for Simulation and Synthesis**

* **Commands Used for Simulation of project:**

1. vcs +v2k 271control.v scalar\_test.v ; ./simv | tee PRE\_SYNTHESIS.txt = to simulate the file
2. gtkwave scalar.vcd & = to check gtkwave of code.
3. dc\_shell -xg -f synthesis.script | tee SYNTHESIS.txt = to check the different design constrains.
4. vcs +v2k -y /export/apps/toshiba/sjsu/verilog/tc240c +libext+.tsbvlibp scalar\_test.v scalar\_netlist.v ; ./simv | tee postsy.txt; = to synthesis the code with particular library.

* **Script for Scalar Processor:**

set link\_library {/apps/toshiba/sjsu/synopsys/tc240c/tc240c.db\_WCCOM25}

set target\_library {/apps/toshiba/sjsu/synopsys/tc240c/tc240c.db\_WCCOM25}

set symbol\_library {/apps/toshiba/sjsu/synopsys/tc240c/tc240c.workview.sdb}

set synthetic\_library {dw\_foundation.sldb standard.sldb}

set\_min\_library /apps/toshiba/sjsu/synopsys/tc240c/tc240c.db\_WCCOM25 -min\_version /apps/toshiba/sjsu/synopsys/tc240c/tc240c.db\_WCCOM25

read\_verilog 271control.v

current\_design SCALAR\_PROCESSOR

link

check\_design

create\_clock CLK -name CLK -period 3.0000000

set\_propagated\_clock CLK

set\_clock\_uncertainty .25 CLK

set\_max\_delay 1 -from [all\_inputs]

set\_max\_delay 1 -to [all\_outputs]

#set\_max\_delay 2 -to sum

set\_load 160 sum

set\_max\_area 900

compile -map\_effort high

report\_cell

report\_net

update\_timing

report\_timing -max\_paths 10

report\_area 5000

report\_power

write -hierarchy -format verilog -output scalar\_netlist.v

quit

**Appendix B**

**Completed Verilog Source Codes and Testbenches**

* **SCALAR PROCESSOR (CONTROLLER)**

//File Name: 271control.v

`timescale 1ns/10ps

//------------------- DEFINITION OF STATES ---------------------

`define IDEL 3'b000

`define FETCH 3'b001

`define DECODE 3'b010

`define UPDATE 3'b011

`define EXECUTE\_1 3'b100

`define EXECUTE\_2 3'b101

//---------------- DEFINITION OF INSTRUCTIONS ------------------

`define NOP 4'b0000

`define JMP 4'b0001

`define RDM 4'b0010

`define WRM 4'b0011

`define CPR 4'b0100

`define ADD 4'b0101

`define SUB 4'b0110

`define LLS 4'b0111

`define LMS 4'b1000

`define CFR 4'b1001

//---------------SCALAR PROCESSOR VERILOG CODE -----------------

module SCALAR\_PROCESSOR(dat,addr, rd, wrt, clk, rst);

//-------------------- OUTPUT SIGNALS --------------------------

output rd, wrt;

output [7:0] addr;

//-------------------- INPUT SIGNALS ---------------------------

input clk, rst;

//-------------------- INOUT SIGNALS ---------------------------

inout [7:0] dat ;

//---------------------- STATE NETS ----------------------------

reg [2:0] state;

reg [2:0] n\_state;

//---------------- ALU REGISTERS & WIRES -----------------------

reg [7:0] OP1, OP2;

reg OPRN,sign,carry,zero;

wire [7:0] ALU\_OUT;

wire cin,cout,ov;

//---------------SCALAR PROCESSOR INTERNAL REGISTERS -----------

reg [7:0] addr;

reg rd, wrt;

reg [3:0] opcode;

reg [1:0] D,S;

reg [7:0] ARRAY [3:0];

reg [7:0] AR, DR, GR, PR,dat\_T;

reg [3:0] FR;

//--------------------ALU MODULE INSTANTIATE -------------------

ALU alu23 (.sum(ALU\_OUT),.a(OP1),.b(OP2),.cin(OPRN),.cout(cout),.ov(ov));

//--------------------- SCALAR PROCESSOR -----------------------

assign dat = ((rd==1'b0)&&(wrt==1'b1))? dat\_T:{8{1'bz} };

always @(\*)

begin

AR = ARRAY [0];

DR = ARRAY [1];

GR = ARRAY [2];

PR = ARRAY [3];

end

always @ (ALU\_OUT)

begin

if (ALU\_OUT ==0)

zero = 1'b1;

else zero = 1'b0;

if (ALU\_OUT[7] ==1'b1)

sign = 1'b1;

else sign =1'b0;

if ((!OPRN && cout) | (OPRN && !cout))

carry=1'b1;

else carry=1'b0;

end

always @ (posedge (clk) or posedge (rst))

begin

if(rst)

state <= `IDEL;

else state <= n\_state;

end

always @ (\*)

begin

case(state)

`IDEL : n\_state = `FETCH;

`FETCH : n\_state = `DECODE;

`DECODE: n\_state = `UPDATE;

`UPDATE: n\_state = `EXECUTE\_1;

`EXECUTE\_1:

begin

if (opcode == `NOP || opcode==`LLS || opcode==`LMS || opcode==`CPR || opcode==`CFR || opcode==`JMP)

n\_state = `IDEL;

else n\_state = `EXECUTE\_2;

end

`EXECUTE\_2:

begin

n\_state = `IDEL;

end

endcase

end

always @ (posedge clk or posedge rst)

begin

if (rst)

begin

AR <= 0;

GR <= 0;

DR <= 0;

PR <= 0;

FR <= 0;

OP1 <= 0;

OP2 <= 0;

ARRAY[3] <=0;

ARRAY[2] <=0;

ARRAY[1] <=0;

ARRAY[0] <=0;

addr <= 0;

rd <= 0;

wrt <= 0;

dat\_T <=0;

D <= 0;

S <= 0;

opcode <= 0;

OPRN <= 0;

end

else

begin

rd <= 0;

wrt <= 0;

case(n\_state)

//------------------------ FETCH STATE -------------------------

`FETCH:

begin

addr <= ARRAY[3];

rd <= 1'b1;

wrt <= 1'b0;

end

//------------------------ DECODE STATE ------------------------

`DECODE:

begin

rd <=1'b1;

wrt <=1'b0;

{opcode,D,S} <= dat;

end

//------------------------ UPDATE STATE ------------------------

`UPDATE:

begin

rd <=1'b0;

wrt <=1'b0;

case(opcode)

`NOP:

begin

$write($time," NOP\n");

ARRAY [3] <= ARRAY[3]+1;

end

`JMP:

begin

$write($time," JMP \n");

end

`RDM:

begin

$write($time," RDM, %02b;\n", D);

ARRAY [3] <= ARRAY[3]+1;

end

`WRM:

begin

$write($time," WRM, %02b;\n", D);

ARRAY [3] <= ARRAY[3]+1;

end

`CPR:

begin

$write($time, " CPR, %02b, %02b;\n", D ,S);

ARRAY [3] <= ARRAY[3]+1;

end

`ADD:

begin

$write($time," ADD, %02b, %02b;\n",D,S);

ARRAY [3] <= ARRAY[3]+1;

end

`SUB:

begin

$write($time," SUB, %02b, %02b;\n", D ,S);

ARRAY [3] <= ARRAY[3]+1;

end

`LLS:

begin

$write($time," LLS, %02h\n",{D,S});

ARRAY [3] <= ARRAY[3]+1;

end

`LMS:

begin

$write($time," LMS, %02h\n",{D,S});

ARRAY [3] <= ARRAY[3]+1;

end

`CFR:

begin

$write($time," CFR;\n");

ARRAY [3] <= ARRAY[3]+1;

end

default: $write("");

endcase

end

//-------------------- EXECUTE\_1 STATE -------------------------

`EXECUTE\_1:

begin

case(opcode)

`NOP:

begin

$write($time,”Executed NOP and No Operation is Perfomed\n\n");

end

`JMP:

begin

ARRAY[3] <= ARRAY[0];

rd <= 1'b0;

wrt <= 1'b0;

$write($time," Executed JMP and Jumped to Instruction stored at %08H Memory Location\n\n",ARRAY[0]);

end

`RDM:

begin

addr <= ARRAY[0];

rd <= 1'b1;

wrt <= 1'b0;

$write($time," Executing RDM...Setting AR = %08h to Memory Address... \n",ARRAY[0]);

end

`WRM:

begin

addr <= ARRAY[0];

dat\_T <= ARRAY[D];

rd <= 1'b0;

wrt<=1'b1;

$write($time," Executing WRM...Setting AR = %08h to Memory Address... \n",ARRAY[0]);

end

`CPR:

begin

ARRAY[D] <= ARRAY[S];

rd <= 1'b0;

wrt <= 1'b0;

$write($time," Executed CPR and Copied the Contents of %02b to %02b register\n",S,D);

end

`ADD:

begin

OP1 <= ARRAY[D];

OP2 <= ARRAY[S];

OPRN <= 1'b0;

rd <= 1'b0;

wrt <= 1'b0;

$write($time," Executing ADD...Providing ALU Opcodes %08h and %08h and Adding them... \n",ARRAY[D],ARRAY[S]);

end

`SUB:

begin

OP1 <= ARRAY[D];

OP2 <= ARRAY[S];

OPRN <= 1'b1;

rd <= 1'b0;

wrt <= 1'b0;

$write($time," Executing SUB...Providing ALU Opcodes %08h and %08h and Adding them... \n",ARRAY[D],ARRAY[S]);

end

`LLS:

begin

ARRAY[2][3:0] <= {D,S};

rd <= 1'b0;

wrt <= 1'b0;

$write($time," Executed LLS and Stored data at LSBs of GR is %02h\n",{D,S});

end

`LMS:

begin

ARRAY[2][7:4] <= {D,S};

rd <= 1'b0;

wrt <= 1'b0;

$write($time," Executed LMS and Stored data at MSBs of GR is %02h\n",{D,S});

end

`CFR:

begin

ARRAY[2][3:0] <= FR;

rd <= 1'b0;

wrt <= 1'b0;

$write($time," Executed CFR and Stored data at LSBs of GR = %02h\n",FR);

End

default:$write(" INVALID INSTRUCTION ");

endcase

end

//-------------------- EXECUTE\_2 STATE ---------------------

`EXECUTE\_2:

begin

case(opcode)

`JMP:

begin

ARRAY[3] <= ARRAY[0];

rd <= 1'b0;

wrt <= 1'b0;

end

`RDM:

begin

ARRAY[D] <= dat;

rd <= 1'b1;

wrt <= 1'b0;

$write($time," Read the Data %08h from Memory Location %08h and Stored into %02b Register\n",dat,addr,D);

end

`WRM:

begin

$write($time," %02b Register Data %08h is Written at Memory Address %08h\n",D,ARRAY[D],addr);

end

`CPR:

begin

rd <= 1'b0;

wrt <= 1'b0;

end

`ADD:

begin

ARRAY[D] <= ALU\_OUT;

FR <= {zero,sign,carry,ov};

rd <= 1'b0;

wrt <= 1'b0;

$write($time," Added the %02b Register Data %08h with %02b Register Data %08h and the Result is Stored into %02b Register\n",D,ARRAY[D],S,ARRAY[S],D);

end

`SUB:

begin

ARRAY[D] <= ALU\_OUT;

FR <= {zero,sign,carry,ov};

rd <= 1'b0;

wrt <= 1'b0;

$write($time," Subtracted the %02b Register Data %08h from %02b Register Data %08h and the Result is Stored into %02b Register\n",S,ARRAY[S],D,ARRAY[D],D);

end

`LLS:

begin

rd <= 1'b0;

wrt <= 1'b0;

end

`LMS:

begin

rd <= 1'b0;

wrt <= 1'b0;

end

`CFR:

begin

rd <= 1'b0;

wrt <= 1'b0;

end

default: $write("");

endcase

end

//-------------------- IDEL STATE -------------------------

`IDEL:

begin

rd <=1'b0;

wrt <=1'b0;

case(opcode)

`RDM:

begin

$write("AR[00]= %08h, DR[01]= %08h, GR[10]= %08h, PR[11]= %08h\n\n",ARRAY[0],ARRAY[1],ARRAY[2],ARRAY[3]);

end

`CPR:

begin

$write("AR[00]= %08h, DR[01]= %08h, GR[10]= %08h, PR[11]= %08h\n\n",ARRAY[0],ARRAY[1],ARRAY[2],ARRAY[3]);

end

`ADD:

begin

$write("AR[00]= %08h, DR[01]= %08h, GR[10]= %08h, PR[11]= %08h\n\n",ARRAY[0],ARRAY[1],ARRAY[2],ARRAY[3]);

end

`SUB:

begin

$write("AR[00]= %08h, DR[01]= %08h, GR[10]= %08h, PR[11]= %08h\n\n",ARRAY[0],ARRAY[1],ARRAY[2],ARRAY[3]);

end

`LLS:

begin

$write("AR[00]= %08h, DR[01]= %08h, GR[10]= %08h, PR[11]= %08h\n\n",ARRAY[0],ARRAY[1],ARRAY[2],ARRAY[3]);

end

`LMS:

begin

$write("AR[00]= %08h, DR[01]= %08h, GR[10]= %08h, PR[11]= %08h\n\n",ARRAY[0],ARRAY[1],ARRAY[2],ARRAY[3]);

end

`CFR:

begin

$write("AR[00]= %08h, DR[01]= %08h, GR[10]= %08h, PR[11]= %08h\n\n",ARRAY[0],ARRAY[1],ARRAY[2],ARRAY[3]);

end

default: $write("");

endcase

end

endcase

end

end

endmodule

* **Arithmetic and Logic Unit (ALU)**

module ALU(sum,cout,a, b,cin,ov);

input [7:0] a;

input [7:0] b;

input cin;

output [7:0] sum;

output cout,ov;

reg [7:0] d;

wire cin7;

RCA\_8 n1\_inst (sum, cout, a, d, cin, cin7);

assign ov = cin7^cout

always @ (\*)

begin

if(!cin)

begin

d = b;

end

else if (cin)

begin

d = ~b;

end

end

endmodule

* **8 BIT RIPPLE CARRY ADDER**

module RCA\_8 (sum, cout, a, b, cin,cin7);

output [7:0] sum;

output cout,ov;

input [7:0] a, b;

input cin;

wire cin1, cin2, cin3,cin4,cin5,cin6;

add\_full U1 (sum[0], cin1, a[0], b[0], cin);

add\_full U2 (sum[1], cin2, a[1], b[1], cin1);

add\_full U3 (sum[2], cin3, a[2], b[2], cin2);

add\_full U4 (sum[3], cin4, a[3], b[3], cin3);

add\_full U5 (sum[4], cin5, a[4], b[4], cin4);

add\_full U6 (sum[5], cin6, a[5], b[5], cin5);

add\_full U7 (sum[6], cin7, a[6], b[6], cin6);

add\_full U8 (sum[7], cout, a[7], b[7], cin7);

endmodule

* **FULL ADDER**

module add\_full (sum, cout, a, b, cin);

input a, b, cin;

output cout, sum;

wire w1, w2, w3;

add\_half U1 (w1, w2, a, b);

add\_half U2 (sum, w3, cin, w1);

assign cout= w2|w3;

endmodule

* **HALF ADDER**

module add\_half (sum, cout, a, b);

input a, b;

output cout, sum;

assign sum = a^b;

assign cout= a&b;

endmodule

* **Testbench for Scalar Processor**

//File name: Scalar\_test.v

`timescale 1ns/10ps;

`define AR 2'b00

`define DR 2'b01

`define GR 2'b10

`define PR 2'b11

`define NOP 4'b0000

`define JMP 4'b0001

`define RDM 4'b0010

`define WRM 4'b0011

`define CPR 4'b0100

`define ADD 4'b0101

`define SUB 4'b0110

`define LLS 4'b0111

`define LMS 4'b1000

`define CFR 4'b1001

`define IDEL 3'b000

`define FETCH 3'b001

`define DECODE 3'b010

`define UPDATE 3'b011

`define EXECUTE\_1 3'b100

`define EXECUTE\_2 3'b101

module SCALAR\_TEST;

wire [7:0] addr;

wire rd, wrt, clk;

wire [7:0] dat;

// reset

reg rst;

reg [2:0] state, n\_state;

//MEMORY STORAGE

reg [7:0] RAM [0:255];

reg [3:0] opcode;

reg [1:0] D,S;

//Integers for RESET and Display

integer u;

// CLOCK INSTANCE

clk\_GENERATOR clk\_gen\_inst(.clk(clk));

// SCALAR PROCESSOR INSTANCE

SCALAR\_PROCESSORS 123 (.dat(dat,.addr(addr,.rd(rd),.wrt(wrt),.clk(clk),.rst(rst));

initial begin

$dumpfile ("scalar.vcd");

$dumpvars (0,SCALAR\_TEST);

end

initial

begin

rst=1'b1;

#11 rst=1'b0;

//----------------------INSTRUCTION MEMORY ---------------------

// 1) ADD AN 8-BIT DATA AT MEMORY LOCATION 80H TO AN IMMEDIATE DATA 45H AND STORE THE RESULT TO MEMORY LOCATION 80H //

RAM[0] = {`LLS,4'h5};

RAM[1] = {`LMS,4'h4};

RAM[2] = {`CPR,`DR,`GR};

RAM[3] = {`LLS,4'h0};

RAM[4] = {`LMS,4'h8};

RAM[5] = {`CPR,`AR,`GR};

RAM[6] = {`RDM,`GR,2'b00};

RAM[7] = {`ADD,`GR,`DR};

RAM[8] = {`WRM,`GR,2'b00};

// 2) ADD AN 8-BIT DATA AT MEMORY LOCATION 81H AND 91H AND STORE THE RESULT TO MEMORY LOCATION 81H //

RAM[9] = {`LLS,4'h1};

RAM[10] = {`LMS,4'h8};

RAM[11] = {`CPR,`AR,`GR};

RAM[12] = {`RDM,`GR,2'b00};

RAM[13] = {`CPR,`DR,`GR};

RAM[14] = {`LLS,4'h1};

RAM[15] = {`LMS,4'h9};

RAM[16] = {`CPR,`AR,`GR};

RAM[17] = {`RDM,`GR,2'b00};

RAM[18] = {`ADD,`DR,`GR};

RAM[19] = {`LLS,4'h1};

RAM[20] = {`LMS,4'h8};

RAM[21] = {`CPR,`AR,`GR};

RAM[22] = {`WRM,`DR,2'b00};

// 3) SUBSTRACT AN 8-BIT DATA AT MEMORY LOCATION 82H FROM ANOTHER 8-BIT DATA AT MEMORY LOCATION 91H AND STORE THE RESULT TO MEMORY LOCATION 81H //

RAM[23] = {`LLS,4'h2};

RAM[24] = {`LMS,4'h8};

RAM[25] = {`CPR,`AR,`GR};

RAM[26] = {`RDM,`GR,2'b00};

RAM[27] = {`CPR,`DR,`GR};

RAM[28] = {`LLS,4'h2};

RAM[29] = {`LMS,4'h9};

RAM[30] = {`CPR,`AR,`GR};

RAM[31] = {`RDM,`GR,2'b00};

RAM[32] = {`SUB,`GR,`DR};

RAM[33] = {`LLS,4'h2};

RAM[34] = {`LMS,4'h8};

RAM[35] = {`CPR,`AR,`GR};

RAM[36] = {`WRM,`GR,2'b00};

// 4) STORE THE CONTENTS OF 4-BIT FLAG REGISTER INTO LEAST SIGNIFICANT 4-BIT OF MEMORY LOCATION F0H //

RAM[37] = {`CFR,4'h0};

RAM[38] = {`CPR,`DR,`GR};

RAM[39] = {`LLS,4'h0};

RAM[40] = {`LMS,4'hF};

RAM[41] = {`CPR,`AR,`GR};

RAM[42] = {`WRM,`DR,2'b00};

RAM[43] = {`NOP,4'h0};

// 5) USE THE JMP INSTRUCTION TO REPEAT THE SEPS FROM 1) TO 4) ONE MORE TIME //

RAM[44] = {`LLS,4'h0};

RAM[45] = {`LMS,4'h0};

RAM[46] = {`CPR,`AR,`GR};

RAM[47] = {`JMP,4'h0};

//----------------------DATA MEMORY ---------------------------

RAM[8'h80] = 8'h44;

RAM[8'h81] = 8'h55;

RAM[8'h82] = 8'h66;

RAM[8'h91] = 8'h77;

RAM[8'h92] = 8'h88;

for (u=0;u<=255;u=u+1)

begin

$display("Address= %02h, Data at %02h = %02h ",u,u,RAM[u]);

end

$monitor (" WRITEEN DATA : RAM[80h]= %h, RAM[81h]= %h, RAM[82h]= %h, RAM[FO]= %h\n",RAM[8'h80],RAM[8'h81],RAM[8'h82],RAM[8'hf0]);

#3874 $finish;

end

//------------------------ MEMORY BANK -------------------------

always @ (posedge (clk) or posedge (rst))

begin

if(rst)

state <= `IDEL;

else state <= n\_state;

end

always @ (\*)

begin

case(state)

`IDEL : n\_state = `FETCH;

`FETCH : n\_state = `DECODE;

`DECODE: n\_state = `UPDATE;

`UPDATE: n\_state = `EXECUTE\_1;

`EXECUTE\_1:begin

if (opcode == `NOP || opcode==`LLS || opcode==`LMS || opcode==`CPR || opcode==`CFR || opcode==`JMP)

n\_state = `IDEL;

else n\_state = `EXECUTE\_2;

end

`EXECUTE\_2:begin

n\_state = `IDEL;

end

endcase

end

always @ (posedge clk )

begin

case(n\_state)

//---------------------------- FETCH STATE ---------------------

`FETCH:

begin

$write($time," ns FETCH STATE: Instruction is Fetched\n");

end

//-----------------------------DECODE STATE -------------------- `DECODE:

begin

$write($time," ns DECODE STATE: Instruction is Decoded\n");

{opcode,D,S} <= dat;

end

//------------------------- UPDATE STATE -----------------------

`UPDATE:

begin

case(opcode)

`NOP:

begin

$write($time," ns Instruction : NOP\n");

$write("UPDATE STATE: PR Register is Incremented to Fetch the next Instruction. . .\n");

end

`JMP:

begin

$write($time," ns Instruction : JMP \n");

$write("UPDATE STATE: PR Register is Incremented to Fetch the next Instruction. . .\n");

end

`RDM:

begin

$write($time," ns Instruction : RDM,%02h ;\n",D);

$write("UPDATE STATE: PR Register is Incremented to Fetch the next Instruction. . .\n");

end

`WRM:

begin

$write($time," ns Instruction : WRM,%02h ;\n",S);

$write("UPDATE STATE: PR Register is Incremented to Fetch the next Instruction. . .\n");

end

`CPR:

begin

$write($time, " ns Instruction : CPR,%02h,%02h ;\n ",D,S);

$write(" UPDATE STATE: PR Register is Incremented to Fetch the next Instruction. . .\n");

end

`ADD:

begin

$write($time," ns Instruction : ADD,%02h,%02h ;\n ",D,S);

$write("UPDATE STATE: PR Register is Incremented to Fetch the next Instruction. . .\n");

end

`SUB:

begin

$write($time," ns Instruction : SUB,%02h,%02h ;\n ",D,S);

