DESIGN OF 32-BIT MICROCONTROLLER PROCESSOR IN SOC

Byreddy Swetha[1], T. Praveen Blesington[2], Fazal Noor Basha[2], B. BhanuMurthy [3]
[1]M.Tech Student, Department of ECE Engineering, KL University, Vijayawada, India
[2] Assoc. Prof, Department of ECE Engineering, KL University, Vijayawada, India
[3] Prof. & Vice-Dean, Quest International University, Malaysia

ABSTRACT

This paper proposes a 32-bit RISC (reduced instruction set computer) microcontroller processor [1] in system on chip (SOC). A system on a chip or system on chip (soC or SOC) is an integrated all components of a computer or other electronic systems into a single chip. A soc is a complete system on chip. A system includes one microcontroller, one microprocessor, memory and peripherals. A microcontroller (sometimes abbreviated µC, uc or MCU) is a small computer on a single integrated circuit containing a processor core, memory, and programmable input/output peripherals. Program memory in the form of NOR flash or OTP ROM is also often included on chip, as well as a typically small amount of RAM. Microcontrollers are designed for embedded applications, in contrast to the microprocessors used in personal computers or other general purpose applications. This paper concerned with the design of a 32-bit Reduced Instruction Set Computer (RISC) processor on a cadence tool. The processor has been designed with Verilog [3][4], synthesized using rc, simulated using nclaunch.

KEYWORDS: *SOC, Microcontrollers, peripherals*.

I. INTRODUCTION

1.1. SOC (system on chip):

A system on a chip or system on chip (SoC or SOC) [8] is an integrated circuit (IC) that integrates all components of a computer or other electronic system into a single chip. It may contain digital, analog, mixed-signal, and often radio-frequency functions—all on a single chip substrate. A typical application is in the area of embedded systems.

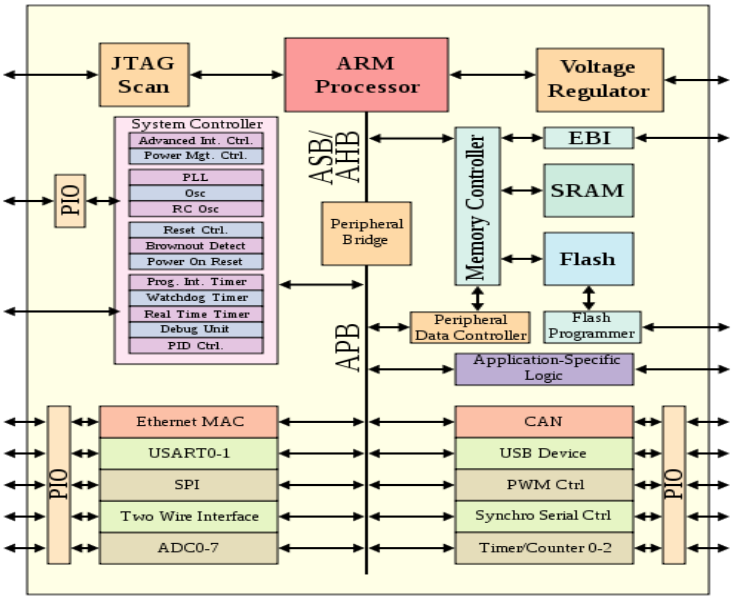

Fig 1: Microcontroller-based system on a chip

1.2. Microcontroller

Microcontroller [5] is a 32-bit fully embedded machine. It has 32-bit RICS microprocessor, data cache, instruction cache, FLASH controller, SDRAM controller, DMA (dynamic memory access), UART [7] (universal asynchronous receiver transmitter), Timer. All the peripherals connected through BUS Arbiter. The basic internal designs of microcontrollers are pretty similar. Figure 1 shows the block diagram of a typical microcontroller. All components are connected via an internal bus and are all integrated on one chip. The modules are connected to the outside world via I/O pins.

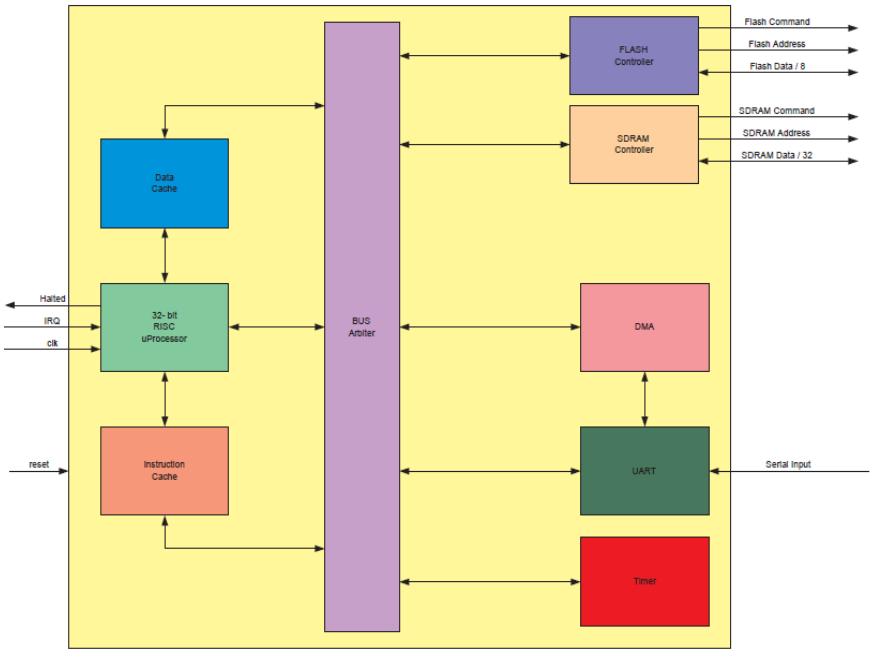

Fig 2: Microcontroller

Processor Core: The CPU of the controller. It contains the arithmetic logic unit, the control unit and the registers (stack pointer, program counter, accumulator register, register file, . . .).
Memory: The memory is sometimes split into program memory and data memory. In larger controllers, a DMA controller handles data transfers between peripheral components and the memory.
Interrupt Controller: Interrupts are useful for interrupting the normal program flow in case of (important) external or internal events. In conjunction with sleep modes, they help to conserve power.
Timer/Counter: Most controllers have at least one and more likely 2-3 Timer/Counters, which can be used to timestamp events, measure intervals, or count events. Many controllers also contain PWM (pulse width modulation) outputs, which can be used to drive motors or for safe breaking (antilock brake system, ABS). Furthermore the PWM output can, in conjunction with an external filter, be used to realize a cheap digital/analog converter.
Digital I/O: Parallel digital I/O ports are one of the main features of microcontrollers. The number of I/O pins varies from 3-4 to over 90, depending on the controller family and the controller type.
Analog I/O: Apart from a few small controllers, most microcontrollers have integrated analog/digital converters, which differ in the number of channels (2-16) and their resolution (8-12 bits). The analog module also generally features an analog comparator. In some cases, the microcontroller includes digital/analog converters.
Interfaces: Controllers generally have at least one serial interface which can be used to download the program and for communication with the development PC in general. Since serial interfaces can also be used to communicate with external peripheral devices, most controllers offer several and varied interfaces like SPI and SCI. Many microcontrollers also contain integrated bus controllers for the most common (field) busses. IIC and CAN controllers lead the field here. Larger microcontrollers may also contain PCI, USB, or Ethernet interfaces.

