**PRACTICA 6**

***Dispositivos***

***1***

***a) Diferencias entre Dispositivos Orientados a Bloques y Orientados a Flujos:***

Orientados a Bloques:

* Transfieren datos en unidades de bloques de tamaño fijo.
* Cada bloque se trata como una unidad independiente con una dirección única.
* Apropiados para operaciones de lectura/escritura en bloques.
* Comúnmente utilizados en dispositivos de almacenamiento, como discos duros y memorias USB.

Orientados a Flujos:

* Transfieren datos de manera continua, como un flujo de información.
* No hay división en bloques discretos; se manejan como una secuencia continua de bits o bytes.
* Apropiados para la transmisión de datos en tiempo real.
* Se encuentran en dispositivos como teclados, ratones, tarjetas de red y dispositivos de entrada/salida (E/S) en general.

***b) Ejemplos de Dispositivos:***

Orientados a Bloques:

* Discos duros.
* Unidades de estado sólido (SSD).
* Memorias USB.

Orientados a Flujos:

* Teclados y ratones (en el contexto de la entrada de datos).
* Tarjetas de red (para la transmisión de datos en redes).
* Impresoras (cuando reciben datos para la impresión).

***c) Diferencias entre Dispositivos de E/S que el SO Debe Considerar:***

* Velocidad de Transferencia: Algunos dispositivos pueden transferir datos a diferentes velocidades, y el sistema operativo debe gestionar eficientemente estas variaciones.
* Prioridades: Algunos dispositivos pueden tener prioridades diferentes en cuanto a acceso y uso de recursos. El sistema operativo debe asignar y gestionar estas prioridades de manera efectiva.
* Métodos de Acceso: Los dispositivos pueden requerir métodos específicos de acceso, como acceso secuencial o aleatorio. El sistema operativo debe proporcionar interfaces coherentes para estos métodos.
* Temporización: La sincronización de operaciones de entrada/salida es crucial para evitar conflictos y asegurar un funcionamiento adecuado. El sistema operativo debe gestionar la temporización de manera eficiente.
* Manejo de Errores: Los dispositivos pueden experimentar errores durante las operaciones de entrada/salida. El sistema operativo debe ser capaz de gestionar y notificar estos errores de manera adecuada.

Estas consideraciones son esenciales para garantizar una gestión eficiente y confiable de los dispositivos de entrada/salida por parte del sistema operativo.

***2. Técnicas de E/S Describa como trabajan las siguientes técnicas de E/S***

**E/S Programada**: la transferencia de datos entre la memoria y los dispositivos de entrada/salida es gestionada directamente por la CPU.

Descripción del proceso:

1. Inicio de la Operación de E/S: El programa de la CPU inicia una operación de entrada/salida mediante instrucciones específicas en el código.
2. Espera Activa: La CPU ejecuta un bucle de espera activa (polling) para verificar constantemente el estado del dispositivo y esperar a que esté listo para la transferencia.
3. Transferencia de Datos: Cuando el dispositivo está listo, la CPU mueve los datos entre la memoria y el dispositivo.
4. Fin de la Operación: La CPU verifica si la operación de E/S ha finalizado y continúa con otras tareas.

**E/S Dirigida por Interrupciones:** la CPU no realiza activamente la verificación del estado del dispositivo; en cambio, espera a que el dispositivo envíe una interrupción para indicar que está listo.

El proceso es el siguiente:

1. Inicio de la Operación de E/S: Similar a la E/S programada, el programa de la CPU inicia una operación de entrada/salida mediante instrucciones específicas.
2. La CPU se Libera: Después de iniciar la operación, la CPU está libre para realizar otras tareas y no pierde tiempo esperando activamente.
3. Interrupción del Dispositivo: Cuando el dispositivo está listo para la transferencia, envía una señal de interrupción a la CPU.
4. Atención de la Interrupción: La CPU interrumpe su tarea actual para atender la interrupción. Guarda el estado actual y ejecuta un controlador de interrupciones específico para la operación de E/S.
5. Transferencia de Datos: La CPU mueve los datos entre la memoria y el dispositivo.
6. Fin de la Operación: Después de completar la operación, la CPU restaura el estado anterior y continúa con la tarea interrumpida.

