# *UNIVERSITY OF VICTORIA*

## *EXAMINATIONS FALL 2008*

## *C SC 230 A01/A02 – Introduction to Computer Architecture*

**NAME:** ______________________________ **STUDENT NO.:** ____________________

**INSTRUCTOR:** R. N. Horspool **DURATION:** 3 hours

**TO BE ANSWERED ON THE PAPER**

| Question No. | Value | Mark | Question No. | Value | Mark |
|---|---|---|---|---|---|
| 1 | *12* | | 7 | *6* | |
| 2 | *15* | | 8 | *6* | |
| 3 | *10* | | 9 | *16* | |
| 4 | *13* | | 10 | *12* | |
| 5 | *10* | | 11 | *8* | |
| 6 | *12* | | | | |
| **TOTAL MARKS** | | | | **120** | |

## INSTRUCTIONS:

1. STUDENTS MUST COUNT THE NUMBER OF PAGES IN THIS EXAMINATION PAPER BEFORE BEGINNING TO WRITE, AND REPORT ANY DISCREPANCY IMMEDIATELY TO THE INVIGILATOR
2. THIS EXAM HAS 12 PAGES, AND A ONE PAGE / 2-SIDED SEPARATE HANDOUT.
3. No aids are permitted. However, a 2-sided ARM instruction set reference page is provided for your use.
4. The marks assigned to each question are shown in square brackets. Partial marks are available for all questions.
5. Please be precise but brief, and use point form where appropriate.
6. It is strongly recommended that you read the entire exam through from beginning to end before beginning to answer the questions.

**Question 1. [12]** Fill in your answers in the right-hand column of the table below:

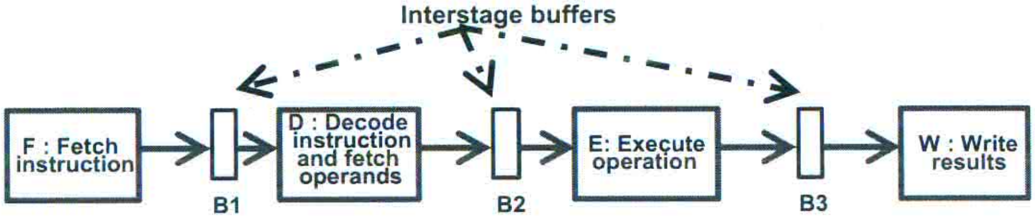
| | | |
|---|---|---|
| **a)** | What is the *binary* representation of the decimal number 431 ? | |
| **b)** | What is the *hexadecimal* representation of the decimal number 431 ? | |
| **c)** | What is the 8-bit two's complement *binary* representation of the decimal number -3 ? | |
| **d)** | What is the 32-bit two's complement *hexadecimal* representation of the decimal number -3 ? | |
| **e)** | What is the *decimal* equivalent of the hexadecimal number 0x1AA ? | |
| **f)** | A machine has 109 opcodes; how many bits are needed for the opcode field of an instruction on that machine ? | |
| **g)** | What is $2^{17} \times 2^{15}$ ? (Give your answer as a power of two.) | |
| **h)** | An array A of 4-byte integers begins in memory at address 0x13A4; what is the address of the element A[36]? (Note: 36 is a decimal number, and the first array element is A[0].) | |
| **i)** | If R1 = 0x3E04 and R2 = 0x10E, what memory location is accessed by the ARM instruction LDR R0, [R1,R2] ? | |
| **j)** | Simplify the following expression, giving your answer in hexadecimal: (0x0A3D2 >> 2) & 0xFFF | |
| **k)** | The bytes at locations 0x100 and 0x101 hold 0x01 and 0x02. What is the decimal value of the number held in the halfword at address 0x100 if big-endian ordering is used ? | |
| **l)** | The bytes at locations 0x100 and 0x101 hold 0x01 and 0x02. What is the decimal value of the number held in the halfword at address 0x100 if little-endian ordering is used ? | |

**Question 2.** [15] Find the word or phrase from the list on the left that best matches the descriptions in the table on the right. Use the numbers to the left of words in the answer, and write them in the right-hand column of the second table. Each answer should be used only once. *Fill in the column labelled "#" below:*

| | |
|---|---|
| 1 | condition code |
| 2 | word |
| 3 | cache |
| 4 | flash memory |
| 5 | compiler |
| 6 | MMU (memory management unit) |
| 7 | signed-magnitude |
| 8 | an interrupt |
| 9 | decoder |
| 10 | opcode |
| 11 | twos-complement |
| 12 | pipeline |
| 13 | DRAM (dynamic random access memory) |
| 14 | virtual memory |
| 15 | cycle stealing |

