Resumen y Preguntas de finales 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.

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: Es el cerebro de la computadora, se encarga de ejecutar instrucciones. Se compone de una UC (unidad de control), la cual es la encargada de dirigir el flujo de instrucciones, y además se compone de una ALU (unidad aritmético-logica), la cual es la encargada de llevar a cabo las operaciones, por ejemplo operaciones aritméticas, lógicas.

Memoria principal: El objetivo de la memoria principal es almacenar datos e instrucciones de los programas que la CPU necesita para ejecutarlos. Es de tipo volátil, lo que significa que su contenido se pierde una vez que se apaga la computadora.

Dispositivos de Entrada/Salida: Los dispositivos de E/S, permite a la computadora interactuar con el mundo exterior, por ejemplo, están los dispositivos de entrada, como mouses o teclados, que permiten ingresar datos a la computadora. También están los dispositivos de salida, como los monitores e impresoras, que muestran el resultado de procesamiento al usuario.

Bus del sistema: Está formado por un conjunto de líneas que conecta diferentes componentes de la computadora, es un camino de comunicación entre dos o mas dispositivos. El bus del sistema, conecta los principales componentes de una computadora que son: El procesador, la memoria y los dispositivos de entrada/salida.

El bus del sistema se divide en:

- Bus de datos: Proporciona un camino de transferencia de datos entre la CPU, la memoria y los dispositivos de e/s.
- Bus de direcciones: Indica la ubicación de memoria los dispositivos a acceder.
- Bus de control: Envía señales de sincronización y coordinación entre los componentes.

Describir el funcionamiento de un sistema basado en microprocesador

Un sistema basado en microprocesador funciona ejecutando un programa almacenado en la memoria, que consisten en una secuencia de instrucciones, las cuales son ejecutadas por la CPU. Esto se lleva a cabo mediante la repetición de la captación y ejecución de instrucciones, mejor conocido como 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).

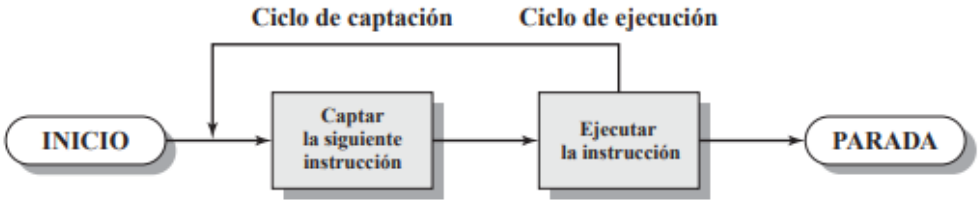


Figura 3.3. Ciclo de instrucción básico.

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.

Que es un bus? Mencione los tipos de buses, y cuáles son los aspectos claves para el diseño de los buses? Describa la jerarquía de buses y su objetivo. Describa funciones y características del Bus PCI.

Un bus, es un conjunto de líneas que conecta varios componentes de la computadora. Es un camino de comunicación entre dos o mas dispositivos. Al bus, se conectan varios dispositivos como periféricos, memoria y CPU, y cualquier señal emitida por uno de los dispositivos, estará disponible para los demás dispositivos conectados al bus. Si dos dispositivos transmiten durante el mismo periodo de tiempo, sus señales pueden distorsionarse, por lo que, en un instante dado, solo un dispositivo puede transmitir con éxito a través del bus.

Tipos de buses:

- **Bus de datos:** Proporciona un camino para la transferencia de datos entre los modulos del sistema. Cuenta con una cantidad de líneas, cada línea transmite de a un bit a la vez, por ende, la cantidad de líneas, afecta a las prestaciones del sistema. A mayor cantidad de líneas, mayor cantidad de bits se pueden transmitir simultáneamente.
- **Bus de direcciones:** Indica la ubicación de memoria de los dispositivos a acceder.
- **Bus de control:** Envía señales de sincronización y coordinación entre los componentes

Los aspectos claves para el diseño de los buses son:

Tipo	**Anchura del bus**
Dedicado	Dirección
Multiplexado	Datos
Método de arbitraje	**Tipo de transferencia de datos**
Centralizado	Lectura
Distribuido	Escritura
Temporización	Lectura-modificación-escritura
Síncrono	Lectura-después de-escritura
Asíncrono	Bloque

Tipo de línea:

- Dedicadas: Controla varios dispositivos, pero de a uno a la vez. Se encarga de un dispositivo y función en particular. Por ejemplo puede usarse para transferencia de datos, en un determinado periodo de tiempo.

- Multiplexadas: Controla varios dispositivos, incluso simultáneamente, las líneas multiplexadas pueden tener varios dispositivos y funciones asignados, por ejemplo, el mismo conjunto de líneas puede usarse para la transferencia de datos, y direcciones al mismo tiempo.

