CSC 230
Spring 2023 (CRN 20763/20764)
Midterm #2: Thursday, 6 April 2023

Time allowed: 60 minutes

	LAST NAME	FIRST NAME	STUDENT NUMBER
STUDENT INFO*			V

* Please type your name as displayed in Bright Space

Marking Guide

Please remember the golden rule about writing an exam:

Don't Panic!

Students must check the number of pages in this examination paper before beginning to write their answers and report any discrepancy immediately.

- **All answers are to be written on this exam paper.**
- The exam is closed book. Other than the AVR instruction set.
- When answering questions, please do not detach any exam pages!
- The total marks for this exam is 40.
- There are 7 printed pages in this document, including this cover page.
- We strongly recommend you read the entire exam through from beginning to end before starting on your answers.
- **Please have your UVic ID card available for inspection by an exam invigilator.**

Question 1 (10 marks 5 each): A processor is clocked at a 16 MHz and executes 5 different instruction types as the following:

- Instruction type 1: Arithmetic (CPI = 2, 22% of instructions)
- Instruction type 2: Logic (CPI = 1, 40% of instructions)
- Instruction type 3: Load/store from cache (CPI = 2, 10% of instructions)
- Instruction type 4: Branch (CPI = 4, 16% of instructions)
- Instruction type 5: Load/store from outside cache (CPI=8, 12% of instructions)

A) What would be the MIPS rate for this CPU?

$$CPI = (2 \times 0.22) + (1 \times 0.40) + (2 \times 0.1) + (4 \times 0.16) + (8 \times 0.12) = 2.64$$

$$\text{MIPS rate} = \frac{16 \times 10^6}{2.64 \times 10^6} \cong 6.1$$

B) How much speedup will be gained if the CPI of the arithmetic instructions reduced to 1 clock?

$$CPI_{new} = (1 \times 0.22) + (1 \times 0.40) + (2 \times 0.1) + (4 \times 0.16) + (8 \times 0.12) = 2.42$$

$$\text{speedup} = \frac{\text{execution time } \textbf{before} \text{ enhancement}}{\text{execution time } \textbf{after} \text{ enhancement}} = \frac{I_C \times CPI_{old} \times \tau}{I_C \times I_C \times CPI_{new} \times \tau} = \frac{2.64}{2.42} \approx 1.1$$

Question 2 (8 marks 1 each): Briefly answer the following:

a) What is the difference between DRAM and SRAM in terms of application?

DRAM → Main memory

SRAM → Cache

b) What are some application of ROM?

ROM → store essential code such as BIOS

c) What's an interrupt handler?

A code or a function to be executed to handle the interrupt.

d) What's a timer? → any of the following is correct:

A timers is a hardware device that keeps a track of an elapsed time between two events.

A timer is simply a counter that counts the number of clock pulses and signals the CPU when it reaches a predefined count.

e) What are the two methods used to transfer parameters and return values between a caller code and a callee?

- Passing parameters & return values in Registers
- Passing parameters & return values using the stack.

f) What's the difference between calling-by-reference and calling-by-value?

- In calling by reference a pointer (address) of the parameter is passed
- In calling by value, the actual value of a parameter is passed.

g) What does a function prototype refer to in C?

- It refers to information used to help the C compiler to know about the function to be called.

h) What does a **volatile** keyword means in a C program and what's the use for it?

- A volatile refers to a variable that could change by hardware or other context.
- It enforces the compiler to continuously use a fresh value of the variable.

Question 3 (10 marks): The following part of C code calls a function written in assembly. The parameters val1 and val2 are passed into r25:r24 and r23:r22 and the return value will be passed back into r25:r24

```
uint16_t val1, val2, val3;

val1 = 0x216;
val2 = 0x037C;
val3 = add_uint16_to_uint16(val1, val2);
```

Complete the following assembly code:

```
.global add_uint16_to_uint16

add_uint16_to_uint16:
    ; write your function after this line
```

```
add r24, r22
adc r25, r23
```

```
    ret
```

Question 4 (12 marks): Write in AVR assembly a function named **`look_at_either_end`** which takes a parameter passed-by-value in `r16`.

- If bit 0 is set in the parameter, but not bit 7, then return the value 3.
- If bit 7 is set in the parameter, but not bit 0, then return the value 1.
- If both bits 0 and 7 are set in the parameter, then return the value 2.
- For any other combination of bits in the parameter, return the value 0.

The return value is to be placed `r25`.

Note that there are few other restrictions:

- You may not destroy the value passed into `r16` (or rather, the value in `r16` when the function starts must be the same value in `r16` when the function returns).
- In your solution you must not use `sbr`, `sbrc`, or `sbrs`.

Some marks will be given for the quality of your answer.

Please note: There are multiple solutions for this problem. You may test by writing little program to call the student's solution function. Make sure that restrictions are considered. Also, consider quality of code, for example: clear code, comments, shorter (please also note that replacing the lines "rjmp end" by just "ret" gives the same answer and shorter execution).

```
look_at_either_end:
       mov  r25, r16
       andi r25, 0b10000001 ; filter b7 & b0
       cpi  r25, 0b10000000 ; b7 high and b0 low
       brne one
       ldi  r25, 1
       rjmp end
one:
       cpi  r25, 0b00000001 ; b7 low and b0 high
       brne two
       ldi  r25, 3
       rjmp end
two:
       cpi  r25, 0b10000001 ; both b7 and b0 are high
       brne three
       ldi  r25, 2
       rjmp end
three:
       clr  r25
end:
       ret
```

FORMULA SHEET

$$\text{speedup} = \frac{\text{execution time } \textbf{before} \text{ enhancement}}{\text{execution time } \textbf{after} \text{ enhancement}}$$

$$\text{Speedup} = \frac{T}{T(1-f)+\frac{Tf}{N}} = \frac{1}{(1-f)+\frac{f}{N}}$$

$$\text{Overall } CPI = \frac{\sum_{i=1}^{n}(CPI_i \times I_i)}{I_c}$$

$$Program\ execution\ Time = I_C \times CPI \times \tau$$

$$\text{Overall CPI} = \frac{(CPI_{ALU} \times I_{ALU}) + (CPI_{MEM} \times I_{MEM}) + (CPI_{BR} \times I_{BR}) + (CPI_J \times I_J)}{I_c}$$

$$\text{MIPS rate} = \frac{I_c}{T \times 10^6} = \frac{f}{CPI \times 10^6}$$

$$Average\ time\ to\ read\ or\ write\ n\ bits\ T_n = T_{Access} + \frac{n\ (bits)}{R\ (bps)}$$