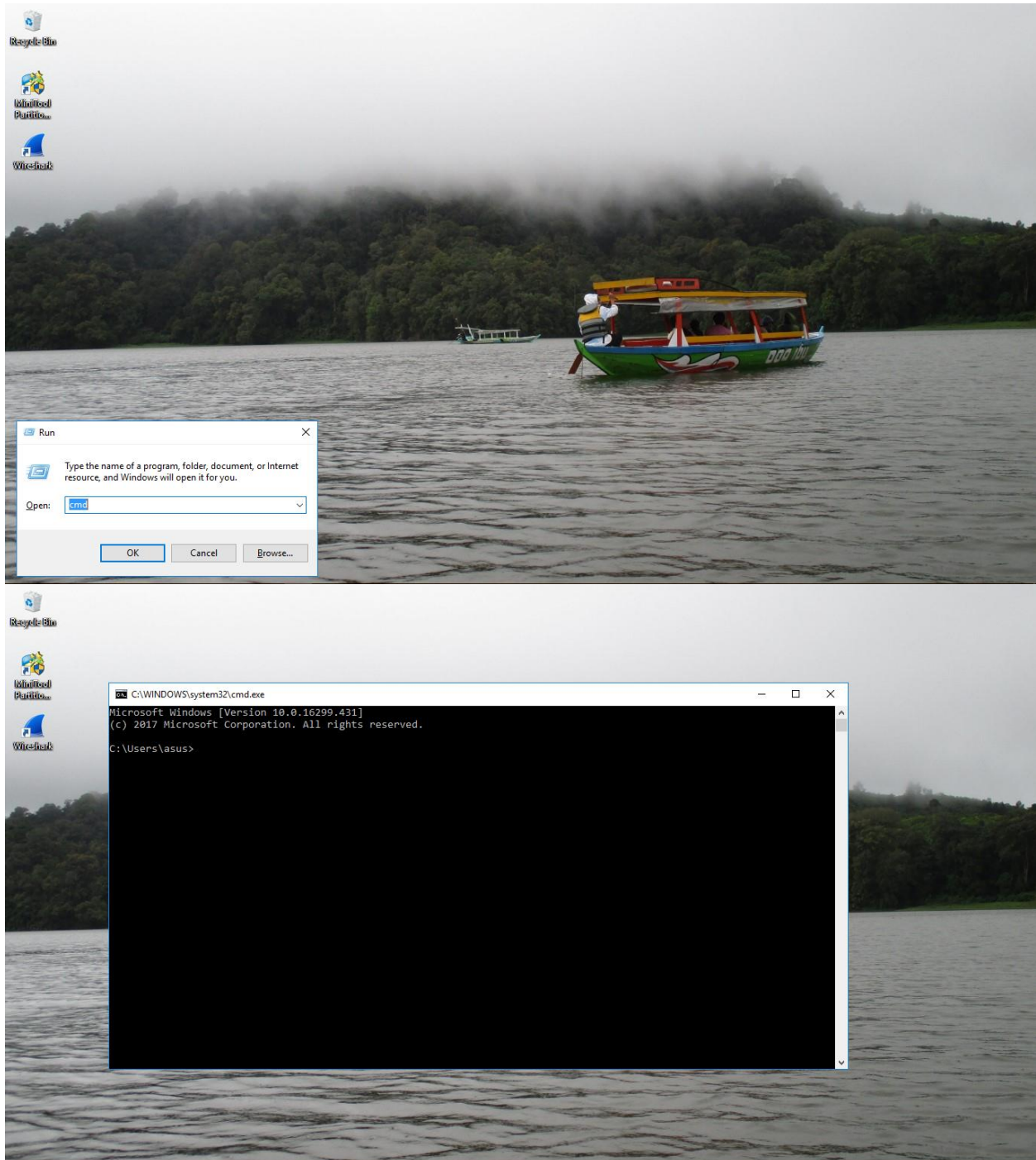


Nama : Muhammad Vicky Al Hasri
NIM : L200170065

Modul 3

Pengenalan Cara “Debugging” Program Bootstrap-loader

1. Buka Commad Prompt dengan cara Windows+R, lalu masuk ke CMD



- Menyiapkan file. Buka *cmd* lalu pindah ke direktori *C:/OS>* lalu ketik *setpath*, pindah ke direktori *LAB/LAB3* ketikkan *S* untuk memulai debugging.

```

Microsoft Windows [Version 10.0.16299.431]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\asus>cd ../../..

C:\>cd OS

C:\OS>setpath

C:\OS>Path=C:\OS\Dev-Cpp\bin;C:\OS\Bochs-2.3.5;c:\OS\Perl;C:\Windows;C:\Windows\System32
C:\OS>cd LAB\LAB3

C:\OS\LAB\LAB3>s

C:\OS\LAB\LAB3>..\..\bochs-2.3.5\bochsrc -q -f bochsrc.bxrc
000000000000[APIC?] local apic in Initializing
=====
Bochs x86 Emulator 2.3.5
Build from CVS snapshot, on September 16, 2007
=====
000000000000[ ] reading configuration from bochsrc.bxrc
000000000000[ ] installing win32 module as the Bochs GUI
000000000000[ ] using log file bochs.log
Next at t=0
(0) [0xfffffff0] f000:ffff (unk. ctxt): jmp far f000:e05b ; ea5be00f0
<bochs:1>

```

- Melihat isi register CS dan IP dengan perintah *r* dan masukkan sinyal berhenti “break point” dengan ketikkan “*vb:0x7C00*”. ketikan *c* <ENTER> untuk meneruskan prosesnya dan nantinya PC Simulator akan menampilkan tulisan Vbooting from Floppy.....

```

r esp: 0x00000000:00000000 rbp: 0x00000000:00000000
r esi: 0x00000000:00000000 rdi: 0x00000000:00000000
r ebx: 0x00000000:00000000 r9: 0x00000000:00000000
r10: 0x00000000:00000000 r11: 0x00000000:00000000
r12: 0x00000000:00000000 r13: 0x00000000:00000000
r14: 0x00000000:00000000 r15: 0x00000000:00000000
rip: 0x00000000:0000ffff
eflags 0x00000002
IOPL=0 id vip vif ac vm rf nt of df if tf sf zf af pf cf
<bochs:2> s
Next at t=1
(0) [0x000fe05b] f000:e05b (unk. ctxt): xor ax, ax ; 31c0
<bochs:3> r
rax: 0x00000000:00000000 rcx: 0x00000000:00000000
rdx: 0x00000000:00000000 rdx: 0x00000000:00000000
r esp: 0x00000000:00000000 rbp: 0x00000000:00000000
