University of Pennsylvania
Department of Electrical and System Engineering
Computer Organization

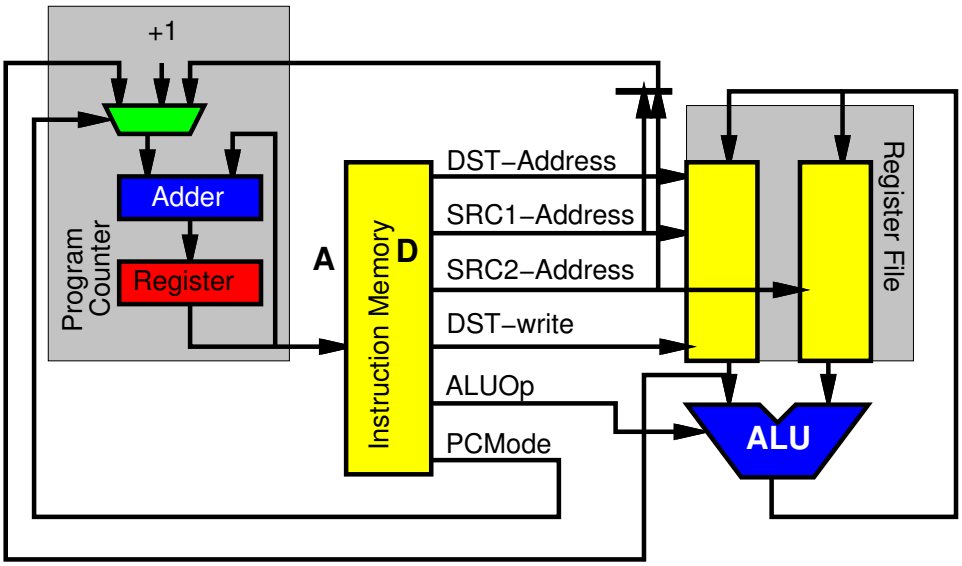
ESE534, Spring 2010 Assignment 3: Microcode Processor Monday, Feb. 1

Due: Monday, February 9, 12:00PM

We will build up an area model for a microcoded processor, write two simple programs, and estimate areas.

- Area(register) = 4
- Area(2-input gate) = 2
- Area of a d-entry, w-bit wide memory = $d(2\log_2(d) + w) + 10w$

This corrected 2/7 – original shown at end.



1. Determine the area of the ALU as a function of width, w.
   (a) Select the encodings for ALU operations in Table 1 (you will want to try to select them to simplify the bitslice and non-bitslice logic).
   (b) Write equations for the ALU bitslice to support the operations indicated in Table 1.
   (c) Add parentheses to the ALU bitslice equations so it is easy to count the number of 2-input gates required.
   (d) How many 2-input gates doe the bitslice require?

(e) Identify any non-bitslice logic you may need, write equations, and report the gate count for this logic.

(f) Write the equation for the area of a w-bit ALU.

2. Determine the area of the register file memory as a function of datapath width w and number of data items r. Assume the two-bank register file model shown above and in class.

   (a) Write the equation for the area of the w-wide, r-deep register file.

3. Determine the area of the instruction memory as a function of r and the number of instructions stored, i.

   (a) How many bits does the instruction memory need to supply?

   (b) Write the equation for the area of the instruction memory as a function of r and i.

4. Determine the area of the program counter as a function of the number of instructions in the instruction memory, i. Assume the program counter block includes the mux shown (also defined by Table 2). You don't have to worry about the Stop case; we'll assume other logic takes care of that.

   (a) How many bits does the program counter need to store?

   (b) Write equations for the bitslice of the program counter (including both the mux selection and addition). [original erroneously said "register file" instead of "program counter"]

   (c) How many 2-input gates are required for each program counter bitslice?

   (d) Write the equation for the area of the program counter supporting an instruction memory with i instructions.

5. Determine the area of the processor as a function of (i, r, w).

   (a) Composing your results above, write the equation for a processor with the parameters (i, r, w).

