INSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE CÓMPUTO

INGENIERIA EN SISTEMAS COMPUTACIONALES

MATERIA: SISTEMAS OPERATIVOS

PROFESOR: ARAUJO DIAZ DAVID

PRESENTA:

RAMIREZ BENITEZ BRAYAN

NÚMERO DE LISTA:

27

GRUPO: 2CV17

TAREA 3:

ADMINISTRACIÓN DE MEMORIA

CIUDAD DE MEXICO ABRIL DE 2021

1. ¿Qué características debe de tener una memoria ideal?

R: Debe ser infinitamente grande y con tiempo de acceso muy corto.

2. ¿Qué es el administrador de memoria y en que consiste su trabajo?

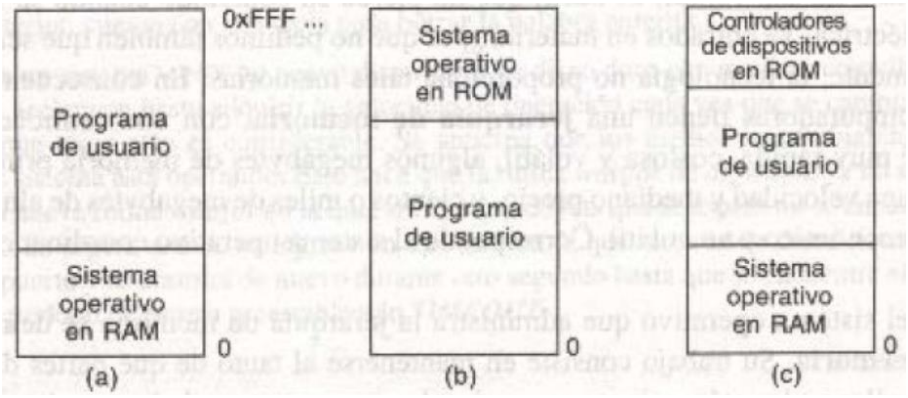
R: Se refiere a los distintos métodos y operaciones que se encargan de obtener la máxima utilidad de la memoria, organizando los procesos y programas que se ejecutan de manera tal que se aproveche de la mejor manera posible el espacio disponible, la operación principal que realiza es la de trasladar la información que deberá ser ejecutada por el procesador, a la memoria principal.

3. ¿Cuáles son las dos clases en la que es posible dividir los sistemas de administración de memoria?

R: Las que trasladan procesos entre la memoria y el disco durante la ejecución (intercambio y paginación) y los que no lo hacen.

4. En que consiste el esquema de administración de memoria por monoprogramación sin intercambio ni paginación. Mencione y describa tres ejemplos de este esquema.

R: En ejecutar solo un programa a la vez, repartiendo la memoria entre ese programa y el sistema operativo



5. En que consiste el esquema de administración de memoria por multiprogramación con particiones fijas. Cuáles son las ventajas y desventajas de usar una sola cola de entrada o colas distintas en este esquema. Describa una solución al problema de discriminación de trabajos pequeños.

R: La forma más fácil de lograr la multiprogramación consiste simplemente en dividir la memoria en particiones, posiblemente desiguales. Cuando llega un trabajo, se le puede colocar en la cola de entrada de la partición pequeña que puede contenerlo o con una sola cola administrar todo.

6. Describa los problemas de relocalización y protección. Mencione y explique cómo se resuelven estos problemas.

R: Cuando se enlaza un programa (es decir, cuando se combinan el programa principal, los procedimientos escritos por el usuario y los procedimientos de biblioteca en un solo espacio de direcciones), el enlazador necesita saber en qué dirección de la memoria comenzará el programa. A esto se le conoce como el problema de la relocalización (reubicación).

El problema de la protección consiste en que un programa mal intencionado siempre puede construir una instrucción nueva y saltar a ella. Puesto que en este sistema los programas usan direcciones de memoria absolutas, no direcciones relativas a un registro no hay manera de impedir que un programa construya una instrucción que lea o escriba cualquier palabra en la memoria.

7. Cuáles son los dos enfoques de administración de memoria. Descríbalos brevemente.

R: El intercambio consiste en traer a la memoria un proceso entero, ejecutarlo durante un rato y volver a guardarlo en disco y el de memoria virtual permite que los programas se ejecuten, aunque solo una parte de ellos este en la memoria principal.

La memoria virtual la idea es que el tamaño combinado de la pila, programa y datos puede exceder la memoria física disponible para ello. El S.O. mantiene en memoria aquellas partes del programa que se deben permanecer en memoria y el resto lo deja en disco, las partes entre el disco y la memoria se intercambian de modo que se vayan necesitando.

8. En que consiste el esquema de administración de memoria por intercambio. Cuál es la diferencia en usar particiones fijas y particiones variables.

