International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347 - 6435 (Online) Volume 2, Issue 4, April 2014)

VHDL Design and Implementation of ASIC Processor Core by Using MIPS Pipelining

G.Triveni[1], Aswini Kumar Gadige[2]

[1]Department of ECE, Kottam College Of Engineering, Kurnool
[2]Assistant professor, department of ECE, Kottam College of Engineering

***Abstract*— The design of 32 bit ASIC processor core by using Microprocessor Without Interlocked Pipeline Stages (MIPS) with the essence of Super Harvard Architecture (SHARC) is proposed. It was implemented in Very High Speed Integrated Circuit HDL language to reduce the instruction set present in the programmable memory. In this processor design, the sub module adder is designed as advanced Carry Select Adder (CSLA), instead of Ripple Carry Adder (RCA) which reduces the clock period, and response time of system. MIPS pipelining methodology will support the parallel execution of instructions. As the result processor turnaround time value decreases and throughput increases. We have implemented the system in Xilinx ISE 9.2i tool. This design is possible to be dumped on Spartan 3E FPGA kit.**

***Keywords*— MIPS, SHARC, ASIC, VHDL**

I. Introduction

Nowadays, ASIC based microprocessors are used in a variety of electronic gadgets such as personal computers, cell phones, and robots. These microprocessor includes millions of components which does the memory transfer from one to the other.

As per the growing technology, the size of processor is smaller and so many portable devices are manufactured. Because memory was expensive in old days, designer of digital system enhanced complication of instruction to reduce program length. Tendency of complication instruction design brought up one traditional instruction design style, which is named as "Complex Instruction Set Computer-CISC" structure. Comparing to CISC, MIPS Processor's CPU have more advantages, such as faster speed, simplified structure, easier implementation. MIPS is an extensive use in embedded system. Due to this reason, developing ASIC with MIPS structure is necessary choice.

Addition usually impacts widely the overall performance of digital systems and a crucial arithmetic function. In electronic applications adders are most widely used. Applications where these are used are multipliers, DSP to execute various algorithms like FFT, FIR and IIR. Wherever concept of multiplication comes adders come in to the picture.

As we know millions of instructions per second are performed in microprocessors. So, speed of operation is the most important constraint to be considered while designing multipliers. Due to device portability miniaturization of device should be high and power consumption should be low. Devices like Mobile, Laptops etc. require more battery backup. So, a VLSI designer has to optimize these three parameters in a design. These constraints are very difficult to achieve so depending on demand or application some compromise between constraints has to be made. Ripple carry adders exhibits the most compact design but the slowest in speed. Whereas carry look ahead is the fastest one but consumes more area.

Carry select adders act as a compromise between the two adders. In 2002, a new concept of hybrid adders is presented to speed up addition process by Wang et al. that gives hybrid carry look-ahead/carry select adders design. In 2008, low power multipliers based on new hybrid full adders is presented in.

The main aim of the project is to give a view regarding a processor core design that includes essence of both SHARC &MIPS that leads to fastest processor for ASIC applications.

In this processor design Ripple Carry Adder (RCA) is replaced by a Carry Select Adder (CSLA), which reduces the clock period. The main module processor, it's sub module ALU, and it's sub module CSLA design flow is as shown in the Fig: 1.1. The objective of this work is not the development of a new architecture, this work aims to study and combine the different advanced methodologies to design ASIC processor core which reduces the turnaround time to give high throughput. This result leads towards design of high speed ASIC processor. To reach this main objective of this paper we have designed the architecture (Block diagram) of 32 bit MIPS processor and individual modules of the processor are implemented by using VHDL (Behavioral and Dataflow functional description), Individual modules are integrated by using Structural architecture style of VHDL, and instead of Ripple Carry Adder (RCA) a Carry Select Adder (CSLA) is used.