1.3. 32-bit RISC microprocessor

RISC: The RISC [6] architecture has simple, hard-wired instructions which often take only one or a few clock cycles to execute. RISC machines feature a small and fixed code size with comparatively few instructions and few addressing modes. As a result, execution of instructions is very fast, but the instruction set is rather simple.

Nowadays, computers are indispensable tools for most of everyday activities. With the rapid development of the silicon technology and the decreasing cost of the integrated circuit, RISC processor is increasing widely used in every field. RISC is an extension of the architecture principles of the Reduced Instruction Set Computer (RISC). The simple design provides exceptional performance and is ideal for use in a broad family of cost-effective, compatible systems.

The main features of RISC processor [2] are the instruction set can be hardwired to speed instruction execution. No microcode is needed for single cycle execution. All instructions are one word (fixed bit) in length. This simplifies the instruction fetch mechanism since the location of instruction boundaries is not a function of the instruction type. The processor has small number of addressing modes. Only load and store instructions access memory. There are no computational instructions that access memory; load/store instructions operate between memory and a register. Control hardware is simplified and the machine cycle time is minimized. The instructions were designed to be easily divisible into parts. This and the fixed size of the instructions allow the instructions to be easily piped. RISC provides a flexible and expandable architecture that maximizes performance from any given semiconductor technology. RISC includes extensions to RISC concepts that help achieve given levels of performance at significantly lower cost than other systems. In paper, 32-bit RISC microcontroller processor is designed, by using cadence tool simulation, synthesis and gate, area reports is obtained.

II. 32-BIT RISC MICROPROCESSOR

The architecture of an 32-bit RISC processor is shown in Figure 2. This architecture consists of arithmetic logic unit, control unit, program counter, instruction register, accumulator .One shared memory for instructions (program) and data with one data bus and one address bus between processor and memory. Instruction and data are fetched in sequential order so that the latency incurred between the machine cycles can be reduced. Three stages of pipelining have been incorporated in the design which increases the speed of operation. The pipelining stages are fetch, decode and execute. In fetch, the instruction and the necessary data are drawn from the memory. Whereas in decode, the instruction and data that are drawn from the memory are separated activating the components and the data path so as to execute And finally in execution, the instruction is performed, the data is manipulated and the result is stored.

The control unit reads the opcode and instruction bits and then creates control signals as outputs that triggers the respective components and data path to perform the desired task. The control unit has two instruction decoders that decode the instruction bits and the decoded output of the control unit is fed as control signal either into Arithmetic logic unit (ALU) or Universal shifter or Barrel shift rotator. The operands are received from register A and register B by the ALU. Depending on the control signal from the control unit the ALU performs either arithmetic or logic operations. After the execution of the instruction, the result is stored in the accumulator register. Input is taken from source register A and is either loaded or shifted in right or left direction based on the control lines activated by the control unit. The shifted data is saved in the destination register which is nothing but the accumulator register. Input data is given from source register A and rotated N number of times based on the opcode fed from the control unit. The rotated data is stored in the accumulator register.

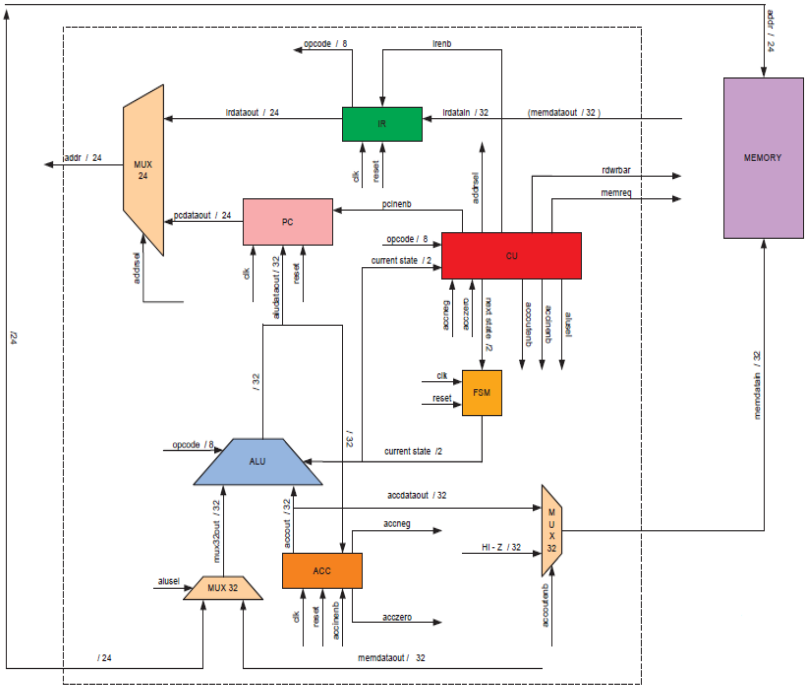

Fig 3: 32-bit RISC micro processor.

2.1. Multiplexer (MUX):

A multiplexer (or **MUX**) is a device that selects one of several analog or digital input signals and forwards the selected input into a single line.

A multiplexer of 2^n inputs has *n* select lines, which are used to select which input line to send to the output. Multiplexers are mainly used to increase the amount of data that can be sent over the network within a certain amount of time and bandwidth. A multiplexer is also called a data selector.

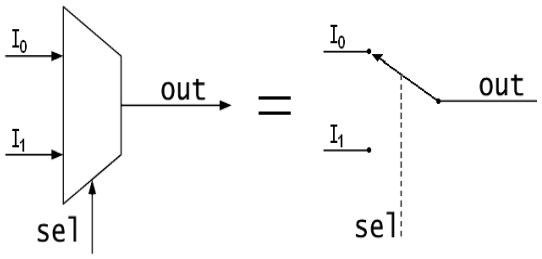

Fig 4: 2X1mux

2.2. ALU (arithmetic logic unit):

The arithmetic/logic unit (ALU) executes all arithmetic and logical operations. The arithmetic/logic unit can perform four kinds of arithmetic operations, or mathematical calculations: addition, subtraction, multiplication, and division. As its name implies, the arithmetic/logic unit also performs logical operations. A logical operation is usually a comparison. The unit can compare numbers, letters, or special characters. The computer can then take action based on the result of the comparison. This is a very important capability.

2.3. ACC (accumulator):

The accumulator is the principal register of the arithmetic logic unit of a microprocessor. Registers are sets of flip-flops which can hold data. The accumulator typically holds the first piece of data for a calculation. If a number from memory is added to that date, the sum replaces the original data in the

accumulator. It is the repository for successive results of arithmetic operations, which may then be transferred to memory, to an output device, etc...

