2014-2학기 컴퓨터구조 기말시험 채점 기준표	
1. 20점	**TLB hit(총 6점)** Program의 모든 address는 **virtual address**이다. 따라서 **TLB**를 참조해서 **virtual address**를 **physical address**로 변환해야 한다. **(2점)** **TLB hit**이면 physical address로 **first-level cache**를 참조하고 cache miss이면 **second-level cache**를 참조하고 (and so on), 마지막 level의 cache에서 cache miss가 발생하면 **DRAM을 access**한다. **(2점)** 이 과정은 hardware에 의해 처리된다. **(2점)** **TLB miss(총 2점)** TLB에 entry가 없으면 **TLB miss(exception)**가 발생하고, OS는 **page table에서 entry를** TLB로 가져온다. 그리고 return from exception 후 instruction을 재실행하면 TLB hit이 되고, 이후 과정은 TLB hit일 경우와 같다. **(2점)** HW 처리도 가능하다고 언급하면 **bonus 1점** **Page fault(총 12점)** TLB miss일 때 main memory에 해당 page가 없으면 **page fault**가 발생하고 이후 과정은 **OS에 의해 처리**된다. **(2점)** OS는 disk를 access해서 main memory로 data를 가져와야 한다. 그런데 이 작업이 오래 걸리기 때문에 OS는 현재 process를 wait queue(**wait state**)로 옮긴다. **(2점)** 필요하면 replacement가 되어야 한다고 언급하면 **bonus 1점** 그리고 **context switch**를 해서 ready queue(**ready state**)에서 대기 중인 다른 process를 실행한다. **(2점)** **Disk의 I/O** 동작이 끝나면 **interrupt**가 발생하고, OS는 page table을 update한 뒤 wait queue에 있는 process를 ready queue로 옮긴다. **(2점)** 그리고 ready queue에서 대기 중인 process는 **process scheduling**에 의해 차례가 되면 CPU를 할당 받는다. **(2점)** Page fault시 page table만 update되었으므로 process가 재실행되면 TLB miss가 발생하고, 이후 과정은 TLB miss일 경우와 같다. 단, 이 경우 first-level, second-level cache에서 모두 cache miss가 발생한다. 따라서 DRAM을 access해서 data를 second-level cache, first-level cache로 가져와야 한다. **(2점)**
2. 10점	**(a)** 캐시의 크기가 커지면 hit time이 증가하지만**(1.5점)** miss rate은 감소한다.**(1.5점) (총 3점)** **(b)** 블록의 크기가 일정 수준 까지 커지면 miss rate은 줄어들지만**(1.5점),** 일정 수준이 지나면 miss rate가 다시 증가한다**(1점),** miss penalty가 증가한다.**(1.5점) (총 4점)** **(c)** set associativity가 커지면 hit time이 증가하지만**(1.5점)** miss rate은 감소한다.**(1.5점) (총 3점)**

2014-2학기 컴퓨터구조 기말시험 채점 기준표

3. 10점

virtual address: 0x0921 (총 5점)

A. (3점, 항목 당 0.5점)

Parameter	Value
VPN	0x24
TLB index	0x0
TLB tag	0x9
TLB hit?(Y/N)	Y
Page fault(Y/N)	N
PPN	0x0D

B. (2점, 항목 당 0.4점)

Parameter	Value
Byte offset	0x1
Cache index	0x8
Cache tag	0x0D
Cache hit(Y/N)	N
Cache byte returned	-

virtual address: 0x0000 (총 5점)

A. (3점, 항목 당 0.5점)

Parameter	Value
VPN	0x00
TLB index	0x0
TLB tag	0x0
TLB hit?(Y/N)	N
Page fault(Y/N)	N
PPN	0x28

B. (2점, 항목 당 0.4점)

Parameter	Value
Byte offset	0x0
Cache index	0x0
Cache tag	0x28
Cache hit(Y/N)	N
Cache byte returned	-

4. 10점

SPEC benchmark에서 결과를 나타내는 방법(총 6점)

Target system의 base runtime 및 runtime을 측정한다. **(2점)** Base runtime과 runtime의 차이점을 설명한다. **(1점)**

