PARCIAL 1 2018

Pregunta 1- Procesador monociclo – Funcionamiento. Modificar el procesador para poder ejecutar un salto incondicional relativo.

Pregunta 2- Procesador de ejecución segmentada. Principio de funcionamiento. Predicción de salto .¿Por qué? ¿Cómo se implementa?

Pregunta 3- Procesador superescalar. Explicar la diferencia con el procesador de ejecución segmentada. Explique la técnica de ejecución de instrucciones out-of-order.

PARCIAL 1 2018 RECUPERATORIO

Pregunta 1- Arquitectura de una computadora: - Arquitectura básica; - Componentes; - Buses del sistema.

1. Arquitectura básica

La arquitectura de una computadora define la estructura y organización de sus componentes, cómo interactúan entre ellos y cómo ejecutan instrucciones. La arquitectura básica generalmente sigue el modelo de von Neumann, en el cual una computadora está organizada en los siguientes bloques principales:

- **CPU (Unidad Central de Procesamiento):** Responsable de procesar las instrucciones. Se divide en:
  - **Unidad de Control (UC):** Interpreta y ejecuta las instrucciones del programa.
  - **Unidad Aritmético-Lógica (ALU):** Realiza operaciones matemáticas y lógicas.
- **Memoria:** Almacena tanto las instrucciones como los datos utilizados por la CPU.
- **Dispositivos de Entrada/Salida (E/S):** Permiten la comunicación entre la computadora y el mundo exterior (por ejemplo, teclado, pantalla).

La arquitectura moderna puede usar modelos más avanzados como el Harvard, donde las instrucciones y los datos tienen caminos separados, lo que permite paralelismo y mayor eficiencia.

2. Componentes de una computadora

La arquitectura de una computadora describe cómo están organizados sus componentes y cómo interactúan para ejecutar instrucciones. Los principales componentes son:

- CPU (Unidad Central de Procesamiento):

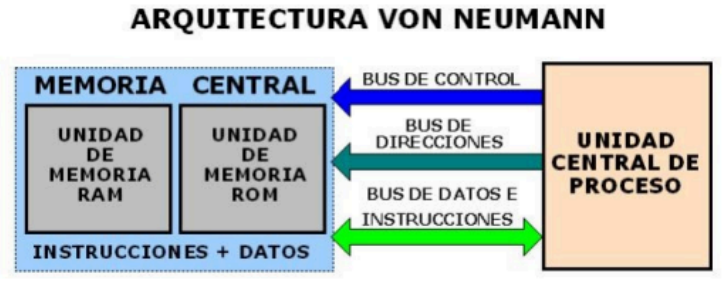
- Compuesta por la **Unidad Aritmético-Lógica (ALU)**, que realiza operaciones matemáticas y lógicas, y la **Unidad de Control**, que interpreta y ejecuta instrucciones.
- **Memoria:** Almacena instrucciones y datos utilizados por el CPU. Existen diferentes tipos de memoria, como la **RAM** (memoria principal) y la **ROM** (memoria de solo lectura).
- **Dispositivos de Entrada/Salida (E/S):** Permiten la interacción entre la computadora y el mundo exterior, como el teclado, ratón, impresoras, etc.
- **Buses del Sistema:** Son los canales de comunicación entre el CPU, la memoria y los dispositivos de E/S. Incluyen el **bus de direcciones**, **bus de datos** y **bus de control**

3. Buses del sistema

Los **buses** son los canales de comunicación a través de los cuales los distintos componentes de la computadora se interconectan. Los buses más importantes son:

- **Bus de datos:** El bus de datos es un canal bidireccional de la unidad de proceso donde se mueven los datos entre los dispositivos, la memoria y la unidad de proceso.
- **Bus de direcciones:** es un canal independiente de la unidad de proceso donde se establece la dirección de memoria del dato en tránsito
- **Bus de control:** es un canal bidireccional de la unidad de proceso que gobierna la operación de los módulos de una computadora.

Cada uno de estos buses trabaja de forma sincronizada para asegurar que los datos, instrucciones y señales de control fluyan eficientemente entre la CPU, la memoria y los dispositivos de E/S.

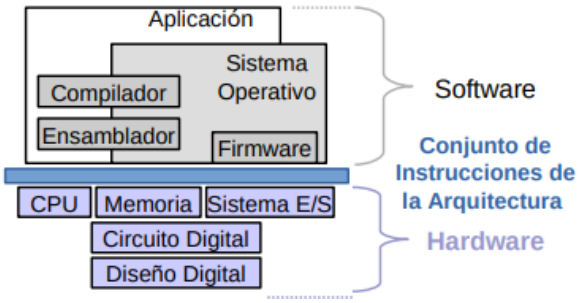


Pregunta 2- ISA ¿Qué es? ¿Cuáles son sus componentes? ¿Cuál es la diferencia entre ISA RISC y un ISA CISC?

ISA: Conjunto de instrucciones de la computadora:

El **conjunto de instrucciones de la arquitectura** (ISA) de una computadora es un **modelo abstracto** que define toda la información necesaria para **programar en lenguaje de máquina** una computadora.

El ISA es **la interfaz entre el hardware y el software.**
La microarquitectura es el conjunto de **técnicas de diseño** utilizadas para implementar el ISA.
Un ISA puede implementarse con diferentes microarquitecturas que **satisfagan diferentes criterios de diseño** (velocidad de operación, consumo energético, costo, etc.). De este modo, el ISA sirve como la interfaz entre el software y el hardware



Sus componentes son:

- Los **modos de operación** soportados (usuario, supervisor, máquina virtual, etc);
- Los **tipos de datos soportados** (bit, nibble, byte, palabra) y su formato (entero, punto flotante);
- El **conjunto de instrucciones** son las operaciones que puede realizar la CPU (movimientos de datos, aritmeticas, lógicas, control de flujo);
- Los **operandos utilizados** por las instrucciones (registros, memoria y puertos) y los modos de direccionamiento, que son los mecanismos para que las instrucciones accedan a los datos;
- La **organización de la memoria** define la ubicación los datos y programas en la memoria, la dirección de reset, y el modelo de entrada/salida utilizado para gestionar los periféricos;
- El **modelo de gestión** de los eventos ajenos al programa (excepciones e interrupciones) y la ubicación de la información relacionado

DIFERENCIA ENTRE UN ISA RISC Y UN ISA CISC

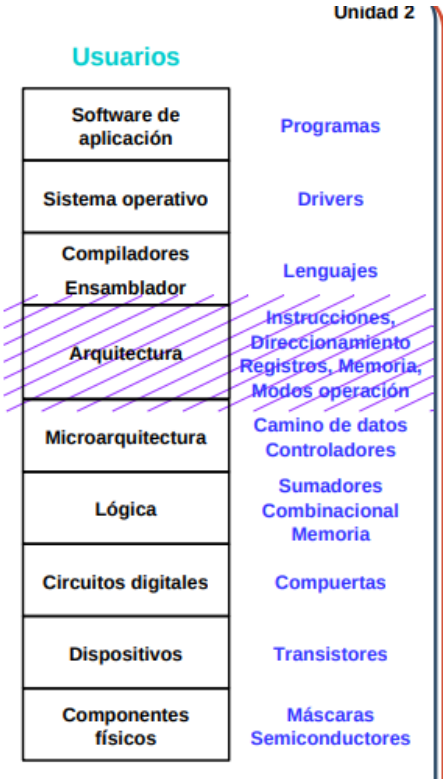
ISA RISC: El **conjunto de instrucciones reducido (RISC)** simplifica la microarquitectura al sólo implementar de manera eficiente las instrucciones más utilizadas, optimizando el desempeño global. Los objetivos son paralelizar las instrucciones y reducir los accesos a memoria.
Las características de los diseños RISC son:

- **Codificación uniforme**; lo que permite una decodificación más rápida.

- **Conjunto de registros homogéneo,** cualquier registro es utilizado en cualquier contexto;
- **Modos de direccionamiento** y tipos de datos soportados sencillos.

ISA CISC: El **conjunto de instrucciones complejas (CISC)** es un modelo de arquitectura de computadores que se caracteriza por ser muy amplio y permitir operaciones complejas entre operandos situados en la memoria o en los registros que hacen que el código sea compacto.
Las características de los diseños CISC son:

- **Codificación no uniforme,** lo que genera códigos compactos pero complica la decodificación de las instrucciones;
- **Ejecución multiciclo de las instrucciones,** lo que dificulta la paralelización de la instrucciones;
- Las instrucciones son implementadas a través de **microprogramación.**



Pregunta 3- Procesador monociclo.

La microarquitectura más simple es la implementación monociclo. La unidad de control se implementa a través de un circuito

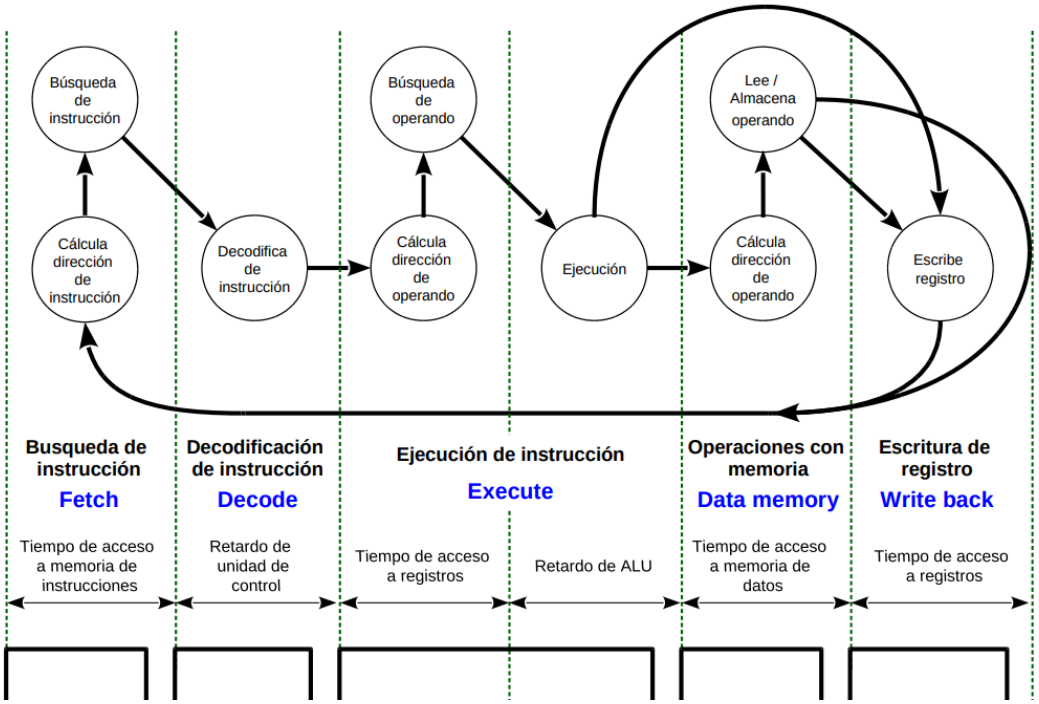
combinacional. Esto simplifica su diseño e implementación circuital (circuito digital muy simple). Pero, todas las instrucciones se ejecutan en un sólo ciclo de reloj. Por lo que el reloj debe tener un periodo igual tiempo de ejecución de la instrucción más lenta.

