CHALMERS

2023-10-11

Exam in DAT 105 (DIT 051) Computer Architecture

Time: 23-10-23 at 14-18 Johanneberg

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

Supporting material/tools: Chalmers approved calculator

Exam Review: More information on this will be available via Canvas

Grading intervals:

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

NOTE 1: Only a Chalmers approved calculator is allowed

NOTE 2: Bonus points from Real-stuff studies and Quizzes will be added to the exam results for approved exams used solely for higher grades.

NOTE 3: Answers must be given in English

GOOD LUCK!
Per Stenström

CHALMERS UNIVERSITY OF TECHNOLOGY
DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
412 96 Göteborg
Visiting address: Rännvägen 5
Phone: 031-772 1761 Fax: 031-772 3663
Org. Nr: 556479-5598
E-mail: pers@chalmers.se

[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

You are supposed to characterize the performance of two computers, A and B, with data regarding three programs P1, P2 and P3.

The following data has been collected.

Assumptions for computer model A	
Execution time for programs P1, P2 and P3:	2s, 10s and 2s, respectively
Number of instruction cache accesses:	10^8, 10^9 and 2 x 10^8 for P1, P2 and P3, resp.
Clocks per instruction not taking into account misses from instruction/data caches	8 for P1-P3
Miss penalty	100 ns
Speedup over reference machine R	10, 10, 10 for P1, P2 and P3, resp.
A's operating frequency	1 GHz
Assumptions for computer model B	
Number of instruction cache accesses:	10^8, 10^9 and 10^8 for P1, P2 and P3, resp.
Clocks per instruction not taking into account misses from instruction/data caches	1 for P1-P3
Miss penalty	100 ns
Speedup over machine A	0.5, 5, 0.5 for P1, P2 and P3, resp.
B's operating frequency	1 GHz

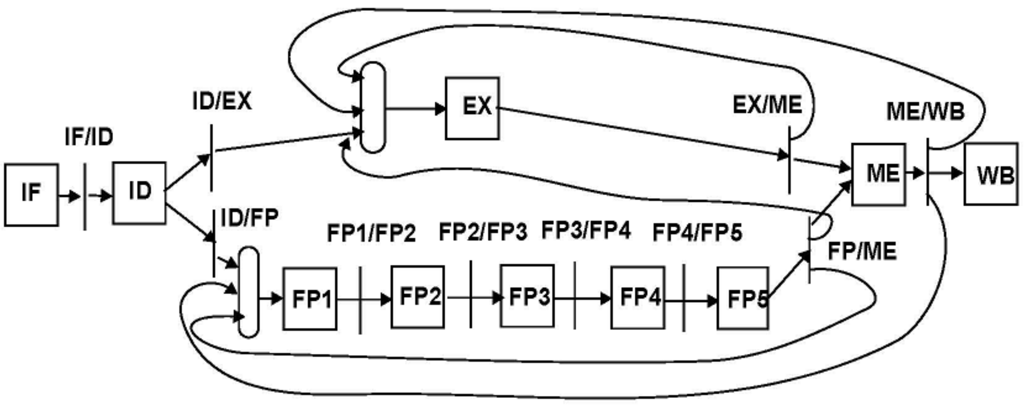
1A) What is the arithmetic mean of execution times for P1-P3 of A and B? Which one is the fastest and by how much? Please comment on the reason for why one is faster than the other **(4 points)**

1B) What is the geometric mean speedup over the reference machine of A and B? Which one is the fastest and by how much? If the result is not consistent with the result in 1A, explain why? **(4 points)**

1C) Establish the number of Misses Per Kilo Instructions (MPKI) for P1 and P2 on A and B. **(4 points)**

ASSIGNMENT 2

We consider in this assignment a pipeline with a 5-stage pipelined floating-point unit and a single-stage execution unit that executes integer, load/store and branch instructions. There are forwarding units from the output of each execution unit and from the memory stage.