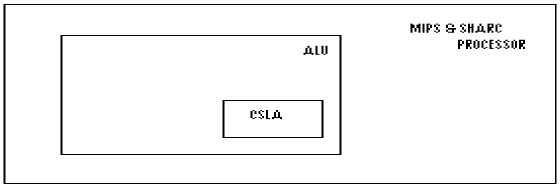


Fig: 1.1 Processor module design

The rest of the paper is organized as follows. In section 2, a brief about SHARC, MIPS Pipelining methodology is discussed. In the same section CARRY Select Adder (CSLA) is introduced. In Section 3 working principle and block diagram representation of new designed ASIC processor is explained. Section 4 provides the results obtained. Section 4 concludes the paper.

II. Design Architecture

(a) SHARC:

There are basically three types of digital computer architecture as follows.

1.Von Neumann 2.Harvard Architecture 3.SHARC

The **von Neumann Architecture** is named after the mathematician and early computer scientist John von Neumann. Von Neumann machines have shared signals and memory for code and data. Thus, the program can be easily modified by itself since it is stored in read-write memory. This architecture has only one bus which is used for both data transfers and instruction fetches, and therefore data transfers and instruction fetches must be scheduled - they cannot be performed at the same time.

The name **Harvard Architecture** comes from the Harvard Mark I relay-based computer. The most obvious characteristic of the Harvard Architecture is that it has physically separate signals and storage for code and data memory. It is possible to access program memory and data memory simultaneously.

SHARC stands for Super Harvard Architecture/ Super Harvard Single Chip Computer. It means more than one set of addresses and data buses. In this program memory configurable as program and data memory parts, which supports processing instruction level parallelism as well as memory access parallelism as shown in the Fig.2.1 This SHARC processor has built-in support for loop control. Up to 6 levels may be used, avoiding the need for normal branching instructions and the normal bookkeeping related to loop exit.

SHARC processor family targets applications ranging from consumer, automotive, and professional audio, to industrial, test and measurement, and medical equipment.

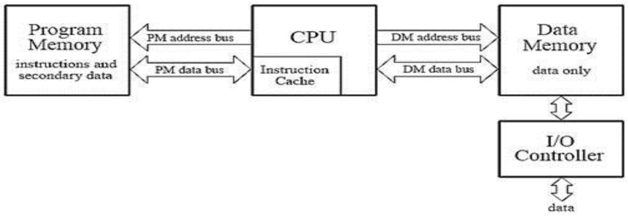


Fig: 2.1 Super Harvard Architecture

b) MIPS PIPELINING:

MIPS (originally an acronym for **Microprocessor without Interlocked Pipeline Stages**) is a reduced instruction set computer (RISC) instruction set architecture (ISA) developed by MIPS Computer Systems (now MIPS Technologies). Another informal full name is Millions of instructions per second. It has load store architecture. The MIPS pipelined processor involves five steps as shown in the Fig: 2.2. The division of an instruction into five stages implies a five-stage pipeline. The execution of an instruction in a processor can be split up into 5 stages as follows.

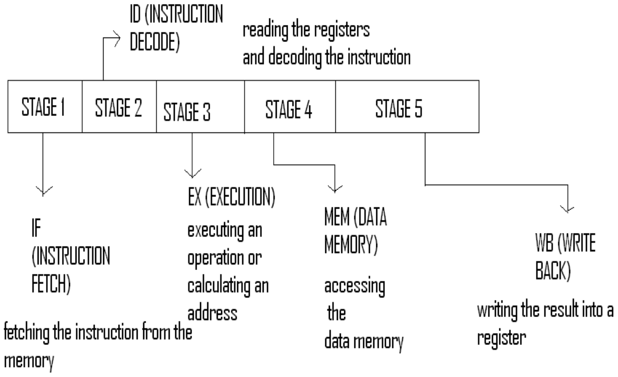

ID (INSTRUCTION DECODE)
reading the registers and decoding the instruction
STAGE 1
STAGE 2
STAGE 3
STAGE 4
STAGE 5
IF (INSTRUCTION FETCH)
fetching the instruction from the memory
EX (EXECUTION)
executing an operation or calculating an address
MEM (DATA MEMORY)
accessing the data memory
WB (WRITE BACK)
writing the result into a register

