Lab 3: Exceptions and exception handling

In this project, you will run a program that causes multiple exceptions. You will step through the exception handling code to better understand how it functions. You will then modify the code so that it no longer causes exceptions. Finally, you will run the original code that causes exceptions but modify the exception handling code to 'fix' the problems encountered and allow the code to run as intended. ***You have to use the QtSPIM simulator for this lab.***

Overview

This project will cover exceptions and exception handling in SPIM. You are given a small program that causes several exceptions, along with a trap handler that is extensively documented to help you step through and understand how the exceptions are handled.

Once you are comfortable with the control flow and instruction execution that occurs for exceptions, you will modify the program so that it no longer throws exceptions. That modified program will be the first of two pieces of code that you submit.

The next step will be to use the original program (which causes the exceptions) but to modify the trap handler so that it can handle the exceptions as specified below. This modified trap handler will be the second of two pieces of code that you submit.

Download the following files from the website of the course:

* The program which causes exceptions: program_with_exceptions

* The default trap handler (with extra comments): commented_trap_handler

Part 1:

Execute the program_with_exceptions program under SPIM. The program should generate Bad Data/Stack Address Exceptions (Exception 7) because the program assumes that the data segment starts at 0 and SPIM assumes that it starts at 0x10000000. You should see the output from the trap handler specifying the exceptions in SPIM's 'Console' window.

Run the program_with_exceptions program again using spim and this time single step through at least two exceptions. Make sure you see how the flow of control is changed by looking at the source code and note where the exceptions occur.

Modify program_with_exceptions to a MIPS program program_without_exceptions by changing two instructions so that it runs correctly. (The program should form the sum

A[0] + A[1] + ... + A[8] = 0 + 1 + 2 + ... + 8 = 36 and print this number. Make no further use of the file program_without_exceptions below.)

[HINT: You will want to use the 'la' instruction.]

Once your modified program is working correctly, without throwing exceptions and generating the correct answer, you have finished the first half of this lab. This modified program will be the first of two things that you turn in and will not be used again for the following activities.

Part 2:

In the spim menu, go to 'Simulator' -> 'Settings'. This will open the 'Settings' dialog. Modify the path to the trap file so that it points to the commented_trap_handler that you downloaded earlier.

In the spim menu, go to 'Simulator' -> 'Reinitialize'. This will cause spim to reinitialize itself and load the new commented_trap_handler.

Load and run the original program_with_exceptions program again. The program should run the same way it did before, causing exceptions.

Now edit the trap handler commented_trap_handler so that it handles a bad data/stack address (Exception 7) as follows, making sure to do all three of the following:

- Print a special new message indicating that you are adjusting for exception 7.
- Add 0x10010000 to $t1.
- Restart the current instruction. (Note: Restart the current instruction, not the next instruction as the current handler does.)

When you run the program you should only get two Exception 7 occurrences at the instructions:

```
lw      $t4, 84($t1)    #Constant 4 stored in $t4

  ....
sw      $t3, 0($t1)     #sum is stored
```

Your program should handle these exceptions so that the correct calculation is performed and a 36 is printed at the end.

Debugging hints:

- You will probably need to do a lot of single-stepping to debug.
- Be sure to use addu when adding addresses so that you don't generate an overflow exception.

- Be very careful not to use any registers in your trap handler that you haven't explicitly saved in kernel static memory. It is best to just use the register $a0 for all your work.
- Whenever you modify the trap handler, be sure to save it and reinitialize spim so that the newest version is loaded.
- Note that when the exception error message is printed by the handler, register $k0 holds the exception number 7, but it is shifted 2 positions to the left, so that $k0 actually holds 28 = 0x1c = 11100 (base 2).
- You should NOT modify the original code program_with_exceptions at all. The original bad data address references should remain. All modifications occur in the edited handler.
- Details about how to use syscalls can be found in Appendix A.

Once your modified trap handler correctly handles the exceptions thrown by program_with_exceptions, you have completed the last portion of this lab.

Turning In Your Code

To turn in your project, you must submit BOTH parts of the code:

1. The modified program_without_exceptions which runs correctly with the original commented_trap_handler and causes no exceptions.

2. The modified commented_trap_handler which handles the bad data reference exceptions in program_with_exceptions as specified above.

Upload your code to Avenue by April 7th