r esi: 0x00000000:00000000 rdi: 0x00000000:00000000
r ebx: 0x00000000:00000000 r9: 0x00000000:00000000
r10: 0x00000000:00000000 r11: 0x00000000:00000000
r12: 0x00000000:00000000 r13: 0x00000000:00000000
r14: 0x00000000:00000000 r15: 0x00000000:00000000
rip: 0x00000000:0000e05b
eflags 0x00000002
IOPL=0 id vip vif ac vm rf nt of df if tf sf zf af pf cf
<bochs:4> vb 0x7C00
<bochs:5> c
(46564928) Breakpoint 10285624, in 0000:7c00 (0x00007c00)
Next at t=2082128
(0) [0x00007c00] 0000:7c00 (unk. ctxt): jmp .+0x003b (0x00007c3e) ; e93b00
<bochs:6>

```

- Bandingkan 10 intruksi yang nantinya akan dieksekusi oleh PC dengan program yang terdapat dalam “boot.asm” dengan perintah *s* . maka akan terlihat hasil

```

r10: 0x00000000:00000000 r11: 0x00000000:00000000
r12: 0x00000000:00000000 r13: 0x00000000:00000000
r14: 0x00000000:00000000 r15: 0x00000000:00000000
rip: 0x00000000:0000e05b
eflags 0x00000002
IOPL=0 id vip vif ac vm rf nt of df if tf sf zf af pf cf
<bochs:4> vb 0x7C00
<bochs:5> c
(53576736) Breakpoint 10285624, in 0000:7c00 (0x00007c00)
Next at t=2082128
(0) [0x00007c00] 0000:7c00 (unk. ctxt): jmp .+0x003b (0x00007c3e) ; e93b00
<bochs:6> s
Next at t=2082129
(0) [0x00007c3e] 0000:7c3e (unk. ctxt): cli ; fa
<bochs:7>
Next at t=2082130
(0) [0x00007c3f] 0000:7c3f (unk. ctxt): mov ax, 0x07c0 ; b8c007
<bochs:8>
Next at t=2082131
(0) [0x00007c42] 0000:7c42 (unk. ctxt): mov ds, ax ; 8ed8
<bochs:9>
Next at t=2082132
(0) [0x00007c44] 0000:7c44 (unk. ctxt): mov es, ax ; 8ec0
<bochs:10>
Next at t=2082133
(0) [0x00007c46] 0000:7c46 (unk. ctxt): mov fs, ax ; 8ee0
<bochs:11>
Next at t=2082134
(0) [0x00007c48] 0000:7c48 (unk. ctxt): mov gs, ax ; 8ee8
<bochs:12>
Next at t=2082135
(0) [0x00007c4a] 0000:7c4a (unk. ctxt): mov ax, 0x0000 ; b80000
<bochs:13>
Next at t=2082136
(0) [0x00007c4d] 0000:7c4d (unk. ctxt): mov ss, ax ; 8ed0
<bochs:14>
Next at t=2082137
(0) [0x00007c4f] 0000:7c4f (unk. ctxt): mov sp, 0xffff ; bcffff
<bochs:15>
Next at t=2082138
(0) [0x00007c52] 0000:7c52 (unk. ctxt): sti ; fb
<bochs:16>

