University of Pennsylvania
Department of Electrical and System Engineering
Computer Organization

ESE534, Spring 2012

Assignment 5: Interconnect & Instructions

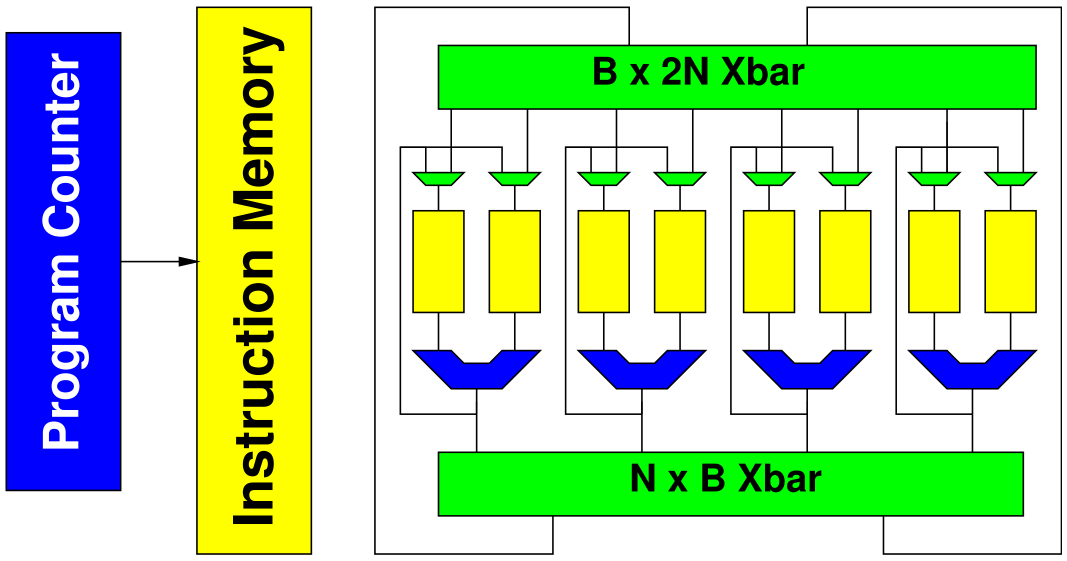
February 13

Problem 1 Due: Monday, February 20, 12:00PM
Problem 2 Due: Monday, February 27, 12:00PM

In this assignment we will explore a parametrized programmable model, examine energy implications, and explore the effects of mismatch between architecture and application.

You should now (after Day 9 lecture) be able to do Problem 1a–e. Problem 1f–g uses a term we will introduce on Day10. Problem 2 parallels ideas we'll see on Day 11.

This assignment asks you to sweep parameters and report costs and distributions of costs as a function of parameters. You definitely want to use a spreadsheet or computer program to generate the data and plot the points.



N is the number of physical ALUs. ALUs produce and consume w bits of data. Data memories local to each ALU store w-bit wide data. There are two data memories per ALU as shown. You can think of the Program Counter as being similar to the one from HW4. It should get inputs from the datapath and instruction memory, but we will assume we do not need to model that any further than the area/energy model for the program counter shown below.

Figure 1: Basic Architecture

Models: Use these models for the basic components. The general shape should look familiar from previous assignments. Numbers and model details here are selected to be relatively easy to work with and plausibly in the right ballpark; they do not attempt to rigorously capture a particular technology.

Component	**Area**	**Energy**
ALU	$100w$	$4w$
Memory Bank	$d(2\log_2(d)+w)+10w$	$d(\log_2(d)+w)+10w$
Crossbar	$N_{in}\times N_{out}\times(w+\log_2(N_{in}))$	$N_{in}\times N_{out}\times(w+\log_2(N_{in}))$
Single-Output Mux	$8N_{in}$	$N_{in}+\log_2(N_{in})$
Program Counter	$50b$	$8b$

Instruction control assumptions:

- Assume ALU op requires 4 bits to control.
- Assume no extra bits for PC control (we used 2 in HW4, but that 2 will generally be small compared to the total instruction width this time; let's assume we can ignore it here).
- Assume you need to supply **both** read and write addresses and write enables for **every** data memory bank this time.