R: En un sistema por lotes la organización de la memoria en particiones fijas es adecuado, pero en un ambiente multiusuario la situación es distinta con el tiempo compartido, ya que existen más usuarios de los que puede albergar la memoria, por lo que es conveniente albergar el exceso de los procesos en disco., por supuesto para ser ejecutados estos procesos deben ser trasladados a la memoria principal. Al traslado de procesos de disco a memoria y viceversa se le llama intercambio.

En particiones fijas el objetivo en todo esto es tener más de un proceso en memoria a la vez, solución posible sería dividir la memoria en n partes al inicio de una sesión de uso de la máquina, pero aun así se obtiene el desperdicio de particiones grandes con una tarea pequeña, la respuesta puede ser tener particiones pequeñas también, las tareas que van llegando se forman hasta que una partición adecuada está disponible, en cuyo momento la tarea se carga en esa partición y se ejecuta hasta terminar mientras que mediante un algoritmo de administración de memoria las particiones variables varían de forma dinámica durante el uso de la máquina, evitando desperdicio de memoria

9. ¿Qué es la compactación de memoria y por qué casi nunca se emplea?

R: El proceso de compactación son unas instancias particulares del problema de asignación de memoria dinámica, y esta se refiere a satisfacer una necesidad de tamaño (N) en una lista de huecos libres. Entre tantas posibilidades existe una que determina el hueco más indicado en el momento de asignar. Los algoritmos de compactación suelen tener un alto costo computacional.

10. En que consiste el esquema de administración de memoria con mapas de bits. Cuál debe de ser la relación entre el mapa de bits, el tamaño de la memoria y el tamaño de la unidad de asignación.

R: Con un mapa de bits, la memoria se divide en unidades de asignación, que pueden ser desde unas cuantas palabras hasta varios kilobytes. A cada unidad de asignación le corresponde un bit del mapa de bits. El bit es 0 si la unidad de asignación está libre y 1 si está ocupada. El tamaño de la unidad de asignación es una cuestión de diseño importante Cuanto más pequeña sea la unidad, mayor será el mapa de bits. Si se escoge una unidad de asignación grande, el mapa de bits será pequeño, pero podría desperdiciarse una unidad de memoria apreciable en la última unidad de asignación del proceso si el tamaño del proceso no es un múltiplo exacto de la unidad de asignación

11. En que consiste el esquema de administración de memoria con listas enlazadas. Describa los algoritmos de primer ajuste, siguiente ajuste, mejor ajuste, peor ajuste y ajuste rápido; diga cómo es posible hacer estos algoritmos más eficientes. ¿Cuál de estos algoritmos es el más eficiente?

R: Es una forma de llevar control de la memoria donde principalmente se trata de mantener una lista enlazada de bloques de memoria asignados y libres, donde cada bloque es un proceso o un hueco entre dos procesos. Normalmente un proceso que termina tiene dos procesos vecinos. Dichos vecinos pueden ser procesos o hueco, lo que da lugar a las cuatro combinaciones.

•La actualización de la lista requiere sustituir una P por una H

•La unión de entradas en una sola

•La fusión de las entradas

Si los procesos y huecos se mantienen en una lista ordenada por dirección, pueden utilizarse varios algoritmos para asignar memoria a un proceso recién creado (o a un proceso existente que se intercambia a memoria). Existen algoritmos para determinar por medio del gestor de memoria cuánta memoria se debe asignar al proceso.

•Algoritmo de primer ajuste (first fit): Consiste en asignar el primer hueco disponible que tenga un espacio suficiente para almacenar el programa. La principal desventaja es el reiterado uso de las primeras posiciones de memoria. Este último

inconveniente repercute negativamente en la circuitería, debido a que se produce un mayor desgaste en dichas posiciones.

•Algoritmo de segundo ajuste (second fit): Se continúa a partir de la posición de la última asignación realizada. Es muy probable que haya un hueco a partir de esa posición, reduciendo la longitud de la búsqueda. De esta forma se resuelve el inconveniente de usar en exceso las primeras posiciones de la memoria. Cuando se alcanza el final de la memoria se vuelve a comenzar la búsqueda desde el principio.

•Algoritmo de mejor ajuste (best fit): Consiste en asignarle al proceso el hueco con menor desperdicio interno, el hueco el cual al serle asignado el proceso deja menos espacio sin utilizar. Su mayor inconveniente es su orden de complejidad (orden lineal, *O(n)*) debido a que hay que recorrer todo el mapa de bits o toda la lista de control (una posible solución sería usar una lista de control encadenada que mantenga los huecos ordenados por tamaño creciente). Otro problema es la fragmentación externa, debido a que se asigna el menor hueco posible, el espacio sobrante será del menor tamaño posible lo que da lugar a huecos de tamaño normalmente insuficiente para contener programas.

