One hour

Special instructions:

- You are expected to complete the exam independently and not confer with anyone during the examination.
- You must not make copies of any of the questions (either written or screenshot)
- You must not share the details of this assessment with anyone
- You must not approach or ask anyone for support with this assessment
- The answers you provide must be your own work and by submitting your answers you are confirming that it is your own work.
- Remember the University's rules on plagiarism and academic malpractice (see: http://www.regulations.manchester.ac.uk/guidance-to-students-on-plagiarism-and-other-forms-of-academic-malpractice/).
- If you are undertaking a professionally regulated degree programme, you'll also be bound by their code of conduct.

THE UNIVERSITY OF MANCHESTER

Faculty of Science and Engineering

School of Engineering

Department of Electrical and Electronic Engineering

Computer Systems Architecture

Academic Year 2020/21 Semester 2

Solutions

Question 1

(a)

(i) Execution time = total number of clock cycles / F_{CLK}
= (15000 x 2 + 20000 x 2 + 4000 x 3 + 6000 x 3 + 10000 x 6) / (3.2 x 10^9) = **50 us**.

(ii) Average CPI = Execution time / Total instructions x F_{CLK} =
= 50 x 10^{-6} / (15000 + 20000 + 4000 + 6000 + 10000) x (3.2 x 10^9) = **2.91**

(iii) The maximum speedup will be obtained when the CPI for the FPADD and FPSUB instructions is 1.

The execution time for the program with these changes to the instruction CPIs will be given by:

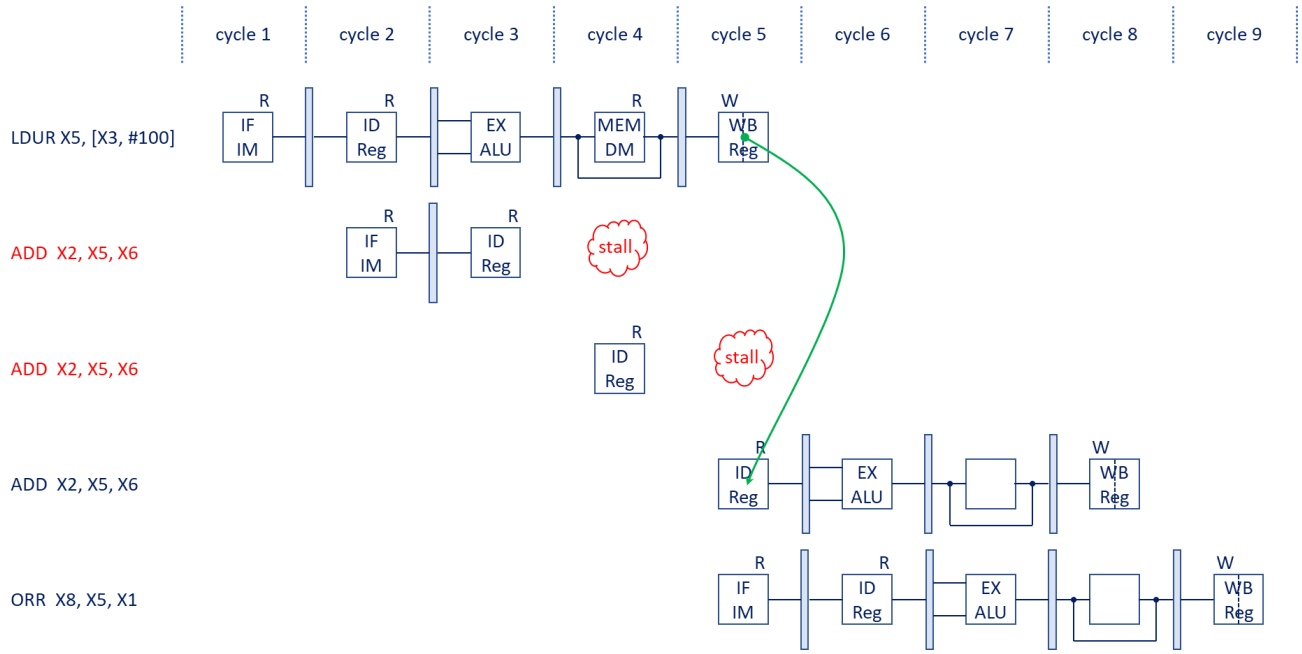
New Execution time = total number of clock cycles / F_{CLK}
= (15000 x 2 + 20000 x 2 + 4000 x 1 + 6000 x 1 + 10000 x 6) / (3.2 x 10^9) = 43.75 us.

