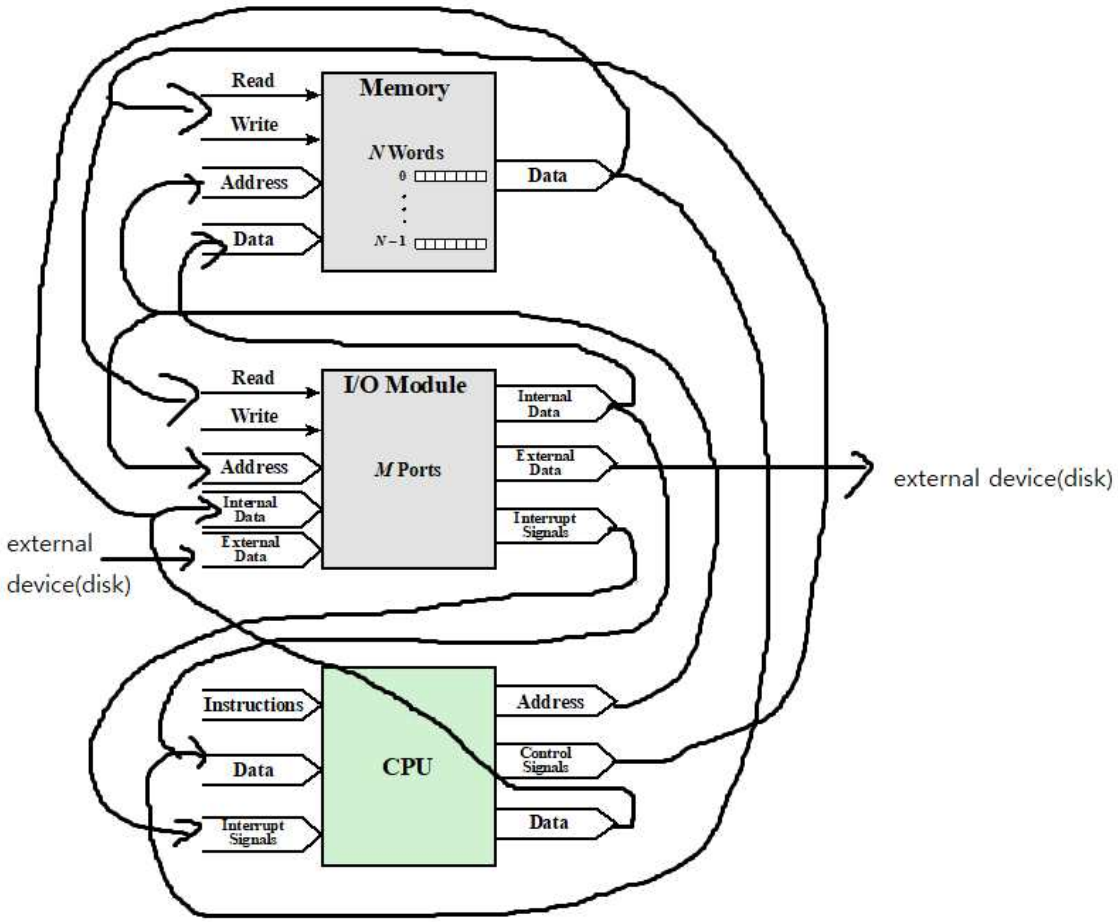
과목명	컴퓨터구조
담당교수	조경산 교수님
학과	소프트웨어학과
학번	32153180
이름	이상민
제출일자	2018.10.17

1. Show the interconnection of three modules in Figure 3.15



-> Memory : data->CPU
I/O Module : Interrupt Signals->CPU
CPU : Address->Memory, I/O Module
Control Signals->Memory(R,W), I/O Module(R/W)
Data->Memory

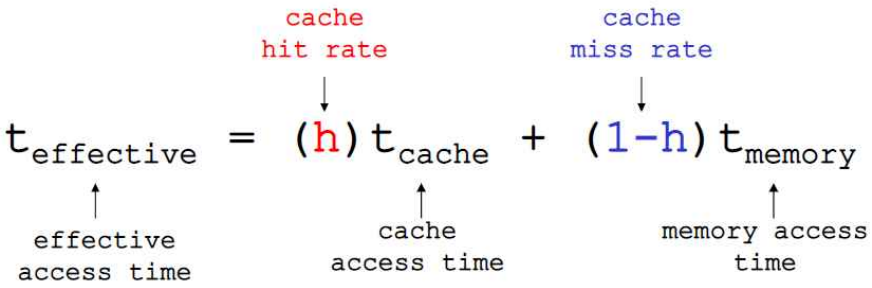
2. Explain the locality of memory reference used in the design of cache memory
-> 프로그램은 실행 시 순차적으로 실행된다. 즉, 100번지에 있는 내용을 처리했다면 그다음 실행될 부분은 101번지에 있을 가능성이 높다. 따라서 CPU에서 실행할 때 프로그램 전부가 아닌, 지금 실행하고 있는 곳의 주변 지역만 적재시키더라도 프로그램 실행에 큰 지장이 없다. 이러한 것을 메모리 참조의 지역성(locality of memory)라고 한다. locality of memory로는 spatial locality아 temporal locality가 있다.
spatial locality는 근처 정보를 참조하는 것이다. 쉽게 말해 CPU가 main memory의 500번지에 있는 데이터를 가져오길 원한다면 main memory는 cache memory에 500번지에 있는 데이터만 복사해두는 것이 아니라 501, 502, 503번지에 있는 데이터까지 복사해두는 방법이다. temporal locality는 main memory에서 cache memory로 자주 쓰는 부분들을 통째로 복사해두는 방법이다.

