IIC2343

Home Content IO

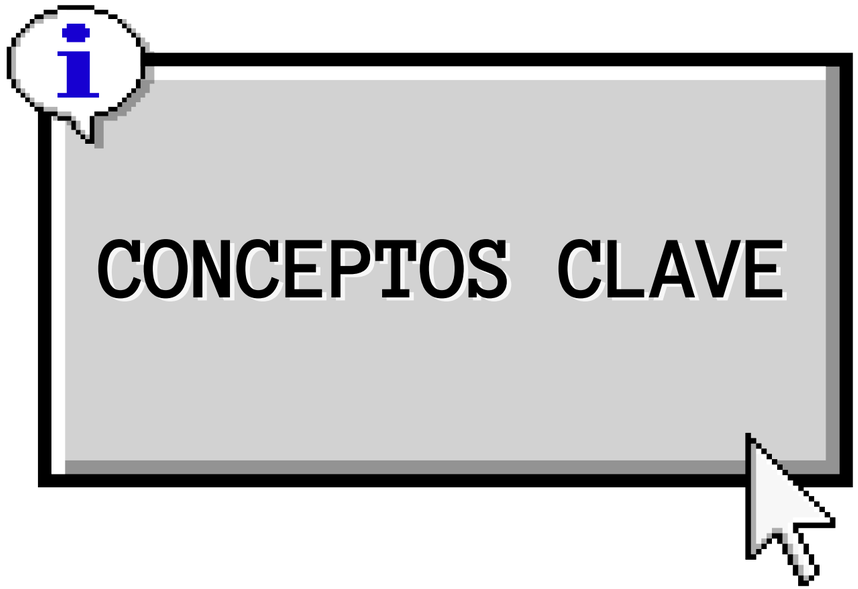
Repaso I2

ARQUI+I/O

JOAQUIN & DANIELA



Start



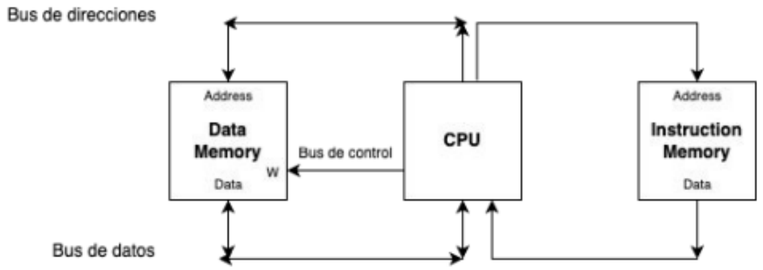
Microarquitectura: Componentes de hardware presentes en el computador

ISA: Tipo, formato y características de las instrucciones soportadas por el computador. **Referido a su programación, cómo se escriben los programas.**

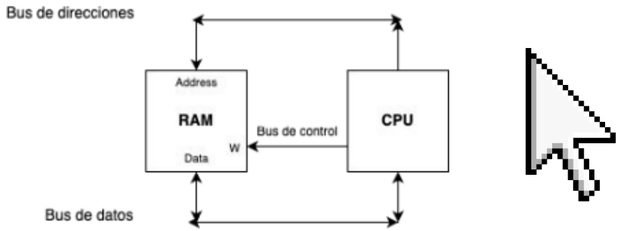
TIPOS DE MICROARQUITECTURA

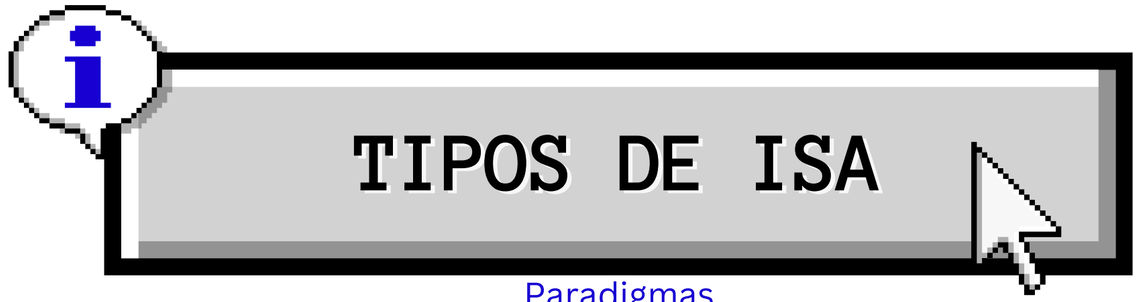
Paradigmas

Arquitectura Harvard: Memoria de datos e instrucciones independientes.



Arquitectura Von Neumann Memoria única que comparte datos e instrucciones. Permite escribir datos como si fueran instrucciones (auto programabilidad).





Paradigmas

RISC: Instrucciones pequeñas y simples.

