CHALMERS

2021-10-15

Exam in DAT 105 (DIT 051) Computer Architecture

Time: October 25, 2021, 14-18 (Lindholmen)

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: Bonus points from Real-stuff studies (4p) and Quizzes (4p) 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

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

A computer design team at Intel is tasked to come up with a new processor design for the next product that can offer higher performance than the existing product on the market. They have good ideas for how to do that but need your help to come back with which option they should choose.

The current product is a multicore architecture with each processor (core) attached to a first-level cache, where all first-level caches are connected to a second-level cache. The second level cache is connected to off-chip memory.

They have analyzed the bottlenecks in the existing product and have found the following:

- Floating-point operations take 10 cycles
- The cache hit time in the first level cache is 2 cycles
- The cache miss penalty from the first level to the second level cache is 20 cycles
- The cache miss penalty from the second level cache to the off-chip memory is 50 cycles
- The miss rate from level one to level two and from level two to the off-chip memory is 10% and 50%, respectively
- The CPI for instructions instructions other than floating-point or memory instructions is 1
- The relative frequency of instructions is 25% floating point, 25% memory instructions
- The operating frequency is 2 GHz

The team has measured the execution times of three programs (P1-P3) on the existing product and found that they are 4, 2 and 6 seconds, respectively.

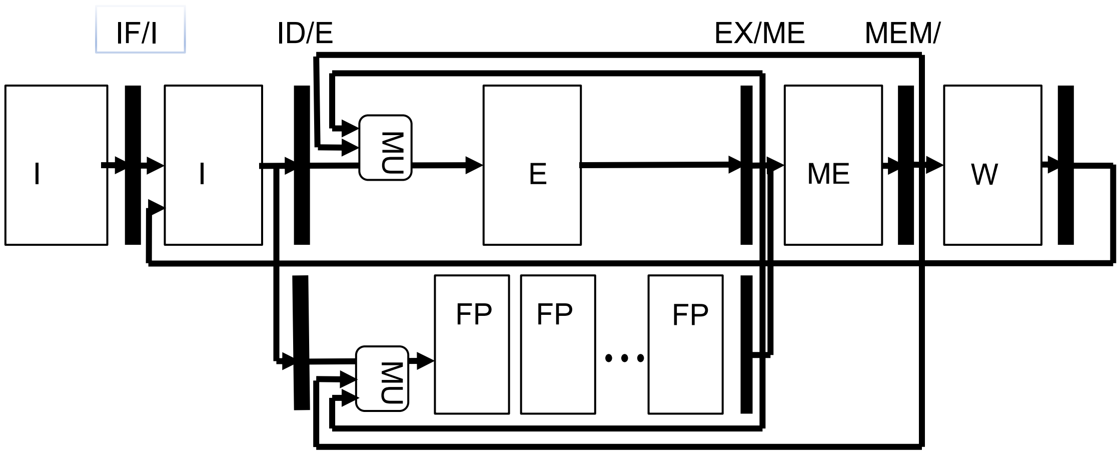
A) What is the execution time of P1-P3 on i) a processor that can speedup floating-point operations by a factor of two and ii) a processor that can speedup the miss-penalty from the second level cache to the off-chip memory by a factor of two. (**8 points)**

B) Determine the best choice of i) and ii) given the geometric means of the two choices. (**2 points)**

C) Which of the two alternatives (i) and ii) in A) would be chosen if we would be able to completely remove the overheads associated with i) floating-point operations and ii) the overhead of misses in the second level cache, respectively? (**2 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.



A design team at AMD is tasked to come up with a pipeline for the next technology node. The pipeline for the existing product is shown above. The following is known for it:

- The clock frequency is 1 GHz
- The combinatorial delay through each of the stages is as follows: 1 ns, 0.5ns, 1ns, 1 ns and 0.5 ns for the IF, ID, EX, MEM and WB stages, respectively, and 5 ns for each of the five FP stages.

The design team now wants to propose a super-pipelined version that can run at twice the clock speed.

