# Week 5 Lab MIPS Assembly language programming

## Objectives

Develop understanding and experience of:

1. Using registers and instructions (including syscalls) with the MARS simulator for MIPS.
2. Creating and calling subroutines in MIPS.

The third part of the lab session will be used to have a brief look at the coursework.

### Part 1 Getting started with MARS/MIPS – using registers and syscalls

The tutor will demonstrate how to get started with the MARS environment for MIPS Assembly language programming. You may use the code provided on Moodle to get started.

1. Create an assembly language program to do the following:
   * Store the decimal integer values 5 and 12 in two of the $t registers
   * Add the contents of those two registers and store the result in one of the $s registers

Assemble and run the program, check the contents of the three registers you have used.

1. Amend your program so that you now store another number in one of the unused $t registers. Add the contents of the three $t registers you have used and store in one of the $s registers. Note that the add instruction can only add two registers together, so you will need more than one add instruction. Test your program, compare to what you get when you add them up in your head or on a calculator. Try different initial values for the three $t registers. Make sure that you ckeck the results in the registers.
2. Create a new copy of your program and amend it so that it now adds decimal 8 to the final result, using an immediate value for the 8. Check the result.
3. Amend the program so that it now copies the final result to another of the $s registers so that the result is in two of the registers.
4. Create a new copy of your program and amend it so that it multiplies three values from $t registers together instead of adding and then and then multiplies by an immediate value (not zero).

The tutor will talk through the work. So far, the only way you can see whether the program has worked is by looking in the registers. The tutor will talk through ideas relating to syscalls, so that you can amend your programs to output the results to the console.

1. Take a copy of your program that adds three numbers and amend it to output the initial values and the result to the console. At this point you will be outputting just the numbers and no text.
2. Take a copy of your program that outputs the numbers and amend it so that it outputs a single character between each number. The single character could be a comma, a space, a tab or a newline character according to your preference.

The tutor will review the work on syscalls.

### Part 2 Using memory and subroutines in MARS/MIPS

The tutor will introduce some concepts on using memory in a MIPS Assembly language program so that you can handle text strings as well as integers and characters in your output to the console.

1. Take a copy of your program that adds three numbers and outputs the initial values and the result. Amend the program so that it outputs a suitable message before giving the initial values and another message before giving the result.

The tutor will introduce the use of subroutines in MIPS Assembly language.

Download the week 5 lab file from Moodle.

1. Open program **week5part2b.asm** in the MARS MIPS simulator. Before you run the program, read it and try to predict what it will output. As before, you will probably need to use a text editor to keep track of the predicted output. Assemble and run the program and check against your prediction.

The tutor will review the work with the group.

### Part 3 Coursework support

This part of the lab will be a little shorter than the other two and is for coursework support. The tutor will look at some aspects of the MIPS part of the coursework with the class pointing out similarities to the work done in this lab session. Don’t forget to try the reinforcement exercises as they are also designed to help with the MIPS Assembly language programming part of the coursework.