**Resumen para Final de Arquitectura de computadoras**

**Este resumen tiene preguntas de finales y contenido teórico de la materia, según el programa que está en la página de Arquitectura de Computadoras.**

**Unidad 1 : Arquitectura y Organización de Computadoras**

Concepto de Arquitectura. Relación con Organización de Computadoras. Repaso del modelo de von Neumann. Descripción del funcionamiento de un sistema basado en un microprocesador. Buses, teoría de operación, buses sincrónicos y asincrónicos. Ejemplos. Repaso de ejecución de instrucciones. Ejecución solapada (“pipeline”). Su aplicación en procesadores contemporáneos. Análisis de prestaciones. Arquitecturas reconfigurables: conceptos. Sistemas embebidos: conceptos.

***Modelo de Von Neumann***

El modelo de von Neumann es una arquitectura de computadoras que se basa en tres conceptos clave:

**•Datos e instrucciones se almacenan en una única memoria de lectura-escritura:** Esta memoria se organiza en posiciones con direcciones secuenciales

•Los contenidos de esta memoria se direccionan según posición, independientemente del tipo de dato que contengan.

•La ejecución de instrucciones se produce de forma secuencial, una tras otra, a menos que se modifique explícitamente el flujo.

**Que elementos componen una maquina con arquitectura Von Neumann? Describir la función de cada uno. (tomada en final de 27/11/2024)**

Una maquina con arquitectura Von Neumann se compone de los siguientes elementos:

**CPU (Unidad central de procesamiento):** La CPU es el cerebro de la computadora. Se encarga de ejecutar las instrucciones de los programas, realizar operaciones, controlar el flujo de datos.

**Memoria principal:** Almacena tanto los programas como los datos que la CPU necesita para ejecutarlos. Es volátil. Lo que significa que su contenido se pierde al apagar la computadora. El acceso a la memoria principal es aleatorio.

**Dispositivos de e/s:** Permiten a la computadora interactuar con el mundo exterior. Los dispositivos de e/s, como teclado, mouse, permiten al usuario introducir datos en la computadora. Los dispositivos de salida como el monitor y impresora, muestran los resultados del procesamiento al usuario.

**Bus del sistema:** Es un camino de comunicación entre dos o más dispositivos. Conjunto de líneas que conectan diferentes componentes de la computadora, permitiendo la transferencia de datos, direcciones e instrucciones entre la CPU,la memoria principal y los dispositivos de e/s.

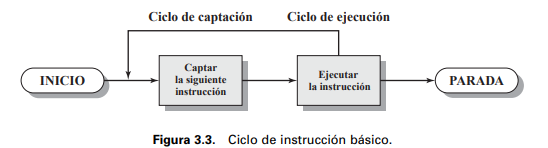
**Describir el funcionamiento de un sistema basado en microprocesador**

Un sistema basado en microprocesador funciona ejecutando un programa almacenado en la memoria, que consiste en una secuencia de instrucciones. La CPU es la encargada de llevar a cabo estas instrucciones. El funcionamiento básico implica la repetición de un ciclo de captación y ejecución de instrucciones, mejor conocido como ciclo de instrucción.

Ciclo de instrucción:

*El procesamiento que requiere una instrucción* *se denomina ciclo de instrucción.*

Un ciclo de instrucción consiste en la captación de la instrucción, seguida de ninguno o varios accesos a operandos, ninguno o varios almacenamientos de operandos y la comprobación de las interrupciones (si están habilitadas).



Al comienzo de cada ciclo de instrucción, la CPU capta una instrucción de memoria. Se utiliza un registro llamado PC (program counter) para saber cuál es la próxima instrucción a ejecutar. (A no ser que se indique otra cosa, la CPU siempre incrementa el PC después de captar cada instrucción).

**Captación de instrucción:** El procesador capta (lee) la siguiente instrucción desde la memoria. Esta instrucción se almacena en el registro de instrucción (IR).

**Ejecución de la instrucción:** El procesador lleva a cabo la operación especificada por la instrucción. Esta operación puede ser aritmética/lógica, de transferencia de datos, operaciones de control, operación de e/s, etc.

Además, en el ciclo de instrucción podemos encontrar la etapa de decodificación de la instrucción, que se encuentra entre la primer y segunda etapa, es decir, en el medio de ellas, luego de captar la siguiente instrucción y antes de ejecutarla.

**Decodificación de la instrucción:** El procesador interpreta el código de operación de la instrucción para determinar la operación a realizar.

**Buses**

**Que es un bus, tipos de buses, aspectos claves para el diseño de buses.**

Un bus, es un camino de comunicación entre dos o más dispositivos. Se trata de un medio de transmisión compartido. Al bus se conectan varios dispositivos y cualquier señal transmitida por uno de los dispositivos, estará disponible para cualquier dispositivo conectado al bus. Si dos dispositivos transmiten durante el mismo periodo de tiempo, sus señales pueden solaparse y distorsionarse

**Solo un dispositivo puede transmitir con éxito en un momento dado.**

Es importante tener un método de arbitraje, para controlar el permiso del bus a los dispositivos.

Existen varios **tipos de buses** dentro de una computadora

**Bus de datos:** Proporcionan un camino de comunicación para transmitir datos entre los módulos del sistema. Cuenta con una cantidad de líneas, conocidas como anchura del bus. Una línea puede transmitir de a un bit a la vez, por ende, el número de líneas determina cuantos bits se pueden transferir al mismo tiempo.

**Bus de dirección:** Se utilizan para designar fuente o destino del dato situado en el bus de datos.

**Bus de control:** Se utilizan para controlar el acceso y el uso de las líneas de datos y dirección.

**Jerarquía de buses**

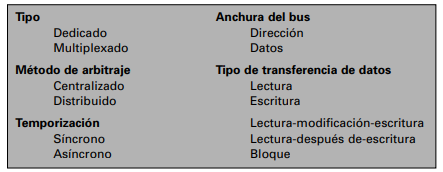
La mayoría de las computadoras utilizan varios buses, normalmente organizados jerárquicamente. Un bus principal, llamado bus del sistema, conecta los componentes principales (CPU,Memoria,E/S). Otros tipos de buses pueden conectar dispositivos de alta velocidad al procesador o la memoria, formando una jerarquía de buses.

La jerarquía de buses permite optimizar el rendimiento al proporcionar caminos de comunicación especializados para diferentes tipos de dispositivos y velocidades de transferencia.

Si se conecta un gran número de dispositivos al bus, las prestaciones pueden disminuir, hay dos causas principales:

* 1. A mas dispositivos conectados al bus, mayor es el retardo de propagación
  2. El bus puede convertirse en un cuello de botella a medida que las peticiones de transferencia acumuladas se aproximan a la capacidad del bus. (este problema se puede resolver incrementando la velocidad a la que el bus puede transferir datos, se podría usar buses más anchos)

**Cuáles son los elementos/aspectos claves para el diseño de un bus?**

****

**Tipos de líneas del bus:** Pueden ser de dos tipos (Dedicadas o Multiplexadas)

**Línea dedicada:** Una línea de bus dedicada, está permanentemente asignada a una función o a un dispositivo en particular. Puede usarse para transferir datos, direcciones o señales de control, pero solo una función y dispositivo a la vez.

**Línea multiplexada:** Comparte varias funciones o dispositivos. Transmite diferentes tipos de información en el mismo conjunto de líneas de comunicación. Puede encargarse de transmitir datos, señales de control, direcciones, etc, en el mismo conjunto de líneas.

**Métodos de arbitraje**

Son esenciales para controlar el acceso y el permiso al bus, generalmente en un bus se conectan varios dispositivos, y como un dispositivo puede transmitir con éxito en un momento dado, se necesita un método de arbitraje.

**Hay dos tipos de arbitraje**

**Arbitraje centralizado:** Existe un único dispositivo de hardware, llamado controlador o arbitro del bus, el cual es el responsable de asignar tiempo en el bus.

**Arbitraje distribuido:** No hay un controlador, sino que cada modulo dispone de la lógica necesaria para controlar el acceso al bus. Los módulos deben actuar conjuntamente para asignarse el acceso al bus.

**Temporización**

El termino temporización hace referencia a la forma en la que se coordinan los eventos en el bus. Los buses utilizan temporización síncrona o asíncrona.

**Temporización síncrona:** La presencia de un evento en el bus, está determinada por un reloj. El bus incluye una línea de reloj a través de la que se transmite una secuencia en la que se alternan intervalos regulares de igual duración a uno y a cero.

**Temporización asíncrona**: La presencia de un evento en el bus es consecuencia y depende de que se produzca un evento previo.

**Anchura del bus**

La anchura del bus afecta a las prestaciones del sistema. Cuanto más ancho es el bus de datos, mayor es el número de bits que se transmiten a la vez.

Cuanto más ancho es el bus de direcciones => Mayor es el rango de posiciones a las que se puede hacer referencia.

●**Número de líneas de dirección:** Determina la cantidad de memoria que puede direccionar el sistema.

●**Número de líneas de datos:** Define cuántos bits se pueden transferir simultáneamente, afectando la velocidad de transferencia.

**Tipo de Transferencia de Datos:**

●**Lectura:** Transferencia de datos del esclavo al maestro.

●**Escritura:** Transferencia de datos del maestro al esclavo.

●**Lectura-modificación-escritura:** Una operación indivisible para leer, modificar y escribir datos en una ubicación de memoria.

●**Lectura-después-de-escritura:** Una operación indivisible para escribir un dato y luego leerlo para verificar.

●**Transferencia de bloque:** Un ciclo de dirección seguido de varios ciclos de datos para transferir un bloque de información

**BUS PCI**

El bus PCI es un **bus de ancho de banda elevado**, **independiente del procesador**, que **permite conectar y comunicar periféricos con la placa madre**. El bus PCI **proporciona una interfaz de alta velocidad**, permitiendo la transferencia de datos rápida entre los periféricos y el procesador o la memoria.

Proporciona un conjunto de funciones de uso general. **Utiliza temporización síncrona, un esquema de arbitraje centralizado, y utiliza líneas multiplexadas para compartir datos y direcciones en el mismo conjunto de líneas.**

Características del Bus PCI:

**Alto ancho de banda:** Ofrece altas tasas de transferencia de datos, lo que lo hace adecuado para subsistemas de e/s de alta velocidad, como por ejemplo adaptadores de pantalla grafica, controladores de interfaz de red.

**Fácil implementación:** Se implementa con pocos circuitos integrados. Está diseñado para ajustarse económicamente a los requisitos de e/s de los sistemas actuales.

**Unidad 2 : Subsistema Unidad Central de Procesos**

Repaso de máquinas que ejecutan instrucciones. Ejemplificación en procesadores típicos: IA32. Análisis del conjunto de instrucciones de procesadores de uso comercial. Concepto de máquinas CISC y RISC. Lineamientos básicos en el diseño de un procesador RISC. Análisis de prestaciones. Ejemplos: procesadores MIPS y ARM. Interrupciones: tratamiento general. Interrupciones por software y por hardware, vectores, descripción y tratamiento particular de cada una. Relación entre las interrupciones y el manejo de operaciones de E/S.

**Analisis del conjunto de instrucciones**

El análisis del conjunto de instrucciones implica examinar las características de las instrucciones máquina, incluyendo el código de operación (codop), los operandos y los modos de direccionamiento.

Desde el punto de vista del diseñador, el conjunto de instrucciones máquina constituye la especificación o requisitos funcionales del procesador.

El repertorio de instrucciones es el medio que tiene el programador para controlar el procesador

**Repertorio de instrucciones: Características y funciones**

El repertorio de instrucciones, define las operaciones que el procesador puede llevar a cabo.

Los elementos esenciales de las instrucciones de las computadoras son:

**Código de operación (codop):** Especifica la operación que se va a realizar, por ejemplo operaciones aritméticas, lógicas, transferencia de datos entre registros, etc.

**Referencia a operandos origen:** Indica los operandos que sirven como entrada para la instrucción.

**Referencia a operandos Destino:** Específica donde se almacena el resultado de la operación. No todas las operaciones producen un resultado, pero aquellas operaciones que lo producen, deben indicar el destino.

**Referencia a la siguiente instrucción:** Indica al procesador donde encontrar la siguiente instrucción a ejecutar después de completar la instrucción actual. Esta referencia se encuentra en el PC (Program Counter)

**Los aspectos fundamentales para el diseño de repertorio de instrucciones son:**

• **El repertorio de operaciones:** cuántas y qué operaciones considerar, y cuán complejas deben ser.

• **Los tipos de datos:** los distintos tipos de datos con los que se efectúan operaciones.

• **Los formatos de instrucciones**: longitud de la instrucción (en bits), número de direcciones, tamaño de los distintos campos, etc.

• **Los registros:** número de registros del procesador que pueden ser referenciados por las ins trucciones, y su uso.

• **El direccionamiento**: el modo o modos de direccionamiento mediante los cuales puede especificarse la dirección de un operando.

**Procesadores RISC y CISC**

**CISC (Conjunto de Instrucciones Complejo):** Los procesadores CISC tienen un conjunto amplio y complejo de instrucciones, se utilizan para instrucciones más complejas. Permite realizar tareas mas avanzadas en una sola instrucción. **Cada instrucción podría requerir mas de un ciclo de reloj.** Este tipo de arquitectura cuenta con una gran variedad de modos de direccionamiento y se utiliza comúnmente en lenguajes de alto nivel. **Su enfoque se centra en la ejecución secuencial de instrucciones.**

**Ventajas**: Los programas tienden a ser más compactos, ya que las instrucciones más potentes permiten escribir código más corto y con menos líneas. Además, se reduce la necesidad de acceder a la memoria, ya que las instrucciones complejas pueden realizar varias operaciones sin requerir un acceso frecuente a la memoria principal. Un ejemplo representativo de procesador CISC es el Intel Pentium.

