TDIU11

Operating systems

Mass-Storage Systems

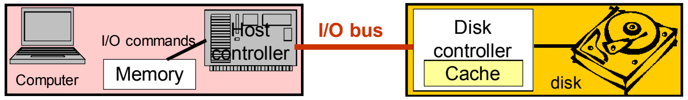
[SGG7/8/9] Chapter 12

Christoph Kessler, IDA,
Linköpings universitet.

Mass-Storage Systems

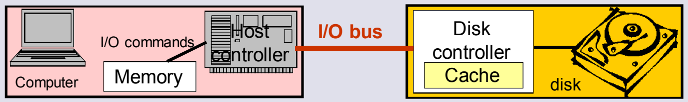
- Disk Attachment
- Disk Structure
- Disk Logical Structure
- Disk Access Time
- Disk Scheduling
- Disk Management
- Solid-State Disks and Hybrid Drives
- RAID Structure
- Tertiary Storage Devices (Tapes)

Disk Attachment



- **Host-attached storage**
  - accessed through I/O ports talking to **I/O buses** (such as EIDE, ATA, SATA, USB, FC, SCSI)
- **Network-attached storage**
  - special storage server
  - accessed by client computers through remote procedure calls (e.g. NFS for Unix, CIFS for Windows) via TCP over an IP network
  - often implemented as → RAID disk arrays

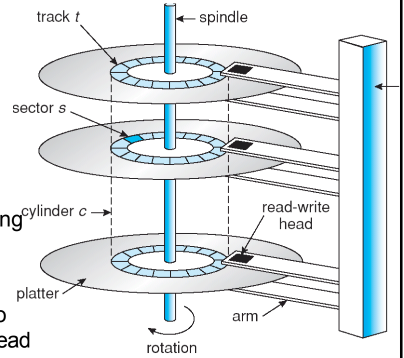
Disk Attachment



- **Host-attached storage** accessed through I/O ports talking to **I/O buses** (such as EIDE, ATA, SATA, USB, FC, SCSI)
  - **SCSI** is a bus, up to 16 devices on one cable,
    - **SCSI initiator** requests operation and
    - **SCSI targets** perform tasks
    - Each target can have up to 8 **logical units** (disks attached to device controller), e.g. RAID components
  - **FC** ("Fiber Channel") is high-speed serial architecture
    - Can be switched fabric with 24-bit address space – the basis of **storage area networks** (**SAN**s) where many hosts attach to many storage units
    - Can be **arbitrated loop** (**FC-AL**) of 126 devices

Disk – Physical Structure

- **Magnetic disks** provide bulk of secondary storage of modern computers
  - Drives rotate at 60 to 250 times per second
  - **Transfer rate** is rate at which data flow between drive and computer
  - **Head positioning time** (**random-access time**)
    = time to move disk arm to desired cylinder (**seek time,** ~3...12ms)
    + time for desired sector to rotate under the disk head (**rotational latency**)
  - **Head crash** – disk head making contact with the disk surface
    - That's bad
    - Accelerometers can help to avoid this by parking the head



Hard Disks

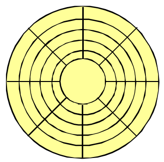
Spindle [rpm]	Average latency [ms]
4200	7.14
5400	5.56
7200	4.17
10000	3
15000	2

- Platters range from .85” to 14” (historically)
  - Commonly 3.5”, 2.5”
- Up to 10TB per drive (as of 2015)
- Performance
  - Transfer Rate – around 1Gb/sec
  - Seek time from 3ms to 12ms
    – 9ms common for desktop drives
  - Average seek time measured or calculated based on 1/3 of tracks
  - Latency based on spindle speed
    - 1 / (RPM / 60) = 60 / RPM
  - Average latency = ½ latency

Disk – Logical Structure

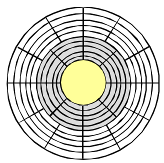
Older/smaller disks:

- Same number of sectors for all cylinders
- Constant angular velocity (CAV)
- Data more dense packed on inner cylinders



Newer/larger disks:

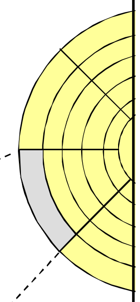
- Different numbers of sectors for different groups of cylinders
- Still CAV
- Smaller difference between dense and sparse cylinders

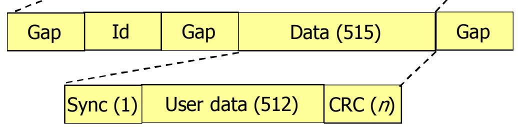


Disk – Logical Structure

One sector of one track contains one **block** of data (typically 512 bytes user data)...

