IEEE International Conference on Advances in Engineering & Technology Research (ICAETR - 2014), August 01-02, 2014, Dr. Virendra Swarup Group of Institutions, Unnao, India

Synthesis and Simulation of a 32Bit MIPS RISC Processor using VHDL

Mr. S. P. Ritpurkar
Department of Electronics and Telecommunication
B. D. College of Engineering, Sevagram, India
email: ritpurkarsagar@gmail.com

Prof. M. N. Thakare
Department of Electronics and Telecommunication
B. D. College of Engineering, Sevagram, India
email: mnt_ent@rediffmail.com

Prof. G. D. Korde
Department of Electronics and Telecommunication
B. D. College of Engineering, Sevagram, India
email: gdkorde@rediffmail.com

***Abstract*—The main objective of the project is to design and simulate 32Bit MIPS (Microprocessor Interlocked Pipeline Stages) RISC (Reduced Instruction Set Computer) Processor using VHDL (Very High Speed Integrated Circuit Hardware Description Language). In this paper, we analyze Instruction fetch module, Decoder module, Execution module which includes 32Bit Floating point ALU, Flag register of 32Bit, MIPS Instruction Set, and 32Bit general purpose registers and design theory based on 32Bit MIPS RISC Processor. Furthermore, we use pipeline concept which involves Instruction Fetch, Instruction Decode, Execution, Memory and Write Back modules of MIPS RISC processor based on 32Bit MIPS Instruction set in a single clock cycle. All the modules in the design are coded in VHDL, as it is very useful language with its concept of concurrency to cope successfully with the parallelism of digital hardware. Finally, Synthesis and Simulation of the design is done in XILINX 13.1i ISE Simulator.**

Keywords-Data Path; Instruction Set; MIPS; Pipeline; RISC; VHDL; XILINX 13.1i.

I. INTRODUCTION

CISC stands for Complex Instruction Set Computer. The main aim of CISC architecture is to complete a task in as few lines of instruction as possible. CISC Instruction attributes are, it has a Large number of instruction set, Complex and extensive instructions, Extensive manipulation of low-level computational elements for memory, binary arithmetic and addressing, Efficient micro encoding of machine instructions and Processor contents relatively fewer registers. Due to complex architecture of CISC, each instruction takes longer to execute, including simpler instructions. System 360– IBM, VAX–Digital Equipment Corporation, PDP-11–Digital Equipment Corporation, Motorola 68000 family and Intel x86 architecture processors uses the CISC architecture.

RISC stands for Reduced Instruction Set Computer. The main aim of RISC architecture is to produce a simpler and faster set of instructions. RISC architecture is developed as a reaction to CISC. Instructions of RISC should not take as long to run as this is the down side to CISC design. RISC provides programmers with simpler instructions that can execute quickly. RISC attributes are, it contents a smaller number of instructions, Instructions can execute in one machine cycle or less, Encoded instructions are the same size and architecture contains many registers which are organized into register file. Simple instructions allow RISC processor to be easier to design and cheaper to produce. It requires less number of transistors allowing the processor to be smaller. It is easier to create powerful optimized compilers since there are fewer instructions in the instruction set. Apple iPods, iPhone, iPad, Palm and PocketPC, PDAs and Smartphones uses RISC architecture.

MIPS Technologies has a strong customer licensee base in home electronics and portable media players; 75 percent of Blu-ray Disc players are running on MIPS Technologies processors. In the digital home, the company's processors are predominately found in digital TVs and set-top boxes. The Sony PlayStation Portable uses two processors based on the MIPS32 4K processor. Within the networking segment, licensees include Cavium Networks and Net logic Microsystems. Licensees using MIPS to build smart phones and tablets include Actions Semiconductor and Ingenic Semiconductor. Tablets based on MIPS include the Cruz tablets from Velocity Micro. TCL Corporation is using MIPS processors for the development of smart phones. Companies can also obtain an MIPS architectural license for designing their own CPU cores using the MIPS instruction set. MIPS Technologies is predominately used in conjunction with Android and Linux operating systems [12]. So, designing of RISC Processor based on MIPS Instruction set is the necessary choice.