3. Explain the hierarchical memory system

-> 메모리에서 고려해야 할 것은 3가지로 access time, capacity, cost가 있다. access time은 CPU가 요구하는 만큼 적어야 좋고, capacity는 program이 요구하는 만큼 커야 좋다. 마지막으로 cost는 사용자가 요구하는 만큼 싸야 좋다. 이것들을 만족시키기 위해서 메모리는 cache memory, main memory, virtual memory가 계층 구조를 이룬다. virtual memory는 매우 크다고 가정을 하는데, 흔히 disk가 여기에 해당한다.

4. If the effective access time of main memory=20ns, the cache access time=10ns, what is the cache hit ratio? (Main memory access time=100ns)

-> access time을 구하는 식은 아래와 같다.



문제에서 준 값들을 대입해보면, 20=h*10+(1-h)*100=100-90*h이므로 cache hit ratio인 h=8/9이다.

5. Consider a byte addressable main memory of 256bytes, block size of 8bytes, and a cache consisting of 8 lines

1) Specify the number of bits of memory address?

-> main memory의 주소지정 가능한 byte 수가 256개이다. $256=2^8$이므로 8bits이다.

2) How many bits in memory address are used for TAG in direct, associative and 2-way set associative mapping

-> block size가 8bytes이므로 $8=2^3$, word는 3bits이다. cache는 8 lines이므로 $8=2^3$, line도 3bits이다.

direct : direct mapping의 memory address는 tag, line, word의 합으로 이루어진다. 따라서 memory address(8bits)=tag+line(3bits)+word(3bits)이므로 tag는 2bits이다.

associative : associative mapping의 memory address는 tag와 word의 합으로 이루어진다. memory address(8bits)=tag+word(3bits)이므로 tag는 5bits이다.

2-way associative : cache는 8 lines인데 이것은 set 속에 있는 line 수와 set 수의 곱으로 나타낼 수 있다. set 속에 있는 line 수는 2이므로 set 수는 $4=2^2$이므로 set는 2bits다. 2-say associative의 memory address는 tag, set, word의 합으로 이루어진다. 따라서 memory address(8bits)=tag+set(2bits)+word(3bits)이므로 tag는 3bits이다.

3) Compare the mapping process of direct and associative mapping

-> mapping은 cache memory에서만 사용하는 방식으로 physical address가 cache memory에 있는지 없는지, 만약 있다면 어디에 있는지를 알아내는 방법이다.

direct mapping은 주소 접근에 용이하지만 hit ratio가 낮다. 또 CPU가 원하는 정보가 항상 없을 수 있다. 반면에 associative mapping은 main memory가 cache memory의 비어있는 아무 곳에 mapping을 하는 것인데, hit ratio가 높다. 그러나 복잡하다는 단점이 있다.

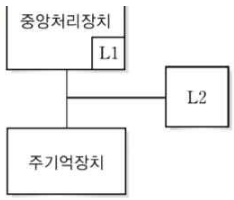
6. Explain the case when write through policy is better than write back policy?
-> write through policy는 가장 단순한 기술로 모든 writing 작업이 cache memory뿐만 아니라 main memory에서도 수행되는 것을 말한다. 반면에 write back policy는 cache memory에서만 writing 작업을 하고 main memory에서는 나중에 replacement 할 때 작업이 이루어진다. write back policy는 cache memory data와 main memory data가 달라질 경우도 있기 때문에 항상 서로 같은 내용을 유지하고 싶으면 write through policy를 사용하면 된다.

7. Explain advantages of a split cache compared to a unified cache
Which of L1, L2, and L2 cache is proper for split cache?
-> unified cache는 Von Neumann Architecture로 program과 data가 같은 memory 안에 존재하는 것을 말한다. 반면에 split cache는 Havard Architecture로 program과 data가 다른 memory에 존재하여 instruction fetch/decode unit, execution unit 사이의 cache contention(캐시 경합)을 피할 수 있다. 근래에 컴퓨터들은 L1 cache를 split cache로, L2 cache를 unified cache로 사용한다.

