

# 18-742 Advanced Computer Architecture

## Test 2

**April 14, 1998**

Name (please print): \_\_\_\_\_

**Instructions:**

***DO NOT OPEN TEST UNTIL TOLD TO START  
YOU HAVE UNTIL 12:20 PM TO COMPLETE THIS TEST***

The exam is composed of four problems containing a total of 12 sub-problems, adding up to 100 points overall. The point value is indicated for each sub-problem. Show all work in the space provided. If you have to make an assumption then state what it was. Answers unaccompanied by supporting work will not receive full credit. The exam is closed book, closed notes, and “closed neighbors.” You are on your honor to have erased any course-relevant material from your calculator prior to the start of the test. Please print your initials at the top of each page in case the pages of your test get accidentally separated. You may separate the pages of the test if you like, and we will re-staple them when you hand it in.

Good luck!

## Grading Sheet

1a) (8) \_\_\_\_\_

1b) (8) \_\_\_\_\_

1c) (6) \_\_\_\_\_

2a) (15) \_\_\_\_\_

2b) (8) \_\_\_\_\_

2c) (8) \_\_\_\_\_

2d) (8) \_\_\_\_\_

3a) (8) \_\_\_\_\_

3b) (8) \_\_\_\_\_

3c) (8) \_\_\_\_\_

4a) (8) \_\_\_\_\_

4b) (7) \_\_\_\_\_

**TOTAL:** (100) \_\_\_\_\_

**Problem 1 (Vector Computing)**

You have a chaining vector computer with the following bandwidths available for 8-byte data values:

- 17 memory banks at 88 Million B/sec each
- One bus at 440 Million B/sec
- A VDS with three concurrent connections at 440 Million B/sec each
- A pipelined vector multiplier at 55 MFLOPS; 3 clock latency; 1 result per clock throughput
- A VRF with three bidirectional ports at 440 Million B/sec each

The following table gives the latency for vector multiplications at different vector sizes:

<b>Vector Length</b>	<b>Total Latency (clocks)</b>	<b>Achieved MFLOPS</b>
<b>1</b>	13	
<b>2</b>	16	
<b>3</b>	19	
<b>4</b>	22	
<b>5</b>	25	
<b>6</b>	28	

(8 points)

a) Fill in the Achieved MFLOPS column for the above table. (Assume no other operations are concurrent with the addition.) Write the computation for vector length 5 below:

(8 points)

b) For vector multiplication, what is  $R_{\infty}$  on this architecture? Show computations for potential bottlenecks.

(6 points)

c) What is  $N_{1/2}$  on this architecture? Use a linear interpolation with the bracketing pair of data points in the table, and report to two decimal places. Or, if you want to use a purely equation-based approach that's an acceptable alternative.

**Problem 2 – Disk Drives (& Interleaved Memory (& some Software Tuning))**

You've just landed a job at NASA and are working with high-resolution mapping data. You have map data at 1 meter resolution, and want to study the Mississippi River basin in the U.S. To do this you're going to perform computations on the rows and columns of a 2-D matrix of size 1M elements x 1M elements x 8 bytes per element (*i.e.*, a 1,048,576 element-square matrix, covering about a 650 x 650 mile area.) You decide that because this is bigger than physical memory, you're going to have to rely upon virtual memory to store the matrix on an array of disks.

The biggest disk drive you can get through the government procurement system has the following characteristics:

- 16 platters
- 2 sides per platter; one head per side
- 1024 tracks per side
- 2048 sectors per track
- 512 byte sectors
  
- Assume a very simplistic timing model for the disk to move a head for each time it changes tracks:  
2 msec + 10 usec/track

You decide that it's worth your while to rewrite the part of the OS that maps data to the swap drives. You assign memory pages to the disk drives so that each consecutive 8KB virtual memory page is placed in the same spot on a consecutively numbered disk drive, with the starting sector number increased when addressing wraps back around to the first drive. Within each 8KB chunk, the data is placed on consecutive sectors. This is the "usual" way of interleaving with no skew factor, except that an "interleave module" is a disk drive, and each module is accessed in chunks of 8 KB virtual memory pages. Furthermore, you place data so as to absolutely minimize head movement when accessing the array via sequential memory addresses (in this case accessing rows is done sequentially).