A) Propose a super-pipelined design that can cope with the higher clock frequency (**4 points**)

B) What will be the latency of the following instructions on the new pipeline: ADD, LD, and BEQZ? (**4 points**)

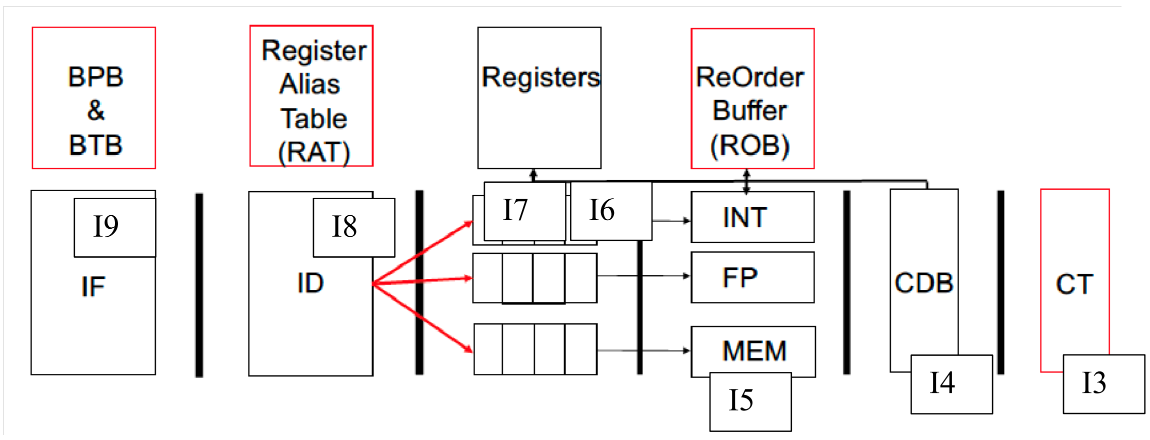
C) Determine the number of cycles it takes to execute I1-I6 below on the new pipeline from the first instruction enters the EX/FP unit until the last instruction exists from the WB stage. (**4 points**)

```
I1: LABEL: LD F1,0(R1)
I2:        ADD F3,F2,F1
I3:        SD F3,0(R1)
I4:        ADDI R1,R1,#8
I5:        SUBI R2,R2,#1
I6:        BNEZ R2,R0, LABEL
```

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, meaning that the prediction is changed every in response to two mispredictions in a row. 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:       L.S    F1, 0(R2)
I3:       ADD.S  F2, F1, F0
I4:       ADD.S  F1, F3, F4
I5:       SD     F4, 0(R1)
I6:       ADDI   R1, R1,#8
I7:       ADDI   R2, R2,#8
I8:       SUBI   R3, R3,#1
I9:       BNEZ   R3, LOOP
```

3A)
The diagram shows a snapshot of the execution of the program and in which pipeline stage each individual instruction is in that snapshot. Which of the operands are available in the register file and which are available from the ROB? **(3 points).**

3B)
Please fill out the content of the RAT given the content of the ROB below for the instructions under execution and their operands. **(3 points)**

RAT

Register	Tag	Status
R1	--	--
R2	--	--
R3	--	--
F0	--	--
F1	--	--
F2	--	--
F3	--	--
F4	--	--

ROB

Entry	Instruction	Dest.reg	Value	Complete
16	ADD.S F2,F1,F0	F2	1.2E-4	YES
17	ADD.S F1,F3,F4	F1	5.8E-3	YES
18	SD F2,0(R1)	--	N/A	NO
19	ADDI R1,R1,#4	R1	N/A	NO
20	ADDI R2,R2,#4	R2	N/A	NO
21	SUBI R3,R3,#1	R3	N/A	NO
22	BNEZ R3,Loop	--	N/A	NO

3C) Explain in detail what happens in the Commit (CT) stage and the ROB when a branch that is mispredicted is committed (**4 points)**

3D) What happens if there is a miss in a branch target buffer? (**2 points)**

ASSIGNMENT 4

4A) Explain which of the statements, below, are **not true** and why they are **not true.**
For a lockup-free (or non-blocking) cache the following holds: **(2 points)**

i) It blocks if there is an outstanding cache miss to the same block
ii) It blocks on primary misses
iii) It blocks on secondary misses
iv) It blocks only if all MSHRs are being in use

4B) Let's assume a VLIW architecture where two memory instructions, one floating-point instruction and an integer instruction can be scheduled in each cycle. Consider the following code:

```
LOOP: L.D F0, 0(R1)       – O1
      ADD.D F4,F0,F2      – O2
      S.D F4, 0(R1)       – O3
```

Software pipeline this loop and show the prolog, kernel, and epilog. (**6 points)**

4D) Consider a fully associative cache with 8 blocks and the following sequence of block accesses:

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 0, 1, 2, 4, 5, 6, 7, 8

Determine the number of cold, capacity and conflict misses in the cache. Assume OPT for determination 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. 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:

P1 and P2 will issue the following two sequences of memory operations:

R_1 W_1=1 R_2 W_2=2 R_2 R_1

What should be returned by each of the four read requests in a coherent cache system? You should motivate your answer. (**4 points)**

5B)

We want to use software prefetching to hide the latency of cache misses. In the code below, the block size is four vector elements and the cache miss latency is the same as the time to execute two iterations. Insert a prefetch instruction (PREF disp(Rx)) so that cache miss latency is hidden. (**4 points)**