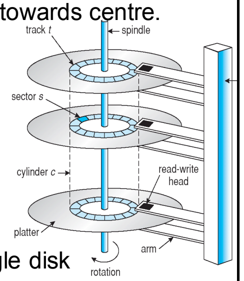
- Initial sector limit and block-id fields ("gap" = empty space of specific length)
- Data field containing:
  - Synchronisation bit
  - 512 bytes writeable data
  - *n*-byte Cyclic Redundancy Check (CRC) or **E**rror **C**orrecting **C**ode





Disk – Addressing

- Disk is viewed as a large 1D array of logical blocks
  - **logical block** is the smallest unit of transfer.
- Starting in cylinder #0 (outermost) continuing towards centre. Within each cylinder,
  - First block is on track #0 on surface #0 ...followed by all blocks in that track
  - Continues on track #0 on surface #1...
- Logical Block Addressing: Block 0..*max*
- A **partition** is a set of adjacent cylinders
  - → Virtual disk; can have several on a single disk
  - One partition = one file system
- Blocks can be marked as damaged and will then not be used any more

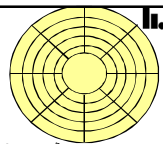


Disk–API towards OS

- OS may issue ***immediate*** read or write requests of blocks of data ...disk is expected to service them immediately as they arrive
  - Used e.g. by data base systems to ensure writing of log records asap (no delay tolerated)
  - Used when OS wants full control over disk scheduling, e.g., when accessing raw disk (such as swap space)
- OS may send multiple read and write requests (mixed) of blocks of data...
  - disk may service these in any order to maximize throughput

    ...what is the resulting access time?
    ...what is a good disk scheduling policy?

Disk – Access Time



Disk access time has 2 major components:

- **Seek time** – time to move R/W-heads to right cylinder
- **Rotational latency** – time for the right block to appear

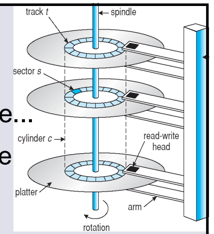
} Both in the order of several ms

Rotational speeds: ~400-12000 rpm and above

Example: Read a file of 128kB (= 256 blocks of 512 bytes)

- 12000 rpm = 200 rps → 5 ms/round
- Disk has 32 sectors (1 block/sector) → 5 ms / 32 = 156.25 μs/block to read
- Average seek time: 10ms
- Average rotational latency (12000rpm): 2.5ms
- File A – all blocks in sequence: 10+2.5+256*0.156 = 52.5 ms
- File B – blocks scattered over disk: 256*(10+2.5+0.156) = 3240 ms

Example revisited



Modern disk drives have an internal block cache...

...may read an entire cylinder of data into cache on each rotation!

- File A – all blocks in sequence: 10+2.5+256*0.156 = 52.5 ms ...data spans over 256/32 = 8 tracks
  - With one platter (2 tracks/cylinder) we need to rotate the disk 4 turns plus the time to move between adjacent cylinders (~0.1ms): 10+4*(0.1+5) = 30.4 ms
  - With 4 platters (8 tracks/cylinder) all data is read to internal disk cache after 10+5 = 15 ms
- File B – all blocks scattered all over the place... ...using the cache does not help much.

Disk Scheduling

- The operating system is responsible for using hardware efficiently — for the disk drives, this means having a fast access time and disk bandwidth
- Minimize seek time
  - Seek time ~ seek distance
- **Disk bandwidth** = total #bytes transferred, divided by the total time between the first request for service and the completion of the last transfer
- There are many sources of disk I/O requests
  - OS
  - System processes
  - Users processes
- I/O request includes input or output mode, disk address, memory address, number of sectors to transfer
- OS maintains queue of requests, per disk or device
- Idle disk can immediately work on I/O request, busy disk means work must queue
  - Optimization algorithms only make sense when a queue exists

