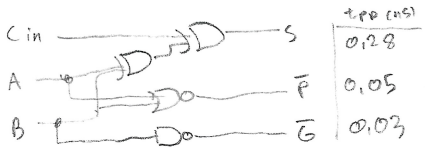
6.004 Design Project implementation notes continued

(carry-lookahead with alternating logic continued)

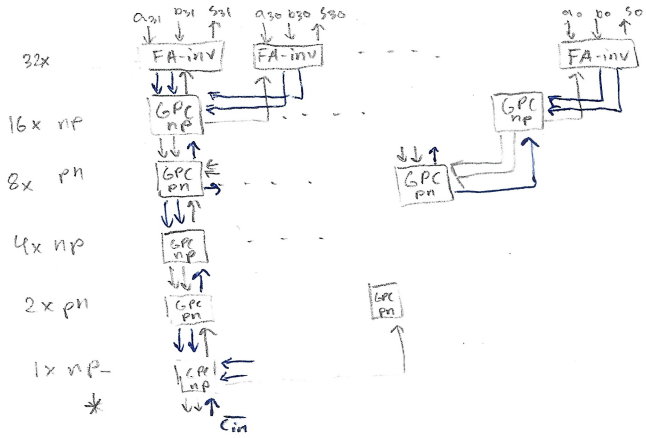
FA-inv



	t_{PD} (ns)
	0.28
	0.05
	0.03

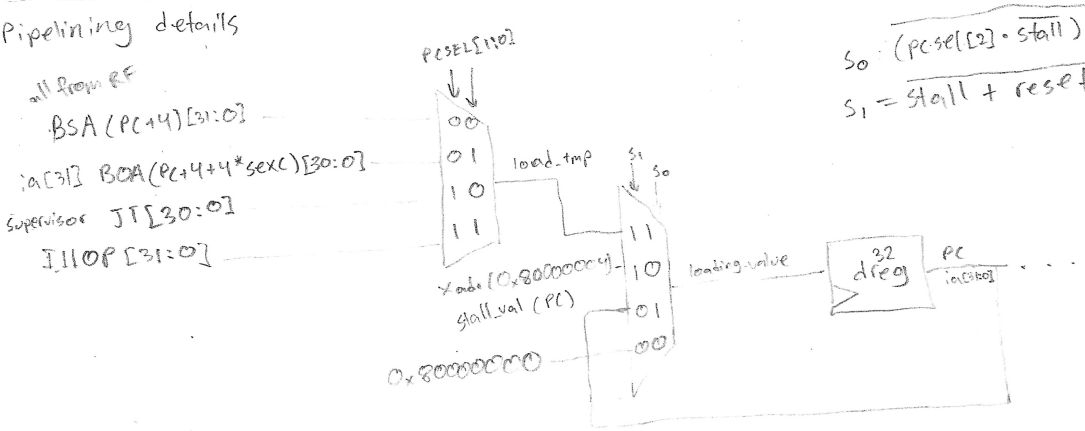
size: 74 λ^2

32 bit approach



* note: we can conserve a little time and size by inverting C_{in} within the 1x GPC np- so that we can pass the positive form immediately as C_L

Pipelining details

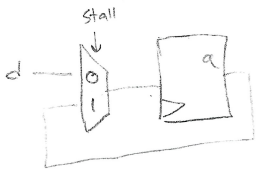


$S_0 = \overline{(pcsel[2] \cdot \overline{stall}) + reset}$

$S_1 = \overline{stall + reset}$

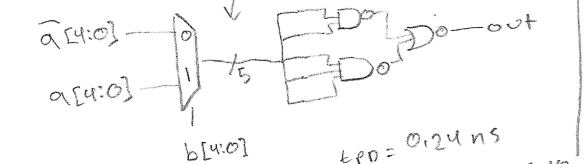
reset is given higher authority, then stall, then xaddr, and finally by default is load_tmp

Stall dreg



simple helper device for the state registers after the IF pipeline stage

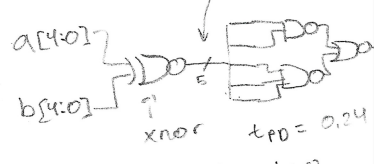
match 5



t_{PD} = 0.24 ns

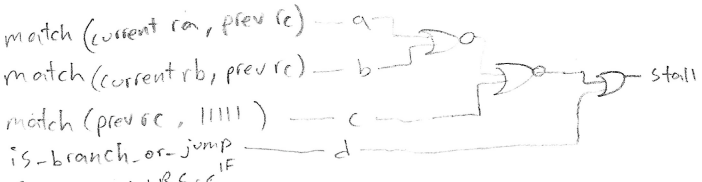
size = 1.97 + 0.93 + 5.10 = 218

better (slightly) match 5



size = 168

Two-stage pipeline control signals (using stall rather than bypass for simplicity, to handle Data & Control hazar



$d + (a+b) \cdot \overline{c} = \overline{(a+b) \cdot \overline{c}} + d = \overline{(a+b) + c} + d$

we want to stall if we need a register that is being written to in the second stage, as long as it is not r31. This requires exactly 1 stall for data hazar and exactly 1 stall for control hazard.

(ie branch_or_jump is temporarily computed by OR3 of pcsel bits)

NOP = ADD(r31, r31, r31) = 0b100000111111111111111100000000000

BNE(R31, 0, XP) = 0b 01110 11110 11111 0000000000000000 (did not work)

5-stage pipeline changes to ctl signal compute.

ra2sel only 0 for OP instruction (opcode == 10xxxx), so just $ra2sel = \overline{OP[5] \cdot OP[4]}$

moe computed alongside mwe and is just the opposite (could be computed faster if necessary)

wasel & wdsel still by ROM

- computing wdsel in ALU stage and piping it down, to be used to determine bypass values to consider
  - but wait to finalize it (taking IRQ into account) until WB stage

pcsel computed in RF stage to take Z into account