•Algoritmo de peor ajuste (worst fit): Al contrario que el criterio anterior, se busca el hueco con mayor desperdicio interno, el hueco el cual al serle asignado el proceso deja más espacio sin utilizar, y se corta de él el trozo necesario (así la porción sobrante será del mayor tamaño posible y será más aprovechable). Tiene el mismo inconveniente en cuanto a orden de complejidad que el mejor ajuste (debido a la longitud de las búsquedas) y la fragmentación no resulta demasiado eficiente.

Los cuatro algoritmos anteriores pueden acelerarse manteniendo listas separadas para los procesos y los huecos. El precio de esta aceleración es una complejidad adicional y ralentización cuando se libera la memoria, ya que los segmentos liberados deben eliminarse de la lista de procesos e insertarse en la lista de huecos.

12. Cuál fue la primera solución al problema que se presenta cuando los programas ya no caben en memoria.

R: Consiste en que las aplicaciones mantengan parte de su información en disco, moviéndola a la memoria principal cuando sea necesario

13. ¿Cuál es la idea básica de la memoria virtual?

R: La idea básica detrás de la memoria virtual es que el tamaño combinado del programa, sus datos y su pila pueden exceder la cantidad de memoria física disponible. El sistema operativo mantiene en la memoria principal aquellas partes del programa que se están usando en cada momento, manteniendo el resto de las partes del programa en el disco.

14. Describa detalladamente la técnica de administración de memoria virtual por paginación.

R: La mayoría de los sistemas con memoria virtual usan una técnica denominada paginación. Estas direcciones generadas por el programa se denominan direcciones virtuales y constituyen el espacio de direcciones virtual. En ordenadores sin memoria virtual, la dirección virtual se coloca directamente sobre el bus de memoria y eso hace que la palabra de memoria física con esa dirección se lea o escriba. Cuando se utiliza memoria virtual, las direcciones virtuales no se envían directamente al bus de memoria, sino que van a una unidad de Gestión de memoria (MMU), que establece una correspondencia entre las direcciones virtuales y las direcciones físicas de la memoria. El espacio de direcciones virtual se divide en unidades llamadas páginas. Las unidades correspondientes en la memoria física se denominan marcos de página. Y la relación entre las direcciones virtuales y las direcciones de la memoria física viene dada por la tabla de páginas. El número de página se utiliza como índice para consultar la tabla de páginas, obteniendo el número del marco de página correspondiente a esa página virtual. Si el bit de presencia es 0, se generará una excepción que cederá el control al sistema operativo. Si el bit es 1, el número de marco de página encontrado en la tabla de páginas se copia en los tres bits de mayor orden del registro de salida junto con el desplazamiento de 12 bits que se copia (sin ninguna modificación) de la dirección virtual recibida. Juntos, esos dos campos forman una dirección física de 15 bits. Finalmente, el registro de salida se vuelca al bus de memoria como la dirección de memoria física a la que efectivamente se va a acceder.

15. ¿Cuál es el trabajo de la unidad de administración de memoria (MMU)?

R: Establece una correspondencia entre las direcciones virtuales y las direcciones físicas de la memoria.

16. ¿Cuál es el propósito de las tablas de página? ¿Qué problemas se presentan?

R: El propósito de la tabla de páginas es establecer una correspondencia aplicando las páginas virtuales sobre los marcos de página. Matemáticamente, la tabla de páginas es una función, con el número de página virtual como argumento y el número de marco de página como resultado. Utilizando el resultado de esta función, el campo de página virtual de una dirección virtual puede reemplazarse por un campo de marco de página, formando así una dirección de memoria física. A pesar de lo sencillo de esta descripción, hay que resolver los siguientes problemas:

•La tabla de páginas puede ser extremadamente grande.

•La traducción de direcciones debe realizarse muy rápidamente

17. En qué consisten las tablas de página multinivel. Muestre y explique un ejemplo.

R: El objetivo es paginar la tabla de páginas lo que permite que esta no esté cargada completa en memoria y que no ocupe direcciones consecutivas. En los s.o con tablas de páginas multinivel los números de página se dividen en dos partes: los bits más significativos indican el directorio de páginas correspondiente y con los bits menos significativos el índice del directorio en el que se encuentra la página buscada.

Tabla de páginas para todo el sistema: Es una variante de las tablas de página multinivel, en la que el dispositivo traductor sólo contiene una entrada por cada proceso, indexadas por PID, y una dirección de tabla que contiene las páginas de dicho proceso. Así, ante una conmutación de procesos basta con cambiar de PID.

18. ¿Por qué se presenta un fallo de página?

R: Es una excepción arrojada cuando un programa informático requiere una dirección que no se encuentra en la memoria principal actualmente. Aunque el término sugiere un mal funcionamiento, se trata de un procedimiento normal dentro de la marcha del programa.

19. ¿Qué bits componen una entrada de tabla? Describa cada uno de ellos.