II. INSTRUCTION SET ARCHITECTURE

A. MIPS INSTRUCTION FORMATS

The architecture of MIPS RISC Processor is based on four MIPS Instruction formats R-Type, I-Type, J-Type and I/O Type illustrated in the figures below;

a) R-Type (Register Format)

(31 to 26)	(25 to 21)	(20 to 16)	(15 to 11)	(10 to 6)	(5 to 0)
Opcode	Rs	Rt	Rd	Shamt	Function

Figure 1. R-type (register format)

978-1-4799-6393-5/14/$31.00 ©2014 IEEE

Figure 1 shows the Instruction format of Register type. In this format there are total six fields, Opcode is of 6Bit, which is used to select the type of Instruction format, Source register (Rs), Target register (Rt) and Destination register (Rd) are of 5Bit each which are used for data storage. Shamt stands for Shift amount is of 5Bit used for shifting of data and Function field is of 6Bit used for selecting which type of function to be performed.

b) I-Type (Immediate Format)

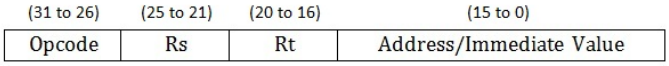

Figure 2. I-type (immediate format)

Figure 2 shows the Instruction format of Immediate type. In this format there are total four fields, Opcode is of 6Bit, which is used to select the type of Instruction format, Source register (Rs) and Target register (Rt) are of 5Bit each which are used for data storage. There is another field called as Address/Immediate Value field of 16Bit used for immediate data.

c) J-Type (Branch Format)

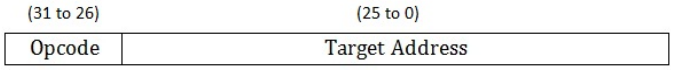

Figure 3. J-type (branch format)

Figure 3 shows the Instruction format of Branch type. In this format there are only two fields, Opcode is of 6Bit, which is used to select the type of Instruction format. Target address of 26Bit is used to specify on which address to branch.

d) I/O-Type (Input/Output Format)

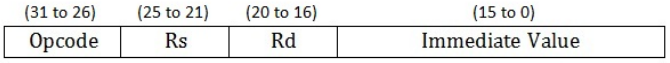

Figure 4. I/O-type (input/output format)

Figure 4 shows the Instruction format of Input/Output type. In this format there are total four fields, all the fields are similar to R-Type and I-Type formats. This type is used for read and write port data instructions.

Because of these four formats, execution of any instruction becomes faster, hence the proposed processor is also called as High Performance Processor.

B. MIPS INSTRUCTION SET

As proposed Processor is 32Bit MIPS RISC Processor, all the instructions are 32Bit in length which uses the four different types of formats. Instruction formats are vary from instruction to instruction. The design has eight 32Bit general purpose registers viz. A, B, C, D, E, F, G and H. Proposed MIPS RISC Processor supports more than 30 instructions as summarized in Table I. It supports addressing modes viz. Register addressing mode, Immediate addressing mode, Register indirect addressing, Implicit/Inherent/Implied addressing and direct addressing mode.

