FPGA Implementation and Functional Verification of a Pipelined MIPS Processor

[1] Balaji valli, [2] A. Uday Kumar, [3] B.Vijay Bhaskar
[1] PG Scholor, [2] Asst. Professors, [3]Assoc. Professor,
[1, 2, 3]Avanthi's St. Theressa Institute of Engineering & Technology, Garividi ,A.P.

Abstract:
This project targets the implementation design of a pipelined MIPS RISC Processor using VHDL (Very high speed integrated circuit Hardware Description Language). In this paper MIPS instruction format, instruction data path, decoder modules are analyzed. Furthermore, instruction fetch (IF) module of a CPU is designed based on RISC CPU instruction set. Function of IF module mainly includes fetch instruction and latch module address arithmetic module check validity of instruction module synchronous control module.

Keywords: MIPS, RISC, CPU, VHDL, FPGA, ID, IF, EX, MEM

1. Introduction

Pipeline is one of the basic techniques to improve the CPU's performance. This paper based on MIPS instruction set, designed a five-stage pipeline CPU. MIPS processor is basically a RISC micro processor. RISC CPU has extensive use in embedded system. Developing CPU with RISC structure is necessary choice. The MIPS Architecture defines thirty-two, 32-bit general purpose registers (GPRs). Instruction Set Architecture (ISA) of processor is composed of instruction set and corresponding registers. Program based on same ISA can run on the same instruction set. MIPS instruction has been developed from 32-bit MIPSI to 64-bit MIPSIII and MIPSIV since it was created. To assure downward compatibility, every generation production of MIPS instruction directly extends new instruction based on old instruction but not abnegates any old instruction, so MIPS processor of 64-bit instruction set can execute 32-bit instruction.

2. Implementation of Pipelined MIPS Processor

There are three formats of MIPS instructions:

(i) Register Format (R-type)

OPCODE (31 to 26)	RS (25to21)	RT (20to16)	RD (15to11)	Shift (10to6)	FUN (5to 0)

The first two 5-bit register specifications are the two read registers and the last 5-bit register specification is the destination register, that the instruction will use. The last 6-bits in the instruction are used as function bits.

(ii) Immediate Format (I-type)

OPCODE (31 to 26)	RS (25to21)	RT (20to16)	RD (15to11)	Shift (10to6)	FUN (5to 0)

The I-type is similar to the R-type except the second read register and the 5 function bits are replaced by a 16-bit immediate value. Each I-type opcode can have only one instruction because it has no function bits like the R-type.

(iii) Jump Type Format (J- type)

The J-type format consists of a 6-bit opcode and remaining bit indicates branching address.

OPCODE (31 to 26)	RS (25to21)	RT (20to16)	RD (15to11)	Shift (10to6)	FUN (5to 0)

3. Architecture

The MIPS pipelined processor involves five steps, the division of an instruction into five stages implies a five-stage pipeline:

1. Instruction Fetch (IF): fetching the instruction from the memory
2. Instruction Decode (ID): reading the registers and decoding the instruction

3. Execution (EX): executing an operation or calculating an address
4. Data Memory (MEM): accessing the data memory
5. Write Back (WB): writing the result into a register.

The key to pipelining the single-cycle implementation of the MIPS processor is the introduction of pipeline registers that are used to separate the data path into the five sections IF, ID, EX, MEM and WB. Pipeline registers are used to store the values used by an instruction as it proceeds through the subsequent stages. The MIPS pipelined registers are labeled according to the stages they separate. (e.g. IF/ID, ID/EX, EX/MEM, MEM/WB)

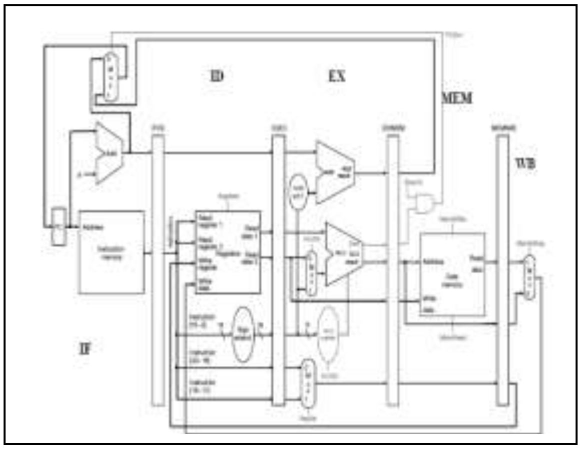

Fig.1. Pipelined MIPS Processor

To implement the MIPS pipelined processor, pipeline registers are placed into the corresponding VHDL modules that generate the input to the particular pipeline register. For example, the Instruction Fetch component will generate the 32-bit instruction and the PC+4 value and store them into the IF/ID pipeline register. When that instruction moves to the Instruction Decode stages it extracts those saved values from the IF/ID pipeline register.

