Study on cpu and memory hierarchy

Technology, Computer

CPU must have compatibility with the memory in the computer system. Memory cannot increase the speed of the processor however it can help the processor increase its performance. If the CPU doesn’t get the data it requires, it would sit idle wasting CPU clock cycles that would decrease the overall throughput and execution of the processes. If data needs to be accessed to and from hard disk which is slower as compared to the main memory, more and more cycles are wasted decreasing the efficiency of the system.

Conclusion- For better performance of the system, faster program execution and proper CPU utilization the compatibility between CPU and memory is required.

A computer employs RAM chips of 256*8 & ROM chips of 1024*8. The computer society system needs 2K byte of RAM & 4KBYTE of ROM, & 4 interface units. Each with 4 registers. A memory mapped I/O configuration is used. The 2 highest order bits of the address bus are connected with address 00 or RAM and 01 for ROM & 10 for interface. How many RAM & ROM chips are needed? Draw a memory address map.

2 kb of RAM required i. e. 2? 1024(bytes) = 2048 bytes (Since 1 kb = 1024 bytes)

RAM = = 8 chips; and

4kb of ROM is required i. e. 4 x 1024 = 4096 bytes

Therefore ROM = = 4 chips; and

There are 4 interfaces each having 4 register, So total no. of registers is 4? 4= 16 registers;

Memory address map

Cache Coherence-

Caches allow greater performance by storing frequently used data. In multiprocessing system, each processor is provided with its own cache and they all share the same memory or address space so it is possible for more than one processor to access a data item at a time. What if one processor updates the data item without informing the other processors, inconsistencies may result and cause incorrect executions and the problem of inconsistencies is known as Cache Coherence in computing.

The coherence of caches is obtained if the following conditions are met. However these read and write operations are said to be one after another and this is not possible due to memory access latency and a write by first processor may not be seen by a read from second processor if the read is made within a very small time after the write has been made.

Case 1 Case 2

Processor P1 reads memory location X and then writes back to same location X while no other processor is not accessing the memory location X.

Processor P1 reads and then processor P2 writes to and from same location X and currently the location would return value written by processor P2 only.

Processor P1 and P2 writes to same memory location X in a sequence and currently the value returned would be decided as per the sequence.

Mechanisms-

Bus Snooping- In Bus Snooping each cache is connected through the same bus and it is where every CPU knows who has a copy of its cached data. So each CPU watches continually for write activity concerned with data addresses which it has cached. This assumes that all communication can be seen by all. However it is far more complex to implement.

Directory Based Approach- In a directory-based system, the data being shared is placed in a common directory that preserves the coherence connecting caches. The directory acts as a filter through which the processor must ask authorization to fill an entry from the primary memory to its cache. When an entry is distorted the directory either updates or invalidates the other caches with that entry.

The MESI protocol is the best suited protocol to avoid cache coherence, where M stands for MODIFIED, E stands for EXCLUSIVE, S stands for SHARED and I stands for INVALID.

Write Back Cache-

Cache uses two approaches to write data back to main memory.

Write Through

Write Back

It is the simplest one in which all write all operations are made to the main memory as well as to cache; ensuring main memory is always valid. Any other CPU- cache module can monitor traffic to main memory to update the data in its own cache, but always results in substantial memory traffic.

It minimizes memory writes. In write back method modifications to data in the cache aren’t copied to the cache source until absolutely necessary. It is also known as copy back cache . In write back updates are made only in the cache. When an update occurs UPDATE bit are set associated with the slot and when the block is replaced it is checked whether the UPDATE bit is set or not. If it is set then data is written back to main memory.

For Example- Intel processors since the 80486 uses back caching.

Problem with this kind of implementation is that performance improvement comes with a slight threat that data may be vanished if the system crashes and more complex circuitry.

Onboard Cache-

Cache is a part of multi-level storage strategy which is used to increase the performance of CPU by providing a bridge in between the slower memory RAM and CPU. The cache that is the part of the CPU is known as off-board cache and the cache which is present on the motherboard is known as on-board cache. Generally L1 cache is referred as off-board and L2 is known as on-board. Sometimes L3 cache is also present on the motherboard along with L2. Now a day’s specific CPU vendors incorporates L2 as a part of CPU and L3 on motherboard.