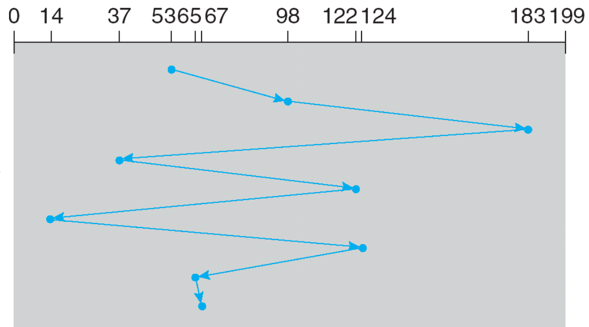
Disk Scheduling

- Among multiple pending requests (read / write block), choose the next to be serviced
  - Objective: Minimize seek time
    - Seek time ~ seek distance
- Several algorithms exist →
- Running example:
  - 200 tracks (0..199)
  - Request queue:
    tracks 98, 183, 37, 122, 14, 124, 65, 67
  - Head currently on track 53

FCFS (FIFO)

… i.e., no disk scheduling at all

queue = 98, 183, 37, 122, 14, 124, 65, 67
head starts at 53

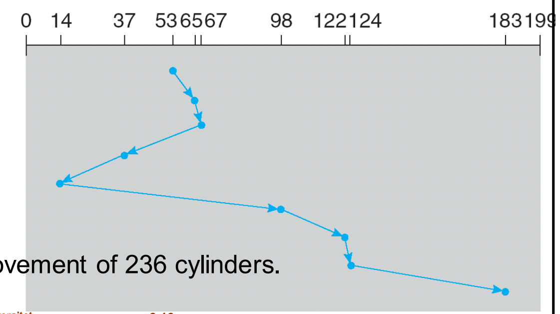


-> total head movement of 640 cylinders.

SSTF

- Selects the request with the **shortest seek time** from the current head position.
- May cause starvation of some requests.

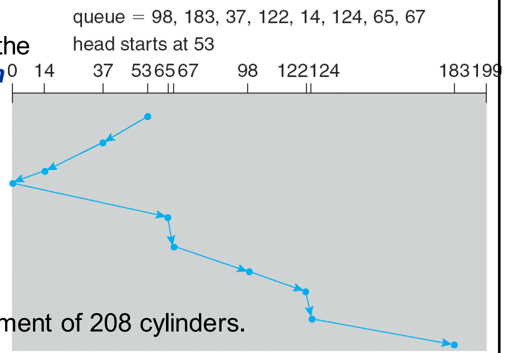
queue = 98, 183, 37, 122, 14, 124, 65, 67
head starts at 53



total head movement of 236 cylinders.

SCAN

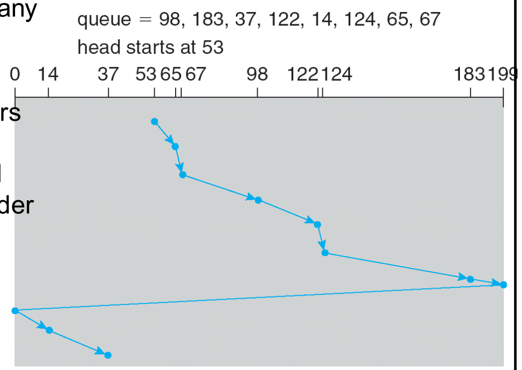
- The disk arm starts at one end of the disk, and moves toward the other end, servicing requests until it gets to the other end of the disk, where the head movement is reversed and servicing continues.
- Sometimes called the ***elevator algorithm***
- Biased towards accessing middle sectors faster



total head movement of 208 cylinders.

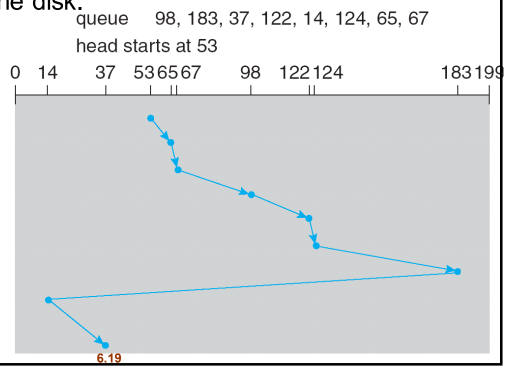
C-SCAN

- The head moves from one end of the disk to the other, servicing requests as it goes. When it reaches the other end, however, it immediately returns to the beginning of the disk, without servicing any requests on the return trip.
- Treats the cylinders as a circular list that wraps around from the last cylinder to the first one.
- Provides a more uniform wait time than SCAN.



