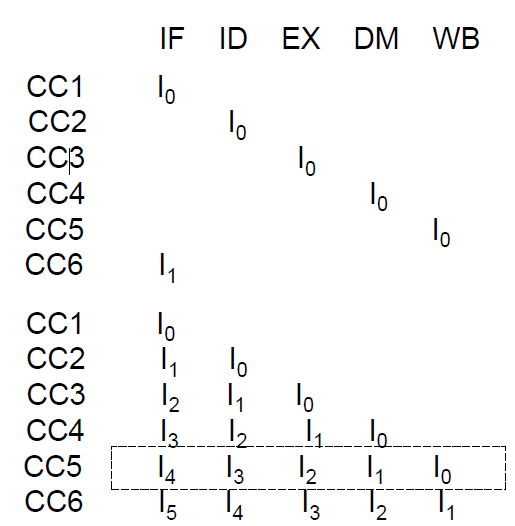
#HW12 2017069598 박상지

multicycle vs pipelining



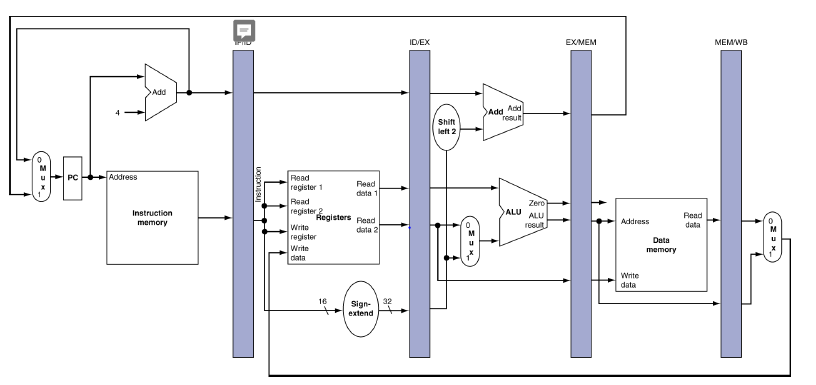
multicycle을 사용할 경우 clock cycle time이 200ps일 때, CPI는 5가 된다.

하지만 Pipeline 5-stage로 돌릴 경우 cct는 똑같지만, CPI는 1이 된다. 하나의 하드웨어로 서로 다른 instruction을 돌리기 때문에 성능이 높아진다. 모든 instruction이 이상적으로 똑 같은 시간이 걸린다면 이상적인 속도는 각 stage의 수와 같게 된다. 하지만 각 stage가 균일한 속도를 갖고 있지 않기 때문에 가장 느린 stage를 speedup 기준을 잡는다. pipeline은 매 cycle 마다 instruction을 fetch를 하는 것이 이상적이지만 hazard 때문에 매 cycle마다 instruction을 fetch할 수 없다.

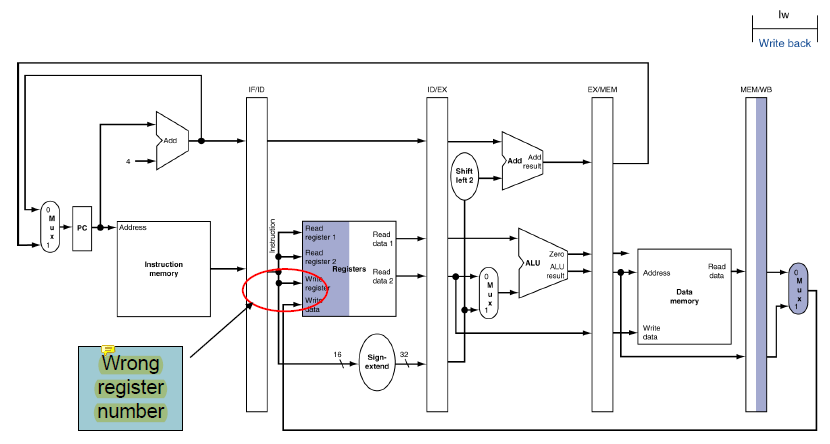
pipelined Datapath and control

파이프라인을 사용할 경우 각 stage마다 서로 다른 instruction간에 데이터와 control signal들이 간섭되면 안된다. 각 instruction은 clock이 올라 갈 때마다 fetch가 되기 때문에 각 stage마다 신호를 insulation을 해주는 역할이 필요하다.