**Desventajas**: Las instrucciones suelen tener ciclos de ejecución más largos, debido a que las operaciones más complejas pueden necesitar más de un ciclo de reloj. Además, la decodificación y ejecución de estas instrucciones **requieren un hardware más complejo y costoso.**

**RISC (Conjunto de instrucciones reducido):** Este tipo de procesadores priorizan la simplicidad y velocidad. Tienen un repertorio de instrucciones pequeño/reducido, y se compone de **instrucciones sencillas que se ejecutan en un ciclo de reloj.**  Posee un número limitado de modos de direccionamiento, pero con la diferencia que posee una gran cantidad de registros de propósito general. **Este tipo de procesadores, hace énfasis en segmentación de instrucciones para ejecutar múltiples instrucciones en paralelo.**

**Ventajas:** Requiere hardware mas simple, y mas económico. Tiene ciclos de instrucciones mas cortos, ya que solo requieren un ciclo de reloj.

**Desventajas:** Se necesitan mas instrucciones para realizar tareas mas complejas, lo que puede resultar en programas largos. Ademas, al tener instrucciones mas simples, requiere un mayor numero de accesos a la memoria principal.

**Interrupciones**

Las interrupciones son un mecanismo que permite alterar el proceso de ejecución normal de la CPU. Una interrupción permite que la CPU suspenda lo que está ejecutando y responda a una rutina de interrupción. Las interrupciones son señales que indican al procesador que debe atender a un evento.

Mientras la CPU esta ejecutando, si recibe un pedido de interrupción, y esa interrupción esta habilitada, se detiene la ejecución del programa en curso, la cpu salva su contexto, guarda el PC y el registro de estado y salta a una rutina de gestión de interrupción. Una vez que termina de atender la interrupción, se recupera el PC, y se sigue con la siguiente instrucción, siempre y cuando haya otra instrucción que ejecutar y que no haya ningún otra interrupción para atender.

**Por que son importante las interrupciones?**

Las interrupciones mejoran la eficiencia del procesador, permitiendo manejar múltiples tareas y eventos. Además, permite que el procesador se enfoque en las tareas más críticas o urgentes, suspendiendo la ejecución normal para atender eventos importantes. Sin interrupciones, un procesador tendría que esperar a que se complete una tarea antes de pasar a la siguiente. Las interrupciones facilitan la ejecución de múltiples programas de manera concurrente.

**Tipos de interrupciones:**

**Interrupción por SOFTWARE:** Generadas como resultado de la ejecución de una instrucción del programa actual. Por ejemplo overflow, dividir por cero, intentar ejecutar una instrucción inexistente, o intentar acceder fuera del espacio de memoria permitido. Estas interrupciones son **sincrónicas** porque ocurren como resultado de la ejecución del programa actual.

**Manejo típico de las interrupciones internas: El SO invoca una rutina para gestionar el error (puede ser mostrar un mensaje de error, detener el programa,etc).**

**Interrupción por HARDWARE:** Generadas por eventos externos al procesador, como dispositivos periféricos o señales de control. Por ejemplo interrupción por temporización (un reloj interno genera interrupciones periódicas para que el sistema realice algún tipo de tarea), otro ejemplo son los controladores de e/s (un dispositivo periférico indica que ha terminado una operación de e/s o que requiere atención). Estas interrupciones son **asincrónicas**, no están vinculadas a la ejecución del programa actual, sino que este tipo de interrupciones pueden ocurrir en cualquier momento.

**Manejo típico de las interrupciones externas: Se ejecuta una rutina específica asociada al evento externo, por ejemplo leer datos de un disco, o procesar una entrada del usuario.**

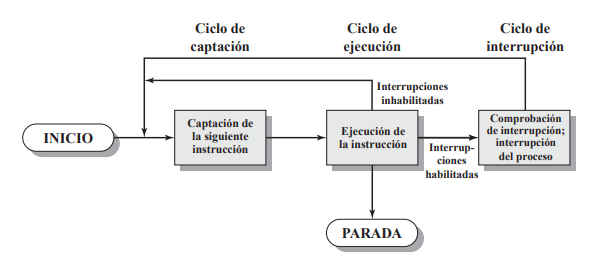
**¿Por qué son necesarias las interrupciones?**

1. **Mejoran la eficiencia del procesador**:
   * Sin interrupciones, el procesador tendría que emplear técnicas como el *polling*, donde revisa continuamente el estado de los dispositivos. Esto resulta ineficiente porque consume ciclos de reloj incluso cuando no hay eventos que manejar.
   * Con las interrupciones, el procesador puede concentrarse en otras tareas y responder solo cuando ocurre un evento.
2. **Permiten la multitarea**:
   * Las interrupciones son esenciales para implementar sistemas operativos multitarea. Estas permiten cambiar entre procesos (context switching) y atender múltiples tareas aparentemente al mismo tiempo.
3. **Soporte para dispositivos de E/S más lentos**:
   * La mayoría de los dispositivos de E/S son más lentos que el procesador. Las interrupciones permiten que el procesador no desperdicie tiempo esperando que un dispositivo esté listo.
4. **Respuesta inmediata a eventos críticos**:
   * Permiten manejar eventos urgentes, como señales de hardware críticas (fallas en la alimentación eléctrica, fallas de hardware, etc.), sin demoras.

**Interrupciones y el ciclo de instrucción:**

Con el uso de interrupciones, el procesador puede dedicarse a ejecutar otras instrucciones mientras una operación de E/S está en curso.

El ciclo de instrucción con interrupciones es similar al ciclo básico, pero incluye una etapa adicional al final de cada instrucción: después de ejecutar una instrucción y antes de captar la siguiente, el procesador verifica si las interrupciones están habilitadas y si hay una interrupción pendiente. Si ambas condiciones se cumplen, suspende temporalmente la ejecución del programa principal, guarda su contexto, atiende la interrupción ejecutando la rutina correspondiente y luego restaura el contexto para continuar con la ejecución del programa que se interrumpió.



En el ciclo de interrupción, el procesador comprueba si se ha generado alguna interrupción, indicada por la presencia de una señal de interrupción.

* Si no hay señales de interrupción: El procesador continua el ciclo de captación y accede a la siguiente instrucción del programa en curso.
* Si hay alguna interrupción pendiente, el procesador hace lo siguiente:
  + Suspende la ejecución del programa en curso y guarda su contexto (guarda la dirección de la próxima instrucción a ejecutar)
  + Carga el PC (program counter) con la dirección de comienzo de una rutina de gestión de interrupción.

**Tratamiento de múltiples interrupciones:**

Existen dos alternativas principales para tratar las interrupciones múltiples:

**Primera alternativa: Desactivar las interrupciones**  
En este enfoque, se desactivan las interrupciones mientras se está procesando una interrupción. Una vez deshabilitadas, el procesador ignora cualquier nueva señal de interrupción hasta que se vuelvan a habilitar. Si ocurre una interrupción mientras está desactivada, se mantiene pendiente y se examina nuevamente cuando las interrupciones se reactiven.

* **Ventajas:** Esta solución es simple y efectiva, ya que las interrupciones se manejan de manera secuencial.
* **Desventajas:** No tiene en cuenta la prioridad de las interrupciones, lo que puede generar un retraso en el manejo de interrupciones importantes si no se gestionan adecuadamente.

**Segunda alternativa: Definir prioridades a las interrupciones**  
Este enfoque permite que las interrupciones de mayor prioridad interrumpan a aquellas de menor prioridad. Así, una interrupción con mayor urgencia puede interrumpir el procesamiento de una interrupción de menor prioridad, asegurando que se atiendan las tareas más críticas.

* **Ventajas:** Las interrupciones de mayor prioridad son atendidas rápidamente, lo que es crucial en sistemas donde las tareas de alta prioridad deben ser resueltas con urgencia.
* **Desventajas:** Las interrupciones de menor prioridad pueden sufrir inanición, ya que si constantemente se presentan interrupciones de mayor prioridad, estas nunca se procesan.

**Hay 3 Formas de detectar pedidos de interrupción:**

Opción 1: Líneas de interrupción:

* Cada dispositivo tiene una línea física conectada a la CPU, de esta forma, se puede reconocer al dispositivo que emite una solicitud de interrupción.

Opción 2: Encuesta o Polling:

* Existe una línea física de pedidos de interrupción, la cual comparten varios dispositivos. En este método, la CPU pregunta a cada dispositivo conectado, para saber cual fue el que emitió la solicitud de interrupción.

**Este método puede resultar ineficiente, ya que se le debe preguntar a cada dispositivo conectado.**

Opción 3: Vector de Interrupciones:

* Cada interrupción está asociada con un id único, este id sirve como índice para acceder a la rutina de interrupción especifica.
* Con el id de interrupción se busca en la tabla de interrupción, la dirección del comienzo de la rutina de interrupción para ser atendida.

**Interrupciones ENMASCARABLES y NO ENMASCARABLES**

***Las interrupciones Enmascarables*:** Son aquellas que pueden ser ignoradas temporalmente, se pueden inhabilitar mediante un mecanismo del procesador, lo que permite al SO controlar cuando se atienden y cuando no. Generalmente las interrupciones enmascarables no son interrupciones de alta prioridad, ni interrupciones criticas que deben ser atendidas de inmediato.

***Las interrupciones No Enmascarables:*** Son aquellas que no pueden ser ignoradas, ya que indican eventos de alta prioridad, y deberían ser atendidas lo antes posible. Son atendidas por el procesador, indican eventos críticos como condiciones de fallo del sistema.

**Cual es la relación entre las interrupciones y las operaciones de E/S?**

**E/S por interrupción:** Antes de iniciar la operación, el controlador de E/S se configura para generar una interrupción cuando la operación E/S se complete. De esta manera, mientras el controlador de E/S se encarga de realizar la operación, el procesador puede continuar ejecutando otras tareas y no queda inactivo. Una vez que se complete la operación, se genera una interrupción, entonces el procesador interrumpe sus tareas y ejecuta una rutina de servicio de interrupciones. Después de manejar la interrupción, el procesador retorna la tarea que estaba ejecutando antes de ser interrumpido**. Este método permite al procesador realizar otras tareas mientras espera que se complete la operación de E/S**

**Entrada / Salida**

Un modulo de E/S es un componente esencial en la computadora**, su función principal, es actuar como intermediario entre el procesador, los dispositivos periféricos y la memoria.** El modulo de E/S **se encarga de gestionar el flujo de datos y control** entre ellos, **permite al procesador que interactué con una amplia variedad de dispositivos** de forma simplificada. Es capaz de reconocer y generar direcciones asociadas a los periféricos que controla.

**Funciones de un modulo de E/S:**

* **Control y temporización:** 
  + Coordinan el flujo de datos entre el procesador, la memoria y los dispositivos periféricos. Debido a las diferencias de velocidad entre estos componentes, el modulo de e/s debe sincronizar las operaciones para garantizar una transferencia de datos eficiente.
* **Comunicación con el procesador:** 
  + Decodifican las ordenes enviadas al procesador.
  + Reportan el estado del dispositivo al procesador.
  + Manejan las interrupciones generadas por los dispositivos.
* **Comunicación con los dispositivos periféricos:**
  + Envían y reciben datos entre el dispositivo y el modulo de e/s

**Diseño de un modulo de e/s**

Un modulo de e/s consta de los siguientes componentes:

* + **Registro de datos:** Almacenan temporalmente los datos que se transfieren entre el dispositivo y el procesador.
  + **Registro de control y estado:** Proporcionan información sobre el estado actual del modulo y del dispositivo.
  + **Logica de interfaz con el bus del sistema:** Permite la comunicación con el procesador y la memoria principal a través del bus del sistema.

**Estructura de un modulo de e/s:**

**La estructura de un modulo de e/s, esta compuesta por:**

* **Interfaz al bus del sistema**: El modulo se conecta al resto del sistema, a través de un conjunto de líneas:
  + Lineas de dirección: Se utilizan para identificar el modulo de e/s y los dispositivos que controla.
  + Lineas de datos: Proporcionan un camino para la transferencia bidireccional de datos entre el modulo de e/s y el procesador o la memoria. El ancho del bus, determina cuantos bits se pueden transferir simultáneamente.
  + Lineas de control: Se utilizan para coordinar las operaciones entre el procesador y el modulo de e/s.
* **Registros:** 
  + Registros de datos: Almacenan temporalmente datos que se transfieren entre el periférico y el resto del sistema.
  + Registro de estado: Proporcionan información sobre el estado actual del modulo de e/s.
  + Registro de control: Se utiliza para configurar el modo de operación del modulo de e/s.
* **Interfaz a dispositivos externos:** Contiene la lógica y los circuitos específicos necesarios para la comunicación con cada uno de los dispositivos externos que controla el modulo.

**Conclusión de modulos de E/S:** Los módulos de e/s son esenciales para el funcionamiento de un sistema de computo**, permitiendo la interacción con el mundo exterior y la utilización de una amplia variedad de dispositivos periféricos.**

**Técnicas de E/S:**

**E/S programada con espera de respuesta:** Se refiere a que el procesador emite una solicitud de operación de E/S. **El procesador espera activamente hasta que el dispositivo de E/S complete la operación** y le avise al procesador que la operación ha finalizado. Durante este periodo de espera, **el procesador permanece inactivo**, ya que se dedica a verificar repetidamente el estado de la operación de E/S. Una vez que el procesador recibe la señal de finalización de la operación, puede seguir ejecutando.

Este método, resulta **ineficiente para operaciones prolongadas**, ya que el procesador quedaría inactivo durante largos periodos de tiempo. Es mas adecuado usarlo cuando tenemos operaciones cortas, donde el tiempo de espera sea breve.

