User Id: weizhe.goh@digipen.edu

Started: 2019.11.14 11:06:52

Score:

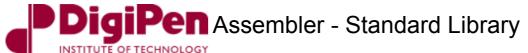
DigiPen

CS

Vadim Surov

CS100

Presentation #17



This presentation ia about using standard library in assembler code

1 Intro

- Our online compiler is GCC. GCC compiler links the C library by default.
- This means that we can call printf without changing the build process at all.
- We will see more examples calling useful standard functions in this presentation. But first let's define a macro for output instructions.

2 macro

- Assembly directives .macro and .endm allow us to define macros that generate assembly code.
- For example, following definition specifies a macro PRINTSTR that output a string using printf call.

```
Run
.macro PRINTSTR fmt
          \fmt, %edi
    mov
          %eax, %eax # Clear AL
    xor
    call
            printf
.endm
    .data
text: .asciz "Hello World!"
    .text
    .global main
main:
    push %rbx # For alignment
    PRINTSTR $text
          %eax, %eax # return 0;
    xor
    pop
          %rbx
    ret
```

Hello World!

3 time

• Current time is used to seed random number. Let's try to call time function to get the time.

```
Run
.macro PRINT fmt, v
    mov
          \fmt, %edi
    mov
          \v, %esi
    xor
          %eax, %eax # Clear AL
    call
            printf
.endm
    .data
fmt: .asciz "%d"
    .text
    .global main
main:
    push %rbx # For alignment
    mov
           $0, %rdi
    call
            time
    PRINT $fmt, %eax
          %eax, %eax # return 0;
    xor
          %rbx
    pop
    ret
1574310433
```

• Result is the number of seconds since 00:00 hours, Jan 1, 1970.

4 div

- To get a random number in a certain range we need to find reminder of a division of an arbitrary random number. In c we would use modulo operator %.
- Next code uses div instruction that takes %rdx:%rax and divide it by operand %rbx. Quotient stored in %rax. Remainder stored in %rdx register.

```
Run
.macro PRINT fmt, v
          \fmt, %edi
    mov
          \v, %esi
   mov
          %eax, %eax # Clear AL
   xor
            printf
    call
.endm
    .data
fmt: .asciz "%d\n"
    .text
    .global main
main:
    push %rbx # For alignment
   mov $0, %rdx
    mov $103, %rax
    mov $10, %rbx
    divq %rbx
    PRINT $fmt, %edx
          %eax, %eax # return 0;
    xor
          %rbx
    pop
    ret
```

• What is output if we call PRINT twice to print values in %edx and %eax? Why the second output is wrong?

5 rand

• Finally, the random number is generated here.

```
Run
.macro PRINT fmt, v
          \fmt, %edi
   mov
          \v, %esi
    mov
    xor
          %eax, %eax # Clear AL
    call
            printf
. {\it endm}
    .data
fmt: .asciz "%d"
    .text
    .global main
main:
    push %rbx # For alignment
    mov
          $0, %rdi
    call
          time
          %rax, %rdi
    mov
    call srand
    call
          rand
          $0, %rdx
   mov
          $100, %rbx
    mov
    divg %rbx
    PRINT $fmt, %edx
          %eax, %eax # return 0;
    xor
          %rbx
    pop
    ret
```

79

- To practice, modify code above to output a random value from range [10,30].
- Using loop instuction generate a sequence of 10 random numbers.

6 strcpy

 Next example demonstrates how to copy string using strcpy.

```
Run
.macro PRINTSTR str
    mov
          \str, %rdi
          %eax, %eax # Clear AL
    xor
    call
            printf
.endm
    .data
src: .asciz "Hello World!"
dst: .skip 20,0
    .text
    .global main
main:
    push %rbx # For alignment
          $dst, %rdi
    mov
    mov
          $src, %rsi
    call
          strcpy
    PRINTSTR $dst
          %eax, %eax # return 0;
    xor
          %rbx
    pop
    ret
Hello World!
```

7 todo sprintf

• Make an example using sprintf function.

```
Run
.macro PRINTSTR str
          \str, %rdi
   mov
          %eax, %eax # Clear AL
    xor
    call
            printf
.endm
    .data
str: .skip 20,0
    .text
    .global main
main:
    push %rbx # For alignment
    call sprintf
    PRINTSTR $str
          %eax, %eax # return 0;
          %rbx
    pop
    ret
Command terminated by signal 11
```

100 References

Manual – The GNU Assembler manual

stdlib C Language Library

By signing this document you fully agree that all information provided therein is complete and true in all respects.

Responder sign:

Copyright © 2019 | Powered by MyTA | www.mytaonline.com