$write("UPDATE STATE: PR Register is Incremented to Fetch the next Instruction. . .\n");

end

`LLS:

begin

$write($time," ns Instruction : LLS,%02h ;\n ",dat[3:0]);

$write("UPDATE STATE: PR Register is Incremented to Fetch the next Instruction. . .\n");

end

`LMS:

begin

$write($time," ns Instruction : LMS,%02h ;\n",dat[3:0]);

$write("UPDATE STATE: PR Register is Incremented to Fetch the next Instruction. . .\n");

end

`CFR:

begin

$write($time," ns Instruction : CFR;\n");

$write("UPDATE STATE: PR Register is Incremented to Fetch the next Instruction. . .\n");

//ARRAY [3] <= ARRAY[3]+1;

end

default: $write("");

endcase

end

//------------------------- EXECUTE\_1 STATE --------------------

`EXECUTE\_1:

begin

case(opcode)

`NOP:

begin

$write($time," ns EXECUTE\_1 STATE: Executed NOP and No Operation is Perfomed\n\n");

end

`JMP:

begin

$write($time," ns EXECUTE\_1 STATE: Executed JMP and Jumped to Instruction stored at Memory Location 00H\n\n");

end

`RDM:

begin

$write($time," ns EXECUTE\_1 STATE: Executing RDM...Setting AR to Memory Address... \n");

end

`WRM:

begin

$write($time," ns EXECUTE\_1 STATE: Executing WRM...Setting AR to Memory Address... \n");

end

`CPR:

begin

$write($time," ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of register %02h to %02h ;\n ",S,D);

end

`ADD:

begin

$write($time," ns EXECUTE\_1 STATE: Executing ADD...Providing ALU inputs %02h and %02h Register Contents and Adding them...\n",S,D);

end

`SUB:

begin

$write($time," ns EXECUTE\_1 STATE: Executing SUB...Providing ALU inputs %02h and %02h Register Contents and Subtracting them...\n",S,D);

end

`LLS:

begin

$write($time," ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR \n");

end

`LMS:

begin

$write($time," ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR \n");

end

`CFR:

begin

$write($time," ns EXECUTE\_1 STATE: Executed CFR and Stored data FR to LSBs of GR \n");

end

default:$write($time," ns EXECUTE\_1 STATE: INVALID INSTRUCTION ");

endcase

end

`EXECUTE\_2:

begin

case(opcode)

`RDM:

begin

$write($time," ns EXECUTE\_2 STATE: Read the Data %08h from Memory Location %08h and Stored into %02b Register\n",dat,addr,D);

end

`WRM:

begin

$write($time," ns EXECUTE\_2 STATE: %02b Register Data is Written at Memory Address %08h\n",D,addr);

end

`ADD:

begin

$write($time," ns EXECUTE\_2 STATE: Added the %02b Register Data with %02b Register Data and the Result is Stored into %02b Register\n",D,S,D);

end

`SUB:

begin

$write($time," ns EXECUTE\_2 STATE: Subtracted the %02b Register Data from %02b Register Data and the Result is Stored into %02b Register\n",S,D,D);

end

endcase

end

`IDEL:

begin

end

endcase

end

assign dat = ((rd==1'b1)&&(wrt==1'b0))? RAM [addr]:{8{1'bz} };

always @ (posedge clk)

begin

if ((rd==1'b0)&&(wrt==1'b1)) // write operation

RAM [addr] <= dat;

end

endmodule

//-----------------------CLOCK GENERATOR MODULE ----------------

module clk\_GENERATOR(clk);

output clk;

reg clk;

initial

begin

clk = 1'b1;

end

always

begin

#4 clk <= ~clk;

end

endmodule

**Appendix C**

**Reports and Circuits from EDA Tools**

**C.1 Reports (contents) from RTL (Pre-synthesis) Simulations (VCS or NCVERILOG)**

Chronologic VCS simulator copyright 1991-2014

Contains Synopsys proprietary information.

Compiler version I-2014.03-2; Runtime version I-2014.03-2; Dec 5 03:56 2014

Address= 00, Data at 00 = 75

Address= 01, Data at 01 = 84

Address= 02, Data at 02 = 46

Address= 03, Data at 03 = 70

Address= 04, Data at 04 = 88

Address= 05, Data at 05 = 42

Address= 06, Data at 06 = 28

Address= 07, Data at 07 = 59

Address= 08, Data at 08 = 38

Address= 09, Data at 09 = 71

Address= 0a, Data at 0a = 88

Address= 0b, Data at 0b = 42

Address= 0c, Data at 0c = 28

Address= 0d, Data at 0d = 46

Address= 0e, Data at 0e = 71

Address= 0f, Data at 0f = 89

Address= 10, Data at 10 = 42

Address= 11, Data at 11 = 28

Address= 12, Data at 12 = 56

Address= 13, Data at 13 = 71

Address= 14, Data at 14 = 88

Address= 15, Data at 15 = 42

Address= 16, Data at 16 = 34

Address= 17, Data at 17 = 72

Address= 18, Data at 18 = 88

Address= 19, Data at 19 = 42

Address= 1a, Data at 1a = 28

Address= 1b, Data at 1b = 46

Address= 1c, Data at 1c = 72

Address= 1d, Data at 1d = 89

Address= 1e, Data at 1e = 42

Address= 1f, Data at 1f = 28

Address= 20, Data at 20 = 69

Address= 21, Data at 21 = 72

Address= 22, Data at 22 = 88

Address= 23, Data at 23 = 42

Address= 24, Data at 24 = 38

Address= 25, Data at 25 = 90

Address= 26, Data at 26 = 46

Address= 27, Data at 27 = 70

Address= 28, Data at 28 = 8f

Address= 29, Data at 29 = 42

Address= 2a, Data at 2a = 34

Address= 2b, Data at 2b = 00

Address= 2c, Data at 2c = 70

Address= 2d, Data at 2d = 80

Address= 2e, Data at 2e = 42

Address= 2f, Data at 2f = 10

Address= 30, Data at 30 = xx

Address= 31, Data at 31 = xx

Address= 32, Data at 32 = xx

Address= 33, Data at 33 = xx

Address= 34, Data at 34 = xx

Address= 35, Data at 35 = xx

Address= 36, Data at 36 = xx

Address= 37, Data at 37 = xx

Address= 38, Data at 38 = xx

Address= 39, Data at 39 = xx

Address= 3a, Data at 3a = xx

Address= 3b, Data at 3b = xx

Address= 3c, Data at 3c = xx

Address= 3d, Data at 3d = xx

Address= 3e, Data at 3e = xx

Address= 3f, Data at 3f = xx

Address= 40, Data at 40 = xx

Address= 41, Data at 41 = xx

Address= 42, Data at 42 = xx

Address= 43, Data at 43 = xx

Address= 44, Data at 44 = xx

Address= 45, Data at 45 = xx

Address= 46, Data at 46 = xx

Address= 47, Data at 47 = xx

Address= 48, Data at 48 = xx

Address= 49, Data at 49 = xx

Address= 4a, Data at 4a = xx

Address= 4b, Data at 4b = xx

Address= 4c, Data at 4c = xx

Address= 4d, Data at 4d = xx

Address= 4e, Data at 4e = xx

Address= 4f, Data at 4f = xx

Address= 50, Data at 50 = xx

Address= 51, Data at 51 = xx

Address= 52, Data at 52 = xx

Address= 53, Data at 53 = xx

Address= 54, Data at 54 = xx

Address= 55, Data at 55 = xx

Address= 56, Data at 56 = xx

Address= 57, Data at 57 = xx

Address= 58, Data at 58 = xx

Address= 59, Data at 59 = xx

Address= 5a, Data at 5a = xx

Address= 5b, Data at 5b = xx

Address= 5c, Data at 5c = xx

Address= 5d, Data at 5d = xx

Address= 5e, Data at 5e = xx

Address= 5f, Data at 5f = xx

Address= 60, Data at 60 = xx

Address= 61, Data at 61 = xx

Address= 62, Data at 62 = xx

Address= 63, Data at 63 = xx

Address= 64, Data at 64 = xx

Address= 65, Data at 65 = xx

Address= 66, Data at 66 = xx

Address= 67, Data at 67 = xx

Address= 68, Data at 68 = xx

Address= 69, Data at 69 = xx

Address= 6a, Data at 6a = xx

Address= 6b, Data at 6b = xx

Address= 6c, Data at 6c = xx

Address= 6d, Data at 6d = xx

Address= 6e, Data at 6e = xx

Address= 6f, Data at 6f = xx

Address= 70, Data at 70 = xx

Address= 71, Data at 71 = xx

Address= 72, Data at 72 = xx

Address= 73, Data at 73 = xx

Address= 74, Data at 74 = xx

Address= 75, Data at 75 = xx

Address= 76, Data at 76 = xx

Address= 77, Data at 77 = xx

Address= 78, Data at 78 = xx

Address= 79, Data at 79 = xx

Address= 7a, Data at 7a = xx

Address= 7b, Data at 7b = xx

Address= 7c, Data at 7c = xx

Address= 7d, Data at 7d = xx

Address= 7e, Data at 7e = xx

Address= 7f, Data at 7f = xx

Address= 80, Data at 80 = 44

Address= 81, Data at 81 = 55

Address= 82, Data at 82 = 66

Address= 83, Data at 83 = xx

Address= 84, Data at 84 = xx

Address= 85, Data at 85 = xx

Address= 86, Data at 86 = xx

Address= 87, Data at 87 = xx

Address= 88, Data at 88 = xx

Address= 89, Data at 89 = xx

Address= 8a, Data at 8a = xx

Address= 8b, Data at 8b = xx

Address= 8c, Data at 8c = xx

Address= 8d, Data at 8d = xx

Address= 8e, Data at 8e = xx

Address= 8f, Data at 8f = xx

Address= 90, Data at 90 = xx

Address= 91, Data at 91 = 77

Address= 92, Data at 92 = 88

Address= 93, Data at 93 = xx

Address= 94, Data at 94 = xx

Address= 95, Data at 95 = xx

Address= 96, Data at 96 = xx

Address= 97, Data at 97 = xx

Address= 98, Data at 98 = xx

Address= 99, Data at 99 = xx

Address= 9a, Data at 9a = xx

Address= 9b, Data at 9b = xx

Address= 9c, Data at 9c = xx

Address= 9d, Data at 9d = xx

Address= 9e, Data at 9e = xx

Address= 9f, Data at 9f = xx

Address= a0, Data at a0 = xx

Address= a1, Data at a1 = xx

Address= a2, Data at a2 = xx

Address= a3, Data at a3 = xx

Address= a4, Data at a4 = xx

Address= a5, Data at a5 = xx

Address= a6, Data at a6 = xx

Address= a7, Data at a7 = xx

Address= a8, Data at a8 = xx

Address= a9, Data at a9 = xx

Address= aa, Data at aa = xx

Address= ab, Data at ab = xx

Address= ac, Data at ac = xx

Address= ad, Data at ad = xx

Address= ae, Data at ae = xx

Address= af, Data at af = xx

Address= b0, Data at b0 = xx

Address= b1, Data at b1 = xx

Address= b2, Data at b2 = xx

Address= b3, Data at b3 = xx

Address= b4, Data at b4 = xx

Address= b5, Data at b5 = xx

Address= b6, Data at b6 = xx

Address= b7, Data at b7 = xx

Address= b8, Data at b8 = xx

Address= b9, Data at b9 = xx

Address= ba, Data at ba = xx

Address= bb, Data at bb = xx

Address= bc, Data at bc = xx

Address= bd, Data at bd = xx

Address= be, Data at be = xx

Address= bf, Data at bf = xx

Address= c0, Data at c0 = xx

Address= c1, Data at c1 = xx

Address= c2, Data at c2 = xx

Address= c3, Data at c3 = xx

Address= c4, Data at c4 = xx

Address= c5, Data at c5 = xx

Address= c6, Data at c6 = xx

Address= c7, Data at c7 = xx

Address= c8, Data at c8 = xx

Address= c9, Data at c9 = xx

Address= ca, Data at ca = xx

Address= cb, Data at cb = xx

Address= cc, Data at cc = xx

Address= cd, Data at cd = xx

Address= ce, Data at ce = xx

Address= cf, Data at cf = xx

Address= d0, Data at d0 = xx

Address= d1, Data at d1 = xx

Address= d2, Data at d2 = xx

Address= d3, Data at d3 = xx

Address= d4, Data at d4 = xx

Address= d5, Data at d5 = xx

Address= d6, Data at d6 = xx

Address= d7, Data at d7 = xx

Address= d8, Data at d8 = xx

Address= d9, Data at d9 = xx

Address= da, Data at da = xx

Address= db, Data at db = xx

Address= dc, Data at dc = xx

Address= dd, Data at dd = xx

Address= de, Data at de = xx

Address= df, Data at df = xx

Address= e0, Data at e0 = xx

Address= e1, Data at e1 = xx

Address= e2, Data at e2 = xx

Address= e3, Data at e3 = xx

Address= e4, Data at e4 = xx

Address= e5, Data at e5 = xx

Address= e6, Data at e6 = xx

Address= e7, Data at e7 = xx

Address= e8, Data at e8 = xx

Address= e9, Data at e9 = xx

Address= ea, fData at ea = xx

Address= eb, Data at eb = xx

Address= ec, Data at ec = xx

Address= ed, Data at ed = xx

Address= ee, Data at ee = xx

Address= ef, Data at ef = xx

Address= f0, Data at f0 = xx

Address= f1, Data at f1 = xx

Address= f2, Data at f2 = xx

Address= f3, Data at f3 = xx

Address= f4, Data at f4 = xx

Address= f5, Data at f5 = xx

Address= f6, Data at f6 = xx

Address= f7, Data at f7 = xx

Address= f8, Data at f8 = xx

Address= f9, Data at f9 = xx

Address= fa, Data at fa = xx

Address= fb, Data at fb = xx

Address= fc, Data at fc = xx

Address= fd, Data at fd = xx

Address= fe, Data at fe = xx

Address= ff, Data at ff = xx

WRITEEN DATA : RAM[80h]= 44, RAM[81h]= 55, RAM[82h]= 66, RAM[FO]= xx

* **Case 1:**

16 ns FETCH STATE: Instruction is Fetched

24 ns DECODE STATE: Instruction is Decoded

32 ns Instruction : LLS, 5

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

40 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 5

AR[00]= 00, DR[01]= 00, GR[10]= 05, PR[11]= 01

56 ns FETCH STATE: Instruction is Fetched

64 ns DECODE STATE: Instruction is Decoded

72 ns Instruction : LMS, 4

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

80 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 4

AR[00]= 00, DR[01]= 00, GR[10]= 45, PR[11]= 02

96 ns FETCH STATE: Instruction is Fetched

104 ns DECODE STATE: Instruction is Decoded

112 ns Instruction : CPR, 01, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

120 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 01 reg.

AR[00]= 00, DR[01]= 45, GR[10]= 45, PR[11]= 03

136 ns FETCH STATE: Instruction is Fetched

144 ns DECODE STATE: Instruction is Decoded

152 ns Instruction : LLS, 0

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

160 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 0

AR[00]= 00, DR[01]= 45, GR[10]= 40, PR[11]= 04

176 ns FETCH STATE: Instruction is Fetched

184 ns DECODE STATE: Instruction is Decoded

192 ns Instruction : LMS, 8

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

200 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 8

AR[00]= 00, DR[01]= 45, GR[10]= 80, PR[11]= 05

216 ns FETCH STATE: Instruction is Fetched

224 ns DECODE STATE: Instruction is Decoded

232 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

240 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00 reg.

AR[00]= 80, DR[01]= 45, GR[10]= 80, PR[11]= 06

256 ns FETCH STATE: Instruction is Fetched

264 ns DECODE STATE: Instruction is Decoded

272 ns Instruction : RDM, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

280 ns EXECUTE\_1 STATE: Executing RDM...Setting AR = 80 to Memory Address...

288 ns EXECUTE\_2 STATE: Read the Data 44 from Memory Location 80 and Stored

into 10 Register

AR[00]= 80, DR[01]= 45, GR[10]= 44, PR[11]= 07

304 ns FETCH STATE: Instruction is Fetched

312 ns DECODE STATE: Instruction is Decoded

320 ns Instruction : ADD, 10, 01;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

328 ns EXECUTE\_1 STATE: Executing ADD...Providing ALU Opcodes 44 and 45 and

Adding them...

336 ns EXECUTE\_2 STATE: Added the 10 Register Data 44 with 01 Register Data

45 and the Result is Stored into 10 Register

AR[00]= 80, DR[01]= 45, GR[10]= 89, PR[11]= 08

352 ns FETCH STATE: Instruction is Fetched

360 ns DECODE STATE: Instruction is Decoded

368 ns Instruction : WRM, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

376 ns EXECUTE\_1 STATE: Executing WRM...Setting AR = 80 to Memory Address...

384 ns EXECUTE\_2 STATE: 10 Register Data 89 is Written at Memory Address 80

WRITEEN DATA : RAM[80h]= 89, RAM[81h]= 55, RAM[82h]= 66, RAM[FO]= xx

* **Case 2:**

400 ns FETCH STATE: Instruction is Fetched

408 ns DECODE STATE: Instruction is Decoded

416 ns Instruction : LLS, 1

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

424 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 1

AR[00]= 80, DR[01]= 45, GR[10]= 81, PR[11]= 0a

440 ns FETCH STATE: Instruction is Fetched

448 ns DECODE STATE: Instruction is Decoded

456 ns Instruction : LMS, 8

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

464 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 8

AR[00]= 80, DR[01]= 45, GR[10]= 81, PR[11]= 0b

480 ns FETCH STATE: Instruction is Fetched

488 ns DECODE STATE: Instruction is Decoded

496 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

504 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00

register

AR[00]= 81, DR[01]= 45, GR[10]= 81, PR[11]= 0c

520 ns FETCH STATE: Instruction is Fetched

528 ns DECODE STATE: Instruction is Decoded

536 ns Instruction : RDM, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

544 ns EXECUTE\_1 STATE: Executing RDM...Setting AR = 81 to Memory Address...

552 ns EXECUTE\_2 STATE: Read the Data 55 from Memory Location 81 and Stored

into 10 Register

AR[00]= 81, DR[01]= 45, GR[10]= 55, PR[11]= 0d

568 ns FETCH STATE: Instruction is Fetched

576 ns DECODE STATE: Instruction is Decoded

584 ns Instruction : CPR, 01, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

592 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 01

register

AR[00]= 81, DR[01]= 55, GR[10]= 55, PR[11]= 0e

608 ns FETCH STATE: Instruction is Fetched

616 ns DECODE STATE: Instruction is Decoded

624 ns Instruction : LLS, 1

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

632 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 1

AR[00]= 81, DR[01]= 55, GR[10]= 51, PR[11]= 0f

648 ns FETCH STATE: Instruction is Fetched

656 ns DECODE STATE: Instruction is Decoded

664 ns Instruction : LMS, 9

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

672 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 9

AR[00]= 81, DR[01]= 55, GR[10]= 91, PR[11]= 10

688 ns FETCH STATE: Instruction is Fetched

696 ns DECODE STATE: Instruction is Decoded

704 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

712 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00

register

AR[00]= 91, DR[01]= 55, GR[10]= 91, PR[11]= 11

728 ns FETCH STATE: Instruction is Fetched

736 ns DECODE STATE: Instruction is Decoded

744 ns Instruction : RDM, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

752 ns EXECUTE\_1 STATE: Executing RDM...Setting AR = 91 to Memory Address...

760 ns EXECUTE\_2 STATE: Read the Data 77 from Memory Location 91 and Stored

into 10 Register

AR[00]= 91, DR[01]= 55, GR[10]= 77, PR[11]= 12

776 ns FETCH STATE: Instruction is Fetched

784 ns DECODE STATE: Instruction is Decoded

792 ns Instruction : ADD, 01, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

800 ns EXECUTE\_1 STATE: Executing ADD...Providing ALU Opcodes 55 and 77 and

Adding them...

808 ns EXECUTE\_2 STATE: Added the 01 Register Data 55 with 10 Register Data

77 and the Result is Stored into 01 Register

AR[00]= 91, DR[01]= cc, GR[10]= 77, PR[11]= 13

824 ns FETCH STATE: Instruction is Fetched

832 ns DECODE STATE: Instruction is Decoded

840 ns Instruction : LLS, 1

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

848 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 1

AR[00]= 91, DR[01]= cc, GR[10]= 71, PR[11]= 14

864 ns FETCH STATE: Instruction is Fetched

872 ns DECODE STATE: Instruction is Decoded

880 ns Instruction : LMS, 8

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

888 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 8

AR[00]= 91, DR[01]= cc, GR[10]= 81, PR[11]= 15

904 ns FETCH STATE: Instruction is Fetched

912 ns DECODE STATE: Instruction is Decoded

920 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

928 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00

register

AR[00]= 81, DR[01]= cc, GR[10]= 81, PR[11]= 16

944 ns FETCH STATE: Instruction is Fetched

952 ns DECODE STATE: Instruction is Decoded

960 ns Instruction : WRM, 01;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

968 ns EXECUTE\_1 STATE: Executing WRM...Setting AR = 81 to Memory Address...

976 ns EXECUTE\_2 STATE: 01 Register Data cc is Written at Memory Address 81

WRITTEN DATA : RAM[80h]= 89, RAM[81h]= cc, RAM[82h]= 66, RAM[FO]= xx

* **Case 3:**

992 ns FETCH STATE: Instruction is Fetched

1000 ns DECODE STATE: Instruction is Decoded

1008 ns Instruction : LLS, 2

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1016 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 2

AR[00]= 81, DR[01]= cc, GR[10]= 82, PR[11]= 18

1032 ns FETCH STATE: Instruction is Fetched

1040 ns DECODE STATE: Instruction is Decoded

1048 ns Instruction : LMS, 8

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1056 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 8

AR[00]= 81, DR[01]= cc, GR[10]= 82, PR[11]= 19

1072 ns FETCH STATE: Instruction is Fetched

1080 ns DECODE STATE: Instruction is Decoded

1088 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1096 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00 register

AR[00]= 82, DR[01]= cc, GR[10]= 82, PR[11]= 1a

1112 ns FETCH STATE: Instruction is Fetched

1120 ns DECODE STATE: Instruction is Decoded

1128 ns Instruction : RDM, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1136 ns EXECUTE\_1 STATE: Executing RDM...Setting AR = 82 to Memory Address...

1144 ns EXECUTE\_2 STATE: Read the Data 66 from Memory Location 82 and Stored into 10 Register

AR[00]= 82, DR[01]= cc, GR[10]= 66, PR[11]= 1b

1160 ns FETCH STATE: Instruction is Fetched

1168 ns DECODE STATE: Instruction is Decoded

1176 ns Instruction : CPR, 01, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1184 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 01

register

AR[00]= 82, DR[01]= 66, GR[10]= 66, PR[11]= 1c

1200 ns FETCH STATE: Instruction is Fetched

1208 ns DECODE STATE: Instruction is Decoded

1216 ns Instruction : LLS, 2

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1224 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 2

AR[00]= 82, DR[01]= 66, GR[10]= 62, PR[11]= 1d

1240 ns FETCH STATE: Instruction is Fetched

1248 ns DECODE STATE: Instruction is Decoded

1256 ns Instruction : LMS, 9

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1264 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 9

AR[00]= 82, DR[01]= 66, GR[10]= 92, PR[11]= 1e

1280 ns FETCH STATE: Instruction is Fetched

1288 ns DECODE STATE: Instruction is Decoded

1296 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1304 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00

register

AR[00]= 92, DR[01]= 66, GR[10]= 92, PR[11]= 1f

1320 ns FETCH STATE: Instruction is Fetched

1328 ns DECODE STATE: Instruction is Decoded

1336 ns Instruction : RDM, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1344 ns EXECUTE\_1 STATE: Executing RDM...Setting AR = 92 to Memory Address...