**E/S** **por interrupción:** En este método, antes de iniciar la operación, el controlador de E/S se configura para generar una interrupción cuando la operación se complete. De esta manera, mientras el controlador de E/S se encarga de realizar la operación, el procesador puede continuar ejecutando otras tareas y no queda inactivo. Una vez que se **complete la operación, se genera una interrupción**, entonces el procesador interrumpe sus tareas y ejecuta una rutina de servicio de interrupciones. Después de manejar la interrupción, el procesador retorna la tarea que estaba ejecutando antes de ser interrumpido**. Este método permite al procesador realizar otras tareas mientras espera que se complete la operación de E/S**

**Acceso Directo a Memoria (DMA):** El DMA imita al procesador. **Permite que los dispositivos periféricos transfieran datos** hacia o desde la memoria, **sin la necesidad de la intervención directa del procesador.** El controlador de DMA (DMAC) es el encargado de llevar a cabo las transferencias, liberando a la CPU de la tarea de gestionar cada transferencia, de esta manera, el DMA reduce la carga del procesador.

**Formas de direccionamiento de E/S**

El direccionamiento de e/s se refiere a como el procesador accede a los dispositivos periféricos para enviar o recibir datos.

Algunas formas de direccionamiento de e/s son:

***Direccionamiento por Puertos( I / O PORTS):*** Implica asignar un conjunto de direcciones de memoria especificas a los puertos de e/s. Cada puerto se considera una ubicación de memoria única, a la que el procesador puede acceder para enviar o recibir datos desde o hacia dispositivos periféricos.

***Direccionamiento por Mapeo de Memoria:*** Los registros de control y estado de los periféricos se asignan a direcciones de memoria especificas. El procesador utiliza instrucciones de carga y almacenamiento (LOAD / STORE) para interactuar con los periféricos.

**Puertos de e/s paralelo y serie**

**Puerto paralelo:** Hay múltiples líneas que conectan el modulo de e/s y el periférico, lo que permite la transferencia simultanea de varios bits a través del bus de datos. Se utilizan generalmente para dispositivos de alta velocidad **Cada dispositivo de e/s tiene su propio espacio de direcciones**

**Puerto serie:** Hay solo una línea para transmitir los datos, por lo que los bits deben transmitirse uno a uno de forma secuencial. Este tipo de puertos, son mas adecuados para dispositivos de velocidad mas baja, como impresoras.

**Que es el DMA? Cual es su funcionamiento?**

El DMA es una técnica de transferencia de datos entre el periférico y hacia o desde la memoria, sin intervención directa de la CPU. Esta llevada a cabo por un controlador de DMA (DMAC), el cual es el responsable de llevar a cabo la transferencia.

El DMA es usado para mejorar la transferencia de datos entre dispositivos periféricos y la memoria principal a través del bus del sistema. Cuando un dispositivo necesita transferir datos a la memoria, el DMA toma el control temporalmente del bus del sistema. Al liberar la CPU de la tarea de gestionar cada transferencia, el DMA reduce la carga del procesador.   
**Existen dos modos de transferencia de DMA:**

**Transferencia por bloques:** El DMA solicita el control del bus tantas veces como sea necesario hasta finalizar la transferencia del bloque completo.

**Transferencia por ráfaga**: El DMA no libera el bus hasta haber finalizado la transferencia de todo el bloque de datos.

En general, para transferencias de e/s mas prolongadas, la técnica por bloques es la mas eficiente, ya que permite implementar la transferencia al mismo tiempo que la CPU continua trabajando en otras tarea.

**Unidad 4 :  Subsistema Memoria**

Repaso de la organización jerárquica de la memoria, memoria principal y memoria secundaria. Memoria caché, concepto y descripción, análisis de prestaciones, métodos de implementación típicos, múltiples niveles. Ejemplos. Conceptos de memoria virtual.

**Memoria**

**Métodos de acceso:**

**Acceso Secuencial:** La memoria se organiza en unidades de datos llamadas registros. El acceso a estos registros debe realizarse de manera secuencial, con una secuencia lineal especifica.

**Acceso Directo:** El acceso se lleva a cabo mediante un acceso directo a una vecindad dada, seguido de una búsqueda secuencial.

**Acceso Aleatorio (Random):** Es un tipo de acceso a memoria en el que el procesador puede acceder a cualquier celda de memoria directamente, sin necesidad de recorrer secuencialmente las celdas de memoria previas. Este tipo de acceso es característico de la memoria volátil, como la RAM (memoria de acceso aleatorio), y es fundamental para el rendimiento de los sistemas informáticos modernos, ya que permite un acceso rápido y directo a los datos.

**Jerarquia de memoria**

Las restricciones del diseño de la memoria de una computadora, se puede resumir en tres cuestiones:

¿cuánta capacidad? ¿cómo de rápida? ¿de qué costo?

• A menor tiempo de acceso, mayor coste por bit.

• A mayor capacidad, menor coste por bit.

• A mayor capacidad, mayor tiempo de acceso

**Memoria Principal (RAM):** La memoria RAM almacena temporalmente datos e instrucciones que la CPU necesita en tiempo real. Este tipo de memoria son volátiles, lo que significa que pierde su contenido cuando se apaga la computadora. Cuenta con un método de acceso aleatorio.

**Hay dos tipos de memoria RAM : SRAM y DRAM:**

Tanto la SRAM (RAM estatica) como la DRAM (RAM dinamica), son tipos de memoria de acceso aleatorio (RAM). Los discos, por otro lado, son dispositivos de almacenamiento externo que proporcionan una capacidad mucho mayor que la memoria principal, pero con un costo de acceso mucho mayor al costo de acceso de una memoria principal (RAM).

La SRAM es una memoria aleatoria, rápida, es usada en caches y en sistemas donde la velocidad es importante.

La DRAM es una memoria aleatoria, es más lenta, más barata, y se utiliza en la memoria principal de las computadoras debido a su alta capacidad y alto costo.

**Memoria Secundaria:** Se utiliza para guardar datos de forma no volátil. Es decir, retiene su información, incluso cuando la computadora se encuentra apagada.  **Un ejemplo de almacenamiento secundario es el disco.**

**Disco:** Tiene un costo de acceso mucho mayor que la RAM, por ende, acceder a un disco es costoso por que los datos se graban y luego se recuperan del disco a través de una bobina llamada cabezal, por genera un costo extra para posicionar el cabezal de lectura/escritura.

**Memoria Virtual:**

La memoria virtual es un concepto que permite que un sistema operativo utilice tanto la memoria RAM como el almacenamiento secundario (disco duro) como si fueran una única memoria continua.

Permite que los programas utilicen más memoria de la que realmente está presente físicamente, mejorando así el rendimiento al evitar limitaciones de memoria.

**Objetivo de la jerarquía de memoria**

El objetivo de la jerarquía de memoria es abordar las limitaciones de diseño de la memoria de una computadora, que son la capacidad, la velocidad y el coste.

Es deseable tener una memoria con una gran capacidad,alta velocidad, y bajo coste. Sin embargo es difícil y costoso construir una memoria que cumpla con estos tres requisitos al mismo tiempo.

La jerarquía de memoria resuelve este problema utilizando múltiples niveles de memoria con diferentes características de coste,capacidad y velocidad.

**A medida que se desciende en la jerarquía:**

* **Disminuye el costo por bit**
* **Aumenta la capacidad**
* **Aumenta el tiempo de acceso.**

**Conclusión: La jerarquía de memoria permite obtener un buen equilibrio entre coste, capacidad y velocidad**, aprovechando el principio de localidad para minimizar el tiempo de acceso medio a los datos e instrucciones.

**Memoria Caché**

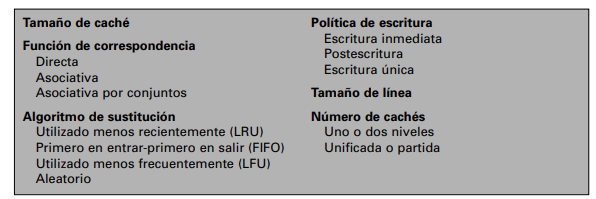
El objetivo de la memoria caché**, es lograr que la velocidad de la memoria sea lo mas rápida posible**. La caché contiene una copia de partes de la memoria principal. Cuando el procesador intenta leer una palabra de memoria, se hace una comprobación para determinar si la palabra está en la caché. Si es así, se entrega la palabra al procesador. Si no, un bloque de memoria principal, se transfiere a la cache y, después, la palabra es entregada al procesador

**La caché se organiza en bloques (también puede llamarse líneas de caché)**. Un bloque representa la cantidad de datos transferidos entre la memoria principal y la caché en una única operación.

**Acceso a los datos en memoria caché:**

La caché se basa en dos principios de localidad:

1. **Principio de localidad referencial**: Establece que los programas tiendes a acceder repetidamente a las mismas posiciones de memoria, o los mismos datos durante un periodo corto de tiempo.
   * + Si un dato o instrucción se utiliza una vez, es probable que se vuelva a utilizar en un futuro .
2. **Principio de localidad espacial**: Si un dato o instrucción se utiliza, es probable que los datos o instrucciones cercanas (en direcciones de memoria), también se utilicen en un futuro cercano.

**Elementos de diseño de caché **

**Formas de correspondencia de la caché**

**Correspondencia directa:** Cada bloque de memoria de la caché se mapea directamente en **un bloque** correspondiente en la memoria principal. Esto significa que cada ubicación en la memoria principal **tiene una única ubicación** en la caché donde puede almacenarse.

**Correspondencia asociativa por conjuntos:** Cada bloque de memoria de la caché tiene **varias ubicaciones** en la memoria principal donde puede almacenarse. Esto permite cierto grado de flexibilidad y reduce la probabilidad de conflictos de caché.

**Correspondencia totalmente asociativa:** Cualquier bloque de memoria de la caché puede almacenarse en cualquier ubicación de la memoria principal.

**Politicas de Reemplazo / Algoritmos de sustitucion de cache:**

***LRU (Least Recently Used - Menos Recientemente Utilizado):*** Reemplaza el bloque de caché que no ha sido utilizado durante el período de tiempo más largo. Es decir, se elimina el bloque que ha sido accedido menos recientemente.

***FIFO (First In, First Out - Primero en Entrar, Primero en Salir):*** El bloque que ha estado en la caché durante el período de tiempo mas largo, es el primero en ser reemplazado.

***LFU (Least Frequently Used - Menos Frecuentemente Utilizado):*** Reemplaza el bloque de caché que ha sido menos frecuentemente accedido. Es decir, se cuenta el número de accesos a cada bloque de la caché y se elimina el bloque con el menor número de accesos.

***Random (Aleatorio):*** El bloque de caché a reemplazar se elige al azar. No se tiene en cuenta el historial de acceso o la frecuencia de uso.

**Politica de escritura en cache:**

Cuando se produce una operación de escritura, surge la cuestión de si actualizar la ubicación de la memoria principal al mismo tiempo que la palabra en caché.

**Para entender las políticas de escritura: Debemos entender que es un acierto y fallo de caché**

**Acierto de caché:** Ocurre cuando el procesador intenta leer una palabra de la memoria y la palabra se encuentra en la caché. En este caso, la palabra se entrega al procesador desde la caché. Lo que es mucho más rápido que acceder a memoria principal.

**Fallo de caché:** Ocurre cuando el procesador intenta leer una palabra de la memoria y la palabra no se encuentra en la caché. En este caso, se produce un fallo de caché, y se debe acceder a memoria principal para obtener la palabra.

**Politicas de escritura en acierto de caché:** Determina si los datos se escriben solo en la caché, o tanto en la caché como en la memoria principal.

**Escritura inmediata:** La información se escribe tanto en la caché como en la memoria principal. Esta es la técnica más sencilla. Garantiza que la memoria principal esté siempre actualizada, pero puede generar un mayor tráfico en el bus de memoria.

**Postescritura:** Las escrituras se realizan solo en la caché y se escribe en la memoria principal solo cuando el bloque se reemplaza en la caché. Esto reduce el tráfico en el bus de memoria y puede mejorar el rendimiento.

**Politicas de escritura en fallo de caché:** Determina si el bloque de memoria principal que contiene la palabra a escribir se carga en la caché o no.

**Escritura y asignación:** Ante un fallo de escritura, el bloque de memoria principal se carga primero en la caché y luego se realiza la escritura en la caché.

**No escritura y asignación:** Escribe los datos directamente en la memoria principal sin cargar el bloque en la caché. Se utiliza para reducir el tráfico en el bus de memoria.

**Tamaño de línea:**

El tamaño de línea se refiere a **la cantidad de datos que se transfieren de la memoria principal a la caché en cada acceso.** Cuando se recupera y ubica en caché un bloque de datos, se recuperan no sólo la palabra deseada sino además algunas palabras adyacentes

• **Bloques más grandes**: Reducen el número de bloques que caben en la caché. **A medida que un bloque se hace más grande,** cada palabra adicional está más lejos de la requerida y por tanto es más improbable que sea necesaria a corto plazo

**Numero de cachés:**

Las caches multinivel son una técnica para mejorar el rendimiento al reducir el tiempo promedio de acceso a la memoria. Se trata de una jerarquía de cachés. Donde cada nivel es mas grande y lento que el anterior, pero mas rápido que la memoria principal.

Cuando el procesador necesita acceder a un dato, primero busca en la caché nivel1 (L1), la mas pequeña y rápida, si el dato se encuentra, se produce un acierto de caché, el acceso es muy rápido. Si el dato no se encuentra, se busca en la caché de nivel2 (L2) que es mas grande y lenta que la L1, pero mas rápida que la memoria principal, si el dato no se encuentra en L2, se produce un fallo y se accede a la memoria principal. Una **ventaja** de usar esta técnica es que reduce el tiempo promedio de acceso a la memoria.

**Unidad 5 : Paralelismo y  mejora de prestaciones**

Concepto de procesamiento paralelo. Paralelismo a nivel instrucción. Procesadores superescalares. Ejemplos. Clasificación de arquitecturas paralelo: taxonomía de Flynn. Ejemplos de aplicación. Arquitecturas Multiprocesador. Memoria compartida o distribuida. Análisis de prestaciones.