(15 points)

- a) Write the equations for addressing the disk drives to implement interleaved memory in this fashion. Assume you are given an 8K-page-aligned byte address (“addr”) and want to compute the disk location of the first sector holding that memory page. Also assume that there are “drives” number of disk drives available, and that they are numerous enough to fit the entire array or more. In your equations assume integer division that truncates any fractional result (so, for example,  $16 / 3 = 5$ ), and that all addresses are zero-based (so the first drive is drive#0, first sector is sector#0, *etc.*). Put all answers in terms of “addr”, “drives”, and numbers – use no other variables.

HINT: don’t forget the “mod” operation in the equations.

Drive# =

Sector# =

Track# =

Side# =

Platter# =

(8 points)

b) What is the minimum number of disk drives necessary for swap space for this array?

(8 points)

c) Given that you have already accessed all the data in the first row, how many total rows of data can be accessed without further moving any disk heads (including the first row)? Ignore virtual memory system page table/directory information – just consider that actual data from the matrix is on disk. Assume that there are 1024 disks in the array, which is not necessarily the minimum number required.

(8 points)

d) Consider that the matrix is interleaved across a 1024 disk array. Ignoring everything except disk head movement time, how long will it take to read a column of data from disk? Assume maximal concurrency when moving disk heads.



**Problem 3 – System Bus**

You are designing a computer system bus with the following assumptions. You've been asked to do a design tradeoff involving reducing the bus cost.

- Clock speed is 66 MHz (everything in this problem is in “bus clocks”).
- Each motherboard socket costs \$.03 in component cost for each gold-plated *contact* (“pin”). In addition there is a flat assembly labor cost of \$.50 for each entire socket. There are 8 such connectors on the motherboard.
- Each card requires a \$2.50 bus interface chip set, and an additional \$.01 per contact for the gold-plated edge connector “fingers”. For this problem you are designing for an entry-level system configuration which is populated with 4 such boards.
- Your initial design uses a 128-signal bus, and includes 64 signals for data and an additional 32 signals for address.
- A bus transaction transfers 256 bits, and takes 7 clocks to complete (*i.e.*, 7 clocks per 256-bit burst transfer of a cache block).
- In the baseline system, the bus can use fully pipelined, split transactions.

(8 points)

a) What is the bus cost for the described configuration (total socket costs + total board costs)?

(8 points)

b) What is the bus cost savings (in dollars) if the address and data lines are multiplexed and no other signals change?

(8 points)

c) How much bus bandwidth (in MB/sec) is lost by multiplexing the address and data lines? The transfer latency is designed such that it stays the same, but split transactions are not supported with a multiplexed design.

**Problem 4 – Error Coding**

A Hamming SEC code includes 11 data bits and 4 check bits according to the below table:

	$d_{11}$	$d_{10}$	$d_9$	$d_8$	$d_7$	$d_6$	$d_5$	$c_4$	$d_4$	$d_3$	$d_2$	$c_3$	$d_1$	$c_2$	$c_1$
$S_1$	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0
$S_2$	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0
$S_3$	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0
$S_4$	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

(8 points)

a) What are the syndrome equations?

$$S_1 =$$

$$S_2 =$$

$$S_3 =$$

$$S_4 =$$

(7 points)

b) Consider a computer with a 78-signal bus. Of the bus signals, 64 are for multiplexed address/data. You decide to add a SEC/DED error correcting code to the address/data lines to improve system dependability. How many total bits are on the bus after this addition, assuming no additional control signals need be added. (HINT: even though there is a very minor equation you can use to answer this, it is also possible to work it out using “common sense” about how these codes really work in terms of finding erroneous bits.)