4. RESULTS

Implementation is done using XILINX 10.1. RTL schematic and Floor plan view are shown in Fig.3 and Fig.4. Simulation snap is shown in Fig.2.

Fig.2. Functional simulation

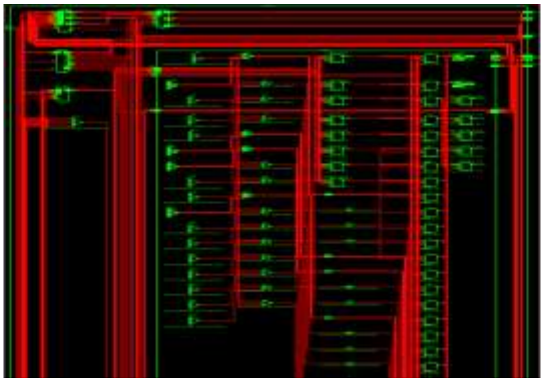

Fig.3. RTL Schematic

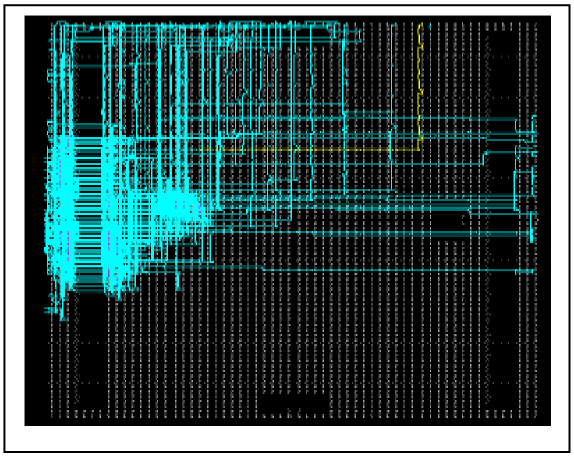

Fig.4. Floor Plan View

Hardware utilization summery is reported in Table. 1.

S.No.	Logic Utilization	Utilized Hardware
1.	Number of Slices	354
2.	Number of Slice Flip Flops	256
3.	Number of 4input LUTs	458
4.	Number of bonded IOBs	85

Table 1. Device Utilization Summery

5. Conclusion

In this paper, top-down design method adopted and VHDL is used for the implementation. Design is implemented on XILINX 11.1. Data Memory and Instruction Memory are generated using XILINX COREGEN.

REFERENCES

[1] Hui Ma, Dinglei Wang, The Design of Five-stage Pipeline CPU Based on MIPS, 978-1-4244-8165-1/11, 2011 IEEE.

[2] 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,

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

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

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

[6] Pan-Song, Huang-JiYe, SOPC Technology Utility Tutorial, Tsinghua University Press, 2006.

[7] MIPS32 4KTMProcessor Core Family Software User's Manual, MIPS Technologies Inc. [M]. BeiJing, Engine Industry Press. 2003.

[8] Yi-Kui, Ding-YueHua, Application of AMCCS5933 Controller in PCI BUS, DCABES2007, 2007.7.759

[9] "Rapid Prototyping of digital Systems, a tutorial approach "By – James O. Hamblen and Michael D.Furman.

[10] Charles E. Gimarc, Veljko M. Mhtinovic, "RISC Principles, Architecture, and Design", Computer Science Press Inc., 1989.

[11] Whte paper, "Wide range of comprehensive tools speed, Development of high-pafomance embedded system", MIPS Technologies Inc.,hnp:/lwww.mips.com/whitepapers/030399Wl .hml (current Sep. 30,2002).

[12] S. Chen, B. Mulgrew, and P. M. Gran< "A clustering technique for digital communications channel equalization using radial basis function networks," *EEE Trans. Neuml Nerworkr,* vol. 4, pp. 570-578, July 1993.