

CS573
Final Exam
May 8, 2002

The following exam is open book and open notes. You may feel free to use whatever additional reference material you wish, but **no calculators** 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. I will take off 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. Assume the three numbers below are eight bit integers protected with a SECDED scheme as presented in class. Assume the order of the bits is

$M_8M_7M_6M_5C_8M_4M_3M_2C_4M_1C_2C_1P$

(this is the order given in the definition in the notes). For each of the numbers, tell whether (1) there is no error, (2) there is a one-bit error (and tell which bit is wrong), or (3) there is a two-bit error.

- (a) 0011001011100
- (b) 0110010100100
- (c) 1110110001001

2. Here's a problem J Strother Moore described during his colloquium Monday (I'm changing the presentation a bit, but it's the same problem):

Suppose you have two processes, each running the following code

```
global int x;
local int t1, t2, t3;
t1 = x;
t2 = x;
t3 = t1 + t2;
x = t3;
```

(assume the initial value of x is 1)

- (a) Using the notation for global memory interactions that we've used in class, show how it is possible to have interleavings of reads and writes to x that will result in x having a final value of 2, 3, or 4, assuming strict consistency.

Just to be clear, the notation I mean is the one that looks something like

```
P1:  R(x)1
-----
P2:
```

- (b) Insert synchronization primitives of your choice to guarantee that the result will be 2.

3. Here's a statement about instruction sets I found on Usenet a while ago:

```
Subject:  Re:  Is RISC dead?  (was:  Re:  K7's FP
performance inches past P-III's)
Date:    1999/05/19
Author:  Piercarlo Grandi <piercarl@Dial.PIPEX.com>
```

Well, my take here (and I said so in "comp.arch" quite a while ago) is that `_architecture_` is dead: with essentially infinite silicon/design budgets virtually any architecture, including x86, can be made to perform; in other words only implementation matters.

Have we reached the point that the instruction set is irrelevant, because with current technology any pig can be made to fly? Are some aspects of the architecture still important, and others less so? Justify your answer.

4. Suppose you are selecting a network topology for transmitting 128 byte messages. You can have either a wormhole-routed 2D square grid with a one-byte flit, or a store-and-forward binary hypercube. Links in the two topologies have identical bandwidths.

How many nodes would you need to have before the hypercube was faster than the grid (determined by worst-case time to send a 128 byte message, assuming no blocking)?

5. Dataflow and scoreboarding (out-of-order execution as used in the Cray 1 or the Pentium Pro) are alternative ways to extract the maximum parallelism from programs. Compare the two approaches. Is one more likely than the other to be able to extract maximum parallelism?
6. Back in the Golden Age of Supercomputers, one of the distinguishing characteristics of a supercomputer was the capability to perform vector operations. Today, current processors seem to have almost completely

abandoned that technology; even things like Intel's MMX or SSE2 extensions to IA32 can only operate on trivially short vectors. Given that there is still a need to perform the sorts of problems that vector instructions were aimed at, why has this happened?