| | Description | # |
|---|---|---|
| A | Memory which retains its contents without power for an indefinite period. | |
| B | Bits which remember whether an instruction has generated a negative, zero or positive result. | |
| C | A group of bits with the size used as data operands by most instructions. | |
| D | Component of the CPU where instructions are executed as a series of stages. | |
| E | A device which translates virtual addresses to real addresses. | |
| F | A logic circuit which generates a 1 value on one of its output lines as selected by the binary number provided as its input. | |
| G | A representation scheme for numbers which provides representations for one more negative number than for positive numbers. | |
| H | A combination of RAM memory and disk memory which the program can access as though it were all RAM memory | |
| I | An asynchronous transfer of control to a system software routine. | |
| J | Memory whose contents must be repeatedly refreshed. | |
| K | A representation scheme for numbers which has representations for both plus zero and minus zero. | |
| L | Means by which DMA (direct memory access) obtains its access to main memory | |
| M | Program that translates from a higher-level notation to assembly language. | |
| N | A command to the CPU to perform an action. | |
| O | A small, fast memory that acts as a buffer for the main memory. | |

**Question 3.** **[10]** Suppose that you need to execute 100 instructions and that each instruction requires 4 clock cycles. In total, the 100 instructions would take 100 × 4 = 400 clock cycles. Now suppose that you are given a pipeline with 4 stages to perform the instructions, as drawn below. How many clock cycles will it take if there are no hazards? Show the answer as a number and also show how you calculated it.



**Question 4.** •**[13]** Assume you have a system which contains various elements which have been discussed in the course, namely Virtual Memory, Cache, Main Memory, Pipelining, MMU (Memory Management Unit), TLB (Translation Lookaside Buffer), Page Table, DMA (Direct Memory Access). Answer the following questions:

**a)** **[1]** Where is the Page Table stored ?

**b)** **[1]** Where is the TLB stored ?

c) [3] Describe the steps which take place when there is a cache hit (the best case scenario).

d) [3] Describe the steps which take place when the needed data is still on disk (the worst case scenario).

e) [3] Describe the steps which take place when the needed data is in main memory, but not in the cache.

f) [2] You may or may not have mentioned the DMA in the steps above. If you included the DMA, give a definition of what it is and what function it had in the previous 3 scenarios. If you did not include the DMA, state why not and still give a definition of what it is.

**Question 5. [10]** The contents of a 2-way set associative cache are as shown below. Addresses are 16 bits, memory is byte addressable, words are 4 bytes, and each line in the cache is 2 words in size. Answer the questions written alongside and below .

| Set# | Valid | Tag | Contents |
|---|---|---|---|
| 0 | 1 | `0x309` | – *something* – |
|  | 1 | `0x04E` | – *something* – |
| 1 | 1 | `0x28E` | – *something* – |
|  | 1 | `0x1F7` | – *something* – |
| 2 | 0 | `0x283` | – *something* – |
|  | 1 | `0x20A` | – *something* – |
| 3 | 1 | `0x049` | – *something* – |
|  | 0 | `0x3AE` | – *something* – |
| 4 | 0 | `0x31F` | – *something* – |
|  | 0 | `0x1E1` | – *something* – |
| 5 | 1 | `0x3AE` | – *something* – |
|  | 1 | `0x0BD` | – *something* – |
| 6 | 1 | `0x1F4` | – *something* – |
|  | 1 | `0x262` | – *something* – |
| 7 | 1 | `0x03D` | – *something* – |
|  | 0 | `0x1F2` | – *something* – |

**a)** **[1]** An address can be decomposed into a tag, a set number and an offset. How many bits does the offset occupy?

**b)** **[1]** How many bits does the set number occupy?

**c)** **[1]** How many bits does the tag occupy?

**d)** **[1]** If the CPU accesses address `0xEBAA`, which set in the cache is checked?

**e)** **[1]** Is the access to `0xEBAA` a hit or a miss?

**f)** **[1]** Is an access to `0x8294` a hit or a miss?

**g)** **[1]** Is an access to `0x7CBA` a hit or a miss?

**h)** [3] Suppose that the cache hit rate is 98%, and that a memory access takes 1 clock cycle for a hit and 10 clock cycles for a miss. What is the average access time to memory when the cache is used?

**Question 6. [12]** A 1024 × 1024 array of 32-bit numbers is to be "normalized" as follows. For each column, the largest element is found and then all elements of the column are divided by this maximum value. Assume that each page in the virtual memory consists of 4K bytes, and that 1M bytes of the main memory are allocated for storing data during this computation.