R: El bit de Presencia. Si este bit es 1, la entrada es válida y puede usarse; si es 0, la página virtual a la que corresponde la entrada no está actualmente en memoria. Acceder a una entrada de la tabla de páginas con el bit de presencia a 0 provoca una falta de página. Los bits de Protección indican qué tipos de acceso están permitidos. En su forma más simple, este campo contiene un bit, que vale 0 si se permite leer y escribir, o 1 si sólo se permite leer. Un esquema más sofisticado utiliza 3 bits, para habilitar/inhibir independientemente la lectura, la escritura y la ejecución de palabras de la página (análogo a los bits rwx). Los bits de página Modificada y Referenciada siguen la pista del uso de la página. Cuando se escribe en una página, el hardware activa automáticamente el bit de Modificada. El bit de Referencia se activa siempre que se referencia una página, ya sea para leer o para escribir. Su función es la de ayudar al sistema operativo a seleccionar la página que elegirá como víctima cuando se presente una falta de página. Finalmente, el último bit permite inhabilitar el uso de la caché para la página. Esta característica es importante en el caso de páginas que contienen direcciones correspondientes a registros de dispositivos, en vez de a posiciones de memoria. Si el sistema operativo está dando vueltas en un bucle de polling esperando a que algún dispositivo de E/S responda a un comando que se le acaba de enviar, es indispensable que el hardware siga extrayendo la palabra del dispositivo, y que no utilice una copia antigua almacenada en la caché. Con este bit, puede desactivarse el uso de la caché para esa página. Las máquinas que tienen un espacio de E/S separado y no utilizan E/S mapeada en memoria no necesitan ese bit.

20. En que consiste el trabajo de los Buffers de consulta para traducción TLB.

R: Es una memoria caché administrada por la unidad de gestión de memoria (MMU), que contiene partes de la tabla de paginación, la cual relaciona las direcciones lógicas con las físicas. Posee un número fijo de entradas y se utiliza para obtener la traducción rápida de direcciones. Si no existe una entrada buscada, se deberá revisar la tabla de paginación y tardará varios ciclos más, sobre todo si la página que contiene la dirección buscada no está en memoria primaria (véase memoria virtual). Si en la tabla de paginación no se encuentra la dirección buscada, saltará una interrupción conocida como fallo de página.

21. Cuando se presenta un fallo de protección en el TLB.

R: Con el manejo software de las TLB, un fallo genera una excepción fallo de TLB, y el sistema operativo debe acceder a las tablas de paginación y realizar la traducción por software. Entonces, el sistema operativo carga la traducción en el TLB y reinicia el programa desde la instrucción que causó el fallo. Como en el sistema de manejo hardware, si el SO no encuentra una traducción válida en las tablas, ocurre un fallo de página y el SO deberá manejarlo de la manera correspondiente.

22. ¿Qué es la administración de TLB por software? Cómo este enfoque puede reducir las fallas y el costo de las mismas.

R: Algunas máquinas RISC modernas, incluidas MIPS, Alpha y HP PA, el sistema operativo carga explícitamente las entradas del TLB. Cuando no se encuentra una página virtual en el TLB, en lugar de que la MMU se remita simplemente a las tablas de páginas para encontrar y obtener la referencia de página requerida, genera una falla de TLB y deja que el sistema operativo se encargue del problema. Éste debe encontrar la página, eliminar una entrada del TLB, introducir la nueva y reiniciar la instrucción que falló. Y, desde luego, todo esto debe hacerse con unas cuantas instrucciones, porque las fallas de TLB podrían ocurrir con mucha mayor frecuencia que las fallas de página.

Estas fallas pueden reducirse manteniendo un caché grande (p. ej., 4K) en software de entradas de TLB en un lugar fijo cuya página siempre se mantiene en el TLB. Si primero se busca en el caché en software, el sistema operativo puede reducir sustancialmente las fallas de TLB.

23. Describa en qué consisten las tablas de páginas invertidas y cuál es su desventaja principal.

R: Es una técnica de paginación en la cual hay una entrada por cada página real (o frame) de la memoria y además incluye información del proceso que posee dicha página. Por lo tanto, en el sistema solo habrá una tabla de páginas invertida y esta solo tendrá una entrada por cada frame en la memoria física. Los sistemas que utilizan tablas de páginas invertidas tienen problemas para implementar el concepto de memoria compartida ya que cada entrada de la tabla de páginas invertida corresponde a solo un frame en memoria.

24. ¿Qué dice el algoritmo de reemplazo de página óptimo? ¿Cuál es su principal problema?

R: Al momento de ocurrir un fallo de página cierto conjunto de páginas se encuentran en la memoria, en la siguiente instrucción se hará referencia a una de estas páginas, otras páginas no se utilizaran sino hasta mucho después, cada página puede ejecutarse con el número de instrucciones ejecutadas antes de la primera referencia a esa página, el algoritmo dice que se elimine la página con la mayor etiqueta; si una página no va a utilizase sino hasta mucho después que otra la eliminación de la primera retrasa el fallo de página lo más posible, el único problema de este algoritmo es que es irrealizable. Al momento del fallo de página el S.O. no tiene forma de saber a qué página se hace referencia.