Las ISA RISC requieren menos gasto energético y suelen usarse en **dispositivos móviles**.

CISC: Muchas instrucciones y con complejidad alta.

Las ISA CISC facilitan la programación, pero gastan más energía, por lo que se usan más en **PC y laptops.**



TIPOS DE ISA

Categorías

Register-Memory: Arquitectura que permite realizar operaciones directamente desde y hacia la memoria.

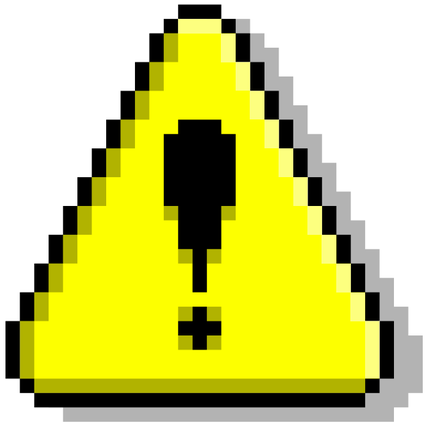
Ejemplo:

1. ADD A, [B] → suma el contenido de la dirección B directamente al registro A.

Load-Store: Arquitectura que separa las operaciones de lectura/escritura de las que usan la ALU. Hay que almacenar los datos de memoria en registros para poder operar

Ejemplo:

1. LOAD R1, [B]
2. ADD A, R1





OISC (One Instruction Set Computer) es un tipo de ISA que usa una sola instrucción para realizar cualquier computación.

Ejemplo: SUBLEQ a,b,c
- Mem[a] = Mem[a] - Mem[b]
- if Mem[a] ≤ 0 → JMP c

IIC2343

Home **Content** IO

DIFERENCIA CLAVE RISC VS CISC

```
LOAD R1, A
LOAD R2, B
MULTIPLY_LOOP:
   ADD R3, R3, R1
   SUB R2, R2, #1
   CMP R2, #0
   BNE MULTIPLY_LOOP
```

RISC Load-Store

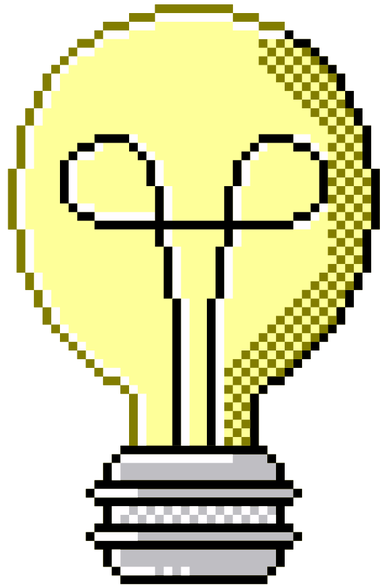
```
MUL A, B  ; Multiplica A por B
```

CISC



NUESTRO COMPUTADOR

El computador construido en el curso presenta una arquitectura de tipo **RISC Register-Memory**



HARVARD VS VON NEUMANN

¿Qué ventajas y desventajas presenta la arquitectura Harvard en comparación a Von Neumann en términos de seguridad y flujo de ejecución?



HARVARD VS VON NEUMANN

¿Qué ventajas y desventajas presenta la arquitectura Harvard en comparación a Von Neumann en términos de seguridad y flujo de ejecución?

Harvard separa físicamente memoria de datos e instrucciones, lo que reduce el riesgo de que datos maliciosos se ejecuten como código. En cambio, Von Neumann permite que los datos se traten como instrucciones, lo cual aumenta la flexibilidad pero también los riesgos de seguridad.

ISA



¿Por qué es posible tener dos computadores con la misma microarquitectura pero distintas ISA?



ISA



¿Por qué es posible tener dos computadores con la misma microarquitectura pero distintas ISA?

Porque la microarquitectura define el hardware interno (registros, ALU, control, etc.), mientras que la ISA define cómo se programa ese hardware. Podemos tener dos procesadores físicamente iguales que interpreten diferentes sets de instrucciones.



COMPILACION EN C



Un programa en C es compilado para una ISA CISC y otra RISC. ¿Cual de las dos compilaciones tendrá como resultado un código de mayor tamaño y por qué ocurre esto?



COMPILACION EN C

Un programa en C es compilado para una ISA CISC y otra RISC. ¿Cual de las dos compilaciones tendrá como resultado un código de mayor tamaño y por qué ocurre esto?

El código para la ISA RISC tendrá un tamaño de compilación mayor, ya que RISC define sets de instrucciones muy simples que redundan en una mayor cantidad de instrucciones en comparación a CISC para hacer una tarea específica





DISPOSITIVOS I/O

Yes

Cancel

Aprovechen de revisar las cápsulas y el contenido de clases para apoyar el estudio :)

I/O



Explique el significado y uso del identificador que envía el controlador de interrupciones luego de recibir la señal INTA por parte de la CPU.



