Proyecto Final Fase 1.2 – Single Cycle Data Path

**Integrantes**

Alan Sebastián Guzmán Márquez

Gabriel Isaí Alcaraz Suarez

Kevin Manuel Sanchez Agredano

Quintero Gonzalez Diego Gerardo

**Introducción**

En esta fase del proyecto después de haber realizado un DataPath de las instrucciones tipo R, ahora realizaremos un “single cycle DathaPath” que es una implementación de procesador en la que cada instrucción se ejecuta en un solo ciclo de reloj, es decir debemos lograr que toda una instrucción pase por todos los módulos en un solo ciclo de reloj.

Además en esta etapa del proyecto se tocaron temas como endianess, Tipos de Arquitectura Von Newman, Harvard, las instrucciones de tipo R en MIPS, la operación ternaria y el proceso de compilación.

**Objetivo**

• Objetivo General

El objetivo principal es agregar e instanciar los otros módulos para completar la Figura 1 y 2 y hacer que la parte de las instrucciones de tipo R y operaciones (DataPath) corra en un solo ciclo.

Con nuevos módulos como la memoria de instrucciones, PC y otros multiplexores, estos módulos pueden observarse en la Figura 1 y 2.

• Objetivos Particulares

* Agregar al decodificador la posibilidad de abrir los archivos en ensamblador y modificarlos en una ventana de texto, guardar los cambios y convertir a binario.
* Agregar la opción al decodificador para generar un archivo para inicializar el banco de registros y memoria de datos. Este archivo debe incluir direcciones del banco de registros y datos de 32 bits para operar.
* Tener una guía de usuario de la fase 1.2 del decodificador

**Desarrollo**

**Ljubisa Bajic**

Ljubisa Bajic es un investigador y profesional en el campo de la arquitectura de computadoras.

En cuanto a las contribuciones específicas de Ljubisa Bajic, aquí hay algunas patentes y trabajos relacionados con su investigación:

* Dinámica de reducción de potencia controlada y circuito para un procesador gráfico: L. Bajic y J. Fry obtuvieron una patente relacionada con la reducción de potencia en procesadores gráficos.
* Cálculo dinámico de peso en un sistema de estimación y gestión de energía digital: L. Bajic también tiene una patente relacionada con el cálculo de peso en sistemas de estimación y gestión de energía.
* Gestión de energía entre múltiples procesadores que comparten una plataforma térmica: L. Bajic y otros colaboradores obtuvieron una patente relacionada con la gestión de energía en sistemas con múltiples procesadores.

**Jim Keller**

Jim Keller, un destacado diseñador de arquitecturas de CPU, ha dejado su huella en la industria tecnológica a lo largo de los años.

*Arquitectura Zen de AMD:*

Jim Keller fue uno de los principales diseñadores detrás de la microarquitectura Zen de AMD.

La arquitectura Zen revolucionó el mercado de procesadores al ofrecer un rendimiento competitivo frente a Intel.

Los procesadores AMD Ryzen se basan en la arquitectura Zen y han sido muy exitosos.

*Microarquitectura K8:*

Keller también estuvo involucrado en la creación de la microarquitectura K8, que dio vida a los procesadores Athlon 64 de AMD.

Los Athlon 64 fueron los primeros procesadores fabricados en 64 bits y tuvieron un gran impacto en la industria.

*Microarquitectura K7:*

Antes de su trabajo en AMD, Keller contribuyó a la microarquitectura K7, que fue la primera en superar la frecuencia de 1 GHz en procesadores.

La K7 también fue la primera arquitectura que preocupó a Intel.

*Ingeniería de Silicio en Intel:*

En 2018, Jim Keller se unió a Intel como vicepresidente senior de Intel Technology, Systems Architecture & Client Group (TSCG) y gerente general de ingeniería de silicio.

Reveló que Intel tenía una estrategia para actualizar la arquitectura y la tecnología de la CPU cada dos años, además de desarrollar una nueva arquitectura CPU cada 5 años.

**Raja Koduri**

Raja Koduri es un ingeniero informático indio y un ejecutivo especializado en hardware de gráficos por computadora. A lo largo de su carrera, ha desempeñado roles clave en empresas líderes de la industria. Aquí tienes algunos aspectos destacados de su trayectoria:

*Carrera temprana:*

Nacido el 31 de agosto de 1968 en Kovvur, distrito de West Godavari, Andhra Pradesh, India, Raja Koduri proviene de una familia telugu.