1352 ns EXECUTE\_2 STATE: Read the Data 88 from Memory Location 92 and Stored into 10 Register

AR[00]= 92, DR[01]= 66, GR[10]= 88, PR[11]= 20

1368 ns FETCH STATE: Instruction is Fetched

1376 ns DECODE STATE: Instruction is Decoded

1384 ns Instruction : SUB, 10, 01;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1392 ns EXECUTE\_1 STATE: Executing SUB...Providing ALU Opcodes 88 and 66 and Adding them...

1400 ns EXECUTE\_2 STATE: Subtracted the 01 Register Data 66 from 10 Register

Data 88 and the Result is Stored into 10 Register

AR[00]= 92, DR[01]= 66, GR[10]= 22, PR[11]= 21

1416 ns FETCH STATE: Instruction is Fetched

1424 ns DECODE STATE: Instruction is Decoded

1432 ns Instruction : LLS, 2

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1440 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 2

AR[00]= 92, DR[01]= 66, GR[10]= 22, PR[11]= 22

1456 ns FETCH STATE: Instruction is Fetched

1464 ns DECODE STATE: Instruction is Decoded

1472 ns Instruction : LMS, 8

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1480 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 8

AR[00]= 92, DR[01]= 66, GR[10]= 82, PR[11]= 23

1496 ns FETCH STATE: Instruction is Fetched

1504 ns DECODE STATE: Instruction is Decoded

1512 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1520 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00

register

AR[00]= 82, DR[01]= 66, GR[10]= 82, PR[11]= 24

1536 ns FETCH STATE: Instruction is Fetched

1544 ns DECODE STATE: Instruction is Decoded

1552 ns Instruction : WRM, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1560 ns EXECUTE\_1 STATE: Executing WRM...Setting AR = 82 to Memory Address...

1568 ns EXECUTE\_2 STATE: 10 Register Data 82 is Written at Memory Address 82

WRITTEN DATA : RAM[80h]= 89, RAM[81h]= cc, RAM[82h]= 82, RAM[FO]= xx

* **Case 4:**

1584 ns FETCH STATE: Instruction is Fetched

1592 ns DECODE STATE: Instruction is Decoded

1600 ns Instruction : CFR;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1608 ns EXECUTE\_1 STATE: Executed CFR and Stored data at LSBs of GR = 1

AR[00]= 82, DR[01]= 66, GR[10]= 81, PR[11]= 26

1624 ns FETCH STATE: Instruction is Fetched

1632 ns DECODE STATE: Instruction is Decoded

1640 ns Instruction : CPR, 01, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1648 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 01 register

AR[00]= 82, DR[01]= 81, GR[10]= 81, PR[11]= 27

1664 ns FETCH STATE: Instruction is Fetched

1672 ns DECODE STATE: Instruction is Decoded

1680 ns Instruction : LLS, 0

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1688 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 0

AR[00]= 82, DR[01]= 81, GR[10]= 80, PR[11]= 28

1704 ns FETCH STATE: Instruction is Fetched

1712 ns DECODE STATE: Instruction is Decoded

1720 ns Instruction : LMS, f

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1728 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = f

AR[00]= 82, DR[01]= 81, GR[10]= f0, PR[11]= 29

1744 ns FETCH STATE: Instruction is Fetched

1752 ns DECODE STATE: Instruction is Decoded

1760 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1768 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00 register

AR[00]= f0, DR[01]= 81, GR[10]= f0, PR[11]= 2a

1784 ns FETCH STATE: Instruction is Fetched

1792 ns DECODE STATE: Instruction is Decoded

1800 ns Instruction : WRM, 01;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1808 ns EXECUTE\_1 STATE: Executing WRM...Setting AR = f0 to Memory Address...

1816 ns EXECUTE\_2 STATE: 01 Register Data 81 is Written at Memory Address f0

WRITTEN DATA : RAM[80h]= 89, RAM[81h]= cc, RAM[82h]= 82, RAM[FO]= 81

1832 ns FETCH STATE: Instruction is Fetched

1840 ns DECODE STATE: Instruction is Decoded

1848 ns Instruction : NOP

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1856 ns EXECUTE\_1 STATE: Executed NOP and No Operation is Perfomed

* **Case 5:**

1872 ns FETCH STATE: Instruction is Fetched

1880 ns DECODE STATE: Instruction is Decoded

1888 ns Instruction : LLS, 0

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1896 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 0

AR[00]= f0, DR[01]= 81, GR[10]= f0, PR[11]= 2d

1912 ns FETCH STATE: Instruction is Fetched

1920 ns DECODE STATE: Instruction is Decoded

1928 ns Instruction : LMS, 0

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1936 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 0

AR[00]= f0, DR[01]= 81, GR[10]= 00, PR[11]= 2e

1952 ns FETCH STATE: Instruction is Fetched

1960 ns DECODE STATE: Instruction is Decoded

1968 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1976 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00

register

AR[00]= 00, DR[01]= 81, GR[10]= 00, PR[11]= 2f

1992 ns FETCH STATE: Instruction is Fetched

2000 ns DECODE STATE: Instruction is Decoded

2008 ns Instruction : JMP

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2016 ns EXECUTE\_1 STATE: Executed JMP and Jumped to Instruction stored at 00

Memory Location

2032 ns FETCH STATE: Instruction is Fetched

2040 ns DECODE STATE: Instruction is Decoded

2048 ns Instruction : LLS, 5

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2056 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 5

AR[00]= 00, DR[01]= 81, GR[10]= 05, PR[11]= 01

2072 ns FETCH STATE: Instruction is Fetched

2080 ns DECODE STATE: Instruction is Decoded

2088 ns Instruction : LMS, 4

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2096 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 4

AR[00]= 00, DR[01]= 81, GR[10]= 45, PR[11]= 02

2112 ns FETCH STATE: Instruction is Fetched

2120 ns DECODE STATE: Instruction is Decoded

2128 ns Instruction : CPR, 01, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2136 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 01 register

AR[00]= 00, DR[01]= 45, GR[10]= 45, PR[11]= 03

2152 ns FETCH STATE: Instruction is Fetched

2160 ns DECODE STATE: Instruction is Decoded

2168 ns Instruction : LLS, 0

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2176 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 0

AR[00]= 00, DR[01]= 45, GR[10]= 40, PR[11]= 04

2192 ns FETCH STATE: Instruction is Fetched

2200 ns DECODE STATE: Instruction is Decoded

2208 ns Instruction : LMS, 8

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2216 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 8

AR[00]= 00, DR[01]= 45, GR[10]= 80, PR[11]= 05

2232 ns FETCH STATE: Instruction is Fetched

2240 ns DECODE STATE: Instruction is Decoded

2248 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2256 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00

register

AR[00]= 80, DR[01]= 45, GR[10]= 80, PR[11]= 06

2272 ns FETCH STATE: Instruction is Fetched

2280 ns DECODE STATE: Instruction is Decoded

2288 ns Instruction : RDM, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2296 ns EXECUTE\_1 STATE: Executing RDM...Setting AR = 80 to Memory Address...

2304 ns EXECUTE\_2 STATE: Read the Data 89 from Memory Location 80 and Stored into 10 Register

AR[00]= 80, DR[01]= 45, GR[10]= 89, PR[11]= 07

2320 ns FETCH STATE: Instruction is Fetched

2328 ns DECODE STATE: Instruction is Decoded

2336 ns Instruction : ADD, 10, 01;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2344 ns EXECUTE\_1 STATE: Executing ADD...Providing ALU Opcodes 89 and 45 and

Adding them...

2352 ns EXECUTE\_2 STATE: Added the 10 Register Data 89 with 01 Register Data 45 and the Result is Stored into 10 Register

AR[00]= 80, DR[01]= 45, GR[10]= ce, PR[11]= 08

2368 ns FETCH STATE: Instruction is Fetched

2376 ns DECODE STATE: Instruction is Decoded

2384 ns Instruction : WRM, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2392 ns EXECUTE\_1 STATE: Executing WRM...Setting AR = 80 to Memory

Address...

2400 ns EXECUTE\_2 STATE: 10 Register Data ce is Written at Memory Address 80

WRITTEN DATA : RAM[80h]= ce, RAM[81h]= cc, RAM[82h]= 82, RAM[FO]= 81

2416 ns FETCH STATE: Instruction is Fetched

2424 ns DECODE STATE: Instruction is Decoded

2432 ns Instruction : LLS, 1

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2440 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 1

AR[00]= 80, DR[01]= 45, GR[10]= c1, PR[11]= 0a

2456 ns FETCH STATE: Instruction is Fetched

2464 ns DECODE STATE: Instruction is Decoded

2472 ns Instruction : LMS, 8

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2480 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 8

AR[00]= 80, DR[01]= 45, GR[10]= 81, PR[11]= 0b

2496 ns FETCH STATE: Instruction is Fetched

2504 ns DECODE STATE: Instruction is Decoded

2512 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2520 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00

register

AR[00]= 81, DR[01]= 45, GR[10]= 81, PR[11]= 0c

2536 ns FETCH STATE: Instruction is Fetched

2544 ns DECODE STATE: Instruction is Decoded

2552 ns Instruction : RDM, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2560 ns EXECUTE\_1 STATE: Executing RDM...Setting AR = 81 to Memory Address...

2568 ns EXECUTE\_2 STATE: Read the Data cc from Memory Location 81 and Stored

into 10 Register

AR[00]= 81, DR[01]= 45, GR[10]= cc, PR[11]= 0d

2584 ns FETCH STATE: Instruction is Fetched

2592 ns DECODE STATE: Instruction is Decoded

2600 ns Instruction : CPR, 01, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2608 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 01

register

AR[00]= 81, DR[01]= cc, GR[10]= cc, PR[11]= 0e

2624 ns FETCH STATE: Instruction is Fetched

2632 ns DECODE STATE: Instruction is Decoded

2640 ns Instruction : LLS, 1

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2648 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 1

AR[00]= 81, DR[01]= cc, GR[10]= c1, PR[11]= 0f

2664 ns FETCH STATE: Instruction is Fetched

2672 ns DECODE STATE: Instruction is Decoded

2680 ns Instruction : LMS, 9

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2688 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 9

AR[00]= 81, DR[01]= cc, GR[10]= 91, PR[11]= 10

2704 ns FETCH STATE: Instruction is Fetched

2712 ns DECODE STATE: Instruction is Decoded

2720 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2728 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00

register

AR[00]= 91, DR[01]= cc, GR[10]= 91, PR[11]= 11

2744 ns FETCH STATE: Instruction is Fetched

2752 ns DECODE STATE: Instruction is Decoded

2760 ns Instruction : RDM, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2768 ns EXECUTE\_1 STATE: Executing RDM...Setting AR = 91 to Memory Address...

2776 ns EXECUTE\_2 STATE: Read the Data 77 from Memory Location 91 and Stored

into 10 Register

AR[00]= 91, DR[01]= cc, GR[10]= 77, PR[11]= 12

2792 ns FETCH STATE: Instruction is Fetched

2800 ns DECODE STATE: Instruction is Decoded

2808 ns Instruction : ADD, 01, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2816 ns EXECUTE\_1 STATE: Executing ADD...Providing ALU Opcodes cc and 77 and

Adding them...

2824 ns EXECUTE\_2 STATE: Added the 01 Register Data cc with 10 Register Data

77 and the Result is Stored into 01 Register

AR[00]= 91, DR[01]= 43, GR[10]= 77, PR[11]= 13

2840 ns FETCH STATE: Instruction is Fetched

2848 ns DECODE STATE: Instruction is Decoded

2856 ns Instruction : LLS, 1

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2864 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 1

AR[00]= 91, DR[01]= 43, GR[10]= 71, PR[11]= 14

2880 ns FETCH STATE: Instruction is Fetched

2888 ns DECODE STATE: Instruction is Decoded

2896 ns Instruction : LMS, 8

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2904 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 8

AR[00]= 91, DR[01]= 43, GR[10]= 81, PR[11]= 15

2920 ns FETCH STATE: Instruction is Fetched

2928 ns DECODE STATE: Instruction is Decoded

2936 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2944 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00

register

AR[00]= 81, DR[01]= 43, GR[10]= 81, PR[11]= 16

2960 ns FETCH STATE: Instruction is Fetched

2968 ns DECODE STATE: Instruction is Decoded

2976 ns Instruction : WRM, 01;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2984 ns EXECUTE\_1 STATE: Executing WRM...Setting AR = 81 to Memory Address...

2992 ns EXECUTE\_2 STATE: 01 Register Data 43 is Written at Memory Address 81

WRITTEN DATA : RAM[80h]= ce,RAM[81h]= 43, RAM[82h]= 82, RAM[FO]= 81

3008 ns FETCH STATE: Instruction is Fetched

3016 ns DECODE STATE: Instruction is Decoded

3024 ns Instruction : LLS, 2

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

3032 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 2

AR[00]= 81, DR[01]= 43, GR[10]= 82, PR[11]= 18

3048 ns FETCH STATE: Instruction is Fetched

3056 ns DECODE STATE: Instruction is Decoded

3064 ns Instruction : LMS, 8

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

3072 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 8

AR[00]= 81, DR[01]= 43, GR[10]= 82, PR[11]= 19

3088 ns FETCH STATE: Instruction is Fetched

3096 ns DECODE STATE: Instruction is Decoded

3104 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

3112 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00

register

AR[00]= 82, DR[01]= 43, GR[10]= 82, PR[11]= 1a

3128 ns FETCH STATE: Instruction is Fetched

3136 ns DECODE STATE: Instruction is Decoded

3144 ns Instruction : RDM, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

3152 ns EXECUTE\_1 STATE: Executing RDM...Setting AR = 82 to Memory Address...

3160 ns EXECUTE\_2 STATE: Read the Data 82 from Memory Location 82 and Stored into 10 Register

AR[00]= 82, DR[01]= 43, GR[10]= 82, PR[11]= 1b

3176 ns FETCH STATE: Instruction is Fetched

3184 ns DECODE STATE: Instruction is Decoded

3192 ns Instruction : CPR, 01, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

3200 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 01

register

AR[00]= 82, DR[01]= 82, GR[10]= 82, PR[11]= 1c

3216 ns FETCH STATE: Instruction is Fetched

3224 ns DECODE STATE: Instruction is Decoded

3232 ns Instruction : LLS, 2

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

3240 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 2

AR[00]= 82, DR[01]= 82, GR[10]= 82, PR[11]= 1d

3256 ns FETCH STATE: Instruction is Fetched

3264 ns DECODE STATE: Instruction is Decoded

3272 ns Instruction : LMS, 9

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

3280 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 9

AR[00]= 82, DR[01]= 82, GR[10]= 92, PR[11]= 1e

3296 ns FETCH STATE: Instruction is Fetched

3304 ns DECODE STATE: Instruction is Decoded

3312 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

3320 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00

register

AR[00]= 92, DR[01]= 82, GR[10]= 92, PR[11]= 1f

3336 ns FETCH STATE: Instruction is Fetched

3344 ns DECODE STATE: Instruction is Decoded

3352 ns Instruction : RDM, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

3360 ns EXECUTE\_1 STATE: Executing RDM...Setting AR = 92 to Memory Address...

3368 ns EXECUTE\_2 STATE: Read the Data 88 from Memory Location 92 and Stored

into 10 Register

AR[00]= 92, DR[01]= 82, GR[10]= 88, PR[11]= 20

3384 ns FETCH STATE: Instruction is Fetched

3392 ns DECODE STATE: Instruction is Decoded

3400 ns Instruction : SUB, 10, 01;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

3408 ns EXECUTE\_1 STATE: Executing SUB...Providing ALU Opcodes 88 and 82 and

Adding them...

3416 ns EXECUTE\_2 STATE: Subtracted the 01 Register Data 82 from 10 Register

Data 88 and the Result is Stored into 10 Register

AR[00]= 92, DR[01]= 82, GR[10]= 06, PR[11]= 21

3432 ns FETCH STATE: Instruction is Fetched

3440 ns DECODE STATE: Instruction is Decoded

3448 ns Instruction : LLS, 2

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

3456 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 2

AR[00]= 92, DR[01]= 82, GR[10]= 02, PR[11]= 22

3472 ns FETCH STATE: Instruction is Fetched

3480 ns DECODE STATE: Instruction is Decoded

3488 ns Instruction : LMS, 8

UPDATE STATE:PR Register is Incremented to Fetch the next

Instruction.

3496 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 8

AR[00]= 92, DR[01]= 82, GR[10]= 82, PR[11]= 23

3512 ns FETCH STATE: Instruction is Fetched

3520 ns DECODE STATE: Instruction is Decoded

3528 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

3536 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00

register

AR[00]= 82, DR[01]= 82, GR[10]= 82, PR[11]= 24

3552 ns FETCH STATE: Instruction is Fetched

3560 ns DECODE STATE: Instruction is Decoded

3568 ns Instruction : WRM, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction

3576 ns EXECUTE\_1 STATE: Executing WRM...Setting AR = 82 to Memory

Address...

3584 ns EXECUTE\_2 STATE: 10 Register Data 82 is Written at Memory Address 82

3600 ns FETCH STATE: Instruction is Fetched

3608 ns DECODE STATE: Instruction is Decoded

3616 ns Instruction : CFR;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

3624 ns EXECUTE\_1 STATE: Executed CFR and Stored data at LSBs of GR = 0

AR[00]= 82, DR[01]= 82, GR[10]= 80, PR[11]= 26

3640 ns FETCH STATE: Instruction is Fetched

3648 ns DECODE STATE: Instruction is Decoded

3656 ns Instruction : CPR, 01, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

3664 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 01

register

AR[00]= 82, DR[01]= 80, GR[10]= 80, PR[11]= 27

3680 ns FETCH STATE: Instruction is Fetched

3688 ns DECODE STATE: Instruction is Decoded

3696 ns Instruction : LLS, 0

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

3704 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 0

AR[00]= 82, DR[01]= 80, GR[10]= 80, PR[11]= 28

3720 ns FETCH STATE: Instruction is Fetched

3728 ns DECODE STATE: Instruction is Decoded

3736 ns Instruction : LMS, f

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

3744 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = f

AR[00]= 82, DR[01]= 80, GR[10]= f0, PR[11]= 29

3760 ns FETCH STATE: Instruction is Fetched

3768 ns DECODE STATE: Instruction is Decoded

3776 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

3784 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00 register

AR[00]= f0, DR[01]= 80, GR[10]= f0, PR[11]= 2a

3800 ns FETCH STATE: Instruction is Fetched

3808 ns DECODE STATE: Instruction is Decoded

3816 ns Instruction : WRM, 01;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

3824 ns EXECUTE\_1 STATE: Executing WRM...Setting AR = f0 to Memory Address...

3832 ns EXECUTE\_2 STATE: 01 Register Data 80 is Written at Memory Address f0

WRITTEN DATA : RAM[80h]= ce, RAM[81h]= 43,RAM[82h]= 82, RAM[FO]= 80

3848 ns FETCH STATE: Instruction is Fetched

3856 ns DECODE STATE: Instruction is Decoded

3864 ns Instruction : NOP

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

3872 ns EXECUTE\_1 STATE: Executed NOP and No Operation is Perfomed

3888 ns FETCH STATE: Instruction is Fetched

$finish called from file "scalar\_test.v", line 133.

$finish at simulation time 388900

V C S S i m u l a t i o n R e p o r t

Time: 3889000 ps

CPU Time: 0.270 seconds; Data structure size: 0.0Mb

Fri Dec 5 03:56:12 2014

**C.2 Reports (contents) from Netlist (Post-synthesis) Simulations (VCS or NCVERILOG)**

Chronologic VCS simulator copyright 1991-2014

Contains Synopsys proprietary information.