2.4. IR (instruction register):

Modern processors can even do some of the steps of out of order as decoding on several instructions is done in parallel. Decoding the opcode in the instruction register includes determining the instruction, where its operands are in memory, retrieving the operands from memory, allocating processor resources to execute the command. The output of IR is available to control circuits which generate the timing signals that controls the various processing elements involved in executing the instruction.

2.5. CU (control unit):

The control unit design is based on using FSM (Finite State Machine) and we designed it in a way that allows each state to run at one clock cycle, the first state is the reset which is initializes the CPU internal registers and variables. The machine goes to the reset state by enabling the reset signal for a certain number of clocks. Following the reset state would be the instruction fetching and decoding states which will enable the appropriate signals for reading instruction data from the ROM then decoding the parts of the instruction. The decoding state will also select the next state depending on the instruction, since every instruction has its own set of states, the control unit will jump to the correct state based on the instruction given. After all states of a running instruction are finished, the last one will return to the fetch state which will allow us to process the next instruction in the program.

2.6. PC (program counter):

The program counter or PC (also called the instruction pointer, or instruction address register, or just part of the instruction sequencer in some computers) is a processor register that indicates where the computer is in its instruction sequence. Depending on the details of the particular the computer, the PC holds either the address of the instruction being executed, or the address of the next instruction to be executed. In most processors, the instruction pointer is incremented automatically after fetching a program instruction, so that instructions are normally retrieved sequentially from memory, with certain instructions, such as branches, jumps and subroutine calls and returns, interrupting the sequence by placing a new value in the program counter.

III. RESULTS

3.1. MUX:

A multiplexer of 2^n inputs has *n* select lines, which are used to select which input line to send to the output.

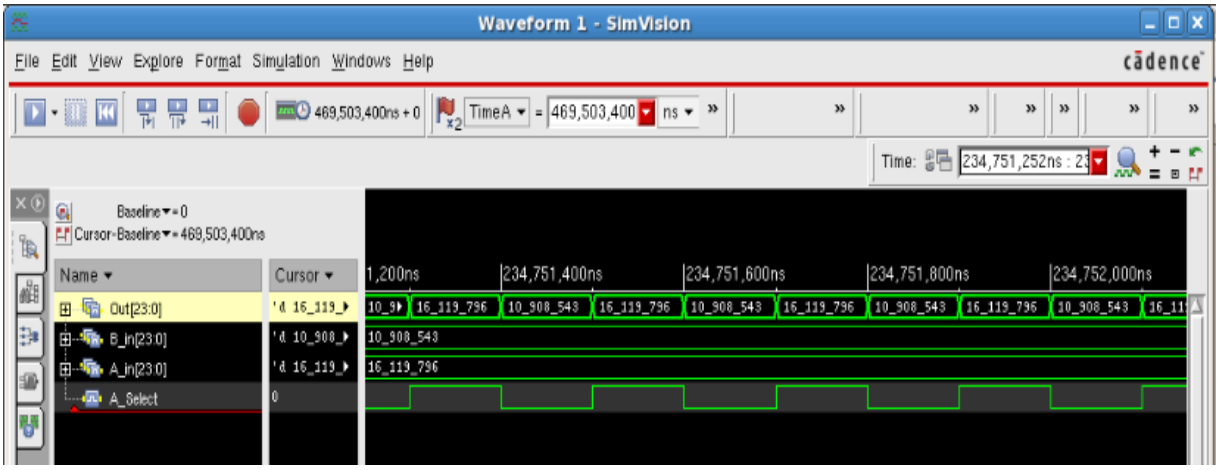

Fig 5: MUX12 simulation result

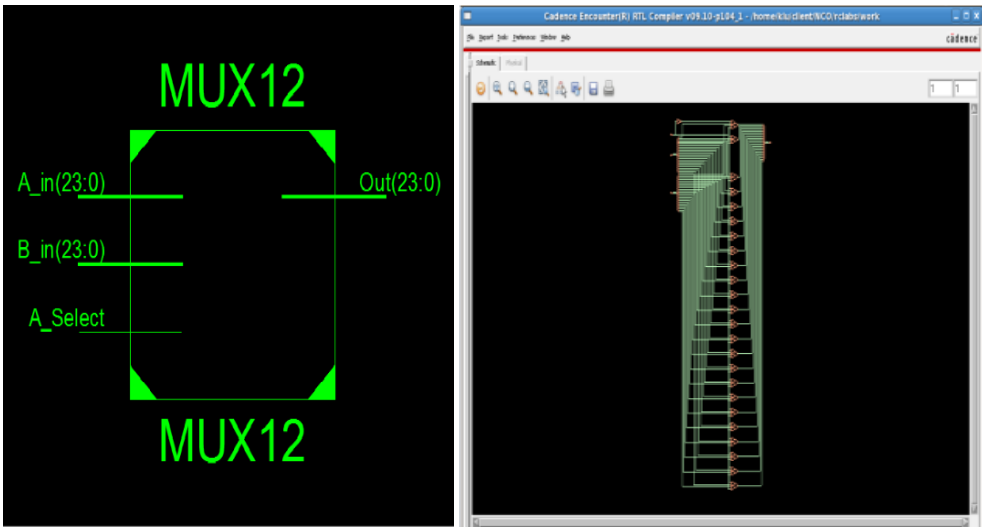

Fig 6: Top block of 24-bit MUX12

```
Terminal
File  Edit  View  Terminal  Tabs  Help
rc:/> report gate
===============================================================
  Generated by:              Encounter(R) RTL Compiler v09.10-p104_1
  Generated on:              Jan 22 2013  01:13:12 PM
  Module:                    MUX12
  Technology library:        slow_normal 1.0
  Operating conditions:      slow (balanced_tree)
  Wireload mode:             enclosed
  Area mode:                 timing library
===============================================================

 Gate    Instances    Area        Library
-----------------------------------------
AO22XL          24   135.475     slow_normal
INVXL            1     2.117     slow_normal
-----------------------------------------
total           25   137.592


  Type    Instances    Area  Area %
-------------------------------------
inverter          1   2.117     1.5
logic            24 135.475    98.5
-------------------------------------
total            25 137.592   100.0
```