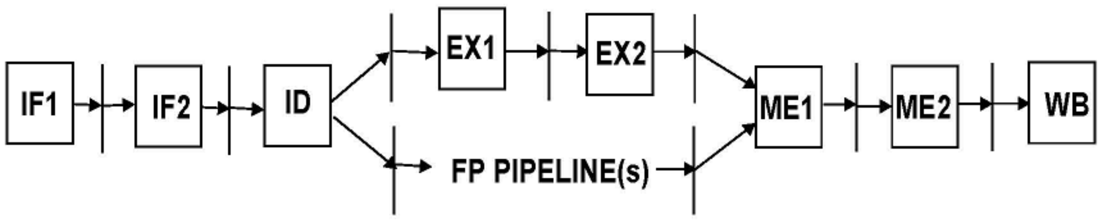
2A) Consider the following instruction sequence:

I1: LD F1,0(R1)
I2: LD F3,0(R1)
I3: ADD F0,F1,F2
I4: ADD F4,F3,F0
I5: ADDI R2,R2,#1
I6: SD F4, 0(R1)

Determine the number of cycles it takes from I1 is issued from the ID-stage until I6 enters the ME-stage in the case that the floating-point unit is fully pipelined. (**4 points)**

2B) The model used in 2A) is now changed to cope with higher clock frequencies as follows.

- The IF stage is replaced by two IF stages: IF1 and IF2
- The integer execution unit is replaced by two **fully pipelined** stages: EX1 and EX2
- The FP pipeline is fully pipelined with initiation rate 1
- The ME stage is replaced by two **fully pipelined** stages



Consider the same program as in 2A) and determine the number of cycles it takes from I1 is issued from the ID-stage until I6 enters the ME1-stage. Assuming that we can double the frequency of this pipeline, which of the pipelines is the fastest? **(4 points)**

2C) Consider the following loop and convert it into a software pipelined loop for a VLIW architecture showing the prolog and kernel code. The VLIW architecture has two memory execution-units, a floating-point unit and an integer unit where the latency is 1, 4 and 0 cycles for the memory, floating point and integer units, respectively. **(4 points)**

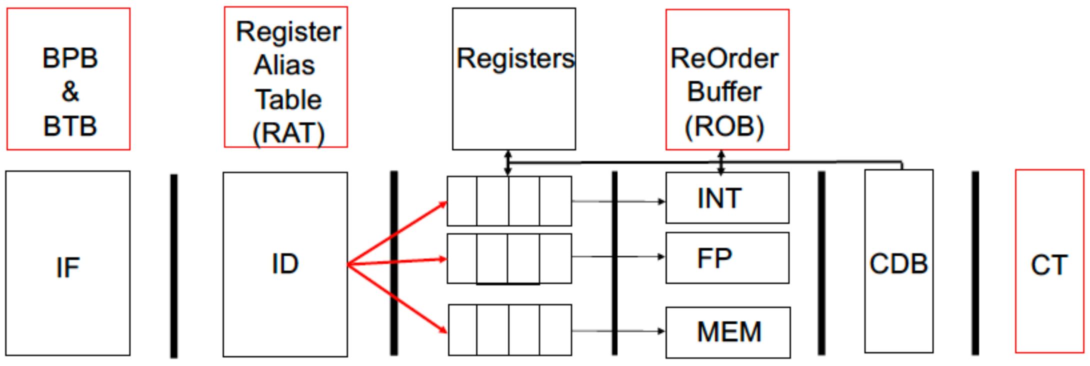
```
LOOP: LD F0, 0(R1)
      ADD F4, F0, F1
      SD F4, 0(R1)
      ADDI R1, R1,#8
      SUBI R3, R3,#1
      BNEZ R3, LOOP
```

ASSIGNMENT 3

The diagram below shows a pipeline with support for speculative execution. There are three functional units: one for integer/branch instructions (INT); one for floating-point instructions (FP) and one for memory instructions (MEM). There is a branch prediction mechanism (BPB) using 2 bits for prediction that are initialized to Not Taken. It takes two mispredictions to change the prediction. In addition, a branch target buffer (BTB) is in the IF stage.

