

SECTION .data

SECTION .bss

SECTION .text

global \_start

\_start:

nop ; This no-op keeps gdb happy...

mov eax,4 ; Specify sys\_write call

mov ebx,1 ; Specify File Descriptor 1: Standard Output

mov al,01101111b ;copying 01101111b to the al registers

and al,00101101b ;sets al to be 01101111b

mov al,6Dh ; copying 6Dh to the al register

and al,4Ah ; ands 4Ah to 6Dh which should give B7h

mov al,00001111b ; copying 00001111b to the al register

or al,61h ; or-ing 00001111b with 61h, which should give 70h/01110000h

mov al,94h ; copying 94h to the al register

xor al,37h xor-ing 94h with 37h, which should give A3h

mov al,7Ah ; copying 7Ah to the al register

sub al,67h ; subtracts 7Ah with 67h, which should give 13h

mov al,3Dh ; copying 3Dh to the al register

add al,34h ; adding 3Dh with 34h, which should give 71h

mov al,9Bh ; copying 9B to the al register

not al ; reverses the bits in al register

mov al,37h ; copying 37h to the al register

neg al ; makes the value in al to negative with 2's complement, which should give 9h

int 80H ; Make kernel call

mov eax,1 ; Code for Exit Syscall

mov ebx,0 ; Return a code of zero

int 80H ; Make kernel call

This is the code I ran, I have comented behind what each line is supposed to do / what I think it does. The result was that it exited normally. I ran all of them at once because when I ran them individually some of them started to hang.