Fig 7: Gate and Area report of MUX12

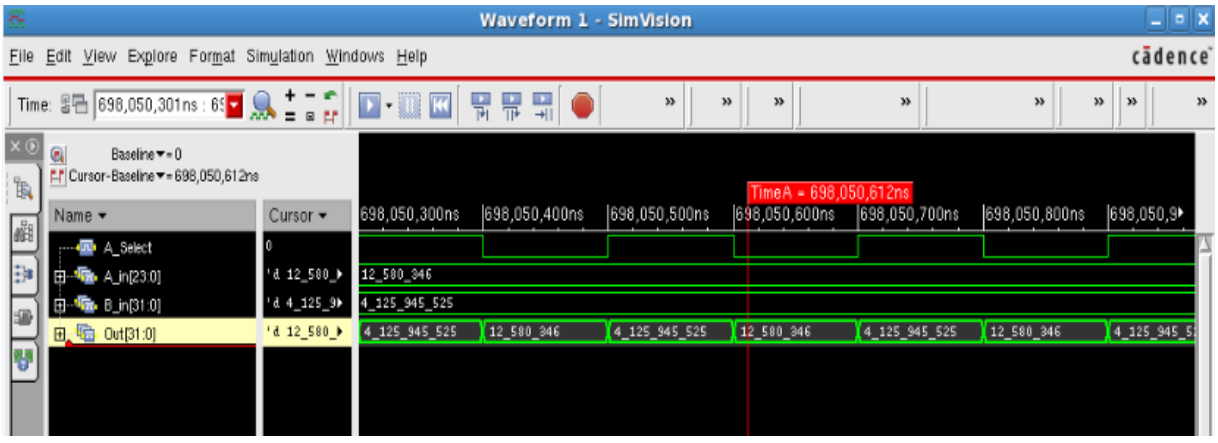

Fig 8: MUX16 (adress width=32) simulation result.

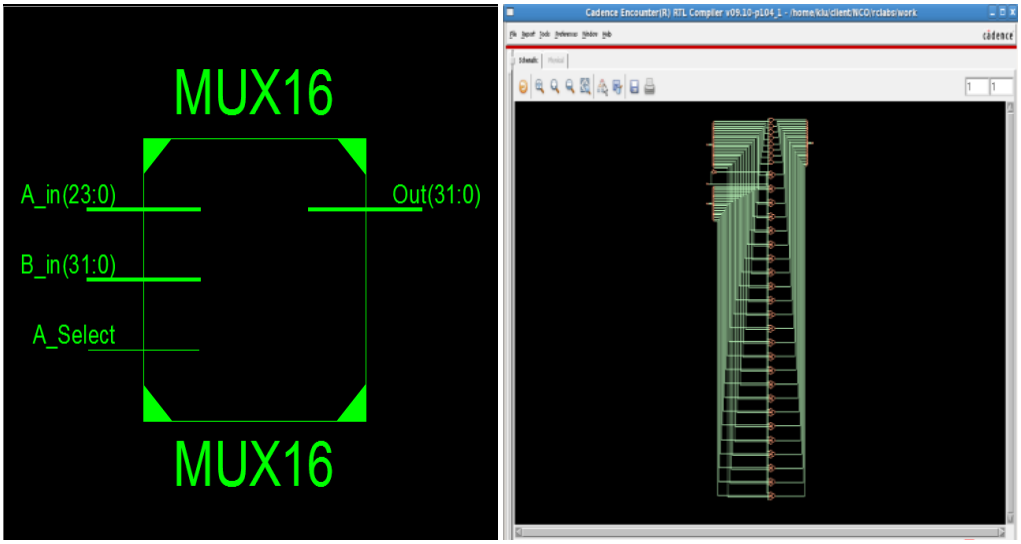

Fig 9: Top block of 32-bit MUX16

```
Terminal
File  Edit  View  Terminal  Tabs  Help
rc:/> report gate
===============================================================
  Generated by:           Encounter(R) RTL Compiler v09.10-p104_1
  Generated on:           Jan 22 2013  01:13:12 PM
  Module:                 MUX12
  Technology library:     slow_normal 1.0
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
===============================================================

 Gate    Instances    Area        Library
-----------------------------------------
AO22XL          24  135.475     slow_normal
INVXL            1    2.117     slow_normal
-----------------------------------------
total           25  137.592


  Type    Instances    Area  Area %
-------------------------------------
inverter          1   2.117     1.5
logic            24 135.475    98.5
-------------------------------------
total            25 137.592   100.0
```

Fig 10: Gate and Area report of MUX16

3.2. ALU (arithmetic logic unit):

The arithmetic/logic unit (ALU) executes all arithmetic and logical operations. The arithmetic/logic unit can perform four kinds of arithmetic operations, or mathematical calculations: addition, subtraction, multiplication, and division. As its name implies, the arithmetic/logic unit also performs logical operations