C-LOOK

- Variant of C-SCAN
- Arm only goes as far as the last request in each direction, then reverses direction immediately, without first going all the way to the end of the disk.



Selecting a Disk-Scheduling Algorithm

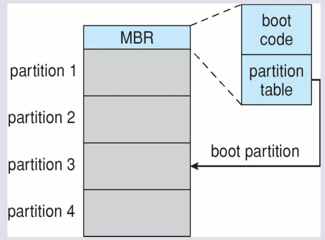
- For few requests – all behave like FCFS
- SCAN and C-SCAN perform better for systems that place a heavy load on the disk.
- Requests for disk service can be influenced by the file-allocation method.
  - Try to keep blocks of one file close to each other
- The disk-scheduling algorithm should be written as a separate module of the operating system, allowing it to be replaced with a different algorithm if necessary.
  - Or, it could be part of the disk controller (outside the OS)
- Either SSTF or LOOK is a reasonable choice for the default algorithm if the goal is throughput.
- Priority scheduling if some requests are more important
- What about rotational latency?
  - Difficult for OS to calculate
- How does disk-based queueing affect OS queue ordering efforts?

Disk Management

- ***Low-level formatting***, or ***physical formatting*** — Dividing a disk into sectors that the disk controller can read and write.
- To use a disk to hold files, the OS still needs to record its own data structures on the disk.
  - ***Partition*** the disk into one or more groups of cylinders.
  - ***Logical formatting*** or "making a file system".
- **Boot block** initializes system.
  - The bootstrap is stored in ROM.
  - *Bootstrap loader* program.
- Methods such as ***sector sparing*** used to handle bad blocks.

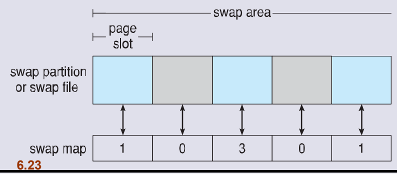
Disk Management (Cont.)

- Raw disk access for apps that want to do their own block management, keep OS out of the way (databases for example)
- Boot block initializes system
  - The bootstrap is stored in ROM
  - **Bootstrap loader** program stored in boot blocks of boot partition



Swap-Space Management

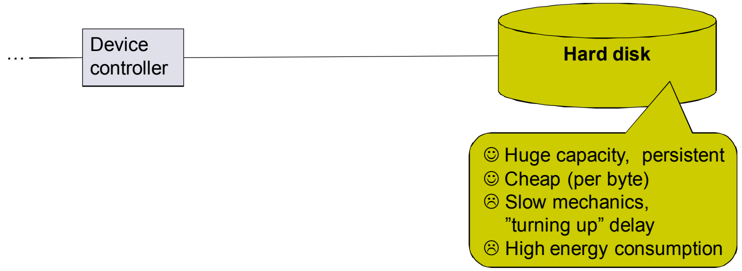
- Swap-space — Virtual memory uses disk space as an extension of main memory
  - Less common now due to memory capacity increases
- Swap-space can be carved out of the normal file system, or, more commonly, it can be in a separate disk partition (raw)
- Swap-space management
  - BSD allocates swap space when process starts; holds text segment (the program) and data segment
  - Kernel uses **swap maps** to track swap-space use
  - Solaris 2 allocates swap space only when a dirty page is forced out of physical memory
- What if a system runs out of swap space?
- Some systems allow multiple swap spaces



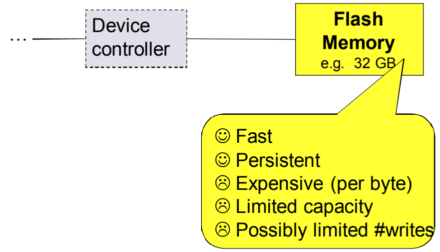
Operating System Issues

- Major OS jobs are
  - to manage physical devices and
  - to present a virtual machine abstraction to applications
- For hard disks, the OS provides two abstraction:
  - **Raw disk** – an array of data blocks, no file system
    - Used exclusively and managed by an application program e.g., some database systems
    - RAID administrative information
    - Swap space (faster than if realized as a single large file)
    - Virtual memory backing store
  - **"Cooked disk" / File system** – OS queues and schedules the interleaved requests from several applications.

