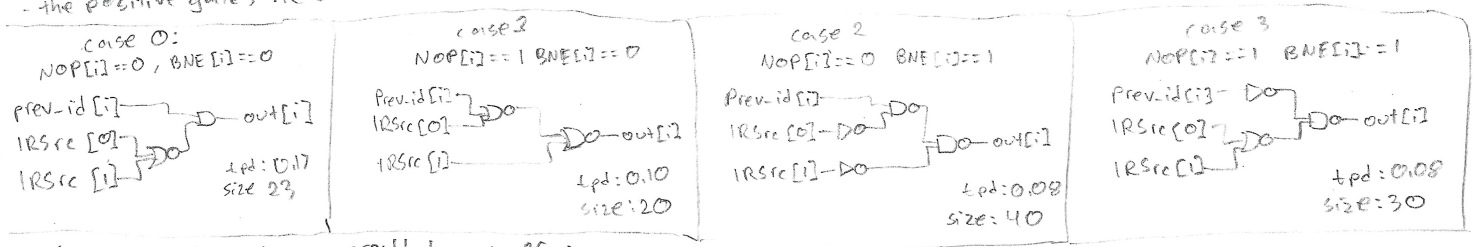
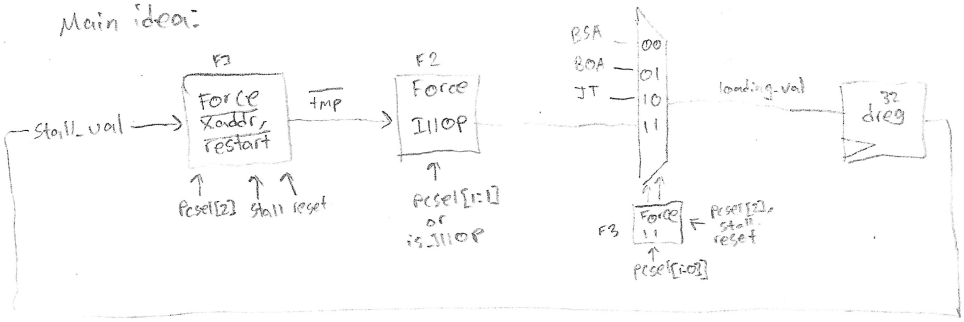
- ID_select size optimization
  - since we know in advance the constants to force under certain conditions, we can avoid the muxes
  - the positive gates here can be swapped for inverting + inverter if ever in critical path (it turns out to be)



* can save on size overall by buffering + reusing the inv + nor of IRSrc values

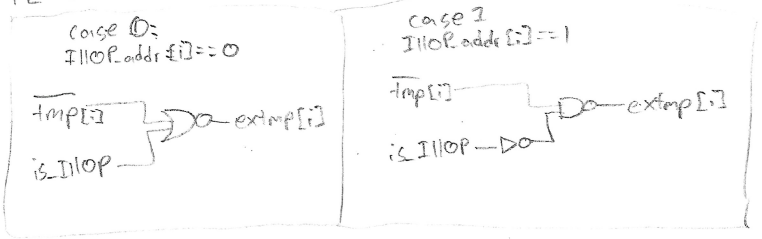
PC optimization (since stall, pcsel[2] generally are stable faster than pcsel[1:0], pcsel[1] is faster than pcsel[0], and when IllOP happens, both pcsel[1:0] are generated much faster.)

Main idea:

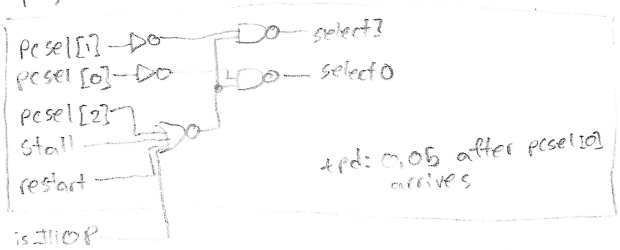


F3 produces the inverted form of either stall_val, Xaddr, or restart
(similar to ID_select, but with prev_id ≈ $\overline{\text{stall\_val}}$, NOP ≈ $\overline{\text{Xaddr}}$, BNE ≈ $\overline{\text{reset\_addr}}$, IRSrc[0] ≈ $\overline{\text{stall}}$ IRSrc[1] ≈ restart

F2 (can use is_IllOP + n_is_IllOP directly from RF stage)



F3



↖ add is_IllOP here and remove inverters, if using the dangerous inverted pcsel values that have not taken is_IllOP into consideration.

* I didn't end up using the dangerous implementations, since it could at best provide 0.02 - 0.05 ns savings per cycle, after enough tuning, but seems like a not very good design practice.