MIPS Programming, SPIM

Tom Kelliher, CS 220

Sept. 28, 2011

1 Administrivia

Announcements

Assignment

Read Sections 2.1–2.10, and 2.12 of the SPIM S20 manual.

From Last Time

Control structures in MIPS assembly.

Outline

1. Using SPIM.
2. Lab exercise.

Coming Up

More MIPS programming.

2 Using SPIM

Things to notice:

1. Structure of a MIPS assembly language program.
2. System calls: `syscall`. Exit from your program:

```
li $v0, 10
syscall
```

3. I/O:
   (a) Reading an integer.
   (b) Writing a string or integer.
4. Debugging:
   (a) Creating global labels with `.globl`.
   (b) Setting and hitting breakpoints. Continuing from a breakpoint.
   (c) Printing register values.

3 Example: addn

3.1 addn.c

```
#include <stdio.h>

int main()
{
   char *prmpt1 = "How many inputs? ";
   char *prmpt2 = "Next input: ";
```

```
   char *result = "The sum is ";
   char *nl = "\n";

   int n;
   int sum;
   int temp;

   printf("%s", prmpt1);
   scanf("%d", &n);
   sum = 0;

   while (n > 0)
   {
      printf("%s", prmpt2);
      scanf("%d", &temp);
      sum = sum + temp;
      n = n - 1;
   }

   printf("%s", result);
   printf("%d", sum);
   printf("%s", nl);

   return 0;
}
```

3.2 addn.s

```
# addn.s
# Input: A number of inputs, n, and n integers.
# Output: The sum of the n inputs.
# Demonstrates reading and writing integers.

# Register usage:
#    $t0: how many integers remain to be read.
#    $t1: sum of the integers read so far.

            .data                          # Constants.
prmpt1:     .asciiz "How many inputs? "
prmpt2:     .asciiz "Next input: "
sum:        .asciiz "The sum is "
nl:         .asciiz "\n"
```

```
            .text                   # Main.
            .globl main

main:       li $v0, 4               # Syscall to print prompt string.
            la $a0, prmpt1
            syscall

            li $v0, 5               # Syscall to read an integer.
            syscall                 # Result returned in $v0.
            move $t0, $v0           # n stored in $t0.

            li $t1, 0               # sum stored in $t1 -- clear it.

            .globl while
while:      blez $t0, endwhile      # Read n integers.
            li $v0, 4               # Prompt for next integer
            la $a0, prmpt2
            syscall

            li $v0, 5               # Read next integer.
            syscall
            add $t1, $t1, $v0       # Increase sum by new input.

            sub $t0, $t0, 1         # Decrement n.

            b while

endwhile:   li $v0, 4               # Print result string.
            la $a0, sum
            syscall

            move $a0, $t1           # Print sum.
            li $v0, 1
            syscall

            li $v0, 4               # Print a newline character.
            la $a0, nl
            syscall

            li $v0, 10              # Syscall to exit.
            syscall
```

4 Lab Exercise

See lab handout.