TABLE I
INSTRUCTION SET SUMMARY

Instruction	Description
ADD Rd, Rs, Rt	Rd<= Rs + Rt
ADDI Rt, Rs, Immediate	Rt<= Rs + Immediate
SUB Rd, Rs, Rt	Rd<= Rs - Rt
SUBI Rt, Rs, Immediate	Rt<= Rs - Immediate
AND Rd, Rs, Rt	Rd<= Rs and Rt
ANDI Rt, Rs, Immediate	Rt<= Rs and Immediate
OR Rd, Rs, Rt	Rd<= Rs or Rt
ORI Rt, Rs, Immediate	Rt<= Rs or Immediate
XOR Rd, Rs, Rt	Rd<= Rs xor Rt
XORI Rt, Rs, Immediate	Rt<= Rs xor Immediate
NOT Rd, Rs	Rd<= not Rs
MOVE Rd, Rs	Rd<= Rs
MOVI Rt, Immediate	Rt<= Immediate
MOVI Rt, D16	Rt<= 16Bit Immediate Lower
MOVI Rt, D32	Rt<= 16Bit Immediate Upper
MOVI Rt, D32, Rs	Rt<= 16Bit Immediate Upper and 16Bit Lower from Rs
RRA Rd, Rs	Rotate Right Arithmetic
RLA Rd, Rs	Rotate Left Arithmetic
RRL Rd, Rs	Rotate Right Logical
RLL Rd, Rs	Rotate Left Logical
IN immediate	Input Port<= Immediate
IN Rs	Input Port<= Rs
OUT Rd	Rd<= Output Port
PUSH	Pushed data into stack
POP	Pop data from stack
J Target	Unconditional Jump
JZ Target	Jump to target if result is zero
JP Target	Jump to target if result is Positive
JC Target	Jump to target if Carry
IN immediate	Input Port<= Immediate
IN Rs	Input Port<= Rs
OUT Rd	Rd<= Output Port
HLT	Halt the program execution

III. DATAFLOW

Data flow is achieved using data path of the hardware, which defines data flow. There is no clear difference between control and data. Operation code, operand, memory address, memory value, register address, register value, jump destination address and content are usually included in data, but control part consist of control signal of unit, time control signal and interrupt control signal, and these signals are not always defined clearly.

a) R-Format Data Path

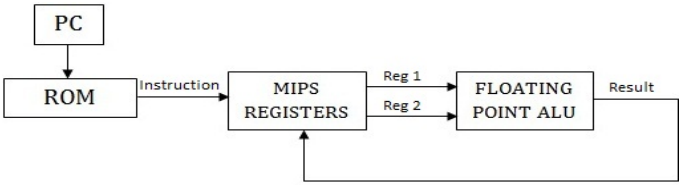

Figure 5. R-format instruction data path

In R-Format data path, instruction is fetch from memory in terms of opcode and analyzes instruction into different parts. For operation, two registers are specified by MIPS registers and then FP ALU executes instruction. Finally, after execution FP ALU outputs result to MIPS registers, which then stores the result into destination register. Figure 5 shows R-Format data path.

For example, ADD Rd, Rs, Rt instruction, which adds the contents of source register (Rs) with target register (Rt) and stores the result into destination register (Rd) (Rd <= Rs+ Rt). Data flow of this instruction shows as following: PC fetches ADD Rd, Rs, Rt instruction from memory. At first, the instruction reads the contents of the two registers Rs and Rt and put the values to FP ALU. After arithmetic operation is performed, FP ALU writes back result to register Rd. In this way, data flow of R-Format instruction is performed.

b) RI-Format Data Path

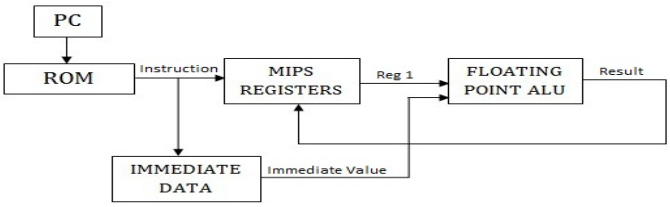

Figure 6. RI-Format instruction data path

RI-Format data path is similar to R-Format data path. The only difference is that the target register of R-format instruction is replaced by immediate value of RI-Format data path. The immediate is 32-bit value which is fed to FP ALU as the second operand. Finally, write-back result to MIPS registers. RI-Format data path is shown in Figure 6. This format includes ADDI Rt, Rs, Immediate instruction, SUBI Rt, Rs, Immediate instruction, ANDI Rt, Rs, Immediate instruction, etc.