Fig: 2.2 MIPS five stage pipelining

The MIPS Instruction format allows the CPU to load up the instruction and the data it needed in a single cycle, whereas some other processors requires separate cycles to load the opcode and the data. This was one of the major performance improvements that MIPS offered. Design flow of MIPS pipelining while executing the instructions is as shown in the Fig.2.3

In this design we can avoid interlocks while executing the instructions. Hence parallel execution of instruction will results reduction in turnaround time. This concept can be illustrated by using Fig.3.1

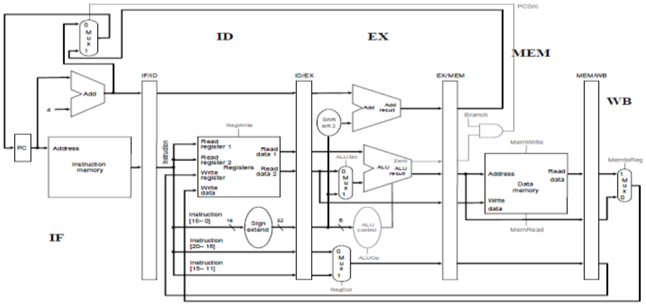


Fig: 2.3MIPS Pipelined Data path

This MIPS technology delivers low power consumption. It requires smaller silicon area than other CPU's. Several optional extensions are available. The MIPS instruction format allows the CPU to load up the instructions and the data it needed in a single cycle.

(C) CARRY SELECT ADDER:

Binary addition is a fundamental operation in most Digital Circuits. There are a variety of adders, each has certain performance. Each type of adder is selected depending on where the adder is to be used. In digital adders, the speed of addition is limited by the time required to transmit a carry through the adder. Carry Select Adder (CSLA)[2] is one of the fastest adders used in many data-processing processors to perform fast arithmetic functions.

The basic idea of the carry-select adder is to use blocks of two ripple-carry adders, one of which is fed with a constant 0 carry-in while the other is fed with a constant 1 carry-in as shown in the Fig.2.5.

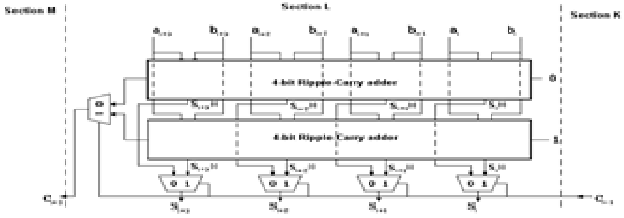


Fig: 2.5 Carry Select Adder

III. WORKING PRINCIPLE

(a)Interlocks in Processor:

In early days interlocks were present in the processor while executing the instructions. Because of these interlocks there should be a defined clock period for the execution of instruction in each stage of pipeline. Due to this execution of instruction may not be completed at individual stages which may lead a garbage value.

- Interlocks avoid the data hazards.
- But due to interlocks the execution time will increase.

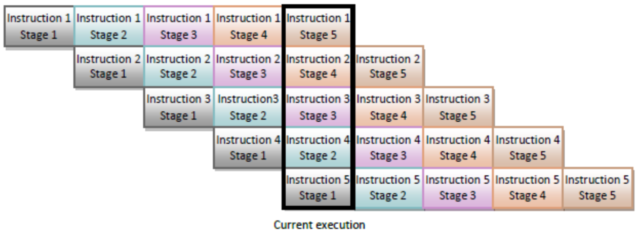
(b)Without Interlocks:

So to avoid this situation we are using advanced processor technology called MIPS, which executes instructions without interlocks at individual stages. Due to this feature an instruction can be executed completely at each stage. But the drawback in this is there may be an increment in execution time which affects the overall speed performance of the processor. For this problem the solution is pipelining technology, which supports parallel execution of instructions.

