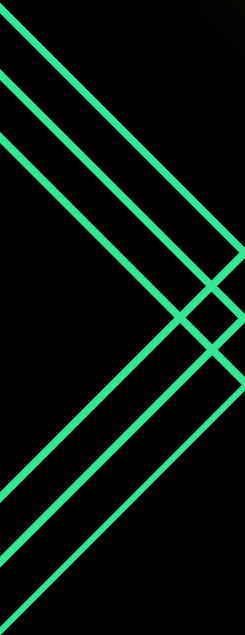


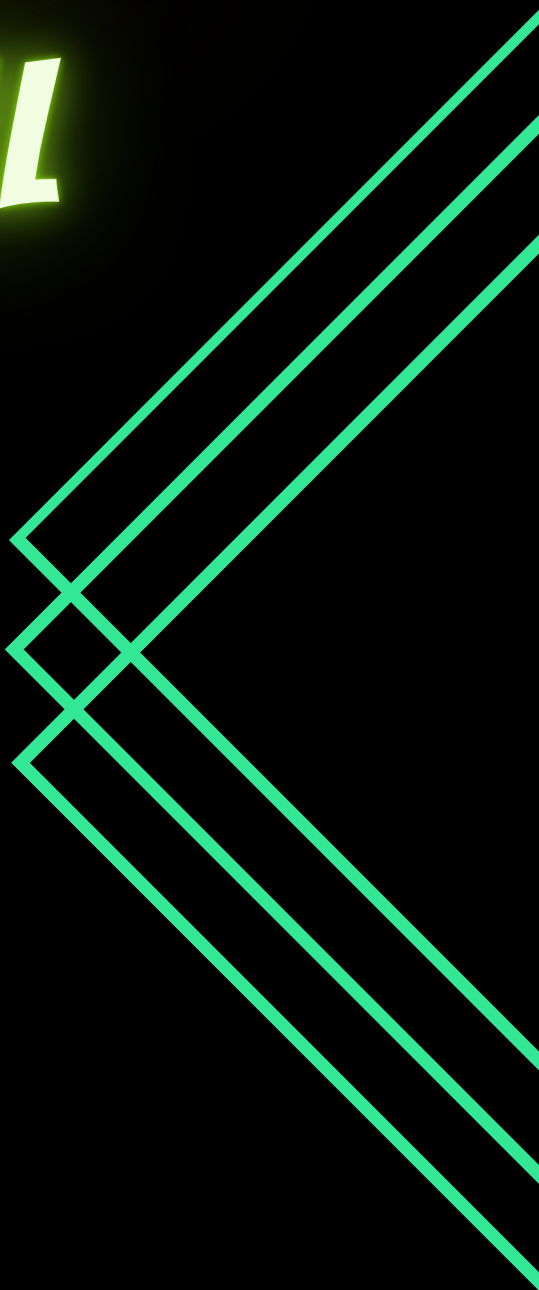


Reporte

EQUIPO NULL



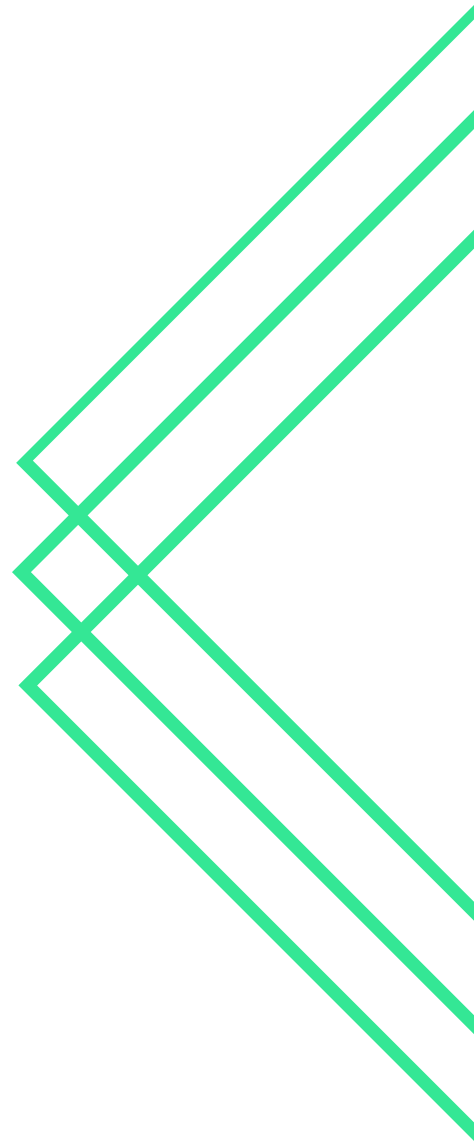
Díaz López Juan Enrique
Vences Arellano Ángel Yahir
Ortiz Pérez Emiliano