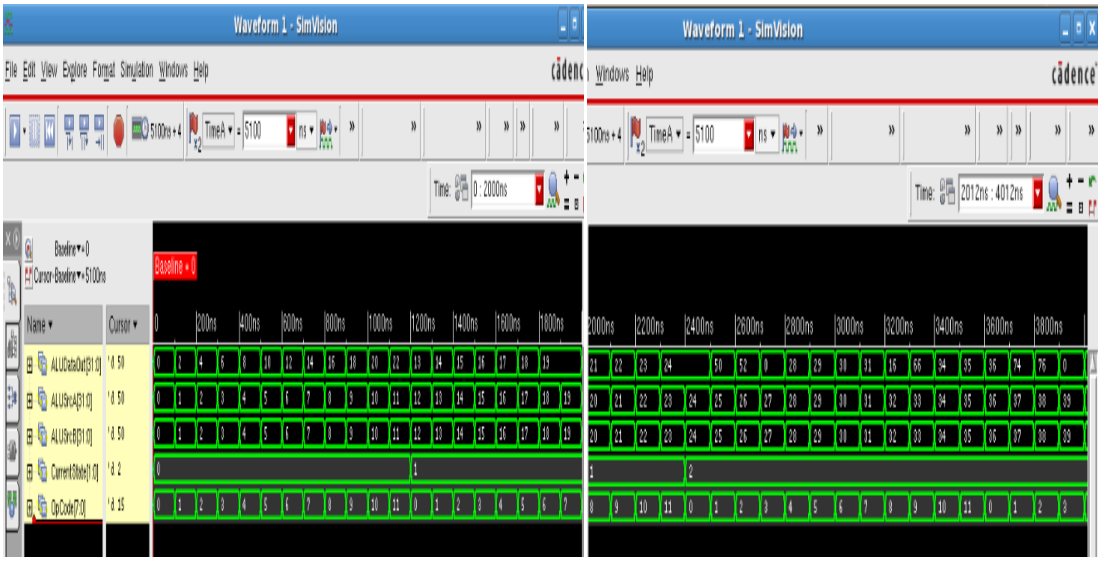

Fig 11: ALU simulation result

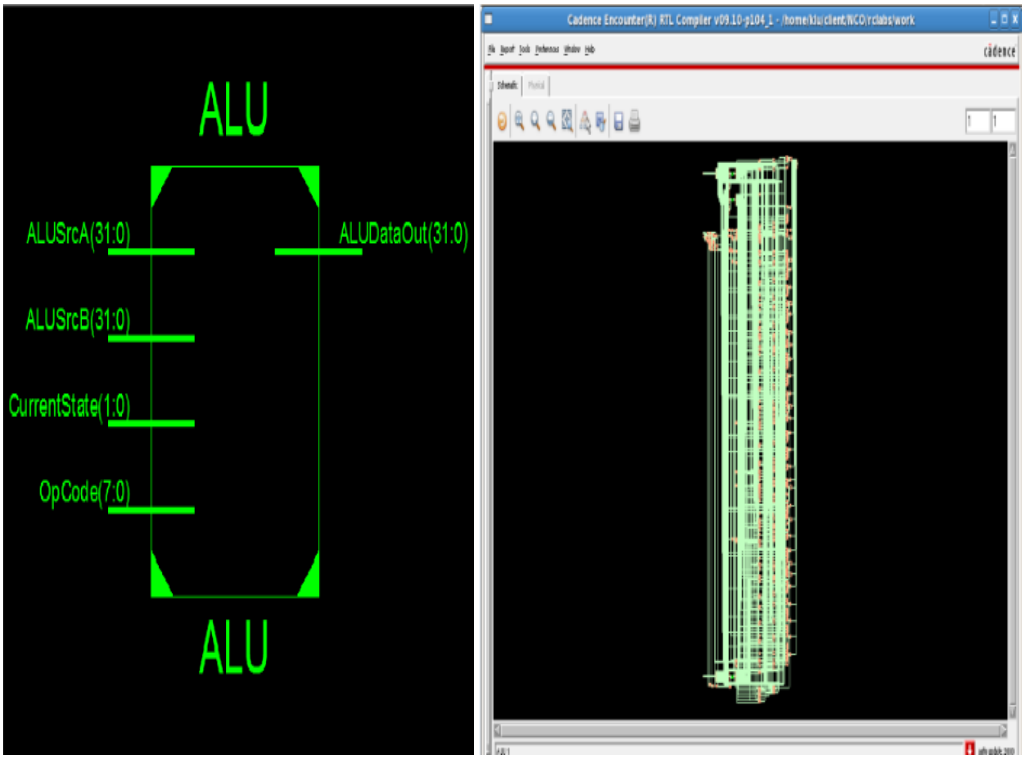

Fig 12: Top block of 32-bit ALU

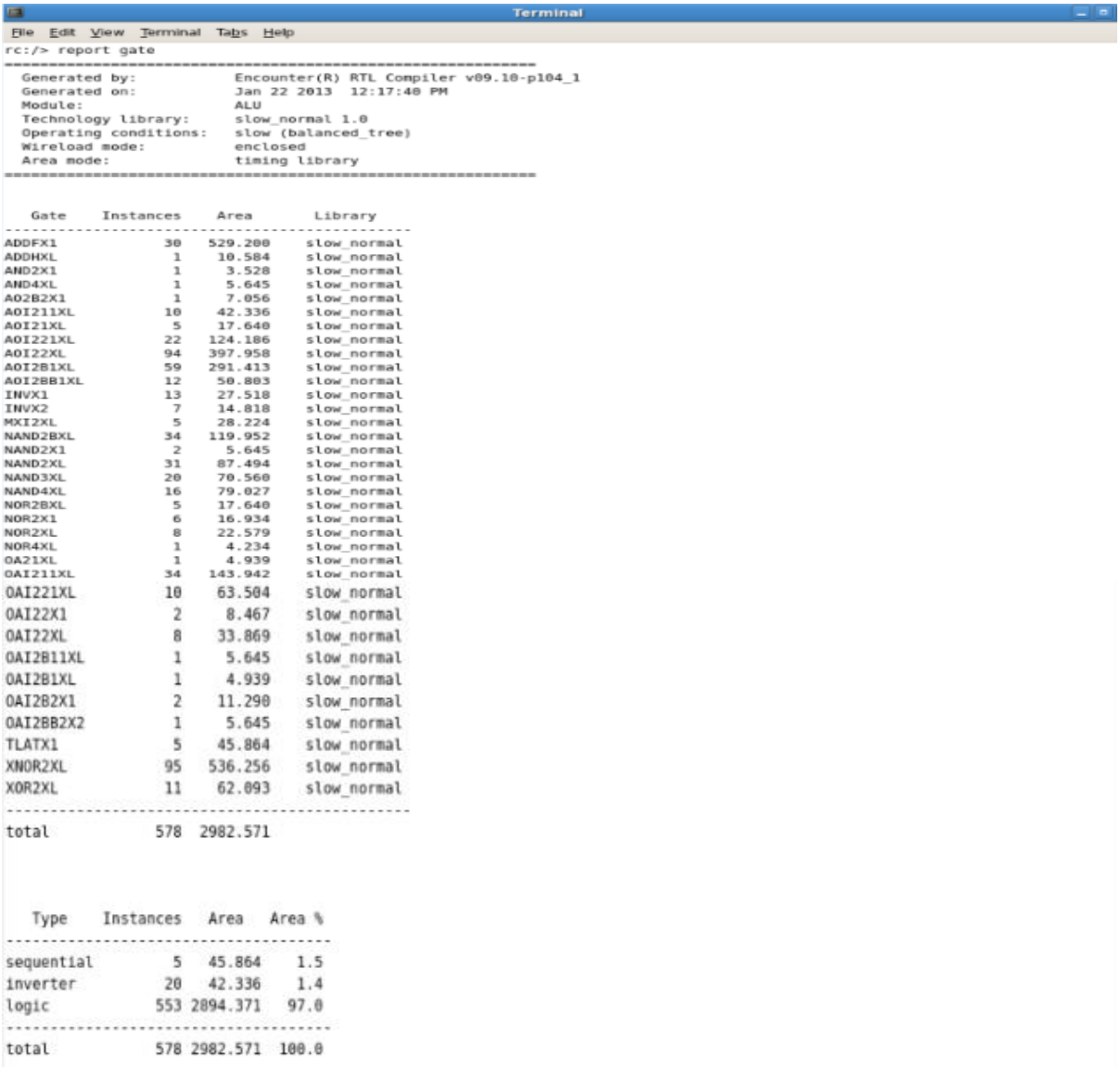

Fig 13: gate and area report of ALU.

3.3. ACC (accumulator):

It is used to differentiate zero, positive and negative values at the output.