***DMA (Acceso Directo a Memoria):*** un controlador de DMA toma el control temporal del bus del sistema para transferir datos directamente entre la memoria y los dispositivos, sin intervención constante de la CPU. Descripción del proceso:

1. Inicio de la Operación de DMA: La CPU inicia la operación de E/S y programa el controlador de DMA con la dirección de memoria de inicio, la cantidad de datos a transferir y otra información relevante.
2. Controlador de DMA toma Control: El controlador de DMA toma el control del bus del sistema, permitiendo que la CPU realice otras tareas.
3. Transferencia Directa: El controlador de DMA transfiere datos directamente entre la memoria y el dispositivo sin la intervención activa de la CPU.
4. Fin de la Operación de DMA: Una vez completada la transferencia, el controlador de DMA notifica a la CPU y libera el control del bus.

La técnica de DMA reduce la carga de trabajo de la CPU y acelera las operaciones de E/S al permitir transferencias directas de datos entre la memoria y los dispositivos.

***3. La técnica de E/S programa puede trabajar de dos formas:***

**E/S Mapeada**: se asigna a cada dispositivo de entrada/salida una dirección de memoria específica, de manera que los registros del dispositivo se "mapean" en direcciones de memoria. El proceso de funcionamiento es el siguiente:

* Asignación de Direcciones: Cada dispositivo de E/S tiene un conjunto de registros de control y de estado que se asocian con direcciones de memoria específicas.
* Acceso Directo a Registros: La CPU accede directamente a estos registros utilizando instrucciones de carga (in) y almacenamiento (out) en direcciones de memoria asignadas.

Ventajas: Facilita el acceso a los registros del dispositivo y simplicidad en la programación.

Desventajas: Utiliza mucho ancho de banda del bus de memoria, ya que la CPU debe acceder directamente a los registros del dispositivo.

**E/S Aislada**: los datos se transfieren entre la CPU y el dispositivo a través de registros de E/S ubicados en la interfaz de E/S. La CPU y el dispositivo interactúan a través de comandos de E/S específicos. El proceso de funcionamiento es el siguiente:

* Instrucciones Específicas de E/S: Se utilizan instrucciones especiales de E/S para transferir datos y controlar la operación del dispositivo.
* Registro de Estado y Control: El dispositivo tiene un registro de estado y control al que la CPU escribe comandos y desde el cual lee el estado.
* Transferencia de Datos: La CPU inicia la transferencia de datos enviando comandos al registro de control. El dispositivo lee estos comandos y realiza la transferencia de datos según sea necesario.

Ventajas: Utiliza menos ancho de banda del bus de memoria ya que no se accede directamente a las direcciones de memoria.

Desventajas: Puede ser más complejo de programar en comparación con la E/S mapeada debido a la necesidad de instrucciones específicas de E/S.

En resumen, la E/S mapeada asigna direcciones de memoria a los registros de E/S, permitiendo un acceso directo por parte de la CPU, mientras que la E/S aislada utiliza registros de E/S específicos y comandos de E/S para la transferencia de datos y control entre la CPU y el dispositivo.

***4. Enuncie las metas que debe perseguir un SO para la administración de la entrada salida***

Las metas clave que un sistema operativo (SO) debe perseguir para administrar eficientemente la entrada/salida (E/S) son:

* Eficiencia: Minimizar el tiempo de espera durante las operaciones de E/S.
* Equidad: Distribuir justamente los recursos de E/S entre procesos.
* Transparencia: Proporcionar una interfaz uniforme para los programas de aplicación.
* Sincronización: Gestionar la sincronización de operaciones de E/S de manera efectiva.
* Manejo de Errores: Detectar y manejar errores durante las operaciones de E/S.
* Optimización de la Capacidad: Planificar y optimizar el uso de dispositivos de E/S.
* Capacidad de Adaptación: Adaptarse a las características variables de dispositivos de E/S.
* Manejo de Dispositivos Diversos: Proporcionar una interfaz común para dispositivos diversos.
* Balance de Cargas: Distribuir operaciones de E/S equitativamente.
* Seguridad: Implementar medidas de seguridad contra operaciones no autorizadas.
* Uso Eficiente de la CPU: Minimizar la participación activa de la CPU en operaciones de E/S.