Metodo de arbitraje: Controlan el acceso y el permiso al bus. Generalmente a un bus se conectan varios dispositivos, y como un dispositivo puede transmitir con éxito en un momento dado, es necesario tener un método de arbitraje.

- Arbitraje centralizado: Hay un dispositivo de hardware, conocido como controlador o arbitro del bus, el cual es el responsable de controlar el acceso y permiso del bus a los diferentes módulos del sistema.
-
- Arbitraje distribuido: No hay un controlador, sino que cada modulo dispone de la lógica necesaria para controlar el acceso y el permiso al bus. En esta técnica, los modulos deben actuar conjuntamente para compartirse el acceso del bus.

Temporización: Indica la forma en la que se generan los eventos en el bus.

- Síncrona: La presencia de un evento, depende de una señal de reloj. Hay una línea física conectada al bus, por la cual se mandan intervalos entre cero y uno.
- Asíncrona: No hay un reloj, sino que, la presencia de un evento, depende de que se haya generado un evento previamente.
-

Anchura del bus:

La anchura del bus, afecta a las prestaciones del sistema, cuanto más ancho es el bus, mayor cantidad de líneas se encuentran, por ende, se pueden transmitir más bits simultáneamente. Dado que cada línea puede transmitir 1 bit, si incrementamos la cantidad de líneas, incrementamos la velocidad de transferencia al poder transmitir más bits simultáneamente.

Jerarquia de buses:

La jerarquía de buses, permite otorgar caminos de comunicación especializados dependiendo de cada dispositivo y velocidades. Ya que si se conecta un gran número de dispositivos a un bus, sus prestaciones pueden disminuir. Dentro de una computadora, hay distintos tipos de buses, por ejemplo el bus del sistema, es el encargado de comunicar los componentes esenciales como son la CPU, la memoria y los dispositivos de e/s. Mientras puede haber otros buses que conecten dispositivos de alta velocidad al procesador o la memoria, de esta manera se forma una jerarquía de buses.

Bus PCI:
El **bus PCI,** es un bus de **ancho de banda elevado**, **independiente del procesador**, que permite conectar dispositivos periféricos a la placa madre de una computadora. Proporciona una interfaz de alta velocidad, permitiendo la transferencia de datos entre periféricos y la CPU o memoria, de manera rápida.
Caracteristicas:

- El bus PCI, utiliza un esquema de arbitraje centralizado, temporización síncrona, y un conjunto de líneas multiplexadas para transmitir distintos tipos de información en el mismo conjunto de líneas.
- Alto ancho de banda: Permite una transferencia de datos rápida, lo que lo hace adecuado para subsistemas de e/s de alta velocidad.
- Facil implementación: Se implementa con pocos circuitos integrados.

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.

Describa el análisis del conjunto de instrucciones. Cuales son los elementos esenciales de las instrucciones?

El análisis del conjunto de instrucciones, implica examinar las características de las instrucciones maquina, incluyendo el código de operación (codop), los operandos y los modos de direccionamiento. Desde el punto de vista del diseñador, el conjunto de instrucciones constituye la especificación o requisitos del procesador. Ademas, el repertorio de instrucciones, es el medio que tiene el programador para controlar el procesador.

Elementos esenciales de las instrucciones:

- Codigo de operación: Especifica la operación que se va a realizar, por ejemplo, operaciones aritméticas,lógicas, de transferencia de datos,etc.
- Referencia a operando origen: Especifica lo/los datos que sirven de entrada para la instrucción.
- Referencia a operando destino: Especifica donde se almacena el dato. Cabe aclarar que, no todas las instrucciones producen un resultado, pero aquellas que lo producen, deben tener especificado en donde almacenarlo.
- Referencia a la siguiente instrucción: Le indica al procesador cual es la siguiente instrucción a ejecutar. Se almacena en el registro PC (program counter)

RISC VS CISC: **Describa y diferencie las arquitecturas RISC de las CISC**

RISC	CISC
Esta arquitectura tiene repertorio de instrucciones reducido. Se componen de instrucciones simples, que se ejecutan en un ciclo de reloj. Posee un gran numero de registro Las operaciones son de registro a registro. Hace énfasis en segmentación de instrucciones para ejecutar multiples instrucciones en paralelo.	Esta arquitectura posee un repertorio de instrucciones amplio. Se componen de instrucciones complejas, que pueden tardar mas de un ciclo de reloj Posee gran variedad de modos de direccionamiento. Es utilizado en lenguajes de alto nivel. Hace énfasis en la ejecución secuencial de instrucciones.
Ventajas: Se requiere hardware más simple. Los ciclos de instrucciones son más cortos. Desventajas: Se necesitan más instrucciones para realizar tareas complejas.	Ventajas: Los programas tienden a ser mas cortos debido a sus operaciones complejas Desventajas: Se requiere hardware mas costoso. Tiene ciclos de instrucciones mas largos.

Interrupciones

Que es una interrupción? Que tipos de interrupciones existen? Como es el tratamiento de múltiples interrupciones? Por que son necesarias las interrupciones?