For example, when ADDI Rt, Rs, Immediate instruction executes, PC fetches ADDI Rt, Rs, Immediate instruction from memory and Rt value is put to FP ALU. At the same time, immediate value is also put to FP ALU. Finally, after FP ALU completes addition of the two operands, it writes back result to MIPS registers. In this way, data flow of RI-Format instruction is performed. Other instructions such as subtraction, multiplication, division, ANDing, ORing, etc can be done in similar way as of addition.

c) Load Word Data Path

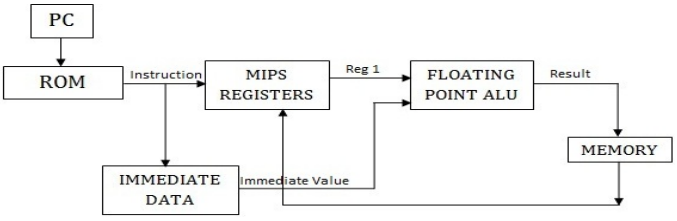

Figure 7. Load word instruction data path

Operation of load word data path is similar to I-Format data path. The difference is that result is written to memory in load word data but in I Format, result is written to register. In load word data path, data is fetched from memory and load it to MIPS registers. Load word data path is as shown in Figure. 7.

LW instruction is the only instruction in load word data path. It works as shown below: PC fetches the opcode of LW instruction from memory. Rs register is used to load data then it send Rs register value to FP ALU, at the same time, immediate value is send to the FP ALU. The result of addition of the two numbers is stored to the memory address, and then, it copies the contents of the memory address to MIPS registers. In this way, data flow of LW instruction is performed.

d) Memory Word Data Path

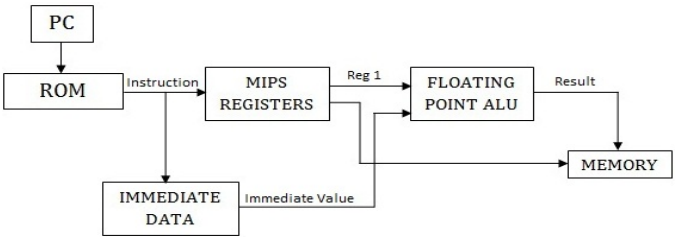

Figure 8. Memory word instruction data path

Memory word data path is similar to load word data path, the only difference is that destination of the execution is now memory not MIPS registers.

SW instruction is the only instruction in memory word data path. PC fetches the opcode of SW instruction from memory. Rs register is used to load data then it send Rs register value to FP ALU, at the same time, immediate value is send to the FP ALU. The result of addition of the two numbers is stored to the memory address. Memory Word instruction data path is as shown in Figure 8. In this way, data flow of SW instruction is performed.

e) J-Format Data Path

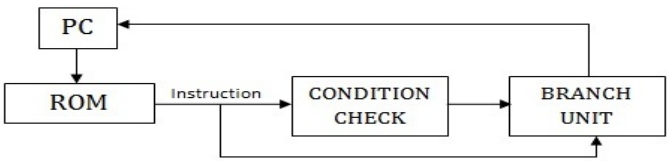

Figure 9. J-format instruction data path

Figure 9 shows the J-Format Instruction data path. In J-Format instruction data path, program counter fetches the opcode of the instruction from the memory, and then there are two ways to execute that instruction. First is Unconditional branch and the next is conditional branch. In Unconditional branch the pointer is jump to the target address directly but in conditional branch, it has to satisfy the condition then the branch can be done.

978-1-4799-6393-5/14/$31.00 ©2014 IEEE

For example, J Target instruction, it is an unconditional branch instruction. In this instruction, first the opcode is fetched from the memory by PC then it is fed to the branch unit directly and then it branches to program counter where the Target address of the instruction is present. In this way, J-Format instruction data path works.

f) I/O-Format Data Path

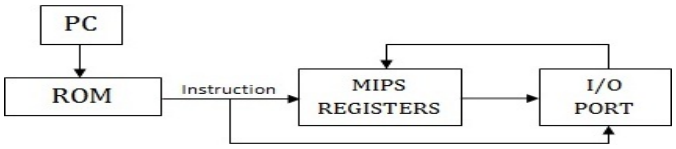

Figure 10. I/O-format instruction data path

Figure 10 shows the I/O-Format data path.In I/O-Format instruction data path, program counter fetches the opcode of the instruction from the memory, and then there are two ways to execute that instruction. First is register data transfer and the next is immediate data transfer.

