Advantages and disadvantages of paging and segmentation computer science essay

Technology, Computer

To use the processor and the I/O facilities efficiently, it is desirable to maintain many processes, as possible, in main memory. In addition, it is desirable to free programmers from size restrictions in program development than to restrict them with small sizes (that happened in the older computers). The restriction to a predefined size redirects the programmers effort from the use of better programming techniques to a continuously effort to make fit in that size a solution, not necessarily the optimal one. The way to address both of these concerns is virtual memory (VM). Virtual memory systems are an abstraction of the primary memory in a von Neumann computer. Even in a time of decreasing physical memory costs, contemporary computers devote considerable resources to supporting virtual address spaces that are much larger than the physical memory allocated to a process. Contemporary software relies heavily on virtual memory to support applications such as image management with huge memory requirements. (Sami & Hamed , 2007) .

1. 1 Implementing Virtual Memory

To basic approaches to providing virtual memory are: paging and

segmentation.

Paging. With paging, each process is divided into relatively small, fixed-size pages. Paging systems transfer fixed-sized blocks of information between primary and secondary memories. Because of the fixed pages size and page frame size, the translation from a binary virtual address to a corresponding physical address is relatively simple, provided the system has an efficient table lookup mechanism. Paging systems use associative memories to implement page translation tables. Paging uses single-component addresses, like those used to address cell within any particular segment. In paging, the virtual address space is a linear sequence of virtual address (a format that differs from the hierarchical segmentation address space. In a paging system, the programmer has no specific mechanism for informing the virtual memory system about logical units of the virtual address space, as is done in segmentation. Instead, the virtual memory manager is completely responsible for defining the fixed-size unit of transfer – the page – to be moved back and forth between the primary and secondary memories. The programmer need not be aware of the units of virtual address space loaded into or unloaded from the physical memory. In fact, the page size is transparent to the process. (Philip , 1998) .

Segmentation. Segmentation provides for the use of pieces of varying size. It is also possible combine segmentation and paging in a single memory-management scheme. Segmentation is an alternative to paging. It differs from paging in that the unit transfer between primary and secondary memories varies. The size of the segments, are also explicitly known by the programmer. Translating a segment virtual address to a physical.

Segmentation is an extension of the ideas suggested by the use of relocation-limit registers for relocating and bound checking blocks of memory. The program parts to be loaded or unloaded are defined by the programmer as variable-sized segments. Segment may be defined explicitly by language directives it implicit by program semantics as the: text, data and stack segments created by the UNIX C compiler. Address is more complex that translating a paging virtual address. (Michael , 2008) .

1. 2 Process Management

Process management refers to the full spectrum of as services to support the orderly administration of a collection of processes. The processor manager is responsible for creating the environment in which the sequential process executes, including implementing resource management.

The community of processes that exists in the as at any given time is derived from the initial process that is created when the computer begins operation. The initial process boots up the as , which, in turn, can create other processes to service interactive users, printers, network connections and so on. A program image is created from a set of source modules and previously compiled library modules in relocate-able form. The link-editor combines the various relocate-able object modules to create an absolute program in secondary memory. The loader places the absolute program into the primary memory when a process executes the program. The program image, along with other entities that the process can reference, constitutes the process address space. The address space can be stored in different parts of the machine’s memory hierarchy during execution.

1. 3 compares their advantages and disadvantages of Paging and Segmentation

Advantages of Paging and Segmentation

Disadvantages of Paging and Segmentation

Paging

No external fragmentation

Segments can grow without any reshuffling

Can run process when some pages are swapped to disk

Increases flexibility of sharing

Segmentation

Supports sparse address spaces

– Decreases size of page tables

– If segment not used, not need for page table

Increases flexibility of sharing

of Both

Increases flexibility of sharing

– Share either single page or entire segment

Overhead of accessing memory

aˆ? Page tables reside in main memory

aˆ? Overhead reference for every real memory reference

Large page tables

aˆ? Must allocate page tables contiguously

aˆ? More problematic with more address bits

Page table size

– Assume 2 bits for segment, 18 bits for page number, 12 bits for offset

2. 0 Mapping Function

Algorithm to block the memory card side cache lines. Method Which country is necessary to define a cache block busy. Three techniques used: direct, associative and associative.