**Segmentación de Cauce**

La **segmentación de cauce** es una técnica para acelerar la velocidad de ejecución, consiste en descomponer el proceso de ejecución de instrucciones en varias fases o etapas. Estas fases o etapas son ejecutadas por unidades separadas, y son capaces de trabajar simultáneamente. Las instrucciones se ejecutan a medida que se liberan unidades, sin la necesidad de esperar a que termine una instrucción para ejecutar la siguiente. **IMPORTANTE: EN LA SEGMENTACIÓN DE CAUCE, CADA ETAPA REQUIERE AL MENOS UN CICLO DE RELOJ COMPLETO.**

***Al estar ejecutando múltiples instrucciones en paralelo, pueden producirse conflictos/atascos.***

**Los conflictos o atascos son situaciones que impiden que la siguiente instrucción se ejecute en el ciclo que le corresponda.**

|  |  |
| --- | --- |
| **Conflicto** | **Solución** |
| **Dependencia de datos:** Ocurren cuando dos o mas instrucciones comparten un mismo dato, y una instrucción necesita el resultado de otra.  **Estructurales:** Ocurren cuando dos o más partes del hardware compiten por el mismo recurso. Es decir, esta provocado por el uso de recursos, como memoria, alu, y registros.  **Dependencia de control:** Surgen cuando la ejecución de una instrucción depende de cómo se ejecute otra. Ejemplo: 1 salto y 2 posubkes caminos. | * Reordenamiento de instrucciones * forwarding (adelantamiento de operandos). * Duplicar los recursos de hardware que generan conflictos. * Segmentar los recursos * Realizar turnos para acceder a los recursos del hardware que generan conflictos. * Tecnicas de software: Salto retardado * Tecnica de hardware: Prediccion de saltos |

**Segmentación de instrucciones**

La segmentación de instrucciones es una técnica que mejora el rendimiento de los procesadores mediante el paralelismo en la ejecución de instrucciones. En la técnica de segmentación de instrucción, el ciclo de instrucción se divide en varias etapas que se ejecutan de forma paralela, cada etapa es independiente, de esta manera, no es necesario que se termine una instrucción para poder ejecutar la siguiente, sino que las instrucciones se van ejecutando a medida que se liberan unidades de CPU.

**Esta técnica, podría verse limitada por la presencia de saltos y dependencias entre instrucciones.**

La segmentación de instrucciones divide el ciclo de instrucción en etapas separadas. Estas etapas son: Captación de instrucción, decodificación de la instrucción, calculo de direcciones de operandos, captación de operandos, ejecución de instrucción y escritura del operando resultado.

**Conclusión:**

En un procesador sin segmentación: estas etapas se ejecutan secuencialmente.

Con segmentación: mientras una instrucción se está ejecutando, la siguiente instrucción ya se esta captando. Esto permite reducir el tiempo total necesario para ejecutar una secuencia de instrucciones.

**Procesamiento Paralelo:**

El procesamiento paralelo hace énfasis en que múltiples tareas se ejecuten simultáneamente para mejorar la eficiencia y el rendimiento de un sistema. El procesamiento paralelo distribuye y ejecuta múltiples tareas al mismo tiempo, ya sea en diferentes núcleos de procesamiento dentro de un procesador, en diferentes procesadores en un sistema multiprocesador, o incluso en sistemas conectados en red. Este enfoque permite abordar problemas más grandes y complejos de manera más rápida y eficiente al dividirlos en tareas más pequeñas y ejecutarlas en paralelo

**Procesamiento Paralelo a nivel de instrucciones:**

El procesamiento paralelo a nivel de instrucciones se refiere a la ejecución simultánea de múltiples instrucciones de un programa en un solo procesador. En lugar de ejecutar una instrucción a la vez, el procesador intenta identificar y ejecutar instrucciones independientes de manera simultánea, aprovechando el paralelismo a nivel de instrucción dentro de un flujo de instrucciones. Esto se logra mediante técnicas como la ejecución fuera de orden, la predicción de saltos y la segmentación de instrucciones para mejorar el rendimiento y la eficiencia del procesador.

El procesamiento paralelo a nivel de instrucciones es fundamental para aumentar el rendimiento de los procesadores modernos al aprovechar al máximo su capacidad.

**Paralelismo a nivel de maquina:**

Es una medida de la capacidad del procesador para sacar provecho al paralelismo a nivel de instrucciones. El paralelismo de la maquina depende del numero de instrucciones que pueden captarse y ejecutarse al mismo tiempo.

**Supersegmentacion**

Es una técnica que aumenta el número de etapas en el pipeline. Con más etapas, puede haber más instrucciones en el pipeline al mismo tiempo, incrementando el paralelismo.

**Procesadores**

**Procesador Superescalar:** Se caracteriza por el uso de múltiples cauces de instrucciones independientes, cada cauce consta con múltiples etapas, lo que permite procesar varias instrucciones a la vez. Este tipo de procesadores, explota el paralelismo a nivel de instrucción. **Ejemplo: El Pentium es un ejemplo de procesador superescalar.**

En los procesadores superescalares, **cada instrucción requiere al menos un ciclo de reloj para ejecutarse.**

Para mejorarlas prestaciones en procesadores superescalares, se pueden utilizar técnicas como predicción de saltos y renombramiento de registros.

**Procesador Supersegmentado:** Divide las etapas del cauce en partes más pequeñas, permitiendo ejecutar dos o mas tareas en cada ciclo de reloj. Esto permite que mas instrucciones se encuentren en el cauce al mismo tiempo, aumentando el paralelismo.

**Resumen:**

En la segmentación de cauce, cada etapa requiere un ciclo de reloj completo, en la supersegmentacion, cada etapa se divide en subetapas que pueden ejecutarse en fracciones de ciclo. En la supersegmentacion, al duplicar la velocidad del reloj interno, se pueden realizar dos tareas en un ciclo de reloj externo, lo que aumenta ademas, el nivel de velocidad de procesamiento.

**Conclusión y diferencias entre superescalar y supersegmentado.**

**Superescalar**: Diseñado para explotar el paralelismo a nivel de instrucción, ejecutando varias instrucciones al mismo tiempo en diferentes unidades funcionales. Es menos complejo de implementar. Cada instrucción requiere al menos un ciclo de reloj. **Su objetivo principal es maximizar la cantidad de instrucciones procesadas en paralelo.**

**Supersegmentado**: Ademas de explotar el paralelismo, se enfoca en aumentar la velocidad de procesamiento dividiendo cada instrucción en mas etapas de pipeline. Es mas complejo de implementar. Permite realizar dos tareas en un ciclo de reloj. **Su objetivo principal es procesar mas instrucciones por unidad de tiempo.**

**Arquitecturas Multiprocesador:**

Son sistemas informáticos que tienen múltiples unidades de procesamiento (procesadores) trabajando en paralelo para ejecutar tareas y programas. Estos sistemas se utilizan para mejorar el rendimiento y la capacidad de procesamiento de las computadoras, permitiendo que múltiples procesadores compartan la carga de trabajo y trabajen juntos en la resolución de problemas.

***Multiprocesamiento simétrico (SMP):*** Es un tipo de arquitectura de procesamiento en paralelo. En un sistema SMP, Todos los procesadores tienen acceso equitativo a los recursos del sistema. Esto significa que los procesos pueden acceder y compartir datos entre sí de forma sencilla. Son especialmente adecuados para procesamiento de imágenes, análisis de datos, en servidores y estaciones de trabajo de gama alta.

***Multiprocesamiento asimétrico (AMP):*** Tienen procesadores con roles diferentes y niveles de acceso a los recursos del sistema. Por ejemplo, puede haber un procesador principal más poderoso que maneje ciertas tareas críticas mientras que otros procesadores más simples se encargan de tareas menos intensivas en recursos. Los sistemas AMP se utilizan a menudo en dispositivos embebidos y sistemas integrados.

**Clusters:**

Los clusters, tienen arquitectura MIMD con memoria distribuida, son grupos de computadoras completas interconectadas que trabajan en conjunto como un único recurso de cómputo, cada computadora es un nodo.. En otras palabras, un cluster es un conjunto de computadoras independientes que colaboran para realizar una tarea en común, dando la apariencia de ser una única máquina más potente.

**Características y beneficios clave de los clusters:**

●Escalabilidad: Los clusters pueden ser muy grandes, incluso superando el rendimiento de las computadoras individuales más potentes. Se pueden añadir nuevas computadoras a un cluster de forma incremental a medida que sea necesario, lo que permite un crecimiento flexible.

●**Alta disponibilidad:** Si una computadora en un cluster falla, las demás pueden seguir funcionando, lo que minimiza el tiempo de inactividad.

**Arquitecturas de memoria compartida – distribuida**

Acceso no uniforme a memoria con coherencia de cache (CC – NUMA): Cada nodo es un SMP, con sus propios procesadores, mecanismo de e/s y memoria principal compartidas. Desde el punto de vista de los procesadores, comparten un único espacio de direcciones.

**ARQUITECTURAS PARALELO Y TAXONOMIA DE FLYNN**

***Clasificación de arquitecturas paralelo y taxonomía de Flynn***

La taxonomía de Flynn clasifica los sistemas en función de dos características principales: el número de flujos de instrucciones y el número de flujos de datos que se manejan simultáneamente.

***SISD (Single Instruction, Single Data - Instrucción Única, Datos Únicos)***: Un único procesador interpreta una única secuencia de instrucciones para operar con los datos almacenados en una única memoria.

***SIMD (Single Instruction, Multiple Data - Instrucción Única, Múltiples Datos):*** Una única instrucción maquina controla el paso a paso de la ejecución simultanea y sincronizada de elementos del proceso.

Se utiliza en sistemas que realizan operaciones paralelas en grandes conjuntos de datos, como gráficos por computadora, procesamiento de imágenes y algunas aplicaciones científicas.

***MISD (Multiple Instruction, Single Data - Múltiples Instrucciones, Datos Únicos):*** Se transmite una secuencia de datos a un conjunto de procesadores, cada uno de ellos ejecuta una secuencia de instrucciones diferente.

***MIMD (Multiple Instruction, Multiple Data - Múltiples Instrucciones, Múltiples Datos):*** Múltiples procesadores ejecutan simultáneamente una secuencia de instrucciones diferentes con conjuntos de datos diferentes. Los SMP, los clusters y los sistemas NUMA abarcan esta categoria

**MIMD de la taxonomía Flynn:**

En un sistema MIMD, múltiples procesadores independientes ejecutan diferentes instrucciones al mismo tiempo en conjuntos distintos de datos. Cada procesador tiene su propio flujo de control y puede ejecutar instrucciones de manera independiente. Adema sen un sistema MIMD, los procesadores pueden trabajar en conjunto para resolver problemas mas complejos mediante la ejecución simultanea de tareas independiente. Los procesadores en un sistema MIMD son capaces de ejecutar tareas en paralelo.

Los MIMD pueden clasificarse en dos categorías:

**MIMD con memoria Compartida**: Varios procesadores comparten un espacio de memoria en común. Pueden comunicarse en esta memoria compartida mediante la lectura y escritura.

**MIMD con memoria distribuida**: Cada procesador tiene su propia memoria local y no comparte memoria física con otros procesadores. La comunicación se lleva a cabo mediante mensajes u cualquier otro mecanismo de comunicación.

**Procesamiento multihebra(multithreading)**

Técnica que consiste en dividir la secuencia de instrucciones en secuencias mas pequeñas, que pueden ejecutarse en paralelo, para permitir paralelismo elevado sin incrementar la complejidad de los circuitos ni el consumo.

**Preguntas de finales:**

**De que depende el paralelismo de una máquina superescalar?**

**(tomada en final de 27/11/2024)**

El paralelismo a nivel de maquina es una medida de la capacidad del procesador para aprovechar el paralelismo en las instrucciones. El paralelismo de la maquina depende del numero de instrucciones que se pueden captar y ejecutar al mismo tiempo y de la velocidad. Tanto el paralelismo en las instrucciones como el paralelismo de la maquina son factores importantes para mejorar las prestaciones.

**La coherencia de datos de un sistema jerárquico se ve afectada por el uso de DMA? (tomada en final de 27/11/2024)**

El uso de DMA, puede afectar la coherencia de datos en un sistema jerárquico de memoria, especialmente cuando se utiliza una cache.

El problema surge porque el DMA permite a los dispositivos periféricos acceder a la maquina principal directamente sin pasar por la cache del procesador. Esto puede llevar a situaciones donde:

* Un dispositivo DMA modifica un dato en la memoria principal, pero la copia de ese dato en la cache del procesador permanece desactualizada.
* El procesador escribe un dato en la cache, pero el dispositivo DMA lee una versión antigua del dado desde la memoria principal, ya que la cache no se ha actualizado todavía.

Estas inconsistencias pueden resultar en comportamientos inesperados y errores en el programa. Por lo tanto, la coherencia de datos, se PODRIA ver afectada por el uso del DMA.,

**Que es una subrutina?**

Una subrutina es una secuencia de instrucciones, que realiza una tarea especifica, puede ser llamada desde diferentes partes de un programa. Una subrutina puede contener llamadas a otras subrutinas. Las subrutinas pueden recibir parámetros.

Requieren dos instrucciones: CALL (invocarla) y RETURN (retornar de la subrutina), ambas instrucciones son de bifurcación.

Cuando se llama a unas subrutina, el procesador guarda la dirección de retorno, que es utilizada para volver al programa principal una vez que se haya terminado de ejecutar la subrutina.

**Ventajas de las subrutinas:**

Permiten reutilizar código, reduce el tamaño de los programas almacenados en memoria, además las subrutinas aportan modularidad, al descomponer problemas grandes en partes mas pequeñas.

**Pasajes de Argumentos a Subrutinas:**

Los argumentos son usados para la comunicación de datos entre el programa principal y una subrutina.

