Resumen MIPS32

Ángel Besteiro

June 3, 2025

Introducción al Lenguaje Máquina y MIPS32

El lenguaje máquina es un concepto fundamental en el trabajo con computadoras, compuesto por "instrucciones" o "pasos a seguir" que el equipo ejecuta para completar tareas específicas.Aunque existen numerosos lenguajes de programación con similitudes en sus requerimientos y propósitos, cada uno posee sus propias ventajas, desventajas y características adicionales.

El MIPS32 es un lenguaje de bajo nivel, cercano al lenguaje máquina, donde el procesador opera con instrucciones binarias.Este lenguaje permite dar instrucciones que son traducidas a binario y, a su vez, retorna resultados que son interpretados del binario para su comprensión.El lenguaje al que se refiere este escrito es MIPS32 ensamblador se basa en una serie de instrucciones preestablecidas para agilizar el código y el flujo del algoritmo.

Memoria y Registros

El sistema cuenta con memoria y registros.La memoria es donde se almacenan datos de distinto tipo.Los registros también son memoria, pero son espacios donde el sistema puede operar directamente.Para usar los datos en operaciones, estos deben ser movilizados de la memoria a un registro y luego devueltos a su origen o a otra ubicación en la memoria.Se dispone de 32 registros, algunos de uso específico y otros más generales.

Tipos de Operaciones

Operaciones Aritméticas

Estas operaciones son básicas en cualquier sistema e incluyen suma y resta.

- **add:** Formato: `add($s1,$s2,$s3)`. [span$_1$1]($start_span$)*Realizalasumadelasdosúltimas*"*variables*"(
`addi($s1,$s2,100)`.

- **sub (subtract):** Funciona como una resta. Formato: `sub($s1,$s2,$s3)`. Realiza la diferencia de los últimos componentes (`$s2` y `$s3`) y la coloca en el primero (`$s1`).

Operaciones de Transferencia de Datos

Instrucciones específicas para movilizar datos entre espacios de memoria y registros.

- **load word (lw):** Moviliza una palabra de una dirección de memoria a un registro.
- **store word (sw):** Moviliza una palabra de un registro a una dirección de memoria.
- **load half (lh):** Carga media palabra de memoria a registros.
- **store half (sh):** Carga media palabra de registros a memoria.
- **load byte (lb):** Carga un byte de memoria a registros.
- **store byte (sb):** Carga un byte de registros a memoria.
- **load upper immediate (lui):** Carga una constante de 16 bits en la parte superior de un registro.

Operaciones Lógicas

Instrucciones que contemplan expresiones lógicas.

- **and:** Opera bit a bit con tres registros, implementando la compuerta AND.
- **or:** Opera bit a bit con tres registros, implementando la compuerta OR.
- **not:** Opera bit a bit con tres registros, implementando la compuerta NOT.
- **andi (and immediate):** Opera bit a bit con registros y una constante, implementando la compuerta AND.
- **ori (or immediate):** Opera bit a bit con registros y una constante, implementando la compuerta OR.
- **sll (shift left logical):** Desplazamiento lógico a la izquierda por una constante.
- **srl (shift right logical):** Desplazamiento lógico a la derecha por una constante.

Salto Condicional

Instrucciones para control de flujo basadas en condiciones.

- **beq (branch on equal):** Comprueba la igualdad.
- **bne (branch on not equal):** Comprueba la no igualdad.
- **slt (set on less than):** Compara si es menor que.
- **slti (set on less than immediate):** Compara si es menor que una constante.

Salto Incondicional

(start$_s pan) Instrucciones para saltos directos en el flujo del programa.

j (jump): Salta a un destino.

jr (jump register): Utilizado para retornar de un procedimiento.

jal (jump and link): Utilizado para la llamada a un procedimiento.