25. En que consiste el algoritmo de sustitución de páginas no usadas recientemente.

R: Implica una hipótesis que indica que es mejor eliminar una página modificada sin referencias al menos por lo general un intervalo de reloj, este algoritmo es fácil de comprender, de implantación eficiente y con un rendimiento que, aún sin ser el óptimo si es adecuado en muchos casos.

26. Describa el algoritmo de sustitución de páginas de primera que entra, primera que sale (FIFO).

R: El sistema operativo tiene una lista de todas las páginas que se encuentran en memoria, siendo la primera página la más antigua y la última la más reciente, en un fallo de página, se elimina la primera página y se añade la nueva al final de la lista.

27. Describa en que consiste el algoritmo de sustitución de páginas de segunda oportunidad.

R: Una modificación simple del FIFO que evita deshacerse de una página de uso frecuente inspecciona el bit R de la página más antigua, busca una página antigua sin referencias durante el anterior intervalo de tiempo.

28. En que consiste el algoritmo de sustitución de páginas por reloj.

R: Consiste en mantener las páginas en una lista circular con la forma de un reloj, una manecilla apunta hacia la más antigua. Al ocurrir un fallo de página se inspecciona la página a la que apunta la manecilla si su bit R=0 se retira de la memoria, se inserta la nueva página en su lugar en el reloj y la manecilla avanza una posición, si R=1 la manecilla avanza una posición y el bit se limpia, esto continua hasta encontrar una página con R=0.

29. Describa en que consiste el algoritmo de sustitución de páginas menos recientemente usadas.

R: Consiste en que si las páginas que se han usado mucho en las últimas instrucciones probablemente se usarán mucho en las siguientes. Se aplica cuando ocurre una falla de página este desaloja la página que haya estado más tiempo sin usarse. El sistema operativo examina todos los contadores de la tabla de páginas hasta encontrarse con el más bajo.

30. Describa la simulación de LRU en software; describa su modificación conocida como maduración.

R: Las computadoras no cuentan con ese hardware entonces lo hacen a nivel de software con el algoritmo NFU (No utilizado frecuentemente) donde cada página tiene asociado un contador. Inicialmente todos los contadores están en cero y trabaja de la siguiente forma:

- Inicialmente todos los contadores están en cero.
- Todos los contadores se desplazan un bit a la derecha antes de sumarle el bit R.
- El bit R se suma al bit de la extrema izquierda, no a la extrema derecha.
- Cuando ocurre una falla de página, se elimina la página cuyo contador sea el de valor más bajo.

31. En que consiste la paginación por demanda.

R: Es un sistema de paginación con el cual, además de las ventajas de la paginación convencional, se busca disminuir los tiempos de respuesta y aumentar la cantidad de programas en memoria. Para lograr estos objetivos se hace uso de un intercambiador perezoso (llamado paginador) el cual carga a memoria solo las páginas que serán utilizadas por el programa en ejecución, de esta manera se logra un menor tiempo de carga y un ahorro en cuanto a espacio utilizado por dicho programa, ya que, por un lado, no necesitamos que todo el programa este en memoria para comenzar su ejecución mientras que, por otra parte, al no estar el programa completo en memoria, disminuimos considerablemente el espacio que éste ocupa.

32. ¿Qué es el modelo de conjunto de trabajo?

R: Se define como conjunto de trabajo el rango de páginas con las que opera un proceso durante una fase de su ejecución, es decir, el conjunto de páginas que hay que mantener en memoria para que no se produzcan fallos de página mientras dure dicha fase.

33. ¿Qué es la prepaginación?

R: Es un intento de evitar este alto nivel de paginación inicial. La estrategia consiste en traer a memoria al mismo tiempo todas las páginas que se necesitarán.

34. Describa en que consiste el reemplazo de páginas local y global.

R: El reemplazo global permite que un proceso seleccione un marco para reemplazar de entre todo el conjunto de marcos, incluso si ese marco está asignado a otro proceso; un proceso puede quitar un marco a otro.

El reemplazo local requiere que cada proceso seleccione sólo de su propio conjunto de marcos asignados.

35. En qué consisten los métodos de reparto equitativo y proporcional.

R: Consiste en agrupar procesos por usuarios y grupos o departamentos. Donde debe asegurar que cada grupo recibe una parte proporcional de CPU. Se asigna una tasa de CPU a cada grupo. Esas partes no tienen por qué ser iguales (¡a veces se compran!). La prioridad de un proceso depende de su propia prioridad y del histórico de la totalidad de su grupo.