PARCIAL 2 2018 (parcial 1)

Pregunta 1: Explique cómo funciona y se implementa la gestión de memoria a través de la segmentación de la memoria. ¿Cuáles son las ventajas y desventajas de la segmentación? Explique la diferencia con paginación.

IMPLEMENTACIÓN SEGMENTADA

Para mejorar el desempeño del procesador se **divide la ejecución de la instrucción en etapas** más pequeñas, de modo **sólo utilicen las etapas necesarias** y el periodo del reloj esté determinado por la **etapa más lenta.**



Para comenzar una microarquitectura es una descripción de cómo será implementado el comportamiento descripto por el conjunto de instrucciones(ISA) a través de un sistema digital, esta se representa como diagramas de bloque que describen las interconexiones de los diversos elementos de la microarquitectura, es importante tener un buen diseño de la microarquitectura ya que este afecta directamente al desempeño del sistema, influyendo en el costo, rendimiento, consumo energético, etc. La microarquitectura segmentada es una implementación del conjunto de instrucciones RISC, es un diseño en el cual la ejecución de una

instrucción se divide en varias etapas independientes lo cual requiere registros para almacenar los valores generados por cada una, esto permite aprovechar la diferencia de tiempos de ejecución de las instrucciones, a diferencia de la arquitectura monociclo en la que cada instrucción dura un ciclo de reloj. Otra ventaja de esta implementación es que permite ejecutar mas de una instrucción por ciclo de reloj en el mismo pipeline(solo si estas instrucciones no son dependientes), teóricamente hablando, si el pipeline esta dividido en n etapas, el procesador podrá ejecutar hasta n instrucciones simultáneamente, aun así dividir el pipeline en muchas etapas no aumenta el rendimiento de nuestro procesador, ya que en la práctica aumentar el número de etapas también aumenta tanto el tiempo requerido para llenar y vaciar el pipeline, como la circuitería necesaria para implementarlo

Riesgos de la arquitectura segmentada Los riesgos son situaciones que imposibilitan el comienzo de la ejecución de la próxima instrucción en el próximo ciclo. Se clasifican en, riesgos estructurales(el recurso necesario para ejecutar la instrucción esta ocupado), riesgos de datos(se necesita esperar a que se complete una instrucción previa para disponer del dato necesario en la instrucción actual) y riesgos de control(estos suceden cuando el flujo de programa cambia según el resultado de instrucciones previas, es decir, se realiza una acción de control que altera el flujo de programa).

Paginacion – es una tecnica que divide la memoria fisica y el espacio de direcciones logicas en unidades de tamaño fijo. La MMU asigna unidades de memoria física con unidades de memoria logica de modo que el espacio de direcciones logicas es continuo. La paginacion requiere soporte de hardware en forma de tablas que contienen la dirección física y otros datos (bits de protección de acceso y de estado, entre otros) de cada pagina;

Pregunta 2: Explique la organización y el funcionamiento de la memoria caché. Explique sus parámetros fundamentales y cómo afectan a su desempeño. Explique cómo se realizan las operaciones de lectura y escritura.

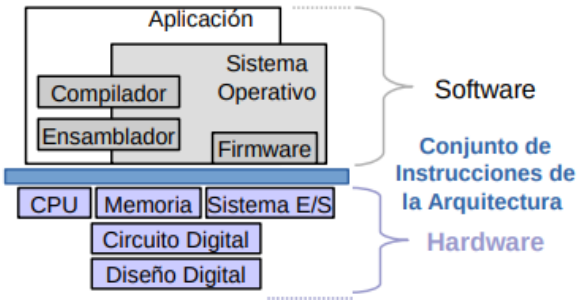
Pregunta 3: Explique cómo funciona la gestión de periféricos por interrupciones. Clasifique y explique los diferentes tipos de interrupciones de acuerdo a su origen. Explique cómo funciona la gestión de interrupciones por consulta. ¿Cuáles son las diferencias y las similitudes con la gestión de periféricos por consulta (polling)?

PARCIAL 1 2019

Pregunta 1) ¿Qué es un ISA? ¿Cuáles son sus componentes? Explíquelos.

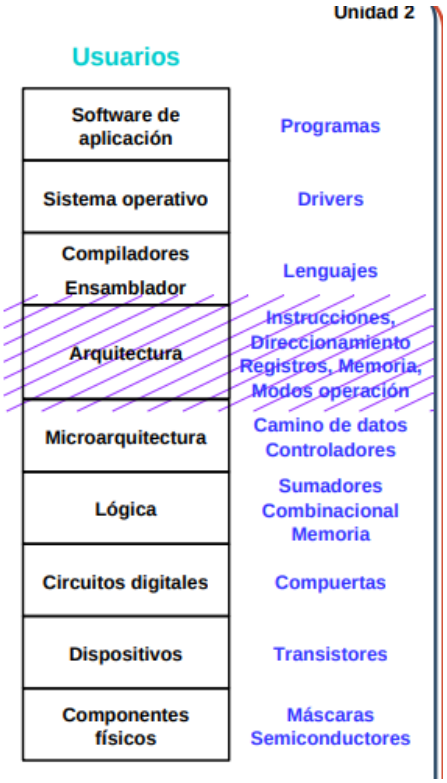
ISA: Conjunto de instrucciones de la computadora:

El **conjunto de instrucciones de la arquitectura** (ISA) de una computadora es un **modelo abstracto** que define toda la información necesaria para **programar en lenguaje de máquina** una computadora.
El ISA es **la interfaz entre el hardware y el software.**
La microarquitectura es el conjunto de **técnicas de diseño** utilizadas para implementar el ISA.
Un ISA puede implementarse con diferentes microarquitecturas que **satisfagan diferentes criterios de diseño** (velocidad de operación, consumo energético, costo, etc.). De este modo, el ISA sirve como la interfaz entre el software y el hardware



Sus componentes son:

- Los **modos de operación** soportados (usuario, supervisor, máquina virtual, etc);
- Los **tipos de datos soportados** (bit, nibble, byte, palabra) y su formato (entero, punto flotante);
- El **conjunto de instrucciones** son las operaciones que puede realizar la CPU (movimientos de datos, aritmeticas, lógicas, control de flujo);
- Los **operandos utilizados** por las instrucciones (registros, memoria y puertos) y los modos de direccionamiento, que son los mecanismos para que las instrucciones accedan a los datos;
- La **organización de la memoria** define la ubicación los datos y programas en la memoria, la dirección de reset, y el modelo de entrada/salida utilizado para gestionar los periféricos;
- El **modelo de gestión** de los eventos ajenos al programa (excepciones e interrupciones) y la ubicación de la información relacionado



Pregunta 2) Arquitectura de una computadora. Componentes. Tipos: ventajas e inconvenientes. Explíquelos.

1. Componentes de una computadora

La arquitectura de una computadora describe cómo están organizados sus componentes y cómo interactúan para ejecutar instrucciones. Los principales componentes son:

- CPU (Unidad Central de Procesamiento):

- Compuesta por la **Unidad Aritmético-Lógica (ALU)**, que realiza operaciones matemáticas y lógicas, y la **Unidad de Control**, que interpreta y ejecuta instrucciones.
- **Memoria:** Almacena instrucciones y datos utilizados por el CPU. Existen diferentes tipos de memoria, como la **RAM** (memoria principal) y la **ROM** (memoria de solo lectura).
- **Dispositivos de Entrada/Salida (E/S):** Permiten la interacción entre la computadora y el mundo exterior, como el teclado, ratón, impresoras, etc.
- **Buses del Sistema:** Son los canales de comunicación entre el CPU, la memoria y los dispositivos de E/S. Incluyen el **bus de direcciones, bus de datos** y **bus de control**

2.Tipos de Arquitectura

Las arquitecturas de computadoras se dividen según la cantidad de instrucciones concurrentes y los flujos de datos disponibles. Los principales tipos son:

- **SISD (Single Instruction Single Data)**: Un solo flujo de instrucciones y datos, es decir, una computadora secuencial.
  - **Ventajas**: Simplicidad de diseño.
  - **Inconvenientes**: Baja eficiencia en tareas que requieren procesamiento paralelo

- **SIMD (Single Instruction Multiple Data)**: Un flujo de instrucciones que opera sobre múltiples datos en paralelo.
  - **Ventajas**: Ideal para operaciones vectoriales, muy eficiente en aplicaciones gráficas o científicas.
  - **Inconvenientes**: Limitado a aplicaciones con datos que pueden ser procesados en paralelo【8†source】.

- **MIMD (Multiple Instruction Multiple Data)**: Varios procesadores ejecutan diferentes instrucciones sobre distintos datos al mismo tiempo.
  - **Ventajas**: Máxima flexibilidad y potencia, ideal para sistemas multitarea y servidores.
  - **Inconvenientes**: Diseño y sincronización más complejos

3. Ventajas e inconvenientes de las arquitecturas

- **Arquitectura Harvard**: Se separan los buses de instrucciones y de datos, lo que permite la ejecución paralela de ambos.
  - **Ventajas**: Mayor velocidad de procesamiento.
  - **Inconvenientes** Más costosa y compleja de implementar.

- **Arquitectura von Neumann** Comparte un solo bus para instrucciones y datos.
  - **Ventajas**: Simplicidad y bajo costo.
  - **Inconvenientes**: Menor velocidad debido al embotellamiento del bus

Conclusión

Cada arquitectura tiene ventajas específicas dependiendo del uso y la eficiencia requerida. Las arquitecturas paralelas como **SIMD** y **MIMD** son más eficientes para tareas especializadas, mientras que las secuenciales como **SISD** y **von Neumann** son más simples y fáciles de implementar.

Pregunta 3) ¿Qué es una dependencia de datos? Tipos. ¿Cómo se solucionan en la microarquitectura del procesador? Diagrama de bloques.

Dependencia de datos: Una dependencia de datos ocurre cuando una instrucción depende de los resultados de otra. Es decir, si una instrucción necesita un dato que aún no ha sido calculado o almacenado por otra instrucción previa. Estas dependencias limitan el paralelismo y afectan la eficiencia de la ejecución en la CPU.

Tipos de dependencia de datos:

Dependencia verdadera (Read-After-Write, RAW): Ocurre cuando una instrucción necesita leer un valor que ha sido escrito por una instrucción anterior. Es la más común y crítica porque se refiere al uso correcto de los datos.

Antidependencia (Write-After-Read, WAR): Se da cuando una instrucción escribe en una ubicación que una instrucción anterior necesita leer.

Dependencia de salida (Write-After-Write, WAW): Sucede cuando dos instrucciones sucesivas escriben en el mismo registro o ubicación de memoria, generando una posible sobrescritura antes de que el primer resultado sea usado .

