**PREG FINALES - Febrero y Marzo**

**1)**

**A) ¿Qué métodos para pasaje de argumentos podemos utilizar en una computadora?**

El pasaje de parámetros de puede realizar a través de:

●Vía registros: en este caso el número de registros es la principal limitación. Además es importante documentar que registros se usan.

Normalmente, un procedimiento típico usa solo unos pocos parámetros, y la profundidad de activación de procedimientos también está dentro de un rango relativamente pequeño. Para explotar esas propiedades se usan múltiples conjuntos pequeños de registros, cada uno asignado a un procedimiento distinto. Una llamada a un procedimiento hace que el procesador conmute automáticamente a una ventana de registro distinta de tamaño fijo en lugar de salvaguardar los registros en memoria. Las ventanas de procedimientos adyacentes están parcialmente solapadas para permitir el paso de parámetros.  
La ventana se divide en tres áreas de tamaño fijo. Los registros de parámetros contienen parámetros pasados al procedimiento actual desde el procedimiento que lo llamó, y los resultados a devolver a este.

las arquitecturas RISC se implementan con un gran banco de registros, se utiliza la técnica de ventana de registros

●Vía memoria: aquí se usa un área definida de memoria (RAM). Difícil de estandarizar

●Vía pila (stack): es el método más ampliamente usado, considerado el verdadero “pasaje de parámetros”. Es independiente de memoria y registros. Hay que comprender bien cómo funciona porque la pila (stack) es usada por el usuario y por el sistema.

El manejo de la pila para anidamiento de subrutinas es el siguiente:  
Antes de llamar a la subrutina se debe apilar los parametros a pasar y la direccion de retorno, al llamar a la subrutina de debe:  
1.Salvar el estado de BP (viejo BP), es decir apilar el valor del BP  
2.Cargar el valor del BP el del SP  
3.Reservar espacio para datos locales, si los hay  
4.Salvar valores de otros registros, si es que se van a modificar los registros  
5.Acceder a parámetros, para acceder a los parámetros se le debe sumar un desplazamiento al BP para acceder  
a la posicion de la pila en la que están los parámetros. En general el desplazamiento es: 2 (es el tamaño de BP apilado) + tamaño de dirección de retorno + total de tamaño de parámetros entre el buscado y BP  
Aquí se puede llamar a otra subrutina o regresar a la anterior, si se vuelve a hacer una llamada se debe apilar los parámetros a pasar a la subrutina y la dirección de retorno. La subrutina llamada debe repetir los pasos anteriores más los que siguen. Sino se hace otra llamada a subrutina también se deberá seguir los siguientes  
pasos:  
7.Retornar parámetro, si es tiene que retornar datos  
8.Regresar correctamente del procedimiento(desapilar todo lo que apilo, volver a cargar a cargar el valor del BP que tenía antes de entrar a la subrutina

**B) ¿Cuáles son las diferencias en la terminación de una subrutina y un gestor de interrupción?**

**C) ¿Cuáles son las diferencias en la invocacion de una subrutina y un gestor de interrupción?**

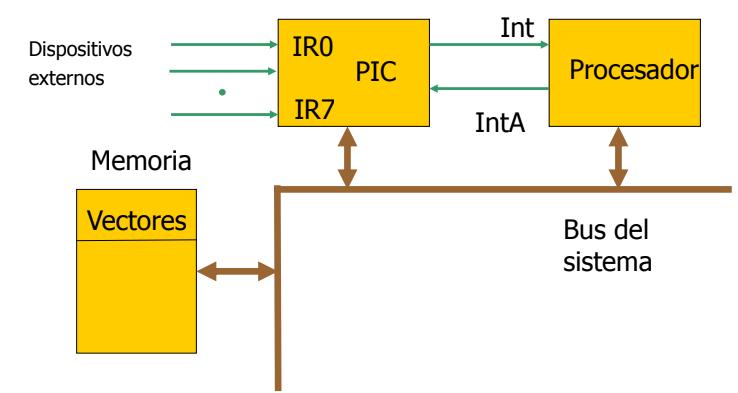
**2)**

**A) Esquematice y describa la estructura interna de un Controlador Programable de Interrupciones.**

El dispositivo controlador programable de interrupciones (PIC) es un chip que sirve si el procesador tiene una única entrada de pedido de interrupciones o si tenemos varios productores de interrupciones.