Su primo es el conocido director de cine S. S. Rajamouli, y otros miembros de su familia también trabajan en la industria cinematográfica como escritores, compositores de música y cantantes.

Obtuvo su licenciatura en electrónica y comunicaciones de la Universidad de Andhra y una Maestría en Tecnología de la IIT Kharagpur1.

*Contribuciones en la industria de gráficos:*

En 1996, Koduri se unió a S3 Graphics y luego se convirtió en director de desarrollo de tecnología avanzada en ATI Technologies.

Después de la adquisición de ATI por parte de Advanced Micro Devices (AMD) en 2006, se desempeñó como Director de Tecnología para gráficos en AMD hasta 2009.

Durante su tiempo en S3 y ATI, contribuyó significativamente al diseño de varias generaciones de arquitecturas de GPU, desde DirectX Ver 3 hasta Ver 11.

Luego, trabajó en Apple Inc., donde se centró en hardware de gráficos, lo que permitió a Apple adoptar pantallas Retina de alta resolución en sus computadoras Mac1.

*AMD y Radeon Technologies Group (RTG):*

En 2013, Raja Koduri regresó a AMD como Vicepresidente de Visual Computing, supervisando tanto el hardware como el software de GPU.

En 2015, AMD reorganizó su división de gráficos y ascendió a Koduri al nivel ejecutivo, nombrándolo Vicepresidente Senior y Arquitecto Jefe del recién formado Radeon Technologies Group.

Bajo su liderazgo, se transformó la arquitectura de las GPU con las generaciones Polaris, Vega y Navi, que se utilizaron en PC, Mac y consolas de juegos (como Xbox y PlayStation).

En 2017, Koduri tomó un año sabático para pasar tiempo con su familia, y durante su ausencia, la CEO de AMD, Lisa Su, asumió el liderazgo de RTG.

Sin embargo, en noviembre de 2017, anunció su renuncia a AMD1.

*Intel y arquitectura Xe:*

Posteriormente, Raja Koduri se unió a Intel como Arquitecto Jefe y Vicepresidente Ejecutivo de la división de arquitectura, gráficos y software (IAGS).

Durante su tiempo en Intel, lideró el desarrollo de la arquitectura Xe, que se destaca por su rendimiento, escalabilidad, eficiencia y compatibilidad con tecnologías de última generación.

En marzo de 2023, después de seis años en el sector de las GPU, Koduri dejó Intel para fundar una start-up de software de juegos generativos basada en inteligencia artificial.

**Endianess**

La endianness (o byte order) se refiere al orden en el que se almacenan los bytes en la memoria de una computadora. Es un concepto fundamental para comprender cómo los datos se representan internamente.

*Big-Endian (BE):*

En big-endian, el byte más significativo (o “big end”) de un valor de datos multibyte se almacena en la dirección de memoria más baja.

Imagina que tienes un número de 32 bits (4 bytes) como 0x12345678. En big-endian, se almacenaría como:

0x12 0x34 0x56 0x78

El byte 0x12 (el más significativo) se encuentra en la dirección de memoria más baja.

*Little-Endian (LE):*

En little-endian, el byte menos significativo (o “little end”) de un valor de datos multibyte se almacena en la dirección de memoria más baja.

Siguiendo el mismo ejemplo, en little-endian, el mismo número se almacenaría como:

0x78 0x56 0x34 0x12

El byte 0x78 (el menos significativo) está en la dirección de memoria más baja.

Ejemplo visual:

Consideremos el número decimal 1,100 (representado en binario como 1100).

Si lo almacenamos en 3 bytes, en big-endian sería:

0b00000000 0b00000100 0b01011100

Y en little-endian:

0b01011100 0b00000100 0b00000000

En resumen, big-endian y little-endian son dos formas diferentes de organizar los bytes en la memoria.

**Tipos de Arquitectura Von Newman y Harvard.**

En el mundo de la informática, existen dos arquitecturas fundamentales que han influido en el diseño de sistemas informáticos: la arquitectura de Von Neumann y la arquitectura de Harvard. A continuación, te explicaré las diferencias entre ambas y cómo se aplican en las computadoras modernas:

*Arquitectura de Von Neumann:*

La arquitectura de Von Neumann, nombrada en honor a su creador, John von Neumann, es el modelo más comúnmente utilizado en los sistemas actuales.