Soluciones:

Adelanto de datos: La unidad de adelanto de datos estará en la etapa ejecución porque la ALU se encuentra en esta etapa. Por lo tanto, debemos pasar los registro de operandos desde la etapa de decodificación a través del registro de canalización para determinar si se deben adelantar los dato

SOLUCION: Los riesgos generados por una lectura antes de una escritura no pueden ser solucionados con adelanto de datos por que el dato no está disponible.
Estos riesgos se eliminan deteniendo de la etapa (stall) que presenta la dependencia, de modo que el dato pueda ser leido

Hay dos tipos de detenciones:

- **Burbujas:** son instrucciones **NOP** insertadas entre dos instrucciones en el datapath.
    - Evita que las instrucciones anteriores del datapath (adelantadas en el código) progresen por el datapath en un ciclo (las adelanta con las señales de control de escritura);
    - Inserta una NOP poniendo a cero los bits de control en el registro del datapath en las fases adecuadas;

- Hace que las instrucciones posteriores en el datpath (anteriores en el código) progresan normalmente.

- **Vacíado:** todas las instrucciones del datapath son sustituidas con instrucciones NOP.
  - Resetea los bits de control para la instrucciones que deben ser eliminada

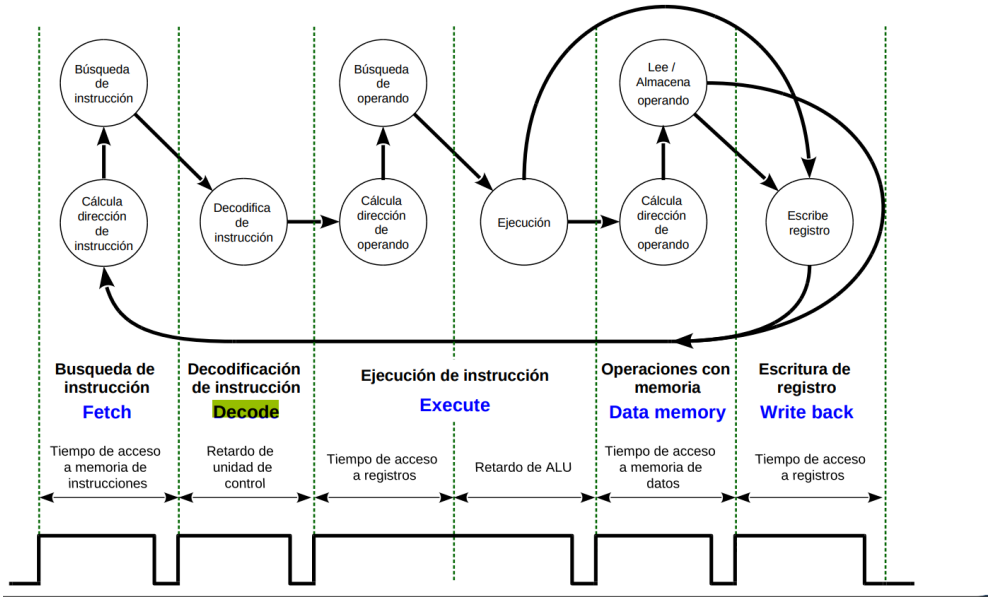
Reordenamiento de instrucciones (Out-of-Order Execution):

- El procesador puede ejecutar las instrucciones en un orden diferente al programado si las dependencias lo permiten. Las instrucciones que no dependen de otras pueden adelantarse en el pipeline, mejorando la utilización de recursos .

Renombrado de registros:

- Para evitar dependencias de tipo WAW y WAR, el procesador puede asignar diferentes nombres o registros físicos a los resultados de instrucciones, eliminando la sobrescritura prematura de datos .

MICROARQUITECTURA SEGMENTADA DIAGRAMA DE BLOQUES:



Pregunta 4) Ejecución fuera de orden. ¿Cómo funciona? Ventajas. Problemas. Algoritmo (el más usado, Tomasulo).

EJECUCION FUERA DE ORDEN

Out-of-Order Issue – Out-of-Order Completion: Las instrucciones son leidas, ejecutadas y los resultados actualizados de acuerdo con la disponibilidad de recursos y las dependencias existentes. Esta politica tiene como objetivo

- Desacoplar la decodificacion de la ejecucion;

- Continuar leyendo y decodificando hasta que la ventana este llena;
- Ejecutar la instruccion cuando la unidad funcional esta disponible.

La ejecución fuera de orden rellena los huecos de tiempo de las instrucciones que están listas para ejecutarse para después reordenar los resultados y aparentar que fueron procesadas de manera normal.
La forma en que las instrucciones son ordenadas en el código original a ejecutar se conoce como orden de programa, El orden en que el procesador maneja las instrucciones es el orden de datos, siendo aquel en que los datos van quedando disponibles

Ventajas

- **Aumento del paralelismo:** Permite ejecutar múltiples instrucciones simultáneamente, aprovechando mejor el hardware disponible.
- **Minimización de esperas:** Al no esperar a que los datos de una instrucción anterior estén listos, el procesador puede seguir ejecutando otras instrucciones que no dependan de esos datos .
- **Eliminación de dependencias:** Utiliza técnicas como el renombrado de registros y estaciones de reserva para evitar las dependencias de datos (WAR y WAW) y mejorar el rendimiento .

Problemas

- **Complejidad:** La implementación de esta técnica requiere hardware sofisticado para gestionar el reordenamiento de instrucciones, la eliminación de dependencias y el manejo de excepciones imprecisas .
- **Mayor consumo de recursos:** Se requiere más ancho de banda para las operaciones con memoria y un mayor número de registros para el renombrado.
- **Excepciones imprecisas:** Puede ser difícil manejar correctamente las excepciones (interrupciones) cuando las instrucciones se ejecutan fuera de su orden original .

Algoritmo de Tomasulo

El **algoritmo de Tomasulo** es uno de los métodos más conocidos para implementar la ejecución fuera de orden. Utiliza varias técnicas clave:

- **Estaciones de reserva**: Las instrucciones esperan en estas estaciones hasta que estén listos todos los operandos.
- **Renombrado de registros**: Se asignan nombres a los registros de forma dinámica para evitar las dependencias de nombres (WAR y WAW).
- **Bus de datos común (CDB)**: Los resultados de las instrucciones se transmiten por este bus a las estaciones de reserva que los necesiten, sin escribir primero en los registros .

El algoritmo asegura que:

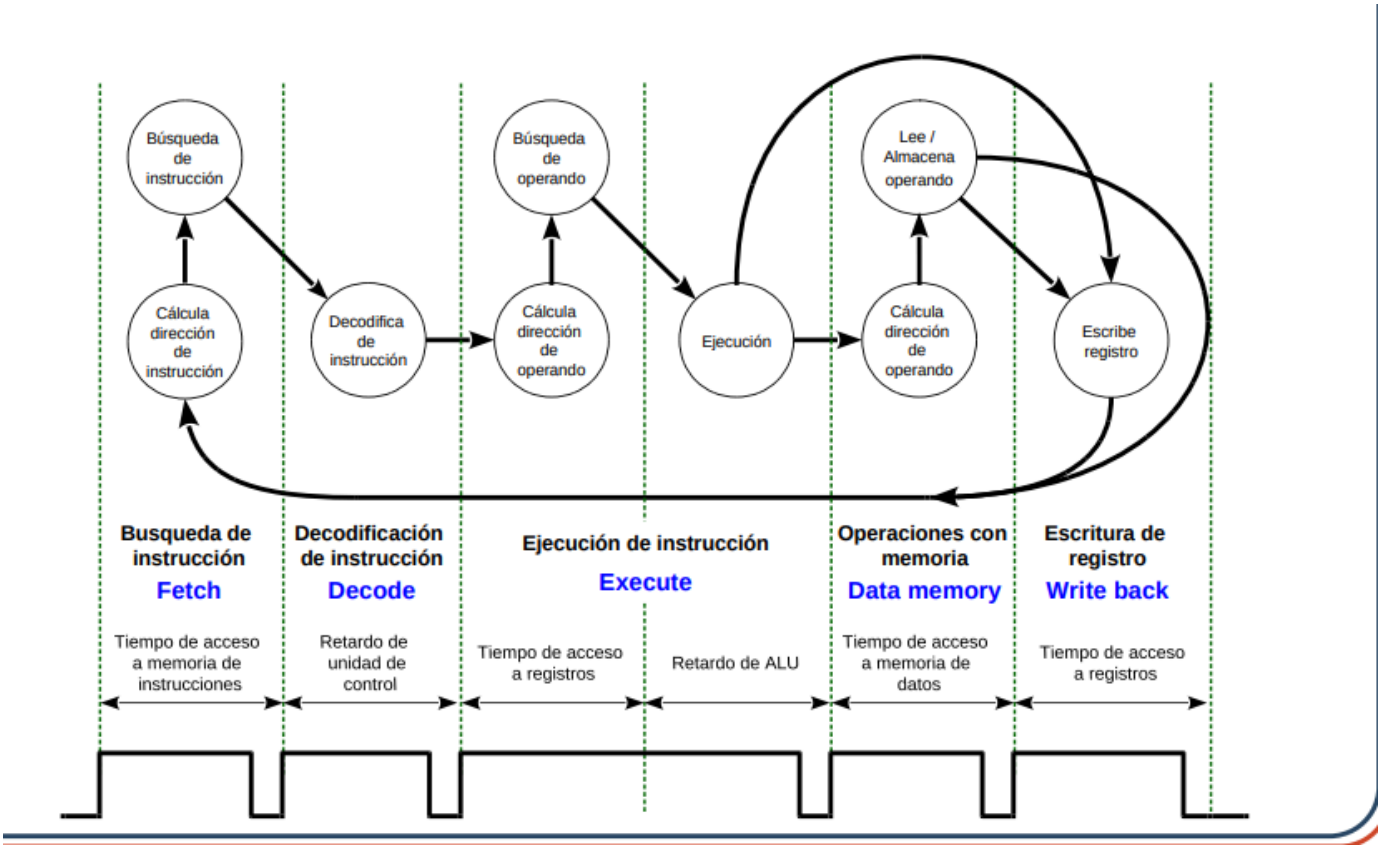
- Las instrucciones se ejecutan en cuanto los operandos estén disponibles, eliminando los riesgos de dependencia de lectura después de escritura (RAW).
- Evita el embotellamiento del acceso a registros gracias al renombrado de estos .

Este enfoque hace que el algoritmo de Tomasulo sea eficaz para aumentar el rendimiento en sistemas superescalares o con múltiples unidades funcionales.

PARCIAL 1 2023

Pregunta 1 (35 puntos) - Explique cómo funciona una microarquitectura segmentada. Explique cuáles son sus ventajas y problemas. ¿Cuál es la arquitectura de computadora óptima para su implementación? Fundamente. ¿Qué son los riesgos de datos? Explique cómo se resuelven. ¿En qué se diferencia de una microarquitectura superescalar?

MICROARQUITECTURA SEGMENTADA:
Para mejorar el desempeño del procesador se divide la ejecución de la instrucción en etapas más pequeñas, de modo sólo utilicen las etapas necesarias y el periodo del reloj esté determinado por la etapa más lenta



Ventajas de la microarquitectura segmentada

- **Mayor rendimiento:** Permite que varias instrucciones se procesen en diferentes etapas de forma simultánea, aumentando el número de instrucciones ejecutadas por ciclo.

- **Mayor uso de los recursos:** Las diferentes etapas del procesador están ocupadas la mayor parte del tiempo, lo que evita que el hardware esté inactivo.
- **Menor tiempo de ejecución:** Las instrucciones se ejecutan en paralelo, reduciendo el tiempo total de ejecución de un conjunto de instrucciones.