8. For a system with 2 levels of cache, Tc1=1st level cache access time, Tc2=2nd level cache access time, Tm=main memory access time, H1=1st level cache hit ratio, H2=2nd level cache hit ratio.
Provide an equation for effective memory access time Ta for a read operation
->

- 계층적 캐시의 구조에서 평균 기억장치 접근시간

$$T_{average} = H_{L1} \times T_{L1} + (H_{L2} - H_{L1}) \times T_{L2} + (1 - H_{L2}) \times T_{main}$$



- $T_{average}$ = 평균 기억장치 접근시간
- T_{main} = 주기억장치 접근시간
- T_{L1} = L1 캐시기억장치 접근시간
- T_{L2} = L2 캐시기억장치 접근시간
- H_{L1} = L1 캐시 적중률
- H_{L2} = L2 캐시 적중률

Ta = H1*Tc1 + (H2-H1)*Tc2 + (1-H2)*Tm
위의 식을 해석해보면 다음과 같다. 처음에 L1 cache에 접근해 원하는 정보가 없을 경우 L2 cache로 접근하고, 거기에도 없을 시 main memory로 접근한다.

9. If 4 clock cycles(address, read, wait, data in each clock) are required for memory read operations in a synchronous bus (clock frequency : 2GHz) with 16 data lines(16bits), calculate the maximum memory transfer rate (in bps)

-> memory transfer rate = clock frequency * data lines / clock cycles

= 2G(clock cycles/sec) * 16(bits) / 4(clock cycles)

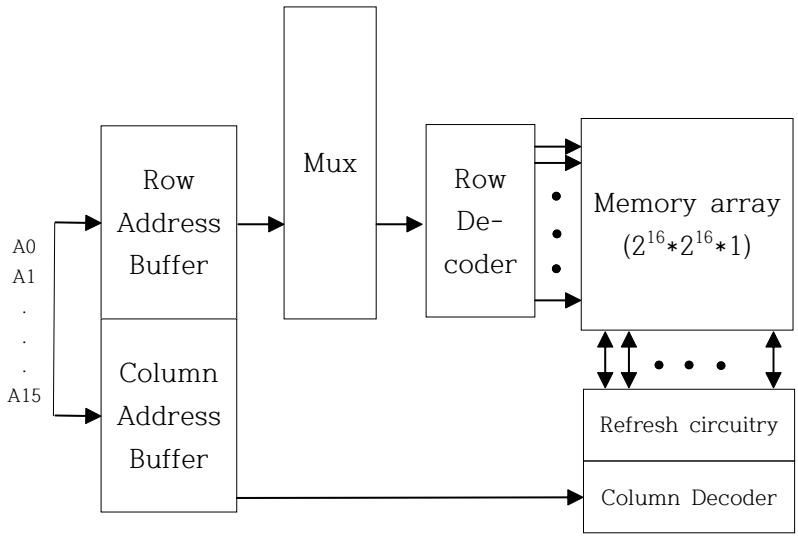
= 8G(bits/sec)

따라서 maximum memory transfer rate는 8Gbps이다.

10. The memory(8GB) of a computer is built from 4G*1 DRAM chips

1) How many DRAM chips are needed? Show the interconnection between the memory chips and the system bus

-> DRAM 4G*1bit=2^2*2^{30}b=2^{32}bit이고, 8GB=2^3*2^{30}*2^3=2^{36}이므로 2^4=16개가 더 필요하다.



2) Each row of a DRAM chip must be refreshed at least once every 1msec. What is the time period between successive refresh requests?

-> 4G*1 DRAM chip에서 4G=2^{32}이기 때문에 DRAM address bus의 개수는 32개의 절반인 16개이다. 매 1msec마다 refresh되기 때문에 time period는 16msec이다.

11. For a disk with average seek time=4ms, 6000rpm, 512bytes/sector, 500setctors/track; We wish to read a file consisting of 250 sectors. Estimate the total time for reading the file. Assume the file is stored as compactly as possible on the disk.

-> rpm은 revolution per minute의 약자로 1분동안의 회전수를 보여준다. 6000rpm은 1분에 6000번 회전을 하고, 이것은 1초에 100번을 회전한다는 뜻이다. 1번 회전을 하는 데 10msec가 걸리기 때문에 rotational latency는 절반인 5msec이다.

track 1개 당 500개의 sector가 있으니 transfer time = 10msec/500 = 0.02msec이다. 문제에서 file이 빽빽하게 저장되어있다 했으니 250개의 sector를 읽는 데 걸리는 시간은 250*0.02msec=5msec이다. 따라서 총 걸리는 시간은 4+5+5인 14msec이다.

12. How dose an external device indicate an Interrupt event to the processor? How does the processor response to this event?

-> interrupt는 현재 실행 중인 프로그램을 중단하고 다른 프로그램의 실행을 요구하는 명령어이다. 그로 인해 시스템의 처리 효율을 높이고, 프로그램의 실행 순서를 바꿔가며 처리할 수 있어 다중 프로그래밍에 사용된다.

external device는 processor에게 하드웨어 신호를 보냄으로써 interrupt가 발생했음을 알린다. processor는 A 프로그램 실행 중 B 프로그램을 위한 interrupt가 발생하면 현재 자신이 가지고 있던 register의 모든 상태를 스택 혹은 프로세스 제어 블록에 저장하고 interrupt handler를 부른다. 이렇게 B 프로그램 작업이 모두 끝나면 전에 저장해둔 스택에서 A의 상태를 processor에 복구한 뒤 다시 실행한다.

이러한 interrupt가 존재하지 않는다면 오버헤드가 급증하고, processor를 낭비하게 된다.