Spreadsheet suggestions:

- Write your equations symbolically and plan out your spreadsheet before starting it.
- Compute constituents (like instruction memory width) in separate columns.
- Compute components (as need for 1e) {instruction memory, ALUs, data memory, interconnect (crossbars and muxes)} in separate columns and add together.
- Compute factors you need for 1c and 1e as different columns in the same spreadsheet.
- If using Excel, it will be useful to know the differences for: A1, A1, $A1, and A$1.

1. Construct an area and energy model for the architecture shown in Figure 1 and identify total energy and area and their distribution to primary components as a function of the number of active compute operators (N).

   We want our architecture to support a total number of operations, Ops_{total}. We make the simplifying assumption that we need one word of data for each of the Ops_{total} operations we support. Unlike the previous assignment, we will assume the memory banks are independently written, so the total memory stored in the $2N$ banks is Ops_{total}. We also assume we can pack the Ops_{total} operations perfectly into the N parallel operations the datapath can perform on each cycle. This means, the instruction memory depth will be $\left\lceil \frac{Ops_{total}}{N} \right\rceil$ so the number of ALU operations described by the instruction memory will remain a constant Ops_{total} as we scale N. The total instruction memory capacity may not remain constant because of the interconnect control required.

   You will need to identify the instructions and their size to determine the instruction width out of the instruction memory as a function of N and B.

   (a) Write an equation for the area of the architecture as a function of N, B, w, and Ops_{total}.

   (b) Write an equation for the energy per cycle of the architecture as a function of N, B, w, and Ops_{total}.

   (c) Assuming $w = 16$ and $Ops_{total} = 2^{14}$, plot the following as a function of N from 1 to 2^{14} (powers of 2 ok) for three cases: $B = 1$, $B = N$, $B = \sqrt{N}$

   - Energy per ALU operation=total-Energy-per-cycle / N
     (this is the energy assignable to an ALU operation including all of the instruction, interconnect, and data memory overheads, not just the energy of the ALU)
   - Area per ALU=total-area/N

   (d) Based on your answers to the previous part, what N is most energy efficient? area efficient? (possibly different answers for each of the three cases)

   (e) Assuming $w = 16$ and $Ops_{total} = 2^{14}$, plot the following as a function of N from 1 to 2^{14} (powers of 2 ok) for three cases: $B = 1$, $B = N$, $B = \sqrt{N}$

   - Percentage of energy going into each of {instruction memory, ALUs, data memory, interconnect (crossbars and muxes)}
   - Percentage of area going into each of {instruction memory, ALUs, data memory, interconnect (crossbars and muxes)}

   (f) Identify the *pinst* for this architecture (what does it contain)? What is its size in terms of w, Ops_{total}, B, N?

   (g) For $w = 16$ and $Ops_{total} = 2^{14}$, $N = 256$, and $B = \sqrt{N}$, compare the area of: (i) storage for each *pinst*, (ii) state data (data memory area per data word), (iii) one ALU, (iv) interconnect area per ALU (total-interconnect-area/N). State their absolute area and then give them as ratios to the smallest one.

2. Explore the cost in energy inefficiency of mismatches between the architecture and the application.

   This part mostly uses the same model as the previous one. However, our goal is to achieve Ops_{total} complete ALU operations. If a mismatch prevents operations from being performed, the number of instructions may need to be larger than Ops_{total} in order to support the desired Ops_{total} operations, and you will need to model the costs for this larger case.

   In the previous, case, we treated B as a parameter. Here, we want to explore what happens if B in the architecture (B_{arch}) is mismatched with the B required by the application (B_{app}) . The basic reasoning and comparison here is similar to the one developed in the *Theoretical Underpinnings* (Chapter 36) assigned for reading.

   (a) How is the architecture inefficient if $B_{app} < B_{arch}$? (This is looking for both a prose description of the source of the inefficiency and an equation to quantify the effect.)

   (b) How do we support applications with $B_{app} > B_{arch}$ on this architecture? What inefficiency results from this? (This is looking for both a prose description of the source of the inefficiency and an equation to quantify the effect.)

   (c) Assuming $N = 256$, $w = 16$, $Ops_{total} = 2^{14}$, what B_{arch} should we choose to remain energy competitive across a wide range of B_{app} values? Both give a value and explain why and how you selected it. Use model equations and/or calculations to support.

   (d) Plot the efficiency of your selected B_{arch} versus B_{app} from 1 to 256.