Procesador MIPS de 32 bits

El procesador MIPS (Microprocessor without Interlocked Pipeline Stages) de 32 bits es una arquitectura de procesador basada en RISC (Reduced Instruction Set Computing). A continuación se detallan algunos de sus elementos y características generales:

1. Unidad de control (Control Unit): es la parte del procesador que se encarga de controlar el flujo de datos y de instrucciones a través de las diferentes unidades del procesador.
2. Unidad aritmético-lógica (Arithmetic Logic Unit, ALU): es la unidad encargada de realizar las operaciones aritméticas y lógicas básicas, como sumar, restar, multiplicar, dividir, comparar y realizar operaciones booleanas.
3. Registros: el procesador MIPS cuenta con 32 registros de propósito general, cada uno de ellos de 32 bits de longitud. Estos registros son utilizados para almacenar datos temporales y direcciones de memoria.
4. Memoria caché: el procesador MIPS utiliza una memoria caché para acelerar el acceso a los datos almacenados en memoria. La memoria caché es una memoria de acceso rápido que almacena copias de los datos más utilizados en la memoria principal.
5. FPU (Floating Point Unit): es la unidad encargada de realizar operaciones aritméticas en punto flotante, como sumas, restas, multiplicaciones y divisiones.
6. Pipeline: el procesador MIPS utiliza una arquitectura de pipeline para mejorar el rendimiento. El pipeline permite que varias instrucciones se ejecuten en paralelo en diferentes etapas del procesador.
7. Conjunto de instrucciones: el conjunto de instrucciones de MIPS es relativamente pequeño, con un total de 32 instrucciones básicas. Esto simplifica el diseño del procesador y facilita la programación.



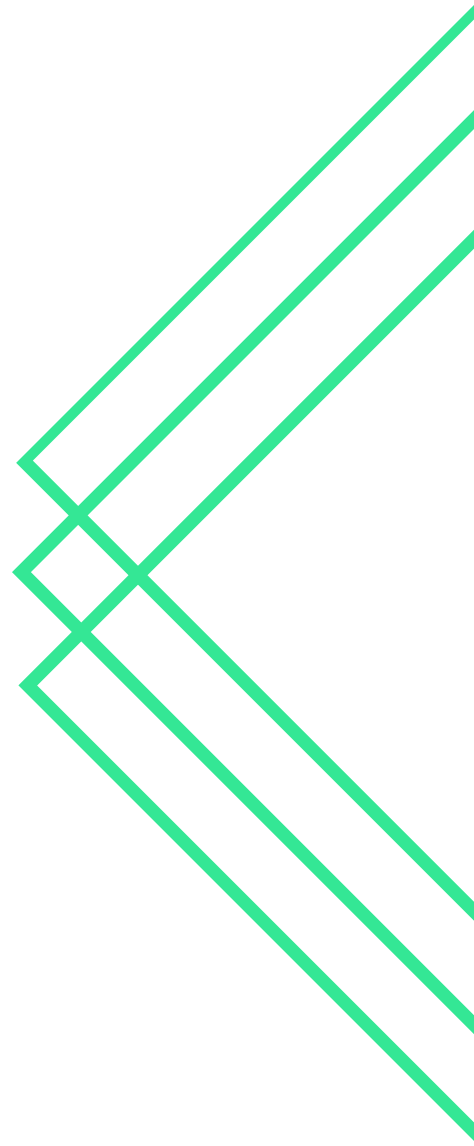
Objetivos

Diseñar un “datapath” con arquitectura tipo MIPS de 32 bits capaz de ejecutar las 28 instrucciones previamente definidas por ustedes, complete la tabla 1 y tabla 2 con la sintaxis y 5 instrucciones elegidas por usted.

Debe de elegir un algoritmo previamente aprobado por su profesor, y que sea posible implementar con el set reducido de instrucciones de la tabla 1 y 2. Este programa

previamente definido en ensamblador debe ser codificado a código binario y pre-cargado en la memoria de instrucciones para que el datapath lo ejecute, recuerde

definir cada uno de los aspectos de dicho programa, secciones de los registros en el banco de registros para base pointers, resultados, resultado de comparaciones, etc.
Asi como los datos pre-cargados en su memoria de datos.



Set de instrucciones

INSTRUCCION ADD:

La instrucción "add" es una instrucción de tipo R utilizada en la arquitectura de computadoras y en el lenguaje de programación ensamblador. La instrucción "add" se utiliza para sumar dos operandos y almacenar el resultado en un registro de destino.

INSTRUCCION SUB:

La instrucción "sub" es una instrucción de tipo R utilizada en la arquitectura de computadoras y en el lenguaje de programación ensamblador. La instrucción "sub" se utiliza para restar dos operandos y almacenar el resultado en un registro de destino.

