2019-01-04

# Exam in DAT 105 (DIT 051) Computer Architecture

**Time:** January 19, 2019 14 – 18

**Person in charge of the exam:** Per Stenström, Phone: 0730-346 340

**Supporting material/tools:** Chalmers approved calculator.

**Exam Review:** On January 31, 2019 10-12 in Room 4128

**Grading intervals:**

- **Fail**: Result < 24
- **Grade 3**: 24 <= Result < 36
- **Grade 4:** 36 <= Result < 48
- **Grade 5:** 48 <= Result

**NOTE 1:** Bonus points from Real-stuff studies and Quizzes will be added to the exam results for approved exams used solely for higher grades.

**NOTE 2:** Answers must be given in English

**GOOD LUCK!**
***Per Stenström***

**[General disclaimer: If you feel that sufficient facts are not provided to solve a problem, either 1) ask the teacher when he visits the exam, or 2) make your own additional assumptions. Additional assumptions will be accepted if they are reasonable and required to solve the problem. Always make sure to motivate your answers.]**

## ASSIGNMENT 1

The tables below show the CPI assuming a single-cycle memory system ($CPI_0$) and number of Misses-Per-Kilo-Instructions (MPKI) on two machines (A and B) and a reference machine (R) with the **same** Instruction Set Architecture (ISA) for two single-threaded programs, P1 and P2, respectively, where P1 executes twice as many instructions as P2 which executes 1 million instructions. The operating frequencies of the three machines (A, B and R) are also shown. The miss penalty is 100 nanoseconds for all three machines.

| **Program P1** | **A** | **B** | **R** |
|---|---|---|---|
| $CPI_0$ | 0.5 | 1.0 | 1 |
| MPKI | 10 | 10 | 1 |
| | | | |
| **Program P2** | **A** | **B** | **R** |
| $CPI_0$ | 2.0 | 1.5 | 1 |
| MPKI | 10 | 10 | 1 |

| **Clock freq. (GHz)** | |
|---|---|
| Machine A | **1** |
| Machine B | **1.2** |
| Machine R | **1** |

**1A)** Calculate the execution times for P1 and P2 on A, B and R (**4 points)**

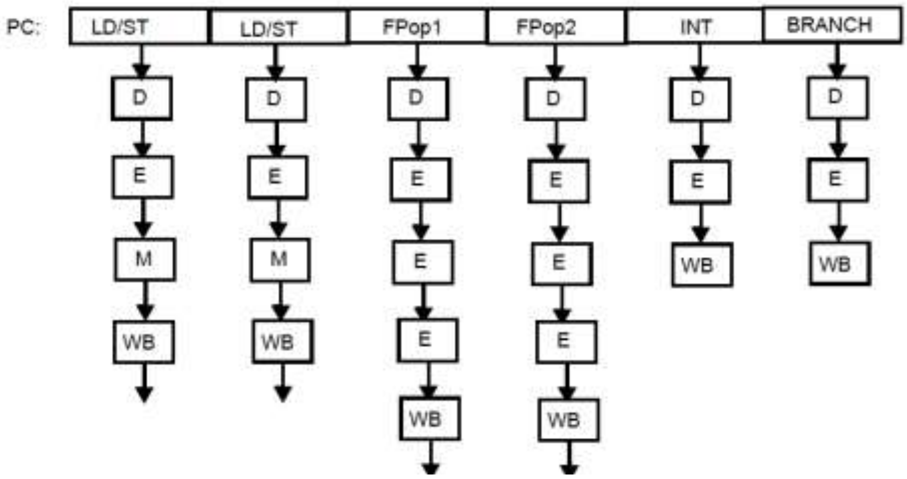
**1B)** Determine which of the machines is the fastest using **geometric means** (**4 points)**

**1C)** Assume that 10% of the execution time of a program is spent on executing the division instruction. What is the speedup obtained on *the entire program* if an ingenious design of a division functional unit would result in a speedup of division instructions by a factor 1000? (**2 points)**

**1D)** Why is arithmetic means of the execution time unreliable as a metric to compare performance of two machines? (**2 points)**

## ASSIGNMENT 2

We consider in this assignment a VLIW architecture that can issue two memory, two floating-point, one integer, and one branch instruction each cycle according to the pipeline organization below. There are no forwarding units.



**2A)** Consider the following code:

```
LOOP: LD F1,0(R1)
      ADD F4, F0,F2
      SD F4,0(R1)
      SUBI R1,R1,#8
      BNE R1, R2, LOOP
```

Show how the instructions in one iteration of the loop above should be scheduled on the VLIW pipeline to maximize instruction level parallelism. (**2 points)**

**2B)** Consider again the code in Assignment 2A. Use loop unrolling seven times to statically schedule the code to maximize instruction level parallelism. (**6 points)**