Los tipos de pasajes de parámetros son:

**Via Registros:** Los argumentos son pasados a través de los registros del procesador.

Estos registros son rapidos de acceder. Este método esta limitado por la cantidad de registros, por lo tanto, es eficiente para un pequeño numero de argumentos.

**Via Memoria:** Los argumentos son almacenados en ubicaciones especificas de la memoria RAM, la subrutina accede a estos valores a través de direcciones de memoria.

Puede ser útil cuando se tienen mas argumentos que los registros, pero el acceso a la memoria es mas lento en comparación con los registros..

**Via Pila (stack):** Los argumentos son apilados antes de invocar a la subrutina, mediante la instrucción de PUSH, y en la subrutina se desapilan para acceder a los valores con la instrucción de POP. Puede ser mas lento debido a las operaciones de apilado y desapilado. La única limitación es tener un buen manejo y entender como trabaja la pila.

**Que es el PIC? Cual es su funcionamiento?**

Se encarga de gestionar las interrupciones, permitiendo que múltiples dispositivos y periféricos compartan líneas de interrupción del procesador. El PIC es el responsable de recibir señales de interrupción de varios dispositivos como periféricos y dirigir estas solicitudes a la CPU. Se puede configurar para asignar prioridades a las interrupciones, es decir, permite establecer niveles de prioridad para las distintas fuentes de interrupción. Además, permite habilitar o deshabilitar (enmascarar) las interrupciones. Cuando una interrupción esta enmascarada, se ignorará. Además el PIC puede operar las interrupciones internamente o en modo cascada, y cuenta con registros internos como son:

IRR: Especifica que interrupciones están pendientes de reconocimiento

ISR: Especifica que interrupciones fueron conocidas y están siendo atendnidas.

EOI: Marca el final de una interrupción. Cuando la CPU ha completado el servicio de interrupción, se envía una señal EOI al PIC.

IMR: Especifica que instrucciones deben ser ignoradas.

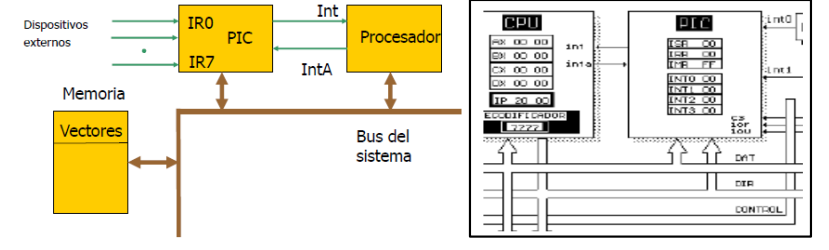
Funcionamiento del PIC

Detección de interrupciones: El PIC monitorea constantemente las líneas de interrupción de los dispositivos periféricos y otros eventos en busca de cambios en su estado. Cuando se detecta una interrupción, el PIC, genera una solicitud de interrupción hacia el procesador.

Priorizacion: El PIC puede configurar y seleccionar aquellas interrupciones según su prioridad.

Enmascaramiento: Antes de notificar al procesador sobre una interrupción, verifica si la línea de interrupción esta habilitada o enmascarada. Si la línea esta enmascarada, la interrupción se ignora.

Esquema del PIC:



**Diferencias en la invocación y finalización de subrutinas y rutinas de interrupción**

Invocación: Las subrutinas se invocan explícitamente mediante una instrucción de llamada. El control del programa pasa a la subrutina y una vez finalizada vuelve al punto donde fue llamada. A una subrutina se le pueden pasar argumentos, pueden ser vía pila, memoria o registros. Las subrutinas generalmente son utilizadas para modularizar el código, facilitando el mantenimiento y legibilidad del programa.

Mientras que, las rutinas de interrupción son activadas en respuesta a eventos específicos, como pueden ser interrupciones de hardware o de software, es decir, no son invocadas por el programador y pueden ocurrir en cualquier tiempo. El control se transfiere a la rutina de interrupción y una vez atendida la interrupción el control vuelve al punto del programa principal donde fue interrumpido. Para que una interrupción sea atendida, debe estar habilitadas las interrupciones o esa interrupción en especifico, caso contrario se ignorara el pedido de interrupción.

En ambos casos, antes de ejecutar una subrutina o una rutina de interrupción, se guarda el contexto del programa, se salva el estado y se guarda el PC que contiene la próxima instrucción a ejecutar.

**Que es el vector de interrupción?**

El vector de interrupción, actúa como un puntero a la ubicación de memoria donde se encuentra la rutina de servicio de interrupción especifica para ese tipo de interrupción.

Cada tipo de interrupción tiene asignado un id que se usa para acceder a la tabla de vectores de interrupción. La tabla contiene 256 sectores de 32 bits.

**Que son las interrupciones vectorizadas?**

Las interrupciones vectorizadas son un mecanismo en el que se utiliza una tabla de vectores de interrupción para gestionar interrupciones. Este mecanismo permite que el sistema maneje multiples tipos de interrupciones de manera eficiente.

Cada interrupción tiene un id que la identifica en la tabla de vectores, esta tabla contiene las direcciones de memoria de las rutinas de manejo de interrupción. Cada entrada en la tabla esta asociada con un numero de interrupción.

Cuando ocurre una interrupción, el procesador utiliza el numero de interrupción para acceder a la tabla y obtener la dirección de la rutina de manejo de interrupción, y luego el procesador salta a esa dirección de memoria y comienza a ejecutar la rutina de manejo de interrupción.

**Describa las limitaciones existentes al paralelismo a nivel de instrucciones**

**Dependencia de datos:** Ocurren cuando una instrucción depende de los resultado de una o mas instrucciones anteriores.

**Dependencia de control:** Se refieren a las instrucciones condicionales, como las instrucciones de salto condicionales, el procesador no puede proveer de antemano con exactitud cual seria la próxima instrucción a ejecutar hasta que se resuelva la condición.

**Limitaciones de ancho de banda de memoria:** El acceso a memoria principal es una operación que puede restringir el paralelismo a nivel de instrucción, si el procesador necesita acceder repetidamente a memoria, es posible que deba esperar a que se completen las operaciones de Lectura/Escritura antes de poder avanzar con las demás instrucciones.

**Control de flujo**

El control de flujo se refiere a las instrucciones que cambian la secuencia de ejecución normal de un programa. Normalmente, las instrucciones se ejecutan secuencialmente en la memoria. Sin embargo, las instrucciones de control de flujo permiten al programa saltar a diferentes partes del código, lo que es esencial para implementar bucles, condicionales y llamadas a procedimientos

**Mecanismos de control de flujo:**

* **Instrucciones de bifurcación (saltos):** Las instrucciones de bifurcación, también conocidas como saltos, hacen que el procesador ejecute una instrucción en una dirección diferente a la siguiente en la secuencia. Un operando de la instrucción de bifurcación especifica la dirección de la siguiente instrucción a ejecutar.

**Hay dos tipos principales de bifurcaciones:**

* + **Saltos Incondicionales:** El salto siempre se produce, independientemente de cualquier condición. **EJEMPLO INSTRUCCIÓN JMP**
  + **Saltos Condicionales:** El salto se produce solo si se cumple una condición específica. **EJEMPLO INSTRUCCIÓN JZ (SALTA SI ES=0)**

- **Saltos Implícitos:** Estas instrucciones implican un salto a una dirección específica sin necesidad de un operando de dirección. Un ejemplo común es la instrucción "incrementar y saltar si es cero" (ISZ).

- **Llamadas a Procedimientos**: Estas instrucciones transfieren el control a un procedimiento (subrutina) y guardan la dirección de retorno para que el programa pueda volver al punto de llamada después de la ejecución del procedimiento.

**Que son los modos de direccionamiento? Para que sirven? Cuales modos de direccionamiento existen?**

El modo de direccionamiento, se refiere al método utilizado para especificar la ubicación de un operando. Un modo de direccionamiento es un esquema especifico para calcular la dirección efectiva de un operando a partir de la información contenida en la instrucción y el estado actual del procesador.

**Algunos modos de direccionamientos mas comunes son:**

●**Inmediato**: El operando está incluido directamente en la instrucción. Este modo es rápido y eficiente para constantes o valores pequeños, pero limita el tamaño del operando.

●**Directo**: La dirección del operando está especificada explícitamente en la instrucción. Es un modo simple pero limita el rango de direcciones que se pueden alcanzar.

●**Indirecto**: La dirección especificada en la instrucción no contiene el operando, sino la dirección de memoria donde se encuentra la dirección del operando. Permite acceder a un rango más amplio de direcciones, pero requiere un acceso adicional a memoria.

●**De Registros:** El operando se encuentra en un registro del procesador. Es el modo más rápido ya que los registros son internos al procesador, pero el número de registros es limitado.

●**Indirecto con Registro:** La dirección del operando se encuentra en un registro del procesador. Similar al indirecto, pero con la dirección almacenada en un registro, lo que reduce un acceso a memoria.

●**Con Desplazamiento:** La dirección efectiva se calcula sumando un desplazamiento a un valor base, que puede ser una dirección o el contenido de un registro. Ofrece gran flexibilidad y se utiliza en varios modos de direccionamiento más específicos, como el relativo, el de registro base y el indexado.

**Describa la técnica de renombramiento de registros en procesadores superescalares:**

El renombramiento de registros es una técnica que permite ejecutar instrucciones en paralelo de manera eficiente. Cuando una instrucción se empieza a ejecutar, se le asigna un registro físico en lugar de uno lógico. Esto permite que múltiples instrucciones utilicen los mismos registros lógicos sin interferencias. Cuando las instrucciones se completan, los resultados se desalojan de los registros físicos y se escriben en los registros lógicos. Este proceso evita conflictos de dependencia de datos, permitiendo una ejecución en paralelo eficiente y de forma independiente.

**Que elementos componen una maquina con arquitectura Von Neumann? Describir la función de cada uno. (tomada en final de 27/11/2024)**

Una maquina con arquitectura Von Neumann se compone de los siguientes elementos:

**CPU (Unidad central de procesamiento):** La CPU es el cerebro de la computadora. Se encarga de ejecutar las instrucciones de los programas, realizar operaciones, controlar el flujo de datos.

**Memoria principal:** Almacena tanto los programas como los datos que la cpu necesita para ejecutarlos. Es volátil. Lo que significa que su contenido se pierde al apagar la computadora. El acceso a la memoria principal es aleatorio.

**Dispositivos de e/s:** Permiten a la computadora interactuar con el mundo exterior. Los dispositivos de e/s, como teclado, mouse, permiten al usuario introducir datos en la computadora. Los dispositivos de salida como el monitor y impresora, muestran los resultados del procesamiento al usuario.

**Bus del sistema:** Es un camino de comunicación entre dos o mas dispositivos. Conjunto de líneas que conectan diferentes componentes de la computadora, permitiendo la transferencia de datos, direcciones e instrucciones entre la CPU,la memoria principal y los dispositivos de e/s.

**De que depende el paralelismo de una máquina superescalar? (tomada en final de 27/11/2024)**

El paralelismo a nivel de maquina es una medida de la capacidad del procesador para aprovechar el paralelismo en las instrucciones. El paralelismo de la maquina depende del numero de instrucciones que se pueden captar y ejecutar al mismo tiempo y de la velocidad. Tanto el paralelismo en las instrucciones como el paralelismo de la maquina son factores importantes para mejorar las prestaciones.

**Que es el DMA? Cual es su funcionamiento?**

El DMA es una técnica de transferencia de datos entre el periférico y hacia o desde la memoria, sin intervención directa de la CPU. Esta llevada a cabo por un controlador de DMA (DMAC), el cual es el responsable de llevar a cabo la transferencia.

El DMA es usado para mejorar la transferencia de datos entre dispositivos periféricos y la memoria principal a través del bus del sistema. Cuando un dispositivo necesita transferir datos a la memoria, el DMA toma el control temporalmente del bus del sistema. Al liberar la CPU de la tarea de gestionar cada transferencia, el DMA reduce la carga del procesador.   
**Existen dos modos de transferencia de DMA:**

Transferencia por bloques: El DMA solicita el control del bus tantas veces como sea necesario hasta finalizar la transferencia del bloque completo.

Transferencia por ráfaga: El DMA no libera el bus hasta haber finalizado la transferencia de todo el bloque de datos.

En general, para transferencias de e/s mas prolongadas, la técnica por bloques es la mas eficiente, ya que permite implementar la transferencia al mismo tiempo que la CPU continua trabajando en otras tarea.

**Que es una pila?**

Una pila es un conjunto ordenado de elementos, en el que solo uno de ellos es accesible en un instante dado. El punto de acceso se denomina cabecera de la pila. El número de elementos en la pila es **variable.** Solo se pueden añadir o eliminar elementos en la cabecera de la pila, por esta razón, se dice que la pila sigue una estructura LIFO.

* Características de una pila:
  + Acceso restringido: Solo se puede acceder al elemento en la cabecera de la pila.
  + Operaciones básicas: PUSH (Apilar/ agrega un elemento a la pila) POP (Desapilar/elimina el elemento de la cabecera de la pila.)
  + La estructura pila, es de longitud variable.

**Cual es la diferencia en la invocación entre un procedimiento y una subrutina?**

La diferencia clave entre la invocación de un procedimiento y una subrutina radica en cómo se maneja la transferencia de control y el intercambio de datos.

Invocación de un Procedimiento:

Los procedimientos **son invocados explícitamente por el programado**r, lo que significa que el programador decide cuándo y cómo llamar al procedimiento.

En un procedimiento, **se pueden pasar parámetros (datos) al procedimiento y este puede devolver un valor al programa principal.**

La pila se utiliza a menudo para manejar la transferencia de control y el paso de parámetros en las llamadas a procedimientos.

Invocación de una Subrutina:

Generalmente son encargadas de manejar fallos/errores durante la ejecución de un programa. Las subrutinas pueden ser llamadas tanto por el programador como implícitamente por el sistema operativo o el hardware. En una subrutina, el intercambio de datos con el programa principal puede ser más limitado o incluso inexistente. El uso de la pila para la gestión de subrutinas depende de la implementación específica.