Compiler version I-2014.03-2; Runtime version I-2014.03-2; Dec 5 15:24 2014

Address= 00, Data at 00 = 75

Address= 01, Data at 01 = 84

Address= 02, Data at 02 = 46

Address= 03, Data at 03 = 70

Address= 04, Data at 04 = 88

Address= 05, Data at 05 = 42

Address= 06, Data at 06 = 28

Address= 07, Data at 07 = 59

Address= 08, Data at 08 = 38

Address= 09, Data at 09 = 71

Address= 0a, Data at 0a = 88

Address= 0b, Data at 0b = 42

Address= 0c, Data at 0c = 28

Address= 0d, Data at 0d = 46

Address= 0e, Data at 0e = 71

Address= 0f, Data at 0f = 89

Address= 10, Data at 10 = 42

Address= 11, Data at 11 = 28

Address= 12, Data at 12 = 56

Address= 13, Data at 13 = 71

Address= 14, Data at 14 = 88

Address= 15, Data at 15 = 42

Address= 16, Data at 16 = 34

Address= 17, Data at 17 = 72

Address= 18, Data at 18 = 88

Address= 19, Data at 19 = 42

Address= 1a, Data at 1a = 28

Address= 1b, Data at 1b = 46

Address= 1c, Data at 1c = 72

Address= 1d, Data at 1d = 89

Address= 1e, Data at 1e = 42

Address= 1f, Data at 1f = 28

Address= 20, Data at 20 = 69

Address= 21, Data at 21 = 72

Address= 22, Data at 22 = 88

Address= 23, Data at 23 = 42

Address= 24, Data at 24 = 38

Address= 25, Data at 25 = 90

Address= 26, Data at 26 = 46

Address= 27, Data at 27 = 70

Address= 28, Data at 28 = 8f

Address= 29, Data at 29 = 42

Address= 2a, Data at 2a = 34

Address= 2b, Data at 2b = 00

Address= 2c, Data at 2c = 70

Address= 2d, Data at 2d = 80

Address= 2e, Data at 2e = 42

Address= 2f, Data at 2f = 10

Address= 30, Data at 30 = xx

Address= 31, Data at 31 = xx

Address= 32, Data at 32 = xx

Address= 33, Data at 33 = xx

Address= 34, Data at 34 = xx

Address= 35, Data at 35 = xx

Address= 36, Data at 36 = xx

Address= 37, Data at 37 = xx

Address= 38, Data at 38 = xx

Address= 39, Data at 39 = xx

Address= 3a, Data at 3a = xx

Address= 3b, Data at 3b = xx

Address= 3c, Data at 3c = xx

Address= 3d, Data at 3d = xx

Address= 3e, Data at 3e = xx

Address= 3f, Data at 3f = xx

Address= 40, Data at 40 = xx

Address= 41, Data at 41 = xx

Address= 42, Data at 42 = xx

Address= 43, Data at 43 = xx

Address= 44, Data at 44 = xx

Address= 45, Data at 45 = xx

Address= 46, Data at 46 = xx

Address= 47, Data at 47 = xx

Address= 48, Data at 48 = xx

Address= 49, Data at 49 = xx

Address= 4a, Data at 4a = xx

Address= 4b, Data at 4b = xx

Address= 4c, Data at 4c = xx

Address= 4d, Data at 4d = xx

Address= 4e, Data at 4e = xx

Address= 4f, Data at 4f = xx

Address= 50, Data at 50 = xx

Address= 51, Data at 51 = xx

Address= 52, Data at 52 = xx

Address= 53, Data at 53 = xx

Address= 54, Data at 54 = xx

Address= 55, Data at 55 = xx

Address= 56, Data at 56 = xx

Address= 57, Data at 57 = xx

Address= 58, Data at 58 = xx

Address= 59, Data at 59 = xx

Address= 5a, Data at 5a = xx

Address= 5b, Data at 5b = xx

Address= 5c, Data at 5c = xx

Address= 5d, Data at 5d = xx

Address= 5e, Data at 5e = xx

Address= 5f, Data at 5f = xx

Address= 60, Data at 60 = xx

Address= 61, Data at 61 = xx

Address= 62, Data at 62 = xx

Address= 63, Data at 63 = xx

Address= 64, Data at 64 = xx

Address= 65, Data at 65 = xx

Address= 66, Data at 66 = xx

Address= 67, Data at 67 = xx

Address= 68, Data at 68 = xx

Address= 69, Data at 69 = xx

Address= 6a, Data at 6a = xx

Address= 6b, Data at 6b = xx

Address= 6c, Data at 6c = xx

Address= 6d, Data at 6d = xx

Address= 6e, Data at 6e = xx

Address= 6f, Data at 6f = xx

Address= 70, Data at 70 = xx

Address= 71, Data at 71 = xx

Address= 72, Data at 72 = xx

Address= 73, Data at 73 = xx

Address= 74, Data at 74 = xx

Address= 75, Data at 75 = xx

Address= 76, Data at 76 = xx

Address= 77, Data at 77 = xx

Address= 78, Data at 78 = xx

Address= 79, Data at 79 = xx

Address= 7a, Data at 7a = xx

Address= 7b, Data at 7b = xx

Address= 7c, Data at 7c = xx

Address= 7d, Data at 7d = xx

Address= 7e, Data at 7e = xx

Address= 7f, Data at 7f = xx

Address= 80, Data at 80 = 44

Address= 81, Data at 81 = 55

Address= 82, Data at 82 = 66

Address= 83, Data at 83 = xx

Address= 84, Data at 84 = xx

Address= 85, Data at 85 = xx

Address= 86, Data at 86 = xx

Address= 87, Data at 87 = xx

Address= 88, Data at 88 = xx

Address= 89, Data at 89 = xx

Address= 8a, Data at 8a = xx

Address= 8b, Data at 8b = xx

Address= 8c, Data at 8c = xx

Address= 8d, Data at 8d = xx

Address= 8e, Data at 8e = xx

Address= 8f, Data at 8f = xx

Address= 90, Data at 90 = xx

Address= 91, Data at 91 = 77

Address= 92, Data at 92 = 88

Address= 93, Data at 93 = xx

Address= 94, Data at 94 = xx

Address= 95, Data at 95 = xx

Address= 96, Data at 96 = xx

Address= 97, Data at 97 = xx

Address= 98, Data at 98 = xx

Address= 99, Data at 99 = xx

Address= 9a, Data at 9a = xx

Address= 9b, Data at 9b = xx

Address= 9c, Data at 9c = xx

Address= 9d, Data at 9d = xx

Address= 9e, Data at 9e = xx

Address= 9f, Data at 9f = xx

Address= a0, Data at a0 = xx

Address= a1, Data at a1 = xx

Address= a2, Data at a2 = xx

Address= a3, Data at a3 = xx

Address= a4, Data at a4 = xx

Address= a5, Data at a5 = xx

Address= a6, Data at a6 = xx

Address= a7, Data at a7 = xx

Address= a8, Data at a8 = xx

Address= a9, Data at a9 = xx

Address= aa, Data at aa = xx

Address= ab, Data at ab = xx

Address= ac, Data at ac = xx

Address= ad, Data at ad = xx

Address= ae, Data at ae = xx

Address= af, Data at af = xx

Address= b0, Data at b0 = xx

Address= b1, Data at b1 = xx

Address= b2, Data at b2 = xx

Address= b3, Data at b3 = xx

Address= b4, Data at b4 = xx

Address= b5, Data at b5 = xx

Address= b6, Data at b6 = xx

Address= b7, Data at b7 = xx

Address= b8, Data at b8 = xx

Address= b9, Data at b9 = xx

Address= ba, Data at ba = xx

Address= bb, Data at bb = xx

Address= bc, Data at bc = xx

Address= bd, Data at bd = xx

Address= be, Data at be = xx

Address= bf, Data at bf = xx

Address= c0, Data at c0 = xx

Address= c1, Data at c1 = xx

Address= c2, Data at c2 = xx

Address= c3, Data at c3 = xx

Address= c4, Data at c4 = xx

Address= c5, Data at c5 = xx

Address= c6, Data at c6 = xx

Address= c7, Data at c7 = xx

Address= c8, Data at c8 = xx

Address= c9, Data at c9 = xx

Address= ca, Data at ca = xx

Address= cb, Data at cb = xx

Address= cc, Data at cc = xx

Address= cd, Data at cd = xx

Address= ce, Data at ce = xx

Address= cf, Data at cf = xx

Address= d0, Data at d0 = xx

Address= d1, Data at d1 = xx

Address= d2, Data at d2 = xx

Address= d3, Data at d3 = xx

Address= d4, Data at d4 = xx

Address= d5, Data at d5 = xx

Address= d6, Data at d6 = xx

Address= d7, Data at d7 = xx

Address= d8, Data at d8 = xx

Address= d9, Data at d9 = xx

Address= da, Data at da = xx

Address= db, Data at db = xx

Address= dc, Data at dc = xx

Address= dd, Data at dd = xx

Address= de, Data at de = xx

Address= df, Data at df = xx

Address= e0, Data at e0 = xx

Address= e1, Data at e1 = xx

Address= e2, Data at e2 = xx

Address= e3, Data at e3 = xx

Address= e4, Data at e4 = xx

Address= e5, Data at e5 = xx

Address= e6, Data at e6 = xx

Address= e7, Data at e7 = xx

Address= e8, Data at e8 = xx

Address= e9, Data at e9 = xx

Address= ea, Data at ea = xx

Address= eb, Data at eb = xx

Address= ec, Data at ec = xx

Address= ed, Data at ed = xx

Address= ee, Data at ee = xx

Address= ef, Data at ef = xx

Address= f0, Data at f0 = xx

Address= f1, Data at f1 = xx

Address= f2, Data at f2 = xx

Address= f3, Data at f3 = xx

Address= f4, Data at f4 = xx

Address= f5, Data at f5 = xx

Address= f6, Data at f6 = xx

Address= f7, Data at f7 = xx

Address= f8, Data at f8 = xx

Address= f9, Data at f9 = xx

Address= fa, Data at fa = xx

Address= fb, Data at fb = xx

Address= fc, Data at fc = xx

Address= fd, Data at fd = xx

Address= fe, Data at fe = xx

Address= ff, Data at ff = xx

WRITEEN DATA : RAM[80h]= 44, RAM[81h]= 55, RAM[82h]= 66, RAM[FO]= xx

* **Case 1:**

16 ns FETCH STATE: Instruction is Fetched

24 ns DECODE STATE: Instruction is Decoded

32 ns Instruction : LLS, 5

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

40 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 5

AR[00]= 00, DR[01]= 00, GR[10]= 05, PR[11]= 01

56 ns FETCH STATE: Instruction is Fetched

64 ns DECODE STATE: Instruction is Decoded

72 ns Instruction : LMS, 4

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

80 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 4

AR[00]= 00, DR[01]= 00, GR[10]= 45, PR[11]= 02

96 ns FETCH STATE: Instruction is Fetched

104 ns DECODE STATE: Instruction is Decoded

112 ns Instruction : CPR, 01, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

120 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 01 reg.

AR[00]= 00, DR[01]= 45, GR[10]= 45, PR[11]= 03

136 ns FETCH STATE: Instruction is Fetched

144 ns DECODE STATE: Instruction is Decoded

152 ns Instruction : LLS, 0

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

160 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 0

AR[00]= 00, DR[01]= 45, GR[10]= 40, PR[11]= 04

176 ns FETCH STATE: Instruction is Fetched

184 ns DECODE STATE: Instruction is Decoded

192 ns Instruction : LMS, 8

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

200 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 8

AR[00]= 00, DR[01]= 45, GR[10]= 80, PR[11]= 05

216 ns FETCH STATE: Instruction is Fetched

224 ns DECODE STATE: Instruction is Decoded

232 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

240 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00 reg.

AR[00]= 80, DR[01]= 45, GR[10]= 80, PR[11]= 06

256 ns FETCH STATE: Instruction is Fetched

264 ns DECODE STATE: Instruction is Decoded

272 ns Instruction : RDM, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

280 ns EXECUTE\_1 STATE: Executing RDM...Setting AR = 80 to Memory Address...

288 ns EXECUTE\_2 STATE: Read the Data 44 from Memory Location 80 and Stored

into 10 Register

AR[00]= 80, DR[01]= 45, GR[10]= 44, PR[11]= 07

304 ns FETCH STATE: Instruction is Fetched

312 ns DECODE STATE: Instruction is Decoded

320 ns Instruction : ADD, 10, 01;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

328 ns EXECUTE\_1 STATE: Executing ADD...Providing ALU Opcodes 44 and 45 and

Adding them...

336 ns EXECUTE\_2 STATE: Added the 10 Register Data 44 with 01 Register Data

45 and the Result is Stored into 10 Register

AR[00]= 80, DR[01]= 45, GR[10]= 89, PR[11]= 08

352 ns FETCH STATE: Instruction is Fetched

360 ns DECODE STATE: Instruction is Decoded

368 ns Instruction : WRM, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

376 ns EXECUTE\_1 STATE: Executing WRM...Setting AR = 80 to Memory Address...

384 ns EXECUTE\_2 STATE: 10 Register Data 89 is Written at Memory Address 80

WRITEEN DATA : RAM[80h]= 89, RAM[81h]= 55, RAM[82h]= 66, RAM[FO]= xx

* **Case 2:**

400 ns FETCH STATE: Instruction is Fetched

408 ns DECODE STATE: Instruction is Decoded

416 ns Instruction : LLS, 1

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

424 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 1

AR[00]= 80, DR[01]= 45, GR[10]= 81, PR[11]= 0a

440 ns FETCH STATE: Instruction is Fetched

448 ns DECODE STATE: Instruction is Decoded

456 ns Instruction : LMS, 8

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

464 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 8

AR[00]= 80, DR[01]= 45, GR[10]= 81, PR[11]= 0b

480 ns FETCH STATE: Instruction is Fetched

488 ns DECODE STATE: Instruction is Decoded

496 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

504 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00

register

AR[00]= 81, DR[01]= 45, GR[10]= 81, PR[11]= 0c

520 ns FETCH STATE: Instruction is Fetched

528 ns DECODE STATE: Instruction is Decoded

536 ns Instruction : RDM, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

544 ns EXECUTE\_1 STATE: Executing RDM...Setting AR = 81 to Memory Address...

552 ns EXECUTE\_2 STATE: Read the Data 55 from Memory Location 81 and Stored

into 10 Register

AR[00]= 81, DR[01]= 45, GR[10]= 55, PR[11]= 0d

568 ns FETCH STATE: Instruction is Fetched

576 ns DECODE STATE: Instruction is Decoded

584 ns Instruction : CPR, 01, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

592 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 01

register

AR[00]= 81, DR[01]= 55, GR[10]= 55, PR[11]= 0e

608 ns FETCH STATE: Instruction is Fetched

616 ns DECODE STATE: Instruction is Decoded

624 ns Instruction : LLS, 1

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

632 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 1

AR[00]= 81, DR[01]= 55, GR[10]= 51, PR[11]= 0f

648 ns FETCH STATE: Instruction is Fetched

656 ns DECODE STATE: Instruction is Decoded

664 ns Instruction : LMS, 9

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

672 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 9

AR[00]= 81, DR[01]= 55, GR[10]= 91, PR[11]= 10

688 ns FETCH STATE: Instruction is Fetched

696 ns DECODE STATE: Instruction is Decoded

704 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

712 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00

register

AR[00]= 91, DR[01]= 55, GR[10]= 91, PR[11]= 11

728 ns FETCH STATE: Instruction is Fetched

736 ns DECODE STATE: Instruction is Decoded

744 ns Instruction : RDM, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

752 ns EXECUTE\_1 STATE: Executing RDM...Setting AR = 91 to Memory Address...

760 ns EXECUTE\_2 STATE: Read the Data 77 from Memory Location 91 and Stored

into 10 Register

AR[00]= 91, DR[01]= 55, GR[10]= 77, PR[11]= 12

776 ns FETCH STATE: Instruction is Fetched

784 ns DECODE STATE: Instruction is Decoded

792 ns Instruction : ADD, 01, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

800 ns EXECUTE\_1 STATE: Executing ADD...Providing ALU Opcodes 55 and 77 and

Adding them...

808 ns EXECUTE\_2 STATE: Added the 01 Register Data 55 with 10 Register Data

77 and the Result is Stored into 01 Register

AR[00]= 91, DR[01]= cc, GR[10]= 77, PR[11]= 13

824 ns FETCH STATE: Instruction is Fetched

832 ns DECODE STATE: Instruction is Decoded

840 ns Instruction : LLS, 1

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

848 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 1

AR[00]= 91, DR[01]= cc, GR[10]= 71, PR[11]= 14

864 ns FETCH STATE: Instruction is Fetched

872 ns DECODE STATE: Instruction is Decoded

880 ns Instruction : LMS, 8

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

888 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 8

AR[00]= 91, DR[01]= cc, GR[10]= 81, PR[11]= 15

904 ns FETCH STATE: Instruction is Fetched

912 ns DECODE STATE: Instruction is Decoded

920 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

928 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00

register

AR[00]= 81, DR[01]= cc, GR[10]= 81, PR[11]= 16

944 ns FETCH STATE: Instruction is Fetched

952 ns DECODE STATE: Instruction is Decoded

960 ns Instruction : WRM, 01;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

968 ns EXECUTE\_1 STATE: Executing WRM...Setting AR = 81 to Memory Address...

976 ns EXECUTE\_2 STATE: 01 Register Data cc is Written at Memory Address 81

WRITTEN DATA : RAM[80h]= 89, RAM[81h]= cc, RAM[82h]= 66, RAM[FO]= xx

* **Case 3:**

992 ns FETCH STATE: Instruction is Fetched

1000 ns DECODE STATE: Instruction is Decoded

1008 ns Instruction : LLS, 2

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1016 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 2

AR[00]= 81, DR[01]= cc, GR[10]= 82, PR[11]= 18

1032 ns FETCH STATE: Instruction is Fetched

1040 ns DECODE STATE: Instruction is Decoded

1048 ns Instruction : LMS, 8

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1056 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 8

AR[00]= 81, DR[01]= cc, GR[10]= 82, PR[11]= 19

1072 ns FETCH STATE: Instruction is Fetched

1080 ns DECODE STATE: Instruction is Decoded

1088 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1096 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00 register

AR[00]= 82, DR[01]= cc, GR[10]= 82, PR[11]= 1a

1112 ns FETCH STATE: Instruction is Fetched

1120 ns DECODE STATE: Instruction is Decoded

1128 ns Instruction : RDM, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1136 ns EXECUTE\_1 STATE: Executing RDM...Setting AR = 82 to Memory Address...

1144 ns EXECUTE\_2 STATE: Read the Data 66 from Memory Location 82 and Stored into 10 Register

AR[00]= 82, DR[01]= cc, GR[10]= 66, PR[11]= 1b

1160 ns FETCH STATE: Instruction is Fetched

1168 ns DECODE STATE: Instruction is Decoded

1176 ns Instruction : CPR, 01, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1184 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 01

register

AR[00]= 82, DR[01]= 66, GR[10]= 66, PR[11]= 1c

1200 ns FETCH STATE: Instruction is Fetched

1208 ns DECODE STATE: Instruction is Decoded

1216 ns Instruction : LLS, 2

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1224 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 2

AR[00]= 82, DR[01]= 66, GR[10]= 62, PR[11]= 1d

1240 ns FETCH STATE: Instruction is Fetched

1248 ns DECODE STATE: Instruction is Decoded

1256 ns Instruction : LMS, 9

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1264 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 9

AR[00]= 82, DR[01]= 66, GR[10]= 92, PR[11]= 1e

1280 ns FETCH STATE: Instruction is Fetched

1288 ns DECODE STATE: Instruction is Decoded

1296 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1304 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00

register

AR[00]= 92, DR[01]= 66, GR[10]= 92, PR[11]= 1f

1320 ns FETCH STATE: Instruction is Fetched

1328 ns DECODE STATE: Instruction is Decoded

1336 ns Instruction : RDM, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1344 ns EXECUTE\_1 STATE: Executing RDM...Setting AR = 92 to Memory Address...

1352 ns EXECUTE\_2 STATE: Read the Data 88 from Memory Location 92 and Stored into 10 Register

AR[00]= 92, DR[01]= 66, GR[10]= 88, PR[11]= 20

1368 ns FETCH STATE: Instruction is Fetched

1376 ns DECODE STATE: Instruction is Decoded

1384 ns Instruction : SUB, 10, 01;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1392 ns EXECUTE\_1 STATE: Executing SUB...Providing ALU Opcodes 88 and 66 and Adding them...

1400 ns EXECUTE\_2 STATE: Subtracted the 01 Register Data 66 from 10 Register

Data 88 and the Result is Stored into 10 Register

AR[00]= 92, DR[01]= 66, GR[10]= 22, PR[11]= 21

1416 ns FETCH STATE: Instruction is Fetched

1424 ns DECODE STATE: Instruction is Decoded

1432 ns Instruction : LLS, 2

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1440 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 2

AR[00]= 92, DR[01]= 66, GR[10]= 22, PR[11]= 22

1456 ns FETCH STATE: Instruction is Fetched

1464 ns DECODE STATE: Instruction is Decoded

1472 ns Instruction : LMS, 8

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1480 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 8

AR[00]= 92, DR[01]= 66, GR[10]= 82, PR[11]= 23

1496 ns FETCH STATE: Instruction is Fetched

1504 ns DECODE STATE: Instruction is Decoded

1512 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1520 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00

register

AR[00]= 82, DR[01]= 66, GR[10]= 82, PR[11]= 24

1536 ns FETCH STATE: Instruction is Fetched

1544 ns DECODE STATE: Instruction is Decoded

1552 ns Instruction : WRM, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1560 ns EXECUTE\_1 STATE: Executing WRM...Setting AR = 82 to Memory Address...

1568 ns EXECUTE\_2 STATE: 10 Register Data 82 is Written at Memory Address 82

WRITTEN DATA : RAM[80h]= 89, RAM[81h]= cc, RAM[82h]= 82, RAM[FO]= xx

* **Case 4:**

1584 ns FETCH STATE: Instruction is Fetched

1592 ns DECODE STATE: Instruction is Decoded

1600 ns Instruction : CFR;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1608 ns EXECUTE\_1 STATE: Executed CFR and Stored data at LSBs of GR = 1

AR[00]= 82, DR[01]= 66, GR[10]= 81, PR[11]= 26

1624 ns FETCH STATE: Instruction is Fetched

1632 ns DECODE STATE: Instruction is Decoded

1640 ns Instruction : CPR, 01, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1648 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 01 register

AR[00]= 82, DR[01]= 81, GR[10]= 81, PR[11]= 27

1664 ns FETCH STATE: Instruction is Fetched

1672 ns DECODE STATE: Instruction is Decoded

1680 ns Instruction : LLS, 0

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1688 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 0

AR[00]= 82, DR[01]= 81, GR[10]= 80, PR[11]= 28

1704 ns FETCH STATE: Instruction is Fetched

1712 ns DECODE STATE: Instruction is Decoded

1720 ns Instruction : LMS, f

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1728 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = f

AR[00]= 82, DR[01]= 81, GR[10]= f0, PR[11]= 29

1744 ns FETCH STATE: Instruction is Fetched

1752 ns DECODE STATE: Instruction is Decoded

1760 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1768 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00 register

AR[00]= f0, DR[01]= 81, GR[10]= f0, PR[11]= 2a

1784 ns FETCH STATE: Instruction is Fetched

1792 ns DECODE STATE: Instruction is Decoded

1800 ns Instruction : WRM, 01;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1808 ns EXECUTE\_1 STATE: Executing WRM...Setting AR = f0 to Memory Address...

1816 ns EXECUTE\_2 STATE: 01 Register Data 81 is Written at Memory Address f0

WRITTEN DATA : RAM[80h]= 89, RAM[81h]= cc, RAM[82h]= 82, RAM[FO]= 81

1832 ns FETCH STATE: Instruction is Fetched

1840 ns DECODE STATE: Instruction is Decoded

1848 ns Instruction : NOP

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1856 ns EXECUTE\_1 STATE: Executed NOP and No Operation is Perfomed

* **Case 5:**

1872 ns FETCH STATE: Instruction is Fetched

1880 ns DECODE STATE: Instruction is Decoded

1888 ns Instruction : LLS, 0

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1896 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 0

AR[00]= f0, DR[01]= 81, GR[10]= f0, PR[11]= 2d

1912 ns FETCH STATE: Instruction is Fetched

1920 ns DECODE STATE: Instruction is Decoded

1928 ns Instruction : LMS, 0

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1936 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 0

AR[00]= f0, DR[01]= 81, GR[10]= 00, PR[11]= 2e

1952 ns FETCH STATE: Instruction is Fetched

1960 ns DECODE STATE: Instruction is Decoded

1968 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

1976 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00

register

AR[00]= 00, DR[01]= 81, GR[10]= 00, PR[11]= 2f

1992 ns FETCH STATE: Instruction is Fetched

2000 ns DECODE STATE: Instruction is Decoded

2008 ns Instruction : JMP

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2016 ns EXECUTE\_1 STATE: Executed JMP and Jumped to Instruction stored at 00

Memory Location

2032 ns FETCH STATE: Instruction is Fetched

2040 ns DECODE STATE: Instruction is Decoded

2048 ns Instruction : LLS, 5

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2056 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 5

AR[00]= 00, DR[01]= 81, GR[10]= 05, PR[11]= 01

2072 ns FETCH STATE: Instruction is Fetched

2080 ns DECODE STATE: Instruction is Decoded

2088 ns Instruction : LMS, 4

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2096 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 4

AR[00]= 00, DR[01]= 81, GR[10]= 45, PR[11]= 02

2112 ns FETCH STATE: Instruction is Fetched

2120 ns DECODE STATE: Instruction is Decoded

2128 ns Instruction : CPR, 01, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2136 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 01 register

AR[00]= 00, DR[01]= 45, GR[10]= 45, PR[11]= 03

2152 ns FETCH STATE: Instruction is Fetched

2160 ns DECODE STATE: Instruction is Decoded

2168 ns Instruction : LLS, 0

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2176 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 0

AR[00]= 00, DR[01]= 45, GR[10]= 40, PR[11]= 04

2192 ns FETCH STATE: Instruction is Fetched

2200 ns DECODE STATE: Instruction is Decoded

2208 ns Instruction : LMS, 8

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2216 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 8

AR[00]= 00, DR[01]= 45, GR[10]= 80, PR[11]= 05

2232 ns FETCH STATE: Instruction is Fetched

2240 ns DECODE STATE: Instruction is Decoded

2248 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2256 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00

register

AR[00]= 80, DR[01]= 45, GR[10]= 80, PR[11]= 06

2272 ns FETCH STATE: Instruction is Fetched

2280 ns DECODE STATE: Instruction is Decoded

2288 ns Instruction : RDM, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2296 ns EXECUTE\_1 STATE: Executing RDM...Setting AR = 80 to Memory Address...