For example, instruction IN Rs, in this instruction first the opcode of the instruction is fetch from the memory by PC, then it is fed to MIPS registers block and then register data is fed to I/O port block. In this way, I/O-Format instruction data path is performed. The same procedure is done for immediate data, the only difference is that register is replace by immediate data.

IV. PIPELINE DESIGN

Decomposition of Pipeline increases throughput rate of the instructions. Clock cycle is decided by the running time of slowest stage. In general, pipeline includes five stages: instruction fetch (IF), instruction decoder (ID), execution (EXE), memory/ IO (MEM) and write-back (WB) stage.

a) Instruction Fetch (IF)

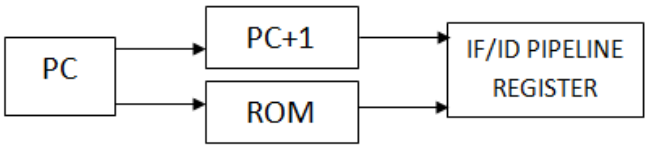

Figure 11. Instruction fetch stage

The first stage of a pipelining is instruction fetch (IF) stage. The operation of the IF stage starts when the program counter (PC) is sent out to fetch the instruction from memory. Instruction and PC is stored in IF/ID pipeline register as temporary memory for next clock cycle.

IF stage generally depends on program counter's (PC) current value. Processor fetches instruction from memory based on PC value and then PC incremented by 1 automatically. Finally, all these information is send to IF/ID pipeline register then it is fed to decoder unit. IF stage operation is shown in Figure 11.

b) Instruction Decoder (ID)

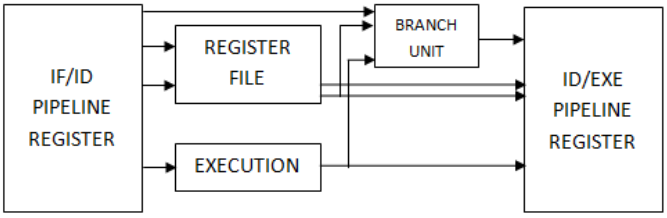

Figure 12. Instruction decoder stage

When the instruction is fetched from the IF stage, the opcode is sent to the decoder unit. ID stage sends control command to other units of processor based on opcode of the instruction. Read register fetches data from MIPS registers. Branch unit is also included in ID stage. Input of ID stage is from IF stage.

ID stage includes four instruction formats R-Type, I-Type, J-Type and I/O-Type. Depending on the opcode of the instruction, the operation will be performed using above four formats. Figure 12 shows ID stage operation.

c) Execution (EXE)

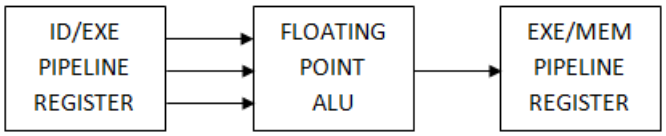

Figure 13. Execution stage

After the instruction decode stage, the decoded opcode is sent to the execution stage. EXE stage executes arithmetic and logical operations. Main part of EXE stage is FP ALU. Function of EXE stage is to perform operations such as addition, subtraction, ANDing, ORing, etc. FP ALU sends result to EXE/MEM pipeline register. Figure 13 shows operation of EXE stage.

d) Memory and I/O (MEM)

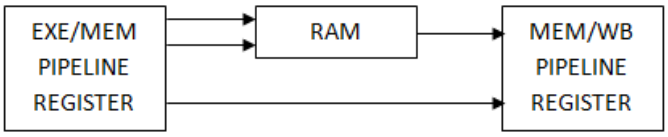

Figure 14. Memory and i/o stage

The function of memory stage is to load and store values to and from memory. Another function is to input data to processor and output data from the processor. If instruction is not memory or IO instruction, then the result is sent to WB stage. Storing data in destination register is the main function after result is calculated. MEM stage operation is as shown in Figure 14.