Es un dispositivo usado para combinar varias fuentes de interrupciones sobre una o más líneas del CPU. Se maneja con prioridades (las de número mas bajo se atienden primero) y tiene un conjunto de registros internos:

* **IRR** (Interrupt Request Register): Registro de petición de interrupción, indica con bit en 1 las interrupciones demandadas hasta el momento.
* **ISR** (In Service Register): Registro de interrupción en servicio, indica con bit en 1 cuál es la interrupción que está siendo atendida.
* **IMR** (Interrupt Mask Register): Registro de maścara de interrupciones, permite el enmascaramiento selectivo de cada una de las entradas de interrupción, indicando con bit en 1. Tras un reset los bits de este registro quedarán en 0. Indica cuáles deben ser ignoradas.
* **EOI** (End of Interruption): Fin de interrupción. Como consecuencia, se pone en 0 el bit del ISR correspondiente.
* **INT0...INT7**: 8 registros, donde carga el valor del vector de interrupción correspondiente



Tareas realizadas por el PIC:

* Puesto que existen muchos dispositivos que pueden solicitar interrupciones, el PIC debe priorizarlas cuando existen varias IRQ’s simultáneas
* Después de enviar una solicitud de interrupción, debe enviar un número de interrupción (número de vector) cuando el procesador indica que está listo para atender la petición
* Mantiene un registro de que se está procesando una interrupción: cuando esto sucede, no envía más peticiones al procesador hasta que este le responde con una señal de EOI (End Of Interrupt), indicando que la rutina de servicio precedente ha terminado o puede aceptar otra interrupción
* Puede enmascarar de forma selectiva cualquiera de las 8 IRQ’s que tiene conectadas

**B) Describa cómo funciona la gestión de E/S programada con espera de respuesta.**

**Técnicas de gestión de E/S** (son 3: E/S programada con espera de respuesta, E/S con interrupciones y E/S con acceso directo a memoria (DMA) ):

• E/S programada con espera de respuesta:

Los datos se intercambian entre el procesador y el módulo de E/S. El procesador ejecuta un

programa que controla directamente la operación de E/S incluyendo: Comprobación de estado del dispositivo, el envío de una orden de lectura o escritura y la transferencia del dato.

Cuando el procesador envía una orden al módulo de E/S, debe esperar hasta que la operación de E/S concluya. Si el procesador es más rápido que el módulo de E/S, el procesador desperdicia este tiempo (permanece ociosa, algo no deseable).

Resumen: Cuando el procesador está ejecutando un programa y encuentra una instrucción

relacionada con una E/S, ejecuta dicha instrucción mandando una orden al módulo de E/S

apropiado. El módulo realiza la acción solicitada y después activa los bits apropiados en el

registro de estado, de E/S. El módulo no interrumpe al procesador, sino que éste es

responsable de comprobar periódicamente el estado del módulo de E/S hasta que encuentra que la operación ha terminado.

Desventaja: El procesador tiene que esperar un tiempo considerable a que el módulo de E/S esté preparado. Espera comprobando repetidamente su estado, degradando el nivel de

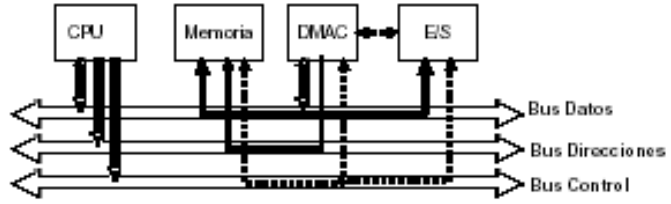
prestaciones de todo el sistema.

