Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Compiler Laboratory: CS39003
3rd year CSE, 5th Semester

Assignment - 1: Annotating Assembly *Marks: 50*
Assign Date: July 26, 2017 Submit Date: 23:55, August 01, 2017

1. Consider the following assembly language program of x86-64 (Intel 64-bit processor), generated from a C program using

```
cc -Wall -S test.c
```

Assembly Program: *test.s*

```
    .file "test.c"
   .text
   .globl func
   .type func, @function
func:
.LFB0:
   .cfi_startproc
   pushq %rbp
   .cfi_def_cfa_offset 16
   .cfi_offset 6, -16
   movq %rsp, %rbp
   .cfi_def_cfa_register 6
   pushq %rbx
   subq $56, %rsp
   .cfi_offset 3, -24
   movl %edi, -52(%rbp)
   movq %fs:40, %rax
   movq %rax, -24(%rbp)
   xorl %eax, %eax
   movq %rsp, %rax
   movq %rax, %rsi
   movl -52(%rbp), %eax
   addl $1, %eax
   movslq %eax, %rdx
   subq $1, %rdx
   movq %rdx, -40(%rbp)
   movslq %eax, %rdx
```

```
   movq %rdx, %r8
   movl $0, %r9d
   movslq %eax, %rdx
   movq %rdx, %rcx
   movl $0, %ebx
   cltq
   salq $2, %rax
   leaq 3(%rax), %rdx
   movl $16, %eax
   subq $1, %rax
   addq %rdx, %rax
   movl $16, %edi
   movl $0, %edx
   divq %rdi
   imulq $16, %rax, %rax
   subq %rax, %rsp
   movq %rsp, %rax
   addq $3, %rax
   shrq $2, %rax
   salq $2, %rax
   movq %rax, -32(%rbp)
   movq -32(%rbp), %rax
   movl $0, (%rax)
   movq -32(%rbp), %rax
   movl $1, 4(%rax)
   movl $2, -44(%rbp)
   jmp .L2
.L3:
   movl -44(%rbp), %eax
   leal -1(%rax), %edx
   movq -32(%rbp), %rax
   movslq %edx, %rdx
   movl (%rax,%rdx,4), %ecx
   movl -44(%rbp), %eax
   leal -2(%rax), %edx
   movq -32(%rbp), %rax
   movslq %edx, %rdx
   movl (%rax,%rdx,4), %eax
   addl %eax, %ecx
   movq -32(%rbp), %rax
   movl -44(%rbp), %edx
   movslq %edx, %rdx
   movl %ecx, (%rax,%rdx,4)
   addl $1, -44(%rbp)
.L2:
   movl -44(%rbp), %eax
   cmpl -52(%rbp), %eax
   jle .L3
```

```
   movq -32(%rbp), %rax
   movl -52(%rbp), %edx
   movslq %edx, %rdx
   movl (%rax,%rdx,4), %eax
   movq %rsi, %rsp
   movq -24(%rbp), %rbx
   xorq %fs:40, %rbx
   je .L5
   call __stack_chk_fail@PLT
.L5:
   movq -8(%rbp), %rbx
   leave
   .cfi_def_cfa 7, 8
   ret
   .cfi_endproc
.LFE0:
   .size func, .-func
   .section .rodata
.LC0:
   .string "%d"
   .text
   .globl main
   .type main, @function
main:
.LFB1:
   .cfi_startproc
   pushq %rbp
   .cfi_def_cfa_offset 16
   .cfi_offset 6, -16
   movq %rsp, %rbp
   .cfi_def_cfa_register 6
   subq $16, %rsp
   movl $9, -4(%rbp)
   movl -4(%rbp), %eax
   movl %eax, %edi
   call func
   movl %eax, %esi
   leaq .LC0(%rip), %rdi
   movl $0, %eax
   call printf@PLT
   call getchar@PLT
   movl $0, %eax
   leave
   .cfi_def_cfa 7, 8
   ret
   .cfi_endproc
.LFE1:
   .size main, .-main
```

```
.ident "GCC: (GNU) 7.1.1 20170630"
.section .note.GNU-stack,"",@progbits
```

2. Copy the assembly language program with the filename as `ass1_roll.s`. Write comments in the assembly language code, explaining each assembly language instruction. Comments for each of these instructions should clearly show the connection between the original C program and the assembly language program, instead of merely describing the instruction's functioning. Please make sure that your commented file `ass1_roll.s` can be compiled to generate executable file.

   Marking scheme: *Partial marking will be there. Comments without connection to C program will get maximum 10. Addtional 30 marks is for the clear indication with the original C program.* **Marks: 40**

3. At the end of your `ass1_roll.s` file, add comments to fully explain what the overall program does. Upload your file (`ass1_roll.s`) in Moodle server. **Marks: 10**