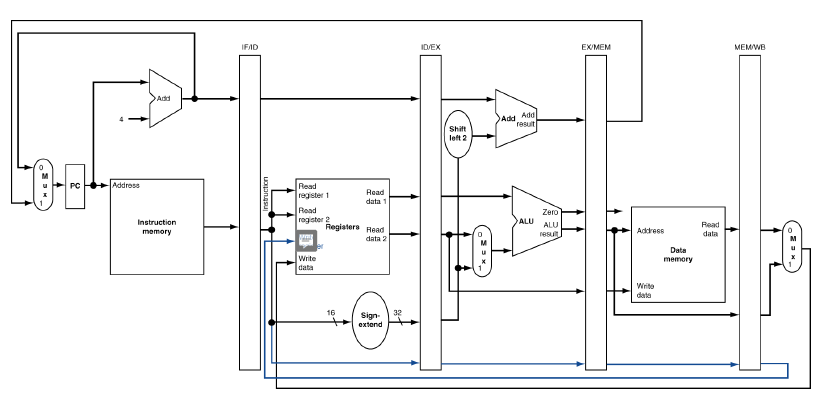
pipeline registers



stage간에 필요한 레지스터를 만들어 준다. instruction이 fetch가 일어나면 다음 clock이 올라오기 전에 해당 register에 결과 값을 저장한다.

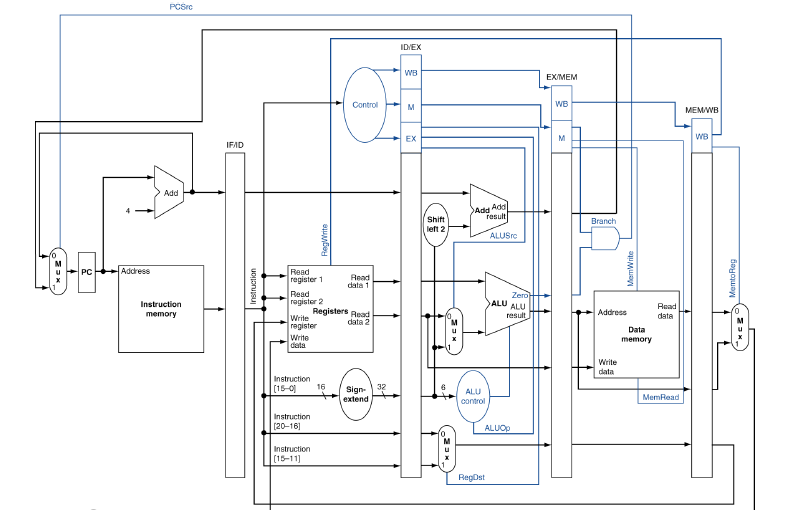


WB에서 쓸 데이터를 주는 것은 5번째 stage에서 주는 것인데 몇 번째 레지스터를 사용할 것인 결정하는 것은 2번째 stage에서 결정한다. 이 부분에서 버그(bug)가 일어난다. 몇 번 레지스터에 어느 데이터를 저장하는 것은 WB에서 결정해야 하는 일이다.



결과 값을 어느 레지스터에 저장할 것인지 write register을 결정하여 각 파이프라인에 계속 저장하면서 마지막 WB에서 사용된다.

Pipelined Control



IF와 ID는 모든 instruction의 공통 부분이다. decode가 끝나지 않았기 때문이다. 해당 stage에 PCSrc 와 RegWrite 신호들이 있다. 위치는 해당 stage에 있지만 신호를 주는 것은 WB(5stage), Branch(4stage)에서 준다. 즉 instruction이 무엇인지 모르기 때문에 control signal이 있을 수 없다.