Implementation of Cache-

In Cache, latency needs to be decreased and hit rate needs to be increased. Larger caches have better hit rates but longer latency. To address this problem, many computers use multiple levels of cache. The smaller and faster one is L1 cache built inside the CPU known as on-chip. If CPU needs data it first checks in L1; if it hits the processor proceeds at high speed.

If the smaller cache misses, the next larger cache (L2) is checked, and so on, before external memory is checked. As the latency difference between main memory and the fastest cache has become larger, some processors have begun to utilize as many as three levels of on-chip cache. For Example- Intel’s Xeon MP Product code-named “ Tulsa”, AMD Phenom II (2008), Intel Core i7 (2008) uses unified L3 cache. However Cache can be implemented by using Direct Mapped, Associative Mapping or Set-Associative Mapping.

Virtual Memory-

For the execution of programs memory required is more than what is actually present. So, the technique used to overcome this size limitation is Virtual Memory which is illusion of memory not physically present. The purpose is to allow multiple programs share same memory allowing splitting up of program into smaller pieces that can be loaded into different parts of memory whenever space can be found.

Implementation of Virtual Memory-

It is implemented using two techniques- one is Demand Paging and other one is Demand Segmentation.

CPU generates address which is not physically present. These are the program addresses referred to as logical addresses, they don’t have any existence outside the program, the actual memory addresses are known as physical addresses. These virtual addresses are mapped or interchanged to its corresponding physical address through a process known as mapping. A page table or look up table is maintained for this purpose.

In Demand paging, valid-invalid bit scheme is used in which a valid-invalid bit is associated with each page. 1 for the page in memory and 0 for not present in memory. During address translation if bit in entry is 0 the page fault occurs. In virtual memory process are divided into chunks of equal size known as pages and chunks of memory in which pages are loaded are known as frames.

In Demand Segmentation each row of the lookup table contains a starting address for a logical block of memory, together with the size of the block and a corresponding starting address in physical memory. Paging and Segmentation operates both the same.

Problem of Fragmentation-

Fixed Memory Partitioning- Operating system occupies fixed portion of main memory and partitions are created for multiple processes but not of same size, so there will be wastage of memory. In most cases the process will not acquire memory provided to it.

Variable Memory Partitioning- In variables-size partitions, the memory allocated is as much it is required by process. However when processes are swapped in, small holes are created leading to problem of fragmentation. Compaction is done to solve problem, but it waste CPU time.

In Virtual Memory demand paging method is implemented, in which memory is partitioned into equal chunks that are relatively small, and each process is divided into small fixed size chunks of some size. The lists of the frames that are free are maintained by the operating system. As the size of the pages and frames are same so suffer less fragmentation problem.

The Memory Hierarchy

The design constraints on a computer’s memory can be summed up by three questions: how much memory is available, how fast it is and how much it will cost? Following are the relationships between these tradeoffs-

Smaller access time, greater cost per bit.

Greater capacity, smaller cost per bit.

Greater capacity, greater access time.

Access Time Increase

Transfer Rate DecreasesCPU Registers

Cache

Cost per/bit Decreases

Capacity IncreasesRAM

Magnetic Disk

Figure -Memory Hierarchy

Memory hierarchy helps in increasing the performance of processor, without hierarchy, faster process won’t help and all time waiting on memory, It provides a large pool of memory that costs as much as the cheap storage near the bottom of the hierarchy, but that serves data to programs at the rate of the fast storage near the top of the hierarchy. It provides a faster access of data stored in the memory. If it is understand how the system moves data up and down the memory hierarchy, then application programs can be written so that data items are stored higher in the hierarchy, where the CPU can access them more quickly.

Addressing modes affecting performance of system-

It simplifies the memory references, produces variable length instruction format and instruction manipulates operands in memory directly. It adds convenience and flexibility to have modes of addressing, and it allows a large range of addressable memory while using a reasonable number of bits. Addressing modes make it easier to write certain type of programs such as loops that uses an index to address different entries in a table or array. For Example- Indexed Addressing. Now a day’s computer allows programmer accessible registers that manipulate data directly between registers.