Name: ____________________________ ID _VOO________

CSC 230 Assignment 4

Spring 2017

This assignment has two parts:

1) **Written Part:** The written problems are to be handed in on **Monday, March 27 in class**. Please write your answers on the assignment document.

2) **Programming Part:** Submit only your C code file on conneX. Be sure to include your name and student number. **Submit it on conneX well before 11:50 pm Saturday, April 1, 2017.**

Assignment 4: Part 1 - Written

Its just too difficult to type this into Connex! Thus, this hard copy assignment, please write your answers on this document. No electronic submissions.

Two things associated with this assignment that are intended to ease the end of term:

→ The number of marks is in excess of 50, but you only need to do enough questions such that you achieve a minimum of 40 marks total.

→ While the due date for the written part is actually March 27, it will be accepted until 2:30 pm on April 3, the last day of class. During that class, we will grade the assignment together.

Question 1. [4 marks] When running a particular program with N memory accesses, a system with a cache and paged virtual memory generates a total of M cache misses and F page faults. T1 is the time for a cache hit, T2 the time for a main memory hit, T3 the time to load a page into main memory from disk. Answer the following questions and justify all your answers to get full marks.

(a) [2] What is the cache hit ratio?

(b) [2] What is the main memory hit ratio? That is, what proportion of memory accesses do not generate a page fault?

Name: ______________________________ ID _VOO_________

Question 2. [5 marks] A processor accesses main memory with an average access time of *T2*. A smaller cache memory is interposed between the processor and main memory. The cache has a significantly faster access time of *T1* < *T2*. The cache holds, at any time, copies of some main memory words and is designed so that the words more likely to be accessed in the near future are in the cache. Assume that the probability that the next word accessed by the processor is in the cache is *H* (known as the Hit ratio).

(a) [1] For any single memory access, what is the theoretical speedup of accessing the word in the cache rather than in main memory?

(b) [2] Let *T* be the average access time. Express *T* as a function of *T1*, *T2* and *H*.

(c) [2] In practice, a system may be designed so that the processor must first access the cache to determine if the word is in the cache and, if it is not, then access main memory, so that on a miss (opposite of a hit), memory access time is *T1*+*T2*. Express *T* as a function of *T1*, *T2* and *H*, with the new assumption.

Question 3. [2 marks] Figure 1 shows the possible organization of a 4-stage pipeline.

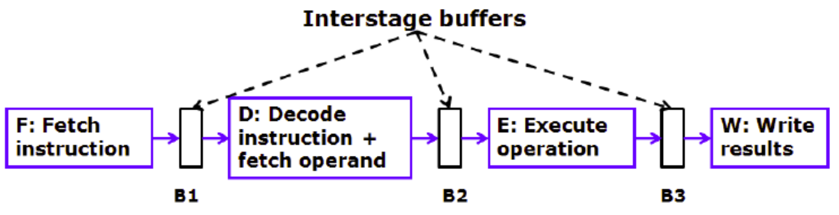


FIGURE 1. A 4-stage Pipeline

(a) [1] Redraw the diagram showing how the organization could be enhanced by including an instruction queue.

(b) [1] Redraw the diagram showing how the organization should *normally* be enhanced in a super scalar design.

Question 4. [3 marks] 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 (=1,000 MHz, in cycles per seconds).

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

(b) **[1]** Given the number of clock cycles from above, how much time, in ms, are they equivalent to? (An expression, rather than a calculated result, is acceptable.)

[1] Let your answer from above be N ms. What fraction of time is spent refreshing the DRAM chip? (Give you answer as a simple expression using N – Do not evaluate it.)

Question 5. [**3 marks]** Suppose that execution time for a program is directly proportional *only* to instruction access time. Access time to an instruction is *2 ns* from the cache and *20 ns* from memory. The probability of a cache hit is 96%. In the case of a cache miss, the instruction is fetched from main memory and copied into the cache, and then a second access must take place to copy the instruction from the cache (this time it will be a hit).

(a) [1] Compute the execution time of a program with 100 instructions without the cache (an expression is fine).

____

(b) [1] Compute the execution time of a program with 100 instructions with the cache (an expression is fine).

____

(c) [1] If the cache size is doubled, the probability of not finding an instruction is cut in half. Compute the execution time of a program with 100 instructions with the larger cache (an expression is fine).

____

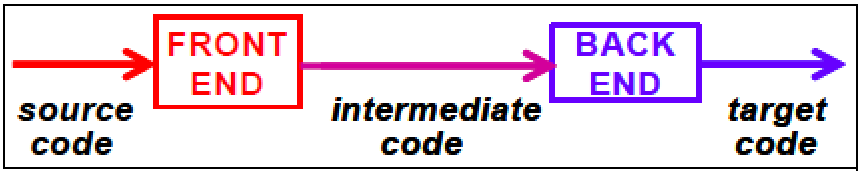
Question 6. [6 marks] You have a system with Virtual memory, a DMA, a Page Table, a TLB, RAM memory, disks, and two levels of cache L1 and L2, with L1 being further subdivided into an L1 Data cache and an L1 Instruction cache.

