Computer Organisation COMP2008

Lab Sheet 1 (Starts session week 2, due in week 3)

Student Name and Number	
Date, Grade and Tutor signature, **max mark 3**	

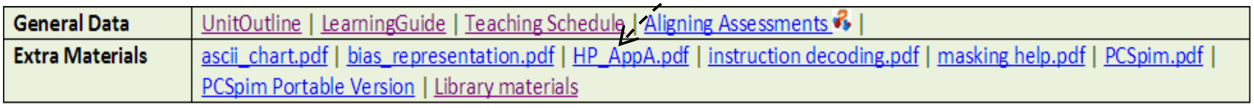
Keep this cover sheet marked and signed by the tutor.

1. Preparation [Total max. mark: 1]

The main goals of today's lab are: to understand how to write simple programs, to become familiar with PC Spim – the MIPS processor simulator, and to learn how to use its features to debug your programs. Note for 64-bit users: The first time you run PCSpim, it may complain about a missing exception handler (exceptions.s). If you see this message, open Simulator->Settings, look under "Load exception file", and change the path to the following (or something similar):

C:\Program Files (x86)\PCSpim\exceptions.s

1. Before the lab please study "**PCspim.pdf**" and section A.9 (Page 40) in "**HP_AppA.pdf**" (available on the website), as well as lecture notes and relevant sections from the textbook.



General Data	UnitOutline | LearningGuide | Teaching Schedule | Aligning Assessments |
Extra Materials	ascii_chart.pdf | bias_representation.pdf | HP_AppA.pdf | instruction decoding.pdf | masking help.pdf | PCSpim.pdf | PCSpim Portable Version | Library materials

2. The following exercise is to familiarise yourself with **PCSpim**. Note: If you use **QtSpim**, your exercise here will be based on QtSpim.

   a) Make a **hand drawing** of the PCSpim user interface [**0.5 marks**]

   Note 1: Screen captures are not accepted.

   Note 2: Your hand drawing does not need to copy all the information on the user interface; however, you must include at least 3 to 5 complete lines from each area (pane).

   Note 3: If a single drawing cannot display all the required information, you may provide multiple drawings.

   b) Explain in writing what is the role of four main horizontal areas (panes) in the PCSpim user interface [**0.5 marks**].

2. Workshop Tasks [Total max. mark: 2]

Please answer the lab Questions listed below in writing, print or neatly write your answer:

1. Create a folder for this subject (e.g. CO_COMP2008), and within it create a subfolder for Lab01. Copy two assembly language files *hello*.s and *helloimproved*.s (see Prac. 01 code on the website) to the newly created folder. **NOTE:** PCSpim assembly programs are text files, they can be edited with any text editor, for example NotePad, NotePad++, etc.

2. Start PCSpim. Open file *hello*.s in PCSpim and run it. Meanwhile open the same file in a text editor (e.g. NotePad++, for easy view) and analyse the code. You must read and fully understand all comments! (# ← comments in the code). **NOTE:** The PCSpim simulator is only for running assembly programs and doesn't support code editing; it's not a good code viewer either. For editing assembly programs, you have to use a text editor. So for the lab tasks regarding assembly programs, you'll switch between PCSpim and a text editor.

3. Open *helloimproved.s* in PCSpim and run it. Also open the same file in a text editor (for easy view) and analyse the code; compare it with the *hello.s* code.

   Run *helloimproved*.s in PCSpim again. Pick up a line of code statement from PCSpim and identify the following formats for the code statement: the original text of the instruction, the assembled text, and the machine code.
   Question (0.3): Answer the question in writing. List a couple of lines of code statements from PCSpim. Point out which are the *original text* format of the instructions, the *assembled* version, and the *machine code* version. Also, briefly explain main differences in between (addressing technical details).

4. Run the *helloimproved*.s several times, and observe its results in the console window. Identify the data segment, and the registers. Then single-step through the program, observing the changes in the registers. After the whole program has been executed, experiment with the 'clearing registers' action.
   Question (0.2): What is the effect of clearing registers? Why simulator provides this operation?

5. Still with *helloimproved*.s. Find comment '# line xx' and insert left to the comment mark '#' the following: `sub $s0, $t0` Save and load the program.
   Questions (0.2): Observe what's happening when loading or running the program. What was the effect from this code modification? Explain 'why' with more technical details.
   [Restore the program to the original form by deleting '`sub $s0, $t0`']

6. Modify the *helloimproved*.s to print on the screen two additional lines of text defined by yourself: e.g. one line with your name (or the subject title), and one line with any text you like. Note: don't change the

existing message defined in the program; insert at least two labels defined by yourself. Save your program as *myfirstprog.s* Remember to write clear, descriptive comments for your part of the code. **Demonstration (1):** demonstrate to the tutor the program you wrote. Be prepared to explain the code, and answer additional questions about the code and PCSpim.

7. **Questions (0.3):** When running *helloimproved.*s, after the output text is printed to the console, why the cursor is placed in the next (blank) line? What is the role of instruction ‘syscall’? How syscall *services* and *arguments* are selected and specified in the code?

3. Assessment

When you complete all tasks, present to the tutor a **printed copy of your program source code** with your name and student number included in the comments in your code (#…), and typed or neatly written answers to all questions listed in the lab sheet (note that amount of questions vary per lab).

Warning: Any source code duplicated amongst students will result in a zero mark, and possible further action according to the WSU policy on plagiarism (see the subject learning guide).