



**FINAL YEAR PROJECT REPORT**

**DESIGN AND SIMULATION OF MIPS PROCESSOR IN  
HDL VERILOG**

**By**

**Asmatullah Khan Babar  
Muhammad Faizan Choudhry  
Muhammad Abdullah Shaikh**

**Bachelor of Computer Engineering**

**DEPARTMENT OF COMPUTER SCIENCES & ENGINEERING  
BAHRIA UNIVERSITY KARACHI**

**2007**



# ACKNOWLEDGEMENT

First of all we would like to thank Almighty ALLAH for giving us immense strength, good knowledge, great power for doing things in our life and to accomplish our project on time. Second of all we would like to thank our parents who prayed for our success in every part of life.

The journey for creating this project has been exciting one and it wouldn't be possible without the help of friends and many other professionals. We would like to thank all those people who have contributed in our project. But particularly we would like to thank the following faculty members for their special contributions:

- Eng. Salman Zafar
- Eng. Aley Imran Rizvi

In addition to these faculty members of our university, there are two other personal whose names needs a definite mentioning.

- Eng. Shahab Tahzeeb (NED University)
- Eng. Umair Siddiqui (BIMCS)

We would also like to thanks Director, Head of Computer Science Department and Project Coordinator for providing us the best electronics and computer labs for our project.



# ABSTRACT

This project deals with the design and simulation of a MIPS processor, which is 32-bit version of the MIPS R2000 processor. The data path of this processor consists of 32, 32-bit registers. In this project we have implemented some of the basic instructions of the MIPS processor such as ADD, AND, SUB, SLT, OR, JUMP, BEQ, SW and LW. These simple instructions are based on only three formats which are R, I and J. The design has a 32-bit ALU which calculates series of operations according to the instruction format. In order to accomplish this task we used Verilog HDL for designing the modules and the software we used for designing is Xilinx and the other software is ModelSIM that is used for generating the waveforms, that helps us in verifying the right functionality being performed by each and every signal and component we have designed.

4.1	Clocking elements	25
4.2	Single cycle functionality	28
4.3	Multi-cycle design without control signals	28
4.4	Multi-cycle design with control signals	30
4.5	Fetch stage control signals	31
4.6	Decode Control Signals Settings	31
4.7	Execute Control Signals Settings	32
4.8	Complete Finite State Machine	33
4.9	Activation of units in instruction Fetch cycle of R-type instruction	37



# TABLE OF CONTENTS

<b>1</b>	<b>INTRODUCTION</b> .....	<b>2</b>
<b>2</b>	<b>BACKGROUND AND LITERATURE REVIEW</b> .....	<b>5</b>
2.1	Processor .....	5
2.2	RISC and CISC machines .....	5
2.3	MIPS (RISC) Design Principles.....	7
2.4	List of Components.....	8
2.4.1	32-Bit ALU.....	8
2.4.2	ALU Control.....	8
2.4.3	Memory .....	9
2.4.4	Register File.....	10
2.4.5	Temporary Registers .....	11
2.4.6	Sign Extension .....	11
2.4.7	Multiplexers.....	12
2.5	Verilog .....	12
2.5.1	Levels of Abstraction .....	12
2.5.2	Synthesizing Verilog.....	14
2.6	MIPS Register Convention .....	15
<b>3</b>	<b>Aim and Statement of Problem</b> .....	<b>17</b>
3.1	Project Plans .....	17
3.2	Tasks .....	18
3.3	Requirements.....	19
<b>4</b>	<b>ANALYSIS AND DESIGN</b> .....	<b>21</b>
4.1	Architecture.....	21
4.2	Instruction Execution Cycles .....	22
4.2.1	Instruction Fetch .....	23
4.2.2	Instructions Decode and Register Fetch.....	23
4.2.3	Execution, Memory Address Computation, or Branch Completion.....	24
4.2.4	Memory Access or R-type Instruction Completion.....	24
4.2.5	Memory Read Completion .....	24
4.3	Clocking Methodologies.....	25
4.4	Datapath .....	26
4.4.1	The Multicycle Datapath without control signals.....	27
4.4.2	The Multicycle Datapath with Control Signals .....	29
4.5	Instruction Fetch Control Signals.....	31
4.6	Instruction Decode Control Signals.....	31
4.7	Instruction Execute Control Signals .....	32
4.8	Complete Finite State Machine .....	32
4.9	Control Signals .....	34
4.10	Register Transfer Language .....	35
4.11	Execution of R type Instruction.....	37



4.11.1	Fetch Cycle .....	37
4.11.2	Decode Cycle .....	38
4.11.3	Execute Cycle .....	39
4.11.4	Write Back Cycle .....	40
4.12	Instruction Formats .....	41
4.12.1	R-type Format .....	41
4.12.2	I-type Format .....	41
4.12.3	J-type Format .....	42
4.12.4	Memory reference .....	42
4.12.5	R-type instruction .....	43
4.12.6	Branch instruction .....	43
4.3.4	J-type instruction .....	44
4.13	MIPS Addressing Modes .....	44
4.13.1	Register Addressing .....	45
4.13.2	PC-Relative Addressing .....	45
4.13.3	Pseudo-Direct Addressing .....	46
4.13.4	Base Addressing .....	46
4.13.5	Indirect Addressing .....	46
4.14	MIPS Processor Instruction Set Architecture .....	47
4.14.1	Arithmetic instructions .....	47
4.14.2	Logical Instructions .....	49
4.14.3	Data Transfer Instructions .....	51
4.14.4	Conditional Branch Instructions .....	51
4.14.5	Unconditional Jump Instructions .....	53
<b>5</b>	<b>Implementation .....</b>	<b>55</b>
5.1	Arithmetic Logic Unit .....	56
5.2	Alu Control unit .....	56
5.3	Register File .....	56
5.4	Control Unit .....	57
5.5	2 to 1 multiplexer .....	57
5.6	3 to 1 multiplexer .....	57
5.7	4 to 1 multiplexer .....	57
5.8	D Flip Flop .....	58
5.9	memory .....	58
5.10	Shift Left .....	58
5.11	Sign Extend .....	59
5.12	oscillator .....	59
<b>6</b>	<b>Testing .....</b>	<b>61</b>
6.1	Arithmetic Logic Unit Simulation .....	62
6.2	Alu Control Unit Simulation .....	63
6.3	2 to 1 multiplexer .....	64
6.4	3 to 1 multiplexer .....	65
6.5	4 to 1 multiplexer .....	66
6.6	D Flip Flop .....	67



6.7	Add Operation .....	68
6.8	Load word Operation .....	69
6.9	Beq operation .....	70
<b>7</b>	<b>Result.....</b>	<b>72</b>
<b>8</b>	<b>Discussion.....</b>	<b>74</b>
<b>9</b>	<b>Conclusion.....</b>	<b>77</b>
<b>10</b>	<b>Future work.....</b>	<b>79</b>
	<b>REFERENCES.....</b>	<b>81</b>
	<b>APPENDIX A.....</b>	<b>84</b>
	<b>APPENDIX B.....</b>	<b>87</b>
	<b>APPENDIX C.....</b>	<b>89</b>

# INTRODUCTION