36. Describa el algoritmo de asignación de frecuencia de fallas de página.

R: Se usa el bit de referencia y se define un umbral F. Cuando ocurre un fallo de página se observa cuanto tiempo ha transcurrido desde el último fallo. Si es inferior a F la página referenciada se añade al conjunto de trabajo del proceso y los bits de referenciado de las demás páginas se ponen a cero; en otro caso se sacan de la memoria todas las páginas del proceso con el bit de referencia a uno y a las demás se les pone a cero. Se puede mejorar utilizando dos umbrales: uno inferior para reducir el tamaño del conjunto de trabajo y otro superior para ampliarlo. Se utiliza con alcance global y tamaño del conjunto residente fijo.

37. ¿Qué son los segmentos y por qué se emplean?

R: La segmentación es un esquema de administración de la memoria que soporta la visión que el usuario tiene de la misma. Un espacio de direcciones lógicas es una colección de segmentos. Cada segmento tiene un nombre y una longitud. Las direcciones especifican tanto el nombre del segmento como el desplazamiento dentro del segmento. Por lo tanto, el usuario especifica cada dirección mediante dos cantidades: un nombre de segmento y un desplazamiento.

38. Como se especifica una dirección en memoria segmentada o bidimensional.

R: Para especificar una dirección en esta memoria segmentada o bidimensional, el programa debe proporcionar una dirección de dos partes: un número de segmento y una dirección dentro de ese segmento.

39. Realice una tabla comparativa entre paginación y segmentación.

R:

Consideración	Segmentación	Paginación
Espacios de direcciones lineales	Varios	1
¿Puede distinguirse los procedimientos de los datos y protegerse de forma independiente?	Si	No
¿Puede manejar fácilmente tablas cuyo tamaño fluctúa	Si	No
Razón por la que se invento	Para dividir los programas y los datos en espacios de direcciones lógicamente independientes y facilitar la compartición y la protección	Para tener un espacio de direcciones lineal grande sin tener que adquirir más memoria física
¿Facilita la compartición de procedimientos entre usuarios?	Si	No
¿El programador necesita estar consciente de que está usando esta técnica?	Si	No
¿El espacio de direcciones total puede exceder el tamaño de la memoria física?	Si	Si

40. En que difiere la implementación de la segmentación respecto a la paginación.

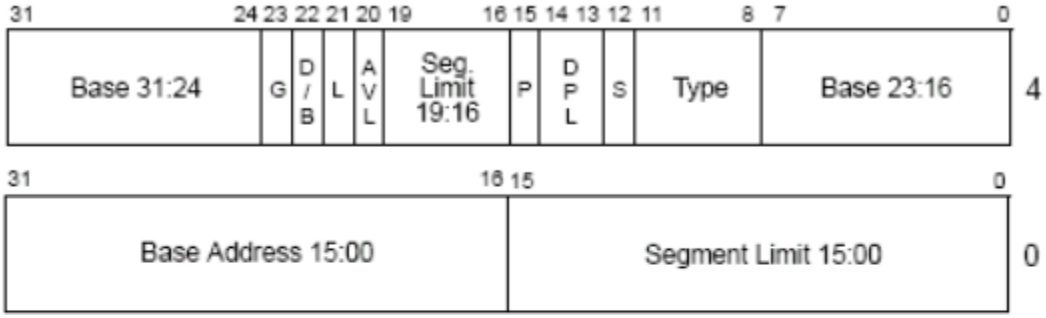
R: La paginación y la segmentación son ambos esquemas de gestión de memoria**.** La paginación permite que la memoria se divida en bloques de tamaño fijo, mientras que la segmentación divide el espacio de memoria en segmentos del tamaño de bloque variable. Cuando la paginación conduce a una fragmentación interna**,** la segmentación conduce a una fragmentación externa.

41. Cuáles son las dos tablas en las que consiste la memoria virtual de Pentium. Descríbalas y muestre la estructura de un selector.

R: Consiste en dos tablas, la tabla de descriptores locales (LDT) y la tabla de descriptores globales (GDT) Cada programa tiene su propia LDT, pero hay una sola GDT compartida por todos los programas de la computadora. La LDT describe los segmentos locales de cada programa, incluidos los de código, datos, pila, etc. Mientras que la GDT describe los segmentos del sistema, incluido el sistema operativo.

