Design & Implementation of 16-bit RISC Processor

O. V. P. R. Shiva Kumar[1] P. V. S. K. Pavan[2] Mithin Varghese[3] T. Madhulika Reddy[4]
[1,2,3,4]Department of Electronics & Communication Engineering
[1,2,3,4]Geethanjali College of Engineering & Technology, Cheeryal, Keesara, Medchal, Hyderabad, India

*Abstract— **Objectives:*** This paper presents the design of a 16 bit Reduced Instruction Set Computing (RISC) processor. We have used modified Harvard architecture that uses separate memories for its instruction & data memory response where as in the other architecture by von Neumann, has only one shared memory for instruction and data, with one data bus and address bus with between data memory & processor memory. It is pipelined to a depth of 5 stages with stages corresponding to: Instruction Fetch, Instruction Decode and Register Fetch, ALU Operation, Memory Operation, and Register File Write.

Findings: Various functional blocks of the processor such as the Control Unit, Instruction Decoder, Instruction Register unit and Arithmetic and Logical Unit (ALU) are designed using the Cadence tool.

Key words: 16-bit RISC Processor

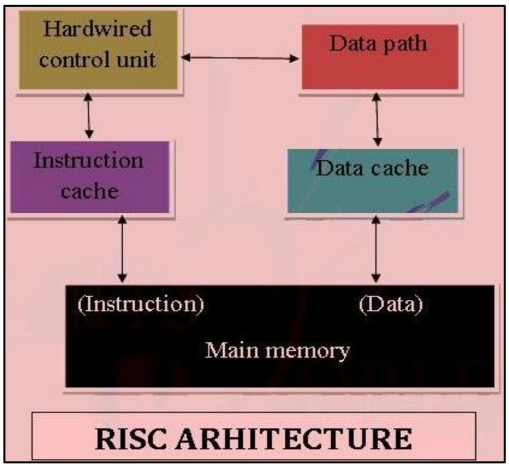


Fig. 1: RISC Architecture

I. Introduction

Microprocessor is a profound example of the Very Large Scale Integration (VLSI) industry, which takes the input data in the form of 0’s and 1’s and processes it according to the instructions. It is expected to yield the output according to the specified instruction at the maximally possible speed. A set of instructions (program) or group of programs (software) are written to the microprocessor, to perform the task and compute the output. A physical set of hardware modules accomplish the said purpose. The primarily used such modules are the Arithmetic Logic Unit (ALU), Control Unit, Registers and Instruction Execution Unit. The design of an efficient hardware architecture involves the capability to operate with maximum performance even while consuming lower power and reduced silicon area. The emergence of the Reduced Instruction Set Computing (RISC) processor was an evolution in the computing research platform during the recent years. It has a very simple and denied architecture with fewer fixed length instructions, as compared to the Complex Instruction Set Computing (CISC) which, on the other hand had complex architecture with more number of instructions in the instruction set. The RISC processor has the following special features:

- All the instructions are of fixed length
- All the instructions are executed in single clock cycle
- Microcode is not allowed and explicit instructions are used.

II. RISC Architecture

RISC (Reduced Instruction Set Computer) is used in portable devices due to its power efficiency. For Example, Apple iPod and Nintendo DS. RISC is a type of microprocessor architecture that uses highly-optimized set of instructions. RISC does the opposite, reducing the cycles per instruction at the cost of the number of instructions per program Pipelining is one of the unique feature of RISC. It is performed by overlapping the execution of several instructions.

It has a high performance advantage over CISC.

III. Why RISC?

Various attempts have been made to increase the instruction execution rates by overlapping the execution of more than one instruction since the earliest day of computing. The most common ways of overlapping are pre-fetching, pipelining and superscalar operation.

A. *Pre-Fetching*

The process of fetching next instruction or instructions into an event queue before the current instruction is complete is called pre-fetching. The earliest 16-bit microprocessor, the Intel 8086/8, pre-fetches into a non-board queue up to six bytes following the byte currently being executed thereby making them immediately available for decoding and execution, without latency.

B. *Pipelining*

Pipelining instructions means starting or issuing an instruction prior to the completion of the currently executing one. The current generation of machines carries this to a considerable extent. The PowerPC 601 has 20 separate pipeline stages in which various portions of various instructions are executing simultaneously.

