Universidad Nacional Autónoma de México

Facultad de Ingeniería

Diseño Digital VSLI

Alumno: Alfonso Murrieta Villegas

1. **¿Qué es un procesador ?**

Es un circuito integrado responsable de las ejecuciones de cálculos, direccionamiento y también de tomar todas las decisiones lógicas que como resultado nos dan diferentes acciones en un sistema digital o de computo.

Además un procesador tiene lo que se denomina como pipeline que es todo este conjunto de etapas por las que las instrucciones o nmónicos pasan a lo largo de este, algunos ejemplos de estas etapas pueden ser el fetch, decoder o execute.

2. **Tipos de procesadores hay**

Existen muchas formas de clasificar a los procesadores, a continuación algunas de estas formas:

2.1 Mediante su Arquitectura

RISC = Reduced Instruction Set Computing, son aquellos que se identifican principalmente por tener una cantidad fija y pequeña de instrucciones

CISC = Complex Instruction Set Computing, son aquellos que tiene mucha mayor cantidad de instrucciones lo que hace que el hacer operaciones o acciones más cpmplejas resulte mucho más fácil de programarlas además de contener menos líneas de código

2.2 Mediante su propósito

ASIC = Los circuitos integrados de aplicación específica se construyen para aplicaciones específicas.

DSP = Sirven principalmente para el proceso digital de señales

3. **¿Qué es MIPS, y cuáles son los tres tipos de instrucciones: R, J y I ?**

MIPS o en inglés " **M**icroprocessor without **I**nterlocked **P**ipeline **S**tages" es un tipo especifico de microprocesador de arquitectura RISC, el cual cada instrucción que se realiza se divide en distintas etapas como son Fetch, Decode, Execute y writeback.

Sus tres tipos de instrucciones son

- R = Es utilizado para las instrucciones aritméticas y lógicas.
    - (6 bits opcode,5 bits rs, 5 bits rt, 5 bits rd, 5 bits shamt, 6 bits function code)

- I = Son las instrucciones de transferencias comúnmente utilizadas en condiciones e instrucciones con operandos inmediates.
    - (6 bits opcode,5 bits rs, 5 bits rt, 16 bits function code)
- J = Utilizado principalmente para operaciones de bifurcación, es decir saltos a direcciones de memoria en específico.
    - (6 bits opcode,26 bits pseudo-direct address)

4. **¿ Cuál es la diferencia entre la "alu automatizada" y un procesador "microAlu", qué le falta o sobra?**

La principal diferencia y algo que debemos entender es que nuestra ALU es un elemento que compone a nuestro microALU (Procesador), por lo que al momento de saber que falta o sobra lo podemos asociar directamente a los elementos que no contiene un ALU:

1. Nuestra ALU no contiene una unidad de **decodificación** para los datos asumidos o leídos previamente por una unidad **fetch**, y esto solamente hablando respecto a la forma en que se procesan los datos, es decir respecto al pipeline del procesamiento de datos
2. Realmente, el ALU que realizamos sólo se encarga de realizar las operaciones (Tanto lógicas como aritmétcias) que directamente le mandemos, sin embargo y la mayor diferencia respecto a nuestro microALU (Procesador) es todo el pipeline por el que una **instrucción** (La cuál no sólo lleva valores) debe ser procesada.
3. En la ALU sólo hay valores y resultados, en nuestra microALU en cambio, ya se habla de **instrucciones** que de por medio llevan direcciones de memoria, valores, tipo de operación, etc.

5. **¿De qué sirve hacer un procesador, y cuál seria la diferencia de hacer el sistema solo como "máquinas de estado" ?**

El construir un procesador incluso aunque sea básico nos hace realmente entender desde que lo conforma físicamente hasta cómo es que una instrucción es procesada además de todas las etapas por las que debe pasar internamente.

Además el conocer internamente además del pipeline y work flow de un procesador nos ayuda a poder desarrollar habilidades respecto a arquitectura y desarrollo de computadoras.