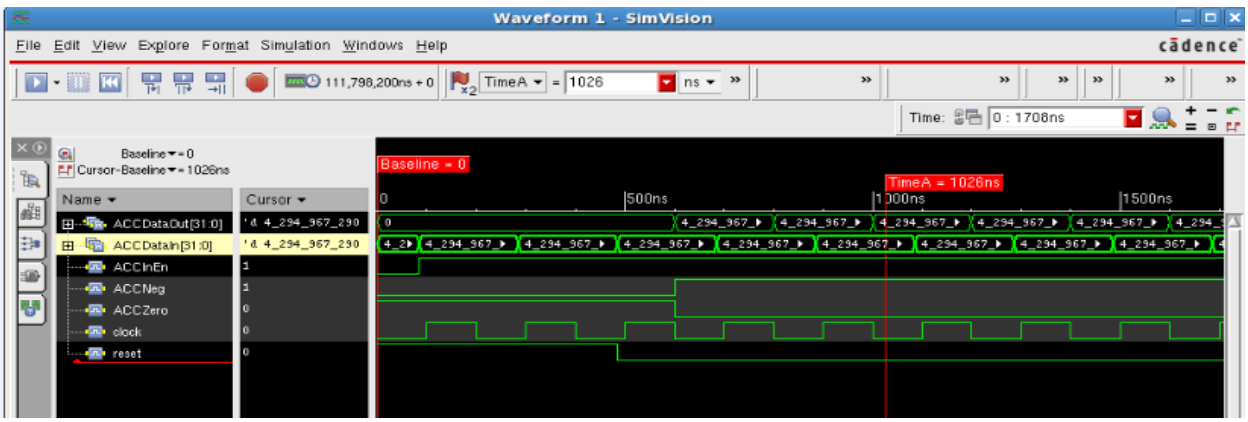

Fig 14: ACC simulation result

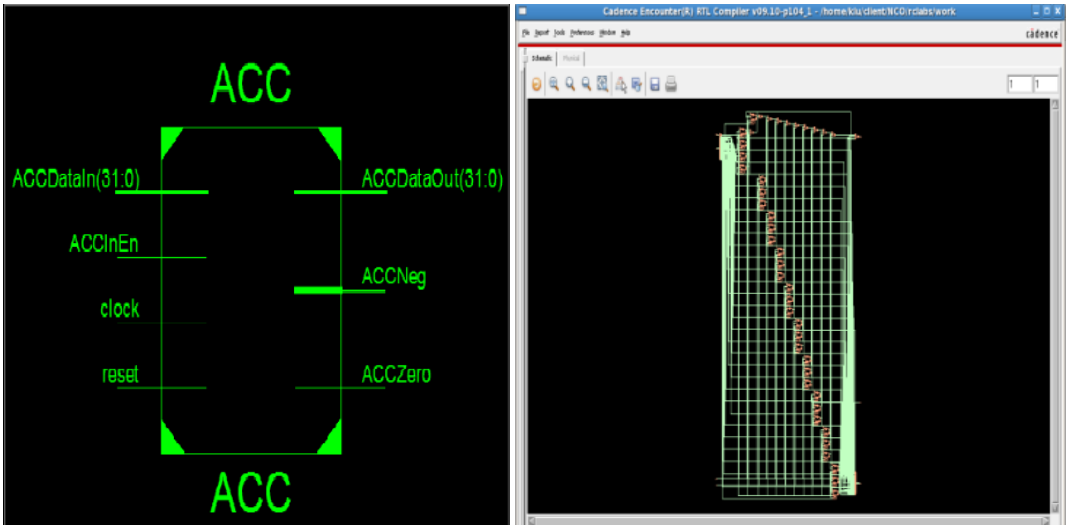

Fig 15: Synthesis report of 32-bit ACC

```
Terminal
File  Edit  View  Terminal  Tabs  Help

rc:/> report gate
==================================================================
  Generated by:           Encounter(R) RTL Compiler v09.10-p104_1
  Generated on:           Jan 22 2013  11:50:44 AM
  Module:                 acc
  Technology library:     slow_normal 1.0
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
==================================================================

   Gate      Instances     Area        Library
-------------------------------------------------
INVXL                1     2.117     slow_normal
NAND2BXL             1     3.528     slow_normal
NOR4BXL              8    45.158     slow_normal
NOR4XL               2     8.467     slow_normal
SDFFNSRHX1          32   948.326     slow_normal
-------------------------------------------------
total               44  1007.597

   Type     Instances    Area    Area %
----------------------------------------
sequential          32  948.326    94.1
inverter             1    2.117     0.2
logic               11   57.154     5.7
----------------------------------------
total               44 1007.597   100.0
```

Fig 16: Gate and Area report of ACC

3.4. IR (instruction register):

If reset is zero and enable is one at the negedge of clock pulse the 32 bit input will be stored in two registers, (1). 23-bit stored in operand out (2). 8-bit stored in opcodeout.

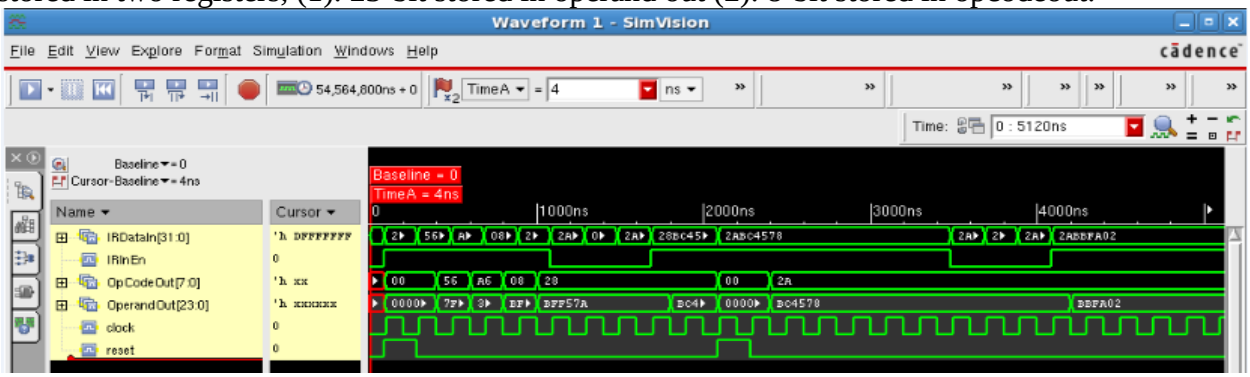

Fig 17: IR simulation result

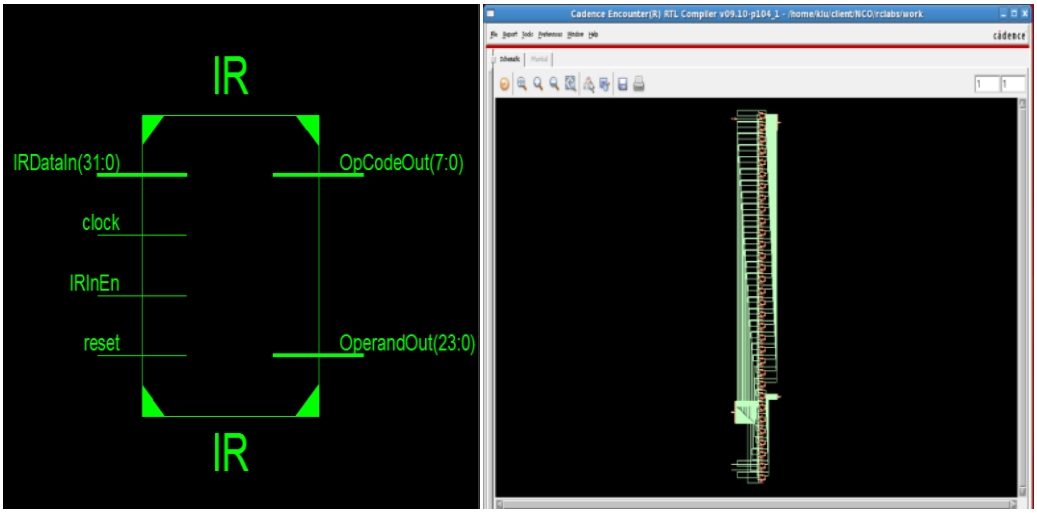

Fig 18: synthesis report of 32-bit IR.

```
Terminal
File  Edit  View  Terminal  Tabs  Help
rc:/> report gate
===============================================================
  Generated by:           Encounter(R) RTL Compiler v09.10-p104_1
  Generated on:           Jan 22 2013  01:51:50 PM
  Module:                 IR
  Technology library:     slow_normal 1.0
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
===============================================================

   Gate      Instances   Area       Library
-----------------------------------------------
INVXL                1    2.117    slow_normal
SDFFNSRHX1          32  948.326    slow_normal
-----------------------------------------------
total               33  950.443


   Type     Instances   Area  Area %
---------------------------------------
sequential          32 948.326   99.8
inverter             1   2.117    0.2
---------------------------------------
total               33 950.443  100.0
```