**C) Esquematice y describa la estructura interna de Acceso directo a memoria.**

**Funcionamiento del DMA**

El DMA (Direct Memory Access) requiere un módulo adicional en el bus del sistema. El módulo DMA es capaz de imitar al procesador y de recibir el control del sistema cedido por el procesador. Necesita dicho control para transferir datos a y desde memoria a través del bus del sistema. Para hacerlo, el DMA debe usar el bus sólo cuando el procesador no lo necesita o debe forzar al procesador a que suspenda temporalmente su funcionamiento (técnica denominada robo de ciclo o cycle stealing).

Básicamente es capaz de controlar una transferencia de datos entre un periférico y memoria sin intervención de la CPU.



Cuando el procesador desea leer o escribir un bloque de datos, envía una orden al módulo de DMA con la información:

* Si se solicita una lectura o una escritura, usando la línea de control
* La dirección del dispositivo de E/S en cuestión, usando la línea de datos
* La posición inicial de memoria a partir de donde se lee o escribe, usando la línea de datos y almacenada por el DMA en su registro de direcciones
* El número de palabras a leer o escribir, usando la línea de datos y almacenando en el registro de cuenta de datos

Después el procesador continúa con otro trabajo, habiendo delegado la operación de E/S al módulo de DMA. El DMA transfiere el bloque completo de datos, palabra a palabra, directamente desde o hacia memoria sin que pase por el procesador. Cuando la transferencia terminó, el módulo de DMA envía una señal de interrupción al procesador. Así el procesador solo interviene al comienzo y al final de la transferencia.

Nota: No es una interrupción, el procesador no guarda el contexto, sino que espera durante un ciclo de bus. Eso hace que el procesador sea más lento ejecutando programas, aunque para una transferencia de E/S de varias palabras el DMA es mucho mas eficiente que la E/S mediante interrupciones o programada.

**3)**

**A) ¿Cuáles son los elementos a tener en cuenta para el diseño de una memoria Cache?**

Los principales elementos a tener en cuenta son 6:

* **Tamaño de caché**: Se necesita hacer un balance entre el tamaño (mientras más grande es más costoso) y la performance, que lo ideal es tener mucho tamaño, pero si son muy grandes tienden a ser ligeramente más lentas. También está limitado por la superficie disponible de chip y de tarjeta.
* **Función de correspondencia**: Son directa, asociativa y asociativa por conjunto.

-Directa es la correspondencia más sencilla. Consiste en hacer corresponder cada bloque de la memoria principal a solo una línea posible de la cache.

i = j mod m; donde i es el número de línea de caché, j el número de bloque de memoria principal, m el número de líneas de la caché.

La ventaja es que es muy sencilla de implementar. La desventaja es que hay una posición concreta de chache para cada bloque dado. Por ello, si un programa referencia repetidas veces a palabras de los bloques diferentes asignados en la misma línea dichos bloquea se estarían intercambiando continuamente en la caché, y la tasa de aciertos sería baja.

-La correspondencia asociativa supera la desventaja de la directa, permitiendo que cada bloque de memoria principal pueda cargarse en cualquier línea de la cache. La lógica de control de la cache interpreta una dirección de memoria simplemente como una etiqueta y un campo de palabra. El campo de etiqueta identifica inequívocamente un bloque de memoria principal. Para determinar si un bloque está en la cache, su lógica de control debe examinar simultáneamente todas las etiquetas de líneas para buscar una coincidencia. La desventaja es la compleja circuitería necesaria para examinar en paralelo las

etiquetas de todas las líneas de cache.

-La correspondencia asociativa por conjuntos intenta recoger lo positivo de las dos técnicas anteriores. La cache se divide en v conjuntos, cada uno de k líneas. Las relaciones que se tienen son:

m = v\*k

i=j mod v;

i = número de conjunto de cache

j = número de bloque de memoria principal

m = número de líneas de la cache

* **Algoritmos de sustitución**: LRU; FIFO; LFU, aleatorio
* **Política de escritura:**

-Escritura inmediata: todas las operaciones de escritura se hacen tanto en la caché como en la memoria principal, para evitar inconsistencias.