Traditional Hard Disk Drives

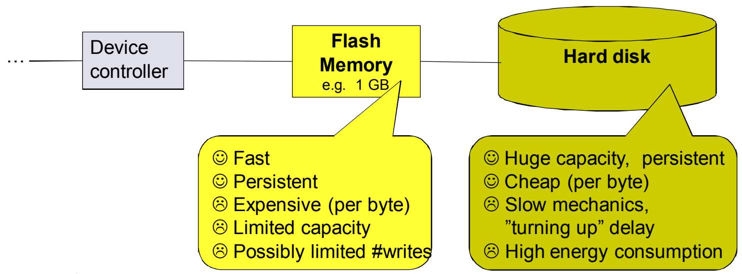


Solid State Drives (SSD)



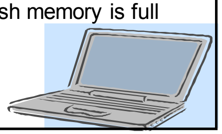
- No mechanical parts – good for smartphones, notepads etc.
  - No need of disk scheduling etc.
- Various technologies available, e.g.
  - Flash memory
  - Battery-backed DRAM

Hybrid Drives



Hybrid drive: Mirror boot system files and frequently used files / blocks in flash memory (~ explicitly managed cache)

- ☺ Faster startup (saves 8..25s for notebooks)
- ☺ Write-block accesses buffered until write buffer in flash memory is full
  - ☺ And can use disk scheduling internally
- ☺ Extends notebook battery lifetime by ~10%
- ☹ Needs management support by OS



Solid-State Disks

- Nonvolatile memory used like a hard drive
  - Many technology variations
- Can be more reliable than HDDs
- More expensive (8 to 9 times) per MB
- Maybe have shorter life span
- Less capacity
- But much faster
- No moving parts, so no seek time or rotational latency

TDIU11

Operating systems

APPENDIX

RAID

Tertiary storage

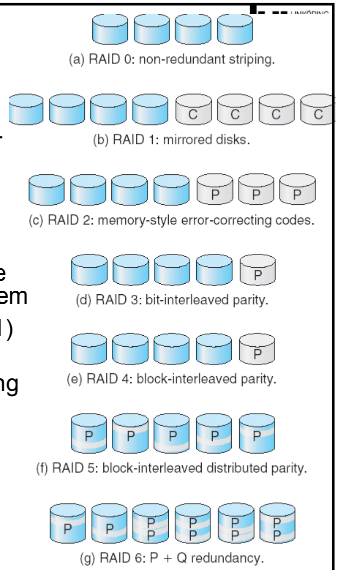
Christoph Kessler, IDA,
Linköpings universitet.

RAID Structure

- RAID – redundant array of inexpensive/independant disks
  - multiple disk drives provides reliability via **redundancy**
- Increases the **mean time to failure**
- **Mean time to repair –** exposure time when another failure could cause data loss
- **Mean time to data loss** based on above factors
- If mirrored disks fail independently, consider disk with 100,000h mean time to failure and 10h mean time to repair
  - Mean time to data loss is $100{,}000^2 / (2 * 10) = 500 * 10^6$ hours, or 57,000 years!
- Frequently combined with **NVRAM** to improve write performance
- Several improvements in disk-use techniques involve the use of multiple disks working cooperatively

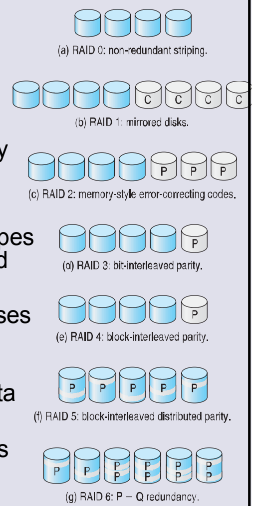
RAID

- **RAID** – multiple disk drives provides **reliability** via **redundancy**.
- ***Disk striping*** uses a group of disks as one storage unit →
  - Parallel access
- RAID schemes improve performance and / or reliability of the storage system
  - ***Mirroring*** or ***shadowing*** (RAID 1) keeps duplicate of each disk.
  - Can combine mirroring and striping (RAID 0+1 and RAID 1+0) →
  - ***Block interleaved parity*** (RAID 4,5,6) uses much less redundancy.
- 6 RAID levels