Problemas de la microarquitectura segmentada

- **Riesgos de datos y control:** Las instrucciones que dependen de los resultados de instrucciones previas pueden provocar conflictos, causando interrupciones en el flujo de la segmentación.
- **Complejidad en el control:** Necesita mecanismos de control más avanzados para manejar los riesgos y asegurar que las instrucciones se ejecuten en el orden correcto.
- **Costo de hardware:** Requiere más componentes de hardware para implementar registros intermedios entre cada etapa.

RIESGOS DE DATOS:

Los **riesgos** son situaciones que imposibilitan el comienzo de la ejecución de la próxima instrucción en el próximo ciclo.

Se clasifican en:

- **Riesgos estructurales:** Un recurso requerido para la ejecución está ocupado;
- **Riegos de Datos:** Se necesita esperar a que se complete una instrucción previa para disponer del dato necesario en la instrucción actual.
- **Riegos de Control:** La decision sobre una acción de control de flujo de programa depende de una instrucción previa.

Los riesgos de datos ocurren cuando una instrucción se refiere a un resultado que aún no se ha calculado o recuperado. En estas situaciones, las instrucciones muestran la dependencia de los datos que modifican los datos en diferentes etapas del datapath

Hay tres situaciones en las que puede ocurrir un peligro de datos:

- **Lectura despues de una escritura (Read-After-Write):** refiere a una situación en la que una instrucción se refiere a un resultado que aún no se ha calculado o recuperado. Esto puede ocurrir porque aunque una instrucción se ejecuta después de una instrucción anterior que ha sido procesado parcialmente por el dataph (Dependencias de datos).

SOLUCION: Adelanto de datos. La unidad de adelanto de datos estará en la etapa ejecución porque la ALU se encuentra en esta etapa. Por lo tanto, debemos pasar los registro de operandos desde la

etapa de decodificación a través del registro de canalización para determinar si se deben adelantar los datos

- **Escritura despues de una lectura (Write-After-Read):** refiere a una situación en la que una instrucción escribe antes de que otra la lea (Dependencias de nombre).
- **Escritura despues de una escritura (Write-After-Write):** refiere a una situación en la que dos instrucciones sucesivas escriben el mismo registro, presentando una dependencia de nombre (Reutilización de registros)

SOLUCION: Los riesgos generados por una lectura antes de una escritura no pueden ser solucionados con adelanto de datos porque el dato no está disponible.
Estos riesgos se eliminan deteniendo de la etapa (stall) que presenta la dependencia, de modo que el dato pueda ser leido

Hay dos tipos de detenciones:

- **Burbujas:** son instrucciones **NOP** insertadas entre dos instrucciones en el datapath.
    - Evita que las instrucciones anteriores del datapath (adelantadas en el código) progresen por el datapath en un ciclo (las adelanta con las señales de control de escritura);
    - Inserta una NOP poniendo a cero los bits de control en el registro del datapath en las fases adecuadas;
    - Hace que las instrucciones posteriores en el datpath (anteriores en el código) progresan normalmente.
- **Vacíado:** todas las instrucciones del datapath son sustituidas con instrucciones NOP.
    - Resetea los bits de control para la instrucciones que deben ser eliminada

Reordenamiento de instrucciones (Out-of-Order Execution):

- El procesador puede ejecutar las instrucciones en un orden diferente al programado si las dependencias lo permiten. Las instrucciones que no dependen de otras pueden adelantarse en el pipeline, mejorando la utilización de recursos .

Renombrado de registros:

- Para evitar dependencias de tipo WAW y WAR, el procesador puede asignar diferentes nombres o registros físicos a los resultados de instrucciones, eliminando la sobrescritura prematura de datos .

ARQUITECTURA SUPERESCALAR:

Las microarquitecturas y técnicas que permiten mejorar el ILP de los procesadores son:

Segmentación: descomponer las instrucciones en subtareas de manera que diferentes subtareas puedan ejecutarse al mismo tiempo;
Arquitectura superescalar: incrementar los recursos (unidades funcionales y registros) disponibles, de modo que la ejecución de las instrucciones individuales sea realizada en diferentes partes del procesador.

El término **superescalar** es una microarquitectura que mejora la performance de la ejecucion de instrucciones escalares a partir del aumento de recursos (registros y unidades de ejecucion) que operan de manera independiente, implementado paralelismo a nivel de máquina (MLP).

Un procesador superescalar lee múltiples instrucciones al mismo tiempo. Intentando encontrar instrucciones que puedan ejecutarse de manera independiente.
La esencia del enfoque superescalar es la posibilidad de ejecutar instrucciones en paralelo en diferentes datapath

En esta microarquitectura el ILP se incrementa a partir de combinar
Optimización en el compilador: los codigos de programa generados se optimizan para maximizar el ILP teniendo en cuenta las caracteristicas del procesador que ejecuta el codigo;
Técnicas de hardware: los procesadores se diseñan par minimizar los efectos de las dependencias que quedan en el programa

Esta mejora está limitada por:
Dependencia de lectura-escritura: no se puede ejecutar la siguiente instruccion por que depende del resultado anterior;
Dependencia de procedimiento: no se puede ejecutar en paralelo instruciones antes y despues de un salto
Conflictos de recursos: dos o más instruccions requieren el acceso al mismo recurso al mismo tiempo

Pregunta 2 (35 puntos) - Explique cómo funciona una microarquitectura multihilo. Explique cuáles son sus ventajas y problemas. ¿Es lo mismo que una microarquitectura multinúcleo? ¿En qué se diferencian? Fundamente. ¿Cuáles son los dos elementos fundamentales para el desempeño de un procesador multinúcleo? Fundamente.

MICROARQUITECTURA MULTIHILO:
Generalmente las computadoras ejecutan varios programas al mismo tiempo (también conocidos como procesos ó hilos) utilizando multiplexación de tiempo.
Las instrucciones de los diferentes hilos no tienen dependencias por lo que se puede aprovechar el paralelismo de nivel de hilo para mejorar el rendimiento del procesador a través de la ejecución concurrente de los hilos

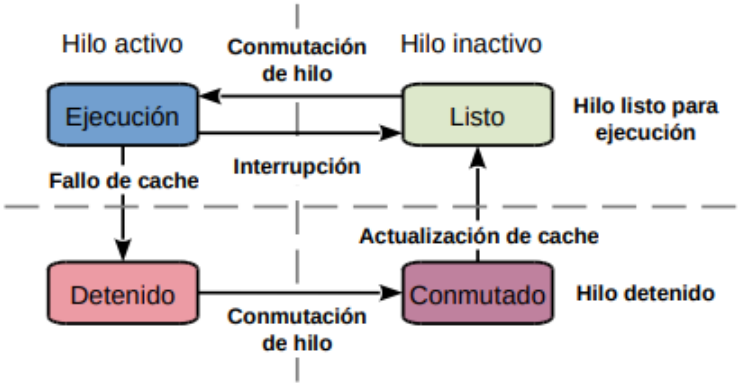
En resumen, una microarquitectura multihilo permite la ejecución concurrente de múltiples hilos en un procesador físico, aprovechando el paralelismo a nivel de instrucción y mejorando el rendimiento del procesador

En esta implica que el procesador ejecuta instrucciones de varios hilos diferentes de modo que el **procesador físico** actuá como **varios procesadores lógicos.** Para ello es necesario proveer al procesador físico de:

- Múltiples contextos de hardware;
- Capacidad de programar hilos de hardware;
- Capacidad de cambio de contexto;
- Ocultar eficazmente las latencia largas

Hay dos opciones:
Multihilo temporal – Los recursos se asignan a cada hilo durante un periodo de tiempo (fijo o variable) dependiendo de los conflictos de recursos y dependencias de datos (multihilo grueso y multihilo entrelazado ó fino);

Multihilo simultáneo – Los recursos se asignan, en cada ciclo de operación, a los hilos que pueden ejecutarse en función de los recursos disponibles y las dependencias de datos.



La implementación más simple del multihilo temporal es el multihilo grueso, se basa en: Ejecutar un hilo hasta que es bloqueado por un evento de latencia prolongada y conmutar a un hilo que esté listo para ejecutarse

VENTAJAS:

- Supera las limitaciones producidas por el bajo ILP de un solo hilo
- Oculta los riesgos de control y otras latencias prolongadas.

DESVENTAJAS:

- Sobrecarga de trabajo a la jerarquía de la memoria;
- Aumento de la complejidad de la unidad de control;
- Aumento de la cantidad de los recursos (caché, predicción de rama, TLB, etc

MICROARQUITECTURA MULTINUCLEO:

Un **procesador multinúcleo** es un sistema de procesamiento **compuesto por varios núcleos** (o CPUs) **independientes.**

Los núcleos se integran en un solo circuito integrado, o múltiples circuitos integrados (dies) empaquetados en un solo chip.

Los núcleos se pueden acoplar entre sí de forma fuerte o débil, compartiendo cache, implementando métodos de comunicación basados en paso de mensajes o memoria compartida

Un **procesador de muchos núcleos** es aquel en el que la cantidad de núcleos es tan grande que las técnicas de procesamiento paralelo ya no son eficientes y **se requiera de una red de comunicaciones** en el chip para operar eficientemente.

Las arquitecturas multinúcleo organizan la memoria de dos maneras:
Memoria compartida – la memoria es única y está compartida por todos los procesadores;

Memoria distribuida - cada procesador tiene su propia memoria local y sus contenidos no están replicados.

VENTAJAS:

- **Rendimiento -** Un procesador de varios núcleos puede realizar más trabajo que un procesador de un solo núcleo por dos motivos: el paralelismo natural (MLP explicito) y velocidades superiores. Como los núcleos están próximos, se logran velocidades de reloj más altas debido a que las señales viajar distancias cortas y son persistentes.
- **Fiabilidad y mutitarea -** los programas se asignan en diferentes núcleos por lo que siempre que hay un defecto se limita a un solo núcleo y pueden tolerar, en mayor medida, fallas.
- **Interacciones de software -** incluso si hay software ejecutado en diferentes núcleos, seguirá interactuando con uno y otro. Un procesador de múltiples núcleos se somete a un proceso conocido como aislamiento espacial y temporal. Estos procesos aseguran que los subprocesos centrales nunca se retrasen.
- **Consumo de energía -** solo la parte del procesador que opera consume la energía. Además, los elementos comunes tienen un consumo independiente de los núcleos que están operando. Por ello, son más eficientes desde el punto de vista energético

DESVENTAJAS:

- **Velocidad de la aplicación -** aunque está diseñado para realizar múltiples tareas, su velocidad no es lo suficientemente significativa. Cada vez que se ejecuta una aplicación, hay que conmutar de un núcleo a otro y es necesario actualizar la caché, lo que compensando su velocidad del núcleo
- **Jitter -** el aumento del número de núcleos produce interferencias en la ejecución que provoca una pérdida de sincronización, lo que lleva a una degradación del rendimiento de los programas debido al conflicto de recursos.
- **Análisis -** realizar varias tareas requiere gestionar los accesos a la memoria, por lo que es difícil determinar y coordinar los tiempos de acceso y las interferencias que se producen.
- **Recursos compartidos -** los recursos (memoria principal, buses del sistema y periféricos) son compartidos por todos los núcleos por lo que cualquier aplicación tenderá a interferir la ejecución de los programas en los diferentes núcleos.
- **Interferencias -** una interferencia es causada por el uso de recursos compartidos, planteando problemas de aislamiento espacial y temporal de cada programa. Esta posibilidad es mayor si hay varios núcleos operando simultáneamente, por lo

que es imposible analizar todas y cada una de las posibles interferencias

ELEMENTOS MÁS IMPORTANTES MULTINUCLEO:

Si en la arquitectura, los nucleos son homogeneos(elementos de procesamiento iguales) o heterogeneos(elemetnos de procesamiento diferentes)
Los sistemas de comunicación entre los núcleos, que determina cuánto tiempo deben esperar los núcleos antes de poder acceder a los datos.
Si la memoria es compartida o distribuida también

Escalabilidad del software: El software debe estar diseñado para aprovechar varios núcleos. Esto implica que las tareas y procesos deben estar bien paralelizados para distribuirse entre los diferentes núcleos del procesador
Coherencia y comunicación entre cachés: En un procesador multinúcleo, cada núcleo suele tener su propia caché, pero también debe comunicarse con otras cachés de los núcleos vecinos para garantizar la coherencia de los datos (cuando varios núcleos acceden a los mismos datos).