978-1-4799-6393-5/14/$31.00 ©2014 IEEE

e) Write-Back (WB)

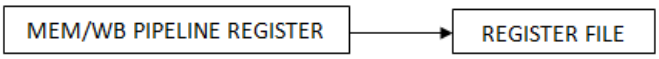

Figure 15. Write-back stage

The Write-Back stage writes result, store data and input data to and from register file (MIPS registers). The aim of WB stage is to write data to destination register which is available in MIPS registers. For example, ADD Rd, Rs, Rt instruction gives result in Rd register to improve the execution speed. Figure 15 shows WB stage operation.

V. EXPERIMENTAL RESULTS

A. RTL View

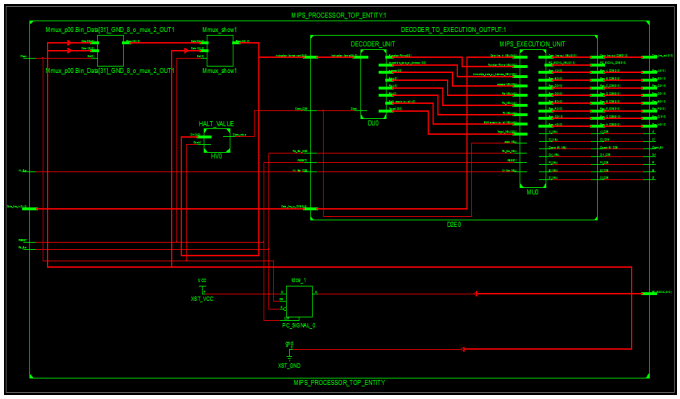

Figure 16. RTL View of MIPS RISC Processor

Figure 16 shows the RTL (Register Transfer Logic) view of 32Bit MIPS RISC Processor. It comprises of Instruction fetch unit, Instruction decoder unit, execution unit and memory unit. Function of instruction fetch unit is to fetch opcode from memory using PC and give it to instruction decoder unit. Function of instruction decoder unit is to receive the opcode from instruction fetch unit and depending on opcode, instruction format is selected and then it is fed to execution unit for execution of the instruction. Function of execution unit is to perform the specific operation according to the opcode of the instruction. It consists of Floating point ALU (FP ALU), Flag register, Instruction set, MIPS registers and operation select. Function of memory unit is to store the opcodes and to retrieve the data to and from memory.

B. Simulation Results

Figure 17. Simulation result of R-type instruction format

Figure 17 shows the simulation result of R-Type instruction format, instruction ADD D, A, B is performed. First register A data and register B data is readed and then addition of this two registers is stored into the destination register D.

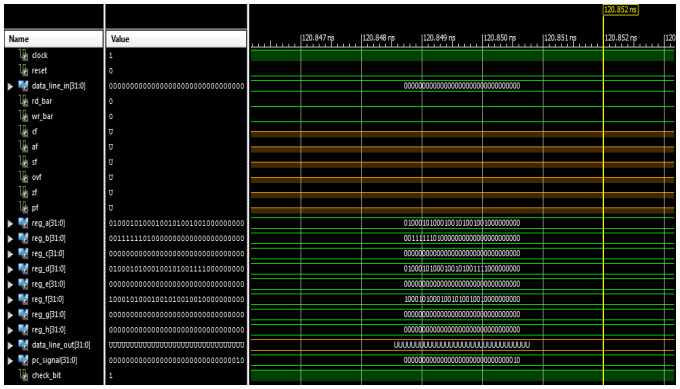

Figure 18. Simulation result of I-type instruction format

Figure 18 shows the simulation result of I-Type instruction format, instruction MVI F, IMMEDIATE is performed. Immediate data is readed and is stored into the destination register F.