I/O



Explique el significado y uso del identificador que envía el controlador de interrupciones luego de recibir la señal INTA por parte de la CPU.

Al recibir la señal INTA, la CPU espera obtener el identificador asociado al dispositivo cuya interrupción será atendida. Este será enviado por el controlador de interrupciones. Luego, la CPU hace uso de este valor como índice del vector de interrupciones para obtener el puntero de la ISR del dispositivo atendido, permitiendo así la ejecución de su subrutina.



I/O



Cuando usted trabaja en su computador, puede seguir haciendo uso del mismo mientras transfiere documentos de este a dispositivos externos y vice-versa. ¿A qué se puede deber esto? Justifique su respuesta en base a argumentos relativos a la comunicación entre la CPU y dispositivos I/O.



I/O



Cuando usted trabaja en su computador, puede seguir haciendo uso del mismo mientras transfiere documentos de este a dispositivos externos y vice-versa. ¿A qué se puede deber esto? Justifique su respuesta en base a argumentos relativos a la comunicación entre la CPU y dispositivos I/O.

Esto se debe al uso del controlador DMA (Direct Memory Access). Este se encarga de llevar acabo la transferencia de datos entre los dispositivos I/O y la memoria del computador, evitando que la CPU ocupe tiempo de cómputo en esto y, permitiendo así, la ejecución de otros programas.



APLICACIÓN I/O

(c) (4 ptos.) Suponga que existe un componente de control que implementa la ISA de RISC-V para un sistema de monitoreo de trenes. Este tiene puertos de entrada y salida, cuyos registros de estado y comando son de 32 bits y se encuentran mapeados en memoria desde la dirección 0x100AB000. Las siguientes tablas muestran los registros y sus posibles valores:

Offset	Nombre	Descripción
0x00	`reg_tr`	Registro de comandos del tren
0x04	`sensor_te`	Registro de estado de temperatura
0x08	`sensor_pr`	Registro de estado de protuberancia
0x0C	`rad_cl`	Registro de comandos de la radio

Nombre	Comando o Estado	Valor
`reg_tr`	Freno de emergencia	127
`sensor_te`	Temperatura alta	255
`sensor_te`	Temperatura baja	1
`sensor_te`	Temperatura normal	127
`sensor_pr`	Protuberancias	4
`sensor_pr`	Rieles libres de obstáculos	2
`rad_cl`	Mandar reporte de sensores	1

Suponga que bajo este contexto se ejecuta el siguiente programa en RISC-V:

```
.text
 main:
   actionOne:
     li s0, 0x100AB000
     addi s1, s0, 4
     addi s2, s1, 4
     addi s3, s2, 4
     lw s1, 0(s1)
     lw s2, 0(s2)
     li s4, -1
     li s5, 1
     li s6, 127
   actionTwo:
     mv a0, s1
     li a1, 127
     call actionThree
     beq a0, s4, actionFour
     mv a0, s2
     li a1, 2
     call actionThree
     beq a0, s4, actionFour
     sw s5, 0(s3)
     j end
   actionThree:
     beq a0, a1, actionEnd
     li a0, -1
     actionEnd:
       ret
   actionFour:
     sw s6, 0(s0)
     sw s5, 0(s3)
 end:
```

Explique de forma concisa, según el programa y las tablas adjuntas, lo que realizan las acciones `actionOne`, `actionTwo`, `actionThree` y `actionFour`.

APLICACIÓN I/O

Solución: Se otorgan **2 ptos.** por cada acción correctamente descrita:

- `actionOne`: A partir de la dirección base 0x100AB000, carga en los registros `s*` las direcciones de los registros de comando `reg_tr` y `rad_cl` y los valores de los registros de estado `sensor_te` y `sensor_pr`. Adicionalmente, se respaldan los valores -1, 1 y 127. En este caso, -1 representa un estado indeseado; 1 representa el comando de envío de reporte de sensores; y 127 representa el comando de freno de emergencia.
- `actionTwo`: Revisa los estados de cada registro a través de la subrutina `actionThree`. Si se tiene un estado indeseado, ejecuta el código de `actionFour`.
- `actionThree`: Subrutina que revisa si el argumento `a0` es igual a `a1` (estado de registro y estado deseado, respectivamente). Si son iguales, la subrutina termina sin cambios; en otro caso, se modifica el valor de retorno `a0` a -1 para señalar que el estado es no deseado.
- `actionFour`: Acción que ejecuta los comandos de freno de emergencia y envío de reporte de sensores. Solo se ejecuta en caso de estado no deseado.

IIC2343

Home Content IO

ÉXITO

Recuerden comprender de manera profunda los conceptos y las distintas formas de aplicación. Siempre con calma y confianza en sus conocimientos :)

Study Review Confirm