PARCIAL 1 2023 Recuperatorio

Pregunta 1 (35 puntos) - ¿Qué son los riesgos de control? Explique cuáles son las posibles soluciones. Explique cómo funciona un predictor fijo. Explique cuáles son sus ventajas y problemas. ¿Cuáles son los lugares donde se pueden implementar los predictores? ¿En qué se diferencia una predicción estática de una dinámica?

RIESGOS DE CONTROL:
Cada instrucción depende de un conjunto de saltos y esta dependencia, orden de ejecución, debe preservarse para preservar el orden del programa;
Las dependencias de control **pueden violarse si no afectan** los resultados del programa.
LO IMPORTANTE es preservar el comportamiento del flujo de datos.

Dependencias de control y excepciones
El comportamiento de excepciones debe preservarse. Cualquier cambio en el orden de ejecución no debe cambiar como las excepciones son atendidas en la ejecución.

Dependencias de control y flujo de datos
Se debe mantener el flujo de datos entre instrucciones productoras y consumidoras de datos.

Cuando se detecta una instrucción de salto se debe determinar el tipo de salto a ejecutar, la informacion necesaria para ejecutarlo y la direccion del salto.

Cuando se detecta un **salto incondicional** debe determinarse la fuente de informacion necesaria para calcular la direccion final del salto, ya que el salto se toma siempre

- La direccion final se encuentra en la instrucción misma (salto absoluto);
- La direccion final se obtiene componiendo informacion disponibles en algun registro y parte de la instrucción (salto relativo)

Cuando se detecta un **salto condicional sin resolver** deben determinarse i) el camino del salto y ii) la fuente de informacion necesaria para calcular la direccion final del salto. El camino a seguir depende de la estructura del lazo considerado.

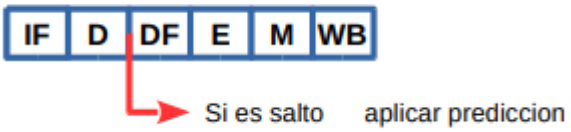
- Si se **predice como tomado** se calcula la direccion de destino y **la ejecucion continua de forma especulativa** a partir de dicha direccion;
- Si **se predice como no tomado** la e**jecucion continua de forma especulativa**

Cuando se **resuelve la condicion del salto**

- Si la prediccion **fue correcta** se confirma la ejecucion y **continua normalmente;**
- Si la prediccion **no fue correcta se descartan** las instrucciones ejecutadas especulativamente (flushing) y se reanuda la ejecucion por el camino correcto.

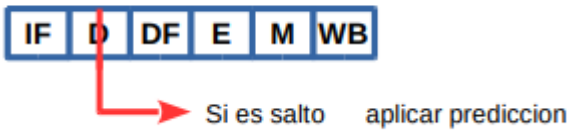
La fuente de informacion necesaria para calcular la direccion final del salto se puede encontrar en
- La instrucción misma; o
- Se obtiene componiendo informacion disponibles en algun registro y parte de la instrucción.

Problema: no se conoce el tipo de instrucción hasta que finaliza la etapa de decodificacion (Decode), lo que introduce un retardo en la deteccion del salto.

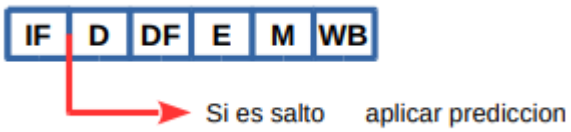


HAY TRES ALTERNATIVAS:

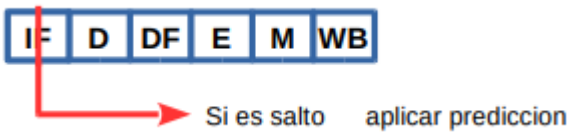
Deteccion en paralelo con la decodificacion: utiliza un decodificador de saltos dedicado para detectar las instrucciones de salto antes del final de la etapa de decodificacion



Deteccion despues de la busqueda: detecta las instrucciones de salto en el buffer de instrucciones antes de que sean decodificadas;



Deteccion durante la busqueda: detecta las instrucciones de salto al tiempo que se leen la memoria de instrucciones



RIESGOS DE CONTROL - PREDICCION FIJA

La predicción estática es la técnica más sencilla que no utiliza información sobre el historial dinámico de ejecución del código, sino que predice el resultado basándose sólo en la instrucción.
Hay dos posibilidades:

- **Se ejecuta**
  - Cada instrucción de salto se ejecuta;
  - Tiene un número mayor de aciertos;
  - El hardware necesario para implementarlo es más complejo.
- **No se ejecuta**
  - Cada instrucción de salto no se ejecuta;
  - Tiene un número menor de aciertos;
  - El hardware necesario para implementarlo es más sencillo.

La predicción estática se basa en la dirección del salto a ejecutar;

Se ejecuta siempre si la dirección destino es menor que la que se encuentra la instrucción de salto (salto hacia atrás). Este tipo de salto generalmente se utiliza en bucles

No se ejecuta siempre si la dirección destino es mayor que la que se encuentra la instrucción de salto (salto hacia adelante). Este tipo de salto generalmente se utilizan en instrucciones IF – THEN - ELSE

Ventajas:

- Implementación más simple.
- Bajo costo en hardware.

Problemas:

- Baja precisión, especialmente en estructuras complejas.
- No se adapta al comportamiento del programa en tiempo de ejecución.

RIESGOS DE CONTROL - PREDICCION DINAMICA

Un predictor dinámico trabaja en tiempo de ejecución intentando aprender el comportamiento del programa para predecir, **con la máxima tasa de aciertos**, si un salto será o no tomado.

La información que se intercambia entre las etapas de búsqueda de instrucción, decodificación y ejecución se almacena junto con el resultado en l**a tabla de historia de saltos** (Branch History Table - BHT -) que está direccionada por los bits más bajos del contador de programa

La BHT es una memoria de doble puerto que:

- Lee la dirección de destino asociada a una dirección en la etapa de búsqueda;
- Actualiza la dirección de destino asociada a una dirección en la etapa de decodificación;
- Escribe la dirección de destino asociado a una dirección que no está en la tabla

Ventajas:

- Alta tasa de aciertos en predicción.
- Adapta las predicciones a medida que el programa se ejecuta, mejorando el rendimiento en bucles y condiciones repetitivas.

Problemas:

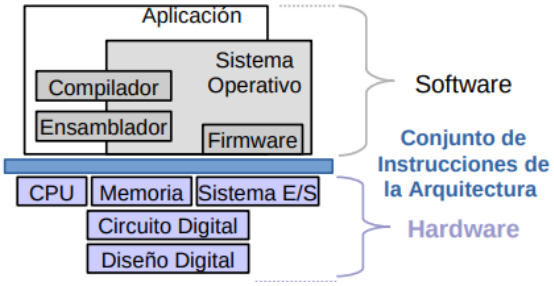
- Más complejidad en el hardware.
- Mayor costo en términos de consumo de energía y tiempo de acceso a la memoria.

Lugares donde se pueden implementar los predictores:

1. En la etapa de búsqueda de instrucciones: Se detecta el salto al leer la memoria de instrucciones.
2. En la etapa de decodificación: Un decodificador dedicado predice el salto antes de finalizar esta etapa.
3. En el buffer de instrucciones: El salto se detecta antes de decodificar las instrucciones.

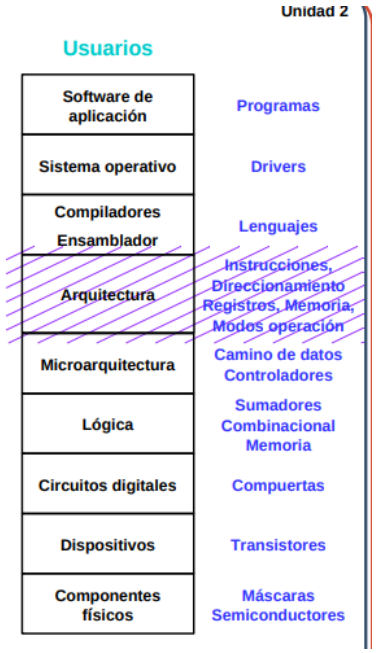
Pregunta 2 (20 puntos) - Explique qué es y qué elementos constituyen el conjunto de instrucciones de la arquitectura (ISA). Describa los elementos más importantes del mismo. De acuerdo con la cantidad de operandos, ¿cómo se clasifican las instrucciones? ¿Cuáles son las ventajas de las instrucciones con tres operandos y registros de propósitos generales (arquitectura Register-Register) comparadas con las dos operandos en registro y uno en memoria (arquitectura Register-Memory)?

QUÉ ES Y QUÉ ELEMENTOS CONSTITUYEN EL ISA: El conjunto de instrucciones de la arquitectura (ISA) de una computadora es un modelo abstracto que define toda la información necesaria para programar en lenguaje de máquina una computadora.
El ISA es la interfaz entre el hardware y el software. La microarquitectura es el conjunto de técnicas de diseño utilizadas para implementar el ISA.



QUÉ ELEMENTOS LA COMPONEN:
El **conjunto de instrucciones de la arquitectura** está compuesto por:

- Los **modos de operación** soportados (usuario, supervisor, máquina virtual, etc);
- Los **tipos de datos soportados** (bit, nibble, byte, palabra) y su formato (entero, punto flotante);
- El **conjunto de instrucciones** son las operaciones que puede realizar la CPU (movimientos de datos, aritmeticas, lógicas, control de flujo);
- Los **operandos** utilizados por las instrucciones (registros, memoria y puertos) y los **modos de direccionamiento**, que son los mecanismos para que las instrucciones accedan a los datos;
- La **organización de la memoria** define la **ubicación** los datos y programas en la memoria, la **dirección de reset**, y el **modelo de entrada/salida** utilizado para gestionar los periféricos;
- El **modelo de gestión de los eventos** ajenos al programa (excepciones e interrupciones) y la ubicación de la información relacionada.