Las interrupciones son un mecanismo que permite alterar el proceso de ejecución normal de la CPU. Una interrupcion permite que el procesador suspenda la tarea que esta realizando y responda a una rutina de atención. Mejorar la eficiencia del procesador, al permitir que se enfoque en su ejecución de tareas y responsa solo cuando se genere un evento. Las interrupciones, indican al procesador que debe atender a un evento,

Mientras la CPU está ejecutando instrucciones, si recibe un pedido de interrupción, y esa interrupción se encuentra habilitada, se suspende la ejecución del programa actual, se salva el contexto (guardando el registro de estado y el PC), y se carga el PC con la dirección de la rutina de manejo de interrupción, para poder atender la interrupción.

Una vez que la CPU termina de atender la interrupción, recupera el PC y el registro de estado, y continua con su ejecución normal.

Interrupciones ENMASCARABLES y NO ENMASCARABLES

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.

Las interrupciones Enmascarables**:** Son aquellas que pueden ser ignoradas temporalmente o pueden descartarse, 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.

Existen tres tipos de interrupciones

- Interrupciones por software: Sirven para realizar llamadas a rutinas del Sistema Operativo, se invocan mediante la instrucción INT, la cual ejecuta subrutinas del Sistema Operativo.

- Traps: Son generados por la ejecución del programa actual, como por ejemplo puede ser ejecutar una instrucción inexistente, overflow en una operación,dividir por cero, intentar acceder a un espacio de memoria fuera del permitido. Este tipo de interrupciones son sincronicas, porque se relacionan con el programa en curso.

- Interrupciones de hardware: Son generadas por eventos externos al procesador, por ejemplo por dispositivos de e/s. Este tipo de interrupciones son asincrónicas, porque no están relacionadas con el programa actual. El PIC es el encargado de responder a estas interrupciones.

En las interrupciones por hardware, el que interrumpe es un dispositivo y la CPU pasa el control a una subrutina especial llamada manejador de interrupciones. En las interrupciones por software, el que interrumpe es el mismo programa, y la CPU pasa el control al sistema operativo, que se encarga de ejecutar una subrutina de sistema.

Por que son necesarias las interrupciones?

Las interrupciones son necesarias, porque mejoran la eficiencia del procesador, permite que el procesador ejecute tareas normalmente, y solo responda cuando se genere un pedido de interrupción. Además, sin interrupciones, el procesador tendría que emplear técnicas como el polling, donde debería estar revisando continuamente el estado de los dispositivos, esto resulta ineficiente, porque consume ciclos de reloj, incluso cuando no hay interrupciones pendientes.

Ciclo de interrupción:

- Se comprueba si se ha solicitado alguna interrupción.
- Si no hay señal de interrupción, se capta la siguiente instrucción.
- Si hay algún pedido de interrupción pendiente:
    - Suspende la ejecución del programa en curso
    - Guarda su contexto (próxima instrucción a ejecutar y el estado del procesador)
    - Carga el PC con la dirección de comienzo de una rutina de gestión de interrupción.
    - Finalizada la rutina de gestión, el procesador retorna la ejecución hacia el punto 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.

¿A que método de atención lo conocemos como de interrupciones vectorizadas?

En el método de interrupciones vectorizadas, cuando se produce una interrupción, se utiliza un vector de interrupción para direccionar directamente la ejecución del procesador a la ubicación de memoria de la rutina de manejo de interrupciones correspondiente. Cada interrupción está asociada con un id. Se utiliza el número de identificación de la interrupción como índice para buscar en la tabla de vectores, luego el procesador salta directamente a esa dirección de memoria y comienza a ejecutar la rutina de manejo de la interrupción.

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

Que es un modulo de entrada/salida? Cual es su estructura? Cuales son sus funciones?

Un modulo de entrada /salida, es un componente esencial en una computadora. Actúa como intermediario entre la CPU, la memoria, y los dispositivos de e/s. Su función principal es controlar uno o mas dispositivos, gestionando el flujo de datos y control entre ellos. Permite al procesador que interactuar con una amplia variedad de dispositivos de manera simplificada, es capaz de generar espacios de direcciones asociadas a los dispositivos que controla.

Estructura de un modulo de entrada/salida:

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

Estructura de un módulo de E/S: El módulo se conecta a la computadora 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 Por último, el módulo de E/S posee la lógica específica para realizar una interfaz con cada uno de los dispositivos que controla.

Funciones de un modulo de entrada/salida:

Además de actuar como intermediario entre los componentes mencionados anteriormente, el modulo de entrada/salida, debe ser capaz de controlar y coordinar el flujo de datos entre los componentes, dado que la velocidad entre los componentes es distinta, el modulo de e/s debe encargarse de adaptar las velocidades y convertir formatos para garantizar una transferencia optima. Además, supervisa el estado de los periféricos, informando cambios en sus estados.

