Computer Organisation COMP2008

Lab Sheet 7 (starts session week 8, [Week 9: intra-session break] due in week 10)

Student Name and Number	
Date, Grade and Tutor signature, **max mark 4**	

Keep this cover sheet marked and signed by the tutor.

1. Preparation

The main goal of today's lab is to understand the assembly language techniques for bit operations, including masking, arithmetic and logical shifts etc. Study sections of the textbook and the lecture notes explaining logical operations. See also *masking help.pdf* on the website. Get assembly language file *split.s* provided for this lab.

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

Self study task:

1. Explain the difference between logical shift, arithmetic shift and rotation.
2. Give a simple example of masking technique that can be used to extract single position bit value from a word. Illustrate your answers with hand drawings.

2. Workshop Tasks [Total max. mark: 4]

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

Task I (0.5 marks): Paper work.

- **Question 1** [0.3 mark]: Get file *split.s* provided for this lab. View the program in a text editor and experiment with it in PCSpim to understand how it works. Then answer the following question:

  When running *split.s* and enter the number "131071", the output on the console is: "The halves are: 1 and –1". Explain why in hand-drawing (Calculator in Scientific mode may help).

- **Question 2** [0.2 mark]: What type(s) of shift would you use to effectively extract a signed number stored in a group of bits somewhere in a word? (*masking help.pdf* is helpful)

Task II (1.5 marks): Write a MIPS program which performs the tasks below (make sure to follow the conventions for register usage):

a. The main program reads a number p from the keyboard, and calls the procedure ***extract*** with p passed as a parameter.

b. The procedure *extract* extracts a **5**-bit field starting at bit **3** from a 32-bit value p. (hint: Refer to Lecture 5, slides 29-30).

c. The extracted bit field is placed in a return register with the least significant bit at position 0, and returned to the main program.

d. The main program displays on the console **both** signed and unsigned values of the extracted bit field.

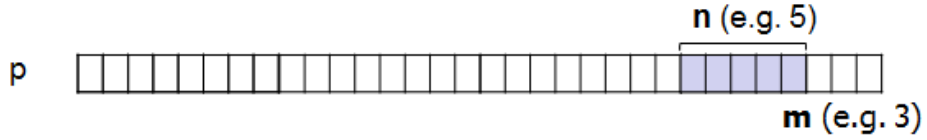
e. The main program will loop, until it is terminated by entering *0* as a value for *p*.

Task III [Advanced, Total max. mark: 2] This is an advanced task, please attempt it only if you completed, and fully understand the first task.

Write a new program which is an expanded version of the program you wrote in workshop **Task II**. Details below:

a. The main stream of the new program should allow user to enter three parameters from the keyboard: value of ***p*** (as before), and two new parameters: ***n*** and ***m***. Then the main stream calls the procedure ***extractExt*** for number extraction with **p**, **n** and **m** passed as parameters.

   Because **p** is a 32-bit value, the sum of **n** and **m** should not be greater than 32. The main program should terminate with an error message if this condition is not satisfied. Otherwise, the main program will loop for reading parameters and calling ***extractExt*** until it is terminated by entering *0* as a value for *p*.



b. The expanded procedure ***extractExt*** has to extract an **n**-bit field starting at bit **m** from a 32-bit value **p**. For this purpose, ***extractEx*** needs to build a mask programmatically each time n and m are entered from the keyboard (note that in the first workshop task values of n and m were defined as constants: n=5 and m=3, thus the mask could be coded as a constant value). Hint: Refer to 'masking help.pdf'.

c. The bit field extracted should be placed in a return register with the least significant bit at position 0, and returned to the main program.

d. The main program displays on the console **both** signed and unsigned values of the extracted bit field.

Note 1: Task II and Task III should be developed and demonstrated separately; Task III doesn't cover Task II.

Note 2: Some testing data for your convenience to test your programs:

Inputs			Outputs	
p	n	m	unsigned number	signed number
173	5	4	10	10
173	3	5	5	-3
137	4	4	8	-8

Demonstration: Demonstrate to the tutor your running programs for grading; be prepared to explain the code.

3. Assessment notice

When you ready, 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 (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 USU 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.