Computer Organization

Lab#2: Pipelined CPU Implementation

2018-19775 이상엽

1. Pipeline register 추가

Ifid\_reg 모듈 내부에 PC, pc\_plus\_4, instruction 을 reg type으로 추가로 선언해주어 clk가 positive edge 일 때 input 값들을 reg에 넣어주고 아웃풋 wire에 assign했습니다. 또한 Flush signal과 stall signal을 인풋으로 받아 flush 와 stall 해야될 때의 아웃풋값이 어떻게 될지 조건문을 이용해 다루었습니다. Idex\_reg 모듈 내부에 clk, flush, stall signal을 마찬가지로 input으로 받게 했으며 flush해야하는 경우 control signal들에는 모두 0을, 넘겨야 하는 값들에도 모두 0을 넣었습니다. Stall 해야하는 경우 PC와 pc\_plus\_4 등 넘겨야 하는 값들에는 이전 사이클에 갖고 있던 값을 그대로 갖고 있도록 했으며 control signal에는 0을 넣었습니다. Exmem\_reg 모듈에는 skeleton에 flush signal만을 추가했으며 다른 pipeline register 모듈과 마찬가지로 skeleton에 있는 값들을 positive edge일 때 넘깁니다.

2. Forwarding module

Data forwarding 이 필요한 경우를 위해 forwarding module에 forward\_a, forward\_b signal을 output으로 내보내도록 했습니다. Ex stage에 있는 insturction의 rs1 또는 rs2값들이 mem stage 또는 wb stage에 있는 instruction의 rd 값과 같고 register file에 write한다면 각 케이스에 맞는 forward signal을 아웃풋으로 내보냅니다. Simple\_cpu에서 forward\_a, forward\_b wire를 선언해 forwardA mux와 forwardB mux의 select signal로 연결했습니다. Mux의 아웃풋값은 alu unit에 들어가며, forwardA mux의 아웃풋은 또한 jal/jalr을 구분하기 위해 jump\_out\_mux에 들어가 EA를 계산할 때 imm를 PC에 더할지 rs1에 더할지 결정되며 Branch control unit에 들어갑니다. fowardB mux의 아웃풋은 또한 s type의 명령어를 위해 ex/mem pipeline register에 들어갑니다.

3. Hazard module

Branch taken의 경우 target instruction 전까지의 instruction들을 flush할 수 있도록 hazard module에서 flush signal을 내보냅니다. 또한 load instruction이후 바로 register에 load한 값을 사용하는 instruction이 나올 경우 hazard module에서 stall signal을 보냅니다. Simple cpu에서 stall signal은 PC에 NEXT\_PC값을 넣기 전 조건문으로, ifid\_reg 모듈의 인풋과 idex\_reg 모듈의 인풋으로 들어가 stall을 수행하며 flush signal은 ifid\_reg, idex\_reg, exmem\_reg 모듈에 들어가 flush를 수행합니다.