Program Assignment 0

Sep 19, 2018

By Jing-Jia Liou

Program Assignment 0 – Tracing Registers' Usage

Due: 23:59 Oct. 2, 2018

Introduction

- In this homework, you will write a C/C++ code to analyze register usage of an assembly code. There are three parts of this assignment: (1) Generate an assembly code from a provided C/C++ code (2) Write C/C++ code for analysis (3) Provide an extra test case of assembly code.

Penalty of plagiarism

- Each time you submit a plagiarized code, your grade of the homework will be discounted to 90%. The check is done in batch every hour. The check is based on code similarity of 30% threshold (30% of your code is identical structurally to another code).
- The discount will be **accumulated each time** you submit a plagiarized code. Please do not use trial-and-error to adjust your codes.
- If your final version (last submission before due date) is a plagiarized code, no credit will be given.

Fill Student Contact Form

Please fill this form first.

Part I : Generate assembly code via compiler

We assume you are logged in at a Workstation (EE3450B, EE3450C, EE3450D, etc.) or use a provided Docker image. (The detailed tutorial to setup an offline RISC-V testing environment is here.)

1. Please create a directory in your home and cd to the directory (Assume pa0/).
2. Please download the example code and store it as example.c.
3. Use the following command to generate assembly code from the example C code. (Note that the output filename will be **example.s**)

```
4. $ riscv64-unknown-elf-gcc -S example.c
```

5. Use vim or any text editor to view the generated assembly code.

Part II : Tracing the usage of registers (80%)

- Please count the occurence of 32 RISC-V architecture registers with a C/C++ code. There is no need to identify read/write operations.
- The names of 32 RISC-V registers are listed in the following table:

Name	Description
zero	Hard-wired zero
ra	Return address
sp	Stack pointer
gp	Global pointer
tp	Thread pointer
t0~2	Temporaries
s0/fp	Saved registers/Frame pointer
s1	Saved registers
a0~1	Function arguments/return values
a2~7	Function arguments
s2~11	Saved registers
t3~6	Temporaries

- Assume your code is named as “count.cpp” and you have compiled the code into an executable with the following command(Note that here we use clang instead of g++ to show warning message):

```
$ clang++ -Wall -o count count.cpp
```

- The program execution and outputs are shown below. Note that your output format **must** be exactly the same as the example shows. Otherwise your homework will be rejected.

```
$ ./count < example.s

REG: zero, count= 1

REG: ra, count= 4

REG: sp, count= 16

REG: gp, count= 0

REG: tp, count= 0

REG: fp, count= 0

REG: t0, count= 0

REG: t1, count= 0
```

```
REG: t2, count= 0
REG: t3, count= 0
REG: t4, count= 0
REG: t5, count= 0
REG: t6, count= 0
REG: a0, count= 7
REG: a1, count= 2
REG: a2, count= 0
REG: a3, count= 0
REG: a4, count= 6
REG: a5, count= 45
REG: a6, count= 0
REG: a7, count= 0
REG: s0, count= 28
REG: s1, count= 0
REG: s2, count= 0
REG: s3, count= 0
REG: s4, count= 0
REG: s5, count= 0
REG: s6, count= 0
REG: s7, count= 0
REG: s8, count= 0
REG: s9, count= 0
REG: s10, count= 0
REG: s11, count= 0
$
```

If a register is not used, please still report occurrence to be 0.

This example code shows how to report registers' usage.

```
for(auto reg : regs) {
    cout <<"REG : "<< reg.getname() <<" ,count= "<< reg.getcount() << endl;
}
```

Part III : Create and submit your own test case (20%)

- Please use another C/C++ code to generate another assembly code to use as a test case. (test.c)
- Your test case **must not** have any additional arguments or user prompt(s).
- Check if the code is executable. If not, no credit will be given in this part. (the output is arbitrary in this case as long as the program can be executed).

```
$ riscv64-unknown-elf-gcc -o test  test.c
$ spike pk test
13
$
```

- Before submission, make sure the total number of register reference counts are between 200 and 1000 (to limit the analysis time).

Check output format of your analysis code

- Please use the following command to check if your output format is correct(assume you have logged in ee3450B/C/D):

```
$ ./count < example.s > output.txt
$ ~ee345000/pa0/bin/check < output.txt
Your format is correct.
$
```

- Access test cases from other students
  - Please check the test cases under ~ee345000/pa0/testcase in workstations (EE3450B, EE3450C, EE3450D, etc.).

Submission

- Please rename your C++ code and testcase as hw0_ID.cpp and hw0_ID.s, respectively. For example, if student ID is **103061232**, the names will be hw0_103061232.cpp and hw0_103061232.s.

- Submit your homework via the link.