(a) [3] The CPU wants to access some data and the data is in L2, but not in L1. Describe the series of events precisely and concisely.

(b) [3] Repeat, assuming the data is in RAM memory and not in the L1 or L2 caches.

Question 7. [4 marks]

(a) [3] The figure below shows the structure of a compiler decomposed into a "Front End" and a "Back End" with *Intermediate Code* generated in between.

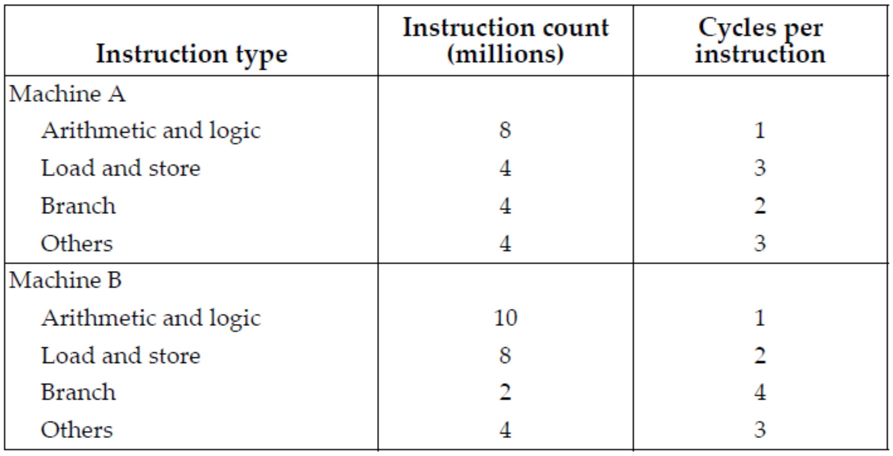


Explain briefly, in point form, what the phases are in the front end, that is, when the high level source code is translated into intermediate code. In an interview situation, you should not take more than 2 minutes; be similarly concise and precise.

(b)

(c) [1] In the explanation above, somewhere, the notion of a *Symbol Table* should have been introduced. Describe in more depth what the function of the Symbol Table is within the process.

Question 8. [6 marks] Consider two different machines, with two different instruction sets, and with the same clock rate of 200MHz. The following measurements are recorded on the two machines running a given set of benchmark programs:



Instruction type	Instruction count (millions)	Cycles per instruction
Machine A		
Arithmetic and logic	8	1
Load and store	4	3
Branch	4	2
Others	4	3
Machine B		
Arithmetic and logic	10	1
Load and store	8	2
Branch	2	4
Others	4	3

(a) [2] Determine the effective CPI for each machine (show your calculations).

[2] Compute the execution time in ns of the benchmarks for each machine (show work).

(b) [2] A common measure of performance for a processor is the rate at which instructions are executed, expressed as millions of instructions per second (MIPS), referred to as the MIPS rate. We can express the MIPS rate in terms of the clock rate and CPI as follows:

$$\text{MIPS rate} = \frac{\text{Clock rate}}{\text{CPI} \times 10^6}$$

State the MIPS rate for each machine above (ie, show the complete equation for the calculation, final computation not required).

Question 9. [8 marks] Here is an example of how the cache works:

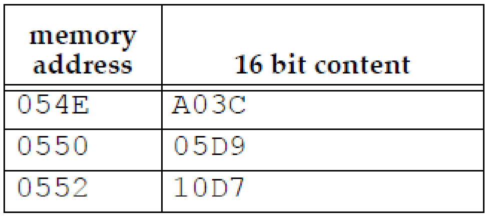
A computer uses a small direct-mapped cache between the main memory and the processor. The cache has four 16-bit words, and each word has an associated 13-bit tag, as shown in the figure on the right, where, however, the tag has been ignored for this task. When a miss occurs during a read operation, the requested word is read from the main memory and sent to the processor. At the same time, it is copied into the cache, and its block number is stored in the associated tag.

	16 bit content
address 0	
address 2	
address 4	
address 6	

Consider the following loop in a program where all instructions and operands are 16 bits long and the code starts at address 0x02EC:

```
LOOP:ADD (R1)+,R0 02EC
DECR R2 02EE
BNE LOOP 02F0
```