En resumen, **un procedimiento es una unidad de código modular invocada a propósito** **por el programador** para realizar una tarea específica. **Puede recibir parámetros y devolver un valor, y la pila se utiliza comúnmente para manejar su ejecución**. **Una subrutina, generalmente es invocada para manejar ciertos fallos/errores durante la ejecución de un programa.**

**Diferencias en la invocación y finalización de subrutinas y rutinas de interrupción**

**(parecida a la pregunta anterior, pero ahora nos pide la finalización también.)**

Invocación: Las subrutinas se invocan explícitamente mediante una instrucción de llamada. El control del programa pasa a la subrutina y una vez finalizada vuelve al punto donde fue llamada. A una subrutina se le pueden pasar argumentos, pueden ser via pila, memoria o registros. Las subrutinas generalmente son utilizadas para modularizar el código, facilitando el mantenimiento y legibilidad del programa.

Mientras que, las rutinas de interrupción son activadas en respuesta a eventos específicos, como pueden ser interrupciones de hardware o de software, es decir, no son invocadas por el programador y pueden ocurrir en cualquier tiempo. El control se transfiere a la rutina de interrupción y una vez atendida la interrupción el control vuelve al punto del programa principal donde fue interrumpido. Para que una interrupción sea atendida, debe estar habilitadas las interrupciones o esa interrupción en especifico, caso contrario se ignorara el pedido de interrupción.

En ambos casos, antes de ejecutar una subrutina o una rutina de interrupción, se guarda el contexto del programa, se salva el estado y se guarda el PC que contiene la próxima instrucción a ejecutar.

**En que se basa el paralelismo a nivel de maquina?**

El paralelismo a nivel de máquina se mide por la cantidad de instrucciones que un procesador puede captar y ejecutar simultáneamente. Cuanto mayor sea el paralelismo de la máquina, mayor será su capacidad para aprovechar el paralelismo a nivel de instrucción y, por lo tanto, mayor será su rendimiento.

Factores que influyen en el paralelismo a nivel de máquina:

●Número de cauces paralelos: Un procesador con más cauces paralelos puede ejecutar más instrucciones al mismo tiempo.

●Velocidad y sofisticación de los mecanismos de detección de instrucciones independientes: El procesador debe ser capaz de identificar rápidamente qué instrucciones pueden ejecutarse en paralelo sin violar las dependencias entre ellas.

●Técnicas de hardware como la duplicación de recursos, la emisión desordenada y el renombramiento de registros:

***Cual es la relación entre las interrupciones y el manejo de operaciones de e/s?***

e/s programada por interrupción: El controlador de e/s se configura para emitir una interrupción cuando se complete la operación de e/s o cuando necesite atención. De esta manera, cuando un dispositivo de e/s completa la operación o necesite atención, puede generar una interrupción para notificarle al procesador. De esta manera, el procesador no tiene que esperar activamente, sino que puede atender y realizar otras tareas mientras espera la respuesta del dispositivo.

**Cuales son los modos de ubicación de los módulos de entrada/salida? Descríbalos.**

* **E/S asignada en memoria (Memory-mapped):** En este modo de ubicación, existe un único espacio de direcciones para las posiciones de memoria y dispositivos de E/S. El procesador trata los registros de estado y datos de los modulos de E/S como si fueran posiciones de memoria. Esto permite usar las mismas instrucciones maquina tanto para acceder a la memoria, como a los dispositivos de E/S.
* **E/S aislada:** En este modo de ubicación, los espacios de direcciones para la memoria y los dispositivos de E/S están separados. Esto significa que hay un espacio de direcciones para la memoria principal, y otro espacio de direcciones para los dispositivos de E/S. Se necesitan líneas de lectura y escritura en memoria junto con líneas para órdenes. Las líneas de órdenes especifican si la dirección se refiere a una posición de memoria o a un dispositivo de E/S.

**Que son las bifurcaciones (SALTOS), y cuales existen?**

**(tomada en final de marzo 2024)**

La instrucción de bifurcación, también llamada instrucción de salto, tiene como uno de sus operandos, la dirección de la siguiente instrucción a ejecutar.

Las instrucciones de salto más frecuentes son las de salto condicional, es decir, si se cumple una condición, se efectúa la bifurcación (o el salto), se actualiza el PC (program counter que contiene la dirección de la próxima instrucción a ejecutar), se actualiza con la dirección especificada en el operando. En caso contrario de que no se ejecute la bifurcación, se ejecuta la siguiente instrucción.

Hay dos tipos de saltos:

**Saltos condicionales:** Cambia el flujo de ejecución solo si se cumple una condición especifica. En MIPS, las instrucciones de salto condicional, evalúan el valor de los registros y saltan a una dirección especifica solo si la condición es verdadera.

**Saltos incondicionales:** Este tipo de instrucciones, cambia siempre el flujo de ejecución de un programa a una dirección especifica, sin evaluar ninguna condición.

**Describa las características que diferencian los SMP respecto a los CLUSTERS?**

Los SMP se caracterizan por su arquitectura de memoria compartida, donde multiples procesadores comparten el mismo espacio de memoria y recursos brindando equidad en el acceso a los recursos sin jerarquía. La comunicación entre los procesadores es directa, a través de la memoria compartida.

Los clusters, se componen de nodos independientes interconectados mediante una red, cada uno con su propia memoria y recursos. La comunicación entre nodos se realiza via red, lo que puede generar latencia.

La principal ventaja de un SMP es que resulta mas fácil de gestionar y configurar que un cluster, ademas necesita menos espacio físico y consume menos energía.

**Que es una pila?**

Una pila, es un conjunto ordenado de elementos, en el que solo uno de ellos es accesible en un instante dado, el punto de acceso se denomina cabecera de la pila. El numero de elementos de la pila es variable. Solo se realizan operaciones en la cabecera de la pila, las operaciones son PUSH (apilar) y POP (desapilar).

La pila sigue una estructura LIFO (Ultimo en entrar, primero en salir)

Finales resueltos (Los saque de otro resumen)

Final 1:

1. Describir el mecanismo de interrupción. Mencionar cuales son los tipos de interrupciones. Describir el tratamiento de múltiples interrupciones.

Las interrupciones son un mecanismo que permite alterar el proceso de ejecución normal de la cpu. Una interrupción, permite que la cpu suspenda la tarea que esta ejecutando y responda a una rutina de interrupción.

Mientras la CPU está ejecutando, si recibe un pedido de interrupción, y la interrupción **no se encuentra enmascarada** (es decir, esta habilitada), la cpu suspende la ejecución del programa en curso, guarda su contexto, carga el pc (program counter) que contiene la dirección de la próxima instrucción a ejecutar y salva el registro de estado.

Una vez que guarda todo el contexto de lo que estaba ejecutando antes de ser interrumpido, se atiende dicha interrupción, se ejecutan las rutinas de interrupción correspondientes. Al finalizar la rutina de interrupción, evalúa si hay otra interrupción pendiente, si hay algún pedido de interrupción, y esa interrupción se encuentra habilitada, se debe guardar el contexto del programa y atender la interrupción.

Si no hay ningún pedido de interrupción, el procesador recupera el pc y el registro de estado y continua con su proceso de ejecución.

***Hay dos tipos de interrupciones:***

***Interrupciones por Hardware***: Este tipo de interrupciones son generadas por dispositivos de e/s. Pueden no estar relacionadas con el programa en curso. Se dice que son “asincronicas” porque pueden generarse en cualquier momento. Al momento de atender la interrupción se le pasa el control al vector de interrupción, el cual es el encargado de llevar a cabo las rutinas de interrupción necesarias.

***Interrupciones por Software:*** Este tipo de interrupciones están relacionadas con el programa en curso, pueden generarse por división por cero, overflow en una operación, entre otras cosas. Para atender estas interrupciones se le pasa el control al S.O, el cual es el encargado de llevar a cabo las rutinas de interrupción necesarias.

***Interrupciones internas:*** Son producidas por eventos dentro del procesador, o del sistema en si mismo, algunos ejemplos pueden ser división por cero, intentar acceder a zonas de memoria prohibidas. Este tipo de interrupciones se asocian con las interrupciones de Software.

***Interrupciones externas:*** Son producidas por eventos externos al procesador, por dispositivos de e/s, periféricos y no están relacionadas con el programa en curso. Se asocian con las interrupciones de Hardware.

**Tratamiento de multiples interrupciones:**

Multiples dispositivos pueden interrumpir simultáneamente, por eso se debe tener un manejo correcto y eficiente de las interrupciones. Para el tratamiento de multiples interrupciones hay dos casos:

Primera alternativa: Atender las interrupciones según sean generadas, es decir, se genera como una cola de interrupciones al deshabilitar las interrupciones mientras se atienda una interrupción. Es decir, mientras se atiende una interrupción se deshabilitan las demás interrupciones para evitar que interrumpa el servicio de atención actual, de esta manera, si se genera un pedido de interrupción mientras se atiende otra, quedaría pendiente. Cuando se termine de tratar la interrupción actual, se activarían las interrupciones y se atenderá. Esta alternativa no tiene en cuenta las prioridades de las interrupciones.

Segunda alternativa: Asignarle prioridades a las interrupciones y atenderlas por orden de prioridad. Es decir, una interrupción de mayor prioridad debe ser atendida antes que una de menor prioridad. Las interrupciones de prioridad alta indican eventos críticos que deben ser atendidos lo antes posible. De esta manera , si se esta atendiendo una interrupción de prioridad baja, y llega una interrupción de prioridad alta, se interrumpe la ejecución de la proridad menor y se atiende la interrupción de prioridad mayor.

1. **Como es la estructura de un modulo de e/s?** Describir la funcion del DMA

El modulo de e/s facilita la interaccion entre la CPU y los dispositivos periféricos.

La estructura de un modulo de e/s es la siguiente:

* **Controlador de e/s:** *Se encarga de gestionar la comunicación entre la CPU y los dispositivos periféricos*. Tiene varios registros internos para controlar el flujo de datos.
* **Registro de control**: Estos registros *son usados por la CPU para configurar el funcionamiento del dispositivo periférico.*
* **Buffers de datos:** Los buffers de datos son *áreas de memoria usadas para almacenar temporalmente los datos que se transfieren entre la CPU y el dispositivo periférico.*
* **Interfaz de e/s:** *Proporciona los medios físicos para la conexión entre el controlador de e/s y el periférico*. Pueden ser puertos de e/s, conectores, cables
* **Circuitos de control y temporización**: Son los *encargados de sincronizar las operaciones de e/s y garantizar la integridad de los datos durante la transferencia*.

***Funcion del DMA:***

La función del DMA es transferir los datos entre la memoria principal y los periféricos sin la intervención directa de la CPU. Esta transferencia esta llevada a cabo por un controlador de dma (DMAC). El uso del DMA libera a la CPU de la tarea de gestionar cada transferencia. Por lo tanto, esta técnica reduce la carga del procesador.

Para llevar a cabo cada transferencia, el DMA debe solicitar el uso del bus del sistema.

Hay dos tipos de transferencias DMA:

Transferencia por bloques: El DMAC solicita el uso del bus tantas veces como sea necesario hasta finalizar la transferencia.

Transferencia por ráfaga: El DMAC solicita el uso del bus, y no lo libera hasta finalizar la transferencia.

1. Que es la segmentación de instrucciones, como mejora el rendimiento? Describir tipos de dependencia que afectan el funcionamiento de los cauces.

La segmentación de instrucciones, consiste en dividir el proceso de ejecución de las instrucciones en varias fases o etapas. Cada etapa es ejecutada por una unidad independiente. Esto introduce un nivel de paralelismo a nivel de instrucciones, donde varias instrucciones pueden ejecutarse simultáneamente. Las instrucciones se ejecutan a medida que se liberan unidades, sin la necesidad de esperar que termine una ejecución para ejecutar la siguiente.

La segmentación de instrucciones mejora el rendimiento de los procesadores modernos, ya que puede ser más eficiente y optimizar el rendimiento ejecutar instrucciones paralelamente. Entre las mejoras se encuentran:

*Mejora del tiempo de ejecución:* En aplicaciones que ejecutan un gran numero de instrucciones, es mas eficiente ejecutarlas paralelamente antes que ejecutarlas secuencialmente.

*Mejora de la utilización de recursos:* Permite que diferentes unidades funcionales trabajen independientemente sin tener que esperar que otras instrucciones se completen.

Al estar ejecutando instrucciones paralelamente, pueden producirse conflictos por dependencias que afecten el funcionamiento y el tiempo de ejecución de los causes.

Entre los conflictos de dependencia se encuentran:

*Conflictos por Dependencia de Datos:* Son generados cuando dos o mas instrucciones comparten un mismo dato, y una instrucción necesita de un dato que todavía no esta disponible.

*Conflicto por Dependencia de Control:* Son generados cuando la ejecución de una instrucción depende de cómo se ejecute otra. Un ejemplo es un salto y dos posibles caminos.

*Conflictos Estructurales:* Son generados cuando dos o mas partes del hardware comparten el mismo recurso. Ejemplo : memoria, alu.

Algunas soluciones para estos conflictos pueden ser:

Reordenamiento de instrucciones, forwarding (adelantamiento de operandos),

instrucciones NOP (tener en cuenta que la instrucción nop retrasa un ciclo de reloj.)

**Caracteristicas que posee un procesador superescalar**

Los procesadores superescalares tienen la capacidad de ejecutar multiples instrucciones en paralelo, pueden realizar varias operaciones en cada ciclo de reloj.

Utilizan la segmentación de instrucciones para dividir la ejecución de las instrucciones en varias fases o etapas. Tienen varias unidades funcionales, conocidas como unidades de ejecución. Los procesadores superescalares usan la técnica de renombramiento de registros para evitar conflictos de dependencia de datos, y tienen técnicas avanzadas en predicción de saltos.