Al lograr estas metas, un SO mejora la eficiencia y confiabilidad del sistema en operaciones de entrada/salida.

***5. Drivers***

***a) ¿Qué son?***

Los drivers, también conocidos como controladores, son programas de software que permiten que un sistema operativo (SO) y un dispositivo de hardware interactúen y se comuniquen entre sí. Los drivers actúan como intermediarios, facilitando la traducción de las solicitudes del sistema operativo en comandos y operaciones que el dispositivo de hardware puede entender y ejecutar.

***b) ¿Qué funciones mínimas deben proveer?***

* Inicialización y Configuración: Configuración inicial del dispositivo y preparación para su uso.
* Gestión de Recursos: Asignación y liberación de recursos del sistema utilizados por el dispositivo.
* Transferencia de Datos: Facilitar la transferencia eficiente de datos entre el dispositivo y la memoria del sistema.
* Control de Estado: Monitorear y reportar el estado del dispositivo al sistema operativo.
* Interrupciones y Eventos: Manejar interrupciones y eventos generados por el dispositivo.
* Control de Energía: Gestionar el consumo de energía y los modos de suspensión o reposo del dispositivo.
* Interfaz con el Sistema Operativo: Proporcionar una interfaz consistente y compatible con el sistema operativo para que las aplicaciones y el kernel interactúen con el dispositivo.

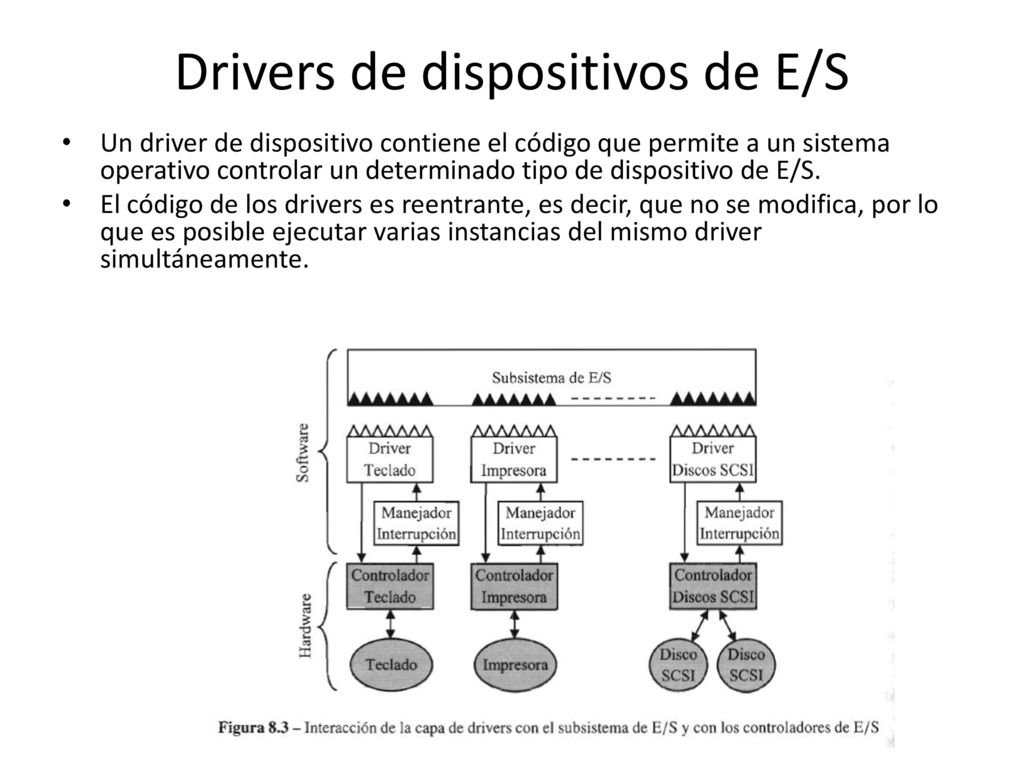
***c) ¿Quién determina cuales deben ser estas funciones?***

Las funciones que deben ser proporcionadas por los drivers son determinadas principalmente por los desarrolladores de hardware y el fabricante del dispositivo. Los estándares de la industria y las especificaciones técnicas también pueden influir en las funciones esenciales que deben ser implementadas en un driver.