Características:

Utiliza una única memoria para almacenar tanto instrucciones como datos.

La CPU accede a una ubicación de memoria específica para leer la siguiente instrucción y luego ejecutarla.

Programa y datos comparten la misma memoria y bus de datos.

Ventajas:

Es simple y fácil de implementar.

Permite la modificación del programa durante la ejecución.

Uso en computadoras modernas: La arquitectura de Von Neumann es la más común y se encuentra en la mayoría de los ordenadores personales.

*Arquitectura de Harvard:*

La arquitectura de Harvard separa físicamente la memoria para instrucciones y la memoria para datos.

Características:

Utiliza memorias separadas para instrucciones y datos.

Permite que la CPU acceda simultáneamente a instrucciones y datos, mejorando el rendimiento potencialmente.

Utiliza buses de datos separados para instrucciones y datos.

Ventajas:

Puede ser más eficiente en términos de rendimiento debido a la separación de memorias y buses de datos.

Uso en computadoras modernas: La arquitectura de Harvard se encuentra más en dispositivos embebidos y sistemas especializados.

**El MIPS**

El procesador MIPS (Microprocessor without Interlocked Pipeline Stages) es una familia de microprocesadores de arquitectura RISC (Reduced Instruction Set Computer) desarrollados por MIPS Technologies. A continuación algunos datos sobre los elementos y características generales del procesador MIPS de 32 bits:

1. *Arquitectura de 32 bits:*

Las primeras arquitecturas MIPS fueron de 32 bits, con rutas de datos y registros de 32 bits de ancho.

Posteriormente, se implementaron versiones en 64 bits.

1. *Conjunto de instrucciones:*

El conjunto de instrucciones MIPS se compone de operaciones básicas que se ejecutan en un solo ciclo de reloj.

Las instrucciones son de tamaño fijo y presentan diferentes formatos, como las instrucciones de carga y almacenamiento, operaciones aritméticas y lógicas, y bifurcaciones (instrucciones tipo R, I y J).

1. *Modos de funcionamiento:*

El procesador MIPS tiene varios modos de funcionamiento:

Usuario: para ejecutar programas de usuario.

Núcleo: para operaciones privilegiadas del sistema operativo.

Supervisor: para tareas de administración y control.

Depuración: para depurar programas.

1. *Instrucciones de carga y almacenamiento:*

Las instrucciones de carga y almacenamiento permiten leer y escribir en memoria utilizando registros.

Ejemplos:

lw $t0, dir: Carga en el registro $t0 el contenido de la palabra de memoria cuya dirección es dir.

sw $t0, dir: Almacena en la memoria direccionada por dir el contenido del registro $t0.

1. *Bifurcaciones (control de flujo):*

Las bifurcaciones permiten cambiar el flujo de ejecución del programa.

Ejemplos:

beq $t0, $t1, destino: Bifurca si los registros $t0 y $t1 son iguales.

bne $t0, $t1, destino: Bifurca si los registros $t0 y $t1 no son iguales.

1. *Convenciones de uso de registros:*

En MIPS, existen convenciones para el uso de registros, principalmente en llamadas a procedimientos.

La convención predeterminada sigue las pautas utilizadas por el compilador GCC.

**Instrucciones tipo R:**

Las instrucciones tipo R son un conjunto de instrucciones aritmético-lógicas utilizadas en arquitecturas de procesadores como MIPS.

Estas instrucciones operan exclusivamente en registros y siguen un formato específico.

El formato de una instrucción tipo R consta de los siguientes campos:

Función (6 bits): Indica la operación a realizar (por ejemplo, suma, resta, AND, OR, etc.).

SHAMT (5 bits): Representa el desplazamiento de datos en operaciones de rotación.

Registros (3 campos de 5 bits cada uno): Especifican las direcciones de los registros involucrados en la operación.

OpCode (6 bits): Siempre es “000000” para instrucciones tipo R.

Ejemplo: La instrucción “add” realiza una suma aritmética entre dos registros1.

**Operación Ternaria:**

La operación ternaria es una expresión que involucra tres operandos y un operador.

Ejemplo: condición ? valor\_si\_verdadero : valor\_si\_falso.

Si la condición es verdadera, se evalúa el primer valor; de lo contrario, se evalúa el segundo valor.

**Proceso de Compilación:**

El proceso de compilación transforma código fuente en lenguaje de máquina ejecutable.

