section .data

multiplicand dd 0x00000006 ; Example multiplicand (6)

multiplier dd 0xFFFFFFFC ; Example multiplier (-4)

product dd 0 ; Product (result)

section .bss

count resd 1 ; Counter for iterations

section .text

global \_start

\_start:

; Load values into registers

mov eax, [multiplicand] ; Load multiplicand into EAX

mov ebx, [multiplier] ; Load multiplier into EBX

mov ecx, 32 ; Initialize counter for 32 iterations

mov edx, 0 ; Clear EDX for product (upper part)

booth\_loop:

; Check last bit of multiplier and Q-1 (EBX and EDX)

bt ebx, 0 ; Check bit 0 of multiplier (Q0)

jc .subtract ; If Q0 is 1 and Q-1 (EDX bit 0) is 0, do subtraction

bt edx, 0 ; Check bit 0 of product (Q-1)

jnc .no\_operation ; If Q0 is 0 and Q-1 is 1, do nothing

jmp .add ; If Q0 is 0 and Q-1 is 0, do addition

.add:

add edx, eax ; Add multiplicand to product

jmp .shift

.subtract:

sub edx, eax ; Subtract multiplicand from product

.shift:

; Perform arithmetic right shift on (EDX:EBX) pair

shrd ebx, edx, 1 ; Shift EDX:EBX right by 1 bit

sar edx, 1 ; Sign-extend right shift on EDX

loop booth\_loop ; Decrement ECX and loop

.no\_operation:

; Store result in the product variable

mov [product], ebx ; Store lower 32 bits in product

; Exit the program (for demonstration purposes, using an infinite loop)

jmp $