2304 ns EXECUTE\_2 STATE: Read the Data 89 from Memory Location 80 and Stored into 10 Register

AR[00]= 80, DR[01]= 45, GR[10]= 89, PR[11]= 07

2320 ns FETCH STATE: Instruction is Fetched

2328 ns DECODE STATE: Instruction is Decoded

2336 ns Instruction : ADD, 10, 01;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2344 ns EXECUTE\_1 STATE: Executing ADD...Providing ALU Opcodes 89 and 45 and

Adding them...

2352 ns EXECUTE\_2 STATE: Added the 10 Register Data 89 with 01 Register Data 45 and the Result is Stored into 10 Register

AR[00]= 80, DR[01]= 45, GR[10]= ce, PR[11]= 08

2368 ns FETCH STATE: Instruction is Fetched

2376 ns DECODE STATE: Instruction is Decoded

2384 ns Instruction : WRM, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2392 ns EXECUTE\_1 STATE: Executing WRM...Setting AR = 80 to Memory

Address...

2400 ns EXECUTE\_2 STATE: 10 Register Data ce is Written at Memory Address 80

WRITTEN DATA : RAM[80h]= ce, RAM[81h]= cc, RAM[82h]= 82, RAM[FO]= 81

2416 ns FETCH STATE: Instruction is Fetched

2424 ns DECODE STATE: Instruction is Decoded

2432 ns Instruction : LLS, 1

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2440 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 1

AR[00]= 80, DR[01]= 45, GR[10]= c1, PR[11]= 0a

2456 ns FETCH STATE: Instruction is Fetched

2464 ns DECODE STATE: Instruction is Decoded

2472 ns Instruction : LMS, 8

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2480 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 8

AR[00]= 80, DR[01]= 45, GR[10]= 81, PR[11]= 0b

2496 ns FETCH STATE: Instruction is Fetched

2504 ns DECODE STATE: Instruction is Decoded

2512 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2520 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00

register

AR[00]= 81, DR[01]= 45, GR[10]= 81, PR[11]= 0c

2536 ns FETCH STATE: Instruction is Fetched

2544 ns DECODE STATE: Instruction is Decoded

2552 ns Instruction : RDM, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2560 ns EXECUTE\_1 STATE: Executing RDM...Setting AR = 81 to Memory Address...

2568 ns EXECUTE\_2 STATE: Read the Data cc from Memory Location 81 and Stored

into 10 Register

AR[00]= 81, DR[01]= 45, GR[10]= cc, PR[11]= 0d

2584 ns FETCH STATE: Instruction is Fetched

2592 ns DECODE STATE: Instruction is Decoded

2600 ns Instruction : CPR, 01, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2608 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 01

register

AR[00]= 81, DR[01]= cc, GR[10]= cc, PR[11]= 0e

2624 ns FETCH STATE: Instruction is Fetched

2632 ns DECODE STATE: Instruction is Decoded

2640 ns Instruction : LLS, 1

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2648 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 1

AR[00]= 81, DR[01]= cc, GR[10]= c1, PR[11]= 0f

2664 ns FETCH STATE: Instruction is Fetched

2672 ns DECODE STATE: Instruction is Decoded

2680 ns Instruction : LMS, 9

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2688 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 9

AR[00]= 81, DR[01]= cc, GR[10]= 91, PR[11]= 10

2704 ns FETCH STATE: Instruction is Fetched

2712 ns DECODE STATE: Instruction is Decoded

2720 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2728 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00

register

AR[00]= 91, DR[01]= cc, GR[10]= 91, PR[11]= 11

2744 ns FETCH STATE: Instruction is Fetched

2752 ns DECODE STATE: Instruction is Decoded

2760 ns Instruction : RDM, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2768 ns EXECUTE\_1 STATE: Executing RDM...Setting AR = 91 to Memory Address...

2776 ns EXECUTE\_2 STATE: Read the Data 77 from Memory Location 91 and Stored

into 10 Register

AR[00]= 91, DR[01]= cc, GR[10]= 77, PR[11]= 12

2792 ns FETCH STATE: Instruction is Fetched

2800 ns DECODE STATE: Instruction is Decoded

2808 ns Instruction : ADD, 01, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2816 ns EXECUTE\_1 STATE: Executing ADD...Providing ALU Opcodes cc and 77 and

Adding them...

2824 ns EXECUTE\_2 STATE: Added the 01 Register Data cc with 10 Register Data

77 and the Result is Stored into 01 Register

AR[00]= 91, DR[01]= 43, GR[10]= 77, PR[11]= 13

2840 ns FETCH STATE: Instruction is Fetched

2848 ns DECODE STATE: Instruction is Decoded

2856 ns Instruction : LLS, 1

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2864 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 1

AR[00]= 91, DR[01]= 43, GR[10]= 71, PR[11]= 14

2880 ns FETCH STATE: Instruction is Fetched

2888 ns DECODE STATE: Instruction is Decoded

2896 ns Instruction : LMS, 8

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2904 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 8

AR[00]= 91, DR[01]= 43, GR[10]= 81, PR[11]= 15

2920 ns FETCH STATE: Instruction is Fetched

2928 ns DECODE STATE: Instruction is Decoded

2936 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2944 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00

register

AR[00]= 81, DR[01]= 43, GR[10]= 81, PR[11]= 16

2960 ns FETCH STATE: Instruction is Fetched

2968 ns DECODE STATE: Instruction is Decoded

2976 ns Instruction : WRM, 01;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

2984 ns EXECUTE\_1 STATE: Executing WRM...Setting AR = 81 to Memory Address...

2992 ns EXECUTE\_2 STATE: 01 Register Data 43 is Written at Memory Address 81

WRITTEN DATA : RAM[80h]= ce,RAM[81h]= 43, RAM[82h]= 82, RAM[FO]= 81

3008 ns FETCH STATE: Instruction is Fetched

3016 ns DECODE STATE: Instruction is Decoded

3024 ns Instruction : LLS, 2

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

3032 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 2

AR[00]= 81, DR[01]= 43, GR[10]= 82, PR[11]= 18

3048 ns FETCH STATE: Instruction is Fetched

3056 ns DECODE STATE: Instruction is Decoded

3064 ns Instruction : LMS, 8

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

3072 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 8

AR[00]= 81, DR[01]= 43, GR[10]= 82, PR[11]= 19

3088 ns FETCH STATE: Instruction is Fetched

3096 ns DECODE STATE: Instruction is Decoded

3104 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

3112 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00

register

AR[00]= 82, DR[01]= 43, GR[10]= 82, PR[11]= 1a

3128 ns FETCH STATE: Instruction is Fetched

3136 ns DECODE STATE: Instruction is Decoded

3144 ns Instruction : RDM, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

3152 ns EXECUTE\_1 STATE: Executing RDM...Setting AR = 82 to Memory Address...

3160 ns EXECUTE\_2 STATE: Read the Data 82 from Memory Location 82 and Stored into 10 Register

AR[00]= 82, DR[01]= 43, GR[10]= 82, PR[11]= 1b

3176 ns FETCH STATE: Instruction is Fetched

3184 ns DECODE STATE: Instruction is Decoded

3192 ns Instruction : CPR, 01, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

3200 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 01

register

AR[00]= 82, DR[01]= 82, GR[10]= 82, PR[11]= 1c

3216 ns FETCH STATE: Instruction is Fetched

3224 ns DECODE STATE: Instruction is Decoded

3232 ns Instruction : LLS, 2

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

3240 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 2

AR[00]= 82, DR[01]= 82, GR[10]= 82, PR[11]= 1d

3256 ns FETCH STATE: Instruction is Fetched

3264 ns DECODE STATE: Instruction is Decoded

3272 ns Instruction : LMS, 9

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

3280 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 9

AR[00]= 82, DR[01]= 82, GR[10]= 92, PR[11]= 1e

3296 ns FETCH STATE: Instruction is Fetched

3304 ns DECODE STATE: Instruction is Decoded

3312 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

3320 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00

register

AR[00]= 92, DR[01]= 82, GR[10]= 92, PR[11]= 1f

3336 ns FETCH STATE: Instruction is Fetched

3344 ns DECODE STATE: Instruction is Decoded

3352 ns Instruction : RDM, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

3360 ns EXECUTE\_1 STATE: Executing RDM...Setting AR = 92 to Memory Address...

3368 ns EXECUTE\_2 STATE: Read the Data 88 from Memory Location 92 and Stored

into 10 Register

AR[00]= 92, DR[01]= 82, GR[10]= 88, PR[11]= 20

3384 ns FETCH STATE: Instruction is Fetched

3392 ns DECODE STATE: Instruction is Decoded

3400 ns Instruction : SUB, 10, 01;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

3408 ns EXECUTE\_1 STATE: Executing SUB...Providing ALU Opcodes 88 and 82 and

Adding them...

3416 ns EXECUTE\_2 STATE: Subtracted the 01 Register Data 82 from 10 Register

Data 88 and the Result is Stored into 10 Register

AR[00]= 92, DR[01]= 82, GR[10]= 06, PR[11]= 21

3432 ns FETCH STATE: Instruction is Fetched

3440 ns DECODE STATE: Instruction is Decoded

3448 ns Instruction : LLS, 2

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

3456 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 2

AR[00]= 92, DR[01]= 82, GR[10]= 02, PR[11]= 22

3472 ns FETCH STATE: Instruction is Fetched

3480 ns DECODE STATE: Instruction is Decoded

3488 ns Instruction : LMS, 8

UPDATE STATE:PR Register is Incremented to Fetch the next

Instruction.

3496 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 8

AR[00]= 92, DR[01]= 82, GR[10]= 82, PR[11]= 23

3512 ns FETCH STATE: Instruction is Fetched

3520 ns DECODE STATE: Instruction is Decoded

3528 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

3536 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00

register

AR[00]= 82, DR[01]= 82, GR[10]= 82, PR[11]= 24

3552 ns FETCH STATE: Instruction is Fetched

3560 ns DECODE STATE: Instruction is Decoded

3568 ns Instruction : WRM, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction

3576 ns EXECUTE\_1 STATE: Executing WRM...Setting AR = 82 to Memory

Address...

3584 ns EXECUTE\_2 STATE: 10 Register Data 82 is Written at Memory Address 82

3600 ns FETCH STATE: Instruction is Fetched

3608 ns DECODE STATE: Instruction is Decoded

3616 ns Instruction : CFR;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

3624 ns EXECUTE\_1 STATE: Executed CFR and Stored data at LSBs of GR = 0

AR[00]= 82, DR[01]= 82, GR[10]= 80, PR[11]= 26

3640 ns FETCH STATE: Instruction is Fetched

3648 ns DECODE STATE: Instruction is Decoded

3656 ns Instruction : CPR, 01, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

3664 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 01

register

AR[00]= 82, DR[01]= 80, GR[10]= 80, PR[11]= 27

3680 ns FETCH STATE: Instruction is Fetched

3688 ns DECODE STATE: Instruction is Decoded

3696 ns Instruction : LLS, 0

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

3704 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 0

AR[00]= 82, DR[01]= 80, GR[10]= 80, PR[11]= 28

3720 ns FETCH STATE: Instruction is Fetched

3728 ns DECODE STATE: Instruction is Decoded

3736 ns Instruction : LMS, f

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

3744 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = f

AR[00]= 82, DR[01]= 80, GR[10]= f0, PR[11]= 29

3760 ns FETCH STATE: Instruction is Fetched

3768 ns DECODE STATE: Instruction is Decoded

3776 ns Instruction : CPR, 00, 10;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

3784 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 10 to 00 register

AR[00]= f0, DR[01]= 80, GR[10]= f0, PR[11]= 2a

3800 ns FETCH STATE: Instruction is Fetched

3808 ns DECODE STATE: Instruction is Decoded

3816 ns Instruction : WRM, 01;

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

3824 ns EXECUTE\_1 STATE: Executing WRM...Setting AR = f0 to Memory Address...

3832 ns EXECUTE\_2 STATE: 01 Register Data 80 is Written at Memory Address f0

WRITTEN DATA : RAM[80h]= ce, RAM[81h]= 43,RAM[82h]= 82, RAM[FO]= 80

3848 ns FETCH STATE: Instruction is Fetched

3856 ns DECODE STATE: Instruction is Decoded

3864 ns Instruction : NOP

UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

3872 ns EXECUTE\_1 STATE: Executed NOP and No Operation is Perfomed

3888 ns FETCH STATE: Instruction is Fetched

$finish called from file "scalar\_test.v", line 133.

$finish at simulation time 388900

V C S S i m u l a t i o n R e p o r t

Time: 3889000 ps

CPU Time: 0.270 seconds; Data structure size: 0.0Mb

Fri Dec 5 03:56:12 2014

**C.3 Reports (contents) from Synthesis (Design Compiler)**

Warning: DC is only available in Tcl and XG mode. The -tcl\_mode and -xg\_mode options are no longer required.

DC Professional (TM)

DC Expert (TM)

DC Ultra (TM)

FloorPlan Manager (TM)

HDL Compiler (TM)

VHDL Compiler (TM)

Library Compiler (TM)

DesignWare Developer (TM)

DFT Compiler (TM)

BSD Compiler

Power Compiler (TM)

Version C-2009.06-SP5 for linux -- Jan 15, 2010

Copyright (c) 1988-2009 by Synopsys, Inc.

ALL RIGHTS RESERVED

This software and the associated documentation are confidential and

proprietary to Synopsys, Inc. Your use or disclosure of this software

is subject to the terms and conditions of a written license agreement

between you, or your company, and Synopsys, Inc.

The above trademark notice does not imply that you are licensed to use

all of the listed products. You are licensed to use only those products

for which you have lawfully obtained a valid license key.

Initializing...

set link\_library {/apps/toshiba/sjsu/synopsys/tc240c/tc240c.db\_WCCOM25}

/apps/toshiba/sjsu/synopsys/tc240c/tc240c.db\_WCCOM25

set target\_library {/apps/toshiba/sjsu/synopsys/tc240c/tc240c.db\_WCCOM25}

/apps/toshiba/sjsu/synopsys/tc240c/tc240c.db\_WCCOM25

set symbol\_library {/apps/toshiba/sjsu/synopsys/tc240c/tc240c.workview.sdb}

/apps/toshiba/sjsu/synopsys/tc240c/tc240c.workview.sdb

set synthetic\_library {dw\_foundation.sldb standard.sldb}

dw\_foundation.sldb standard.sldb

set\_min\_library /apps/toshiba/sjsu/synopsys/tc240c/tc240c.db\_WCCOM25 -min\_version /apps/toshiba/sjsu/synopsys/tc240c/tc240c.db\_WCCOM25

Loading db file '/apps/toshiba/sjsu/synopsys/tc240c/tc240c.db\_WCCOM25'

Error: Minimum version must be a different library. (TIM-98)

0

read\_verilog 271control.v

Loading db file '/apps/synopsys/SYNTH/libraries/syn/gtech.db'

Loading db file '/apps/synopsys/SYNTH/libraries/syn/standard.sldb'

Loading link library 'tc240c'

Warning: Function '=' leaked 1 allocations for 16 bytes. (EQN-21)

Loading link library 'gtech'

Loading verilog file '/home/pa/pate8552/271new/271control.v'

Detecting input file type automatically (-rtl or -netlist).

Running DC verilog reader

Reading with Presto HDL Compiler (equivalent to -rtl option).

Running PRESTO HDLC

Loading db file '/apps/synopsys/SYNTH/libraries/syn/dw\_foundation.sldb'

Warning: The following synthetic libraries should be added to

the list of link libraries:

'dw\_foundation.sldb'. (UISN-26)

Compiling source file /home/pa/pate8552/271new/271control.v

Warning: Little argument or return value checking implemented for system task or function '$time'. (VER-209)

Warning: /home/pa/pate8552/271new/271control.v:173: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:180: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:187: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:187: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:194: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:194: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:201: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:201: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:208: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:215: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:240: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:248: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:256: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:265: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:265: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:273: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:273: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:283: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:283: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:293: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:301: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:309: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:331: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:331: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:331: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:339: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:339: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:339: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:350: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:350: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:350: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:350: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:350: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:359: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:359: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:359: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:359: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:359: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:398: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:398: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:398: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:398: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:403: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:403: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:403: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:403: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:408: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:408: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:408: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:408: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:413: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:413: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:413: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:413: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:418: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:418: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:418: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:418: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:423: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:423: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:423: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:423: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:428: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:428: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:428: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:428: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:173: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:180: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:187: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:187: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:194: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:194: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:201: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:201: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:208: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:215: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:240: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:248: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:256: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:265: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:265: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:273: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:273: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:283: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:283: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:293: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:301: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:309: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:331: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:331: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:331: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:339: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:339: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:339: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:350: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:350: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:350: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:350: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:350: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:359: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:359: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:359: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:359: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:359: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:398: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:398: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:398: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:398: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:403: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:403: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:403: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:403: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:408: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:408: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:408: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:408: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:413: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:413: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:413: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:413: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:418: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:418: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:418: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:418: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:423: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:423: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:423: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:423: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:428: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:428: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:428: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:428: Invalid escape sequence '\0' in call to '$display'. (VER-941)

Warning: /home/pa/pate8552/271new/271control.v:75: 'OPRN' is being read, but does not appear in the sensitivity list of the block. (ELAB-292)

Warning: /home/pa/pate8552/271new/271control.v:75: 'cout' is being read, but does not appear in the sensitivity list of the block. (ELAB-292)

Statistics for case statements in always block at line 87 in file

'/home/pa/pate8552/271new/271control.v'

===============================================

| Line | full/ parallel |

===============================================

| 89 | no/auto |

===============================================

$display output: 00 ns FETCH STATE: Instruction is Fetched

$display output: 00 ns DECODE STATE: Instruction is Decoded

$display output: 00 ns Instruction : NOP

$display output: UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

$display output: 00 ns Instruction : JMP

$display output: UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

$display output: 00 ns Instruction : RDM, 2b;

?$display output: UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

$display output: 00 ns Instruction : WRM, 2b;

?$display output: UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

$display output: 00 ns Instruction : CPR, 2b, 2b;

??$display output: UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

$display output: 00 ns Instruction : ADD, 2b, 2b;

??$display output: UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

$display output: 00 ns Instruction : SUB, 2b, 2b;

??$display output: UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

$display output: 00 ns Instruction : LLS, 2h

?$display output: UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

$display output: 00 ns Instruction : LMS, 2h

?$display output: UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

$display output: 00 ns Instruction : CFR;

$display output: UPDATE STATE:PR Register is Incremented to Fetch the next Instruction.

$display output: $display output: 00 ns EXECUTE\_1 STATE: Executed NOP and No Operation is Perfomed

$display output: 00 ns EXECUTE\_1 STATE: Executed JMP and Jumped to Instruction stored at 8H Memory Location

?$display output: 00 ns EXECUTE\_1 STATE: Executing RDM...Setting AR = 8h to Memory Address...

?$display output: 00 ns EXECUTE\_1 STATE: Executing WRM...Setting AR = 8h to Memory Address...

?$display output: 00 ns EXECUTE\_1 STATE: Executed CPR and Copied the Contents of 2b to 2b register

??$display output: 00 ns EXECUTE\_1 STATE: Executing ADD...Providing ALU Opcodes 8h and 8h and Adding them...

??$display output: 00 ns EXECUTE\_1 STATE: Executing SUB...Providing ALU Opcodes 8h and 8h and Adding them...

??$display output: 00 ns EXECUTE\_1 STATE: Executed LLS and Stored data at LSBs of GR = 2h

?$display output: 00 ns EXECUTE\_1 STATE: Executed LMS and Stored data at MSBs of GR = 2h

?$display output: 00 ns EXECUTE\_1 STATE: Executed CFR and Stored data at LSBs of GR = 2h

?$display output: 00 ns EXECUTE\_1 STATE: INVALID INSTRUCTION $display output: 00 ns EXECUTE\_2 STATE: Read the Data 8h from Memory Location 8h and Stored into 2b Register

???$display output: 00 ns EXECUTE\_2 STATE: 2b Register Data 8h is Written at Memory Address 8h

???$display output: 00 ns EXECUTE\_2 STATE: Added the 2b Register Data 8h with 2b Register Data 8h and the Result is Stored into 2b Register

?????$display output: 00 ns EXECUTE\_2 STATE: Subtracted the 2b Register Data 8h from 2b Register Data 8h and the Result is Stored into 2b Register

?????$display output: $display output: AR[00]= 8h, DR[01]= 8h, GR[10]= 8h, PR[11]= 8h

????$display output: AR[00]= 8h, DR[01]= 8h, GR[10]= 8h, PR[11]= 8h

????$display output: AR[00]= 8h, DR[01]= 8h, GR[10]= 8h, PR[11]= 8h

????$display output: AR[00]= 8h, DR[01]= 8h, GR[10]= 8h, PR[11]= 8h

????$display output: AR[00]= 8h, DR[01]= 8h, GR[10]= 8h, PR[11]= 8h

????$display output: AR[00]= 8h, DR[01]= 8h, GR[10]= 8h, PR[11]= 8h

????$display output: AR[00]= 8h, DR[01]= 8h, GR[10]= 8h, PR[11]= 8h

????$display output:

Statistics for case statements in always block at line 105 in file

'/home/pa/pate8552/271new/271control.v'

===============================================

| Line | full/ parallel |

===============================================

| 135 | no/auto |

| 157 | auto/auto |

| 233 | auto/auto |

| 321 | auto/auto |

| 394 | auto/auto |

===============================================

Inferred memory devices in process

in routine SCALAR\_PROCESSOR line 80 in file

'/home/pa/pate8552/271new/271control.v'.

===============================================================================

| Register Name | Type | Width | Bus | MB | AR | AS | SR | SS | ST |

===============================================================================

| state\_reg | Flip-flop | 3 | Y | N | Y | N | N | N | N |

===============================================================================

Inferred memory devices in process

in routine SCALAR\_PROCESSOR line 87 in file

'/home/pa/pate8552/271new/271control.v'.

===========================================================================

| Register Name | Type | Width | Bus | MB | AR | AS | SR | SS | ST |

===========================================================================

| n\_state\_reg | Latch | 3 | Y | N | N | N | - | - | - |

===========================================================================

Inferred memory devices in process

in routine SCALAR\_PROCESSOR line 105 in file

'/home/pa/pate8552/271new/271control.v'.