Cuales son las técnicas de entrada/salida? Que modos de direccionamientos de entrada/salida existen? Mencionelos.

Tecnicas de Entrada/Salida:

- Entrada/Salida con espera de respuesta: En esta técnica, el procesador emite una solicitud de operación de entrada/salida, **y espera activamente hasta que el dispositivo de entrada/salida complete la operación** y le avise al procesador que ha finalizado la operación. **Durante este periodo de espera, el procesador permanece inactivo**, ya que se dedica a verificar constantemente el estado de la operación de e/s. Una vez que el procesador recibe la señal de finalización de la operación, puede seguir ejecutando normalmente. **Esta técnica es ineficiente para operaciones prolongadas, ya que el procesador quedaría inactivo durante largos periodos de tiempo, esperando a la finalización de la operación de entrada/salida.**

- Entrada/Salida por interrupción: En esta técnica, antes de iniciar la operación de entrada/salida, el controlador de e/s, se configura para emitir una interrupción cuando la operación de entrada/salida se complete. De esta manera, el procesador puede seguir ejecutando otras tareas, mientras espera que se complete la operación, y no queda inactivo.. Una vez que la operación se completa, se genera la interrupción, y el procesador suspende la ejecución del programa en curso, salva su contexto, guardando el PC y el registro de estado, y atiende la interrupción. Después de manejar la interrupción, el procesador retorna con las tareas que venía ejecutando anteriormente. **Esta técnica permite al procesador, continuar con su funcionamiento normal, mientras se realiza la operación de entrada/salida, en esta técnica, el procesador no permanece inactivo durante ningún periodo de tiempo.**

- Acceso directo a memoria (DMA): Es un mecanismo que permite la transferencia de datos entre un periférico hacia o desde la memoria sin la intervención directa del procesador. Su objetivo es lograr una transferencia de datos lo más rápida posible.
  El DMAC (controlador de DMA) es el encargado de gestionar la transferencia, el procesador solo interactúa al principio y al final de la transferencia, mientras que el DMAC es el responsable de controlar la operación intermedia. Al liberar al procesador de la tarea de gestionar cada transferencia, la técnica del DMA, reduce la carga del procesador, permitiendo al procesador ejecutar otras tareas mientras se lleva a cabo la transferencia. Una vez finalizada la transferencia, el DMAC genera una interrupcion para notificarle al procesador que ha finalizado la transferencia.

Como se conectan los periféricos?

Generalmente a través de un puerto. Un puerto es una interfaz entre el periférico y el modulo de e/s. Existen dos tipos de puertos.

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.

Modos de direccionamiento de entrada/salida:

- Direccionamiento por puertos (I/O PORTS): Implica asignar un conjunto de direcciones especificas a los puertos de entrada/salida. De esta manera, cada puerto, se considera una ubicación de memoria única, a la cual el procesador puede acceder para enviar o recibir datos desde o hacia dispositivos periféricos. No se requieren instrucciones adicionales.

- Direccionamiento mapeado en memoria: Los registros de control y estado de los periféricos, son asignados a direcciones de memoria especificas. El procesador utiliza instrucciones de carga y almacenamiento (LOAD/STORE) para interactuar con los periféricos.

Cuál es la mejor técnica de entrada/salida? (suelen preguntarlo en finales)

Depende, depende del periférico que sea, de la cantidad de datos que vaya a transferir.

Por ejemplo, si se está utilizando un teclado, la técnica adecuada es e/s por interrupcion, de esta manera, cuando se presiona una tecla, el teclado genera una interrupcion y el procesador lee el código de la tecla. Con el mouse pasa lo mismo, es conveniente la e/s por interrupcion, ya que los movimientos del mouse y los clics generan interrupciones.

En general, para dispositivos de velocidad baja, como teclados y mouses, la técnica de e/s con interrupcion es eficiente. Pero para dispositivos de alta velocidad que requieran transferencias de grandes volúmenes de datos, el DMA es la mejor opción.

Si es un periférico rápido, la técnica de DMA va a ser la más adecuada.

Que es el DMA? Cual es su funcionamiento? Cuales son las etapas de transferencia DMA?

El DMA es un mecanismo que permite la transferencia de datos entre un periférico hacia o desde la memoria sin la intervención directa del procesador. Su objetivo es lograr una transferencia de datos lo más rápida posible. El DMAC (controlador de DMA) es el encargado de gestionar la transferencia, el procesador solo interactúa al principio y al final de la transferencia, mientras que el DMAC es el responsable de controlar la operación intermedia. Al liberar al procesador de la tarea de gestionar cada transferencia, la técnica del DMA, reduce la carga del procesador, permitiendo al procesador ejecutar otras tareas mientras se lleva a cabo la transferencia. Una vez finalizada la transferencia, el DMAC genera una interrupcion para notificarle al procesador que ha finalizado la transferencia.