C. *Superscalar Operation*

Superscalar operation refers to a processor that can issue more than one instruction simultaneously. The PPC 601 has independent integer, floating-point and branch units, each of which can be executing an instruction simultaneously.

IV. RISC Processor Resources

A. *Program Counter (PC)*

Program counter contain the next instruction address to be executed. This address will be input the program RAM to access a specific line of instructions. Normally, PC would be increased after every instruction executed to point to the next address except if flow control instructions is executed which modify the bits contain in the PC.

B. Program/Instructions Memory

Program memory contains the list of instruction to be executed. In Von Neumann architecture machine, program memory and data memory use the common RAM, while in Harvard architecture machine, program memory and data memory use separate RAMs.

C. Instruction Register

Instruction register contains the current instruction. It stores the current register temporarily and connects to various other logic devices such as control logic, and register files. When the next instruction is executed, it will overwrite the content of this instruction register.

D. Register File (General Purpose Registers)

In RISC machine, register files are the important characteristic. It serves as the general purpose register to store temporary data that is executed by specific instruction. Register files are pretty similar to the RAM except that it doesn't have as much capacity as RAM and thus reduce the cost. Typically, registers are faster than RAM that makes execution of register-register instruction could be faster.

E. Arithmetic & Logic Unit

ALU is the unit that does the manipulation to the data such as addition, subtraction, logical AND, logical OR and many more.

F. Status Register

Update the flags when a specific instruction requires the flags to be updated. The flags can is the input to the control unit and will take effect on the sequence of the program execution. The flags can be zero, negative, carry and over flow.

G. Data Memory

Data memory is the storage device that store data from the program executed. It could be the constants, variables, address etc. Normally, data that are stores here are not a frequently used data as accessing the memory is slow thus make the program execution slower.

H. Control Unit

Control logic is among most important modules that make up a processor. It controls the sequence and datapath flow of an instruction. When an instruction is executed, it fetch and decode the opcode of that instruction and generate the control logic signals to the appropriate modules such as register files, ALU and memory.

I. Address Bus & Data Bus

Bus is used to simplify the movement of data from point to point in a computer. Bus is analogous to a highway and the devices are analogous to junctions that connect to this highway. By having both address bus and data bus, it is possible to reduce the number of wires that interconnect with in a computer but, it introduces a complexity. In a used system, only one communication from point to point could happen at a time. Thus a careful synchronization needs to be taken care.