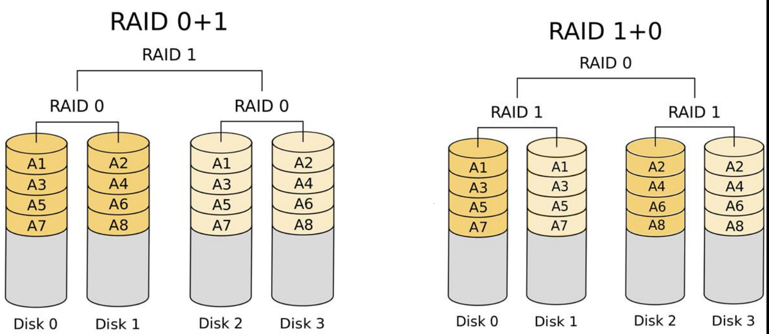


RAID (Cont.)

- Disk **striping** uses a group of disks as one storage unit
- RAID is arranged into six different levels
- RAID schemes improve performance and improve the reliability of the storage system by storing redundant data
  - **Mirroring** or **shadowing** (**RAID 1**)
  - Striped mirrors (**RAID 1+0**) or mirrored stripes (**RAID 0+1**) provides high performance and high reliability
  - **Block interleaved parity** (**RAID 4, 5, 6**) uses much less redundancy
- RAID within a storage array can still fail if the array fails, so automatic **replication** of the data between arrays is common
- Frequently, a small number of **hot-spare** disks are left unallocated, automatically replacing a failed disk and having data rebuilt onto them

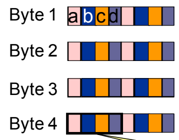


RAID (0 + 1) and (1 + 0)



RAID (0 + 1) and (1 + 0)

Non-redundant bit-level (below) or block-level striping (RAID 0)



….

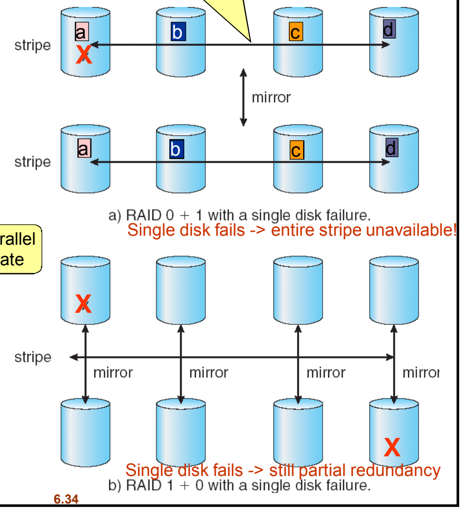
Accessed in parallel -> higher data rate

combined with

disk-mirroring (RAID 1)

Redundancy -> higher reliability but also high cost

RAID (0+1): Each stripe is mirrored



RAID – Other Features

- Regardless of where RAID implemented, other useful features can be added
- **Snapshot** is a view of file system before a set of changes take place (i.e. at a point in time)
- Replication is automatic duplication of writes between separate sites
  - For redundancy and disaster recovery
  - Can be synchronous or asynchronous
- Hot spare disk is unused, automatically used by RAID production if a disk fails to replace the failed disk and rebuild the RAID set if possible
  - Decreases mean time to repair

Tertiary Storage Devices

- Low cost is the defining characteristic of tertiary storage.
- Generally, tertiary storage is built using *removable media*
  - Magnetic tapes
  - CD-ROMs / DVDs
  - And other types

Magnetic Tapes

- Relatively permanent and holds large quantities of data
  - Evolved from open spools to cartridges
  - 20-200GB typical storage
- Kept in spool and wound or rewound past read-write head
- Once data under head, transfer rates comparable to disk
  - 140MB/s and greater
- Random access ~1000 times slower than disk
- Less expensive and holds more data than disk.
- Economical medium for purposes that do not require fast random access, e.g., backup, storage of rarely-used data, transfer medium between systems
- Large tape installations typically use robotic tape changers that move tapes between tape drives and storage slots in a tape library.
  - stacker – library that holds a few tapes
  - silo – library that holds thousands of tapes
- A disk-resident file can be *archived* to tape for low cost storage; the computer can *stage* it back into disk storage for active use.