

CS 573
Final Exam
May 5, 2003

The following exam is open book and open notes. You may feel free to use whatever additional reference material you wish, but **no electronic aids** are allowed. Please note the following instructions. There will be a ten point deduction for failure to comply with them:

- start each problem on a new sheet of paper
- write your social security number, but not your name, on each sheet of paper you turn in
- show your work whenever appropriate. There can be no partial credit unless I see how answers were arrived
- be succinct. You may lose points for facts that, while true, are not relevant to the question at hand

You have until 12:30 to finish the exam. The questions are equally weighted.

1. Some computers (including VAX) use a single set of registers to hold both integer and floating point values. Most, however, use a set of integer registers and a separate set of floating point registers. How does this decision affect the ease of constructing the CPU - in which case is the register file(s) easier? In which case are the datapaths simpler?
2. The Intel Pentium 4 processor uses a “trace cache” to store decoded instructions, instead of a normal instruction cache storing them in their original non-decoded form. So far as I’ve been able to find, they have been quite secretive regarding exactly what goes in a decoded instruction in the trace cache (as well they should be!). One thing they *could* put there would be branch prediction information (so every decoded instruction in the trace cache would also have information regarding whether or not it is a branch, and a prediction as to whether it would be taken if it is). How this would compare to using a normal branch target buffer - would it cost more or less? would it be more or less accurate? would it be a good idea?
3. Calculate the frame check sequence for the following 12 bit message and 4 bit polynomial:

Message: 0x9b3
Polynomial: $x^3 + x^2 + 1$

4. Consider the following snippet of C code, to be executed on 2 processors. x is a shared variable, and the system uses serial consistency.

P1	P2
$x = 1;$	$x = 2;$
$x = x + 2;$	$x = x + 1;$

In answering the following questions, use the notation for memory accesses we’ve been using throughout the semester.

- (a) Show how this code can produce a sequence of memory reads and writes such that x can end up with a value of 2.
- (b) Show how it can produce a sequence of memory reads and writes such that x can end up with a value of 4.
- (c) Show how a sync point S can be inserted in the code which will guarantee that x will end up with a value of 3.

5. Some of the relevant characteristics of the Massively Parallel Processor are:

- Operations can be performed on all of the processing elements simultaneously, or can be “masked” by a special bit plane called the G plane. A masked operation is only performed by PEs whose element of the G plane has a value of 1 (of course, they are still performed simultaneously for all enabled PEs).
- It is possible to perform Boolean operations involving the G plane. In particular, it is possible to set elements of the G plane to 0 when an arbitrary condition is satisfied for some other plane.
- There is a one-bit output available called `sum_or`. When an operation is performed in the machine, this bit takes a value of 1 if the result of the operation is 1 on any PE in the machine.

So... it turns out MPP is able to find the largest element of an array in time linear in the number of bits in the values in the array. Describe, using either pseudocode or clearly in English, how this can be performed. *Hint: start with the most significant bit of the array, and see if any of the PEs have a 1 in that bit. Remember that once a PE has been established as not containing the greatest value, it doesn't ever need to be considered again.*

6. Draw a 9×9 Benes network using 3×3 crossbars switches. Since I've put a figure showing the switches on this page, that might be a good starting point.