(c)Why pipelining?

Pipelining, a standard feature in MIPS processors, is a technique used to improve both clock speed and overall performance. Pipelining allows a processor to work on different steps of the instruction at the same time, thus more instruction can be executed in a shorter period of time. When the Stanford team was researching the RISC architecture, pipelining was a well known technique, and obviously nowadays it is not an exclusive part of the RISC domain. But it hadn't been developed into its full potential.

Pipelining can potentially reduce the number of cycles per instruction by a factor equal to the depth of the pipeline, but fulfilling this potential requires the pipeline always be filled.



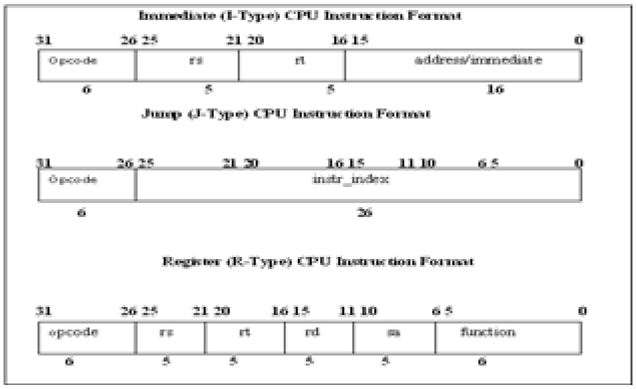
Instruction 1 Stage 1	Instruction 1 Stage 2	Instruction 1 Stage 3	Instruction 1 Stage 4	Instruction 1 Stage 5				
	Instruction 2 Stage 1	Instruction 2 Stage 2	Instruction 2 Stage 3	Instruction 2 Stage 4	Instruction 2 Stage 5			
		Instruction 3 Stage 1	Instruction3 Stage 2	Instruction 3 Stage 3	Instruction 3 Stage 4	Instruction 3 Stage 5		
			Instruction 4 Stage 1	Instruction 4 Stage 2	Instruction 4 Stage 3	Instruction 4 Stage 4	Instruction 4 Stage 5	
				Instruction 5 Stage 1	Instruction 5 Stage 2	Instruction 5 Stage 3	Instruction 5 Stage 4	Instruction 5 Stage 5

Fig: 3.1. Parallel execution of instructions to reduce delay

(d)MIPS Instruction Set:

MIPS instructions [1] can be classified according to their format, and the related diagrams are as shown in the Fig: 3.2.

- *R-Type:* contains all instructions that do not require an immediate value or memory address to specify an operand. This includes arithmetic and logical instructions.
- *I-Type:* contains the instructions that must operate with a 16-bit immediate operand.
- *J-Type:* composed of two direct J ump instructions. These instructions require a 26-bit memory address to specify their operands



Immediate (I-Type) CPU Instruction Format

31 26	25 21	20 16	15 0
Opcode	rs	rt	address/immediate
6	5	5	16

Jump (J-Type) CPU Instruction Format

31 26	25 0
Opcode	instr_index
6	26

Register (R-Type) CPU Instruction Format

31 26	25 21	20 16	15 11	10 6	5 0
opcode	rs	rt	rd	sa	function
6	5	5	5	5	6

Fig: 3.2. MIPS Instruction Types Formats

(e)Necessity of SHARC:

SHARC architecture consists of "Instruction Cache" block, which stores the data related to recently executed instructions. If again same data is required for the processor to execute any other instruction, then the processor will check in this cache for hit(compatibility between current executing instruction requirement with respect to data stored in the Instruction Cache), if hit occur then the processor can avoid all the remaining modules involvement in the execution of that particular instruction. This overall process reduces the execution time which leads towards high speed performance.