For the functional units, it takes a single cycle to execute an INT instruction, three cycles for an FP instruction, a single cycle for a load and two cycles for a store in the MEM unit.

The pipeline supports register renaming using a register alias table (RAT) and data hazards are handled using the Tomasulo algorithm. Speculation is enabled by a reorder buffer and speculatively executed instructions are committed in the commit stage (CT).



Consider the following program:

```
I1: LOOP: L.S    F0, 0(R1)
I2:       ADD.S  F2, F1, F0
I3:       SD     F2, 0(R1)
I4:       ADDI   R1, R1,#8
I5:       SUBI   R3, R3,#1
I6:       BNEZ   R3, LOOP
```

3A) The Branch Predictor will predict that the branch is taken in the first iteration. Now consider the second iteration:

i) Explain in detail how the ROB and the RAT are instrumental in detecting the RAW hazard between I1 and I2 on one hand and I2 and I3 on the other. **(2 points)**
ii) What is the content of the ROB when instruction I4 is issued? (**2 points)**

3B) Show with a pipeline timing diagram, cycle-by-cycle, when each of the instructions I1-I6 enters a specific pipeline stage if I1 has entered ID at cycle 0 and when I6 reaches the CDB stage. **(3 points)**

3C) Determine which instructions are executed speculatively when the outcome of the branch in the first iteration is committed. (**3 points)**

3D) Assume that the loop in 3A) is executed 100 times. What is the fraction of correct branch predictions in percent? (**2 points)**

ASSIGNMENT 4

4A) Explain which of the statements, below, are **true** and why they are **true.**
In a two-level **exclusive** memory-hierarchy the following holds:

i) A block in the upper level always exists in the lower level
ii) A block in the upper level does not exist in the lower level
iii) A block in the lower level may also exist in the upper level
iv) A block in the lower level does not exist in the upper level

Note: A wrong answer cancels a correct answer. **(2 points)**

4B) A computer architect wants to establish the relative performance between a system with a blocking and a non-blocking cache and runs the following program:

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

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 20. Both caches (blocking and non-blocking) have a block size of four words. In the non-blocking cache, there are only 2 miss-status-holding registers.

How much faster does the program run on the system with a non-blocking cache. A convincing explanation that is easy to follow is needed for full points. (**6 points)**

4C) Consider a direct-mapped cache with 4 blocks and the following sequence of block accesses:

1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 1, 2, 3, 4, 17, 18, 1, 2

Determine the number of cold, capacity and conflict misses in the cache. Assume OPT when determining the number of capacity misses. (**4 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-back write policy. X_i=R_i and X_i=W_i, mean a read and a write request to the *same* address X from processor *i,* respectively, where W_i=*C* means that the value C is written by processor *i*. Now consider the following access sequence assuming that X is not present in any cache from the beginning and that X originally contains the value 0:

R_1
W_1=1
R_2
W_2=2
R_2
R_1

What is returned by the two last read operations from processor 1 and 2 and why doesn´t it conform with the programmer's expectation?
(3 points)

5B) How can a simple change of the cache protocol make sure that the correct value is returned to processor 2? **(3 points)**

5C) A computer architect wants to convert a five-stage pipeline with the stages IF, ID, EX, ME and WB to a multithreaded pipeline supporting four threads. Explain in detail how the pipeline must be changed in terms of new pipeline stages and architectural resources to support *fine-grain (or interleaved) multithreading.* (**6 points).**

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

Solutions to the exam in DAT105/DIT 051 2023-10-23

ASSIGNMENT 1

1A)
A: Execution times for P1 – P3 are given (2s, 10s and 2s, resp.) so we can calculate the arithmetic mean as follows **$\underline{T_{avg}=(2+10+2)/3=4.7s}$**

B: Speedup of B over A for P1 – P3 are given (0.5, 5 and 0.5, resp.) so the execution times for P1 – P3 on B are: 4s, 2s and 4s, resp, and the arithmetic mean is **$\underline{T_{avg}=(4+2+4)/3=3.3s}$**.

