由

x=x\_h\*2^64+x\_l, y=y\_h\*2^64+y\_l,

可得x\*y = (x\_h\*2^64+x\_l) \* (y\_h\*2^64+y\_l)

= x\*y\*2^128 + 2^64\*(x\_h\*y\_l+x\_l\*y\_h) + x\_l\*y\_l

由提示设128位乘积结果p=x\*y ，p可以表示为p=p\_h\*2^64 + p\_l;

则可建立等式 x\_h\*y\_h\*2^128 + 2^64\*(x\_h\*y\_l+x\_l\*y\_h) + x\_l\*y\_l=p\_h\*2^64 + p\_l

两边同时除以2^64得

p\_h + p\_l/2^64 = x\_h\*y\_h\*2^64 + (x\_h\*y\_l+x\_l\*y\_h) + x\_l\*y\_l/2^64

其中x\_h\*y\_h\*2^64位数超过上限64舍去，p\_l/2^64位数为0舍去，等式化简为

p\_h = ( x\_h\*y\_l + x\_l\*y\_h ) + x\_l\*y\_l/2^64

其中 x\_l\*y\_l/2^64 为x\_l和y\_l的乘积的结果的高64位

则对高64位我们只需计算x\_h\*y\_l 和 x\_l\*y\_h，以及x\_l\*y\_l/2^64

低64即为x\_l\*y\_l%2^64

dest in %rdi , x in %rsi , y in %rdx

stroe\_prod:

movq %rdx, %rax //将%rdx(y)的值赋值给%rax

cqto //将%rax(y)扩展到128位，高位在%rdx(y\_h)中，低位在%rax(y\_l)中

movq %rsi, %rcx //将%rsi(x)的值赋值给%rcx

sarq $63, %rcx //获得%rcx的高64位(x\_h)，x\_l在%rsi中

imulq %rax, %rcx //获得y\_l\*x\_h，结果存储在%rcx中

imulq %rsi, %rdx //获得y\_h\*x\_l，结果存储在%rdx中

addq %rdx, %rcx //获得y\_h\*x\_l+y\_l\*x\_h，结果存储在%rcx中

mulq %rsi //%rax与%rsi相乘获得x\_l\*y\_l，高位在%rcx中，低位在%rax中

addq %rcx, %rdx //将x\_l\*y\_l与y\_h\*x\_l+y\_l\*x\_h相加，得到和存储在rdx中

movq %rax,(%rdx) //将x\*y低64位传递给%rdx

movq %rdx,8(%rdi)//将x\*y高64位(x\_l\*y\_l与y\_h\*x\_l+y\_l\*x\_h)传递给%rdx+8

ret