Final 2:

1. Describir el pasaje de argumentos a subrutinas:

El pasaje de argumentos a subrutinas se utiliza para comunicar datos entre el programa principal y la subrutina. Las formas de pasajes de parámetros son:

Via Registros: Los registros tienen una ventaja, los registros son de rápido acceso. En este método, los datos se cargan en los registros antes de invocar a la subrutina y dentro de la subrutina acceden a los datos a través de los registros. La única desventaja es que los registros son limitados. Este método puede ser eficiente cuando se tiene una pequeña cantidad de argumentos.

Via Memoria: Los datos se almacenan en direcciones especificas de la memoria RAM. Dentro de la subrutina se accede a las direcciones de memoria para acceder a los datos. Este método es de acceso mas lento que los registros, pero puede ser eficiente cuando se tiene una cantidad mayor de argumentos.

Via Pila (stack): Se considera el verdadero pasaje de parámetros. En este método, los datos son apilados antes de invocar a la subrutina, y dentro de la subrutina se desapilan para acceder a los datos. Existen instrucciones para apilar y desapilar.

Este método puede resultar mas lento que los anteriores, debido a las operaciones de apilado y desapilado.

Describir la técnica de renombramientos de registros

La técnica de renombramiento de registros que permite ejecutar instrucciones en paralelo de manera eficiente. Esta técnica consiste en que, al ejecutar las instrucciones, se le asigna un registro físico en vez de un registro lógico. Esto permite que multiples instrucciones usen los mismos registros lógicos sin interferencias.

Cuando la ejecución de una instrucción se completa, el resultado se desaloja del registro físico y se escribe en el registro lógico. El objetivo de usar la técnica de renombramiento de registros es evitar conflictos por dependencias de datos.

1. Caracteristicas de un bus

Un bus, es un camino de comunicación entre dos o mas dispositivos. Un bus, se usa como medio de transmisión compartido, puede ser transmisión de datos, de direcciones, señales de control, entre diferentes componentes del sistema.

Al bus se conectan varios dispositivos, cualquier señal emitida en el bus, va a estar disponible para todos los dispositivos conectados.

Dado que en un bus, solo un dispositivo puede transmitir en un momento dado con éxito, se necesita un control o un arbitraje. De esta manera, hay dos tipos de arbitrajes:

*Arbitraje Centralizado:* Un dispositivo, conocido como controlador o arbitro del bus, es el encargado de asignar el acceso al bus a los distintos dispositivos.

*Arbitraje Distribuido:* En este tipo de arbitraje, no hay un controlador, sino que cada modulo dispone de la lógica necesaria para controlar y tomar el acceso del bus. En este tipo de arbitraje, los módulos deben trabajar coordinadamente

*A medida que aumentan los dispositivos conectados al bus, el retardo de propagación también aumenta.*

Hay dos tipos de buses:

Buses Sincronicos:

* + La transmisión de datos esta sincronizada por un reloj común para todos los dispositivos conectados al bus.
  + Los datos se transmiten en intervalos determinados por un pulso de reloj
  + Todos los dispositivos conectados al bus, deben operar a la misma frecuencia para evitar problemas de sincronización.

Buses Asincronicos:

* + Cada dispositivo puede operar a su propio ritmo.
  + Los datos se envían de manera independiente, sin la necesidad de depender de una señal de reloj.

1. Que es el MIMD de la Taxonomía de Flynn

En un sistema MIMD, múltiples procesadores independientes, ejecutan instrucciones al mismo tiempo en conjunto distintos de datos. Cada procesador tiene su propio flujo y puede ejecutar instrucciones de manera independiente. Ademas en un sistema MIMD, los procesadores pueden trabajar en conjunto para resolver problemas mas complejos. Los procesadores en un sistema MIMD son capaces de ejecutar tareas en paralelo.

Los sistemas MIMD pueden clasificarse en dos categorías:

***MIMD con memoria compartida:*** Varios procesadores comparten un espacio de memoria en común, y se pueden comunicar a través de esta memoria con operaciones de lectura y escritura.

***MIMD con memoria distribuida:*** Cada procesador tiene su propia memoria local, y no comparten espacio de memoria con otros procesadores. La comunicación se puede dar por mensajes o cualquier otro mecanismo de comunicación.

Final 3:

1. Explicar las formas de direccionamiento de e/s:

El direccionamiento de e/s se refiere a como el procesador accede a los dispositivos periféricos para enviar o recibir datos.

Algunas formas de direccionamiento de e/s son:

***Direccionamiento por Puertos( I / O PORTS):*** Implica asignar un conjunto de direcciones de memoria especificas a los puertos de e/s. Cada puerto se considera una ubicación de memoria única, a la que el procesador puede acceder para enviar o recibir datos desde o hacia dispositivos periféricos.

***Direccionamiento por Mapeo de Memoria:*** Los registros de control y estado de los periféricos se asignan a direcciones de memoria especificas. El procesador utiliza instrucciones de carga y almacenamiento (LOAD / STORE) para interactuar con los periféricos.

***Interrupciones de e/s:*** El procesador utiliza interrupciones para ser notificado de eventos en los dispositivos. Cuando un dispositivo requiere atención del procesador, genera una interrupcion y el procesador interrumpe su ejecución en curso para atender la interrupcion.

***Acceso Directo a Memoria (DMA):*** Permite que los dispositivos periféricos transfieran datos hacia o desde la memoria, sin la necesidad de la intervención directa del procesador. El controlador de DMA (DMAC) es el encargado de llevar a cabo las transferencias, liberando a la CPU de la tarea de gestionar cada transferencia, de esta manera, el DMA reduce la carga del procesador.

**Describa las distintas fuentes de interrupción (tipos de interrupción) que conozca y el tratamiento a realizar cuando hay múltiples interrupciones.**

Las interrupciones son el mecanismo mediante el cual se puede interrumpir el procesamiento normal de la CPU. Estas pueden ser de origen interno o externo a la CPU. Las interrupciones pueden ser:

• Por Hardware: Son las llamadas “verdaderas” interrupciones. Son generadas por dispositivos de E/S (como teclado o mouse) para señalar eventos (como la pulsación de una tecla). El tratamiento implica determinar la fuente de la interrupción, ejecutar la rutina de manejo correspondiente y, si hay múltiples interrupciones de hardware, priorizarlas según su importancia y manejarlas secuencialmente en función de su prioridad.

• Traps/Excepciones: Son interrupciones por hardware creadas por el procesador en respuesta a ciertos eventos como: condiciones excepcionales (overflow en la ALU), falla de programa (tratar de ejecutar una instrucción no definida), fallas de hardware (error de paridad de memoria) o, accesos no alineados o a zonas de memoria restringidas.

• Por Software: Son generadas por instrucciones explícitas que afectan al procesador de la misma manera que las interrupciones por hardware, generalmente usadas para hacer llamadas a funciones del sistema operativo

**Describa las limitaciones existentes al paralelismo a nivel de instrucciones**.

El paralelismo a nivel de instrucciones es una técnica que permite que múltiples instrucciones se ejecuten en paralelo dentro de un procesador. Algunas de sus limitaciones son:

**- Dependencias de Datos:** Las dependencias de datos ocurren cuando una instrucción depende de los resultados de una o más instrucciones anteriores. Esto puede limitar la cantidad de instrucciones que pueden ejecutarse en paralelo, ya que algunas de ellas deben esperar a que se completen las instrucciones precedentes.

**- Dependencias de Control:** Las dependencias de control se refieren a las instrucciones condicionales, como las instrucciones de salto condicional. El procesador no puede prever de antemano cuál será la próxima instrucción a ejecutar hasta que se resuelva la condición.

- **Limitaciones de Ancho de Bande de Memoria:** El acceso a memoria principal es una operación que puede restringir el paralelismo a nivel de instrucción. Si el procesador necesita acceder repetidamente a la memoria, es posible que deba esperar a que se completen las operaciones de lectura/escritura antes de poder avanzar con otras instrucciones.

**- Tamaño de Ventana de Ejecución:** La cantidad máxima de instrucciones que se pueden emitir y ejecutar en paralelo puede ser limitada por cuestiones de diseño de hardware o por la arquitectura del procesador.

**- Saltos y Llamadas a Subrutinas:** Las instrucciones de salto y las llamadas a subrutinas cambian el flujo de ejecución y pueden requerir la cancelación de instrucciones en vuelo que ya no son válidas debido al salto. Esto puede llevar a penalizaciones de tiempo significativas.

**¿Qué ventajas proporciona la implementación de la segmentación de cauce**?

La implementación de la segmentación de cauce proporciona varias ventajas, entre ellas:

o Mejora el rendimiento: La ejecución en paralelo acelera la velocidad de ejecución de las instrucciones. o Aprovechamiento de recursos

◊ Mientras una unidad de ejecución realiza una operación, las etapas anteriores y posteriores pueden estar ocupadas con otras instrucciones, lo que permite un uso más completo de las unidades funcionales.

o Mayor paralelismo : Al permitir que múltiples instrucciones se ejecuten al mismo tiempo, la segmentación de cauce aumenta el nivel de paralelismo a nivel de instrucciones en un procesador.

o Reducción del ciclo de reloj por instrucción: Reduce el tiempo necesario para ejecutar una instrucción completa al dividirla en etapas más pequeñas.

o Mejora de la predicción de saltos: Puede combinarse con técnicas de predicción de saltos para reducir las penalizaciones por fallos de predicción de saltos, lo que resulta en una ejecución más eficiente en las instrucciones

**Explique los atascos producidos por saltos**

Los atascos producidos por saltos son situaciones en las que la ejecución de instrucciones en un cauce segmentado se ve afectado negativamente debido a la presencia de instrucciones de salto condicional o incondicional. Hay tres tipos principales de atascos de control causados por saltos:

• **Salto condicional tomado,** cuando la condición se resuelve como verdadera, lo que significa que se debe tomar el salto. Las instrucciones que siguen al salto condicional y ya han avanzado deben descartarse, lo que conduce a un desperdicio de ciclos de reloj y una penalización en el rendimiento.

• **Salto condicional no tomado,** cuando la condición se resuelve como falsa, lo que significa que no se toma el salto. Las instrucciones que siguen al salto deben continuar su ejecución.

**• Salto incondicional,** ocurre cuando se encuentra una instrucción de salto incondicional. Las instrucciones que siguen ya han avanzado en el cauce y se encuentra en diferentes etapas de ejecución. Cuando se toma el salto, estas instrucciones en etapas posteriores deben ser descartadas, lo que resulta en un desperdicio de ciclos de reloj y una penalización en el rendimiento.

**Describa las características funcionales del Acceso Directo a Memoria (DMA) (Etapas de transferencia).**

El Acceso Directo a Memoria permite que un dispositivo periférico acceda a memoria principal (RAM) directamente sin la intervención de la CPU. El DMA acelera la transferencia de datos entre la memoria y los dispositivos periféricos, lo que libera recursos de la CPU para otras tareas. Las características funcionales del DMA incluyen varias etapas de transferencia:

**- Solicitud:** La primera etapa implica que un dispositivo periférico envíe una solicitud de acceso a la memoria al controlador DMA. Esta solicitud incluye información sobre la dirección de memoria de origen y destino, la cantidad de datos a transferir y el sentido de la transferencia

**. - Selección del Canal DMA:** Los sistemas informáticos pueden tener múltiples canales, cada uno dedicado a un tipo específico de dispositivo o función. En esta etapa, el sistema debe asignar el canal DMA a la solicitud entrante.

**- Configuración:** Una vez seleccionado el canal DMA, se configura para que coincida con los requisitos de la transferencia de datos. Esto incluye la configuración de las direcciones de inicio y finalización en la memoria, el tamaño de la transferencia y otras características relevantes.

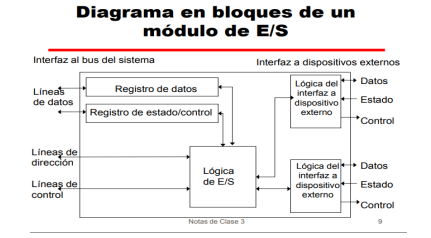
**- Acceso Directo:** El controlador DMA se comunica directamente con la memoria principal y el dispositivo periférico para iniciar la transferencia de datos. La CPU no participa en la transferencia en sí, lo que permite que la CPU realice otras tareas mientras se lleva a cabo la transferencia.

**- Transferencia de Datos:** El controlador DMA transfiere datos entre la memoria y el dispositivo periférico utilizando el canal DMA configurado previamente. La transferencia puede ser en una dirección o bidireccional.

**- Finalización**: Cuando se completa la transferencia, el controlador DMA notifica al dispositivo periférico y actualiza cualquier estado relevante. La CPU puede ser notificada de la finalización de la transferencia mediante una interrupción o un mecanismo similar.

**- Liberación del Canal DMA:** Después de completar la transferencia, el canal DMA se libera para su uso posterior. Esto permite que otros dispositivos o solicitudes utilicen el canal DMA según sea necesario.

**¿Cómo es la estructura de un módulo de E/S? (Esquematice y describa**)



Un módulo de E/S es un componente clave que permite la comunicación entre la CPU y dispositivos periféricos. El funcionamiento de un módulo de E/S permite que el procesador vea una amplia gama de dispositivos de una forma simplificada y oculta los detalles de temporización, formatos y electromecánica propios de los dispositivos para que el procesador pueda funcionar únicamente con órdenes de lectura y escritura o abrir y cerrar fichero.