Associative Mapping

In associative mapping, when a request is made for cash, the requested address is compared in the same directory with all entries in the directory. If the requested address is found (directory hit), the appropriate place in the cache is fetched and returned to the processor, otherwise, a miss occurs. (figure 1) .

Associative Mapping Cache Figure (1), (Philip , 1998)

Associative Mapping Summary

Address length = (s+w) bits

Number of addressable units = 2^(s+w) words or bytes

Block Size = line size = 2^w words or bytes

Number of blocks in main memory = 2^(s+w)/2^w = 2^s

Number of lines in cache = undetermined

Size of tag = s bits

Associative Mapping Pros and Cons

Flexibility as to which block to replace when a new block is read into cache

Replacement algorithms designed to maximize cache hit

ratio

Complex circuitry required to examine the tags of all cache lines in parallel

direct mapping

In a direct mapping cache Lower Row address bits are used to access the directory. Several address line card in the same place in the cache directory, upper address bits (tag bits) should be compared with address to ensure a hit. If the comparison is not valid, the result is a cache miss, or simply a miss. The address given to the cache by the processor actually is subdivided into several pieces, each of which has a different role in accessing data (figure 2) .

Direct Mapping Cache Figure (2), (Philip , 1998)

set associative Mapping

Operates in a fashion somewhat similar to the direct-mapped cache. Bits from the line address are used to address a cache directory. However, now there are multiple choices: two, four, or more complete line addresses may be present in the directory. Each of these line addresses corresponds to a location in a sub-cache. The collection of these sub-caches forms the total cache array. In a set associative cache, as in the direct-mapped cache, all of these sub-arrays can be accessed simultaneously, together with the cache directory. If any of the entries in the cache directory match the reference address, and there is a hit, the particular sub-cache array is selected and out gated back to the processor (figure 3) (William , 2000)

Set Associative Mapping Cache Figure (3) ,(Philip , 1998)

2. 4 Replacement Algorithms

Direct Mapping

No choice

Each block only maps to one line

Must replace that line

Associative and Set Associative.

Must be implemented in hardware for speed.

Most effective – Least Recently Used (LRU)

Replace the block in the set that has been in cache the longest with no references to it .

2-way set associative – each line includes a USE bit .

First-in-first-out (FIFO)

Replace the block in the set that has been in the cache the

longest.

Uses a round-robin or circular buffer technique .

Least Frequently Used (LFU) .

Replace the block in the set that has experienced the fewest

references.

Associate a counter with each line

Pick a line at random – not based usage .

Only slightly inferior in performance to algorithms based on

usage .

3. 0What is RAID

The basic idea of RAID (Redundant Array of Independent Disks) is to combine multiple cheap disks in an array of disk drives to obtain performance, capacity and reliability that exceeds that of a large disk. The array of drives appears to the host computer as one logical drive.

The Mean Time Between Failure (MTBF) of the array is equal to the MTBF of an individual drive, divided by the number of drives in the array. Because of this, the MTBF of a non-redundant array (RAID 0) is too low for mission-critical systems. However, disk arrays can be made fault tolerant by redundantly storing information in various ways.

Five types of array architectures, RAID 1 to RAID 5 were originally determined each provides disk fault tolerance with different compromises in features and performance. In addition to these five redundant array architectures, it has become popular to refer to a non-redundant array of disk drives as a RAID 0 array. RAID 0 is the fastest and most efficient array type but offers no fault tolerance. RAID 0 requires a minimum of two drives. (William , 2000).

3. 1 Performance and Data Redundancy

Increasing Logical Drive Performance Without an array controller, connecting

extra physical disks to a system increases the total storage capacity. However,

it has no effect on the efficiency of read/write operations, because data can

only be transferred to one physical disk at a time (see Figure 3).

Figure (3) ,(William , 2000)

With an array controller, connecting extra physical disks to a system increases both the total storage capacity and the read/write efficiency. The capacity of several physical disks is combined into one or more virtual units called logical drives (also called logical volumes). The read/write heads of all of the physical disks in a logical drive are active simultaneously; improving I/O performance and reducing the total time required for data transfer (see Figure 4). (William, 2000)

Figure (4), (William , 2000)

Because the read/write heads for each physical disk are active simultaneously, the same amount of data is written to each disk during any given time interval. Each unit of data is called a block. The blocks form a set of data stripes that are spread evenly over all the physical disks in a logical drive (see Figure 5), (William, 2000).

