RV32A instructions

11개 명령어 존재

Syntax: [RV32A, RV64A Instructions — riscv-isa-pages documentation (msyksphinz-self.github.io)](https://msyksphinz-self.github.io/riscv-isadoc/html/rva.html)

위 링크 참고 바람

각 instruction의 기능 설명

-lr.w & sc.w: Multithead synchronization을 위한 기능. 기본적으로 동작은 load/store와 동일. Load 시 해당 메모리 주소에 대해 reservation을 설정한다(복잡한 것은 아니고, 단순히 표시를 해놓는 듯 하다). 만약 이 주소에 대해 sc.w로 쓰기를 수행하는데, 이 사이에 메모리가 update되지 않은 경우에만 쓰기를 수행한다. Sc.w의 return은 성공 시 0, 실패 시 nonzero임.

-amoswap: [rs1] ->rd, swap [rs1] and rs2

-amo+OP 명령어들: rd <- [rs1] OP rs2

\*개인 의견

->lr.w와 sc.w만 구현하면 나머지 instruction들은 반드시 구현할 필요는 없음. Load reserved와 store conditional pair는 CAS의 더 범용적인 호환이므로, 이 두 instruction을 통해 SW 적으로 semaphore의 구현이 가능함. 나머지 명령어들은 synchronization을 편하게 해주는 수단이지 필수는 아닌 것으로 보임. 다만, 두 명령어를 실제 어떻게 implementation할지는 문제가 됨. 이에 대해서는 실제 구현하면서 다른 구현 사례를 찾아봐야 할 것으로 보임. 지금 생각나는 방식은 해당 memory에 대해 additional bit(reservation bit)의 set, 해당 memory 주소 저장하는 buffer의 이용의 두 가지임. 전자의 경우 memory overhead가 매우 커진다는 점이 문제이고, 후자의 경우 reservation의 개수에 한계가 생긴다는 문제점이 있음.