42. Mencione los bits de un descriptor de segmento de código de Pentium.

R:



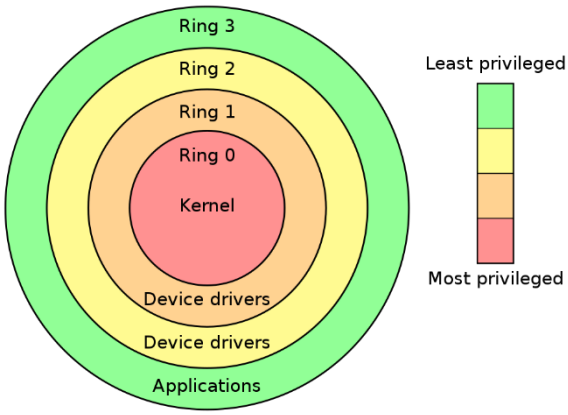
L — 64-bit code segment (IA-32e mode only)
AVL — Available for use by system software
BASE — Segment base address
D/B — Default operation size (0 = 16-bit segment; 1 = 32-bit segment)
DPL — Descriptor privilege level
G — Granularity
LIMIT — Segment Limit
P — Segment present
S — Descriptor type (0 = system; 1 = code or data)
TYPE — Segment type

43. Como se convierte un par (selector, desplazamiento) en una dirección lineal.

R: El Pentium dispone de un mecanismo hardware llamado MMU, Unidad de Manejo de Memoria, por el cual convierte la dirección virtual de 46 bits a dirección física de 32 bits. La MMU recoge la dirección virtual de 46 bits y lo introduce en la Unidad de Segmentación, que funciona siempre. La Unidad de Segmentación contiene la tabla de Segmentos, que determina y registra los segmentos que están en la memoria principal y que posición ocupan. Si el Segmento solicitado está en la memoria principal, se traduce la dirección virtual a dirección lineal de 32 bits

44. Muestre un ejemplo de cómo Pentium maneja los anillos de protección.

R: En modo protegido, hay cuatro niveles de privilegio o anillos, numerados de 0 a 3. El código del núcleo (kernel) del sistema operativo, que necesita usar instrucciones privilegiadas se ejecuta en el anillo 0, y las aplicaciones del usuario se ejecutan normalmente en el anillo 3. El sistema operativo puede asignar los anillos 1 y 2 a servicios de sistema, como protocolos de red o la gerencia de ventanas, que las aplicaciones pueden llamar. El hacer esto, les permite a los servicios acceder directamente los datos de la aplicación, mientras que se protegen de éstas, así como el núcleo queda protegido de los servicios. Sin embargo, esto requiere, al sistema operativo, especificar la protección de memoria a nivel de segmento (porque en el 80386, la protección de nivel de página no puede distinguir entre los anillos 0, 1, y 2) y puede ser difícil si el sistema operativo necesita ser portable a procesadores que soporten solamente dos anillos. En lugar de ello, un sistema operativo puede alcanzar una protección equivalente o más fuerte al ejecutar los servicios en el anillo 3 pero en un diferente espacio de dirección. Sin embargo, esto cuesta más, al tener una conmutación de contexto más compleja a la hora de la llamada, a menos que al servicio le sea dado un Task State Segment (segmento de estado de tarea) separado, el procesador primero debe cambiar al anillo 0 para cambiar el espacio de dirección, y después volver al anillo 3 para ejecutar el servicio. Si el código que se ejecuta no tiene suficientes privilegios, el resultado es generalmente una excepción que el sistema operativo puede manejar; pero también están las instrucciones que hacen los mismos chequeos sin levantar excepciones.



45. ¿Qué tipo de intercambio de almacenamiento real emplean los sistemas UNIX?

R: En el sistema operativo UNIX, la gestión de memoria ha variado de las versiones antiguas a las actuales. Antes, UNIX se basaba sólo en el intercambio (swapping) donde se empleaban particiones variables sin ningún tipo de esquema de memoria virtual. Las versiones actuales se basan en la memoria virtual paginada, utilizando para ello la paginación combinado con el intercambio.

46. ¿Qué contiene el mapa de páginas de los sistemas UNIX?

R: En un mapa tradicional de UNIX, el kernel y sus estructuras de datos asociadas residen en la parte alta del espacio de direcciones. El texto inicial y las áreas de datos empiezan en o cerca del principio de la memoria. Típicamente, los primeros 4 o 8 Kbyte de memoria son conservados fuera de los límites del proceso. La razón de esta restricción es para una depuración de programa fácil; indirectamente a través de un apuntador nulo causara un fallo de dirección inválida, en lugar de leer o escribir el texto de programa. La localización de memoria hecha por el proceso en ejecución usando la rutina de librería malloc ( ) (o la llamada al sistema sbrk) son hechas de la parte que empieza inmediatamente siguiente al área de datos y crece hasta las direcciones más altas. El vector de argumento y los vectores de ambiente están en la parte más alta de la porción de usuario del espacio de direcciones. La pila de usuario empieza justo debajo de estos vectores y crece hasta las direcciones más bajas.

47. ¿Cuáles son los tres estados que pueden adoptar los procesos en memoria en los sistemas UNIX?

R: En ejecución, en espera y bloqueado

48. ¿Qué estructuras de datos mantienen los sistemas UNIX para controlar la paginación?