-Post-escritura: La información sólo se actualiza en la caché. Se marca como actualizada bit de “sucio”.

La memoria principal se actualiza en el reemplazo y puede contener información errónea en algún momento.

* **Tamaño de línea**: A medida que aumenta el tamaño del bloque, la tasa de aciertos primero aumenta debido al principio de localidad, al aumentar el tamaño del bloque, más datos útiles son llevados a la memoria caché. Sin embargo, la tasa de aciertos comenzará a decrecer cuando el tamaño de bloque se haga aún mayor. Hay dos efectos: bloques más grandes reducen el número de bloques que caben en la

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

* **Número de cachés:** Con el aumento de densidad de integración ha sido posible tener una cache en el mismo chip del procesador. Esto reduce la actividad del bus externo del procesador y por lo tanto reduce los tiempos de ejecución, e incrementa las prestaciones globales del sistema. Los diseños más actuales incluyen tanto cache on-chip como internos. La estructura más sencilla se denomina cache de

dos niveles, siendo la cache interna el nivel 1 (L1) y la externa el nivel 2 (L2). En general, el uso de cache multinivel mejora las prestaciones, pero aumenta la complejidad del diseño.

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

si se ve afectada.

En una estructura en la que más de un procesador tiene un a cache y la memoria principal es compartida, si se modifican los datos de una cache, se invalida no solamente la palabra correspondiente de memoria principal, sino también la misma palabra en otras caches. Un sistema que evite este problema se dice que mantiene la  
coherencia de caché. Entre las posibles aproximaciones a la coherencia de cache se incluyen:  
 - Vigilancia del bus con escritura inmediata: cada controlador de cache monitoriza las líneas de direcciones para detectar operaciones de escritura en memoria, por parte de otros maestros en el bus.  
Si otro maestro escribe en una posición de memoria compartida que también reside en la memoria cache, el controlador de cache invalida el elemento de la cache. Esta estrategia depende del uso de una política de escritura inmediata por parte de todos los controladores de cache.  
 - Transparencia hardware: se utiliza hardware adicional para asegurar que todas las actualizaciones de memoria principal vía cache quedan reflejadas en todas las cache  
 - Memoria excluida de cache: solo una porción de memoria principal se comparte con más de un procesador y esta se diseña como no transferible a cache. En un sistema de este tipo todos los accesos a la memoria compartida son fallos de cache porque la memoria compartida nunca se copia a cache.

**4)**

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

Un procesador superescalar es aquel que usa múltiples cauces de instrucciones  
independientes. Cada cauce consta de múltiples etapas, de modo que puede tratar varias  
instrucciones a la vez. El hecho de que haya varios cauces introduce un nuevo nivel de  
paralelismo, permitiendo que varios flujos de instrucciones se procesen simultáneamente. Un procesador superescalar saca provecho de lo que se conoce como “paralelismo a nivel de instrucciones” que hace referencia al grado en que las instrucciones de un programa pueden ejecutarse en paralelo.  
Un procesador superescalar capta varias instrucciones a la vez y a continuación, intenta  
encontrar instrucciones cercanas que sean independientes entre sí y puedan, por consiguiente, ejecutarse en paralelo. Si la entrada de una instrucción depende de la salida de una instrucción precedente, la segunda instrucción no puede completar su ejecución al mismo tiempo ni antes que la primera, por lo que al identificar estas dependencias, el procesador puede emitir y completar instrucciones en un orden diferente al del código máquina original.  
El procesador puede eliminar algunas dependencias innecesarias mediante el uso de registros adicionales y el renombramiento de las referencias a registros en el código original. Para maximizar la utilización del cauce de instrucciones y aumentar el rendimiento no usa saltos retardados (como los RISC) sino que hacen predicción de saltos.  
Las instrucciones comunes (aritmética entera y de coma flotante, cargas, almacenamientos y bifurcaciones condicionales) pueden iniciar su ejecución simultáneamente y ejecutarse de  
manera independiente (en diferentes cauces).  
El enfoque superescalar conlleva a la duplicación de algunas o todas las partes de la  
CPU/ALU:  
 • Captar múltiples instrucciones al mismo tiempo  
 • Ejecutar sumas y multiplicaciones simultáneamente  
 • Ejecutar carga/almacenamiento, mientras se lleva a cabo una operación en ALU  
El grado de paralelismo y la aceleración de la máquina aumentan, ya que se ejecutan más  
instrucciones en paralelo.