Existen dos modos de transferencia DMA:

- **Transferencia por Bloque:** El DMAC solicita el acceso al bus, cuando el procesador le otorga el acceso, el DMAC, realiza la transferencia de una palabra, y luego libera el bus. En esta técnica, el DMAC**,** solicita el control del bus tantas veces como sea necesario, hasta finalizar la transferencia del bloque completo. Ventaja: No degrada el rendimiento del sistema, Desventaja: La transferencia suelen ser mas lenta.

- **Transferencia por Ráfaga:** El DMAC solicita el acceso al bus, cuando el procesador le otorga el acceso, el DMAC realiza la transferencia y no libera el bus hasta haber finalizado la transferencia. Ventajas: La transferencia se hace de manera rápida, Desventaja: Se degrada el rendimiento del sistema, ya que el procesador durante el periodo de transferencia no puede acceder a memoria.

Etapas de una transferencia DMA:

- **Inicialización de transferencia:** La CPU debe enviar los parámetros de transferencia hacia el DMAC y la interfaz del periférico.
    - Se debe configurar el DMAC:
        - Numero de bytes
        - Tipo de transferencia (Lectura/Escritura)
        - Direccion de memoria inicial para la transferencia
        - Numero de canal de DMA
- **Realizacion de la transferencia:**
    - Cuando el periférico está listo para realizar la transferencia , se lo indica al DMAC.
    - El DMAC pide el control del bus y realiza la transferencia entre el periférico y la memoria.
- **Finalización de la transferencia:** El DMAC libera el bus y devuelve el control a la CPU. El DMAC, suele activar una señal de interrupción para indicar a la CPU la finalización de la e/s.

Memoria:

Cuales son los métodos de acceso a memoria?

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.

Describa la memoria principal, la memoria secundaria y el concepto de memoria virtual

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.

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.

Cual es el objetivo de la jerarquía de memoria?

El objetivo de la jerarquía de memoria, es abordar las principales limitaciones para el diseño de una memoria, que son la velocidad, la capacidad y el costo por bit. Es deseable tener una memoria de gran capacidad, velocidad y bajo coste. Sin embargo es difícil construir una memoria que cumpla con los tres requisitos. La jerarquía de memoria establece que a medida que se ascienden los niveles, disminuye la capacidad, disminuye el costo de acceso y aumenta el costo por bit.

Memoria cache:

Que es una memoria cache? Cual es su objetivo? Cuáles son los elementos para el diseño de cache? Describa las políticas de escritura en acierto y en fallo de cache.

El objetivo de la memoria cache, es lograr que la velocidad de la memoria sea lo más rápida posible. La cache contiene 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 se encuentra en la cache, si es así, se entrega la palabra al procesador desde la cache, lo cual es rápido, si la palabra no se encuentra en la cache, se debe acceder a memoria principal, un bloque de memoria principal se transfiere a la cache y después la palabra es entregada al procesador, este método es más costoso.

La cache esta organizada en bloques **(también puede llamarse líneas de cache).** Un bloque representa la cantidad de datos transferidos entre la memoria principal y la cache en un solo acceso.

La cache se basa en dos principios de localidad:

- Localidad referencial: Establece que los programas suelen acceder repetidamente a las mismas posiciones de memoria, o los mismos datos durante un periodo corto de tiempo. Este principio, establece que si un dato se accede, es muy probable que se vuelva a acceder en un futuro no tan lejano.

- 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 para el diseño de la cache:

Función de correspondencia de cache:

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 sustitución 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.

Si se pretende mejorar el tiempo de acceso medio a memoria cache. ¿Sobre qué parámetros será necesario trabajar y propone como medida para hacerlo?
Para mejorar el tiempo de acceso a la memoria caché es importante trabajar en varios parámetros y considerar diferentes medidas:

1. Tamaño de la caché (capacidad) → Aumentar el tamaño de la memoria caché permite que más datos se almacenen en ella, lo que reduce la necesidad de acceder a la memoria principal con frecuencia.
2. Política de reemplazo → Utilizar una política de reemplazo adecuada puede ayudar a mantener los datos más relevantes en la caché y reducir los fallos de caché.
3. Localidad espacial y temporal → Optimizar los algoritmos y estructuras de datos utilizados en el software para aprovechar la localidad espacial y la localidad temporal.

Entre otros…

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.

Que es la segmentación de cauce? Cuanto mejora el rendimiento?

La segmentación de cauce es una técnica que está orientada a organizar el hardware de la CPU para optimizar su rendimiento, permite realizar más de una tarea al mismo tiempo. El objetivo de utilizar la segmentación de cauce, es logar una mayor cantidad de operaciones en el mismo tiempo. Consiste en dividir el proceso de ejecución de las instrucciones en varias fases o etapas que permitan una ejecución simultánea. Esta técnica explota el paralelismo entre las instrucciones de flujo secuencial. Es una técnica propia de las arquitecturas RISC.