**Estructura de un módulo de E/S:**

El módulo se conecta al resto del computador a través de un conjunto de líneas (por ejemplo, líneas del bus del sistema). Los datos que se transfieren a y desde el módulo se almacenan temporalmente en uno o más registros de datos. Además, puede haber uno o más registros de estado que proporcionan información del estado presente. Un registro de estado también puede funcionar como un registro de control, para recibir la información de control del procesador. La lógica que hay en el módulo interactúa con el procesador a través de una serie de líneas de control. Éstas son las que utiliza el procesador para proporcionar órdenes al módulo de E/S. El módulo también debe ser capaz de reconocer y generar las direcciones asociadas a los dispositivos que controla. Cada módulo de E/S tiene una dirección única o, si controla más de un dispositivo externo, un conjunto único de direcciones. Por último, el módulo de E/S posee la lógica específica para la interfaz con cada uno de los dispositivos que controla.

**¿La coherencia de datos de un sistema jerárquico de memoria se ve afectado por el uso de DMA?**

La coherencia de datos puede verse afectada por el uso de DMA, ya que si se realizan operaciones sobre un dato, éste se actualizará en caché y si no se vacía en memoria antes de que un dispositivo intente accederlo, se podría estar utilizando un valor erróneo, y viceversa.

**¿Cuáles son los modos de ubicación de los módulos de entrada salida?**

Los módulos de E/S pueden ser ubicados en direcciones asociadas a la E/S mapeada en memoria y la E/S aislada

**– E/S asignada en memoria (memory-mapped):** Los dispositivos de E/S y memoria comparten un único espacio de direcciones. Tendremos una cierta cantidad de direcciones de memoria asignada a las operaciones de E/S, por lo tanto no habrá necesidad de manejar instrucciones distintas

**. - E/S aislada:** Tendremos espacios de direcciones separados, es decir que tendremos direcciones de memoria principal y por otro lado tendremos direcciones asociadas a las E/S. Necesitaremos líneas especiales de E/S y de memoria, y órdenes específicas para E/S (conjunto limitado de instrucciones).

**Analice las políticas de escritura de caché desde el punto de vista de la coherencia de datos.**

**Acierto de caché:** Ocurre cuando el procesador intenta leer una palabra de la memoria y la palabra se encuentra en la caché. En este caso, la palabra se entrega al procesador desde la caché. Lo que es mucho mas rápido que acceder a memoria principal.

**Fallo de caché:** Ocurre cuando el procesador intenta leer una palabra de la memoria y la palabra no se encuentra en la caché. En este caso, se produce un fallo de caché, y se debe acceder a memoria principal para obtener la palabra.

**Politicas de escritura en acierto de caché:**  Determina si los datos se escriben solo en la caché, o tanto en la caché como en la memoria principal.

**Escritura inmediata:** La información se escribe tanto en la caché como en la memoria principal. Esta es la técnica más sencilla. Garantiza que la memoria principal esté siempre actualizada, pero puede generar un mayor tráfico en el bus de memoria.

**Postescritura:**  Las escrituras se realizan solo en la caché y se escribe en la memoria principal solo cuando el bloque se reemplaza en la caché. Esto reduce el tráfico en el bus de memoria y puede mejorar el rendimiento.

**Politicas de escritura en fallo de caché:** Determina si el bloque de memoria principal que contiene la palabra a escribir se carga en la caché o no.

**Escritura y asignación:** Ante un fallo de escritura, el bloque de memoria principal se carga primero en la caché y luego se realiza la escritura en la caché.

**No escritura y asignación:** Escribe los datos directamente en la memoria principal sin cargar el bloque en la caché. Se utiliza para reducir el tráfico en el bus de memoria.

**Describa los elementos a tener en cuenta en el diseño de una memoria caché. Analice ventajas y desventajas de poseer varios niveles de caché.**

Los elementos básicos a tener en cuenta en el diseño de una memoria caché son:

• **El tamaño de la caché :** Es imposible predecir un tamaño óptimo, pero se debe tener en cuenta que: cuanto más grande es, mayor es el número de puertas implicadas en direccionar la caché; cachés grandes tienden a ser ligeramente más lentas; el tamaño de caché está limitado por las superficies disponibles de chip y tarjetas.

**• Función de correspondencia**: Se necesita un algoritmo que haga corresponder bloques de memoria principal a líneas de caché. La elección de la función de correspondencia determina cómo se organiza la caché. Pueden utilizarse tres técnicas:

o Directa: consiste en hacer corresponder cada bloque de memoria principal a sólo una línea posible de caché

o Asociativa: cada bloque de memoria principal puede cargarse en cualquier línea de la caché,

o Asociativa por conjuntos: un bloque puede almacenarse en un conjunto restringido de lugares en la caché.

**• Algoritmo de sustitución** : Una vez que se ha llenado la caché, para introducir un nuevo bloque debe sustituirse uno de los bloques existentes.

Para las correspondencias asociativas se requieren algoritmos de sustitución:

**o LRU (last-recently used):** Se sustituye el bloque que se ha mantenido en la caché por más tiempo sin haber sido referenciado.

**o FIFO (first-in first-out):** Se sustituye el bloque del conjunto que ha estado más tiempo en la caché (el primero que ingresó).

**o LFU (least-frequently used):** Se sustituye aquel bloque del conjunto que ha experimentado menos referencias.

**o Otra alternativa es la sustitución aleatoria, elegir una línea al azar para ser sustituida**.

**• Política de escritura:** Cuando se debe reemplazar un bloque de la caché, si se ha realizado al menos una operación de escritura sobre una palabra de la línea correspondiente, la memoria principal debe actualizarse rescribiendo la línea de caché en el bloque de memoria.

o Escritura inmediata: Todas las operaciones de escritura se hacen tanto en caché como en memoria principal.

o Postescritura: Las actualizaciones se hacen sólo en caché, y se sobrescriben en memoria principal cuando el bloque es sustituido.

• Tamaño de línea : Cuando se recupera y ubica en caché un bloque de datos, se recuperan no sólo la palabra sino algunas adyacentes. Se producen dos efectos:

o Bloques más grandes reducen el número de bloques que caben en la caché. Dado que cada bloque captado se escribe sobre contenidos anteriores de la caché, un número reducido de bloques da lugar a que se sobrescriban datos poco después de haber sido captados.

o A medida que un bloque se hace más grande, cada palabra adicional está más lejos de la requerida y por tanto es más improbable que sea necesaria a corto plazo.

• Número de cachés: Hay dos aspectos relacionados con este tema, el número de niveles de caché, y el uso de caché unificada frente al de cachés separadas.

o Cachés multinivel: En general, el uso de un segundo nivel de caché mejora las prestaciones. No obstante, el uso de cachés multinivel complica todos los aspectos de diseño de la caché.

o Caché unificada frente a cachés separadas: Muchos diseños contenían una sola caché para almacenar las referencias tanto a datos como a instrucciones. Más recientemente, se ha hecho normal separar la caché en dos: una dedicada a instrucciones y otra a datos. La ventaja clave del diseño de una caché partida es que elimina la competición por la caché entre el procesador de instrucciones y la unidad de ejecución. Esto es importante en diseños que cuentan con segmentación de cauce de instrucciones.

**.Justifique el uso de dos niveles de caché**.

El uso de dos niveles de caché (L1 y L2) es una característica clave en la jerarquía de memoria. Algunos motivos para el uso de dos niveles de caché son:

**- Reducción del tiempo de acceso promedio:** Permite el acceso rápido a datos de uso común sin la necesidad de acceder a la memoria principal (RAM). El L1 es el más rápido pero el más pequeño y cercano al núcleo del procesador, mientras que el L2 es más grande pero un poco más lento.

**- Aprovechamiento de la localidad de referencia:** Permite explotar la localidad espacial y temporal, ya que los datos accedidos en el L1 pueden propagarse al L2 si son utilizados repetidamente.

- **Reducción de la latencia de acceso a la memoria principal:** Cuando un dato no está en el L1 pero sí en el L2, aún se puede acceder a él más rápidamente que si tuviera que recuperarse directamente desde la memoria principal. Esto ayuda a reducir la latencia general de acceso a memoria, mejorando el rendimiento del sistema.

**Describa tres características que usted considere las más importantes de las arquitecturas RISC.**

Algunas características de las arquitecturas RISC son:

**- Una instrucción por ciclo:** Se ejecuta una instrucción máquina cada ciclo máquina. Deben ejecutarse más rápido ya que no hay que acceder a la memoria de control de microprograma durante la ejecución de la instrucción.

**- Operaciones registro a registro:** Simplifica el repertorio de instrucciones y por lo tanto la unidad de control. Además, fomenta la optimización del uso de registros. Los registros tienen un costo de acceso rápido.

**- Modos de direccionamiento sencillos:** Esta característica simplifica el repertorio de instrucciones y la unidad de control.

**- Formato de instrucción sencillo:** Se suele usar un formato o unos pocos. La longitud de las instrucciones es fija y alineada en los límites de una palabra. Las posiciones de los campos también son fijas. Los formatos sencillos simplifican la unidad de control.

**Describa las características que diferencian los procesadores RISC respectos a los CISC**.

Las arquitecturas RISC se caracterizan por un conjunto de instrucciones simple, instrucciones de longitud fija y una ejecución más simple, lo que las hace adecuadas para una ejecución más eficiente y predecible.

Las arquitecturas CISC, por otro lado, tienen un conjunto de instrucciones más complejo y versátil, lo que las hace adecuadas para tareas más variadas pero a costa de una mayor complejidad de hardware y menos previsibilidad en el tiempo de ejecución. No existe una clara barrera diferenciadora, muchos diseños incluyen características de ambos criterios. No existe un par de máquinas RISC y CISC directamente comparables, no hay un conjunto de programas de prueba definitivo

**¿De qué depende el paralelismo de una máquina superescalar**?

Las limitaciones del paralelismo son:

**- Dependencia de datos verdadera:** Cuando una instrucción necesita leer datos que son el resultado de una instrucción previa.

**- Dependencia relativa del procesamiento:** Las instrucciones que siguen a una bifurcación tienen una dependencia relativa al procedimiento en esa bifurcación y no pueden ejecutarse hasta que se ejecute el salto.

**- Conflicto en los recursos:** Es una pugna de dos o más instrucciones por el mismo recurso al mismo tiempo.

**- Dependencia de salida:** Se produce cuando dos instrucciones tratan de escribir en un mismo registro.

**- Antidependencia:** Cuando una instrucción intenta escribir datos antes de que otra instrucción previa haya terminado de leer esos mismos datos.

**¿Qué características posee un procesador supersegmentado frente a un superescalar?**

La supersegmentación aprovecha el hecho de que muchas etapas del cauce realizan tareas que requieren menos de medio ciclo de reloj. De este modo, se dobla la velocidad de reloj interna, lo que permite la realización de dos tareas en un ciclo de reloj externo. Por otro lado, un procesador superescalar puede ejecutar instrucciones en diferentes cauces de manera independiente y concurrente.

**Mencione las principales diferencias entre un bus PCI y SCSI**.

**Bus PCI (Peripheral Component Interconnect Interconexión de Componente Periférico):** Es un bus de ancho de banda elevado, independiente del procesador, que se puede utilizar como bus de periféricos o bus para una arquitectura de entreplanta.

**Bus SCSI (Small Computer System Interface - Pequeña interfaz del sistema de cómputo)**: Es una interfaz estándar para la transferencia de datos entre distintos dispositivos del bus de la computadora. Se utiliza para comunicar dispositivos rápidos, como discos CD-ROM, dispositivos de audio y dispositivos de almacenamiento externo de datos. Requiere un controlador de interfaz.

**Tipos de buses, temporización y métodos de arbitraje**

Las líneas de un bus pueden ser:

**- Dedicadas:** Está permanentemente asignada a una función o a un subconjuntos físico de componentes del computador.

**- Multiplexadas:** Utiliza las mismas línea para usos diferentes, por ejemplo se transmiten la información de dirección y datos a través del mismo conjunto de líneas utilizando una línea de control de Dirección Válida. La ventaja es el uso de menos líneas (ahorra espacio y costes). La desventaja es que necesita una circuitería más compleja en cada módulo. Solo un dispositivo puede transmitir con éxito en un momento dado, por lo que se necesitan algún método de arbitraje para hacerlo:

**- Centralizado:** Un único dispositivo hardware (controlador del bus o árbitro) es responsable de asignar tiempos en el bus.

**- Distribuido**: No existe un control central, cada módulo dispone de lógica para controlar el acceso y los módulos actúan conjuntamente para compartir el bus.

• El objetivo de ambos es designar un dispositivo (el procesador o un módulo de E/S) como maestro del bus, el cual puede iniciar una transferencia de datos con otro dispositivo que actúa como esclavo.

***La temporización es la forma en la que se coordinan los eventos del bus.***

**• Sincrónica:** La presencia de un evento en el bus está determinada por un reloj. El bus incluye una línea de reloj a través de la que se transmite una secuencia en la que se alternan intervalos regulares de igual duración a uno y a cero. Un único intervalo a uno seguido de otro a cero se conoce como ciclo de reloj o ciclo de bus y define un intervalo de tiempo unidad (time slot).

**• Asincrónica:** La presencia de un evento en el bus es consecuencia y depende de que se produzca un evento previo. - La temporización sincrónica es más fácil de implementar y comprobar, pero es menos flexible ya que todos los dispositivos deben usar la misma frecuencia de reloj

**Características de los Cluster.**

Se puede definir un Cluster como un grupo de computadores completos interconectados que trabajan conjuntamente como un único recurso de cómputo, creándose la ilusión de que se trata de una sola máquina. Cada computador del Cluster se denomina nodo. Sus principales características son:

**- Escalabilidad absoluta:** Es posible configurar Clusters grandes que incluso superan las prestaciones de los computadores independientes más potentes.

**- Escalabilidad incremental:** Un Cluster se configura de forma que sea posible añadir nuevos sistemas a él en ampliaciones sucesivas.

**- Alta disponibilidad:** Puesto que cada nodo es un computador autónomo, el fallo de uno de los nodos no significa la pérdida del servicio.

**- Mejor relación precio-prestaciones:** al utilizar elementos estandarizados, es posible configurar un Cluster con mayor o igual potencia de cómputo que un computador independiente mayor, a mucho menos costo.