Speedup = 50.0 / 43.75 = **1.14**

[3, 3, 4 marks]

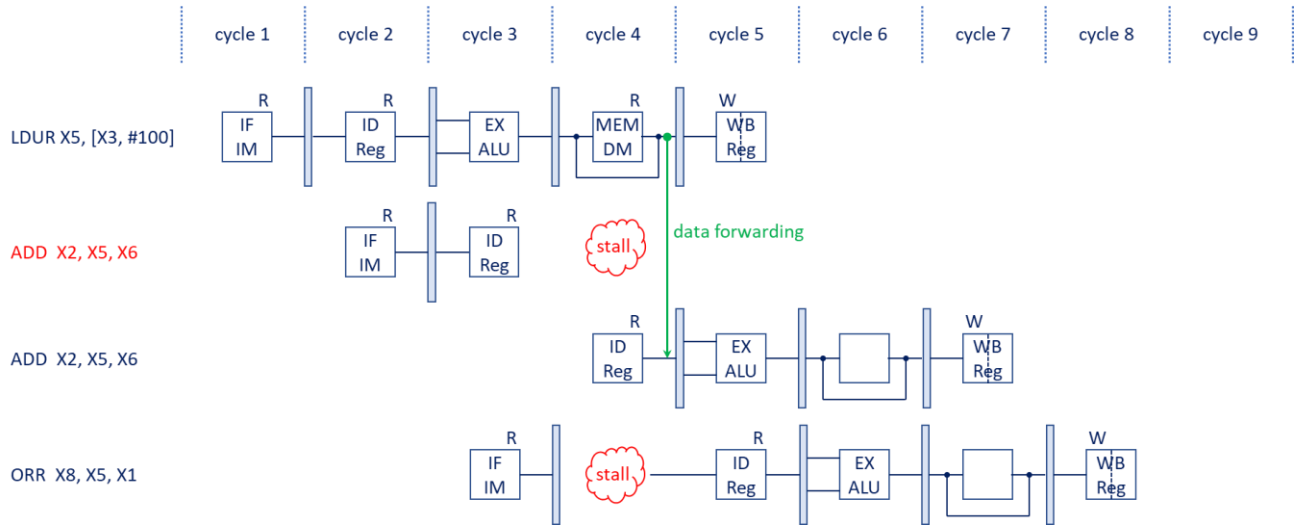
(b)

(i)



- Deduct 2 marks for not correctly indicating all 5 pipeline stages
- Deduct 2 marks for not clearly identifying cycles
- Deduct 2 marks for not indicating a double pipeline stall for the ADD instruction
- Deduct 1 mark for each other error

(ii)



- Deduct 2 marks for not correctly indicating all 5 pipeline stages
- Deduct 2 marks for not clearly identifying cycles
- Deduct 2 marks for not indicating a single pipeline stall for the ADD instruction
- Deduct 2 marks for not correctly indicating the forwarding
- Deduct 1 mark for each other error

(iii) By exploiting the fact that the ADD X4, X6, X6 instruction does not have any data dependencies, the instruction order can be changed as shown below to remove the data dependency for register X5. This removes the stall shown the pipeline diagram for part (ii).

```
LDUR X5, [X3, #100]

ADD  X4, X6, X6

ADD  X2, X5, X6

ORR  X8, X5, X1
```

[6, 6, 3 marks]

Total [25 marks]

Question 2

(a)

(i) Number of cache blocks = 128 x 2^{10} / (64 x 2) = **1024**

(ii) Bits required for block offset = $Log_2(64)$ = 6
Bits required for cache block ID = $Log_2(1024)$ = 10
Tag bits = 24 – 6 – 10 = **8 bits**

(iii) Address = 0x3F2B7E = 0011 1111 0010 1011 0111 1110
Cache block ID is bits A_{15} … A_6 = 1010 1101 = **0xAD**

(iv) A main memory address is structured as follows:

T T T T T T T T C C C C C C C C C C B B B B B B

T – tag bit
C – cache block ID
B – block offset

The highest address will have the following bit pattern:

1 1 1 1 1 1 1 1 0 0 0 0 0 1 0 1 0 0 1 1 1 1 1 1

= **0xFF053F**

[2, 2, 3, 4 marks]

(b)

(i)

PUSH A	[TOS] = [A]
POP A	[A] = [TOS]
ADD	[TOS]= [TOS] + [TOS–1]
MUL	[TOS]= [TOS] * [TOS–1]

```
PUSH A      ; [A]
PUSH B      ; [B, A]
PUSH C      ; [C, B, A]
ADD         ; [(C+B), A]
ADD         ; [(C+B+A)]
PUSH A      ; [A, (C+B+A)]
PUSH C      ; [C, A, (C+B+A)]
ADD         ; [(C+A), (C+B+A)]
MUL         ; [((C+A) x (C+B+A))]
POP A       ; [ ]
```

There are multiple correct solutions to implementing the arithmetic statement.

Marking:

- 2 marks for explicitly defining the four required operations
- 3 marks for correct instruction sequence
- 2 marks for correct commenting (stack content) of all instructions

(ii) Maximum stack depth (for this implementation) is 3
Maximum memory requirement = 3 x 4 = **12 bytes**

(iii) Execution time for a PUSH/POP instruction is 50 ns IM read + 50 ns DM read + 50 ns DM write = 150 ns.

Execution time for an ALU instruction is 50 ns IM + read 50 ns DM read + 50 ns DM read + 5 ns ALU operation + 50 ns DM write = 205 ns.

The program contains 6 PUSH / POP instructions and 4 ALU instructions. Therefore, the total execution time is 6 x 150 + 4 x 205 = **1720 ns**.

[7, 2, 5 marks]

Total [25 marks]