Con respecto a cuanto mejora el rendimiento:

Teniendo en cuenta el el máximo rendimiento sin penalizaciones, seria tener CPI =1, lo que significa una instrucción por ciclo. El incremento potencial de la segmentación de cauce, es proporcional al número de etapas del cauce. Con la segmentación de cauce logramos productividad por que en un ciclo de reloj vamos a tener varios etapas diferentes de ejecución de instrucciones, pero no necesariamente va a reducir el tiempo de ejecución de la instrucción.

Podemos decir que la mejora se puede expresar con la siguiente formula. teniendo en cuenta que es el rendimiento máximo:

Si K es el numero de etapas del cauce: Velocidad procesador segmentado * Vel secuencial * K

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

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

- Mejora el rendimiento → La ejecución en paralelo acelera la velocidad de ejecución de las instrucciones.
- 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.
- 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.
- Reducción del ciclo de reloj por instrucción → Reduce el tiempo necesario para ejecutar una instrucción completa al dividirla en etapas más pequeñas.

Que es la 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.

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.

Que es el 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. 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.

Que es el paralelismo a nivel de maquina? De que pedende el paralelismo a nivel de maquina?

El paralelismo a nivel de maquina, es una medida de la capacidad del procesador para aprovechar el paralelismo a nivel de instrucciones. El paralelismo a nivel de maquina, depende de la cantidad de instrucciones independientes que puedan captarse y ejecutarse al mismo tiempo y de mecanismos para localizar instrucciones independientes.

Describa los procesadores superescalares, supersegmentados, y diferéncielos.

Un procesador superescalar: Se caracteriza por el uso de múltiples cauces de instrucciones independientes, cada cauce consta con multiples etapas, lo que permite ejecutar varias instrucciones a la vez. Cada instrucción requiere al menos un ciclo de reloj para ejecutarse. Para mejorar las prestaciones de estos procesadores, se pueden utilizar técnicas como renombramiento de registros y predicción de saltos. Se enfoca en el paralelismo a nivel de instrucciones, ya que ejecuta multiples instrucciones en paralelo.

Un procesador supersegmentado: Muchas operaciones no necesitan un ciclo de reloj completo. Los procesadores supersegmentados dividen las etapas del cauce en varias sub-etapas más pequeñas y por lo tanto mas rápidas. De esta manera, permite realizar dos operaciones en un ciclo de reloj. Este tipo de procesadores aumenta el grado de paralelismo.

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 más etapas de pipeline. Es más complejo de implementar. Permite realizar dos tareas en un ciclo de reloj. **Su objetivo principal es procesar mas instrucciones por unidad de tiempo.**

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.

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.

Funcionamiento de un Cluster.

El funcionamiento de un Cluster implica la coordinación y utilización de múltiples nodos o computadoras interconectadas para trabajar juntas como una única entidad de procesamiento. Para gestionar y coordinar las operaciones de los nodos se utiliza software de gestión de Clusters, que supervisa el estado de los nodos, administra las tareas y permite la comunicación entre ellos.

Las tareas o cargas de trabajo se distribuyen entre los nodos del Cluster. Esto puede hacerse de diferentes maneras, dependiendo de la aplicación y del software utilizado. Implementa estrategias de balance de cargas que distribuyen las tareas de manera uniforme entre los nodos disponibles.

Los nodos del Cluster se comunican entre sí a través de la red de interconexión. Esto puede incluir la transferencia de datos, mensajes de control o la coordinación de tareas.

Incorporan mecanismos de tolerancia de fallos para garantizar la disponibilidad continua de los servicios. Si un nodo falla, las tareas pueden reasignarse a otros nodos en funcionamiento para evitar la interrupción del servicio.

1. Describa las características que diferencian los SMP respecto a los Cluster.

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

Por otro lado, 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 a través de la red, lo que puede introducir latencia. Los Clusters ofrecen escalabilidad horizontal al permitir la adición de nodos para aumentar la capacidad de procesamiento, adecuándose a aplicaciones distribuidas.

La principal ventaja de un SMP es que resulta más fácil de gestionar y configurar que un Cluster, además necesita menos espacio físico y consume menos energía, y son plataformas estables y bien establecidas.

Los Clusters son superiores en términos de escalabilidad absoluta e incremental, y además también son superiores en términos de disponibilidad, puesto que todos los componentes del sistema pueden hacerse altamente redundantes.

Compare los sistemas MP y Cluster.

Los sistemas MP utilizan memoria compartida al igual que los SMP, pero pueden asignar roles específicos a los procesadores, introduciendo variabilidad en la equidad de acceso a los recursos. La comunicación entre procesadores puede ser más compleja debido a responsabilidades específicas de cada procesador.
Por otro lado, 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 a través de la red, lo que puede introducir latencia. Los Clusters ofrecen escalabilidad horizontal al permitir la adición de nodos para aumentar la capacidad de procesamiento, adecuándose a aplicaciones distribuidas.

Compare las políticas de emisión de instrucciones en procesadores superescalares..