An array can be stored in memory in row-major order or column-major order. Row-major order for this array implies that row 0 of the array would be held in the 1st page of memory, row 1 in the 2nd page, and so on. Column-major order implies that column 0 would be held in the 1st page of memory, column 1 in the 2nd page of memory, and so on.

a) **[4]** How many pages are read in from disk and how many changed (dirty) pages are written back to disk if column-major order is used? Assume that all dirty pages must be written back to disk when the array normalization is finished.

b) **[4]** How many pages are read in from disk and how many changed (dirty) pages are written back to disk if row-major order is used? Again, assume that all dirty pages must be written back to disk afterwards.

c) **[4]** Suppose that it takes 40ms to transfer a page in either direction between main memory and disk. Estimate the total time needed to perform this normalization for both row-major order and column-major order.

**Question 7.** **[6]** Assume that the processor has a pipeline with 4 stages, namely for “Fetching instructions”, for “Decoding instructions and fetching operands”, for “Executing operations” and for “Writing results” with 3 inter-stage buffers. The two instructions below are found in that order in a segment of code to be executed with this processor and it is possible that they may cause a delay in the normal performance.

```
MUL R1,R2,R3
STR R4,[R5,R1]
```

a)[2] Why might they cause a delay?

b)[1] What is this situation called?

c) [2] Draw a timing diagram for a general pipeline of this type which shows this situation

d) [1] What solutions are possible either in software or in hardware to avoid this kind of delay? State one.

**Question 8. [6]** Assume that dynamic random access memory (DRAM) must be refreshed every 64ms. Suppose that we have a DRAM chip which has 8192 rows of 8 bytes each, and that each row can be refreshed in 4 clock cycles. The clock speed is 1GHz.

**a)** [1] How many clock cycles are needed to refresh the entire DRAM chip?

**b)** [1] How many clock cycles is 64ms equivalent to?

**c)** [2] What fraction of time is spent refreshing the DRAM chip? (Give you answer as a simple expression – do not attempt to evaluate it.)

**d)** [2] Where in the computer is DRAM normally used (in preference to SRAM) and why?

**Question 9. [16]** You are to write ARM code for a function which counts how many bits are set in a word. Some C code for this function is shown below.

```
int CountBits( int word ) {
     int result = 0;
     while( word != 0 ) {
          if (word < 0)       /* test the leftmost bit (the sign bit) */
               result++;      /* increment if the bit was one */
          word = word << 1;   /* shift left one place */
     }
     return result;
}
```

Write this function in ARM assembler. You do not need to write the code for the main calling routine. Assume that the argument named *word* is passed in register R1. The result is to be returned in register R0. Your function is not allowed to have any side-effects. Make sure you include appropriate comments to receive full marks.

```
CountBits:          @ your ARM code continues below
```

**Question 10. [12]** A very short mystery ARM assembler program is shown below. Label A in the program is at address `0x1048`. Read the code and then answer the questions written alongside and below.

```
1               .text
2    _start:    ldr      r0, =A
3               mov      r1, #6
4               bl       MMM
5               swi      0x11
6
7    MMM:       stmfd    sp!, {r1-r4,lr}
8               mov      r1, r1, LSL #2
9               add      r2, r0, r1
10              bal      MM2
11   MM1:       ldr      r3, [r0]
12              ldr      r4, [r2]
13              str      r4, [r0]
14              str      r3, [r2]
15              add      r0, r0 #4
16   MM2:       sub      r2, r2,#4
17              cmp      r0, r2
18              blt      MM1
19              ldmfd    sp!, {r1-r4,pc}
20
21              .data
22   A:         .word    10, 20, 30, 40
23              .word    50, 60, 70, 80
24              .end
```

**a) [1]** What value does r1 contain at line 10?

**b)** [1] What value does r2 contain at line 10?

**c) [1]** How many times does the `blt` instruction at line 18 jump to label MM1?

**d) [4]** What values are held in array A when the program halts? (List all 8 values, giving them in decimal.)

**e) [1]** In words, explain what the MMM function does.

**f) [4]** The MMM function can be made at least 2 instructions shorter. Give the line numbers of two instructions which can be removed and explain what other instructions have to be changed to allow their removal.

**Question 11. [8]**

**a)** [2] What is the speedup that can be expected if the instruction set of a certain computer is changed so that the branch instruction takes one clock cycle instead of three clock cycles, if branch instructions account for 20% of all instructions executed by a certain program. Assume that other instructions average three clock cycles per instruction and that nothing else is altered by the change.

**b)** [3] On most computers, a conditional branch instruction requires a different number of clock cycles depending on whether the branch is taken or is not taken. Why is that?

**c)** [3] Describe one possible approach that the designer of the computer's architecture can use to reduce the number of cycles needed by a conditional branch instruction in the worse of the two cases of part (b).

**END**