SHARC instructions may contain a 32-bit immediate operand. Instructions without this operand are generally able to perform two or more operations simultaneously. SHARC processor integrates large memory arrays and Application Specific peripherals designed to simply product development and reduce time to market

(f)Design flow:

The architecture (Block diagram) of 32 bit MIPS processor is designed [7]. This processor consists of 17 sub modules in it.

Each individual modules of the processor is implemented using VHDL (Behavioral and Dataflow functional description). Individual modules are integrated using Structural architecture style of VHDL and then Simulated.

Here we concentrated on the sub module called ALU in which general Ripple Carry Adder (RCA) is replaced by a high speed Carry select Adder (CSLA) (Fig:1.1).

It is because; the MIPS microprocessor instructions only go through the ALU once and never after a memory access. If an operand from memory is required it is loaded into the register bank first and only then used in subsequent operations. This allows the creation of a very simple architecture which is easy to pipeline. The designed 32 bit MIPS architecture (Block Diagram) by using five stage pipelining technology is as shown in the Fig: 3.3

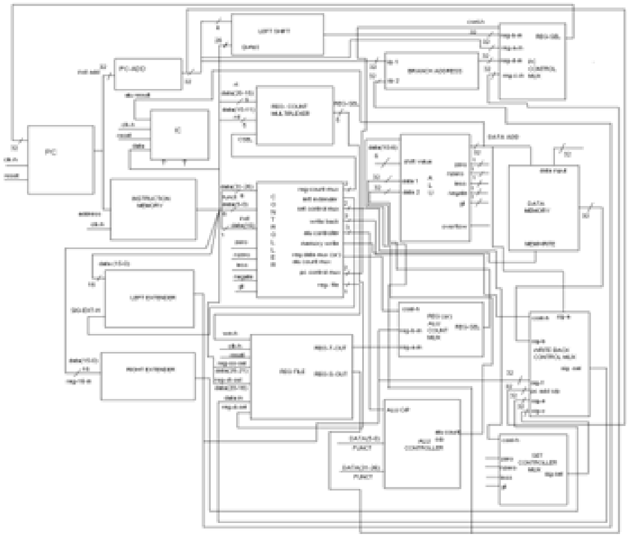


Fig: 3.3 Block Diagram Representation of MIPS Architecture

IV. Results

In this section simulation of MIPS instruction types is discussed and their related results are produced.

(i)Immediate Type:

I-type is used for the Load and Stores instructions. A load instruction loads a value from memory into a register. A store instruction stores a value from a register to memory. The load and store instructions use the sum of the offset value in the address/immediate field and the base register in the $rs field to address the memory.

Branch instructions (e.g. beq, bne) are I-type instructions which use the addition of an offset value from the current address in the address/immediate field along with the program counter (PC) to compute the branch target address; this is considered PC-relative addressing.

Arithmetic Addition:

Mnemonic – addi $r1, $r2, 32.

$r1 = $r2 + 32.

Here the binary instruction for immediate addition is

Opcode	rs	rd	immediate value		
001001	00011	01001	00000	00000	100000

31 26 **r2** 21 **r1** 15 0

Load operation load the data from data memory to register as Shown in Fig: 6.1

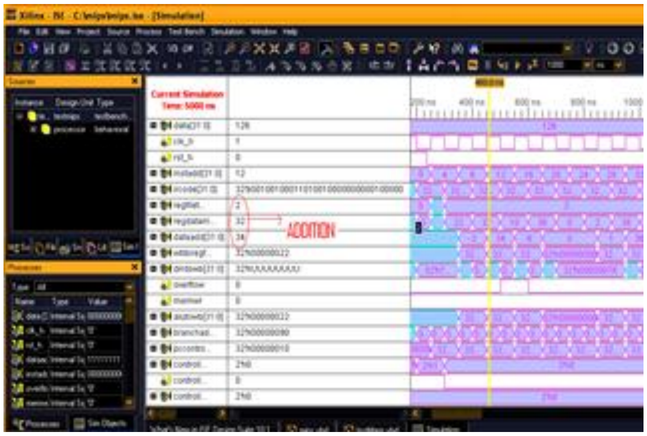

Xilinx - ISE - C:\mips\mips.ise - [Simulation]
Current Simulation Time: 5000 ns
ADDITION

Fig: 6.1.Immediate type Arithmetic Addition

(ii)Jump Type:

J-type is used for the Jump instructions.