B is 4.7/3.3 =1.4 times faster, meaning 40% faster. It is interesting to see that while A is indeed faster on P1 and P3, P2 is an outlier which will greatly impact A's arithmetic mean which is very sensitive to outliers.

1B)
A: The speedup of A over R for P1 – P3 are given: 10, 10 and 10, resp. The geometric mean of speedup over the reference machine R is SP_{geom}=F(10x10x10), where F(x) is the third root of X which is **<u>10</u>**.

B: Since the speedup of B over A is given, we have that $SP_{A/R}= T_R/T_A$ and $SP_{B/A}= T_A/T_B$. Hence, $T_A=T_B x SP_{B/A}$ and $SP_{A/R}=T_R/(T_B x SP_{A/B})$. This gives us that $SP_{R/B}=SP_{A/B} x SP_{A/R}$ which can be found in the table as 5, 50, 5 for P1 – P3, resp.

Hence, T_{geom}= F(5x50x5) = $1250^{1/3}$ = **<u>10.8</u>**. By using geometric mean of speedup, we can conclude that B is faster than A by a factor 1.08 which is 8%. This is a much smaller gain than what was established with arithmetic mean (40%) in 1A.

1C)
A: We know the number of executed instructions, the CPI without cache misses and the clock frequency. We can then calculate the execution time for the program in the case of no cache misses using T_{exe} = IC x CPI x T_c.

P1 on A: T_{exe} = 10^8x 8 x 1 nanonsecond = 0.8 s. The time spent on cache misses is 2 – 0.8 s = 1.2 s. With a miss penalty of 100 ns, this corresponds to 1.2 x 10^7 misses over 10^8 instructions leading to 0.12 misses per instruction and **<u>120 MPKI</u>**.

P2 on A: T_{exe} = 10^9x 8 x 1 nanonsecond = 8 s. The time spent on cache misses is 10 – 8 s = 2 s. With a miss penalty of 100 ns, this corresponds to 2 x 10^7 misses over 10^9 instructions leading to 0.02 misses per instruction and **<u>20 MPKI</u>**.

B: We calculated the execution times of B on P1 and P2 in 1A): 4s and 2s, resp., so let's use the same methodology as above to calculate MPKI.

P1 on B: $T_{exe} = 10^8 \text{x } 1 \text{ x } 1$ nanonsecond = 0.1 s. The time spent on cache misses is 4 – 0.1 s = 3.9 s. With a miss penalty of 100 ns, this corresponds to 3.9 x 10^7 misses over 10^8 instructions leading to 0.39 misses per instruction and **390 MPKI**.

P2 on B: $T_{exe} = 10^9 \text{x } 1 \text{ x } 1$ nanonsecond = 1 s. The time spent on cache misses is 2 – 1 s = 1 s. With a miss penalty of 100 ns, this corresponds to 1 x 10^7 misses over 10^9 instructions leading to 0.01 misses per instruction and **10 MPKI**.