R: Estructuras de datos para la gestión de memoria a bajo nivel y demanda de páginas (sistema de paginación):

- •Tabla de páginas ⇒ Normalmente, hay una tabla por proceso, con una entrada para cada página de la memoria virtual del proceso.
- •Descriptores de bloques de disco ⇒ Asociado a cada página de un proceso hay una entrada en la tabla que describe la copia en el disco de la página virtual.
- •Tabla de marcos de página (pfdata) ⇒ Describe cada marca de la memoria real y está indexada por el número de marco.
- •Tabla de uso de swap⇒ Existe una tabla de uso de swap por cada dispositivo de intercambio, con una entrada para cada página en dicho dispositivo.

49. Describa el trabajo del intercambiador en sistemas UNIX.

R: Intercambia procesos a MP e intenta intercambiar procesos a disco si necesita espacio en MP, se bloquea si no hay trabajo, se ejecuta en modo supervisor, el reloj lo desbloquea cada segundo y el núcleo cada vez que se bloquee un proceso y después se planifica (prioridad + alta)

50. Como se da la asignación dinámica de memoria en los sistemas UNIX.

R: Algunos sistemas UNIX utilizaban el intercambio en un sistema de particiones dinámicas. El movimiento de procesos entre la memoria principal y el disco lo realizaba el planificador de nivel medio, conocido como el intercambiador (swapper a medio plazo). El intercambio de la memoria principal al disco (*swapping out*) se iniciaba cuando el S.O. precisa memoria libre y estaba toda ocupa debido a alguno de los siguientes eventos:

Referencias

6 HiperpaginaciÃ³n. (s. f.). Hiperpaginacion. Recuperado 20 de abril de 2021, de http://lsi.vc.ehu.es/pablogn/docencia/manuales/SO/TemasSOuJaen/MEMORIAVIRTUAL/6Hiperpaginacion.htm

Administración de memoria. (s. f.). Administracion de memoria. Recuperado 20 de abril de 2021, de https://sistemas.com/administracion-de-memoria.php

Algoritomo fallos de pagina. (s. f.). PDF. Recuperado 20 de abril de 2021, de https://www.udg.co.cu/cmap/sistemas_operativos/paginacion/dosim107/archivos/AlgoritmoFallosPagina.html

CapÃtulo 08. (s. f.). calameo.com. Recuperado 20 de abril de 2021, de https://es.calameo.com/read/00032726541d3f8914727

Eslabon. Estación para Laboratorios Online. (s. f.). Estacion para laboratorios Online. Recuperado 20 de abril de 2021, de http://labvirtual.webs.upv.es/Compactacion.htm

Ludeña, N. (s. f.). *Administración de Memoria en UNIX*. Slideshare. Recuperado 20 de abril de 2021, de https://es.slideshare.net/natalialuva/administracin-de-memoria-en-unix

Memoria S.O. (s. f.). Climat. Recuperado 20 de abril de 2021, de https://www.cimat.mx/~alram/SO/memoria.pdf

Memoria virtual. (s. f.). PDF. Recuperado 20 de abril de 2021, de http://www.frsn.utn.edu.ar/tecnicas3/varios/memoria%20virtual%20tanenbaum.pdf

Memoria virtual con multiprogramacion - Wiki de Sistemas Operativos. (s. f.). Memoria Virtual. Recuperado 20 de abril de 2021, de https://1984.lsi.us.es/wiki-ssoo/index.php/Memoria_virtual_con_multiprogramacion

Pablo Turmero, Monografias.com. (s. f.). *Organización y arquitectura de sistemas de memoria - Monografias.com*. Monografias.com. Recuperado 20 de abril de 2021, de https://www.monografias.com/trabajos104/organizacion-y-arquitectura-sistemas-memoria/organizacion-y-arquitectura-sistemas-memoria.shtml

Romero, H. I. (s. f.). *Administración de memoria*. Slideshare. Recuperado 20 de abril de 2021, de https://es.slideshare.net/HederIthamar/administracin-de-memoria

Segmentación - Sistemas Operativos. (s. f.). Segmentacion. Recuperado 20 de abril de 2021, de https://sites.google.com/site/osupaep2010/administracion-de-memoria-1/memoria-virtual/segmentacion

Sistemas Operativos. (s. f.). UAEM. Recuperado 20 de abril de 2021, de http://ri.uaemex.mx/bitstream/handle/20.500.11799/63818/secme-29743.pdf?sequence=1&isAllowed=y

SO multiprogramables con particiones variables - Wiki de Sistemas Operativos. (s. f.). Sistemas Operativos. Recuperado 20 de abril de 2021, de https://1984.lsi.us.es/wiki-ssoo/index.php/SO_multiprogramables_con_particiones_variables#Primer_ajuste

U. (2021, 20 abril). *ADMINISTRACION DE MEMORIA*. Administracion de Memoria. http://juanfiliberto96.blogspot.com/2015/11/administracion-de-memoria.html