INSTRUCCION OR:

La instrucción "or" es una instrucción de tipo R utilizada en la arquitectura de computadoras y en el lenguaje de programación ensamblador. La instrucción "or" se utiliza para realizar una operación lógica OR a nivel de bit entre dos operandos y almacenar el resultado en un registro de destino.

INSTRUCCION AND:

La instrucción "and" es una instrucción de tipo R utilizada en la arquitectura de computadoras y en el lenguaje de programación ensamblador. La instrucción "and" se utiliza para realizar una operación lógica AND a nivel de bit entre dos operandos y almacenar el resultado en un registro de destino.

INSTRUCCION SLT:

La instrucción "slt" es una instrucción de tipo R utilizada en la arquitectura de computadoras y en el lenguaje de programación ensamblador. La instrucción "slt" se utiliza para comparar dos valores y almacenar el resultado de la comparación en un registro de destino.

INSTRUCCION NOP:

La instrucción "nop" es una instrucción de tipo R utilizada en la arquitectura de computadoras y en el lenguaje de programación ensamblador. La instrucción "nop" (abreviatura de "no operation" o "sin operación" en español) se utiliza para indicar que no se realizará ninguna operación en la ejecución de una instrucción y se utiliza comúnmente como un espacio reservado para futuras expansiones del programa.

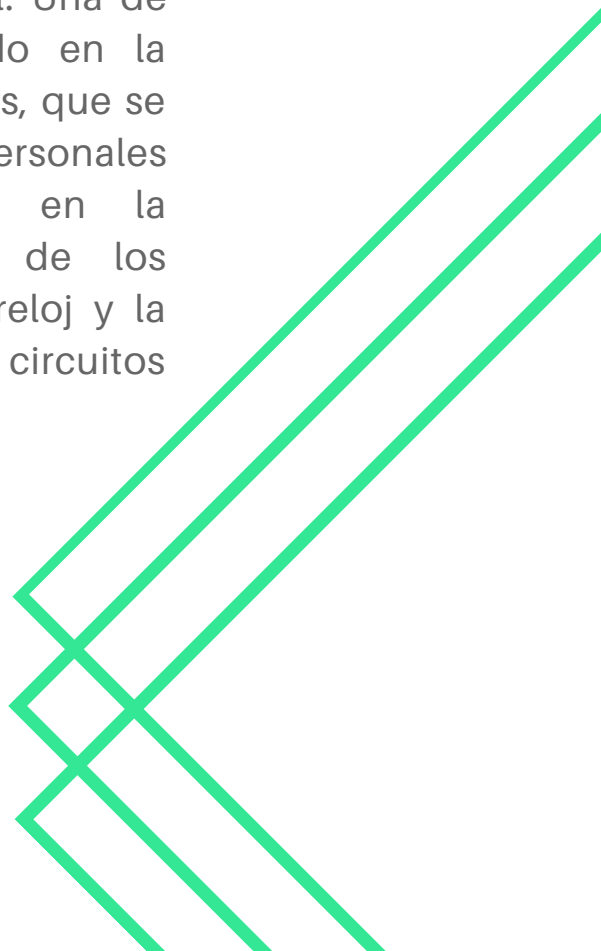
Personajes

LJUBISA BAJIC:

Ljubisa Bajic es un ingeniero en computación que ha trabajado en el diseño de microprocesadores durante más de 30 años. Una de sus contribuciones más importantes en microarquitectura ha sido en la técnica de pipeline en paralelo, que permite que el procesador ejecute múltiples instrucciones simultáneamente. Bajic también ha trabajado en la mejora de la eficiencia energética de los procesadores y ha desarrollado técnicas para reducir el consumo de energía de los circuitos de procesamiento de datos.

JIM KELLER:

Jim Keller es otro ingeniero en computación que ha sido fundamental en la evolución de los microprocesadores. Keller ha trabajado en el diseño de procesadores de alta gama durante más de 20 años y ha liderado equipos de diseño en empresas como AMD, Apple e Intel. Una de sus contribuciones más importantes ha sido en la arquitectura de los procesadores x86 de 64 bits, que se utilizan en la mayoría de los ordenadores personales modernos. Keller también ha trabajado en la optimización de la eficiencia energética de los procesadores, la mejora de la velocidad de reloj y la reducción del tamaño de los transistores en los circuitos integrados.



Instrucciones de nuestro algoritmo

Las instrucciones de tipo R que puede realizar nuestro algoritmo son dos:

La instrucción ADD la cual nos ayudara a realizar cierta suma de digitos y a su vez almacenar el resultado en una cierta direccion del banco de registros.

la instruccion DIV esta nos ayudara a dividir el contenido de un registro por otro registro o un valor inmediato, y almacenar el resultado en otro registro.

