OS Assignment 5 Part 1

Aryan GD Singh
2019459

System Bootup and Switching to Protected Mode

Setup

We start in Real mode, in Real mode we can only access 1MB of memory, so to get access to the rest of the memory we need to enable the A20 bit using a system interrupt.

```
mov ax, 0x2401
int 0x15 ; enable A20 bit
```

Next we need to set up the Global Descriptor Table, which defines a 32 bit segment, and then load it using the lgdt instruction.

```
lgdt [gdt_pointer]
```

We also set the 1st bit of the cr0 register to 1 to indicate the system is in protected mode.

```
mov eax, cr0
or eax, 0x1 ; setting 1st bit of cr0
mov cr0, eax
```

Printing Hello World!

Now NASM is set to 32 bit, we set ax register to point to the Data segment and load the string to be printed into the esi register.

```
mov ax, DATA_SEG
mov esi, hello
```

We write our string to the VGS text buffer, which is at the address b8000(hexadecimal)

```
mov ebx, 0xb8000
```

We print “Hello World!” in loop1 by iterating over the memory and printing the characters one-by-one. Then in loop2 we print the contents of the cr0 register in binary format.

As can be seen, the 1st bit of cr0 register(last bit printed) is 1 because we are working in protected mode.

Compiling

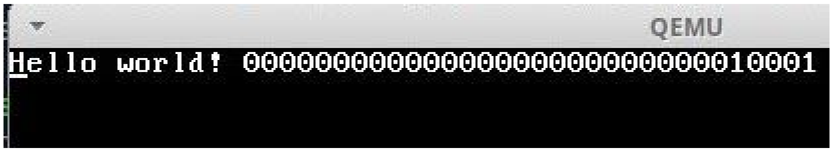
We compile our code using nasm with the command :-

```
nasm booter_2019459.asm -o booter.o
```

We then boot this up using qemu with the command :-

```
Qemu-system-x86_64 -fda booter.o
```

As we can see, Hello World!, and then the contents of cr0 register are printed :-



```
Hello world! 00000000000000000000000000010001
```