En muchos casos, las funciones específicas del driver se definen mediante estándares como USB, PCI, o ACPI, que establecen las normas para la comunicación y el funcionamiento de dispositivos específicos. Las organizaciones industriales y los consorcios a menudo contribuyen a la creación de estándares que ayudan a garantizar la interoperabilidad entre dispositivos y sistemas operativos.

***6 Realice un gráfico que marque la relación entre el Subsistema de E/S, los drivers, los controladores***

***de dispositivos y los dispositivos.***

******

***7. Describa mediante un ejemplo los pasos mínimos que se suceden desde que un proceso genera un requerimiento de E/S hasta que el mismo llega al dispositivo.***

***Generación de la Solicitud de E/S:***

1. Generación de la Solicitud de E/S: El proceso en ejecución en el sistema operativo genera una solicitud de E/S, indicando que desea escribir datos en un archivo en disco.
2. Traducción a Llamada de Sistema: La solicitud de E/S se traduce en una llamada de sistema específica, como write() en sistemas basados en UNIX o WriteFile() en sistemas Windows. Esta llamada de sistema se realiza a través de la interfaz de programación de aplicaciones (API) del sistema operativo.
3. Interrupción de la CPU: La CPU, al recibir la llamada de sistema, pasa a modo kernel y ejecuta el código del kernel asociado con la operación de E/S. Esto puede involucrar una interrupción para cambiar a modo kernel.
4. Planificación y Despacho: Si el sistema operativo está utilizando un planificador de procesos, podría decidir cambiar el contexto y seleccionar otro proceso para ejecutarse mientras espera la finalización de la operación de E/S.
5. Manejo de la Solicitud de E/S: El kernel del sistema operativo maneja la solicitud de E/S, verifica la validez de la operación y prepara los datos para ser escritos en el dispositivo.
6. Llamada al Controlador del Disco: El kernel se comunica con el controlador de disco correspondiente para realizar la operación de escritura. Puede ser un controlador específico para el tipo de disco, como SATA o SCSI.
7. Uso del DMA (Acceso Directo a Memoria): Si se utiliza DMA, el controlador de disco podría programar el DMA para transferir directamente los datos desde la memoria del sistema al búfer del disco, sin la intervención activa de la CPU.
8. Transferencia de Datos al Dispositivo: El controlador de disco realiza la transferencia de datos al dispositivo de almacenamiento.
9. Finalización de la Operación: Una vez completada la escritura, el controlador de disco notifica al kernel sobre la finalización de la operación.
10. Actualización del Estado del Proceso: Si el proceso estaba esperando la operación de E/S, el kernel actualiza el estado del proceso y lo pone en lista de listos para su posible ejecución.

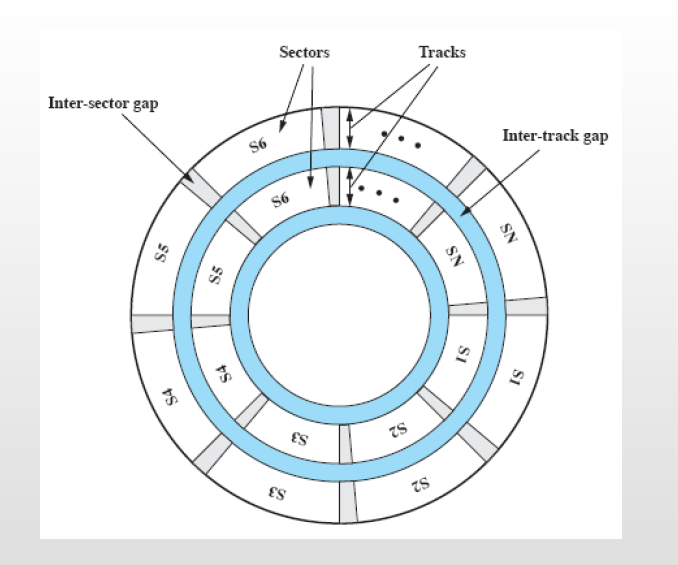
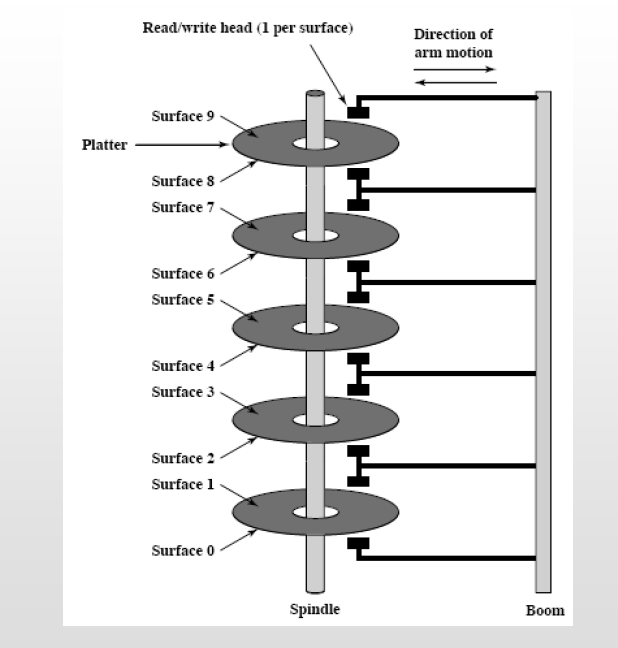
***9. Enuncie que servicios provee el SO para la administración de E/S.***

