Smith, J. E. and A. R. Plezkun [1988]. "Implementing precise interrupts in pipelined processors", *IEEE Trans. on Computers* 37:5 (May), 562–73

Covers the difficulties in interrupting pipelined computers.

Thornton, J. E. [1970]. *Design of a Computer. The Control Data 6600,* Glenview, IL: Scott, Foresman.

A classic book describing a classic computer, considered the first supercomputer.

4.17 Exercises

4.1 Consider the following instruction:

Instruction: `and rd, rs1, rs2`

Interpretation: `Reg[rd] = Reg[rs1] AND Reg[rs2]`

4.1.1 [5] <§4.3>What are the values of control signals generated by the control in Figure 4.10 for this instruction?

4.1.2 [5] <§4.3>Which resources (blocks) perform a useful function for this instruction?

4.1.3 [10] <§4.3>Which resources (blocks) produce no output for this instruction? Which resources produce output that is not used?

4.2 [10] <§4.4>Explain each of the “don’t cares” in Figure 4.18.

4.3 Consider the following instruction mix:

R-type	I-type (non-ld)	Load	Store	Branch	Jump
24%	28%	25%	10%	11%	2%

4.3.1 [5] <§4.4>What fraction of all instructions use data memory?

4.3.2 [5] <§4.4>What fraction of all instructions use instruction memory?

4.3.3 [5] <§4.4>What fraction of all instructions use the sign extend?

4.3.4 [5] <§4.4>What is the sign extend doing during cycles in which its output is not needed?

4.4 When silicon chips are fabricated, defects in materials (e.g., silicon) and manufacturing errors can result in defective circuits.

A very common defect is for one signal wire to get “broken” and always register a logical 0. This is often called a “stuck-at-0” fault.

4.4.1 [5] <§4.4>Which instructions fail to operate correctly if the `MemToReg` wire is stuck at 0?

4.4.2 [5] <§4.4>Which instructions fail to operate correctly if the `ALUSrc` wire is stuck at 0?

4.5 In this exercise, we examine in detail how an instruction is executed in a single-cycle datapath. Problems in this exercise refer to a clock cycle in which the processor fetches the following instruction word: `0x00c6ba23`.

4.5.1 [10] <§4.4>What are the values of the ALU control unit’s inputs for this instruction?

4.5.2 [5] <§4.4>What is the new PC address after this instruction is executed? Highlight the path through which this value is determined.

4.5.3 [10] <§4.4> For each mux, show the values of its inputs and outputs during the execution of this instruction. List values that are register outputs at `Reg [xn]`.

4.5.4 [10] <§4.4> What are the input values for the ALU and the two add units?

4.5.5 [10] <§4.4> What are the values of all inputs for the registers unit?

4.6 Section 4.4 does not discuss I-type instructions like `addi` or `andi`.

4.6.1 [5] <§4.4> What additional logic blocks, if any, are needed to add I-type instructions to the CPU shown in Figure 4.21? Add any necessary logic blocks to Figure 4.21 and explain their purpose.

4.6.2 [10] <§4.4> List the values of the signals generated by the control unit for `addi`. Explain the reasoning for any “don’t care” control signals.

4.7 Problems in this exercise assume that the logic blocks used to implement a processor’s datapath have the following latencies:

I-Mem / D-Mem	Register File	Mux	ALU	Adder	Single gate	Register Read	Register Setup	Sign extend	Control
250 ps	150 ps	25 ps	200 ps	150 ps	5 ps	30 ps	20 ps	50 ps	50 ps

“Register read” is the time needed after the rising clock edge for the new register value to appear on the output. This value applies to the PC only. “Register setup” is the amount of time a register’s data input must be stable