CSE3666 — Lab 1

Mike Medved

January 25th, 2022

1 Prompt

Translate the following C code to RISC-V assembly code. Use a minimum number of instructions. Assume that the values of a, i, j, and r are in registers s1, s2, s3, and s4, respectively. All the variables are signed. Load the constant into register s5 before the loop. Write brief comments in your code. Clearly mark the instructions that controlling the outer loop and the inner loop (for example, using different colors). There are 12 instructions in the solutions.

```
for (i = 1; i < a; i++)
    for (j = 0; j < i; j++)
        r ^= (j + 0x55AABB33);
```

2 Deliverables

```
# CSE 3666 Homework 1, Question 6

.globl main
.text

main:
    # use system call 5 to read integers
    addi    a7, x0, 5
    ecall
    addi    s1, a0, 0 # a in s1

    # initialize i, j, r to 0, and store them into s2, s3, s4 respectively
    addi    s2, x0, 0
    addi    s3, x0, 0
    addi    s4, x0, 0

    # enter the loop routine
    j       loop_i

loop_i:
    addi    s2, s2, 1          # i++
    blt     s2, s1, loop_j     # i < a, then run the inner loop
    bge     s2, s1, exit       # loop condition met, exit

loop_j:
    addi    s3, s3, 1          # j++

    # load 20 most-significant-bits into the temporary register, t0,
    # and perform a sign-extension for the 12 remaining least
    # significant bits. with these two components known, add
    # them together and store the result in the temporary register, t1.
    #
    # now that we have the full 0x55AABB33 value stored in a register,
    # we can add t1 + s3 to obtain the segment "j + 0x55AABB33" and
    # store it in t2. finally, compute the xor of the current value of r,
    # stored in register s4 against the result that is stored within t2.
    lui     t0, 0x55AAC
    addi    t1, t0, 0xFFFFFB33
    add     t2, s3, t1
    xor     s4, s4, t2

    blt     s3, s2, loop_j     # continue the loop until j >= i

exit:
    # load service number 1 into the service register
    # and perform a system call to print the result
    # of the algorithm to the standard output.
    li, a7, 1
    add a0, s4, x0
    ecall

    # load service number 10 into the service register
    # and perform a system call to exit the program
    # with exit code 0.
    addi a7, x0, 10
    ecall
```