**2C)** Consider the following code:

```
for (i = 0; i < N; i++)
   {A[i+2] = A[i] + 2;
    B[i] = A[i+2] + 1;}
```

This is translates into

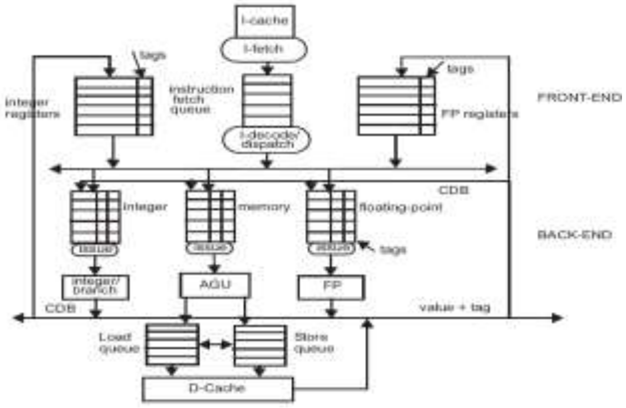
```
Loop:     L.S F0, 0(R2)          ; –O1
          ADD.S F3, F0, F1       ; –O2
          ADD.S F4, F3, F1       ; –O3
          S.S. F3, 16(R2)        ; –O4
          S.S. F4, 0(R3)         ; –O5
```

```
ADDI R2,R2, 8
ADDI R3,R3, 8
BNE R2,R4, Loop
```

Show the code for the kernel in a software pipelined version of the code that maximizes instruction level parallelism. **(4 points)**

## ASSIGNMENT 3

The diagram below shows a pipeline with support for Tomasulo's algorithm. There are two functional units for adding floating-point numbers and a single functional unit for floating-point division. It takes 2 cycles to carry out an addition/subtraction and 5 cycles to carry out a division.



**3A)** Explain *in detail* what happens in each of the three pipeline stages: Issue, Execute, and Write result. In particular explain how data hazards are resolved and in which cycle each instruction in the sequence below enters the different stages by filling out a pipeline diagram similar to the one below for the following instruction sequence. (**6 points)**

```
ADDD   F1, F2, F3          ; –O1
DIVD   F4, F1, F2          ; –O2
SUBD   F2, F4, F6          ; –O3
ADDD   F4, F1, F1          ; –O4
```

| | Cycle 1 | Cycle 2 | Cycle 3 | Cycle 4 | Cycle 5 | Cycle 6 |
|---|---|---|---|---|---|---|
| O1 | Issue | | | | | |
| O2 | | Issue | | | | |
| O3 | | | Issue | | | |
| O4 | | | | Issue | | |

**3B)** Consider the code below as it runs on a pipeline like the one above with support for speculative execution. Assume that the branch instruction is predicted to NOT be taken and that the prediction is validated to be correct when the last instruction has been speculatively executed. Explain in detail how the speculative processor keeps track of the register values during speculative execution and after how many cycles after the branch instruction has been validated, all register values are available in the register file. (**4 points)**

```
BNEZ    R1, LABEL
ADDD   F1, F2, F3
DIVD    F4, F1, F2
SUBD   F2, F5, F6
```

**3C)** Explain how a two-level branch predictor works. (**4 points)**

## ASSIGNMENT 4

**4A)** A computer architect wants to establish the relative performance between a system with a blocking and a non-blocking cache using software prefetching. In software prefetching, prefetch instructions are appropriately inserted in the code shown below. Assume that five instructions are executed per iteration where the CPI for each instruction is one assuming that it doesn't cause a cache miss. On the other hand, if the instruction causes a cache miss, CPI is 100. In addition, prefetch instructions result in CPI=2. Both caches have a block size of four words.

```
for (i=0; i<1000; i++)
     C+=A[i];
```

- Show how the code is annotated with prefetch instructions to hide all cache misses.
- How many MSHRs are needed to make software prefetching maximally effective?
- How much faster does the program run on the system with a non-blocking cache using software prefetching?

A convincing explanation that is easy to follow is needed for full points. (**6 points)**

**4B)**

In the table below, we show MPKI (Misses-Per-Kilo-Instruction) for a number of organizations. The fraction of memory instructions, out of all executed instructions, is 10%. From the data, determine the cold miss rate, the capacity miss rate and the conflict miss rate for a direct-mapped cache. (**3 points)**

| **Cache organization** | **MPKI** |
|---|---|
| 16-KB direct mapped cache | 10 |
| 16-KB 2-way assoc. cache | 8 |

| | |
|---|---|
| 16-KB fully associative cache | 6 |
| Cache with infinite size | 2 |

**4C)** How does sequential prefetching work? **(3 points)**

## ASSIGNMENT 5

**5A)** Consider a multicore system comprising a number of processors (cores) on a chip that are connected to a single-level private cache. The private caches use the *write-through* write policy. $X_i$=$R_i$ and $X_i$=$W_{i,}$ mean a read and a write request to the *same* address from processor *i,* respectively, where $W_i$=*C* means that the value C is written by processor *i*. Now consider the following access sequence:

$R_1$<br>
$R_2$<br>
$W_{1=}0$<br>
$W_{2=}1$<br>
$R_1$<br>
$R_2$

What is returned by the second read operation from processor 1 and what is the reason that the correct value is not returned given the cache write policy assumed? How can we modify the cache controller to make sure that the right value is returned?
(**6 points)**

**5B)**
Assume a write-back cache and the MSI-protocol. What bus transaction will cause a state transition from state I to state S and what transaction will cause a transition from state S to state I? **(2 points)**

**5C)** Explain the concept of interleaved (fine-grain) multithreading. Consider a five-stage pipeline. What additional mechanisms and pipeline stages must be added to support interleaved multithreading? How many cycles are lost on a thread switch? **(4 points)**

****** GOOD LUCK! ******