Laboratorio 3: Taller de Diseño Digital
Tecnológico de Costa Rica

Bryan Esquivel Flores
2020035806
zayus@estudiantec.cr
Alajuela, Costa Rica

Freddy Mora Bolaños
2021040443
freddy.mora@estudiantec.cr
Alajuela, Costa Rica

Andrés Vargas Arce
2018151379
anbo80@estudiantec.cr
Alajuela, Costa Rica

November 29, 2024

1 RISCV

En la página principal de RISCV[1] encontramos lo siguiente:

> RISC-V es un estándar abierto de Arquitectura de Conjunto de Instrucciones (ISA) que permite una nueva era de innovación en procesadores mediante la colaboración abierta.

En [2] podemos encontrar las instructions básicas utilizadas en RV32I,

RV32I Base Instruction Set

imm[31:12]				rd	0110111	LUI
imm[31:12]				rd	0010111	AUIPC
imm[20|10:1|11|19:12]				rd	1101111	JAL
imm[11:0]		rs1	000	rd	1100111	JALR
imm[12|10:5]	rs2	rs1	000	imm[4:1|11]	1100011	BEQ
imm[12|10:5]	rs2	rs1	001	imm[4:1|11]	1100011	BNE
imm[12|10:5]	rs2	rs1	100	imm[4:1|11]	1100011	BLT
imm[12|10:5]	rs2	rs1	101	imm[4:1|11]	1100011	BGE
imm[12|10:5]	rs2	rs1	110	imm[4:1|11]	1100011	BLTU
imm[12|10:5]	rs2	rs1	111	imm[4:1|11]	1100011	BGEU
imm[11:0]		rs1	000	rd	0000011	LB
imm[11:0]		rs1	001	rd	0000011	LH
imm[11:0]		rs1	010	rd	0000011	LW
imm[11:0]		rs1	100	rd	0000011	LBU
imm[11:0]		rs1	101	rd	0000011	LHU
imm[11:5]	rs2	rs1	000	imm[4:0]	0100011	SB
imm[11:5]	rs2	rs1	001	imm[4:0]	0100011	SH
imm[11:5]	rs2	rs1	010	imm[4:0]	0100011	SW
imm[11:0]		rs1	000	rd	0010011	ADDI
imm[11:0]		rs1	010	rd	0010011	SLTI
imm[11:0]		rs1	011	rd	0010011	SLTIU
imm[11:0]		rs1	100	rd	0010011	XORI
imm[11:0]		rs1	110	rd	0010011	ORI
imm[11:0]		rs1	111	rd	0010011	ANDI
0000000	shamt	rs1	001	rd	0010011	SLLI
0000000	shamt	rs1	101	rd	0010011	SRLI
0100000	shamt	rs1	101	rd	0010011	SRAI
0000000	rs2	rs1	000	rd	0110011	ADD
0100000	rs2	rs1	000	rd	0110011	SUB
0000000	rs2	rs1	001	rd	0110011	SLL
0000000	rs2	rs1	010	rd	0110011	SLT
0000000	rs2	rs1	011	rd	0110011	SLTU
0000000	rs2	rs1	100	rd	0110011	XOR
0000000	rs2	rs1	101	rd	0110011	SRL
0100000	rs2	rs1	101	rd	0110011	SRA
0000000	rs2	rs1	110	rd	0110011	OR
0000000	rs2	rs1	111	rd	0110011	AND
fm	pred | succ	rs1	000	rd	0001111	FENCE
1000	0011 | 0011	00000	000	00000	0001111	FENCE.TSO
0000	0001 | 0000	00000	000	00000	0001111	PAUSE
000000000000		00000	000	00000	1110011	ECALL
000000000001		00000	000	00000	1110011	EBREAK

Figura 1: Instrucciones básica RV32I, para números enteros

Dichas instrucciones se dividen en diferentes tipos, estos serían tipo R,I,S,B,U y J, cada uno tiene un formato diferente, este formato indica los datos de cada instrucción a ejercutar,

31 27	26 25	24 20	19 15	14 12	11 7	6 0	
funct7		rs2	rs1	funct3	rd	opcode	R-type
imm[11:0]			rs1	funct3	rd	opcode	I-type
imm[11:5]		rs2	rs1	funct3	imm[4:0]	opcode	S-type
imm[12|10:5]		rs2	rs1	funct3	imm[4:1|11]	opcode	B-type
imm[31:12]					rd	opcode	U-type
imm[20|10:1|11|19:12]					rd	opcode	J-type

Figura 2: Formato instrucciones.

Estas instrucciones son de 32 bits cada una.

2 Toolchain

Para la progamción en RISCV se utiliza el *RISCV GNU Toolchain*.

Primero, de [3]

"GNU es un sistema operativo similar a Unix. Esto significa que es una colección de muchos programas: aplicaciones, bibliotecas, herramientas de desarrollo e incluso juegos. El desarrollo de GNU, iniciado en enero de 1984, se conoce como el Proyecto GNU. Muchos de los programas en GNU se publican bajo los auspicios del Proyecto GNU; a estos los llamamos GNU packages."

Uno de estos GNU packages es el RISCV GNU Toolchain, las instrucciones para su compilación se encuentran en Riscv Collab.

Una vez se instala el toolchain ya se puede programar en RISCV, la simulación se hace por medio de Spike. Spike, es un simulador de ISA RISC-V, implementa un modelo funcional de uno o más versiones de RISC-V.