Fases del compilador:

Análisis léxico: Divide el código en tokens.

Análisis sintáctico: Construye un árbol de sintaxis abstracta.

Análisis semántico: Verifica la coherencia semántica.

Generación de código intermedio: Crea una representación intermedia.

Optimización de código: Mejora el rendimiento.

Generación de código final: Produce el código ejecutable.

***Definición de las instrucciones para nuestro proyecto:***

Las instrucciones de nuestro programa serán en Big endian, pues escribir y leer de izquierda a derecha es lo más conveniente para como tenemos creadas las memorias y bancos de registro, y no tener que estar dando vuelta los bits.

Y en cuanto las instrucciones en ensamblador, pretendemos usar las abreviaciones correspondientes a la operación que se va a realizar, y el número del directorio del banco de registros, todo separado por comas y cada instrucción en una línea diferente.

**Conclusiones**

* Kevin Manuel Sanchez Agredano

La funcionalidad agregada al proyecto permite una gestión completa de archivos, facilitando su visualización, modificación y conversión a binario. La capacidad de generar un archivo binario para cargar la memoria de instrucciones con un ancho de palabra específico (8 bits) y opciones para inicializar el banco de registros y la memoria de datos añade un nivel de control y flexibilidad adicional al sistema. Estas características son fundamentales para el desarrollo y la depuración de programas en lenguaje ensamblador, facilitando la creación y prueba de software para la arquitectura objetivo. Con estas funcionalidades, el proyecto se vuelve más completo y útil para el desarrollo de programas en ensamblador.

* Diego Gerardo Quintero González

En esta etapa del proyecto se han logrado avances al desarrollar y probar los módulos de varios componentes nuevos, como un multiplexor de 5 bits, un modulo PC, un sumador de 32 bits, un desplazador a la izquierda de 2 bits, una extensión de signo y una unidad de control con nuevas salidas. La creación de testbenches para cada uno de estos módulos aseguró su correcto funcionamiento. Además, se logró la integración de muy buena forma de estos nuevos módulos con los existentes de etapas anteriores, estableciendo la realizacion completa del esquema establecido para esta fase del proyecto.

* Gabriel Isaí Alcaraz Suarez

En esta etapa que me tocó hacer la parte de la investigación y guía de usuario, fue fácil para mí comparado con hacer los modulos de verilog o lo que se implementó en el decodificador, y con endianess me dí cuenta que al igual que en el mundo hay países que leen y escriben de derecha a izquierda, también hay programas que se leen así por la computadora.

* Alan Sebastián Guzmán Márquez

En esta fase se logró un avance en todos los aspectos que abarca el proyecto, empezando con que en la parte de decodificador se logró crear un entorno que permite una visualización más agradable y fácil de entender. Gracias a la documentación se llegó a la conclusión que es mejor utilizar Big endian para este proyecto ya que no está pensado en que los bits estén volteados. Finalmente por la parte de Verilog se implementaron nuevos módulos para darle más optimización al proyecto

**Bibliografía**

<https://scholar.google.com/citations?user=zuhBWukAAAAJ>

<https://www.studocu.com/es-mx/document/instituto-de-estudios-universitarios-ac/arquitectura-de-computadoras/procesador-mips-3modos-de-direccionamiento-y-cpu/77928875>

<https://informatecdigital.com/hardware/arquitectura-computadoras-introduccion-a-su-evolucion-y-diseno/>

<https://informatica.uv.es/seguia/FAC/Archivos/MIPS32.pdf>

<https://www.docsity.com/es/principales-caracteristicas-del-procesador-mips-y-de-la-arquitectura-que-implementa/7986545/>

<https://elchapuzasinformatico.com/2020/02/jim-keller-revela-que-creara-una-nueva-arquitectura-cpu-para-intel-cada-5-anos/>

<https://hardzone.es/2018/04/26/intel-ficha-jim-keller/>

<https://www.muycomputer.com/2023/03/22/raja-koduri-y-su-aventura-en-intel-una-mirada-a-fondo/>

<https://www.freecodecamp.org/news/what-is-endianness-big-endian-vs-little-endian/>

<https://www.baeldung.com/cs/big-endian-vs-little-endian>

<https://www.rapidtables.com/prog/endianess.html>

<https://jmjinformatico.es/arquitecturas-von-neumann-y-harvard/>