   (b) For the case where $(i = 16, r = 8, w = 16)$, make a table to show the area breakdown by functional units (ALU, register file, instruction memory, program counter). Show both the absolute area and percentage of total in the table.

   (c) Show a similar breakdown for the case $(i = 256, r = 32, w = 16)$.

6. Write microcode instructions for an 8b×8b multiply on a $w = 16$ processor without branching (PC-Mode=Increment) and estimate the area of the minimum processor to support.

   (a) Write the set of instructions required. Try to minimize the number of instructions and registers needed.

aluop	operation
ADD	out← input1+input2
INV	out← ∼(input1)
SUB	out← input1-input2
XOR	out← input1∧input2
OR	out← input1|input2
INCR	out← input1+1
AND	out← input1&input2
SRA	out← input1>>1; out[$w-1$]=input1[$w-1$]
SRL	out← input1>>1; out[$w-1$]=0
SLA	out← input1<<1
SLL	out← input1<<1

Table 1: Operations supported by the ALU

PC-Mode	encode	PC Behavior
Increment	00	PC←PC+1
Add Instr	01	PC←PC+(SRC1 concat SRC2) treat as signed number so can be negative.
Add Reg	10	PC←PC+register_file[SRC1]
Stop	11	PC←0 and stop executing

Table 2: Program Counter Modes (PC-Mode)

(b) How many cycles does this require to execute?

(c) How many registers do you need to use?

(d) Use your area model to estimate the area of the minimum processor that will support your code.

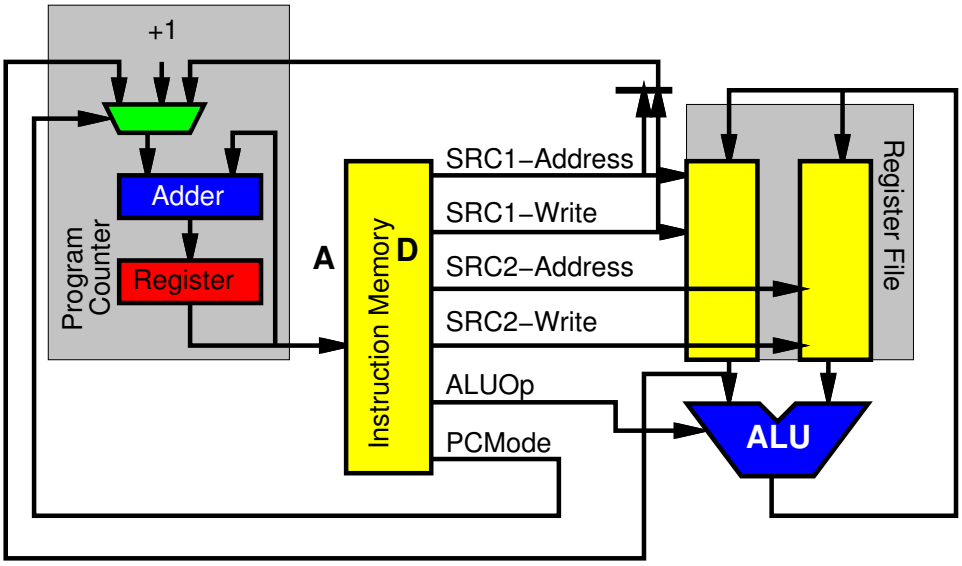
7. Write microcode instructions for an 8b×8b multiply on a $w = 16$ processor with branching (you may use all 3 PC-Modes) and estimate the area of the minimum ALU to support.

   (a) Write the set of instructions required. Try to minimize the number of instructions and registers needed.

   (b) How many cycles does this require to execute?

   (c) How many registers do you need to use?

   (d) Use your area model to estimate the area of the minimum processor that will support your code.



This doesn't show a separate destination address for the result of the operation. It also shows the concatenation of SRC1-address with SRC1-write to form the Add Instr offset rather than SRC1-address being concatenated with SRC2-address as intended and captured in Table 2 above.

Figure 1: Original (erroneous) datapath