DE ACUERDO LA CANTIDAD DE OPERANDOS EL ISA SE CLASIFICA EN:

ISA CON OPERANDOS IMPLICITOS: es un modelo computacional en el cual la memoria de la computadora toma la forma de pilas. Las instrucciones operan con los datos en el tope de las pilas y los reemplazan por el resultado
Ventajas: Buena densidad de código; Requerimientos de hardware bajos; Compiladores sencillos;
Desventajas: La pila es cuello de botella de ejecución; Dificultad para paralelizar ejecución; Necesidad de instrucciones adicionales para cargar datos; Compiladores óptimos son difíciles de desarrollar.

ISA CON UN OPERANDO EXPLÍCITO: es un modelo computacional en cual el acumulador se usa implícitamente para procesar todas las instrucciones y almacenar los resultados.
El formato de instrucción de este modelo utiliza **un campo de dirección** por que uno de los operandos y el destino están implicitos (acumulador).
Ventajas: Instrucciones cortas y muy flexibles; Programa pequeños; Requerimientos de hardware muy bajos; Fácil de diseñar y comprender;
Desventajas: El tráfico de memoria es elevado; El acumulador es un cuello de botella durante la ejecución; Difícil de paralelizar y segmentar la ejecución de instrucciones

ISA CON REGISTROS DE PROPOSITOS GENERALES: es un modelo computacional en cual todos los registros pueden ser utilizados para procesar todas las instrucciones y almacenar los resultados. El formato de instrucción de este modelo utiliza **tres campos de direcciones,** dos para los operandos y uno donde se almacena el resultado.
TRES VARIANTES:
Tres operandos en la memoria de la computadora
Dos operandos en los registros y otro en la memoria de la computadora
Tres operandos en registros con instrucciones load y store para mover datos entre memoria y registros

Cuáles son las ventajas de las instrucciones con tres operandos y registros de propósitos generales (arquitectura Register-Register) comparadas con las dos operandos en registro y uno en memoria (arquitectura Register-Memory)

ARQUITECTURA MEMORIA-MEMORIA:
Ventajas: Requiere pocas instructiones; Compiladores fáciles de desarrollar;
Desventajas: Tráfico de memoria muy alto; Número variable de ciclos de reloj.

ARQUITECTURA MEMORIA-REGISTRO:
Ventajas: Formato de instrucción fácil de codificar; Buena densidad de código;
Desventajas: Los operandos no son equivalentes; Número variable de ciclos de reloj; Número limitado de registros.

ARQUITECTURA REGISTER-REGISTER:
Ventajas: Codificación de instrucciones simple y longitud fija; Número fijo de ciclos de reloj; Fácil de paralelizar y segmentar;
Desventajas: Mayor número de instrucciones; Depende de la calidad del compilador

PARCIAL 2 2019

Pregunta 1: (25 puntos) - Explique el funcionamiento y organizacion de la memoria caché. Detalle cómo se realizan las operaciones de lectura y escritura

Pregunta 3: (25 puntos) Explique el funcionamiento del controlador de video en modo gráfico sin aceleración.

Pregunta 4: (25 puntos) Explique el funcionamiento del acceso directo a memoria (DMA).

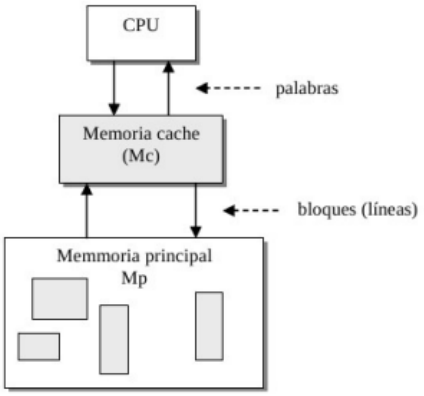
Ejercicio 4: (25 puntos) Explique el funcionamiento del teclado.

PARCIAL 2 2023

Pregunta 1 (30 puntos) - Explique cómo funciona la memoria caché de una computadora y para qué se utiliza. ¿Cuántos niveles de caché hay en una computadora? ¿Están organizados de la misma manera? Explique. Explique los mecanismos de mapeo de datos en una caché. ¿Cuáles son sus ventajas y desventajas? Explique cuáles son las fuentes de error en una caché.

La memoria caché es el nivel más alto en la jerarquia de la memoria, utilizada para reducir el costo promedio, en tiempo y/o energía, para acceder a la información de la memoria principal. Una caché es una memoria más pequeña y más rápida, cercana al núcleo del procesador, que almacena copias de los datos de las ubicaciones de la memoria principal de uso frecuente

La memoria caché se utiliza para **acelerar el acceso del procesador a los datos e instrucciones** que se encuentran en la memoria principal (RAM). Su objetivo principal es reducir la latencia o el tiempo de espera que el procesador experimenta cuando necesita acceder a la memoria, permitiendo que los datos que se usan con mayor frecuencia estén disponibles de inmediato.



El rendimiento de la caché aumenta el desempeño global de la computador al reducir la brecha de velocidad entre el procesador y la memoria a partir del aprovechamiento de la localidad espacial y temporal de la información.

Los procesadores modernos tienen múltiples niveles de caché:

- El **primer nivel (L1)** se ubica **cerca del núcleo** y está **dividida** en un **bloque de datos** y otro para **instrucciones**.
- El **segundo nivel (L2)** se ubica **dentro del procesador, no se divide** y **actúa como repositorio común** para el primer nivel de caché. Cada núcleo de un procesador multinúcleo tiene un caché de este tipo que no se comparte con otros núcleos.

- El **tercer nivel** (**L3**) se ubica **dentro del procesador, no se divide** y se **comparte entre núcleos.**
- El **cuarto nivel (L4)** se ubica **fuera del proceasor** y en la memoria principal, en un chip separado.

Cada nivel de caché está organizado de manera diferente para balancear velocidad y capacidad, con niveles más cercanos al núcleo siendo más rápidos pero de menor tamaño

La caché puede ser **física** (trabaja con direcciones físicas) o **virtual** (trabaja con direcciones virtuales).

Mapeo de la memoria

Determina cómo se asignan los bloques de la memoria principal a las líneas de caché.
Existen tres tipos de mapeo:

Mapeo directo - especifica una sola línea de caché para cada bloque de memoria;
Mapeo asociativo - especifica un conjunto de líneas de caché para cada bloque de memoria;
Mapeo asociativo completo - todos los bloques de la cache estan en un solo conjunto

Operación - Mapeo directo

La memoria caché por mapeo directo contienen un bloque de datos por cada conjunto
Solo necesita mantener tres datos en memoria
Datos de caché – la información actual
Etiqueta de caché – el bloque de datos de memoria;
Estado del bloque – indica si la línea de caché contiene un bloque válido.

El problema de este enfoque es que hay más bloques de memoria que líneas de caché. Una de las ventajas de un caché mapeado directo es que permite una especulación simple y rápida.

Las memorias caches por mapeo directo tienen problemas con la localidad temporal y las fallas obligadas, las cuales se producen cuando el sistema empieza a funcionar y no hay información del mismo

Operación - Mapeo asociativo

Contiene una entrada por cada bloque de datos en cada conjunto, permitiendo tener **localidad espacial.**
Una caché asociativa prueba todas las vías simultáneamente. Al aumentar el número de asociatividad va bajando el número de fallos.

FUNTES DE ERROR:
Operación - desempeño
Un error de la caché es un intento fallido de leer o escribir los datos en caché.
Los errores se clasifican en:

- **Compulsory:** En el primer acceso a un bloque, este debe ser llevado a caché. (**fallo de arranque en frío**)
- **Capacity:** ocurre porque los bloques se descartan de la caché debido a que no puede contener todos los bloques necesarios para la ejecución del programa.
- **Conflict:** Varios bloques se asignan al mismo conjunto de bloques. (**fallo de colisión**)

Cambiar los parámetros de la caché puede tener uno o más tipos de fallas.

Pregunta 2 (20 puntos) - Explique cómo funciona la gestión de periféricos por interrupciones. Explique cuáles son sus ventajas y problemas. ¿Cómo se gestionan las interrupciones anidadas? Explique el mecanismo utilizado. ¿Es lo mismo una interrupción que una excepción? Explique qué se diferencia. ¿Para qué se usan las excepciones?

8.2 Gestión de perifericos

Hay tres mecanismos para gestionar las transferencias de datos:

- Gestion programada;
- Gestion por interrupciones; y
- Gestion por acceso directo.

10.2 Gestión por interrupciones

Una interrupción es una señal (hardware o software) que indica al procesador que un evento de alta prioridad requiere la interrupción del código que el procesador está ejecutando. El procesador responde suspendiendo sus actividades actuales, guardando su estado y ejecutando una rutina de servicio de la interrupcion. Una vez que finaliza, el procesador reanuda las actividades normales.

Las interrupciones:

- **Permiten responder a eventos sincronos y asincronos;**
- **Especifican las rutinas específica de atención de cada evento;**
- **Provee una forma simple de comunicación entre procesos**

Ventajas: el procesador puede ejecutar otro código mientras espera un evento

Inconvenientes: dificulta el cálculo de los tiempos de ejecución y latencia, es complejo gestionar interrupciones anidadas

Las etapas de la ejecución de una interrupción son:
1.Completar la ejecución de la instrucción en curso (si se puede);
2.Determinar la fuente de la interrupción;
3.Analizar si se atiende, o no, la interrupción;
4.Si se atiende, salvar la información del programa en ejecución;
5.Determinar la dirección de la rutina de atencion y ejecutarla; y
6.Una vez finalizada la ejecución de la rutina de interrupción reasumir la ejecución del programa

Las interrupciones se pueden clasificar según su **mecanismo de activación**

Excepciones - se producen por **errores durante la ejecucion de una instrucción o fallas en el hardware**, son **síncronas con el funcionamiento del procesador**

Interrupciones por hardware (IRQ) - son producidas por **eventos externos al programa, son asíncronas a la ejecución del programa** (se pueden producir en cualquier momento independientemente de lo que esté haciendo el procesador). Las causas que las producen **son externas al procesador y están relacionadas con los dispositivos de entrada / salida.**

Interrupciones por software (TRAP) - son producidas por el **programa en ejecución.** Para generarlas existen distintas instrucciones en el **código de máquina** que permiten al programador producir una interrupción

Gestion de Prioridad
El sistema necesita un mecanismo para priorizar las interrupciones y tratar primero las más urgentes. Para ello, existen varias alternativas:

- **Interrupciones simultáneas:** hay dos alternativas para gestionar varias interrupciones activas:
    - – Un dispositivo específico que combina varias fuentes de interrupción en una o más líneas, al tiempo que permite asignar los niveles de prioridad las salidas de interrupción.
    - Un método distribuido que se implementa en la práctica a traves de dos técnicas;
        - **i) Consulta –** es un metodo de gestion en el cual una única rutina debe determinar la fuente de la interrupción al verificar todas las posibles fuentes de interrupción

        - **ii) Conexión en cadena (Daisy Chain)** – es un metodo de gestion en el que los dispositivos están conectados en forma serial.
- **Interrupciones anidadas -** en un sistema de interrupción anidado se permite una interrupción en cualquier momento y en cualquier lugar, incluso cuando la rutina de atencion de un interrupcion se está ejecutando. Solo la rutina de atencion de la interrupcion (ISR) de mayor prioridad será ejecutada inmediatamente. Luego, el segundo ISR de prioridad más alta se ejecutará después de que se complete el más alto
- **Inhibición de interrupciones -** las interrupciones pueden ser desactivadas por el programador en momentos en que el procesador está ejecutando rutinas críticas que no se puede interrumpir. Si llega una interrupción, el procesador la ignora o atiende dependiendo del estado del bit asociado del registro de máscara.