Políticas de emisión de instrucciones:

- Emisión en orden y finalización en orden → Se emiten las instrucciones en el orden exacto en que lo haría una ejecución secuencial, y se escriben los resultado en ese mismo orden.
- Emisión en orden y finalización desordenada → Las instrucciones se pueden completar en cualquier orden una vez que han sido ejecutadas, siempre y cuando no altere el resultado final. La emisión se para cuando hay cualquier tipo de conflicto o dependencia de datos.
- Emisión desordenada y finalización desordenada → Las instrucciones se emiten a la etapa de ejecución en el orden en que están listas para ser ejecutadas, y se pueden completar en cualquier orden, siempre y cuando no altere el resultado final.

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

Que es el MIMD de la Taxonomia de Flynn? (Respuesta mas completa)

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. Los SMP abarcan esta categoría.

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. Los Clusters abarcan esta categoría.

Preguntas de finales:

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 memoria 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.

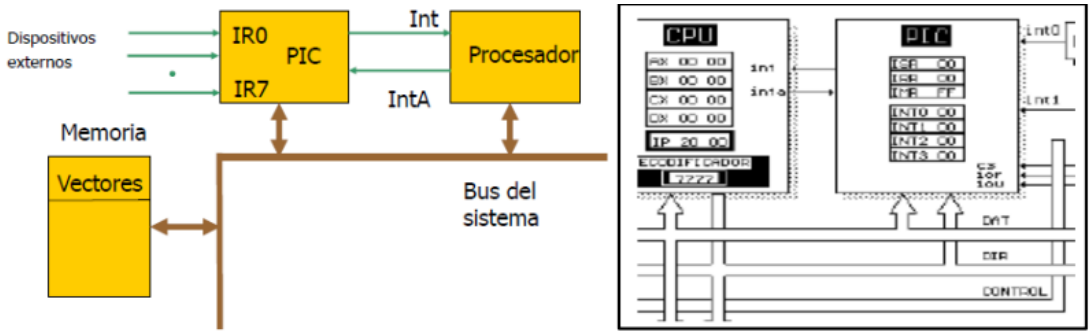
Funcionamiento del PIC

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.

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 pila. El número de elementos de la pila es variable. Las operaciones solo pueden realizarse sobre la cabecera de la pila. La pila sigue una estructura LIFO (Last In, First Out), es decir, el último elemento en entrar, es el primer elemento en salir.

La pila cuenta con operaciones básicas como:

- Agregar elemento: Mediante la instrucción PUSH, se apila un elemento.
- Eliminar un elemento: Mediante la instrucción POP, se desapila un elemento.

ATASCOS / PENALIZACIONES EN CAUCES SEGMENTADOS Y CONTROL DE FLUJO:

Describa las dependencias de los datos que pueden afectar un cauce segmentado.

Los tipos de dependencias de datos son:

- Read After Write (RAW): Ocurre cuando una instrucción en el cauce necesita leer un registro o dato que está siendo escrito por una instrucción anterior en el mismo cauce.
- Write After Read (WAR): Ocurre cuando una instrucción en el cauce necesita escribir un registro o dato que está siendo leído por una instrucción anterior en el mismo cauce. A veces se denominan “antidependencias”.
- Write After Write (WAW): Ocurre cuando dos instrucciones en el cauce necesitan escribir en el mismo registro o dato. A veces se denominan “dependencias de salida”.

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 el problema y posibles soluciones ante riesgos por transferencia de control de programa.

Los riesgos por transferencia de control de programa se refiere a situaciones en las que la ejecución de instrucciones en un procesador segmentado se ve afectada negativamente debido a saltos de programa, como instrucciones de salto condicional o incondicional. Estos riesgos pueden dar lugar a retrasos en la ejecución de instrucciones y pueden afectar el rendimiento general del procesador.

Los riesgos por transferencia de control pueden ser:

- Salto condicional tomado
- Salto condicional no tomado
- Salto incondicional

Las posibles soluciones:

- Adelantar la resolución de los saltos a la etapa de decodificación
    - En ella se decodifican y se sabe que es un salto
    - Se puede evaluar la condición de salto (con restador)
    - Se puede calcular la dirección de salto (con sumador)
- Para tratamiento de saltos hay:
    - Técnica Hardware → Predicción de saltos:
        - Técnicas estáticas:
            - Predecir que nunca se salta: Asume que el salto no se producirá y siempre capta la siguiente instrucción.
            - Predecir que siempre se salta: Asume que el salto se producirá y siempre capta la instrucción destino del salto.
        - Técnicas dinámicas:
            - Conmutador saltar/no saltar: Basado en la historia de las instrucciones. Es eficaz en bucles.
            - Tabla de historia de saltos (Branch-target buffer): Pequeña cache asociada a la etapa de búsqueda (F). Tiene 3 campos, dirección de una instrucción de bifurcación, información de la instrucción destino y N bits de estado (historia de uso).
        - Predecir según el código de operación:
            - Hay instrucciones con más probabilidades de saltar.
        - Precaptar el destino de salto:
            - Se precapta la instrucción destino del salto, además de las instrucciones siguientes a la bifurcación. La instrucción se guarda hasta que se ejecute la instrucción de bifurcación.