Fig 19: Gate and Area report of IR.

3.5. CU (control unit):

It is used to control the all internal modules of microprocessor.

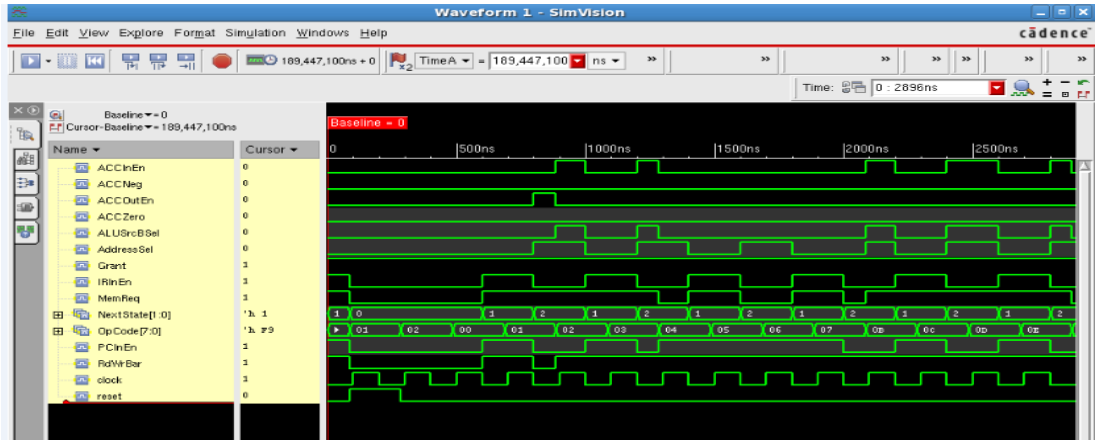

Fig 20: CU simulation result.

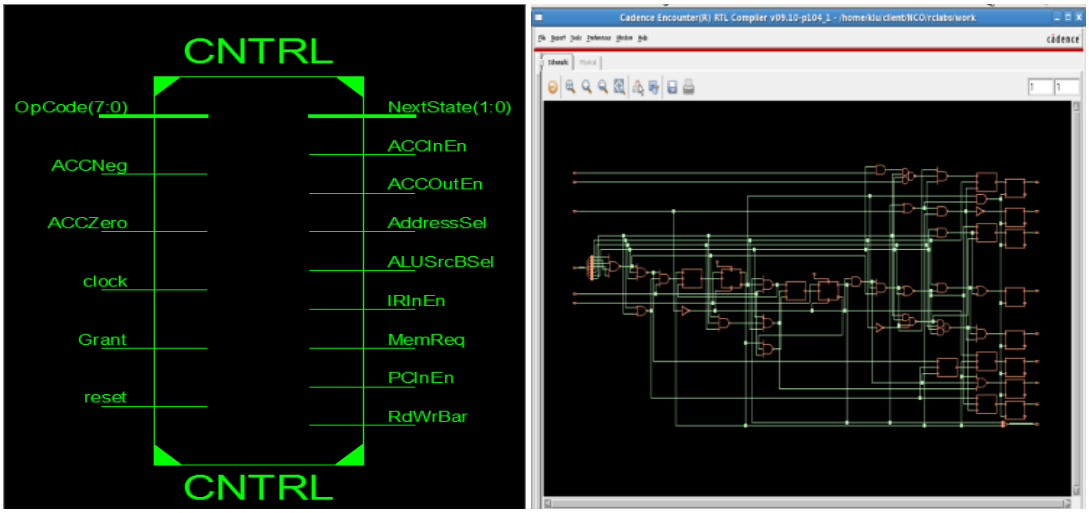

Fig 21: Synthesis report of 32-bit CU.

```
Terminal
File  Edit  View  Terminal  Tabs  Help
rc:/> report gate
---------------------------------------------------------------
   Generated by:              Encounter(R) RTL Compiler v09.10-p104_1
   Generated on:              Jan 23 2013  10:53:26 AM
   Module:                    CNTRL
   Technology library:        slow_normal 1.0
   Operating conditions:      slow (balanced_tree)
   Wireload mode:             enclosed
   Area mode:                 timing library
---------------------------------------------------------------

    Gate     Instances    Area       Library
-------------------------------------------------
AND2X1               1   3.528   slow_normal
AO21X2               1   4.939   slow_normal
AO21XL               1   4.939   slow_normal
AOI211XL             1   4.234   slow_normal
AOI22XL              1   4.234   slow_normal
AOI2B1X1             1   4.939   slow_normal
DFFNSRHX2            2  50.803   slow_normal
INVXL                3   6.350   slow_normal
NAND2BXL             2   7.056   slow_normal
NAND2XL              3   8.467   slow_normal
NAND3BXL             2   9.878   slow_normal
NAND3XL              5  17.640   slow_normal
NOR2XL               4  11.290   slow_normal
NOR3BXL              2   8.467   slow_normal
NOR3XL               1   3.528   slow_normal
NOR4XL               1   4.234   slow_normal
OAI21XL              2   7.056   slow_normal
OAI22XL              1   4.234   slow_normal
OAI2B1XL             1   4.939   slow_normal
OAI2BB1XL            1   4.234   slow_normal
TLATX1               8  73.382   slow_normal
-------------------------------------------------
total               44  248.371

    Type      Instances    Area   Area %
-----------------------------------------
sequential           10  124.186    50.0
inverter              3    6.350     2.6
logic                31  117.835    47.4
-----------------------------------------
total                44  248.371   100.0
```

Fig 22: Gate and Area report of CU.

3.6. PC (program counter):

- If reset is zero enable is one input is stored in the output.
- If reset is zero enable is zero previous output stored in the output.