10.3 Excepciones

Las excepciones son eventos que **rompen la secuencia** de ejecución del programa para **atender errores que ocurren durante la ejecucion de una instrucción o fallas** en el hardware de la computadora.
Son eventos **síncronos** con el funcionamiento del procesador y se clasifican en precisas (cuando se conoce su causa) o imprecisas (cuando no se conoce sus causa), en cuyo caso la causa se determina analizando la traza del programa en curso.
Las causas de excepciones son

- Realizar tareas de depuracion del programa (breakpoint, trace, etc.);
- Realizar operaciones no permitidas (división por cero, codigos desconocidos);
- Accesos ilegales a la memoria (desbordamiento de pila, acceso a posiciones de memoria no permitidas, operaciones no permitidas, violacion de privilegios); y
- Falta de datos para ejecutar instrucciones (tabla de segmentos incompleta, segmento no presente en la memoria)

10.4 Interrupciones

Una interrupción es un mecanismo que rompe la secuencia de ejecución del programa para atender un evento no previsto, asincronico y externo a la ejecución del programa.
Son generadas por los dispositivos de la computadora

Las etapas de la ejecución de una interrupción son:
1.Completar la ejecución de la instrucción en curso;
2.Determinar la fuente de la interrupción;
3.Analizar si se atiende, o no, la interrupción;

4.Si se atiende, **salvar la información del programa** en ejecución;
5.Determinar la dirección de la rutina de atencion y ejecutarla; y
6.Una vez finalizada la ejecución de la rutina de interrupción **reasumir la ejecución** del programa.

Pregunta 3 (30 puntos) - Explique qué es y cómo funciona un puerto serie. Explique cuáles son sus ventajas y desventajas. Explique la principal diferencia entre un puerto serie síncrono y asíncrono. ¿Cuáles son los mecanismos de un puerto asíncrono para lograr la sincronización entre el transmisor y el receptor?

12.3 Puertos serie

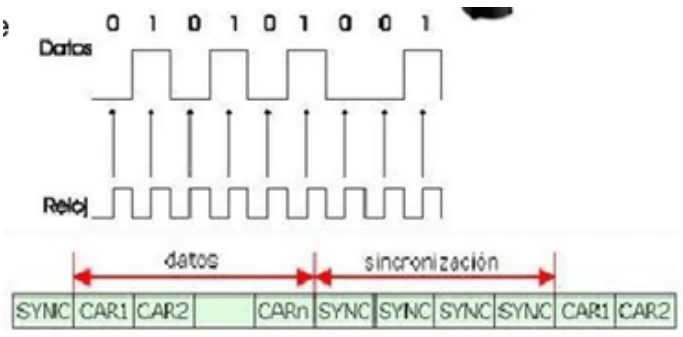
Un puerto serie o puerto en serie es una interfaz de comunicaciones de datos digitales donde la información es transmitida un solo bit a la vez; en contraste con el puerto paralelo que envía varios bits simultáneamente.

12.3.1 Puerto Serie Síncrono

En una transferencia serial síncrona se intercambian una o varias señales de control, junto con los paquetes de informacion, entre emisor y receptor. Las señales de control determinan cuando hay un bit de datos válido en la línea de transmisión

- Puede existir una señal de reloj que controle la sincronización; y/o
- La sincronización se realiza a través de un protocolo solicitud – reconocimiento o estructuras de información predeterminadas

La transmisión es controlada por el emisor. La información se transmite entre dos grupos de datos denominados delimitadores.
Ventajas: Posee un alto rendimiento; Operan a altas velocidades; El flujo de datos es regular.
Desventajas: Los equipamientos son más complejos y de costosos



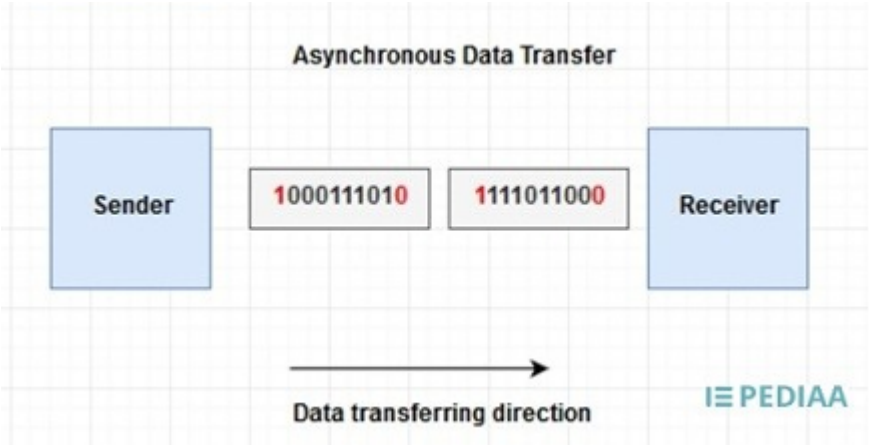
12.3.2 Puerto Serie Asíncrono

Una transferencia asíncrona ocurre cuando no hay relación temporal entre los relojes del transmisor y el receptor. La sincronización se realiza en cada dato transmitido y a través de estructuras de datos.

El ritmo de presentación de la información en el receptor no tiene que coincidir con el del transmisor. No es necesario garantizar un ancho de banda determinado, suministrando el que se disponga.

Por ello es necesario que

- Los datos contengan la información de temporización;
- El receptor muestre la línea a intervalos regulares y mayores que un pulso para detectar la llegada de datos.



TRANSFERENCIA SINCRONA:

- Permite mayores velocidades de transmisión;
- Permite que el receptor pueda interactuar con emisores de frecuencia de reloj variada;
- Permite interconectar una menor variabilidad de dispositivos, ya que emisor y receptor deben cumplir con el mismo protocolo de transmisión.

TRANSFERENCIA ASINCRONA:

- Es más lenta debido a la información adicional y los de captura;
- En caso de errores se recupera la transmisión; Bajo rendimiento de transmisión;
- Equipamiento económico y robusto;
- Aplicable a transmisiones con flujo de datos irregular;
- Permite interconectar una mayor variabilidad de dispositivos

PARA IMPLEMENTAR UN SISTEMA ASINCRONICO:

Se deben acordar los parámetros de señalización

- Tipo de operación (full o half-duplex) a realizar;
- Tamaño del dato (cantidad de bits por carácter) y el orden en que van a ser enviados;
- La velocidad de transmisión;
- El código de detección-corrección de errores utilizado; y
- La estructura de información de sincronización.

Ademas podemos distinguir el puerto serie del paralelo:

El puerto paralelo:

Es teóricamente más rápido
Los datos no necesitan ser preprocesados
Es menos flexible
Es más costoso
Tiene predisposición a sufrir errores en distancias largas

El puerto serie:

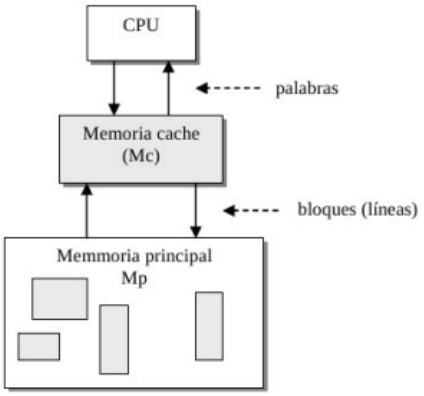
Es teóricamente más lento
Los datos deben ser preprocesados (serializados/deserializados)
Es más flexible
Es mucho menos costoso por su menor número de líneas
Tiene una menor probabilidadsición a sufrir errores, incluso en distancias largas

PARCIAL 2 2023 Recuperatorio

Pregunta 1 (30 puntos) - Explique cómo funciona la memoria caché de una computadora y para qué se utiliza. ¿Cuántos niveles de caché hay en una computadora? ¿Están organizados de la misma manera? Explique. Explique los mecanismos de mapeo de datos en una caché. ¿Cuáles son sus ventajas y desventajas? Explique cuáles son las fuentes de error en una caché.

La memoria caché es el nivel más alto en la jerarquia de la memoria, utilizada para reducir el costo promedio, en tiempo y/o energía, para acceder a la información de la memoria principal. Una caché es una memoria más pequeña y más rápida, cercana al núcleo del procesador, que almacena copias de los datos de las ubicaciones de la memoria principal de uso frecuente

La memoria caché se utiliza para **acelerar el acceso del procesador a los datos e instrucciones** que se encuentran en la memoria principal (RAM). Su objetivo principal es reducir la latencia o el tiempo de espera que el procesador experimenta cuando necesita acceder a la memoria, permitiendo que los datos que se usan con mayor frecuencia estén disponibles de inmediato.



El rendimiento de la caché aumenta el desempeño global de la computador al reducir la brecha de velocidad entre el procesador y la memoria a partir del aprovechamiento de la localidad espacial y temporal de la información.

Los procesadores modernos tienen múltiples niveles de caché:

- El **primer nivel (L1)** se ubica **cerca del núcleo** y está **dividida** en un **bloque de datos** y otro para **instrucciones**.
- El **segundo nivel (L2)** se ubica **dentro del procesador, no se divide** y **actúa como repositorio común** para el primer nivel de caché. Cada núcleo de un procesador multinúcleo tiene un caché de este tipo que no se comparte con otros núcleos.

- El **tercer nivel** (**L3**) se ubica **dentro del procesador, no se divide** y se **comparte entre núcleos.**
- El **cuarto nivel (L4)** se ubica **fuera del proceasor** y en la memoria principal, en chip separado.

Cada nivel de caché está organizado de manera diferente para balancear velocidad y capacidad, con niveles más cercanos al núcleo siendo más rápidos pero de menor tamaño

La caché puede ser **física** (trabaja con direcciones físicas) o **virtual** (trabaja con direcciones virtuales).

Mapeo de la memoria

Determina cómo se asignan los bloques de la memoria principal a las líneas de caché.
Existen tres tipos de mapeo:

Mapeo directo - especifica una sola línea de caché para cada bloque de memoria;
Mapeo asociativo - especifica un conjunto de líneas de caché para cada bloque de memoria;
Mapeo asociativo completo - todos los bloques de la cache estan en un solo conjunto

Operación - Mapeo directo

La memoria caché por mapeo directo contienen un bloque de datos por cada conjunto
Solo necesita mantener tres datos en memoria
Datos de caché – la información actual
Etiqueta de caché – el bloque de datos de memoria;
Estado del bloque – indica si la línea de caché contiene un bloque válido.

El problema de este enfoque es que hay más bloques de memoria que líneas de cache. Una de las ventajas de un caché mapeado directo es que permite una especulación simple y rápida.

Las memorias caches por mapeo directo tienen problemas con la localidad temporal y las fallas obligadas, las cuales se producen cuando el sistema empieza a funcionar y no hay información del mismo

Operación - Mapeo asociativo

Contiene una entrada por cada bloque de datos en cada conjunto, permitiendo tener **localidad espacial.**
Una caché asociativa prueba todas las vías simultáneamente. Al aumentar el número de asociatividad va bajando el número de fallos.