ASSIGNMENT 2

2A)

	C1	**C2**	**C3**	**C4**	**C5**	**C6**	**C7**	**C8**	**C9**	**C10**	**C11**	**C12**	**C13**	**C14**	**C15**	**C16**	**C17**
I1	IF	ID	EX	ME	WB												
I2		IF	ID	EX	ME	WB											
I3			IF	ID	FP1	FP2	FP3	FP4	FP5	ME	WB						
I4				IF	ID	FP1	FP2	FP3	FP4	FP5	ME	WB					
I5					IF	ID	EX	ME	WB								
I6						IF	ID	X	X	X	X	EX	ME	WB			

Since the floating-point functional unit is pipelined, I4 does not have to wait for I3 to complete. Data hazards are marked by X. I6 will suffer from data hazards with respect to I4 and will arrive in the ME stage in cycle 13. Since I1 issues to EX in cycle 3, the total number of cycles is 13-3 = 10 cycles

2B)

	C1	**C2**	**C3**	**C4**	**C5**	**C6**	**C7**	**C8**	**C9**	**C10**	**C11**	**C12**	**C13**	**C14**	**C15**	**C16**	**C17**
I1	IF1	IF2	ID	EX1	EX2	ME1	ME2	WB									
I2		IF1	IF2	ID	EX1	EX2	ME1	ME2	WB								
I3			IF1	IF2	ID	X	X	FP1	FP2	FP3	FP4	FP5	ME1	ME2	WB		
I4				IF1	IF2	ID	X	X	FP1	FP2	FP3	FP4	FP5	ME1	ME2	WB	
I5					IF1	IF2	ID	EX1	EX2	ME1	ME2	WB					
I6						IF1	IF2	ID	X	X	X	X	X	EX1	EX2	ME!	

Again, I1 issues for execution in cycle 3. I6 will be stalled due to a data hazard with respect to I4 in cycle 9 and cannot proceed to ME1 until cycle 16. Hence, the total number of cycles is 16 – 3 = 13. Since the clock frequency can be doubled, the performance of this pipeline for the program is 2x10/13 = 1.5, i.e., 50% higher.

2C)

```
LOOP: LD F0, 0(R1)         # O1
      ADD F3, F0, F1       # O2
      SD F3, 0(R1)         # O3
      ADDI R1, R1,#8
      SUBI R3, R3,#1
      BNEZ R3, LOOP
```

	ITE1	ITE2	ITE3	ITE4	ITE5	ITE6	ITE7
I1	O1						
I2		O1					
I3	O2		O1				
I4		O2		O1			
I5			O2		O1		
I6				O2		O1	
I7	O3				O2		O1
I8		O3				O2	
I9			O3				O2
I10				O3			
I11					O3		
I12						O3	
							O3

Prolog

Kernel

Epilog

The instruction word for the kernel (I7) is SD F4, 0(R1) LD F24, 48(R1) ADD F19,F16,F1

if we assume that LD uses F0,F4, …,F24 as destination operand in the first seven iterations of the original loop and SD uses F3, F7,…, F27 as source operand in the first seven iterations.

ASSIGNMENT 3

3A)

i)

When I1 is dispatched (speculatively), an entry for I1 will be created in the ROB with room for the result of the execution of I1, that is the content of F0. The RAT will register that F0's result is pending so that when I2 is dispatched it will be assigned the ROB entry of F0 and will wait until the instruction that supplies it (I1) will generate the result. The same procedure is applied to the RAW hazard between I2 and I3 and I3 will wait for the result to be available in the ROB.

ii)

I6 (from iteration 1)
I1 (from iteration 2)
I2 (from iteration 2)
I3 (from iteration 2)
I4 (from iteration 2)

3B)

```
I1: LOOP: L.S   F0, 0(R1)
I2:       ADD.S F2, F1, F0
I3:       SD    F2, 0(R1)
I4:       ADDI  R1, R1,#8
I5:       SUBI  R3, R3,#1
I6:       BNEZ  R3, LOOP
```

	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15
I1	ID	IS	ME1	CDB											
I2	IF	ID	IS	X	FP1	FP2	FP3	CDB							
I3			IF	ID	IS	X	X	X	ME1	ME2	CDB				
I4				IF	ID	IS	INT	Y	CDB						
I5					IF	ID	IS	INT	Y	CDB					
I6						IF	ID	IS	INT	Y	Y	CDB			

I6 reaches CDB in cycle 11. X marks data hazards and Y marks structural hazards.