**B) ¿Cuál es el objetivo de usar la técnica de Renombre de Registros en un procesador superescalar?**

Cuando se utilizan técnicas de desordenación los valores de los registros no pueden conocerse completamente en cada instante de tiempo. Las instrucciones entran en conflicto por el uso de registros y el procesador debe detener alguna etapa para resolverlo.

- Las técnicas de software de optimización de registros empeoran la situación.

- Los efectos de las dependencias de salida pueden disminuirse por esta técnica, que consiste en disponer de registros adicionales (internos, ocultos al programador) y asignarlos (por hardware) a instrucciones en conflicto.

- Lo usual es tener duplicado el banco de registros

**5)**

**A) ¿Qué elementos característicos definen un bus?**

Un bus es un camino de comunicación entre dos o más dispositivos. Es un medio de  
transmisión compartido: al bus se conectan varios dispositivos y cualquier señal transmitida por  
uno de esos dispositivos está disponible para que los otros dispositivos conectados al bus  
puedan acceder a ella. Solo un dispositivo puede transmitir con éxito en un momento dado  
(para que sus señales no se solapen y se distorsionen).  
Un bus está constituido por varios caminos de comunicación o líneas. Cada línea es capaz de  
transmitir señales binarias. En un intervalo de tiempo, se puede transmitir una secuencia de  
dígitos binarios a través de una única línea. Se pueden usar varias líneas del bus para  
transmitir dígitos binarios simultáneamente (Por ej: un dato de 8 bits puede transmitirse  
mediante ocho líneas del bus).  
Una computadora tiene distintos tipos de buses que proporcionan comunicación entre sus  
componentes. El bus que conecta los componentes principales (procesador, memoria y E/S) se  
llama bus del sistem.  
**Tipos de buses**Las líneas del bus se pueden dividir en dos tipos genéricos:  
• Dedicadas: está permanentemente asignada a una función o a un subconjunto físico de  
componentes del computador. Ej: el uso de líneas separadas para direcciones y para  
datos.  
 o 16 líneas de direcciones  
 o 16 líneas de datos  
 o 1 línea de control de lectura o escritura  
• Multiplexadas: Uso de las mismas líneas para usos diferentes. Ventaja: uso de menos  
líneas, lo cual ahorra espacio y costes. Desventaja: Necesita una circuitería más  
compleja en cada módulo, además de que pueden reducirse las prestaciones debido a  
que los eventos que deben compartir las mismas líneas no pueden producirse en  
paralelo.  
 o 16 líneas de direcciones ó datos  
 o 1 línea de control de lectura o escritura  
 o 1 línea de control para definir direcciones o datos  
**Método de arbitraje**Más de un módulo puede necesitar el control de un bus y como solo una unidad puede  
transmitir a través del bus en un momento dado, se requiere algún método de arbitraje:  
• Centralizados: Un único dispositivo hardware (llamado controlador del bus o árbitro) es  
responsable de asignar tiempos en el bus. El dispositivo puede estar en un módulo  
separado o ser parte del procesador.  
• Distribuidos: No existe un controlador central, sino que cada módulo dispone de lógica  
para controlar el acceso y los módulos actúan conjuntamente para compartir el bus.  
En ambos métodos, el propósito es designar un dispositivo, el procesador o un módulo de E/S  
como maestro del bus. El maestro puede iniciar una transferencia de datos con otro dispositivo  
que actúa como esclavo en este intercambio concreto.  
**Técnicas de sincronización/temporización**La temporización es la forma en la que se coordinan los eventos en el bus. Puede ser:  
• Temporización síncrona:  
 o La presencia de un evento en el bus está determinada por un reloj.  
 o El bus incluye una línea de reloj a través de la cual se transmite una secuencia  
en la que se alternan intervalos regulares de igual duración.  
 o Un intervalo (de un 1 seguido de un 0) se conoce como ciclo de reloj o ciclo de  
bus y define un intervalo de tiempo unidad.  
 o Todos los dispositivos del bus pueden leer la línea de reloj y todos los eventos  
empiezan al principio del ciclo de reloj (suelen sincronizar en el flanco de  
subida)  
 o La mayoría de los eventos se prolongan durante un único ciclo de reloj  
• Temporización asíncrona:  
 o La presencia de un evento en el bus es consecuencia y depende de que se  
produzca un evento previo  
La síncrona es más fácil de implementar y comprobar, pero es menos flexible que la  
temporización asíncrona, ya que todos los dispositivos de un bus síncrono deben utilizar la  
misma frecuencia de reloj y el sistema no puede aprovechar mejoras en las prestaciones de los  
dispositivos. Con la asíncrona, pueden compartir el bus una mezcla de dispositivos lentos y  
rápidos.  
**Diferencias entre PCI y SCSI**El 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 para una arquitectura de entreplanta. Comparado con otras  
especificaciones comunes de bus, proporciona mejores prestaciones para los subsistemas de  
E/S de alta velocidad (ej: los adaptadores de pantalla gráfica, los controladores de interfaz de  
red, los controladores de disco, etc). El PCI ha sido diseñado específicamente para ajustarse  
económicamente a los requisitos de E/S de los sistemas actuales; se implementa con muy  
pocos circuitos integrados, y permite que otros buses se conecten al bus PCI.  
El PCI está diseñado para permitir una cierta variedad de configuraciones basadas en  
microprocesadores, incluyendo sistemas tanto de uno como de varios procesadores.. Utiliza la  
temporización síncrona y un esquema de arbitraje centralizado.  
SCSI (Small Computer System Interface) sólo se utiliza para dispositivos de almacenamiento y  
debe tener un controlador de interfaz