FUNTES DE ERROR:
Operación - desempeño
Un error de la caché es un intento fallido de leer o escribir los datos en caché.
Los errores se clasifican en:

- **Compulsory:** En el primer acceso a un bloque, este debe ser llevado a caché. (**fallo de arranque en frío**)
- **Capacity:** ocurre porque los bloques se descartan de la caché debido a que no puede contener todos los bloques necesarios para la ejecución del programa.
- **Conflict:** Varios bloques se asignan al mismo conjunto de bloques. (**fallo de colisión**)

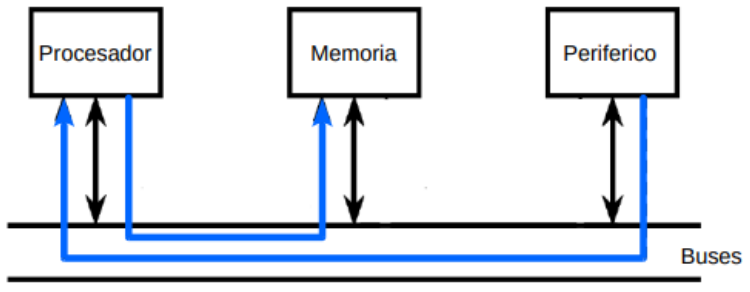
Cambiar los parámetros de la caché puede tener uno o más tipos de fallas.

Pregunta 2 (20 puntos) - Explique cómo funciona la gestión de periféricos por consulta (gestion programada) . Explique cuáles son sus ventajas y problemas. Explique la diferencia entre la gestión por consulta y la gestión por interrupciones. ¿En qué casos utilizaria una o la otra? Justifique.

8.3 Gestión programada

Es un mecanismo por el cual las tareas necesarias para atender los perifericos se organiza en el programa. La rutina atiende cada uno de los perifericos hasta completar la lista de tareas.

El procesador tiene control directo sobre la operación de transferencia: i) Comprobación del estado del dispositivo, ii) envío de comandos de lectura/escritura, iii) transferencia de datos, y iv) la CPU espera que el dispositivo termine la transferencia



Como el orden de atencion es controlado por el programa, de modo que la prioridad de cada tarea puede ser asignarda dinámicamente de acuerdo al estado general del sistema.

Esta asignacion se realiza a traves del cambio de orden de consulta.

Ventajas: se tiene control total sobre el proceso, tiene una alta repetibilidad en la ejecucion, es robusto a los eventos externos sincronicos.

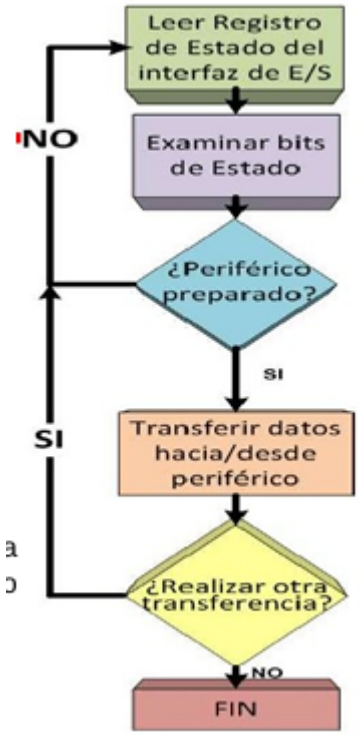
Inconvenientes: tiene altos tiempos de latencia, es ineficiente en el uso del tiempo, no puede atender eventos asincronos y la carga de programacion es elevada.

La tarea básica del programa va a ser atender los periféricos

Los pasos ejecutados durante una gestion programada son:

1. El procesador direcciona un dispositivo y solicita una operación
2. El dispositivo direccionado realiza la operación
3. El dispositivo activa los bits de estado y espera una nueva operación
4. El procesador comprueba periodicamente el estado de esos bits hasta ver que la tarea fue completada.
5. En caso contrario espera y vuelve a comprobar más tarde

Como el procesador tiene que esperar que el módulo termine la operación, el procesador permanece ocioso durante un periodo de tiempo.



10.2 Gestión por interrupciones

Una interrupción es una señal (hardware o software) que indica al procesador que un evento de alta prioridad requiere la interrupción del código que el procesador está ejecutando. El procesador responde suspendiendo sus actividades actuales, guardando su estado y ejecutando una rutina de servicio de la interrupcion. Una vez que finaliza, el procesador reanuda las actividades normales.

Las interrupciones:

- **Permiten responder a eventos sincronos y asincronos;**
- **Especificar las rutinas específica de atención de cada evento;**
- **Provee una forma simple de comunicación entre procesos**

Ventajas: el procesador puede ejecutar otro código mientras espera un evento
Inconvenientes: dificulta el cálculo de los tiempos de ejecución y latencia, es complejo gestionar interrupciones anidadas

Las etapas de la ejecución de una interrupción son:
1.Completar la ejecución de la instrucción en curso (si se puede);
2.Determinar la fuente de la interrupción;
3.Analizar si se atiende, o no, la interrupción;
4.Si se atiende, salvar la información del programa en ejecución;
5.Determinar la dirección de la rutina de atencion y ejecutarla; y

6.Una vez finalizada la ejecución de la rutina de interrupción reasumir la ejecución del programa

CUANDO USAR CADA UNA:

Gestión por consulta (programada)
Cuándo usarla: Cuando sabes exactamente en qué orden deben atenderse los dispositivos y puedes darte el lujo de esperar. Es buena en sistemas simples donde los eventos son predecibles.

Gestión por interrupciones
Cuándo usarla: Cuando pueden suceder cosas inesperadas en cualquier momento y necesitas reaccionar rápido. Es ideal para sistemas más complejos o críticos, como computadoras o sistemas de emergencia.

Pregunta 3 (20 puntos) - Explique qué es y cómo funciona un puerto serie. Explique cuáles son sus ventajas y desventajas. Explique la principal diferencia entre un puerto serie síncrono y asíncrono. ¿Cuáles son los mecanismos de un puerto asíncrono para lograr la sincronización entre el transmisor y el receptor?

12.3 Puertos serie

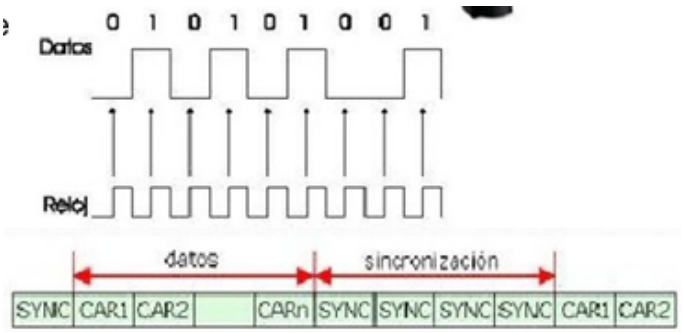
Un puerto serie o puerto en serie es una interfaz de comunicaciones de datos digitales donde la información es transmitida un solo bit a la vez; en contraste con el puerto paralelo que envía varios bits simultáneamente.

12.3.1 Puerto Serie Síncrono

En una transferencia serial síncrona se intercambian una o varias señales de control, junto con los paquetes de informacion, entre emisor y receptor. Las señales de control determinan cuando hay un bit de datos válido en la línea de transmisión

- Puede existir una señal de reloj que controle la sincronización; y/o
- La sincronización se realiza a través de un protocolo solicitud – reconocimiento o estructuras de información predeterminadas

La transmisión es controlada por el emisor. La información se transmite entre dos grupos de datos denominados delimitadores.
Ventajas: Posee un alto rendimiento; Operan a altas velocidades; El flujo de datos es regular.
Desventajas: Los equipamientos son más complejos y de costosos



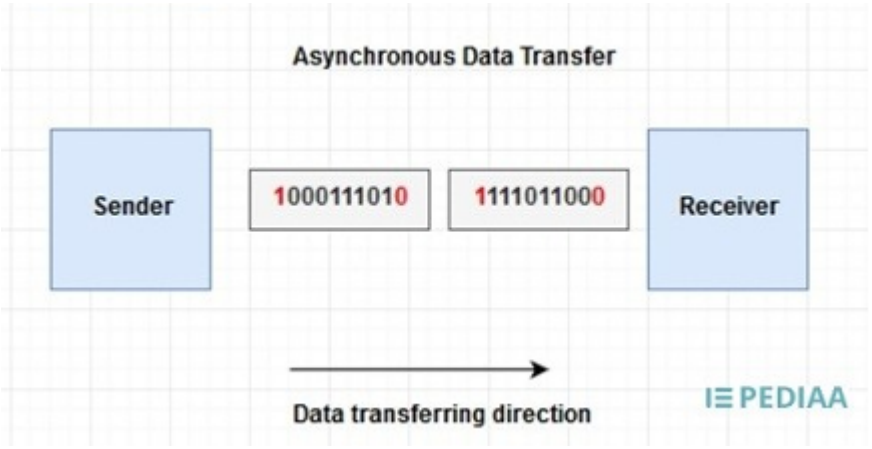
12.3.2 Puerto Serie Asíncrono

Una transferencia asíncrona ocurre cuando no hay relación temporal entre los relojes del transmisor y el receptor. La sincronización se realiza en cada dato transmitido y a través de estructuras de datos.

El ritmo de presentación de la información en el receptor no tiene que coincidir con el del transmisor. No es necesario garantizar un ancho de banda determinado, suministrando el que se disponga.

Por ello es necesario que

- Los datos contengan la información de temporización;
- El receptor muestre la línea a intervalos regulares y mayores que un pulso para detectar la llegada de datos.



TRANSFERENCIA SINCRONA:

- Permite mayores velocidades de transmisión;
- Permite que el receptor pueda interactuar con emisores de frecuencia de reloj variada;
- Permite interconectar una menor variabilidad de dispositivos, ya que emisor y receptor deben cumplir con el mismo protocolo de transmisión.

TRANSFERENCIA ASINCRONA:

- Es más lenta debido a la información adicional y los de captura;
- En caso de errores se recupera la transmisión; Bajo rendimiento de transmisión;
- Equipamiento económico y robusto;
- Aplicable a transmisiones con flujo de datos irregular;
- Permite interconectar una mayor variabilidad de dispositivos

PARA IMPLEMENTAR UN SISTEMA ASINCRONICO:
Se deben acordar los parámetros de señalización

- Tipo de operación (full o half-duplex) a realizar;
- Tamaño del dato (cantidad de bits por carácter) y el orden en que van a ser enviados;
- La velocidad de transmisión;
- El código de detección-corrección de errores utilizado; y
- La estructura de información de sincronización.

Ademas podemos distinguir el puerto serie del paralelo:
El puerto paralelo:
Es teóricamente más rápido
Los datos no necesitan ser preprocesados
Es menos flexible
Es más costoso
Tiene predisposición a sufrir errores en distancias largas

El puerto serie:
Es teóricamente más lento
Los datos deben ser preprocesados (serializados/deserializados)
Es más flexible
Es mucho menos costoso por su menor número de líneas
Tiene una menor probabilidadsición a sufrir errores, incluso en distancias largas