3C)

.

	C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15
I6,1	IF	ID	IS	INT	CDB	CT									
I1,2		IF	ID	IS	ME1	CDB									
I2,2			IF	ID	IS	X	FP1	FP2	FP3	CDB					
I3,2				IF	ID	IS	X	X	X	ME1	ME2	CDB			
I4,2					IF	ID	IS	INT	CDB						
I5,2						IF	ID	IS	INT	Y	CDB				

In the table, I6,1 corresponds to the sixth instruction in the first iteration (the branch). Ix,2 refer to the instructions in the second iteration. We can see that I6,1 will commit in cycle 6. At this time, I2,1 – I4,2 are being speculatively executed and will be either committed or squashed when the branch is correctly predicted or mispredicted, respectively. Hence, I1 -I4 in the second iteration will be speculatively executed when the branch in the first iteration is committed.

3D) Since the predictor is initialized to Not-Taken, the first two branches (that are taken) are mispredicted and the last branch (that is not taken) is also mispredicted. The rest of the branches (97) are correctly predicted so the fraction of correctly predicted branches is 97/100 = 97%.

ASSIGNMENT 4

4A) Explain which of the statements, below, are **true** and why they are **true.**
In a two-level **exclusive** memory-hierarchy the following holds:

i) **False** because in an exclusive memory hierarchy a block can only be at one level.
ii) **True** because in an exclusive memory hierarchy a block can only be at one level.
iii) **False** because in an exclusive memory hierarchy a block can only be at one level.
iv) **True** because in an exclusive memory hierarchy a block can only be at one level

4B)

Blocking cache:
The first iterations takes 20 + 4 = 24 cycles
The next three iterations take 3x5 = 15 cycles as all accesses hit in the cache (block size 4).
Hence, the first four iterations take 39 cycles
All iterations take 39x1000/4 = 250 x 39 cycles = **9750 cycles**

Non-blocking cache:
If we had a sufficient number of miss-status-holding registers (MSHRs), the 20 cycles miss penalty (20 cycles) could be perfectly hidden behind the execution of the 20 instructions of four iterations. Unfortunately, there are only 2 MSHRs. Therefore, the non-blocking cache will not yield any performance improvement beyond the blocking cache.

4C) Consider a direct-mapped cache with 4 blocks and the following sequence of block accesses:

1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 1, 2, 3, 4, 17, 18, 1, 2

Compulsory misses (same as unique block addresses): 12

Capacity misses: (4 blocks, fully assoc., OPT replacement alg.)
```
Replace:  - - - - 4 5 6 7 8 9  - - -10 4  3  - -
Sequence: 1 2 3 4 5 6 7 8 9 10 1 2 3 4 17 18 1 2
Outcome:  M M M M M M M M M M  H H H M M  M  H H
```
Total: 13
Capacity=Total – Cold = 1

Conflict misses: (direct-mapped, 4 blocks)

```
Replace:  - - - - 1 2 3 4 5 6  9 10 7 8 1  2 17 18
Sequence: 1 2 3 4 5 6 7 8 9 10 1 2  3 4 17 18 1 2
Outcome:  M M M M M M M M M M  M M  M M  M  M M M
```

Total: 18

Conflict= Total – Cold - Capacity = 18 – 12 – 1 = 5

ASSIGNMENT 5

5A) The second read from processor 1 should return 2 reflecting the second write by processor 2 but will return the value 1 as the processor will read from its own cache.

5B) By simply invalidating the block in another cache when it sees a write to that block. This will force the second read to experience a cache miss that will return the correct value. This is done in the MSI protocol when the state is S.

5C) In fine-grained (or interleaved) multithreading, the pipeline will switch to a new thread in each cycle in a round-robin fashion. For this reason a new stage is inserted between IF and ID. Apart from that the pipeline needs four program counters, one for each thread, and four registerfiles, one for each thread.