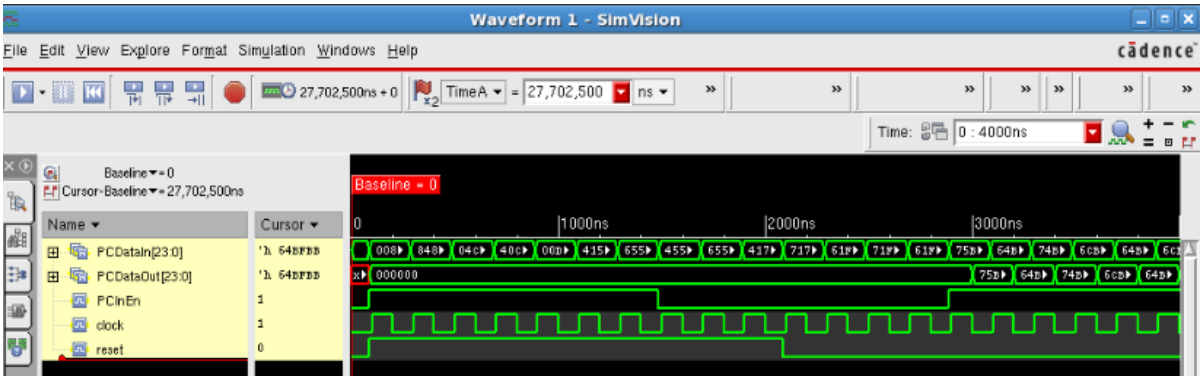

Fig 23: PC simulation result.

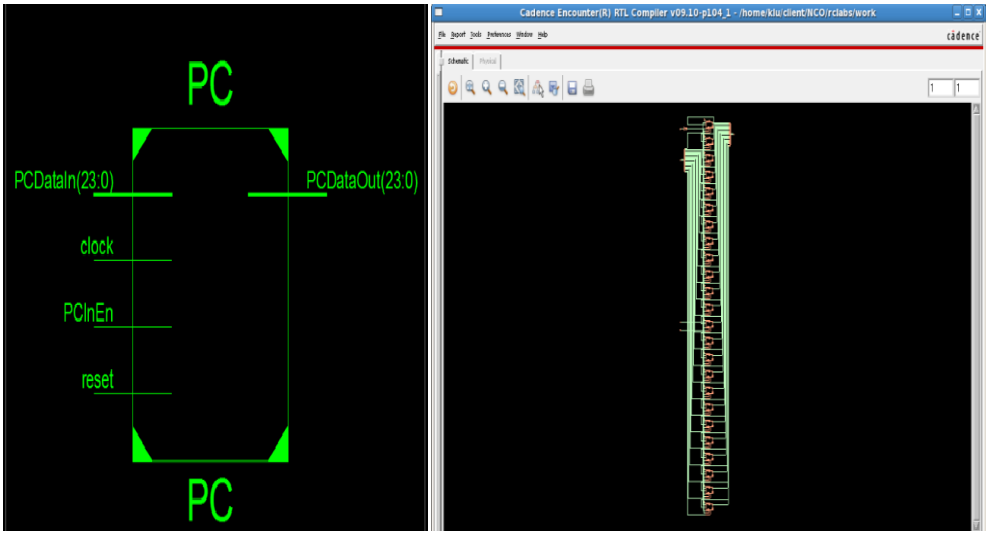

Fig 24: Synthesis report of 32-bit PC.

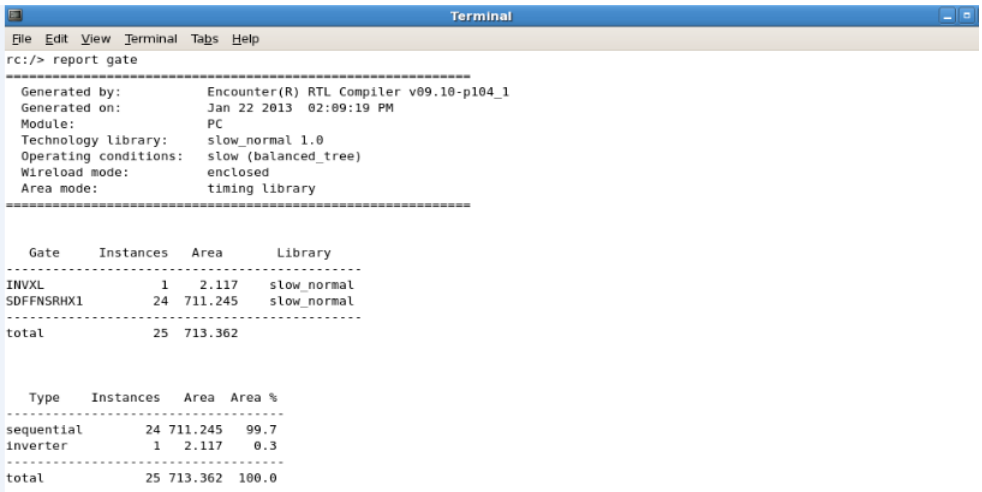

Fig 25: Gate and Area report of PC

IV. CONCLUSION AND FUTURE WORK

A 32-bit RISC Microprocessor in SOC has been designed on a cadence tool. The processor has been designed with Verilog, synthesized using rc, simulated using nclaunch. Future work will be added by designing AHB bus and interface the AHB bus with processor.

ACKNOWLEDGEMENTS

The authors would like to thank everyone who inspired and helped to publish this paper.

REFERENCES

[1]. Samiappa Sakthikumaran, S. Salivahanan and V.S. Kaanchana Bhaaskaran , June 2011, "16-Bit RISC Processor Design For Convolution Application", IEEE International Conference on Recent Trends In Information Technology, pp.394-397.
[2]. Advanced Microprocessors, Daniel Tabak, ISBN 0-07-062843-2, p 79-99.
[3]. Verilog Hardware Description Language, Section 18: Value change dumps (VCD) files, IEEE Std. 1364-2005, 2006.
[4]. Samir Palnitkar, Verilog HDL, A Guide to Digital Design and Synthesis.
[5]. "Introduction to Microcontrollers", Vienna University of Technology Institute of Computer Engineering Embedded Computing Systems Group, February 26, 2007.

[6]. RISC AND CISC Computer Architecture By Farhat Masood
[7]. Praveen Blessington, T.; Bhanu Murthy, B.; Ganesh, G.V.; Prasad, T.S.R; , "Optimal implementation of UART-SPI Interface in SoC," Devices, Circuits and Systems (ICDCS), 2012 International Conference on , vol., no., pp.673-677, 15-16 March 2012, doi: 10.1109/ICDCSyst.2012.6188657
[8]. Krishna, Karthik. T; Praveen Blessington, T.; Bhanu Murthy, B.; Ganesh, G.V.; , "Implementation of SOC'S audio video coprocessor," Devices, Circuits and Systems (ICDCS), 2012 International Conference on , vol., no., pp.5-8, 15-16 March 2012, doi: 0.1109/ICDCSyst.2012.618866

AUTHOR

Byreddy Swetha is graduated B.E (ECE) from JNTUA, Kurnool in the year 2011, Post graduated in M.E (VLSI Design) from KL University vijayawada in the year 2013. Her research interests are in low power vlsi, soc, noc.