Reference time에 대한 base ratio와 ratio를 계산한 후 **(2점)** 최종 결과 값인 각 ratio의 geometric mean을 구한다. **(1점)**

SPEC2006에서 update된 내용(총 4점)

Multi-core 환경에서의 throughput 측정을 위한 metric이 추가되었다. **(2점)**

각 benchmark를 세 번 측정한 후 중앙값을 이용한다. **(2점)**

2014-2학기 컴퓨터구조 기말시험 채점 기준표	
5. 10점	**Y86 ISA state :(총 4점 : 하나 틀릴 시 1점씩 감점, 최소 0점)** ① PC(Program Counter) ② program register ③ condition code ④ Memory ⑤ Program status **((a),(b),(c) 각 2점, 총 6점)** **(a)** pc ← pc + 2 R[%esp] ← R[%esp] -4 M[R[%esp]] ← R[rA] exception 발생 시 program status 변경 가능성이 있음 **(b)** pc ← pc + 6 R[rA] ← M[R[rB] + D] exception 발생 시 program status 변경 가능성이 있음 **(c)** PC ← M[R[%esp]] R[%esp] ← R[%esp] + 4 exception 발생 시 program status 변경 가능성이 있음
6. 10점	**Caller save** Caller는 callee를 호출하기 전에 live한 모든 registers를 memory에 save한다. 그리고 caller는 callee로부터 return된 뒤에 자신이 save했던 registers를 restore한다. **(3점)** **Callee save** Callee는 caller로부터 호출되면 callee 내에서 modify되는 모든 register를 memory에 save한다. 그리고 callee는 return전에 자신이 save했던 register를 restore한다. **(3점)** **Caller save/Callee save** Register들 중 permanent 변수를 담는 register의 경우 callee가 modify할 register들을 save하고, temporary 변수를 담는 register의 경우 caller가 live한 register들을 save해야 한다. Permanent 변수를 담는 register는 담겨있는 변수가 live할 확률이 높으므로 callee가 modify할 register만을 save하면 저장할 register 수를 줄일 수 있고 temporary 변수를 담고 있는 register는 담겨있는 변수가 live하지 않은 확률이 높으므로 caller가 live한 register만을 save하면 저장할 register 수를 줄일 수 있다. 이러한 방식을 통해 저장할 register의 수를 줄여서 효율을 높인다.**(4점)**

2014-2학기 컴퓨터구조 기말시험 채점 기준표	
7. 10점	**(각 2점)** **(1)** valB ← R[%esp] **(2)** M[valE] ← valP **(3)** valE ← 0 + valA **(4)** R[rB] ← valE **(5)** PC ← valP
8. 10점	**한 가지 경우에 대해 올바른 예시를 들었을 시 4점,** **두 가지 경우에 대해 올바른 예시를 들었을 시 7점,** **세 가지 경우에 대해 올바른 예시를 들었을 시 10점,** **답안 예시)** **Structural hazard:** - Instruction fetch와 memory access가 같은 cycle의 다른 stage에서 동시에 일어나는 경우 - Register read와 Register write 등의 작업이 동시에 여러 개가 일어나는 경우 **Data hazard:** - load instruction의 target register와 뒤따르는 instruction의 source register 사이에 dependancy가 존재하는 경우 - op instruction의 target register와 뒤따르는 instruction의 source register 사이에 dependancy가 존재하는 경우 **Control hazard:** - ret instruction의 경우 memory stage까지 가야만 다음 pc 주소를 알 수 있음 - conditional jump instruction의 경우 condition의 taken/not taken 여부를 확인해야만 다음 pc 주소를 알 수 있음
9. 10점	**한 가지 경우에 대해 올바른 예시를 들었을 시 4점,** **두 가지 경우에 대해 올바른 예시를 들었을 시 7점,** **세 가지 경우에 대해 올바른 예시를 들었을 시 10점,** **답안 예시)** Write through/Write back(Consistency와 performance 사이의 trade-off) TLB miss 시의 HW/SW 처리(Performance와 flexibility 사이의 trade-off) Cache의 크기(Performance와 cost 사이의 trade-off)