20160074 화학공학과 고진민

[컴퓨터 SW시스템 개론](http://lms.postech.ac.kr/Main.do?cmd=viewCourseMain&mainDTO.courseId=20180920140072901661213&gubun=class_learner) HW4

1. (5.17)

소스코드 참조요망.

1byte의 char형으로 작업을 수행하던 것을 kbyte로 효율을 k배 높인다.

1. (6.24)
2. 2MB이므로 512byte짜리 섹터로 환산하면 2048개의 sector에 해당한다. Disk sector가 읽을 순서대로 바로바로 저장이 된 best case라면, 1개의 track 당 2000개의 sector가 있으므로 2000개를 읽은 뒤 다음 track으로 넘어라 48개를 읽는다. 이 때 읽을 sector가 찾을 필요 없이 바로 다음에 있으므로 Tseek를 0으로 간주한다. 그러면 전체 읽는 시간은 1바퀴+48/2000바퀴만큼의 회전시간이다. 18000RPM이므로 1바퀴에 1/300초가 걸린다. 계산하면, 0.00341초, 즉 **3.41 ms**가 걸린다.
3. 2048번의 seek가 진행되어야 한다. 한 번의 sector를 읽을 때는 Tavg rot + Tavg seek + Tavg transfer = 1/300/2 + 0.008 + 1/300/2000 = 0.00967초, 즉 9.67 ms가 걸린다. 여기에 2048을 곱하면, **19800 ms**가 걸린다.
4. (6.28)
5. S = 8, B = 4, E = 2이므로 s = 3bit, b = 2bit, t = 8bit이다. Set 2에는 valid cache가 없으므로, hit이 일어날 수 없다.
6. Set 4를 hit 하기 위해서는 s가 100, t = 11000111(C7) 또는 00000101(05)이다. Block offset은 어떤 거여도 상관 없으므로, 11000111100XX 혹은 00000101100XX이다. X는 임의의 bit이다. 이를 16진수로 바꾸면, 0x18F0~0x18F3, 0x00B0~0x00B3 이다.
7. Set 5을 hit 하기 위해서는 s가 101, t = 01110001(71) 이다. 전체 주소는 01110001101XX이다. 16진수로 바꾸면 0x0E34~0x0E37이다.
8. Set 7을 hit 하기 위해서는 s가 111, t = 11011110(DE) 이다. 전체 주소는 11011110111XX이다. 16진수로 바꾸면 0x1BDC~0x1BDF이다.
9. (6.34)

Cache가 전체 32byte이므로 각 array가 16byte씩 나눠쓴다. Row 단위로 받는다고 하면 처음에 각각 Row 0의 데이터를 받지만 src는 계속 row가 바뀌므로, 매번 miss가 뜨고 dst는 row가 바뀔 때만 miss가 뜬다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| dst | Col. 0 | Col. 1 | Col. 2 | Col. 3 |
| **Row 0** | m | h | h | h |
| **Row 1** | m | h | h | h |
| **Row 2** | m | h | h | h |
| **Row 3** | m | h | h | h |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| src | Col. 0 | Col. 1 | Col. 2 | Col. 3 |
| **Row 0** | m | M | M | m |
| **Row 1** | m | M | M | M |
| **Row 2** | m | m | m | M |
| **Row 3** | m | m | m | M |

1. (6.35)

Cache가 전체 128byte이므로 각 array가 64byte씩 나눠 쓴다. 이 경우는 둘 모두 array 전체를 저장할 수 있으므로 각각 처음에 한 번 miss가 뜬 뒤 계속 hit이 뜰 것이다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| dst | Col. 0 | Col. 1 | Col. 2 | Col. 3 |
| **Row 0** | M | h | h | h |
| **Row 1** | H | h | h | h |
| **Row 2** | H | h | h | h |
| **Row 3** | H | h | h | h |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| src | Col. 0 | Col. 1 | Col. 2 | Col. 3 |
| **Row 0** | M | h | h | h |
| **Row 1** | H | h | h | h |
| **Row 2** | H | h | h | h |
| **Row 3** | H | h | h | h |

1. (7.6)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Symbol | Swap.o .symtab? | Symbol type | Module defined | Section |
| buf | yes | Extern | m.o | .data |
| bufp0 | Yes | Global | Swap.o | .data |
| bufp1 | Yes | Global | Swap.o | .bss |
| swap | Yes | Global | Swap.o | .text |
| temp | No | - | - | - |
| incr | Yes | Global | Swap.o | .text |
| count | No | - | - | - |

1. (7.8)
2. 여기서는 **error**가 발생한다. 둘 다 strong definition이기 때문이다.
3. 여기서는 **unknown**이다. 이는 두 경우 모두 weak definition이기 때문이다.
4. 여기서는 **error**가 발생한다. 둘 다 strong definition이기 때문이다.
5. (7.12)
6. ADDR(s) = ADDR(.text) = 0x4004e0

ADDR(r.symbol) = ADDR(swap) = 0x4004f8

refaddr = ADDR(s) + r.offset = 0x4004ea

\*refptr = (unigned) (ADDR(r.symbol) + r.addend - refaddr)

= (unsigned) (0x4004f8 – 0x4 – 0x4004ea) = **0xa**

1. ADDR(s) = ADDR(.text) = 0x4004d0

ADDR(r.symbol) = ADDR(swap) = 0x400500

refaddr = ADDR(s) + r.offset = 0x4004da

\*refptr = (unigned) (ADDR(r.symbol) + r.addend - refaddr)

= (unsigned) (0x400500 – 0x4 – 0x4004da) = **0x22**