El sistema operativo (SO) proporciona servicios clave para administrar operaciones de entrada/salida (E/S), incluyendo:

* Controladores de Dispositivos: Interfaces y controladores para la comunicación con hardware.
* Sistema de Archivos: Gestión de lectura/escritura en dispositivos de almacenamiento.
* Buffers y Cachés: Mejora de eficiencia mediante buffers y cachés.
* Colas de E/S: Organización de solicitudes de E/S para optimizar recursos.
* Temporización y Sincronización: Coordinación de operaciones para evitar conflictos.
* Subsistemas Virtuales: Interfaces uniformes para dispositivos y archivos.
* Gestión de Energía: Control de consumo y modos de suspensión.
* Manejo de Interrupciones: Respuesta eficiente a interrupciones de dispositivos.
* Sistema de Colas de Prioridades: Asignación equitativa de recursos y operaciones.
* Estrategias de Planificación de E/S: Algoritmos para ordenar operaciones de E/S.
* Control de Acceso y Seguridad: Garantía de políticas de seguridad y acceso.
* API de E/S: Interfaces estándar para acceso a servicios de E/S.

***Administración de discos***

***10. Describa en forma sintética, cómo es la organización física de un disco, puede utilizar gráficos para mayor claridad.***

******

******

***11. La velocidad promedio para la obtención de datos de un disco esta dada por la suma de los siguientes tiempos:***

• Seek time (posicionamiento): tiempo que tarda en posicionarse la cabeza en el cilindro.

• Latency time (latencia): tiempo que sucede desde que la cabeza se posiciona en el cilindro hasta que el sector en cuestion pasa por debajo de la misma.

• Transfer time (transferencia): tiempo de transferencia del sector (bloque) del disco a la memoria.

***12. Suponga un disco con las siguientes características:***

* 7 platos con 2 caras utilizables cada uno.
* 1100 cilindros
* 300 sectores por pista, donde cada sector de es 512 bytes.
* Seek Time de 10 ms
* 9000 RPM .
* Velocidad de Transferencia de 10 MiB/s (Mebibytes por segundos). // 1segundo → 1000ms

***a) Calcule la capacidad total del disco.***

Capacidad = caras\*pistas\*sectores por pista\*tamanio sector

7\*(2\*1100\*300\*512) = 2.365.440.000 bytes

Megabytes≈2.365.440.000bytes÷1.048.576bytes/Megabyte

Megabytes ≈ 2.254 MB

***b) ¿Cuantos sectores ocuparía un archivo de tamaño de 3 MiB(Mebibytes)?***

Bytes=3×(10242) // 2 elevado a la 20 pq esto seria un mib

Bytes=3×1.048.576= 3.145.728

3.145.728 / 512 = 6144

Ocuparia 6144 sectores.