Unconditional Jump:

Mnemonic – j 44

Here the binary instruction for store operation is

Opcode			Instruction index		
000001	00000	00000	00000	00000	001011

31 26 15 0

Jump operation jumps to the target address as shown in Fig: .6.2

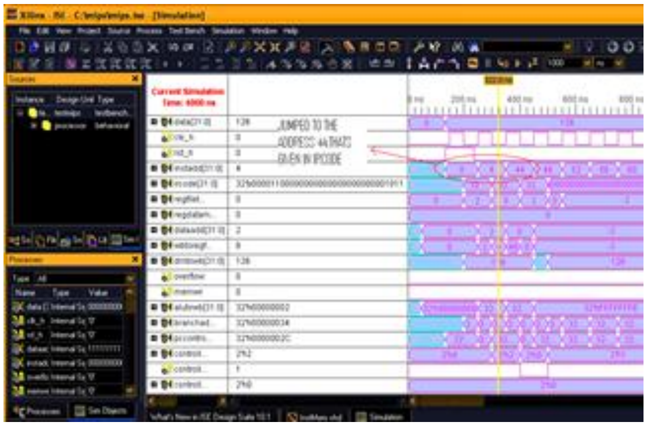

Xilinx - ISE - C:\mips\mips.ise - [Simulation]
Current Simulation Time: 4000 ns
JUMPED TO THE ADDRESS 44 THAT'S GIVEN IN IPCODE

Fig: 6.2. Jump type unconditional

Conditional Jump:

Mnemonic- JZ 24

Here the binary instruction for store operation is

Opcode			Instruction index		
001110	00000	00000	00000	00000	000000

31 26 15 0

If the result of previous instruction is zero then jump to the address 24 as shown in the Fig: 6.3

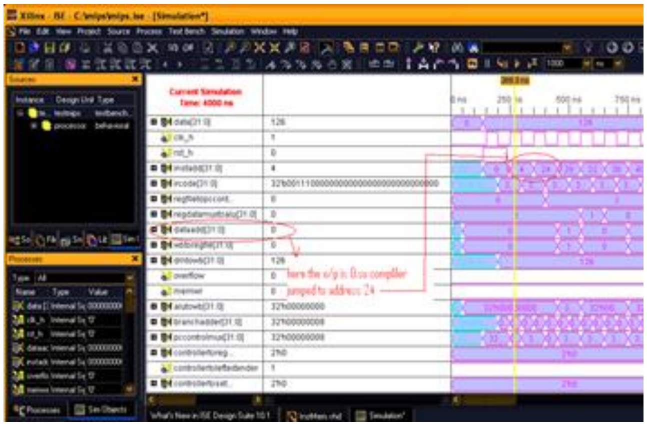

Xilinx - ISE - C:\mips\mips.ise - [Simulation*]
Current Simulation Time: 4000 ns
here the o/p is 0,so compiler jumped to address 24

Fig: 6.3. Jump type conditional

(iii)Register Type:

R-type is used for Arithmetic instructions. Arithmetic instructions or R-type include: ALU Immediate (e.g. addi), three-operand (e.g. add, and, slt), and shift instructions (e.g. sll, srl).

