作者：代兴　邮箱：503268771@qq.com　Vx 公众号：东方架构师

指令说明

inc 指令表示自增。
inc 指令分为 8 位 incb、16 位 incw、32 位 incl、64 位 incq。
inc 指令可以操作寄存器、内存。

语法格式 `inc ee` 表示 `ee = ee + 1` 。

用汇编代码分析

编写代码： inc.s

```
.data

int64 : .quad 7000
int32 : .long 6000

str64 : .string " int64 =  %lld \n"
str64X : .string " int64 =  %#llX \n"
str32 : .string " int32 =  %d \n"

.text
.global main

main :
    pushq %rbp
    movq %rsp, %rbp

    # 64 位，操作寄存器
    movq $8000, %rbx
    incq %rbx           # 64 位。
    movq $str64, %rdi
    movq %rbx, %rsi
    callq printf

    # 32 位，操作寄存器
    movq $0x2222222255555555, %rbx # 64 位。占位
    incl %ebx           # 32 位。触发高位清 0
    movq $str64X, %rdi
    movq %rbx, %rsi
    callq printf

    # 16 位，操作寄存器
    movq $0x22222222FFFFFFFF, %rbx # 64 位。占位
```

```
    incw %bx            # 16 位
    movq $str64X, %rdi
    movq %rbx, %rsi
    callq printf

    # 8 位，操作寄存器
    movq $0x22222222FFFFFFFF, %rbx # 64 位。占位
    incb %bl            # 8 位
    movq $str64X, %rdi
    movq %rbx, %rsi
    callq printf

    # 64 位，操作内存
    incq int64(%rip)   # 64 位
    movq $str64, %rdi
    movq int64(%rip), %rsi
    callq printf

    # 32 位，操作内存
    incl int32(%rip)   # 32 位
    movq $str32, %rdi
    movl int32(%rip), %esi
    callq printf

    popq %rbp
    retq
```

编译代码：

```
gcc inc.s -o inc
```

运行代码：

```
[root@local self]# ./inc
 int64 =  8001
 int64 =  0X55555556
 int64 =  0X22222222FFFF0000
 int64 =  0X22222222FFFFFF00
 int64 =  7001
 int32 =  6001
```

分析结果：

汇编代码	结果和分析

# 64 位，操作寄存器 	
movq $8000, %rbx
incq %rbx # 64 位。
movq $str64, %rdi
movq %rbx, %rsi
callq printf
	int64 = 8001
	
	把 64 位 8000 写到 rbx。
	64 位 rbx 自增。
	结果为 8001。

# 32 位，操作寄存器	int64 = 0X55555556

movq $0x2222222255555555, %rbx # 64 位。占位
incl %ebx # 32 位。触发高位清 0
movq $str64X, %rdi
movq %rbx, %rsi
callq printf
	把 64 位 0x2222222255555555 写到 rbx。
	32 位 ebx 自增。
	触发高位清零，结果为 0X55555556。

# 16 位，操作寄存器 	
movq $0x22222222FFFFFFFF, %rbx # 64 位。占位
incw %bx # 16 位
movq $str64X, %rdi
movq %rbx, %rsi
callq printf
	int64 = 0X22222222FFFF0000
	
	把 64 位 0x22222222FFFFFFFF 写到 rbx。
	16 位 bx 自增。
	结果为 0X22222222FFFF0000。
	低 16 位变化，从 FFFF 变为 0000。
	其他位不变，都为 22222222FFFF。

# 8 位，操作寄存器 	
movq $0x22222222FFFFFFFF, %rbx # 64 位。占位
incb %bl # 8 位
movq $str64X, %rdi
movq %rbx, %rsi
callq printf
	int64 = 0X22222222FFFFFF00
	
	把 64 位 0x22222222FFFFFFFF 写到 rbx。
	8 位 bl 自增。
	结果为 0X22222222FFFFFF00。
	低 8 位变化，从 FF 变为 00。
	其他位不变，都为 22222222FFFFFF。

# 64 位，操作内存 	
incq int64(%rip) # 64 位
movq $str64, %rdi
movq int64(%rip), %rsi
callq printf
	int64 = 7001
	
	64 位 int64 的初始值为 7000。
	64 位 int64 自增。
	结果为 7001。

# 32 位，操作内存 	
incl int32(%rip) # 32 位
movq $str32, %rdi
movl int32(%rip), %esi
callq printf
	int32 = 6001
	
	32 位 int32 的初始值为 6000。
	32 位 int32 自增。
	结果为 6001。