***c) Calcule el tiempo de transferencia real de un archivo de 15 MiB(Mebibytes). grabado en el disco de manera secuencial (todos sus bloques almacenados de manera consecutiva)***

Almacenamiento secuencial = seek + latency + (tiempo transferencia bloque \* #bloques)

Tiempo de transferencia por bloque= 10 x 220 = 10.485.760 bytes, hago una regla de tres simples con el tamanio del sector (512))

Latencia= 9000/2 // esto seria una regla de tres simples p9

#bloques-sectores 15MiB → 15.728.640byres/ 512 bytes = 30720

10ms + (9000/2) + (15MiB/s / (512 / 10MiB)) \* 300)

10ms + 4500ms + (15MiB/s / 51,2) \* 300

4510ms + 0.29\*300

4510 + 87 = 4597ms

***d) Calcule el tiempo de transferencia real de un archivo de 16 MiB(Mebibytes). Grabado en el disco de manera aleatoria.***

Almacenamiento aleatorio= (seek + latency + tiempo transferencia bloque) \* #bloques

Tiempo de transferencia por bloque= 512 bytes / 10MiB/s = 51,2

Latencia= 9000/2 = 4500ms

[ 10ms+4500+ (16 MiB/s / 51,2 MiB/s) ] \* 300

[4510 +0,31 ] \* 300 = 4510,31\*300 = 1.353.093 ms

***Administración de Archivos***

***17. Dados los siguientes métodos de administración de espació de un archivo***

Asignación Contigua:

En la asignación contigua, un archivo se almacena en bloques de sectores de manera adyacente en el disco. Cada archivo ocupa un bloque de sectores continuos. Esto facilita el acceso secuencial al archivo, ya que los bloques están contiguos en el disco.

Ventajas:

* Acceso rápido a archivos secuenciales debido a la contigüidad.
* Fácil de implementar y entender.

Desventajas:

* Fragmentación externa: a medida que se agregan y eliminan archivos, puede haber huecos entre bloques que no se pueden utilizar para almacenar archivos nuevos.
* Ineficiente para archivos de tamaño variable.

Asignación Enlazada:

En la asignación enlazada, un archivo se divide en bloques de tamaño variable y se almacena en ubicaciones no contiguas en el disco. Cada bloque contiene un puntero al siguiente bloque del archivo.

Ventajas:

* No hay fragmentación externa, ya que los bloques pueden estar dispersos en el disco.
* Eficiente para archivos de tamaño variable.

Desventajas:

* Acceso secuencial más lento ya que los bloques no están contiguos.
* Requiere un tiempo adicional para seguir los punteros.

Asignación Indexada:

En la asignación indexada, se utiliza un bloque de índice que contiene punteros a bloques de datos del archivo. Cada entrada en el bloque de índice apunta a un bloque de datos.

Ventajas:

* Acceso directo eficiente a cualquier bloque de datos a través del índice.
* Menos fragmentación externa que la asignación contigua.

Desventajas:

* Requiere más espacio en disco para almacenar el bloque de índice.
* Puede haber fragmentación interna en los bloques de datos si no se utilizan completamente.

***18. Gestión de espacio libre. Dados los siguientes métodos de gestión de espacio libre en un disco:***

Tabla de Bits:

En este método, se utiliza una tabla de bits para realizar un seguimiento del estado de cada bloque en el disco. Cada bit en la tabla representa el estado de un bloque (libre u ocupado).

Ventajas:

* Fácil implementación y comprensión.
* Acceso rápido para verificar el estado de un bloque.

Desventajas:

* Puede haber fragmentación interna si los bloques no tienen el mismo tamaño.
* El espacio requerido para la tabla de bits puede ser significativo en discos grandes.

Lista Ligada:

En este método, se mantiene una lista ligada de bloques libres. Cada nodo de la lista contiene información sobre el bloque libre y un puntero al siguiente bloque libre.

Ventajas:

* Eficiente para gestionar bloques de tamaños variables.
* Menos espacio requerido en comparación con la tabla de bits.

Desventajas:

* Puede haber fragmentación interna.
* Acceso más lento debido a la necesidad de seguir punteros.

Agrupamiento:

En el método de agrupamiento, se agrupan varios bloques libres en un solo grupo. La información sobre estos grupos se almacena en una estructura especial, y cada grupo puede contener bloques de diferentes tamaños.