===============================================================================

| Register Name | Type | Width | Bus | MB | AR | AS | SR | SS | ST |

===============================================================================

| FR\_reg | Flip-flop | 4 | Y | N | Y | N | N | N | N |

| OP2\_reg | Flip-flop | 8 | Y | N | Y | N | N | N | N |

| ARRAY\_reg | Flip-flop | 32 | N | N | Y | N | N | N | N |

| S\_reg | Flip-flop | 2 | Y | N | Y | N | N | N | N |

| rd\_reg | Flip-flop | 1 | N | N | Y | N | N | N | N |

| OP1\_reg | Flip-flop | 8 | Y | N | Y | N | N | N | N |

| addr\_reg | Flip-flop | 8 | Y | N | Y | N | N | N | N |

| wrt\_reg | Flip-flop | 1 | N | N | Y | N | N | N | N |

| OPRN\_reg | Flip-flop | 1 | N | N | Y | N | N | N | N |

| D\_reg | Flip-flop | 2 | Y | N | Y | N | N | N | N |

| opcode\_reg | Flip-flop | 4 | Y | N | Y | N | N | N | N |

| dat\_T\_reg | Flip-flop | 8 | Y | N | Y | N | N | N | N |

===============================================================================

Inferred tri-state devices in process

in routine SCALAR\_PROCESSOR line 55 in file

'/home/pa/pate8552/271new/271control.v'.

=================================================

| Register Name | Type | Width | MB |

=================================================

| dat\_tri | Tri-State Buffer | 8 | N |

=================================================

Statistics for MUX\_OPs

===============================================================

| block name/line | Inputs | Outputs | # sel inputs | MB |

===============================================================

| SCALAR\_PROCESSOR/258 | 4 | 8 | 2 | N |

| SCALAR\_PROCESSOR/266 | 4 | 8 | 2 | N |

===============================================================

Presto compilation completed successfully.

Current design is now '/home/pa/pate8552/271new/SCALAR\_PROCESSOR.db:SCALAR\_PROCESSOR'

Loaded 5 designs.

Current design is 'SCALAR\_PROCESSOR'.

SCALAR\_PROCESSOR ALU RCA\_8 add\_full add\_half

current\_design SCALAR\_PROCESSOR

Current design is 'SCALAR\_PROCESSOR'.

{SCALAR\_PROCESSOR}

link

Linking design 'SCALAR\_PROCESSOR'

Using the following designs and libraries:

--------------------------------------------------------------------------

tc240c (library) /apps/toshiba/sjsu/synopsys/tc240c/tc240c.db\_WCCOM25

1

check\_design

Warning: In design 'SCALAR\_PROCESSOR', cell 'C2376' does not drive any nets. (LINT-1)

Warning: In design 'SCALAR\_PROCESSOR', cell 'C2377' does not drive any nets. (LINT-1)

Warning: In design 'SCALAR\_PROCESSOR', cell 'C2378' does not drive any nets. (LINT-1)

Warning: In design 'SCALAR\_PROCESSOR', cell 'C2379' does not drive any nets. (LINT-1)

Warning: In design 'SCALAR\_PROCESSOR', cell 'C2380' does not drive any nets. (LINT-1)

Warning: In design 'SCALAR\_PROCESSOR', cell 'C2381' does not drive any nets. (LINT-1)

Warning: In design 'SCALAR\_PROCESSOR', cell 'C2382' does not drive any nets. (LINT-1)

Warning: In design 'SCALAR\_PROCESSOR', cell 'C2383' does not drive any nets. (LINT-1)

Warning: In design 'SCALAR\_PROCESSOR', cell 'C2384' does not drive any nets. (LINT-1)

Warning: Net 'dat[0]' has a single tri-state driver. (LINT-63)

Warning: Net 'dat[1]' has a single tri-state driver. (LINT-63)

Warning: Net 'dat[2]' has a single tri-state driver. (LINT-63)

Warning: Net 'dat[3]' has a single tri-state driver. (LINT-63)

Warning: Net 'dat[4]' has a single tri-state driver. (LINT-63)

Warning: Net 'dat[5]' has a single tri-state driver. (LINT-63)

Warning: Net 'dat[6]' has a single tri-state driver. (LINT-63)

Warning: Net 'dat[7]' has a single tri-state driver. (LINT-63)

Information: Design 'SCALAR\_PROCESSOR' has multiply instantiated designs. Use the '-multiple\_designs' switch for more information. (LINT-78)

1

create\_clock CLK -name CLK -period 4.0000000

Warning: Can't find object 'CLK' in design 'SCALAR\_PROCESSOR'. (UID-95)

Error: Value for list 'source\_objects' must have 1 elements. (CMD-036)

0

set\_propagated\_clock CLK

Warning: Can't find object 'CLK' in design 'SCALAR\_PROCESSOR'. (UID-95)

Error: Value for list 'object\_list' must have 1 elements. (CMD-036)

0

set\_clock\_uncertainty .25 CLK

Warning: Can't find object 'CLK' in design 'SCALAR\_PROCESSOR'. (UID-95)

Error: Value for list 'object\_list' must have 1 elements. (CMD-036)

0

set\_max\_delay 1.1 -from [all\_inputs]

1

set\_max\_delay 1.1 -to [all\_outputs]

1

#set\_max\_delay 2 -to sum

set\_load 160 sum

Warning: Can't find object 'sum' in design 'SCALAR\_PROCESSOR'. (UID-95)

Error: Value for list 'objects' must have 1 elements. (CMD-036)

0

set\_max\_area 500

1

compile -map\_effort high

Warning: The following synthetic libraries should be added to

the list of link libraries:

'dw\_foundation.sldb'. (UISN-26)

Information: Checking out the license 'DesignWare'. (SEC-104)

Information: Evaluating DesignWare library utilization. (UISN-27)

============================================================================

| DesignWare Building Block Library | Version | Available |

============================================================================

| Basic DW Building Blocks | C-2009.06-DWBB\_0912 | \* |

| Licensed DW Building Blocks | C-2009.06-DWBB\_0912 | \* |

============================================================================

Warning: Net 'dat[0]' has a single tri-state driver. (LINT-63)

Warning: Net 'dat[1]' has a single tri-state driver. (LINT-63)

Warning: Net 'dat[2]' has a single tri-state driver. (LINT-63)

Warning: Net 'dat[3]' has a single tri-state driver. (LINT-63)

Warning: Net 'dat[4]' has a single tri-state driver. (LINT-63)

Warning: Net 'dat[5]' has a single tri-state driver. (LINT-63)

Warning: Net 'dat[6]' has a single tri-state driver. (LINT-63)

Warning: Net 'dat[7]' has a single tri-state driver. (LINT-63)

Information: There are 9 potential problems in your design. Please run 'check\_design' for more information. (LINT-99)

Warning: Operating condition WCCOM25 set on design SCALAR\_PROCESSOR has different process,

voltage and temperatures parameters than the parameters at which target library

tc240c is characterized. Delays may be inaccurate as a result. (OPT-998)

Beginning Pass 1 Mapping

------------------------

Processing 'add\_half\_0'

Processing 'add\_full\_0'

Processing 'RCA\_8'

Processing 'ALU'

Processing 'SCALAR\_PROCESSOR'

Updating timing information

Information: Updating design information... (UID-85)

Beginning Implementation Selection

----------------------------------

Processing 'SCALAR\_PROCESSOR\_DW01\_inc\_0'

Beginning Mapping Optimizations (High effort)

-------------------------------

ELAPSED WORST NEG TOTAL NEG DESIGN

TIME AREA SLACK SLACK RULE COST ENDPOINT

--------- --------- --------- --------- --------- -------------------------

0:00:02 1597.0 0.04 0.8 0.0

0:00:02 1095.0 0.53 5.4 0.0

0:00:02 1099.5 0.22 2.6 0.0

0:00:02 1103.5 0.11 1.1 0.0

0:00:02 1112.5 0.11 0.9 0.0

0:00:02 1121.5 0.07 0.6 0.0

0:00:02 1119.5 0.06 0.6 0.0

0:00:02 1116.5 0.04 0.4 0.0

0:00:02 1111.5 0.04 0.3 0.0

0:00:02 1113.0 0.03 0.2 0.0

0:00:02 1113.5 0.03 0.2 0.0

0:00:02 1114.0 0.03 0.1 0.0

0:00:02 1114.5 0.03 0.1 0.0

0:00:03 1113.0 0.03 0.1 0.0

0:00:03 1115.5 0.03 0.1 0.0

0:00:03 1115.5 0.03 0.1 0.0

0:00:03 1115.5 0.03 0.1 0.0

0:00:03 1115.5 0.03 0.1 0.0

0:00:03 1115.5 0.03 0.1 0.0

0:00:03 1115.5 0.03 0.1 0.0

0:00:03 1115.5 0.03 0.1 0.0

Beginning Delay Optimization Phase

----------------------------------

ELAPSED WORST NEG TOTAL NEG DESIGN

TIME AREA SLACK SLACK RULE COST ENDPOINT

--------- --------- --------- --------- --------- -------------------------

0:00:03 1115.5 0.03 0.1 0.0

0:00:03 1113.0 0.00 0.0 0.0

Beginning Area-Recovery Phase (max\_area 500)

-----------------------------

ELAPSED WORST NEG TOTAL NEG DESIGN

TIME AREA SLACK SLACK RULE COST ENDPOINT

--------- --------- --------- --------- --------- -------------------------

0:00:03 1113.0 0.00 0.0 0.0

0:00:03 1113.0 0.00 0.0 0.0

0:00:03 1112.0 0.00 0.0 0.0

0:00:03 1112.0 0.00 0.0 0.0

0:00:03 1112.0 0.00 0.0 0.0

0:00:03 1112.0 0.00 0.0 0.0

0:00:03 1112.0 0.00 0.0 0.0

0:00:03 1112.0 0.00 0.0 0.0

0:00:03 1112.0 0.00 0.0 0.0

0:00:03 1112.0 0.00 0.0 0.0

0:00:03 1112.0 0.00 0.0 0.0

0:00:03 1112.0 0.00 0.0 0.0

0:00:03 1111.0 0.00 0.0 0.0

0:00:03 1111.0 0.00 0.0 0.0

0:00:03 1111.0 0.00 0.0 0.0

0:00:03 1111.0 0.00 0.0 0.0

0:00:03 1111.0 0.00 0.0 0.0

0:00:03 1111.0 0.00 0.0 0.0

0:00:03 1111.0 0.00 0.0 0.0

0:00:03 1110.5 0.00 0.0 0.0

0:00:03 1110.0 0.00 0.0 0.0

0:00:03 1109.5 0.00 0.0 0.0

0:00:03 1108.5 0.00 0.0 0.0

0:00:03 1108.0 0.00 0.0 0.0

0:00:03 1108.0 0.00 0.0 0.0

0:00:03 1108.0 0.00 0.0 0.0

0:00:03 1108.0 0.00 0.0 0.0

0:00:03 1108.0 0.00 0.0 0.0

0:00:03 1108.0 0.00 0.0 0.0

0:00:03 1108.0 0.00 0.0 0.0

0:00:03 1108.0 0.00 0.0 0.0

0:00:03 1108.0 0.00 0.0 0.0

Loading db file '/apps/toshiba/sjsu/synopsys/tc240c/tc240c.db\_WCCOM25'

Optimization Complete

---------------------

1

report\_cell

Information: Updating design information... (UID-85)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Report : cell

Design : SCALAR\_PROCESSOR

Version: C-2009.06-SP5

Date : Fri Dec 5 03:58:23 2014

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Attributes:

BO - reference allows boundary optimization

b - black box (unknown)

