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

指令说明

32 位 cltd 指令，把 eax 符号扩展为 edx:eax。
如果 eax 的最高位是 1，则把 edx 的全部位都设为 1。比如，eax=0X81111111，执行后，edx=0XFFFFFFFF。
如果 eax 的最高位是 0，则把 edx 的全部位都设为 0。比如，eax=0X31111111，执行后，edx=0。

64 位 cqto 指令，把 rax 符号扩展为 rdx:rax 。
如果 rax 的最高位是 1，则把 rdx 的全部位都设为 1。比如，rax=0X8111111111111111，执行后，rdx=0XFFFFFFFFFFFFFFFF。
如果 rax 的最高位是 0，则把 rdx 的全部位都设为 0。比如，rax=0X3111111111111111，执行后，rdx=0。

整数有符号除法，需要使用 cltd 指令、cqto 指令，把被除数的位数扩大。
比如，32 位有符号除法。把被除数写到 eax，执行 cltd，符号扩展 eax > edx:eax，后续执行除法指令 idivl。

```
    # 32 位。操作寄存器。
    movl $9, %eax       # 被除数，eax
    cltd                # 符号扩展 eax > edx:eax
    movl $-2, %r8d      # 除数，r8d
    idivl %r8d          # eax / r8d = eax, edx
```

用汇编代码分析

编写代码： cx_cvt.s

```
.data

str_32:
    .string " 32bit   edx = %#11X   eax = %#11X  \n"

str_64:
    .string " 64bit   rdx = %#1911X   rax = %#1911X  \n"

.text
.global main

main :
    pushq %rbp
    movq %rsp , %rbp

    # cltd 32 位。最高位为 1
    movl $0x81111111, %eax  # 32 位。最高位为 1
    cltd            # 符号扩展。eax > edx:eax
    movq $str_32, %rdi
    movl %edx, %esi
```

```
    movl %eax, %edx
    callq printf

    # cltd 32 位。最高位为 0
    movl $0x31111111, %eax  # 32 位。最高位为 0
    cltd            # 符号扩展。eax > edx:eax
    movq $str_32, %rdi
    movl %edx, %esi
    movl %eax, %edx
    callq printf

    # cqto 64 位。最高位为 1
    movq $0x8111111111111111, %rax  # 64 位。最高位为 1
    cqto                    # 符号扩展。rax > rdx:rax
    movq $str_64, %rdi
    movq %rdx, %rsi
    movq %rax, %rdx
    callq printf

    # cqto 64 位。最高位为 0
    movq $0x3111111111111111, %rax  # 64 位。最高位为 0
    cqto                    # 符号扩展。rax > rdx:rax
    movq $str_64, %rdi
    movq %rdx, %rsi
    movq %rax, %rdx
    callq printf

    popq %rbp
    retq
```

编译代码：

```
gcc cx_cvt.s -o cx_cvt
```

运行代码：

```
[root@local cvt]# ./cx_cvt
 32bit   edx =  0XFFFFFFFF   eax =  0X81111111
 32bit   edx =           0   eax =  0X31111111
 64bit   rdx =  0XFFFFFFFFFFFFFFFF   rax =  0X8111111111111111
 64bit   rdx =                   0   rax =  0X3111111111111111
```

分析结果：

汇编代码	结果和分析
# cltd 32 位。最高位为 1 movl $0x81111111, %eax # 32 位。最高位为 1 cltd # 符号扩展。eax > edx:eax	32bit edx = 0XFFFFFFFF eax = 0X81111111 cltd 指令，把 eax 符号扩展为 edx:eax。 eax 的最高位是 1。把 edx 的全部位都设为 1。 edx 的值为 0XFFFFFFFF。
# cltd 32 位。最高位为 0 movl $0x31111111, %eax # 32 位。最高位为 0	32bit edx = 0 eax = 0X31111111

cltd # 符号扩展。eax > edx:eax	cltd 指令，把 eax 符号扩展为 edx:eax。 eax 的最高位是 0。把 edx 的全部位都设为 0。 edx 的值为 0。
# cqto 64 位。最高位为 1 movq $0x8111111111111111, %rax # 64 位。最高位为 1 cqto # 符号扩展。rax > rdx:rax	64bit rdx = 0XFFFFFFFFFFFFFFFF rax = 0X8111111111111111 cqto 指令，把 rax 符号扩展为 rdx:rax 。 rax 的最高位是 1。把 rdx 的全部位都设为 1。 rdx 的值为 0XFFFFFFFFFFFFFFFF。
# cqto 64 位。最高位为 0 movq $0x3111111111111111, %rax # 64 位。最高位为 0 cqto # 符号扩展。rax > rdx:rax	64bit rdx = 0 rax = 0X3111111111111111 cqto 指令，把 rax 符号扩展为 rdx:rax 。 rax 的最高位是 0。把 rdx 的全部位都设为 0。 rdx 的值为 0。