**B) ¿Que son los MIMD de la taxonomia de Flynn?**

La taxonomía de Flynn clasifica a los sistemas de varios procesadores según sus capacidades de  
procesamiento paralelo. Una de las categorías es MIMD: Múltiples secuencias de instrucción,  
múltiples secuencias de datos.  
Significa que un conjunto de procesadores ejecuta simultáneamente secuencias de instrucciones  
diferentes con conjuntos de datos diferentes, Los SMP, los “clusters” y los sistemas NUMA son  
ejemplos de esta categoría.  
En la organización MIMD, los procesadores son de uso general: cada uno es capaz de procesar  
todas las instrucciones necesarias para realizar las transformaciones apropiadas de los datos.  
Los computadores MIMD se pueden dividir según la forma que tienen los procesadores para  
comunicarse:  
• Memoria compartida (fuertemente acoplada): Los procesadores comparten una memoria  
común, entonces cada procesador accede a los programas y datos almacenados en la  
memoria compartida y los procesadores se comunican unos con otros a través de esa  
memoria.  
 o Multiprocesador simétrico (SMP): Varios procesadores comparten una única  
memoria mediante un bus compartido u otro tipo de mecanismo de  
interconexión. El tiempo de acceso a memoria principal es aproximadamente el  
mismo para cualquier procesador.  
 o Acceso no uniforme a memoria (NUMA): El tiempo de acceso a zonas de  
memoria diferentes puede diferir.  
• Memoria distribuida (débilmente acoplada):  
 o Clusters: Conjunto de computadores monoprocesadores independientes, o de  
SMP, que se interconectan. La comunicación entre los computadores es  
mediante conexiones fijas o mediante algún tipo de red.