h - hierarchical

mo - map\_only

n - noncombinational

r - removable

so - sizing only

u - contains unmapped logic

Cell Reference Library Area Attributes

--------------------------------------------------------------------------------

ARRAY\_reg[0][0] CFD2QXL tc240c 5.000000 n, so

ARRAY\_reg[0][1] CFD2QXL tc240c 5.000000 n, so

ARRAY\_reg[0][2] CFD2QXL tc240c 5.000000 n, so

ARRAY\_reg[0][3] CFD2QXL tc240c 5.000000 n, so

ARRAY\_reg[0][4] CFD2QXL tc240c 5.000000 n, so

ARRAY\_reg[0][5] CFD2QXL tc240c 5.000000 n, so

ARRAY\_reg[0][6] CFD2QXL tc240c 5.000000 n, so

ARRAY\_reg[0][7] CFD2QXL tc240c 5.000000 n, so

ARRAY\_reg[1][0] CFD2QXL tc240c 5.000000 n, so

ARRAY\_reg[1][1] CFD2QXL tc240c 5.000000 n, so

ARRAY\_reg[1][2] CFD2QXL tc240c 5.000000 n, so

ARRAY\_reg[1][3] CFD2QXL tc240c 5.000000 n, so

ARRAY\_reg[1][4] CFD2QXL tc240c 5.000000 n, so

ARRAY\_reg[1][5] CFD2QXL tc240c 5.000000 n, so

ARRAY\_reg[1][6] CFD2QXL tc240c 5.000000 n, so

ARRAY\_reg[1][7] CFD2QXL tc240c 5.000000 n, so

ARRAY\_reg[2][0] CFD2QXL tc240c 5.000000 n, so

ARRAY\_reg[2][1] CFD2QXL tc240c 5.000000 n, so

ARRAY\_reg[2][2] CFD2QXL tc240c 5.000000 n, so

ARRAY\_reg[2][3] CFD2QXL tc240c 5.000000 n, so

ARRAY\_reg[2][4] CFD2QX2 tc240c 7.500000 n, so

ARRAY\_reg[2][5] CFD2QX2 tc240c 7.500000 n, so

ARRAY\_reg[2][6] CFD2QXL tc240c 5.000000 n, so

ARRAY\_reg[2][7] CFD2QXL tc240c 5.000000 n, so

ARRAY\_reg[3][0] CFD2QXL tc240c 5.000000 n, so

ARRAY\_reg[3][1] CFD2QXL tc240c 5.000000 n, so

ARRAY\_reg[3][2] CFD2QXL tc240c 5.000000 n, so

ARRAY\_reg[3][3] CFD2QXL tc240c 5.000000 n, so

ARRAY\_reg[3][4] CFD2QXL tc240c 5.000000 n, so

ARRAY\_reg[3][5] CFD2QXL tc240c 5.000000 n, so

ARRAY\_reg[3][6] CFD2QXL tc240c 5.000000 n, so

ARRAY\_reg[3][7] CFD2QXL tc240c 5.000000 n, so

D\_reg[0] CFD2QXL tc240c 5.000000 n, so

D\_reg[1] CFD2QXL tc240c 5.000000 n, so

FR\_reg[0] CFD2QXL tc240c 5.000000 n, so

FR\_reg[1] CFD2QXL tc240c 5.000000 n, so

FR\_reg[2] CFD2QXL tc240c 5.000000 n, so

FR\_reg[3] CFD2QXL tc240c 5.000000 n, so

OP1\_reg[0] CFD2QXL tc240c 5.000000 n, so

OP1\_reg[1] CFD2QXL tc240c 5.000000 n, so

OP1\_reg[2] CFD2QXL tc240c 5.000000 n, so

OP1\_reg[3] CFD2QXL tc240c 5.000000 n, so

OP1\_reg[4] CFD2QXL tc240c 5.000000 n, so

OP1\_reg[5] CFD2QXL tc240c 5.000000 n, so

OP1\_reg[6] CFD2QXL tc240c 5.000000 n, so

OP1\_reg[7] CFD2QXL tc240c 5.000000 n, so

OP2\_reg[0] CFD2QXL tc240c 5.000000 n, so

OP2\_reg[1] CFD2QXL tc240c 5.000000 n, so

OP2\_reg[2] CFD2QXL tc240c 5.000000 n, so

OP2\_reg[3] CFD2QXL tc240c 5.000000 n, so

OP2\_reg[4] CFD2QXL tc240c 5.000000 n, so

OP2\_reg[5] CFD2QXL tc240c 5.000000 n, so

OP2\_reg[6] CFD2QXL tc240c 5.000000 n, so

OP2\_reg[7] CFD2QXL tc240c 5.000000 n, so

OPRN\_reg CFD2QXL tc240c 5.000000 n, so

S\_reg[0] CFD2QXL tc240c 5.000000 n, so

S\_reg[1] CFD2QXL tc240c 5.000000 n, so

U74 CAN4X1 tc240c 2.000000

U92 CND2IX1 tc240c 1.500000

U124 CAOR2X1 tc240c 2.500000

U126 CAOR2X1 tc240c 2.500000

U127 CAOR2X1 tc240c 2.500000

U130 CAOR2X1 tc240c 2.500000

U135 CAOR2X1 tc240c 2.500000

U139 CAOR2X1 tc240c 2.500000

U143 CAOR2X1 tc240c 2.500000

U147 CAOR2X1 tc240c 2.500000

U151 CAOR2X1 tc240c 2.500000

U155 CAOR2X1 tc240c 2.500000

U159 CAOR2X1 tc240c 2.500000

U163 CAOR2X1 tc240c 2.500000

U170 CAN3X2 tc240c 2.000000

U172 CAOR2X1 tc240c 2.500000

U173 CAOR2X1 tc240c 2.500000

U174 CAOR2X1 tc240c 2.500000

U175 CAOR2X1 tc240c 2.500000

U177 COR8X1 tc240c 4.000000

U178 COR2X1 tc240c 1.500000

U180 CAOR2X1 tc240c 2.500000

U184 CAOR2X1 tc240c 2.500000

U188 CAOR2X1 tc240c 2.500000

U189 CAOR2X1 tc240c 2.500000

U190 CAOR2X1 tc240c 2.500000

U191 CAOR2X1 tc240c 2.500000

U192 CAOR2X1 tc240c 2.500000

U193 CAOR2X1 tc240c 2.500000

U194 CAOR2X1 tc240c 2.500000

U195 CAOR2X1 tc240c 2.500000

U196 CAOR2X1 tc240c 2.500000

U197 CAOR2X1 tc240c 2.500000

U198 CAOR2X1 tc240c 2.500000

U199 CAOR2X1 tc240c 2.500000

U200 CAOR2X1 tc240c 2.500000

U201 CAOR2X1 tc240c 2.500000

U202 CAOR2X1 tc240c 2.500000

U203 CAOR2X1 tc240c 2.500000

U204 CAOR2X1 tc240c 2.500000

U209 CAOR2X1 tc240c 2.500000

U210 CAOR2X1 tc240c 2.500000

U211 CAOR2X1 tc240c 2.500000

U212 CAOR2X1 tc240c 2.500000

U219 CAOR2X1 tc240c 2.500000

U223 CAOR1X1 tc240c 2.000000

U245 COND1XL tc240c 1.500000

U246 COND1XL tc240c 1.500000

U247 COAN1X1 tc240c 2.000000

U248 CND3XL tc240c 1.500000

U249 CND5X1 tc240c 3.500000

U250 COAN1X1 tc240c 2.000000

U251 CND3XL tc240c 1.500000

U252 COAN1X1 tc240c 2.000000

U253 COND1XL tc240c 1.500000

U254 CANR2X1 tc240c 2.000000

U255 CANR2X1 tc240c 2.000000

U256 CANR2X2 tc240c 4.000000

U257 COR2X1 tc240c 1.500000

U258 CND2X1 tc240c 1.000000

U259 CANR2X1 tc240c 2.000000

U260 COR2X1 tc240c 1.500000

U261 CND2X1 tc240c 1.000000

U262 CANR2X1 tc240c 2.000000

U263 CND2X1 tc240c 1.000000

U264 CND2XL tc240c 1.000000

U265 CND2X1 tc240c 1.000000

U266 COAN1X1 tc240c 2.000000

U267 CND2X1 tc240c 1.000000

U268 CND2X1 tc240c 1.000000

U269 CND2XL tc240c 1.000000

U270 CND2X1 tc240c 1.000000

U271 CND2X1 tc240c 1.000000

U272 CND2XL tc240c 1.000000

U273 CND2X1 tc240c 1.000000

U274 CND2X1 tc240c 1.000000

U275 CND2XL tc240c 1.000000

U276 CND2X1 tc240c 1.000000

U277 COAN1X1 tc240c 2.000000

U278 CND2X1 tc240c 1.000000

U279 CANR2X1 tc240c 2.000000

U280 CND2X1 tc240c 1.000000

U281 CANR2X1 tc240c 2.000000

U282 COAN1X1 tc240c 2.000000

U283 CND2X1 tc240c 1.000000

U284 COAN1X1 tc240c 2.000000

U285 CND2X1 tc240c 1.000000

U286 COND1XL tc240c 1.500000

U287 CAN2X1 tc240c 1.500000

U288 COR2X1 tc240c 1.500000

U289 COR2X1 tc240c 1.500000

U290 CAN2X1 tc240c 1.500000

U291 CANR2X1 tc240c 2.000000

U292 CANR2X1 tc240c 2.000000

U293 CND2X1 tc240c 1.000000

U294 CAN2X1 tc240c 1.500000

U295 COR2X1 tc240c 1.500000

U296 CANR2X1 tc240c 2.000000

U297 CAN2X1 tc240c 1.500000

U298 COR2X1 tc240c 1.500000

U299 COAN1X1 tc240c 2.000000

U300 CND2X1 tc240c 1.000000

U301 CND2X1 tc240c 1.000000

U302 COR2X1 tc240c 1.500000

U303 CIVXL tc240c 1.000000

U304 CAN2X1 tc240c 1.500000

U305 COR2X1 tc240c 1.500000

U306 COAN1X1 tc240c 2.000000

U307 CND2X1 tc240c 1.000000

U308 CND2X1 tc240c 1.000000

U309 COR2X1 tc240c 1.500000

U310 CIVXL tc240c 1.000000

U311 CNR3XL tc240c 1.500000

U312 CNR2IX4 tc240c 3.500000

U313 CNR2X1 tc240c 1.000000

U314 CIVX2 tc240c 1.000000

U315 CND3XL tc240c 1.500000

U316 CANR4CX1 tc240c 2.000000

U317 CNR2X1 tc240c 1.000000

U318 CNR2X1 tc240c 1.000000

U319 CIVX2 tc240c 1.000000

U320 CAN2X1 tc240c 1.500000

U321 CANR11X1 tc240c 2.000000

U322 CANR1XL tc240c 1.500000

U323 CND4X1 tc240c 2.000000

U324 CAN2X1 tc240c 1.500000

U325 CND3XL tc240c 1.500000

U326 CANR1XL tc240c 1.500000

U327 CND3XL tc240c 1.500000

U328 CNR2X1 tc240c 1.000000

U329 CIVX2 tc240c 1.000000

U330 CIVX2 tc240c 1.000000

U331 CIVX2 tc240c 1.000000

U332 CIVX2 tc240c 1.000000

U333 CANR2X1 tc240c 2.000000

U334 CANR2X1 tc240c 2.000000

U335 CND2X1 tc240c 1.000000

U336 CANR2X1 tc240c 2.000000

U337 CND2X1 tc240c 1.000000

U338 CND2X1 tc240c 1.000000

U339 CANR2X1 tc240c 2.000000

U340 CND2X1 tc240c 1.000000

U341 CND2X1 tc240c 1.000000

U342 COND4CX1 tc240c 2.000000

U343 CND2X1 tc240c 1.000000

U344 CANR2X1 tc240c 2.000000

U345 COND4CX1 tc240c 2.000000

U346 CND2X1 tc240c 1.000000

U347 CANR2X1 tc240c 2.000000

U348 COND4CX1 tc240c 2.000000

U349 CND2X1 tc240c 1.000000

U350 CANR2X1 tc240c 2.000000

U351 COND4CX1 tc240c 2.000000

U352 CND2X1 tc240c 1.000000

U353 CANR2X1 tc240c 2.000000

U354 CAN4X1 tc240c 2.000000

U355 COND11X1 tc240c 2.000000

U356 CANR11X1 tc240c 2.000000

U357 COND4CX1 tc240c 2.000000

U358 CNR3XL tc240c 1.500000

U359 CNR2IX1 tc240c 1.500000

U360 CNR4X1 tc240c 2.000000

U361 CNR3XL tc240c 1.500000

U362 CNR3XL tc240c 1.500000

U363 CNR3XL tc240c 1.500000

U364 CNR3XL tc240c 1.500000

U365 CNR2X1 tc240c 1.000000

U366 CMX4XL tc240c 5.500000 mo

U367 CMX4XL tc240c 5.500000 mo

U368 CMX4XL tc240c 5.500000 mo

U369 CMX4XL tc240c 5.500000 mo

U370 CANR3X1 tc240c 2.000000

U371 COND11X1 tc240c 2.000000

U372 CND3XL tc240c 1.500000

U373 CANR2X1 tc240c 2.000000

U374 CANR2X1 tc240c 2.000000

U375 CANR2X1 tc240c 2.000000

U376 CNR2X1 tc240c 1.000000

U377 CMX4XL tc240c 5.500000 mo

U378 CMX4XL tc240c 5.500000 mo

U379 CMX4XL tc240c 5.500000 mo

U380 CMX4XL tc240c 5.500000 mo

U381 COND2X1 tc240c 2.000000

U382 CENX1 tc240c 2.000000

U383 CND3XL tc240c 1.500000

U384 CND2X1 tc240c 1.000000

U385 CND3XL tc240c 1.500000

U386 COND4CX1 tc240c 2.000000

U387 CND2X1 tc240c 1.000000

U388 COND1XL tc240c 1.500000

U389 CANR2X1 tc240c 2.000000

U390 COND1XL tc240c 1.500000

U391 CANR2X1 tc240c 2.000000

U392 COND1XL tc240c 1.500000

U393 CANR2X1 tc240c 2.000000

U394 COND1XL tc240c 1.500000

U395 CANR2X1 tc240c 2.000000

U396 COND1XL tc240c 1.500000

U397 CANR2X1 tc240c 2.000000

U398 COND1XL tc240c 1.500000

U399 CND2X1 tc240c 1.000000

U400 CAOR2XL tc240c 2.500000

U401 CAOR2XL tc240c 2.500000

U402 CAOR2XL tc240c 2.500000

U403 CAOR2XL tc240c 2.500000

U404 CAOR2XL tc240c 2.500000

U405 CAOR2XL tc240c 2.500000

U406 CAOR2XL tc240c 2.500000

U407 CAOR2XL tc240c 2.500000

U408 COND1XL tc240c 1.500000

U409 CND2X1 tc240c 1.000000

U410 COND1XL tc240c 1.500000

U411 CND2X1 tc240c 1.000000

U412 COND1XL tc240c 1.500000

U413 CND2X1 tc240c 1.000000

U414 CMX4XL tc240c 5.500000 mo

U415 CMX4XL tc240c 5.500000 mo

U416 CMX4XL tc240c 5.500000 mo

U417 CMX4XL tc240c 5.500000 mo

U418 CMX4XL tc240c 5.500000 mo

U419 CMX4XL tc240c 5.500000 mo

U420 CMX4XL tc240c 5.500000 mo

U421 CMX4XL tc240c 5.500000 mo

U422 COND2X1 tc240c 2.000000

U423 COND2X1 tc240c 2.000000

U424 COND2X1 tc240c 2.000000

U425 COND2X1 tc240c 2.000000

U426 CANR2XL tc240c 2.000000

U427 CANR2XL tc240c 2.000000

U428 CANR2XL tc240c 2.000000

U429 CANR2XL tc240c 2.000000

U430 CANR2XL tc240c 2.000000

U431 CANR2XL tc240c 2.000000

U432 CANR2XL tc240c 2.000000

U433 COND1XL tc240c 1.500000

U434 COND1XL tc240c 1.500000

U435 COND1XL tc240c 1.500000

U436 COND1XL tc240c 1.500000

U437 COND1XL tc240c 1.500000

U438 COND1XL tc240c 1.500000

U439 COND1XL tc240c 1.500000

U440 COND1XL tc240c 1.500000

U441 CANR2X1 tc240c 2.000000

U442 COND3X1 tc240c 2.000000

U443 CANR2X1 tc240c 2.000000

U444 COND3X1 tc240c 2.000000

U445 CANR2X1 tc240c 2.000000

U446 CANR2X1 tc240c 2.000000

U447 CIVX3 tc240c 1.500000

U448 CIVX2 tc240c 1.000000

U449 CIVX2 tc240c 1.000000

U450 CIVX2 tc240c 1.000000

U451 CIVX2 tc240c 1.000000

U452 CIVX2 tc240c 1.000000

U453 CIVX2 tc240c 1.000000

U454 CIVX2 tc240c 1.000000

U455 CIVX2 tc240c 1.000000

U456 CIVX2 tc240c 1.000000

U457 CIVX2 tc240c 1.000000

U458 CIVX2 tc240c 1.000000

U459 CIVX2 tc240c 1.000000

U460 CIVX2 tc240c 1.000000

U461 CIVX2 tc240c 1.000000

U462 CIVX2 tc240c 1.000000

U463 CIVX2 tc240c 1.000000

U464 CIVX2 tc240c 1.000000

U465 CIVX2 tc240c 1.000000

U466 CIVX2 tc240c 1.000000

U467 CIVX2 tc240c 1.000000

U468 CIVX2 tc240c 1.000000

U469 CIVX2 tc240c 1.000000

U470 CIVX2 tc240c 1.000000

U471 CIVX2 tc240c 1.000000

U472 CIVX2 tc240c 1.000000

U473 CIVX2 tc240c 1.000000

U474 CIVX2 tc240c 1.000000

U475 CIVX2 tc240c 1.000000

U476 CIVX2 tc240c 1.000000

U477 CIVX2 tc240c 1.000000

U478 CIVX2 tc240c 1.000000

U479 CIVX2 tc240c 1.000000

U480 CIVX2 tc240c 1.000000

U481 CIVX2 tc240c 1.000000

U482 CIVX2 tc240c 1.000000

U483 CIVX2 tc240c 1.000000

U484 CIVX2 tc240c 1.000000

U485 CIVX2 tc240c 1.000000

U486 CIVX2 tc240c 1.000000

U487 CIVX2 tc240c 1.000000

U488 CIVX2 tc240c 1.000000

U489 CIVX2 tc240c 1.000000

U490 CIVX2 tc240c 1.000000

U491 CIVX2 tc240c 1.000000

addr\_reg[0] CFD2QXL tc240c 5.000000 n, so

addr\_reg[1] CFD2QXL tc240c 5.000000 n, so

addr\_reg[2] CFD2QXL tc240c 5.000000 n, so

addr\_reg[3] CFD2QXL tc240c 5.000000 n, so

addr\_reg[4] CFD2QXL tc240c 5.000000 n, so

addr\_reg[5] CFD2QXL tc240c 5.000000 n, so

addr\_reg[6] CFD2QXL tc240c 5.000000 n, so

addr\_reg[7] CFD2QXL tc240c 5.000000 n, so

alu23 ALU 86.000000 h

dat\_T\_reg[0] CFD2QXL tc240c 5.000000 n, so

dat\_T\_reg[1] CFD2QXL tc240c 5.000000 n, so

dat\_T\_reg[2] CFD2QXL tc240c 5.000000 n, so

dat\_T\_reg[3] CFD2QXL tc240c 5.000000 n, so

dat\_T\_reg[4] CFD2QXL tc240c 5.000000 n, so

dat\_T\_reg[5] CFD2QXL tc240c 5.000000 n, so

dat\_T\_reg[6] CFD2QXL tc240c 5.000000 n, so

dat\_T\_reg[7] CFD2QXL tc240c 5.000000 n, so

dat\_tri[0] CTSX2 tc240c 3.000000 n

dat\_tri[1] CTSX2 tc240c 3.000000 n

dat\_tri[2] CTSX2 tc240c 3.000000 n

dat\_tri[3] CTSX2 tc240c 3.000000 n

dat\_tri[4] CTSX2 tc240c 3.000000 n

dat\_tri[5] CTSX2 tc240c 3.000000 n

dat\_tri[6] CTSX2 tc240c 3.000000 n

dat\_tri[7] CTSX2 tc240c 3.000000 n

n\_state\_reg[0] CLDP1QXL tc240c 2.500000 n

n\_state\_reg[1] CLDP1QXL tc240c 2.500000 n

n\_state\_reg[2] CLDP1QXL tc240c 2.500000 n

opcode\_reg[0] CFD2QXL tc240c 5.000000 n, so

opcode\_reg[1] CFD2QXL tc240c 5.000000 n, so

opcode\_reg[2] CFD2QXL tc240c 5.000000 n, so

opcode\_reg[3] CFD2QXL tc240c 5.000000 n, so

r330 SCALAR\_PROCESSOR\_DW01\_inc\_0 24.000000 BO, h

rd\_reg CFD2QX1 tc240c 7.000000 n, so

state\_reg[0] CFD2QXL tc240c 5.000000 n, so

state\_reg[1] CFD2QXL tc240c 5.000000 n, so

state\_reg[2] CFD2QXL tc240c 5.000000 n, so

wrt\_reg CFD2QX1 tc240c 7.000000 n, so

--------------------------------------------------------------------------------

Total 388 cells 1108.000000

1

report\_net

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Report : net

Design : SCALAR\_PROCESSOR

Version: C-2009.06-SP5

Date : Fri Dec 5 03:58:23 2014

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Operating Conditions: WCCOM25 Library: tc240c