Assume that, before this loop is entered, registers R0, R1 and R2 contain: R0 = 0, R1 = 0x054E, R2 = 3. Also assume that main memory contains the data as shown on the right



memory address	16 bit content
054E	A03C
0550	05D9
0552	10D7

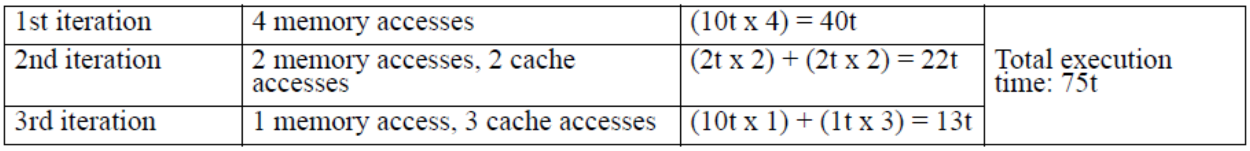
Because this cache is direct mapped, it easy to decide, beforehand, the location in cache to which each word is mapped. For the 3 instructions, the calculation was as follows:

```
instruction                   address of instruction
LOOP: ADD (R1)+,R0            02EC =0000 0010 1110 1100 mapped to cache 4
      DECR R2                 02EE =0000 0010 1110 1110 mapped to cache 6
      BNE LOOP                02F0 =0000 0010 1111 0000 mapped to cache 0
```

For the 3 memory locations, the calculation was as follows:

```
content    address in memory
  A03C           054E = 0000 0101 0100 1110 mapped to cache 6
  05D9           0550 = 0000 0101 0101 0000 mapped to cache 0
  10D7           0552 = 0000 0101 0101 0010 mapped to cache 2
```

Looking at the execution, after this first iteration there have been 4 memory accesses and no cache hits. The second iteration is similar, but some memory accesses are avoided, and in total there are 2 memory accesses and 2 cache accesses. At the end, the table below shows the totals for memory and cache accesses. The example also assumes that the access time to main memory is *10t* and that of cache is *1t*. The execution time for each pass, ignoring the time taken by the processor between memory cycles, is calculated in the right column.



1st iteration	4 memory accesses	(10t x 4) = 40t	Total execution time: 75t
2nd iteration	2 memory accesses, 2 cache accesses	(2t x 2) + (2t x 2) = 22t	
3rd iteration	1 memory access, 3 cache accesses	(10t x 1) + (1t x 3) = 13t	

(a) [6] Repeat the process above using separate instruction and data caches, each of the same size as the previous one. There is no need to show every step with explanation as above, only:

A diagram of the cache after each iteration is completed.
A summary of the memory and cache accesses after each iteration is completed.
A final table showing the total speed of execution.
A final comment on whether separate caches appear to speed up execution.

Provide answers by filling in the tables below.

After 1st iteration				After 2nd iteration				After 3rd iteration			
INSTR. CACHE		**DATA CACHE**		**INSTR. CACHE**		**DATA CACHE**		**INSTR. CACHE**		**DATA CACHE**	
cache line	**16 bit content**	**cache line**	**16 bit content**	**cache line**	**16 bit content**	**cache line**	**16 bit content**	**cache line**	**16 bit content**	**cache line**	**16 bit content**
0		0		0		0		0		0	
2		2		2		2		2		2	
4		4		4		4		4		4	
6		6		6		6		6		6	
Memory accesses =				**Memory accesses =**				**Memory accesses =**			

Iterations	**Total accesses**	**Total time per iteration**	
1st iteration			Total execution time:
2nd iteration			
3rd iteration			

(b) [2] A second possibility is to assume that the cache is used only to store instructions. Data operands are fetched directly from the main memory and not copied into the cache. *It appears that this choice yields faster execution.* Verify this claim and give reasons why this choice lead to faster execution than when both instructions and data are written into the *same* cache?

Question 10: [10 marks]

As discussed in the class for memory hierarchy to work correctly, the main memory is mapped on to the cache memory. Some of the mapping techniques are direct, fully associative and set associative. The key design parameters are: sizes of main and cache memory, lines (blocks) in the cache, blocks in the main memory and mapping function. For this question let us assume that the size of the main memory is 16Mbytes, the size of the cache is 16Kbytes and the size of the block is 32 bytes and the CPU is operating at 16MHz.

a) [1] How many address bits (lines) are needed for the main memory?

b) [1] How many blocks of main memory are there? How many bits are needed to identify these blocks?

c) [1] How many cache lines (slots, blocks) are there and how many bits are needed to identify these?

d) [1] How many tag bits are needed if the memory uses direct-mapped cache?

e) [1] How many tag bits are needed if the memory uses fully-associative cache?

f) [1] How many tag bits are needed if the memory uses a two-way (set size of 2) set-associative cache?

g) [1] How many tag bits are needed if the memory uses a 4-way set-associative cache with a set size of 4?

h) [1] Let us assume that the cache hit rate is 90%, cache access time is 1 cycle and miss penalty is 16 cycles. What is the average memory access time of the CPU?

i) [1] What is the speed-up achieved with the above parameters?

j) [1] What is the speed-up if cache hits ratio changes to 50%?

Assignment 4: Part 2 - Programming

Objective

- Gain confidence with "C" programming for the AVR.

In the last lab you will use the C programming language to write code for the AVR 2560. You can reuse the code from the lab in this assignment. Your program should be written in C and perform the steps described in Part I of Assignment 3.