Ventajas:

* Puede reducir la fragmentación interna al agrupar bloques.
* Más eficiente en términos de espacio que la tabla de bits.

Desventajas:

* Puede haber fragmentación externa.
* Acceso más lento debido a la necesidad de buscar en la estructura de agrupamiento.

Recuento:

En este método, se mantiene un contador que indica la cantidad de bloques libres en el disco. La información sobre bloques libres no se almacena de manera explícita.

Ventajas:

* Requiere menos espacio que la tabla de bits.
* Acceso rápido al contar los bloques libres.

Desventajas:

* Menos eficiente para manejar bloques de tamaños variables.
* Puede haber fragmentación interna.

***19. Gestión de archivos en UNIX.***

El sistema de archivos de UNIX utiliza una versión modificada del esquema de Asignación Indexada para la administración de espacio de los archivos. Cada archivo o directorio esta representado por una estructura que mantiene, entre otra información, las direcciones de lo bloques que contienen los datos del archivo: el I-NODO.

Cada I-NODO contiene 13 direcciones a los bloques de datos, organizadas de la siguiente

manera:

* 10 de direccionamiento directo.
* 1 de direccionamiento indirecto simple.
* 1 de direccionamiento indirecto doble.
* 1 de direccionamiento indirecto triple.

***Realice un gráfico que describa la estructura del I-NODO y de los bloques de datos.***

|  |
| --- |
| **METAINFORMACION** |
| **PUNTERO A BLOQUE DE DATOS (DD)** |
| **PUNTERO A BLOQUE DE DATOS (DD)** |
| **PUNTERO A BLOQUE DE DATOS (DD)** |
| **PUNTERO A BLOQUE DE DATOS (DD)** |
| **PUNTERO A BLOQUE DE DATOS (DD)** |
| **PUNTERO A BLOQUE DE DATOS (DD)** |
| **PUNTERO A BLOQUE DE DATOS (DD)** |
| **PUNTERO A BLOQUE DE DATOS (DD)** |
| **PUNTERO A BLOQUE DE DATOS (DD)** |
| **PUNTERO A BLOQUE DE DATOS (DD)** |
| **PUNTERO A BLOQUE DE DIRECCIONES (DIS)** |
| **PUNTERO A BLOQUE DE DIRECCIONES (DID)** |
| **PUNTERO A BLOQUE DE DIRECCIONES (DIT)** |

***Cada bloque es de 1 Kib(Kibibits). Si cada dirección para referenciar un bloque es de 32 bits:***

***¿Cuántas referencias (direcciones) a bloque pueden contener un bloque de disco?***

1 Kib=2^10 bits= 1024bits

1024 / 32 = 32 bits

***¿Cuál sería el tamaño máximo de un archivo?***

10 \* 1 Kib + 32 \* 1 \* 1kib + 32\*32 \*1\*1kib + 32\*32\*32\*1\*1kb

10kib + 32kib + 1024 kib + 32768 kib

33834kib

***20. Analice las siguientes fórmulas necesarias para localizar un I-NODO en la lista de inodos:***

nro bloque = ((nro de inodo -1)/nro. de inodos por bloque) + bloque de comienzo de la lista de inodos.

***Asumiendo que en el bloque 2 está en el comienzo de la lista de inodos y que hay 8 inodos por bloque: calcule donde se encuentra el inodo 8 y el 9. ¿Dónde estarían para bloque de disco de 16 inodos?***

8

((8-1)/16)+2 = 2,43 = 3 // me parece que se redondea siempre para arriba preguntar

9

( (9-1)/16) + 2 = 2,5 = 3

Desplazamiento del inodo en el bloque = ((nro de inodo - 1) módulo (número de inodos por bloque)) \* medida de inodo del disco.

***Si cada inodo del disco ocupa 64 bytes y hay 8 inodos por bloque de disco, el inodo 8 comienza en el desplazamiento 448 del bloque de disco. ¿Dónde empieza el 6? Si fueran inodos de 128 bytes y 24 inodos por bloque: ¿dónde empezaría el inodo 8?***

((8-1) modulo 8)\*64 = (7 mod 8) \* 64 = 448