Wire Load Model Mode: top

Net Fanout Fanin Load Resistance Pins Attributes

--------------------------------------------------------------------------------

ALU\_OUT[0] 3 1 19.74 0.00 4

ALU\_OUT[1] 3 1 25.84 0.00 4

ALU\_OUT[2] 3 1 21.20 0.00 4

ALU\_OUT[3] 3 1 21.23 0.00 4

ALU\_OUT[4] 4 1 29.83 0.00 5

ALU\_OUT[5] 4 1 28.77 0.00 5

ALU\_OUT[6] 4 1 29.27 0.00 5

ALU\_OUT[7] 4 1 39.54 0.00 5

ARRAY[0][0] 5 1 42.11 0.00 6

ARRAY[0][1] 5 1 42.11 0.00 6

ARRAY[0][2] 5 1 42.11 0.00 6

ARRAY[0][3] 5 1 42.11 0.00 6

ARRAY[0][4] 5 1 42.11 0.00 6

ARRAY[0][5] 5 1 42.11 0.00 6

ARRAY[0][6] 5 1 42.11 0.00 6

ARRAY[0][7] 5 1 42.11 0.00 6

ARRAY[1][0] 3 1 17.66 0.00 4

ARRAY[1][1] 3 1 16.39 0.00 4

ARRAY[1][2] 3 1 17.66 0.00 4

ARRAY[1][3] 3 1 17.66 0.00 4

ARRAY[1][4] 3 1 16.39 0.00 4

ARRAY[1][5] 3 1 16.39 0.00 4

ARRAY[1][6] 3 1 17.66 0.00 4

ARRAY[1][7] 3 1 16.39 0.00 4

ARRAY[2][0] 3 1 18.14 0.00 4

ARRAY[2][1] 3 1 18.14 0.00 4

ARRAY[2][2] 3 1 18.14 0.00 4

ARRAY[2][3] 3 1 18.14 0.00 4

ARRAY[2][4] 3 1 16.49 0.00 4

ARRAY[2][5] 3 1 16.49 0.00 4

ARRAY[2][6] 3 1 16.49 0.00 4

ARRAY[2][7] 3 1 16.49 0.00 4

ARRAY[3][0] 6 1 69.17 0.00 7

ARRAY[3][1] 5 1 52.27 0.00 6

ARRAY[3][2] 5 1 53.08 0.00 6

ARRAY[3][3] 5 1 52.27 0.00 6

ARRAY[3][4] 5 1 52.27 0.00 6

ARRAY[3][5] 5 1 52.27 0.00 6

ARRAY[3][6] 5 1 52.27 0.00 6

ARRAY[3][7] 5 1 53.46 0.00 6

FR[0] 2 1 16.22 0.00 3

FR[1] 2 1 25.70 0.00 3

FR[2] 2 1 16.48 0.00 3

FR[3] 2 1 25.70 0.00 3

N64 13 1 147.32 0.00 14

N65 13 1 105.26 0.00 14

N66 11 1 140.09 0.00 12

N67 11 1 97.91 0.00 12

N92 1 1 2.61 0.00 2

N93 1 1 2.61 0.00 2

N94 3 1 7.76 0.00 4

N95 1 1 2.61 0.00 2

N99 2 1 14.51 0.00 3

N100 2 1 14.51 0.00 3

N101 2 1 14.51 0.00 3

N102 2 1 14.51 0.00 3

N103 2 1 14.90 0.00 3

N104 2 1 14.90 0.00 3

N105 2 1 14.90 0.00 3

N106 2 1 14.90 0.00 3

N107 2 1 24.12 0.00 3

N108 2 1 24.12 0.00 3

N109 2 1 24.12 0.00 3

N110 2 1 24.12 0.00 3

N111 2 1 24.12 0.00 3

N112 2 1 24.12 0.00 3

N113 2 1 24.12 0.00 3

N114 2 1 24.12 0.00 3

N245 1 1 8.79 0.00 2

N246 1 1 8.79 0.00 2

N247 1 1 8.79 0.00 2

N248 1 1 8.79 0.00 2

N249 1 1 8.79 0.00 2

N250 1 1 8.79 0.00 2

N251 1 1 8.79 0.00 2

N253 1 1 8.79 0.00 2

N470 1 1 7.50 0.00 2

OP1[0] 3 1 32.80 0.00 4

OP1[1] 3 1 32.80 0.00 4

OP1[2] 3 1 32.80 0.00 4

OP1[3] 3 1 32.80 0.00 4

OP1[4] 3 1 32.80 0.00 4

OP1[5] 3 1 32.80 0.00 4

OP1[6] 3 1 32.80 0.00 4

OP1[7] 3 1 32.80 0.00 4

OP2[0] 2 1 24.73 0.00 3

OP2[1] 2 1 24.73 0.00 3

OP2[2] 2 1 24.73 0.00 3

OP2[3] 2 1 24.73 0.00 3

OP2[4] 2 1 24.73 0.00 3

OP2[5] 2 1 24.73 0.00 3

OP2[6] 2 1 24.73 0.00 3

OP2[7] 2 1 24.73 0.00 3

OPRN 12 1 196.89 0.00 13

addr[0] 2 1 10.06 0.00 3

addr[1] 2 1 10.06 0.00 3

addr[2] 2 1 10.06 0.00 3

addr[3] 2 1 10.06 0.00 3

addr[4] 2 1 10.06 0.00 3

addr[5] 2 1 10.06 0.00 3

addr[6] 2 1 10.06 0.00 3

addr[7] 2 1 10.06 0.00 3

clk 82 1 240.96 0.00 83

cout 2 1 36.91 0.00 3

dat[0] 4 2 29.37 0.00 5

dat[1] 4 2 34.13 0.00 5

dat[2] 4 2 29.37 0.00 5

dat[3] 4 2 29.37 0.00 5

dat[4] 5 2 38.07 0.00 6

dat[5] 5 2 37.93 0.00 6

dat[6] 5 2 38.07 0.00 6

dat[7] 4 2 40.00 0.00 5

dat\_T[0] 2 1 22.82 0.00 3

dat\_T[1] 2 1 22.82 0.00 3

dat\_T[2] 2 1 22.82 0.00 3

dat\_T[3] 2 1 22.82 0.00 3

dat\_T[4] 2 1 22.82 0.00 3

dat\_T[5] 2 1 22.82 0.00 3

dat\_T[6] 2 1 22.82 0.00 3

dat\_T[7] 2 1 22.82 0.00 3

n18 9 1 49.72 0.00 10

n19 9 1 47.21 0.00 10

n21 1 1 3.99 0.00 2

n22 8 1 74.66 0.00 9

n23 9 1 84.43 0.00 10

n25 1 1 3.99 0.00 2

n27 1 1 3.99 0.00 2

n29 1 1 3.99 0.00 2

n31 1 1 3.99 0.00 2

n33 1 1 3.99 0.00 2

n35 1 1 3.99 0.00 2

n37 1 1 3.99 0.00 2

n39 2 1 11.13 0.00 3

n40 10 1 78.97 0.00 11

n42 8 1 56.80 0.00 9

n43 1 1 7.89 0.00 2

n44 1 1 10.14 0.00 2

n46 10 1 99.92 0.00 11

n48 8 1 81.13 0.00 9

n50 1 1 7.89 0.00 2

n51 1 1 10.14 0.00 2

n54 1 1 9.26 0.00 2

n55 1 1 10.06 0.00 2

n58 1 1 9.26 0.00 2

n59 1 1 10.06 0.00 2

n62 1 1 7.89 0.00 2

n63 1 1 10.14 0.00 2

n64 2 1 14.06 0.00 3

n66 1 1 7.89 0.00 2

n68 2 1 25.79 0.00 3

n70 1 1 7.89 0.00 2

n71 1 1 10.14 0.00 2

n72 2 1 14.06 0.00 3

n74 1 1 7.89 0.00 2

n76 2 1 25.79 0.00 3

n79 3 1 16.42 0.00 4

n81 4 1 20.33 0.00 5

n83 1 1 9.92 0.00 2

n84 8 1 62.82 0.00 9

n85 1 1 10.05 0.00 2

n86 1 1 7.40 0.00 2

n88 3 1 25.87 0.00 4

n89 3 1 36.50 0.00 4

n90 9 1 78.42 0.00 10

n91 1 1 6.65 0.00 2

n93 4 1 36.32 0.00 5

n94 13 1 82.35 0.00 14

n96 2 1 11.89 0.00 3

n97 1 1 6.65 0.00 2

n100 1 1 3.99 0.00 2

n101 1 1 6.65 0.00 2

n105 1 1 7.89 0.00 2

n106 1 1 6.65 0.00 2

n111 1 1 7.89 0.00 2

n113 2 1 20.25 0.00 3

n114 1 1 9.27 0.00 2

n116 8 1 77.24 0.00 9

n117 1 1 10.00 0.00 2

n119 1 1 10.14 0.00 2

n120 5 1 46.20 0.00 6

n121 5 1 48.79 0.00 6

n122 1 1 9.27 0.00 2

n124 1 1 10.00 0.00 2

n126 1 1 3.99 0.00 2

n127 1 1 9.27 0.00 2

n129 1 1 10.00 0.00 2

n131 1 1 10.14 0.00 2

n132 1 1 9.27 0.00 2

n134 1 1 10.00 0.00 2

n136 1 1 3.99 0.00 2

n137 3 1 17.72 0.00 4

n138 4 1 27.72 0.00 5

n140 9 1 60.06 0.00 10

n141 2 1 13.47 0.00 3

n142 2 1 11.07 0.00 3

n143 2 1 13.47 0.00 3

n144 2 1 13.47 0.00 3

n145 2 1 11.07 0.00 3

n146 2 1 11.07 0.00 3

n147 2 1 13.86 0.00 3

n148 2 1 11.07 0.00 3

n150 9 1 73.97 0.00 10

n151 20 1 167.12 0.00 21

n154 6 1 55.49 0.00 7

n156 1 1 3.99 0.00 2

n157 1 1 6.64 0.00 2

n159 1 1 9.97 0.00 2

n161 2 1 11.93 0.00 3

n162 18 1 148.96 0.00 19

n166 1 1 9.92 0.00 2

n167 3 1 20.59 0.00 4

n168 2 1 10.45 0.00 3

n170 1 1 9.17 0.00 2

n171 1 1 10.00 0.00 2

n172 2 1 10.47 0.00 3

n174 8 1 131.77 0.00 9

n175 1 1 2.73 0.00 2

n176 1 1 2.73 0.00 2

n177 1 1 2.73 0.00 2

n178 1 1 2.73 0.00 2

n179 1 1 2.73 0.00 2

n180 1 1 2.73 0.00 2

n181 1 1 2.73 0.00 2

n182 1 1 2.73 0.00 2

n184 1 1 2.73 0.00 2

n185 1 1 2.73 0.00 2

n186 1 1 2.73 0.00 2

n187 1 1 2.73 0.00 2

n188 1 1 2.73 0.00 2

n189 1 1 2.73 0.00 2

n190 1 1 2.73 0.00 2

n191 1 1 2.73 0.00 2

n192 1 1 2.73 0.00 2

n193 1 1 2.73 0.00 2

n194 1 1 2.73 0.00 2

n195 1 1 2.73 0.00 2

n196 1 1 2.73 0.00 2

n197 1 1 2.73 0.00 2

n198 1 1 2.73 0.00 2

n199 1 1 2.73 0.00 2

n200 1 1 2.73 0.00 2

n201 1 1 2.73 0.00 2

n202 1 1 7.78 0.00 2

n203 1 1 7.78 0.00 2

n204 1 1 2.73 0.00 2

n205 1 1 2.73 0.00 2

n206 1 1 2.73 0.00 2

n207 1 1 2.73 0.00 2

n208 1 1 2.73 0.00 2

n209 1 1 2.73 0.00 2

n210 1 1 2.73 0.00 2

n211 1 1 2.73 0.00 2

n212 1 1 2.73 0.00 2

n213 1 1 2.73 0.00 2

n214 1 1 2.73 0.00 2

n215 1 1 2.73 0.00 2

n216 1 1 2.73 0.00 2

n217 1 1 2.73 0.00 2

n218 1 1 2.73 0.00 2

n219 1 1 2.73 0.00 2

n220 1 1 2.73 0.00 2

n221 1 1 2.73 0.00 2

n222 1 1 2.73 0.00 2

n223 1 1 2.73 0.00 2

n224 1 1 2.73 0.00 2

n225 1 1 2.73 0.00 2

n226 1 1 2.73 0.00 2

n227 1 1 2.73 0.00 2

n228 1 1 2.73 0.00 2

n229 1 1 2.73 0.00 2

n230 1 1 2.73 0.00 2

n231 1 1 2.73 0.00 2

n232 1 1 2.73 0.00 2

n233 1 1 2.73 0.00 2

n234 1 1 2.73 0.00 2

n235 1 1 2.73 0.00 2

n236 1 1 2.73 0.00 2

n237 1 1 2.73 0.00 2

n238 1 1 2.73 0.00 2

n239 1 1 2.73 0.00 2

n240 1 1 2.73 0.00 2

n241 1 1 2.73 0.00 2

n242 1 1 2.73 0.00 2

n243 1 1 2.73 0.00 2

n244 1 1 2.73 0.00 2

n245 1 1 2.73 0.00 2

n246 1 1 2.73 0.00 2

n247 1 1 2.73 0.00 2

n248 1 1 2.73 0.00 2

n249 1 1 2.73 0.00 2

n250 1 1 2.73 0.00 2

n251 1 1 2.73 0.00 2

n252 1 1 2.73 0.00 2

n253 1 1 3.66 0.00 2

n254 1 1 3.66 0.00 2

n255 1 1 8.89 0.00 2

n256 1 1 9.38 0.00 2

n257 1 1 10.14 0.00 2

n258 1 1 8.79 0.00 2

n259 1 1 8.79 0.00 2

n260 1 1 10.14 0.00 2

n261 1 1 8.79 0.00 2

n262 1 1 10.14 0.00 2

n263 1 1 8.79 0.00 2

n264 1 1 8.79 0.00 2

n265 1 1 10.14 0.00 2

n266 1 1 8.79 0.00 2

n267 1 1 10.14 0.00 2

n268 1 1 8.79 0.00 2

n269 1 1 10.14 0.00 2

n270 1 1 8.79 0.00 2

n271 1 1 8.79 0.00 2

n272 1 1 8.79 0.00 2

n273 1 1 7.80 0.00 2

n274 1 1 7.14 0.00 2

n275 1 1 7.80 0.00 2

n276 1 1 8.07 0.00 2

n277 1 1 7.80 0.00 2

n278 1 1 8.89 0.00 2

n279 1 1 3.71 0.00 2

n280 1 1 3.99 0.00 2

n281 1 1 8.79 0.00 2

n282 1 1 7.80 0.00 2

n283 1 1 8.89 0.00 2

n284 1 1 3.71 0.00 2

n285 1 1 3.99 0.00 2

n286 1 1 8.79 0.00 2

n287 1 1 8.89 0.00 2

n288 1 1 9.38 0.00 2

n289 1 1 9.38 0.00 2

n290 1 1 9.38 0.00 2

n291 1 1 3.99 0.00 2

n292 1 1 3.99 0.00 2

n293 1 1 3.99 0.00 2

n294 1 1 3.99 0.00 2

n295 1 1 3.99 0.00 2

n296 1 1 3.99 0.00 2

n297 1 1 3.99 0.00 2

n298 1 1 3.99 0.00 2

n299 82 1 915.81 0.00 83

n300 8 1 60.12 0.00 9

n301 8 1 71.68 0.00 9

n302 9 1 79.19 0.00 10

n303 1 1 9.11 0.00 2

n304 1 1 9.11 0.00 2

n305 1 1 9.11 0.00 2

n306 1 1 9.11 0.00 2

n307 17 1 116.45 0.00 18

n308 23 1 188.12 0.00 24

n309 9 1 67.62 0.00 10

n310 8 1 60.92 0.00 9

n311 2 1 11.03 0.00 3

n312 2 1 18.15 0.00 3

n313 7 1 51.93 0.00 8

n314 1 1 4.05 0.00 2

n315 9 1 39.70 0.00 10

n316 9 1 37.94 0.00 10

n317 2 1 16.93 0.00 3

n318 3 1 15.62 0.00 4

n319 1 1 7.51 0.00 2

n320 5 1 31.88 0.00 6

n321 1 1 10.03 0.00 2

n322 1 1 3.70 0.00 2

n323 1 1 9.55 0.00 2

n324 3 1 23.92 0.00 4

n325 3 1 23.04 0.00 4

n326 2 1 10.35 0.00 3

n327 1 1 7.16 0.00 2

n328 3 1 16.86 0.00 4

n329 1 1 8.54 0.00 2

n330 3 1 16.86 0.00 4

n331 1 1 8.54 0.00 2

n332 3 1 14.05 0.00 4

n333 1 1 7.16 0.00 2

n334 3 1 13.91 0.00 4

n335 1 1 7.16 0.00 2

n336 2 1 7.40 0.00 3

n337 1 1 7.16 0.00 2

n338 2 1 10.35 0.00 3

n339 2 1 7.40 0.00 3

n340 1 1 9.97 0.00 2

n341 1 1 7.16 0.00 2

n342 1 1 9.97 0.00 2

n343 1 1 7.16 0.00 2

n344 1 1 3.70 0.00 2

n345 1 1 3.70 0.00 2

n346 1 1 3.70 0.00 2

n347 1 1 3.70 0.00 2

n348 1 1 3.70 0.00 2

n349 1 1 3.70 0.00 2

n350 1 1 3.70 0.00 2

n351 1 1 3.70 0.00 2

n\_state[0] 7 1 35.26 0.00 8

n\_state[1] 4 1 30.37 0.00 5

n\_state[2] 5 1 38.02 0.00 6

opcode[0] 4 1 39.22 0.00 5

opcode[1] 4 1 34.94 0.00 5

opcode[2] 7 1 51.43 0.00 8

opcode[3] 7 1 55.34 0.00 8

ov 1 1 6.74 0.00 2

rd 2 1 39.31 0.00 3

rst 1 1 26.06 0.00 2

state[0] 3 1 34.99 0.00 4

state[1] 4 1 43.37 0.00 5

state[2] 5 1 49.26 0.00 6

wrt 2 1 7.76 0.00 3

--------------------------------------------------------------------------------

Total 406 nets 1200 414 9326.97 0.00 1606

Maximum 82 2 915.81 0.00 83

Average 2.96 1.02 22.97 0.00 3.96

1

update\_timing

Information: Updating design information... (UID-85)

1

report\_timing -max\_paths 10

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Report : timing

-path full

-delay max

-max\_paths 10

Design : SCALAR\_PROCESSOR

Version: C-2009.06-SP5

Date : Fri Dec 5 03:58:23 2014

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Operating Conditions: WCCOM25 Library: tc240c

Wire Load Model Mode: top

Startpoint: wrt\_reg (rising edge-triggered flip-flop)

Endpoint: dat[7] (output port)

Path Group: default

Path Type: max

Point Incr Path

-----------------------------------------------------------

wrt\_reg/CP (CFD2QX1) 0.00 0.00 r

wrt\_reg/Q (CFD2QX1) 0.40 0.40 r

U312/Z (CNR2IX4) 0.40 0.80 r

dat\_tri[7]/Z (CTSX2) 0.30 1.10 f

dat[7] (inout) 0.00 1.10 f

data arrival time 1.10

max\_delay 1.10 1.10

output external delay 0.00 1.10

data required time 1.10

-----------------------------------------------------------

data required time 1.10

data arrival time -1.10

-----------------------------------------------------------

slack (MET) 0.00

Startpoint: dat[7] (input port)

Endpoint: ARRAY\_reg[2][7]

(rising edge-triggered flip-flop)

Path Group: default

Path Type: max

Point Incr Path

-----------------------------------------------------------

input external delay 0.00 0.00 f

dat[7] (inout) 0.00 0.00 f

U256/Z (CANR2X2) 0.13 0.13 r

U252/Z (COAN1X1) 0.26 0.39 r

U422/Z (COND2X1) 0.11 0.51 f

U295/Z (COR2X1) 0.22 0.73 f

ARRAY\_reg[2][7]/D (CFD2QXL) 0.00 0.73 f

data arrival time 0.73

max\_delay 1.10 1.10

library setup time -0.37 0.73

data required time 0.73

-----------------------------------------------------------

data required time 0.73

data arrival time -0.73

-----------------------------------------------------------

slack (MET) 0.00

Startpoint: wrt\_reg (rising edge-triggered flip-flop)

Endpoint: dat[4] (output port)

Path Group: default

Path Type: max

Point Incr Path

-----------------------------------------------------------

wrt\_reg/CP (CFD2QX1) 0.00 0.00 r

wrt\_reg/Q (CFD2QX1) 0.40 0.40 r

U312/Z (CNR2IX4) 0.40 0.80 r

dat\_tri[4]/Z (CTSX2) 0.30 1.10 f

dat[4] (inout) 0.00 1.10 f

data arrival time 1.10

max\_delay 1.10 1.10

output external delay 0.00 1.10

data required time 1.10

-----------------------------------------------------------

data required time 1.10

data arrival time -1.10

-----------------------------------------------------------

slack (MET) 0.00

Startpoint: wrt\_reg (rising edge-triggered flip-flop)

Endpoint: dat[6] (output port)

Path Group: default

Path Type: max

Point Incr Path

-----------------------------------------------------------

wrt\_reg/CP (CFD2QX1) 0.00 0.00 r

wrt\_reg/Q (CFD2QX1) 0.40 0.40 r

U312/Z (CNR2IX4) 0.40 0.80 r

dat\_tri[6]/Z (CTSX2) 0.30 1.10 f

dat[6] (inout) 0.00 1.10 f

data arrival time 1.10

max\_delay 1.10 1.10

output external delay 0.00 1.10

data required time 1.10

-----------------------------------------------------------

data required time 1.10

data arrival time -1.10

-----------------------------------------------------------

slack (MET) 0.00

Startpoint: wrt\_reg (rising edge-triggered flip-flop)

Endpoint: dat[5] (output port)

Path Group: default

Path Type: max

Point Incr Path

-----------------------------------------------------------

wrt\_reg/CP (CFD2QX1) 0.00 0.00 r

wrt\_reg/Q (CFD2QX1) 0.40 0.40 r

U312/Z (CNR2IX4) 0.40 0.80 r

dat\_tri[5]/Z (CTSX2) 0.30 1.10 f

dat[5] (inout) 0.00 1.10 f

data arrival time 1.10

max\_delay 1.10 1.10

output external delay 0.00 1.10

data required time 1.10

-----------------------------------------------------------

data required time 1.10

data arrival time -1.10

-----------------------------------------------------------

slack (MET) 0.00

Startpoint: dat[4] (input port)

Endpoint: ARRAY\_reg[3][4]

(rising edge-triggered flip-flop)

Path Group: default

Path Type: max

Point Incr Path

-----------------------------------------------------------

input external delay 0.00 0.00 f

dat[4] (inout) 0.00 0.00 f

U259/Z (CANR2X1) 0.15 0.15 r

U258/Z (CND2X1) 0.14 0.29 f

U443/Z (CANR2X1) 0.22 0.51 r

U442/Z (COND3X1) 0.21 0.72 f

ARRAY\_reg[3][4]/D (CFD2QXL) 0.00 0.72 f

data arrival time 0.72

max\_delay 1.10 1.10

library setup time -0.37 0.73

data required time 0.73

-----------------------------------------------------------

data required time 0.73

data arrival time -0.72

-----------------------------------------------------------

slack (MET) 0.01

Startpoint: dat[5] (input port)

Endpoint: ARRAY\_reg[3][5]

(rising edge-triggered flip-flop)

Path Group: default

Path Type: max

Point Incr Path

-----------------------------------------------------------

input external delay 0.00 0.00 f

dat[5] (inout) 0.00 0.00 f

U255/Z (CANR2X1) 0.15 0.15 r

U261/Z (CND2X1) 0.14 0.29 f

U445/Z (CANR2X1) 0.22 0.51 r

U444/Z (COND3X1) 0.21 0.72 f

ARRAY\_reg[3][5]/D (CFD2QXL) 0.00 0.72 f

data arrival time 0.72

max\_delay 1.10 1.10

library setup time -0.37 0.73

data required time 0.73

-----------------------------------------------------------

data required time 0.73

data arrival time -0.72

-----------------------------------------------------------

slack (MET) 0.01

Startpoint: wrt\_reg (rising edge-triggered flip-flop)

Endpoint: dat[1] (output port)

Path Group: default

Path Type: max

Point Incr Path

-----------------------------------------------------------

wrt\_reg/CP (CFD2QX1) 0.00 0.00 r

wrt\_reg/Q (CFD2QX1) 0.40 0.40 r

U312/Z (CNR2IX4) 0.40 0.80 r

dat\_tri[1]/Z (CTSX2) 0.29 1.09 f

dat[1] (inout) 0.00 1.09 f

data arrival time 1.09

max\_delay 1.10 1.10

output external delay 0.00 1.10

data required time 1.10

-----------------------------------------------------------

data required time 1.10

data arrival time -1.09

-----------------------------------------------------------

slack (MET) 0.01

Startpoint: dat[4] (input port)

Endpoint: ARRAY\_reg[2][4]

(rising edge-triggered flip-flop)

Path Group: default

Path Type: max

Point Incr Path

-----------------------------------------------------------

input external delay 0.00 0.00 f

dat[4] (inout) 0.00 0.00 f

U262/Z (CANR2X1) 0.14 0.14 r

U299/Z (COAN1X1) 0.26 0.40 r

U425/Z (COND2X1) 0.11 0.51 f

U298/Z (COR2X1) 0.23 0.75 f

ARRAY\_reg[2][4]/D (CFD2QX2) 0.00 0.75 f

data arrival time 0.75

max\_delay 1.10 1.10

library setup time -0.34 0.76

data required time 0.76

-----------------------------------------------------------

data required time 0.76

data arrival time -0.75

-----------------------------------------------------------

slack (MET) 0.01

Startpoint: dat[5] (input port)

Endpoint: ARRAY\_reg[2][5]

(rising edge-triggered flip-flop)

Path Group: default

Path Type: max

Point Incr Path

-----------------------------------------------------------

input external delay 0.00 0.00 f

dat[5] (inout) 0.00 0.00 f

U254/Z (CANR2X1) 0.14 0.14 r

U306/Z (COAN1X1) 0.26 0.40 r

U424/Z (COND2X1) 0.11 0.51 f

U305/Z (COR2X1) 0.23 0.75 f

ARRAY\_reg[2][5]/D (CFD2QX2) 0.00 0.75 f

data arrival time 0.75

max\_delay 1.10 1.10

library setup time -0.34 0.76

data required time 0.76

-----------------------------------------------------------

data required time 0.76

data arrival time -0.75

-----------------------------------------------------------

slack (MET) 0.01

1

report\_area

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Report : area

Design : SCALAR\_PROCESSOR

Version: C-2009.06-SP5

Date : Fri Dec 5 03:58:23 2014

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Library(s) Used:

tc240c (File: /apps/toshiba/sjsu/synopsys/tc240c/tc240c.db\_WCCOM25)

Number of ports: 20

Number of nets: 406

Number of cells: 388

Number of references: 44

Combinational area: 657.500000

Noncombinational area: 450.500000

Net Interconnect area: undefined (No wire load specified)

Total cell area: 1108.000000

Total area: undefined

1

report\_power

Loading db file '/apps/toshiba/sjsu/synopsys/tc240c/tc240c.db\_WCCOM25'

Warning: Main library 'tc240c' does not specify the following unit required for power: 'Leakage Power'. (PWR-424)

Information:PRopagating switching activity (low effort zero delay simulation). (PWR-6)

Warning: There is no defined clock in the design. (PWR-80)

Warning: Design has unannotated primary inputs. (PWR-414)

Warning: Design has unannotated sequential cell outputs. (PWR-415)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Report : power

-analysis\_effort low

Design : SCALAR\_PROCESSOR

Version: C-2009.06-SP5

Date : Fri Dec 5 03:58:24 2014

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Library(s) Used:

tc240c (File: /apps/toshiba/sjsu/synopsys/tc240c/tc240c.db\_WCCOM25)

Operating Conditions: WCCOM25 Library: tc240c

Wire Load Model Mode: top

Global Operating Voltage = 2.3

Power-specific unit information :

Voltage Units = 1V

Capacitance Units = 1.000000ff

Time Units = 1ns

Dynamic Power Units = 1uW (derived from V,C,T units)

Leakage Power Units = Unitless

Cell Internal Power = 2.1754 mW (88%)

Net Switching Power = 301.2914 uW (12%)

---------

Total Dynamic Power = 2.4767 mW (100%)

Cell Leakage Power = 0.0000

1

write -hierarchy -format verilog -output scalar\_netlist.v

Writing verilog file '/home/pa/pate8552/271new/scalar\_netlist.v'.

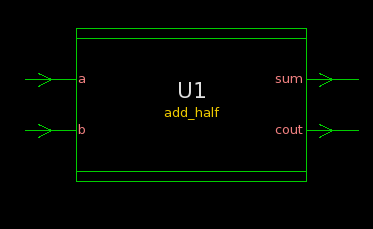
1

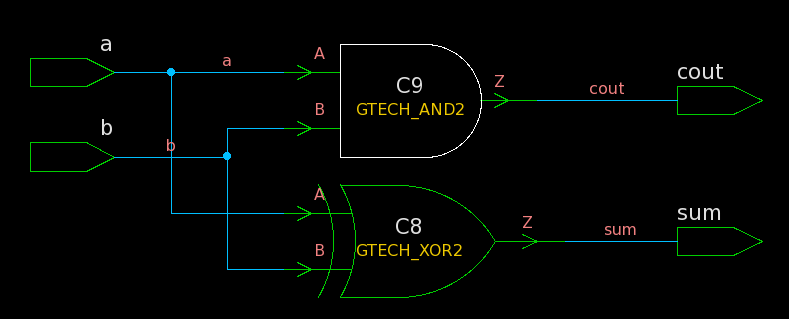
quit

Thank you...

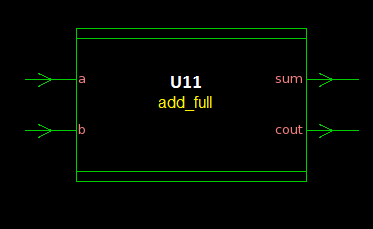
**C.4 Screenshot Circuits from Synthesis (Design Compiler)**

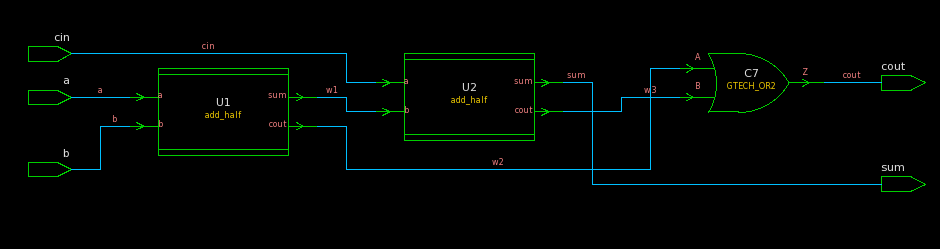
* **Half Adder :**

****

****

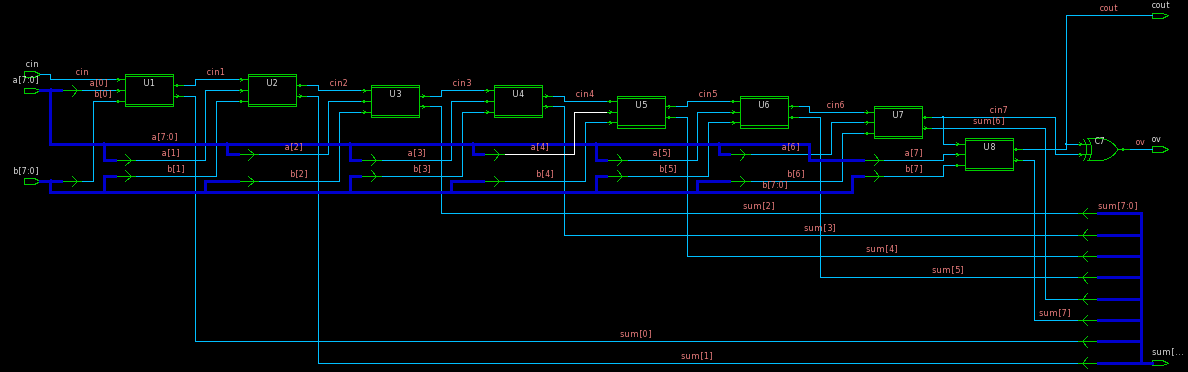
* **Full Adder :**

****

****

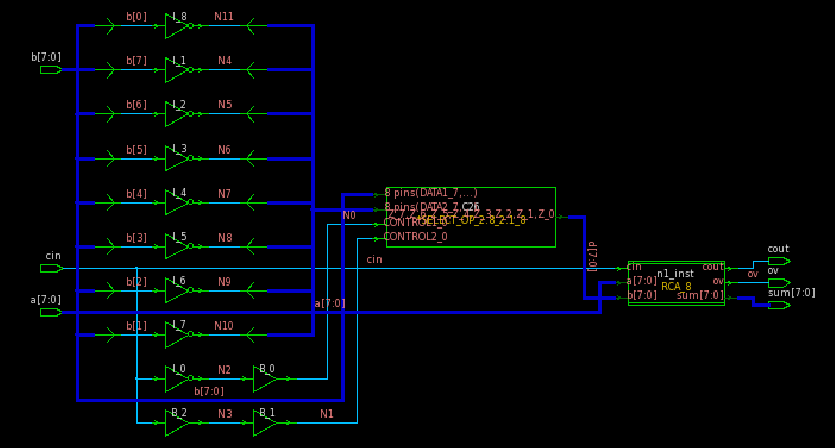
* **8-bit Ripple Carry Adder (RCA)**

****

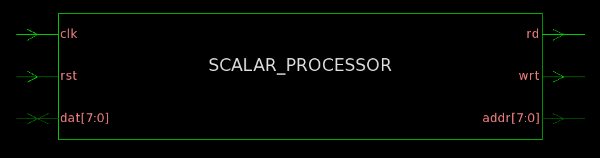
****

* **ALU**

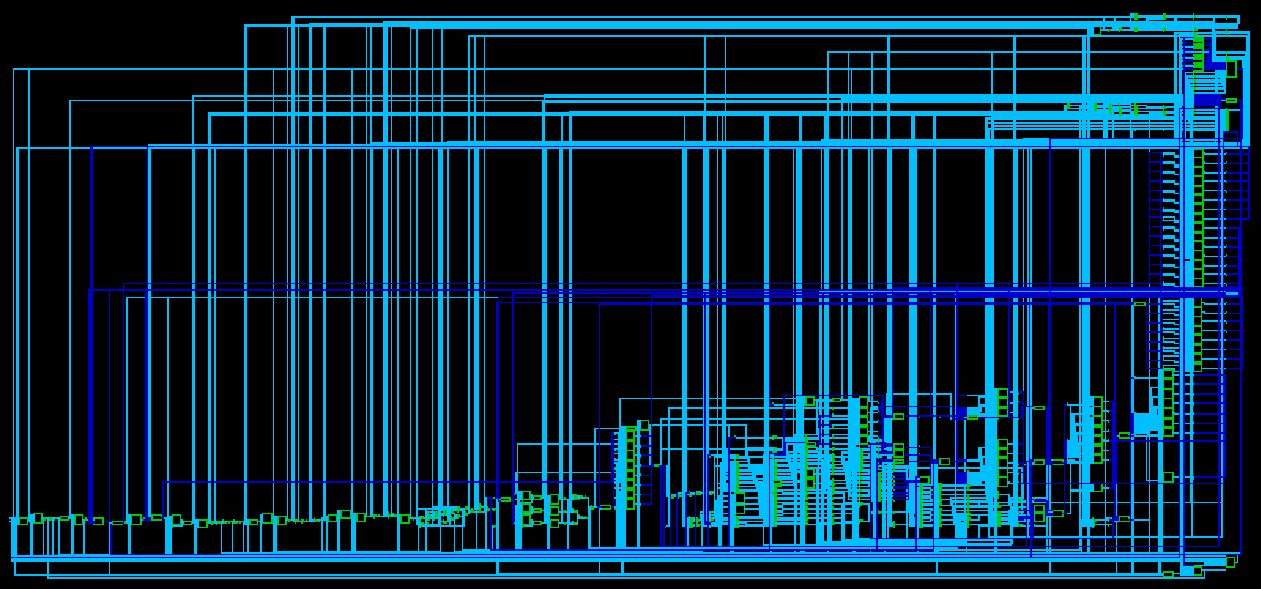
****

****

* **SCALAR Processor**

****

* **Schematic Layout of Scalar Processor**

****