```

```

boot.asm - Notepad
File Edit Format View Help
;=====
;(3) Blok BOOT CODE
;=====
START:
; Mengatur lokasi kode program pada alamat 7C00:0000, dan mengatur REGISTER SEGMENT
; matikan aktifitas interupsi
mov ax, 0x07C0
mov ds, ax
mov es, ax
mov fs, ax
mov gs, ax

; Mengatur lokasi stack
mov ax, 0x0000
mov ss, ax
mov sp, 0xFFFF ; sp bergerak dari alamat atas ke bawah
sti ; aktifkan aktifitas interupsi

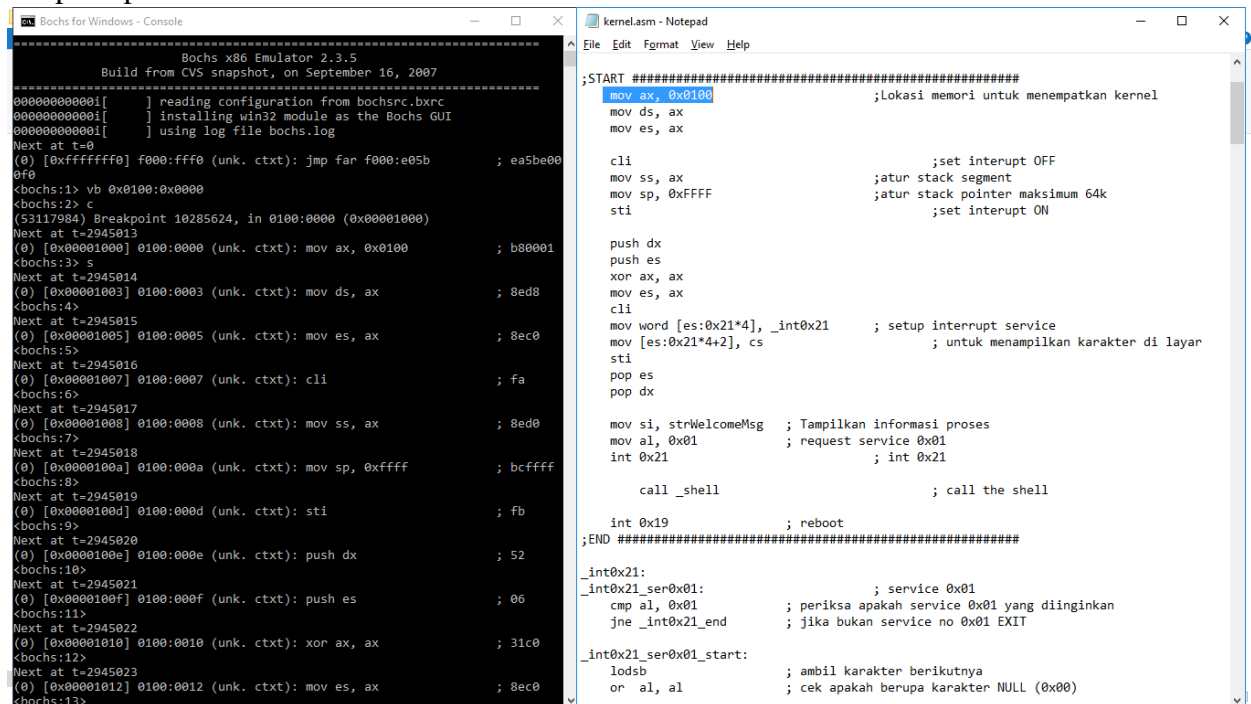
; Menampilkan text di layar
mov si, msgLoading ; mengambil lokasi text yang di simpan dalam 'msgLoading'
call DisplayMessage

LOAD_ROOT:
; menghitung ukuran 'root directory' dan menyimpannya dalam register 'cx'
xor cx, cx
xor dx, dx
mov ax, 0x0020 ; Ukuran satu nama direktori sepanjang 32 byte
mul WORD [MaxRootEntries] ; Total lokasi direktori 32 x 224 (heksa 0x00E0)=7168
div WORD [BytesPerSector] ; lokasi sektor yang digunakan untuk menyimpan direktori 7168/512 (0x0200) = 14
xchg ax, cx ; Ukuran 'root direktori' = 14 sektor

; menghitung jumlah sektor yang digunakan untuk menyimpan FAT