- Técnica Software → Salto retardado o de relleno de ranura de retardo:
    - El compilador introduce instrucciones que se ejecutarán en cualquier caso después de la instrucción de salto, y de no ser posible se utilizan instrucciones NOP.
    - Requiere reordenar las 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, permiten que el procesador ejecute una instrucción en una dirección diferente a la siguiente en la secuencia. Un operando de la instrucción de bifurcación específica 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 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.

Una instruccion de salto, permite que el procesador ejecute una instruccion diferente a la secuencia normal, es decir, permite que el procesador salte a una direccion de memoria (que esta especificada en el operando del salto) y pueda ejecutar instrucciones desde esa direccion.

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 específica, sin evaluar ninguna condición.

Describa problemas y posibles soluciones ante riesgos por transferencia de control de Programa

En la segmentación de cauce, los riesgos por transferencia de control de programa, principalmente son causados por saltos condicionales. Estos saltos pueden interrumpir el flujo
continuo de instrucciones a través del cauce, generando la necesidad de descartar instrucciones que ya están en proceso y cargar nuevas, lo que disminuye el rendimiento.
Problema: Hasta que una instrucción de salto condicional se ejecuta, no se puede determinar
si el salto ocurrirá o no. Esto significa que el procesador puede haber precargado incorrectamente instrucciones que luego deben ser descartadas, generando un retraso o penalización en el rendimiento del cauce.
Soluciones: Técnicas para prevenir la penalización en saltos condicionales.
Predicción de saltos: El procesador intenta predecir si un salto se tomará o no. Si la predicción
es correcta, el cauce continua operando sin interrupción, si es incorrecta, se produce un retraso para corregir el flujo de instrucciones. Esta técnica se puede mejorar utilizando bits de
historia, para conocer todo el historial de saltos y poder predecir, pero vamos a necesitar un buffer para almacenar la historia de los saltos.
Precaptar el destino del salto: Cuando se identifica un salto condicional, se precarga la

instrucción destino del salto, además de la siguiente a la de salto. De esta manera, si se produce el salto, el destino ya habrá sido precaptado.
Salto retardado: Se retrasa el salto. De esta manera, la instrucción siguiente del salto se ejecuta siempre, independientemente si se tomará o no.

Describa como se debe implementar la estructura pila de un procesador tipo RISC, cuyos registros son genéricos (basarse en el procesador MIPS). Como se debería trabajar anidamientos de procedimientos/funciones?

(quiero aclarar que esta respuesta no se si aplica para MIPS, creo que así se trabaja en MSX)

En un procesador RISC como MIPS, la pila se implementa en la memoria. Está gestionada por el registro SP (stack pointer), el cual apunta a la cabecera de la pila. Este registro se debe actualizar a medida que apilemos y desapilemos. La pila en MIPS, tiene la particularidad que crece hacia abajo, debemos tener en cuenta esto para realizar las operaciones. Siempre que apilemos y desapilemos debemos actualizar el SP (nuestra cabecera de pila.)

Como realizar operaciones sobre la pila en MIPS:

- Agregar elemento (instrucción PUSH): Antes de hacer el push, se debe decrementar el valor del SP, antes de almacenar el valor, y luego hacer el PUSH para apilar el dato.
- Eliminar elemento (instrucción POP): Se debe leer el valor del SP, y luego incrementar el SP, para que apunte al nuevo ultimo elemento.

Con respecto a como trabajar con anidamientos de prodecimientos/funciones, cada vez que se hace un CALL, es decir que se llama a una subrutina, se debe apilar la dirección de retorno, se hace teniendo en cuenta el manejo de la pila en MIPS, se debe actualizar el puntero de la pila. Al momento de finalizar la subrutina, cuando se ejecute la instrucción RET, se debe desapilar la dirección de retorno de la pila. Por ende, si dentro de la subrutina se apilan registros o datos adicionales, es fundamental desapilarlos antes de retornar de la subrutina, debemos dejar el SP en la dirección de retorno de la subrutina.

Si tuviéramos multiples anidamientos de subrutinas, debemos manejarlas de la misma forma, apilando y desapilando, pero es importante tener en cuenta que una operación de mas o una operación de menos, puede terminar en una ubicación de memoria distinta a la que debería ser, por eso, se debe tener un buen manejo de la pila e ir actualizando el SP.

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.
- Solucion: Reordenamiento de codigo , o Forwarding (adelantamiento de operandos)

- 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.
- Solucion:

  Tecnica de software=> Salto retardado, tecnica de hardware=> prediccion de saltos.

- 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.

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.

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.

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.

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.**