Complement Condition:

Mnemonic- NOT1 $r1 $r2 $r3
$r1 = complement of $r2

Here the binary instruction for store operation is

Opcode	rs	rd	rt	sa	function
001110	00000	00000	00000	00000	000000

31 26 15 0

Here the data 1 i.e., 00000000000000000000000000000010 is complemented as 11111111111111111111111111111101 as shown in the Fig: 6.4

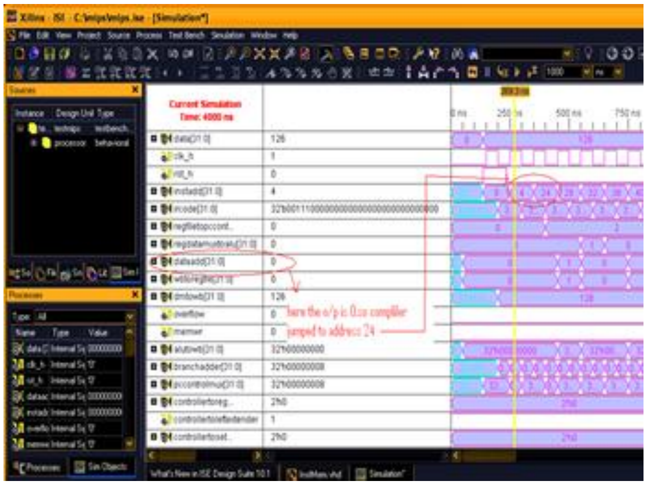

Xilinx - ISE - C:\mips\mips.ise - [Simulation*]
File Edit View Project Source Process Test Bench Simulation Window Help
Current Simulation Time: 4000 ns
0 ns
250 ns
500 ns
750 ns
Instance
Design Unit Type
processor
behavioral
data[31:0]
clk_h
instadd[31:0]
overflow
memwr
Processes
Sim Objects
What's New in ISE Design Suite 10.1
Simulation*
here the o/p is 0.so compiler jumped to address 24

Fig: 6.4. Register type complement condition

V. CONCLUSION

This paper describes the design and implementation of the 32-bit SHARC architecture ASIC processor core by using Microprocessor Without Interlocked Pipeline Stages.

The processor core is designed using the VHDL and verified its instruction sets In this the SHARC architecture allows for a wide range of implementations, from cost-constrained microcontrollers to supercomputers. The usage of Carry Select Adder (CSLA) leads reduction in clock period and response time, which increases the speed performance of the processor and results high throughput. In this project the IP core can be used anywhere, for example this architecture is generally used in ASIC processors.

Acknowledgements

The author would like to thank her guide Aswini Kumar Gadige, Assistant professor, Department of Electronics and Communication Engineering (VLSI BRANCH), Kottam College of Engineering, JNTU Anantapur University, for his valuable suggestions for the implementation of this project.

REFERENCES

[1] Leonardo Augusto Casillo, Ivan Saraiva Silva,IEEE 2012, A Methodology to adapt data path architecture to a MIPS-1 model.

[2] B. Ramkumar and Harish M Kittur, ―Low Power and Area-Efficient Carry Select Adder‖ IEEE Transactions On Very Large Scale Integration (VLSI) Systems, VOL. 20, NO. 2, FEBRUARY 2012

[3] Gyanesh Savita, Vijay Kumar Magraiya, Gajendra Kulshrestha, Vivek Goyal, Designing of Low Power 16-Bit Carry Select Adder with Less Delay in 45 nm CMOS Process Technology.

[4] Donghoon Lee, Kyunsoo Kwon, Wontae Choi, IEEE 2008, a study on 16 bit DSP. Pusan national University.

[5] CS.Wallace, "A suggestion for fast multipliers", IEEE Trans, On Electronic computers, Vol EC, no.3 pp .14-17,Feb.

[6] MIPPS Technologies. MIPS_32 Architecture For Programmers. Vol.II: The MIPS_32 Instruction Set,2008.

[7] Kui YI, Yue Hia DING, A study on MIPS, IEEE 2009, Wuhan Polytechnic University.

[8] http://en.wikipedia.org/wiki/MIPS_architecture.