DESIGNING A 32-BIT PROCESSOR WITH VHDL

EXPLORING THE MIPS ARCHITECTURE

AGHADJOUR ZAKARIA
Embedded Electronic Systems & Systems Control Engineering Student
aghadjour@gmail.com

1. Introduction

VHDL (Very High Speed Integrated Circuit Hardware Description Language) is a hardware description language used to model the behavior of electronic circuits. It allows for the representation of systems whose physical design can then be implemented. One of VHDL's key advantages is its technology and vendor independence, making it both portable and reusable [1].
The **single-cycle MIPS processor** (Microprocessor without Interlocked Pipeline Stages) is a **RISC** (Reduced Instruction Set Computer) implementation capable of executing each instruction in a single clock cycle, with the cycle duration determined by the slowest instruction. Many previous studies have modeled simple single-cycle RISC processors in VHDL. Some of these have focused on the design of MIPS processors that can execute basic instructions like addition, subtraction, load/store, and branching.

2. Instruction Set Architecture

To control a computer's hardware, it is essential to communicate using its native language. The basic elements of this language are called instructions, and the collection of all possible instructions forms the instruction set. For the simplicity of designing a RISC (Reduced Instruction Set Computer) processor, it is necessary to ensure that all instructions are of the same length and follow a single instruction format. In MIPS (Microprocessor without Interlocked Pipeline Stages), instructions are 32 bits long and are classified into three primary formats :

R-type instructions: These are "register-type" instructions. They require three registers as operands: two source registers and one destination register.
I-type instructions: These are "immediate-type" instructions. They use two registers and a 16-bit immediate operand (with a modification in this implementation where the immediate operand is 14 bits).
J-type instructions: These are "jump-type" instructions, used exclusively for jump operations. They employ a single 26-bit address operand (with a modification in this implementation where the address operand is 29 bits).

The following table presents the 26 instructions that can be executed by my processor :

Type R :

OPCODE (8)	Rs(5)	Rt(5)	Rd(5)	Shamt (9)

Type I :

OPCODE (8)	Rs(5)	Rd(5)	Imm (14)

Type J :

OP	Destination (29)

INSTRUCTION	TYPE	OPCODE	RS
ADD rd,rs,rt	R	00000001	
SUB rd,rs,rt	R	00000010	
MUL rd,rs,rt	R	00000011	
DIV rd,rs,rt	R	00000100	
AND rd,rs,rt	R	00000101	
OR rd,rs,rt	R	00000110	
XOR rd,rs,rt	R	00000111	
SLL rd,rs,shamt	R	00001000	
SRL rd,rs,shamt	R	00001001	
SLA rd,rs,shamt	R	00001010	
SRA rd,rs,shamt	R	00001011	
JR regd	I	01001101	11110
ADDI rd,rs,Imm	I	01000001	
SUBI rd,rs,Imm	I	01000010	
MULI rd,rs,Imm	I	01000011	
DIVI rd,rs,Imm	I	01000100	
ANDI rd,rs,Imm	I	01000101	
ORI rd,rs,Imm	I	01000110	
XORI rd,rs,Imm	I	01000111	
NOT rd,rs	I	01001000	
LW rd,rs,Imm	I	01001001	
SW rd,rs,Imm	I	01001010	
BEQ rd,rs,Imm	I	01001011	
BNE rd,rs,Imm	I	01001100	
J destination	J	100	
JAL destination	J	101	

3. Processor Design

The complete design of a 32-bit, single-cycle processor is composed of two key components:

1.32-bit Datapath

2.Control Unit

Architecture and Components of the 32-bit Processor

The 32-bit processor follows a structured design inspired by MIPS architecture. It is composed of several key units that work together to fetch, decode, execute instructions, and manage memory operations efficiently.

1**. Program Counter (PC)**

- The Program Counter (PC) is a dedicated register that holds the address of the next instruction to be fetched from memory.
- It is updated at each clock cycle based on the control logic and execution flow.
- Supports conditional and unconditional branching by updating its value based on computed addresses.

---

2. Program Memory (Instruction Memory)

- Stores the set of instructions for the processor.
- Outputs the instruction at the address specified by the PC.

---

3. Register File

- Contains 32 general-purpose registers, each 32 bits wide.
- Two registers are read simultaneously for instructions requiring operands.
- One register can be written at the end of an instruction cycle.

  Register $31 (Constant Zero Register)

  This register is hardwired to **0** and cannot be modified.

  Register $30 (Reserved for Function PC Storage)

  When a function is called, the previous program counter (PC) value is stored here to allow returning to the correct instruction after execution.

  Register $29 ($v0 - Function Return Value)

This register is designated to **hold the return value of a function**.

---

4. Arithmetic Logic Unit (ALU)

- Performs arithmetic and logical operations based on the instruction opcode.
- Handles comparison operations to support conditional branching.

---

5. Data Memory (Main Memory)

- Serves as the primary memory for load/store operations.

---

6. Control Unit

- Directs the operation of the processor by generating control signals based on the instruction opcode.
- Controls the selection of data paths, read/write operations, and ALU execution.
- Manages state transitions through the different execution phases (fetch, decode, execute, memory access, and write-back).

4 . Conclusion

This processor's architecture is flexible and can be adapted for pipelined execution with minor modifications. By adding register buffering and hazard detection, it can efficiently process multiple instructions simultaneously, improving performance while maintaining a simple and scalable design.