3 Mapa de memoria de un procesador

Es una representación detallada de cómo se organizan y gestionan las diferentes regiones de memoria en un sistema computacional. En el libro [4], la elaboración de un mapa de memoria involucra varias consideraciones fundamentales acerca de la arquitectura de memoria, la jerarquía de memoria, la asignación de direcciones y la interacción entre hardware y software. A continuación, se detallan los pasos y principios fundamentales para crear un mapa de memoria:

3.1 Definir las Regiones de Memoria

El primer paso en la elaboración de un mapa de memoria es definir las distintas regiones de memoria que el procesador y el sistema pueden utilizar. Estas regiones incluyen:

3.1.1 Memoria de solo lectura (ROM):

Usada para almacenar el firmware o el sistema operativo inicial.

3.1.2 Memoria de acceso aleatorio (RAM):

Utilizada para almacenamiento temporal, como variables y pilas de ejecución.

3.1.3 Memoria de almacenamiento externo:

Como discos duros o SSDs, que son más lentos que la RAM, pero tienen mucha más capacidad.

3.1.4 Áreas para dispositivos de entrada/salida (I/O):

Direcciones de memoria mapeadas para interactuar con periféricos como teclados, pantallas y sensores.

3.1.5 Memoria caché:

Espacio de memoria de alta velocidad que se usa para almacenar datos e instrucciones que el procesador necesita frecuentemente. Estos diferentes tipos de memoria son accedidos y gestionados de manera jerárquica, desde la memoria de acceso rápido (caché) hasta la más lenta (almacenamiento externo).

3.2 Asignación de Direcciones de Memoria

Una vez definidas las regiones de memoria, el siguiente paso es asignar direcciones específicas a cada una de ellas. El procesador y el sistema operativo se encargan de gestionar este espacio. En un procesador de 32 bits, por ejemplo, el espacio de direcciones es de 4 GB (2^{32} direcciones). Las direcciones de memoria se asignan en bloques a diferentes áreas: la memoria RAM generalmente ocupa las direcciones centrales, mientras que la ROM y los periféricos suelen estar en los extremos del espacio de direcciones. En el caso de los sistemas de memoria de 32 bits, el espacio de direcciones es limitado, y por lo tanto se debe gestionar cuidadosamente qué parte del espacio de direcciones corresponde a cada tipo de memoria o periferico. Los procesadores pueden segmentar el espacio de direcciones en bloques contiguos para facilitar la asignación de recursos.

3.3 Jerarquía de Memoria

El mapa de memoria debe tener en cuenta la jerarquía de memoria, es decir, la organización de diferentes niveles de almacenamiento con base en velocidad y costo:

3.3.1 Caché L1 y L2:

Estas son memorias rápidas que están muy cerca de la CPU y son usadas para almacenar datos e instrucciones de uso frecuente. La memoria caché mejora significativamente el rendimiento al reducir los tiempos de acceso a la memoria.

3.3.2 RAM principal (DRAM):

Almacena datos que son utilizados por los programas en ejecución. Aunque más lenta que la caché, la RAM tiene una capacidad mucho mayor.

3.3.3 Almacenamiento secundario (disco duro, SSD):

Es mucho más lento que la RAM, pero tiene una capacidad mucho mayor y se usa para almacenamiento persistente de datos.

3.4 Mapeo de Periféricos en la Memoria

En muchos sistemas, los periféricos están mapeados a direcciones de memoria, lo que significa que el procesador accede a ellos de la misma manera que accede a la RAM o ROM, utilizando las instrucciones de carga y almacenamiento. Los dispositivos de I/O (como puertos de red, controladores de pantalla o teclados) están asignados a rangos específicos de direcciones en el mapa de memoria.

3.5 Técnicas de Administración de Memoria

Los procesadores modernos emplean técnicas avanzadas de administración de memoria para maximizar el rendimiento y la seguridad:

3.5.1 Segmentación y paginación:

Son técnicas que permiten dividir el espacio de direcciones en segmentos o páginas más pequeñas, para facilitar la asignación y protección de memoria.

3.5.2 Direcciones virtuales y físicas:

El libro detalla cómo los procesadores modernos utilizan un sistema de direcciones virtuales que el sistema operativo convierte en direcciones físicas mediante el uso de una tabla de páginas. Estas técnicas permiten a los sistemas operativos administrar múltiples procesos y protegen la memoria de un proceso de ser modificada accidentalmente por otro proceso.

4 Memoria RAM y ROM FPGA

Las memorias RAM y ROM de las FPGA son totalmente configurables por medio de Verilog. Si se utiliza Vivado, se puede utilizar la herramienta de IPCores para configurar la memoria de las FPGA soportadas.

Aunque estas son más limitadas, estas no son tan capaces como las que se tienen usualmente en las computadoras.

References

[1] *About RISC-V – RISC-V international*, en, `https://riscv.org/about/`, Accessed: 2024-10-12.

[2] F. Embeddev, *RISC-V instruction set manual, volume i: RISC-V user-level ISA*, en, `https://five-embeddev.com/riscv-user-isa-manual/Priv-v1.12/instr-table.html`, Accessed: 2024-10-10.

[3] G. Who? *The GNU operating system and the free software movement*, en, `https://www.gnu.org/`, Accessed: 2024-10-12.

[4] D. A. P. y John L. Hennessy, *Computer Organization and Design: The Hardware/Software Interface*, 5th ed. Morgan Kaufmann, 2013, ISBN: 978-0124077263.