Por otro lado, realmente cualquier sistema que tenga entradas y salidas, además de que esten retroalimentadas se pueden modelar mediante una máquina de estados, sin embargo, la complejidad que implicaría el representar un sistema con solamente máquinas de estados podría resultar en algo realmente complicado, es por ello que al tratar con entidades de mayor complejidad pero con propósitos generales nos facilita el abordar, construir e implementar cualquier sistema.

El tratar con entidades de propósito general además de estar en una capa de abstracción mayor facilita mucho la creación y aplicación de estos.

6. **¿ Por qué hacer un MIPS? ¿Para qué? ¿Qué se gana?**

Porque es un microprocesador básico además de al tener una arquitectura RISC la cantidad de mnemónicos o instrucciones son pocas por lo que resultad mucho más fácil de programar respecto a un CISC.

Para poder entender cada una de las etapas por las que las instrucciones pasan dentro de un procesador, esto con el propósito de comprender una capa de abstracción de computo mucho más profundo respecto al nivel de hardware y software.

Por último, se gana realmente el entender y saber como funciona uno de los elementos más elementales y circunstanciales de cualquier computadora moderna.

7. **Escribe las instrucciones para la microALU para hacer un "if" de si a es mayor que b entonces c=a+b sino d= a-b NOTA: ¿ Si podría hacerlo este procesador? Requerimos el Jump**

   Opción: escoger un problema c= a+b; d= c+1; tipo R (este es sin aplicación)

- Lo primero que debemos notar es que debido a que en la condición se hace uso de datos previamente guardados en memoria, necesitamos sí o sí un salto a esas respectivas localidades de memoria para poder acceder a los dato, por lo que con el micrALU que se tiene en este momento no es posible hacer.

  Por otro lado, para poder hacer el if podría usarse la siguiente tabla reducida mediante mapas de Karnough:

A	B	A>B	A<B	A=B
0	0	0	0	1
0	1	0	1	0
1	0	1	0	0
1	1	0	0	1

Donde por ejemplo si queremos hacer A = B tendríamos que hacer:

```
XOR A,B, Aux // Aquí usaríamos un nmónico de 3 no de dos

NOT Aux
```

- Descripción de cómo hacer un **c = a - b**, a continuación se muestra la instrucción correspondiente:

00 0110 001 001 0001

Tal que:

- Los primeros 2 bits no tienen uso en esta arquitectura
- Los siguientes 4 bits son los asociados al opcode, es decir, los que están relacionados con el multiplexor de nuestra ALU
- Los siguientes 3 bits son los asociados con la dirección en A (rs)
- Los siguientes 3 bits son los asociados con la dirección en B (rt)
- Los siguientes 4 bits son los asociados con la dirección en C (rd)

8. **¿ Qué le falta al procesador MicroALU para ser mejor? Detalla al menos dos cosas**

1. La primera mejora o complemento a nuestro microALU sin duda alguna sería el poder tener las demás instrucciones o modos de direccionamiento de un MIPS como es el caso de las instrucciones I (Condiciones) y J (Saltos y bifurcaciones), esto debido a que a pesar de ser un

procesador RISC, realmente el implementado actualmente carece de muchas instrucciones que podrían ser utiles para realizar acciones más complejas.

2. Sin duda al hablar de condiciones, bifurcaciones y de saltos en las instrucciones de tipo I y J, necesitamos sin duda una etapa de retroalimentación dentro de nuestro microALU, pues como se puede apreciar en el ejercicio anterior (7) no podemos realizar condicionamientos o incrementos en variables debido a que no se dispone del dato guardado anteriormente al procesado en el momento.

ANEXO

2. **Tipos de procesadores hay**

Con base a la arquitectura de computadora y con base a la arquitectura basada en complejidad de instrucciones del procesador podríamos asimilar de la siguiente forma de categorizar los procesadores.

NOTA: En la siguiente tabla se muestran algunos ejemplos comerciales

	Von Neuman	**>Harvard**
RISC	ARM7	ARM9
CISC	Pentium	SHARC (DSP)