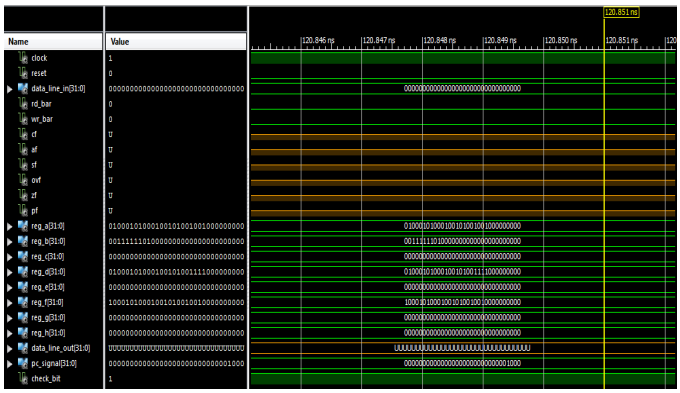

Figure 19. Simulation result of J-type instruction format

Figure 19 shows the simulation result of J-Type instruction format, instruction J TARGET is performed. According to the opcode the jump is done on target address "00000000000000000000000000001000".

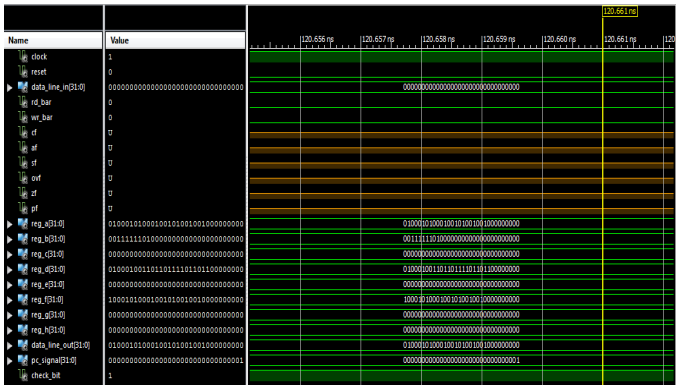

Figure 20. Simulation result of I/O-type instruction format

Figure 20 shows the simulation result of I/O-Type instruction format, instruction IN A is performed. First register A data is readed and then it is stored to the port.

Table II below shows the comparison of various latest RISC Processors with the proposed MIPS RISC processor.

TABLE II
PERFORMANCE TABLE

RISC Processor Features	PowerPC 755/745 RISC Microprocessor (2013)	AM1806 ARM Microprocessor (2014)	MPC 7457 RISC Microprocessor (2013)	AM1808 ARM Microprocessor (2014)	STM32F205xx STM32F207xx (2013)	ADSP-BF592 (2013)	Sitara AM335x ARM Cortex-A8 Microprocessors (2013)	Proposed Design
Field Programmable	No	No	No	No	No	No	No	Yes
Input Voltage (V)	2.0	1.8	1.3	1.8	1.8	1.1	1.5	1.0
Power on chip (W)	5.4	___	___	___	___	___	___	0.546
Ambient Temperature (Min) (°C)	-40	-40	___	-40	-40	-40	-40	-65
Junction Temperature (Max) (°C)	110	90	105	90	105	95	90	125
Leakage Current (µA)	10	___	30	___	1	10	10	15
Storage Temperature (°C)	-65 to 150	-55 to 150	-55 to 150	-55 to 150	-65 to 150	-65 to 150	-55 to 150	-65 to 150
Max. CPU Frequency (MHz)	400	456	1267	456	120	___	___	1350
Max. Delay (ns)	2.5	2.19	0.789	2.19	8.33	___	___	0.741

