02155 - Assignment 1

Practical information

This is a deliverable assignment, meaning that you have to write a small report in English with the requirements listed below. The grade you will obtain for the report will be part of the final grade. There should be one report per group. Groups may contain at most two students. At this point, you do not need to communicate your group members, after we receive your reports we will register the groups and assign you a group number to be used for the final project (the second deliverable assignment is individual).

The report should have a front page similar to the one appended at the end of this document. There are no other requirements on the template to be used for the report.

The report should include the following:

- Detailed problem solutions for the exercises in the section Problems.
- Detailed answers to questions specified in each exercise in the section Lab Exercises.
- Assembly code of the designed procedures properly formatted and commented.

The report should be handed-in only in electronic format (PDF and assembly code as appendix in the PDF) using the Assignment utility in DUT Inside. Hand-in the report as a group.

Feel free to ask or send an e-mail or use Slack if you have questions regarding the practical information and the assignment in general.

Problems

Exercise A1.1 - Textbook problems

Solve the following four problems from the lecture textbook[1].

- Problem 2.34 (page 169).
- Problems 2.39.1 and 2.39.2 (page 170).
- Problem 3.32 (page 230).

[1] David A. Patterson, John L. Hennessy, *Computer Organization and Design - The Hardware/Software Interface*, Risc-V Edition, Morgan Kaufmann, 2017.

Exercise A1.2 - Exam problem (fall 2009)

Consider a benchmark program containing 10^6 instructions with the following mix of operations.

Floating-point multiplication	Floating-point addition	Floating-point division	Other instructions
20%	30%	10%	40%

A processor P1 requires the following number of cycles to execute these instructions.

Floating-point multiplication	Floating-point addition	Floating-point division	Other instructions
8	5	30	4

a) Determine the CPI for the benchmark program.

b) For a clock rate of 200 MHz, what is the CPU execution time?

c) A new divide unit is introduced in the processor. This new divider requires 10 cycles to compute a floating-point division, but the clock cycle is 20% longer. The latency, in terms of cycles, in the other instructions is not changed. What is the speed-up of the new processor over processor P1?

Lab Exercises

In this section of the assignment your task is to write an assembly program capable of calculating the product of two complex numbers. Remember that the product of the two complex numbers $z = a + \boldsymbol{i}b$ and $w = c + \boldsymbol{i}d$ can be expanded as follows

$$(a + \boldsymbol{i}b)(c + \boldsymbol{i}d) = (ac - bd) + \boldsymbol{i}(ad + bc)$$

The program should be able to handle 32-bit signed and unsigned integers where the result fits in a 32-bit register. You do not have to handle overflow. All the procedures you write in this exercise should follow the caller/callee conventions outlined in Section 2.8 and the green sheet of the lecture textbook[1]. You should avoid using the frame pointer by only changing the stack pointer on entry and exit of a routine.

You can use the following template for your solution.

```
        # Course 02155, Assignment 1, A1 template

        .data
aa:     .word a         # Re part of z
bb:     .word b         # Im part of z

cc:     .word c         # Re part of w
dd:     .word d         # Im part of w
```

```
        .text
        .globl main
main:
        lw a0, aa
        lw a1, bb
        lw a2, cc
        lw a3, dd
        jal complexMul # Multiply z and w
        nop
        j end          # Jump to end of program
        nop

        ###################################
        #                                 #
        #         YOUR CODE HERE          #
        #                                 #
        ###################################

end:
        nop
```

Exercise A1.3 - Integers product

Write an assembly routine `myMult` that takes 2 arguments, a and b in registers `a0` and `a1` respectively, and returns their product in register `a0` without using the dedicated RISC-V multiplication instruction. Figure 3.4 (page 185) of the lecture textbook[1] contains an algorithm for integer multiplication, but you must extend it to handle signed numbers.

When you have written `myMult` answer the following questions.

a) Is `myMult` a leaf or non-leaf procedure?

b) Did you need to use the stack? Explain why.

c) What are the overflow conditions for your procedure?

Exercise A1.4 - Complex numbers product

Write an assembly routine `complexMul` that takes two complex numbers z and w and returns their product. This routine should make use of the subroutine you wrote in Exercise A1.3. The routine should expect the real parts of z and w to be placed in `a0` and `a2`, and the imaginary parts in `a1` and `a3`. Likewise, the real part of the result should be returned in `a0` and the imaginary part in `a1`.

When you have written `complexMul` answer the following questions.

a) Is `complexMul` a leaf or non-leaf procedure?

b) Did you need to use the stack? Explain why.

c) What are the overflow conditions for your procedure?

Exercise A1.5 - RISC-V instruction for multiplication

Modify your program to use the RISC-V instruction `mul` instead of your `myMult` procedure.

When you have modified your program answer the following question.

a) How many clock cycles did you save?

Exercise A1.6 - Extra questions

Answer the following questions.

a) How would your program have differed if RISC-V only had 2 argument passing registers, `a0` and `a1`, and one result register `a0`?

b) Which registers are saved and not saved across a procedure call? What does this mean?

c) How would you extend your program to handle 64-bit results of multiplication?

Report front page

The report should have a front page similar the one shown below.

02155 - Computer Architecture and Engineering
Fall <year>

Assignment 1

Group members:
<student number> – <student name>
<student number> – <student name>

This report contains <n> pages

<month> <year>