V. BLOCK DIAGRAM OF RISC PROCESSOR

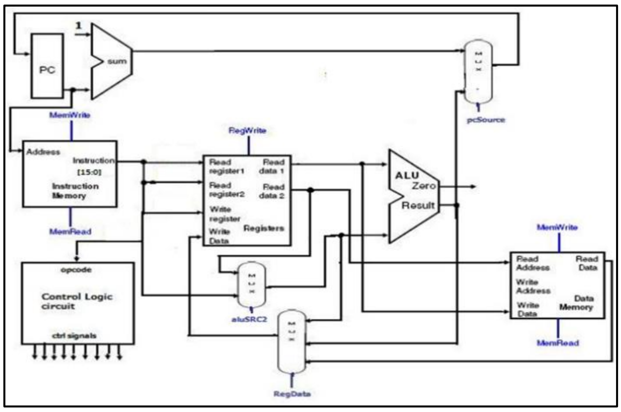


Fig. 2: Block Diagram of RISC Processor

A. RISC Processor Schematic

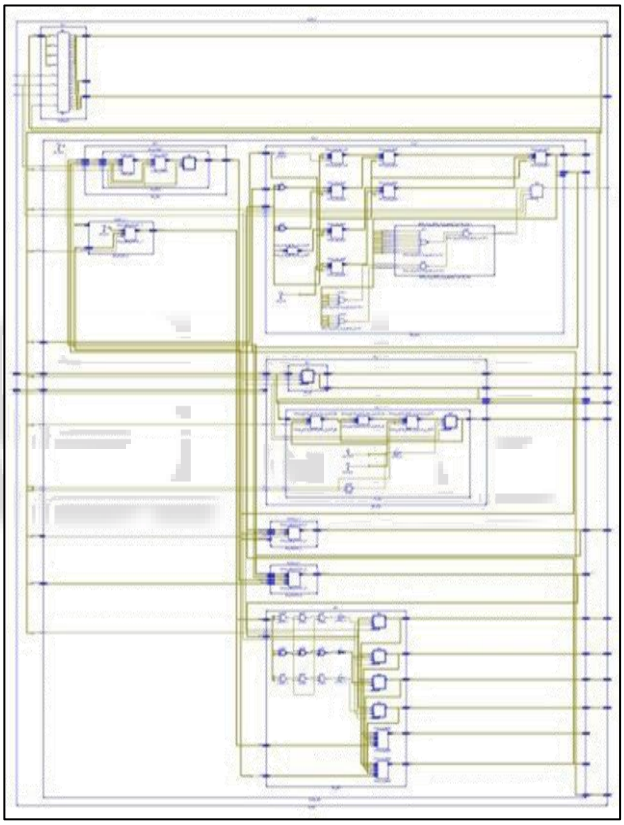
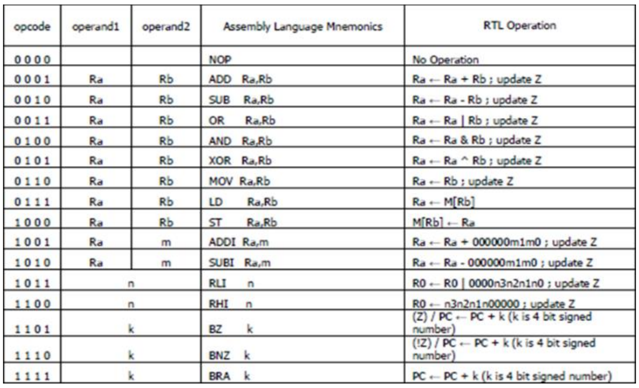


Fig. 3: RISC Processor Schematic

B. Instruction Set & Format



opcode	operand1	operand2	Assembly Language Mnemonics	RTL Operation
0000			NOP	No Operation
0001	Ra	Rb	ADD Ra,Rb	Ra ← Ra + Rb ; update Z
0010	Ra	Rb	SUB Ra,Rb	Ra ← Ra - Rb ; update Z
0011	Ra	Rb	OR Ra,Rb	Ra ← Ra | Rb ; update Z
0100	Ra	Rb	AND Ra,Rb	Ra ← Ra & Rb ; update Z
0101	Ra	Rb	XOR Ra,Rb	Ra ← Ra ^ Rb ; update Z
0110	Ra	Rb	MOV Ra,Rb	Ra ← Rb ; update Z
0111	Ra	Rb	LD Ra,Rb	Ra ← M[Rb]
1000	Ra	Rb	ST Ra,Rb	M[Rb] ← Ra
1001	Ra	m	ADDI Ra,m	Ra ← Ra + 000000m1m0 ; update Z
1010	Ra	m	SUBI Ra,m	Ra ← Ra - 000000m1m0 ; update Z
1011	n		RLI n	R0 ← R0 | 0000n3n2n1n0 ; update Z
1100	n		RHI n	R0 ← n3n2n1n00000 ; update Z
1101	k		BZ k	(Z) / PC ← PC + k (k is 4 bit signed number)
1110	k		BNZ k	(!Z) / PC ← PC + k (k is 4 bit signed number)
1111	k		BRA k	PC ← PC + k (k is 4 bit signed number)

Fig. 4: Instruction Set & Format

VI. Conclusion

A RISC processor has been designed, synthesized, validated and working correctly as expected by running the test pattern. The final result is validated by observing the final data stored data in data memory as well as in the register files. For further study, the performance of this RISC processor can be improved by implementing the 5-state pipeline mechanism as shown on figure below which is based on MIPS architecture.

References

[1] Morris Mano, „Digital Design of Verilog and applications‟, 3rd edition, Prentice Hall, 2008.

[2] Bhaskar, „verilog HDL synthesis‟, 2nd edition, Star Galaxy Pub, 2001.

[3] Dr Mohamed Khalil Hani, Starter‟s Guide to Digital Systems VHDL & Verilog Design, Revised Edition 2, July 2010, Desktop Publisher.

[4] Stephen Brown & Zvonko Vranesic, Fundamentals of Digital Logic with Verilog Design, Second Edition, 2003, McGra