Computer Organisation 300096

Lab Sheet 3 (starts session week 4, due in week 5)

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 goal of today's lab is to analyse simple programs performing arithmetic operations, to learn how to debug programs, and to learn how ASCII characters are placed in memory.

1. Study the lecture notes. Use **PH_AppA.pdf** and **ascii_chart.pdf** (both on the website) as a reference.

Extra Materials	ascii_chart.pdf | bias_representation.pdf | HP_AppA.pdf |

Study at least the following sections from the Patterson and Hennessy textbook:

- Ed.2: Chapters 3.1, 3.2, relevant parts of 3.3 and 3.7 or:
- Ed.3: Chapters 2.1, 2.2, relevant parts of 2.3 and 2.8
- Ed.4: Chapters 2.1, 2.2, relevant parts of 2.3
- Ed.5: Chapters 2.1, 2.2, relevant parts of 2.3

2. Answer the following questions:

a) When the loader loading user's code into PCSpim, the system should have some way to effectively determine the 1st instruction of user's code. Think about the mechanism for PCSpim to locate the **1st instruction** of user's code based on the programs' memory image. Note that PCSpim doesn't have pre-knowledge of user's source code. It is also assumed that you understand the entry point concept from learning Programming Fundamentals.

Question: Describe how would you recognize the right address at **the first instruction** of user's code loaded into PCSpim. Illustrate to your tutor the value of the address you would enter for setting a breakpoint to stop user's program execution at **the first instruction**. What message would you see when starting the program after setting such a breakpoint? [**0.4 marks**]

b) What would you expect to see in a memory location which was loaded (or initialised) with a single ASCII character "**H**", "**a**", and "**v**" respectively? Give answer in two formats: in HEX and in binary. [**0.6 marks**]

	"**H**"	"**a**"	"**v**"
HEX			
Binary			

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

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

1. Create a subfolder for lab 3, and copy to the folder two assembly language files: *temperature.s* as well as the final, corrected *simplecalc.s* you completed in lab 2.

2. Use a text editor to modify the *simplecalc.s* program (you completed in lab 2) to perform the following calculations: (g + h) – (i – j + k), and to output (print) the result on the screen. As usually, save your modified program with a different name.

   Question (0.5): load *simplecalc.s* into PCSpim, insert a breakpoint after all initial values were set, and before the calculations start. Run the program to the breakpoint, and when it stops, set a new value (any number) in one of the registers holding your initial values. Complete the run, and check the result in the console window. Demonstrate to the tutor your program, how it runs, how you can set breakpoints, and how you can modify registers midway through your program execution. Note: in PCSpim, to change register values, use 'Simulator' -> 'Set Value' function.

3. Still with *simplecalc.s* in PCSpim. In the **data segment** of PCSpim, identify the memory locations holding all characters of the message: "have a nice day :)". There are a couple of ways to find the memory addresses. By the way, you may need to use ASCII table "ascii_chart.pdf" for this question.

   Question (0.5): Explain how ASCII characters are placed in memory, illustrate your explanation with a hand drawing. Be able to show how data is placed in memory on PCSpim/QtSpim screen. **Hints:** ASCII characters are packed in word blocks and placed in memory from lower memory address to higher memory address. The addresses are shown as hex values. The address for each word block (4 characters packed in it) is indicated or calculated, the address of each char in the word can be calculated by adding its offset.

   Question (0.5): In this exercise, you are going to change data in memory on PCSpim directly on the fly when running at a certain breakpoint (rather than changing its definition). Now insert a breakpoint right before the program prints on the screen the message: "\nHave a nice day :)". If you are not certain where to insert the breakpoint, use single step to find out. Run the program and stop it at the breakpoint; change the four characters of the message from "*Have*" to: "*-not*" (don't change other characters); complete the run; check the result in the console window. **Hints:** The PCSpim interface is read-only and doesn't support data modification through its interface directly. Note: in PCSpim, to change data at a certain memory address, use 'Simulator' -> 'Set Value' function.

4. Analyse provided code *temperature.s*

   Question (0.3): insert missing code, and fill in missing comments (lines marked #__??). The program needs to run correctly to be assessable. **Note:** this simple program does not have to check for incorrect characters, or out of range number entries.

   Question (0.2): what formula was used to convert temperature from Fahrenheit to Celsius?

3. Assessment notice

Present to the tutor a printed copy of your program source code, with your name and student number included in the comments (#…), and typed or neatly written answers to all questions listed in the lab sheet.

Warning: Any work duplicated amongst students will result in a zero mark, and possible further action according to the WSU policy on plagiarism.

Your tutor may decide to keep the source code printout, but you should always keep the cover sheet marked and signed by the tutor.