Figure (5) ,(William , 2000)

For data in the logical drive to be readable, the data block sequence must be the same in every stripe. This sequencing process is performed by the Smart Array Controller, which sends the data blocks to the physical disk, writing the heads in the correct order. In a striped array, each physical disk in a logical drive contains the same amount of data. If one physical disk has a larger capacity than other physical disks in the same logical drive, the extra capacity cannot be used. A logical drive can extend over more than one channel on the same controller, but it cannot extend over more than one controller. Disk failure, although rare, is potentially catastrophic to an array. If a physical disk fails, the logical drive it is assigned to fails, and all of the data on that logical drive is lost. (Peng, Hai , Xinrong , Qiong & Jiangling , 1997) .

3. 2 differences among all RAID levels

RAID 0 is the fastest and most efficient array type but offers no fault tolerance.

RAID 0 requires a minimum of two drives.

RAID 1 is the best choice for performance-critical, fault-tolerant environments. RAID 1 is the only choice for fault-tolerance if no more than two drives are used.

RAID 2 is seldom used today since ECC is embedded in all hard drives.

RAID 2 is not supported by Adaptec RAID controllers.

RAID 3 can be used to speed up data transfer and provide fault tolerance in single-user environments that access long sequential records. However, RAID 3 does not allow overlapping of multiple I/O operations and requires synchronized-spindle drives to avoid performance degradation with short records. Because RAID 5 with a small stripe size offers. Similar performance, RAID 3 is not supported by Adaptec RAID controllers.

RAID 4 offers no advantages over RAID 5 and does not support multiple simultaneous write operations. RAID 4 is not supported by Adaptec RAID controllers.

RAID 5 combines efficient, fault-tolerant data storage with good performance characteristics. However, write performance and performance during drive failure is slower than with RAID 1. Rebuild operations also require more time than with RAID1 because parity information is also reconstructed. At least three drives are required for RAID 5 arrays.

RAID-6 Striped data with dual distributed parity

RAID-6 is the same as RAID-5 except that it uses a second level of independently calculated and distributed parity information for additional fault tolerance. This extra fault tolerance provides data security in the event two drives fail before a drive can be replaced.

While this RAID level does provide greater fault tolerance than level 5, there is a significant loss in write performance due to the requirement for storing parity twice for each write operation. A RAID-6 configuration also requires N+2 drives to accommodate the additional parity data, which makes it less cost effective than RAID-5 for an equivalent storage capacity.

RAID 10 Stripe set of mirrored arrays

RAID 10 (also called RAID 0/1) is a combination of RAID levels 0 and 1. In this type of implementation a RAID-0 stripe set of the data is created across a 2-disk array for performance benefits. A duplicate of the first stripe set is then mirrored on another 2-disk array for fault tolerance. While this configuration provides all of the performance benefits of RAID-0 and the redundancy of

RAID-1, this level is very costly to implement because a minimum of four disks are necessary to create a RAID 10 configuration.

NOTE A RAID 10 configuration can continue operations even when two disks have failed, provided that the two disks not part of the same RAID-1 mirror set.

RAID 50 Stripe set of parity arrays

RAID level 50 (also called RAID 0/5) is a combination of RAID levels 0 and 5. Multiple RAID-5 arrays are striped together using RAID-0. Parity is maintained separately for each RAID-5 group in the striped array. This level provides the same advantages of RAID-5 for small data transfers with the added performance of striping for disk read/write operations. Also, because parity is calculated independently for each RAID-5 component, if one array is degraded the effect on overall operations is not as significant as for a single RAID-5 array.

However, the overhead incurred by RAID-5 parity generation is still present. Normally this does not cause noticeable degradation unless you are dependent on software-based XOR functionality or have a large number of disks in the array. RAID subsystems that support hardware-based XOR should provide performance nearly equal to a RAID-0 configuration with the added protection of data parity information in the event of a disk failure.

A minimum of six disks are required for a RAID 50 configuration.

NOTE A RAID 50 configuration can continue operations even when two disks have failed, provided that the two disks are not part of the same RAID-5 parity group.(Adaptec inc. (n. d.)) .