VI. CONCLUSION

In this research, we use VHDL to describe the system and uses top-down design method in which initially we design Instruction Fetch unit, then Decoder unit, then Execution unit, finally memory unit. The hierarchy of the design is very clear. It is easy to edit and debug. The modules synthesized and simulated are MIPS Instruction format, Floating point ALU, MIPS Instructions Set, Flag register, MIPS Registers, Operation select, ROM and RAM. More than 30 instructions are successfully designed. The design has been synthesized and simulated using Xilinx 13.1i ISE Simulator. All the goals were achieved and simulation shows that processor is working perfectly. Pipelined design has been achieved with less clock cycles per instruction. Proposed 32Bit MIPS RISC Processor has delay of 0.741ns and maximum operating frequency of 1.350 GHz. So, the proposed processor can be called as a high performance processor.

REFERENCES

[1] Mrs. Rupali S. Balpande, Mrs.Rashmi S. Keote, Design of FPGA based Instruction Fetch & Decode Module of 32-bit RISC (MIPS) Processor, 2011 International Conference on Communication Systems and Network Technologies, 978-0-7695-4437-3/11, 2011 IEEE.

[2] Mamun Bin Ibne Reaz, MEEE, Md. Shabiul Islam, MEEE, Mohd. S. Sulaiman, MEEE, A Single Clock Cycle MIPS RISC Processor Design using VHDL, ICSE2002 Proc. 2002, Penang, Malaysia, 0-7803-7578-S/02/S, 2002 IEEE.

[3] Kui YI, Yue-Hua DING, 32-bit RISC CPU Based on MIPS Instruction Fetch Module Design, 2009 International Joint Conference on Artificial Intelligence, 978-0-7695-3615-6/09, 2009 IEEE.

[4] Rohit Sharma, Vivek Kumar Sehgal, Nitin Nitin1, Pranav Bhasker, Ishita Verma, Design and Implementation of a 64-bit RISC Processor using VHDL, UKSim 2009: 11th International Conference on Computer Modelling and Simulation, 978-0-7695-3593-7/09, 2009 IEEE.

[5] Pravin S. Mane, Indra Gupta, M. K. Vasantha, Implementation of RISC Processor on FPGA, 1-4244-0726-5/06, 2006 IEEE.

[6] Shuchita Pare, Dr. Rita Jain, 32Bit Floating Point Arithmetic Logic Unit ALU Design and Simulation, Dec 2012, IJETECS.

[7] Bai-ZhongYing, Computer Organization, Science Press, 2000.11.

[8] Wang-AiYing, Organization and Structure of Computer, Tsinghua University Press, 2006.

[9] Wang-Yuan Zhen, IBM-PC Macro Asm Program, Huazhong University of Science and Technology Press, 1996.9.

[10] MIPS Technologies, Inc. MIPS32™ Architecture For Programmers Volume II: The MIPS32™ Instruction Set June 9, 2003.

[11] Zheng-WeiMin, Tang-ZhiZhong, Computer System Structure (The second edition), Tsinghua University Press, 2006.

[12] Mr. Sagar P. Ritpurkar, Prof. Mangesh N. Thakare, Prof. Girish D. Korde, Review on 32Bit MIPS RISC Processor using VHDL, International Conference on Advances in Engineering & Technology – 2014 (ICAET-2014), PP 46-50, IOSR.

[13] PowerPC 755/745 RISC Microprocessor 2013 datasheet, e2v semiconductors SAS.

[14] AM1806 ARM Microprocessor 2014 datasheet, TEXAS INSTRUMENTS.

[15] AM1808 ARM Microprocessor 2014 datasheet, TEXAS INSTRUMENTS.

[16] ADSP-BF592 Blackfin Embedded Processor 2013 datasheet, Analog Devices.

[17] Sitara AM335x ARM Cortex-A8 Microprocessors (MPUs) 2013 datasheet, TEXAS INSTRUMENTS.

[18] MPC7457 RISC Microprocessor 2013 datasheet, Freescale Semiconductor.

[19] STM32F205xx, STM32F207xx 2013 datasheet, STMicroelectronics.