```
LOOP: L.D F2,0(R1)
      ADD.D F4,F2,F0
      S.D F4,0(R1)
      SUBI R1,R1,#8
      BNEZ. R1,LOOP
```

5C) Multithreading refers to a general technique to switch to another thread when a high latency operation is encountered. Show how fine-grained multithreading is integrated in a five-stage pipeline. (**4 points).**

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

Solutions to the exam in DAT105/DIT 051 2021-10-25

ASSIGNMENT 1

A) Let's first establish how much time that the existing product (denoted R below) spends on
   i) Execution of floating-point operations
   ii) Handling misses to memory

The execution time of a program on R is
$T = T_{FP}+T_{off\text{-}chip}+T_{other}$, where T_{FP}, $T_{off\text{-}chip}$, and T_{other} are the execution times for handling floating-point, off-chip memory accesses and others, respectively.

$T= IC \times CPI \times Tcc = (IC_{FP} \times CPI_{FP} + IC_{off\text{-}chip} \times CPI_{off\text{-}chip} + IC_{other} \times CPI_{other}) \times Tcc$ (1)

The fractions of instructions in the three categories are given (FP=25%, Off-chip=25% and others=50%). For a given program with IC instructions, it holds that

$T= IC \times (0.25 \times CPI_{FP} + 0.25 \times CPI_{off\text{-}chip} + 0.50 \times CPI_{other}) \times Tcc$

where

$CPI_{other} =1$ according to the assumptions
$CPI_{FP} = 10$ according to the assumptions
$CPI_{off\text{-}chip}= 2+MR_{level\text{-}one} \times (MP_{level\text{-}one} + MR_{level\text{-}two} \times MP_{level\text{-}two})= 2+0.1 \times (20 + 0.5 \times 50)=6.5$

$Tcc= (1/(2 \times 10^9)$ s = 0.5 nanoseconds.

Given the execution time of the three programs, we can establish the number of instructions executed in each of the three programs:

- P1: Execution time 4 seconds: $IC = T/(0.25 \times CPI_{FP} + 0.25 \times CPI_{off\text{-}chip} + 0.50 \times CPI_{other})Tcc) = 4/((0.25 \times 10+0.25 \times 6.5+0.5 \times 1) \times 0.5 \times 10^{-9})=1.7 \times 10^9$ instructions
- P2: Execution time 2 seconds: 0.85×10^9 instructions
- P3: Execution time 6 seconds: 3.4×10^9 instructions

Now we can establish the execution time for P1-P3 on a new product that can execute floating point operations twice as fast as the execution time for each of the programs is given.

- P1: $T = 1.7x10^9x(0.25x10/2 + 0.25xCPI_{off\text{-}chip} + 0.50xCPI_{other})Tcc = 1.7x10^9x(0.25x5 + 0.25x6.5+ 0.5x1)0.5x10^{-9}$ = 2.8 seconds
- P2: T = 1.4 second
- P3: T = 4.2 seconds

For the product that speedups the handling of off-chip accesses by a factor of two we can calculate the $CPI_{off\text{-}chip}$ = 2 + 0.1x(20 + 0.5x50/2)= 5.3

- P1: $T = 1.7x10^9x(0.25x10 + 0.25xCPI_{off\text{-}chip} + 0.50xCPI_{other})Tcc = 1.7x10^9x(0.25x10 + 0.25x5.3 + 0.50x1)0.5x10^{-9}$ = 3.7 seconds
- P2: T = 1.9 second
- P3: T = 5.6 seconds

Hence, it is more important to execute the floating-point operations faster.

B)

Let's refer to the current product as R, the proposed system with accelerated floating-point operations as A and the proposed system with accelerated off-chip memory accesses as C and with the execution times of program P_i on the three systems as $T_{R,Pi,}$ $T_{A,Pi,}$ and $T_{B,\ Pi,}$ respectively. The geometric mean of system A is defined as

$G=((T_{R,P1}/\ T_{A,P1})x(T_{R,P2}/\ T_{A,P2})x(T_{R,P3}/\ T_{A,P3}))^{1/3}= ((4/2.8)x(2/1.4)x(6/4.2))^{1/3}=1.4$

This is the geometric mean performance improvement of speeding up floating-point operations by a factor of two.

The same methodology can be applied to speed up off-chip memory accesses and the performance improvement will not be as much.

C)

If we could cancel the performance overhead of floating point operations on A, the execution time for P1-P3 on A would be:

- P1: $T = 9.4x10^9x(0.25xCPI_{off\text{-}chip} + 0.50xCPI_{other})Tcc = 1.7x10^9x(0.25x6.5+ 0.50x1)0.5x10^{-9}$ = 1.8 seconds
- P2: T = 0.9 second
- P3: T = 2.7 seconds

On the other hand, if we could cancel the performance overhead of off-chip accesses on B, the execution time for P1-P3 on B would be:

- P1:$T=1.7x10^9x(0.25x10 + 0.50xCPI_{other})Tcc = 1.7x10^9x(0.25x10 + 0.50x1)0.5x10^{-9}$ = 2.6 seconds
- P2: T = 1.3 second
- P3: T = 3.9 second

ASSIGNMENT 2

A) The clock frequency for the new design is doubled (2 GHz) with a clock cycle time of 5 ns. Since the IF, EX and ME stages have a combinatorial delay of more than 10ns, they must be partitioned into two stages each: Hence IF becomes IF1 and IF2, EX becomes EX1 and EX2 and ME becomes ME1 and ME2.

B) The latency of an instruction is the number of cycles a subsequent (dependent) instruction has to wait before it can be executed. An integer ADD instruction will now take two cycles so the latency is one cycle. A load instruction will now take one additional cycle (two memory stages) and results in two cycles latency. On the new pipeline it will take two more cycles until the branch condition is established so the latency is four instead of two.

C)

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

It takes 16 cycles until the last instruction has reached the WB stage.

ASSIGNMENT 3

3A) Since I1 has committed (it is not in the ROB), F0 is available in the registerfile. Moreover, F3 and F4 are also in the registerfile as no instruction use them as destination operands. All other register values, if the instruction has completed is available from the ROB. This applies to F1 and F2 whose values are available from the ROB at entry 17 and 16, resp.

3B)

Filled out RAT:

RAT

Register	Tag	Status
R1	19	Pending
R2	20	Pending
R3	21	Pending
F0	--	Commit.
F1	17	Pending
F2	16	Pending
F3	--	Commit
F4	--	Commit

3C) The commit stage flush the branch instruction and all subsequent instructions from the ROB. Since the registerfile reflects the state when the branch instruction is to be executed, one can start executing the instructions from the path corresponding to the outcome of the branch instruction.

3D) The branch will execute as it used to do but instructions will not be fetched until the branch target has been established. The established branch target will be inserted into the branch target buffer.

ASSIGNMENT 4

A) All statements except iv) are not true. The reason is that a non-blocking cache will be able to service cache requests for as long as they can be recorded in the MSHR. So it certainly does not block if there is an outstanding miss to the same block (a secondary miss). A primary miss is miss to a block where there are no recorded non-outstanding misses. Hence ii) is not correct.

B) See the example in Table 3.20 in the textbook.

C) Number of cold misses is the number of unique blocks in the sequence: 11

For capacity misses, we need to simulate the sequence on a fully associative cache with 8 blocks and the OPT replacement algorithm

The first 8 accesses will fill the cache with blocks 0-7. On the ninth access 8 will replace 3 as 3 is not accessed. 9 will replace 8 as 8 is accessed furthest into the future. 10 will replace 9 as 9 is not accessed. 1,2,4,5,6,7 will hit and 8 will miss. So in total there will be 12 misses meaning that the number of capacity misses is 1.

There are no conflict misses as the cache is fully associative.

ASSIGNMENT 5

5A)

The first read happens before the first write so it returns 0. The second read happens after the first wrote so it returns 1. The last two reads happens after the second write so they return 2.

5B) There is a miss every fourth iteration as a block contains four elements. Hence we must prefetch the block accessed four iterations ahead. This corresponds to a displacement of 3 x(-8) = -24 as the address register is decremented by 8 in each iteration. Hence:

```
LOOP: L.D F2,0(R1)
      ADD.D F4,F2,F0
      S.D F4,0(R1)
      PREF -24(R1)
      SUBI R1,R1,#8
      BNEZ. R1,LOOP
```

5C)

A new pipeline stage, thread select (TS) is inserted between the IF and the ID stage. If two threads are used (T0 and T1), TS will alternative between T0 and T1 so that an instruction from each threafd s fetched in two subsequent cycles.