Universidad de Guadalajara
Centro Universitario de Ciencias Exactas e Ingenierías

Ingeniería en Computación

Proyecto Final

Reporte de Trabajo Colaborativo presentado por

Humberto de Jesus Peña Dueñas
Karla Rebeca Hernández Elizarrarás
Elizabeth Arroyo Moreno

Alumnos de Cuarto Semestre ICOM

Arquitectura de Computadoras

Mtro. Jorge Ernesto Lopez Arce Delgado

Guadalajara, Jal. Mayo de 2025

Introducción

[FASE 1]

La arquitectura MIPS (Microprocessor without Interlocked Pipeline Stages), representa una de las implementaciones más influyentes adoptadas dentro del paradigma RISC (Reduced Instruction Set Computer). Su diseño ha sido reconocido su simplicidad y facilidad estructural, su eficiencia operativa y su enfoque modular. Gracias a su organización y a la previsibilidad en el tiempo de ejecución de sus instrucciones,el MIPS es un modelo ideal para la enseñanza de conceptos clave como el control de flujo, la segmentación por etapas y el procesamiento paralelo.

El propósito principal de este proyecto final es desarrollar e implementar un datapath de 32 bits basado en la arquitectura MIPS, capaz de interpretar y ejecutar un subconjunto específico de 28 instrucciones pertenecientes a los tres formatos clásicos definidos por esta arquitectura: tipo R el que es registro, tipo I es el inmediato y tipo J que es de salto. Estas instrucciones han sido seleccionadas por su utilidad representativa en programas típicos de nivel básico-intermedio, dejando la validación desde operaciones aritméticas hasta mecanismos de control de flujo y acceso a memoria.

En particular, las instrucciones de tipo R en MIPS son fundamentales para realizar operaciones aritméticas y lógicas que involucran exclusivamente registros. Este formato de instrucción utiliza campos específicos para identificar los registros fuente y destino, así como la operación a ejecutar, sin incluir valores inmediatos. Las instrucciones tipo R, como add, sub, and y or, permiten manipular datos almacenados en los registros de manera directa, lo que contribuye a la eficiencia y rapidez en la ejecución. Además, estas instrucciones mantienen la filosofía RISC al ser simples y uniformes, facilitando su decodificación y ejecución en un ciclo de reloj. La implementación de estas instrucciones en el datapath requiere un manejo preciso de los registros y la unidad aritmético-lógica (ALU), asegurando que las operaciones se realicen correctamente y que los resultados se almacenen en los registros adecuados.

En conjunto, la integración de los formatos de instrucción tipo R, I y J en el datapath permitirá cubrir un amplio espectro de funcionalidades necesarias para la ejecución de programas representativos, consolidando así una base sólida para el estudio y aplicación de la arquitectura MIPS.

Tablas de Instrucciones

Instrucción	Tipo	Sintaxis
Add	R	Add $rd, $rs, $rt
Sub	R	Sub $rd, $rs, $rt
Mul	R	Mul $rd, $rs, $rt
Div	R	Div $rd, $rs, $rt
Or	R	Or $rd, $rs, $rt
And	R	And $rd, $rs, $rt
Addi	I	Addi $rt, $rs, immediate
Subi	I	Subi $rt, $rs, immediate
Ori	I	Ori $rt, $rs, immediate
Andi	I	Andi $rt, $rs, immediate

Instrucción	Tipo	Sintaxis
Lw	I	Lw $rt, $rs, immediate
Sw	I	Sw $rt, offset($rs)
slt	R	slt $rd, $rs, $rt
Slti	I	Slti $rt, $rs, $immediate
beq	I	beq $rs, $rt, offset
bne	I	bne $rs, $rt, offset
J	J	J target
nop	R	nop
bgtz	I	bgtz $rs, offset

Objetivo

[FASE 1]

Diseñar e implementar un datapath MIPS de 32 bits, basado en la arquitectura RISC, que sea capaz de ejecutar un conjunto específico de 28 instrucciones de los tipos R, I y J. El propósito es simular correctamente la ejecución de un programa escrito en lenguaje ensamblador y su correspondiente traducción a código máquina, verificando tanto el flujo de datos como las señales de control en cada una de las etapas del procesamiento

Desarrollo

[FASE 1]

El datapath está compuesto por varios módulos clave:

- **Contador de Programa (PC):** Guarda la dirección de la próxima instrucción a ejecutar y se incrementa en 4 cada ciclo, salvo si hay un salto (branch o jump).
- **Memoria de Instrucciones:** Es una memoria de solo lectura donde se almacena el programa en código máquina, precargado manualmente.
- **Banco de Registros:** Permite leer simultáneamente dos registros y escribir en uno, usando registros específicos para resultados, punteros y datos temporales según el algoritmo.
- **ALU (Unidad Aritmético-Lógica):** Realiza operaciones aritméticas y lógicas (add, sub, and, or, slt, etc.) controlada por un módulo ALU Control que define la operación según la instrucción.
- **Memoria de Datos:** Almacena datos necesarios para instrucciones tipo carga (Lw) y almacenamiento (Sw), con valores iniciales precargados.
- **Unidad de Control:** Genera señales de control basadas en el opcode para manejar lectura/escritura de registros, operaciones de ALU y acceso a memoria.
- **Buffers (IF/ID, ID/EX, EX/MEM, MEM/WB):** Mantienen la sincronización en el pipeline.

Cada módulo fue desarrollado y probado individualmente en Verilog, integrando así todo el datapath para ejecutar instrucciones MIPS.

Conclusión

[FASE 1]

El desarrollo de este proyecto permitió adquirir una comprensión más profunda del funcionamiento interno de un procesador MIPS desde la perspectiva del hardware. Se evidenció cómo las instrucciones son ejecutadas de forma secuencial y eficiente mediante un datapath segmentado, resaltando la relevancia de cada módulo y etapa del pipeline. La implementación en Verilog, junto con la traducción manual de instrucciones desde lenguaje ensamblador a código máquina, reforzó de manera significativa los conocimientos teóricos y prácticos sobre arquitectura de computadores.

Las simulaciones realizadas permitieron visualizar en detalle el flujo de información entre módulos y la sincronización de señales, validando el comportamiento esperado del sistema completo. Este proyecto demostró ser un desafío técnico enriquecedor, combinando fundamentos de lógica digital, estructuras computacionales y programación de bajo nivel, y brindó una experiencia formativa valiosa para el desarrollo profesional en áreas como diseño de hardware y sistemas embebidos.

Bibliografías

Abd-El-Barr, M., & El-Rewini, H. (2004). *Fundamentals of computer organization and architecture* (Vol. 38). John Wiley & Sons.

Stalling, W. (2005). Organización y Arquitectura de Computadores (7a ed.) Pearson-Prentice-Hall.

Patterson, D. A. (2007). Computer organization and design: The hardware/software interface. Morgan Kaufmann.

Hwang K. (s.f.). Arquitectura de Computadores y Procesamiento Paralelo. Mc Graw-Hill.