```

5. Kemudian berhenti program dengan perintah *q* . kemudian ketikkan *s*, kemudian buat break point pada “ *vb 0x0100:0x0000*” yang berguna menghentikan program pada “kernel.asm”. kemudian lakukan perintah *s* dan lakukan sebanyak 10 kali. Maka akan tampil seperti ini



The screenshot shows the Bochs x86 Emulator window on the left and the kernel.asm Notepad window on the right. The Bochs window displays the execution state, including the current instruction and the next instruction. The Notepad window shows the assembly code for kernel.asm, which includes comments and instructions for setting up the interrupt service and displaying the kernel information.

```
Bochs x86 Emulator 2.3.5
Build from CVS snapshot, on September 16, 2007
=====
000000000001[ ] reading configuration from bochsrc.bxrc
000000000001[ ] installing win32 module as the Bochs GUI
000000000001[ ] using log file bochs.log
Next at t=0
(0) [0xfffffff0] f000:fff0 (unk. ctxt): jmp far f000:e05b ; ea5be00
0f0
<bochs:1> vb 0x0100:0x0000
<bochs:2> c
(53117984) Breakpoint 10285624, in 0100:0000 (0x00001000)
Next at t=2945013
(0) [0x00001000] 0100:0000 (unk. ctxt): mov ax, 0x0100 ; b80001
<bochs:3> s
Next at t=2945014
(0) [0x00001003] 0100:0003 (unk. ctxt): mov ds, ax ; 8ed8
<bochs:4>
Next at t=2945015
(0) [0x00001005] 0100:0005 (unk. ctxt): mov es, ax ; 8ec0
<bochs:5>
Next at t=2945016
(0) [0x00001007] 0100:0007 (unk. ctxt): cli ; fa
<bochs:6>
Next at t=2945017
(0) [0x00001008] 0100:0008 (unk. ctxt): mov ss, ax ; 8ed0
<bochs:7>
Next at t=2945018
(0) [0x0000100a] 0100:000a (unk. ctxt): mov sp, 0xffff ; bcffff
<bochs:8>
Next at t=2945019
(0) [0x0000100d] 0100:000d (unk. ctxt): sti ; fb
<bochs:9>
Next at t=2945020
(0) [0x0000100e] 0100:000e (unk. ctxt): push dx ; 52
<bochs:10>
Next at t=2945021
(0) [0x0000100f] 0100:000f (unk. ctxt): push es ; 06
<bochs:11>
Next at t=2945022
(0) [0x00001010] 0100:0010 (unk. ctxt): xor ax, ax ; 31c0
<bochs:12>
Next at t=2945023
(0) [0x00001012] 0100:0012 (unk. ctxt): mov es, ax ; 8ec0
<bochs:13>
```

```
kernel.asm - Notepad
File Edit Format View Help
;START #####
mov ax, 0x0100 ;lokasi memori untuk menempatkan kernel
mov ds, ax
mov es, ax

cli ;set interrupt OFF
mov ss, ax ;atur stack segment
mov sp, 0xFFFF ;atur stack pointer maksimum 64k
sti ;set interrupt ON

push dx
push es
xor ax, ax
mov es, ax
cli
mov word [es:0x21*4], _int0x21 ; setup interrupt service
mov [es:0x21*4+2], cs ; untuk menampilkan karakter di layar
sti
pop es
pop dx

mov si, strWelcomeMsg ; Tampilkan informasi proses
mov al, 0x01 ; request service 0x01
int 0x21 ; int 0x21

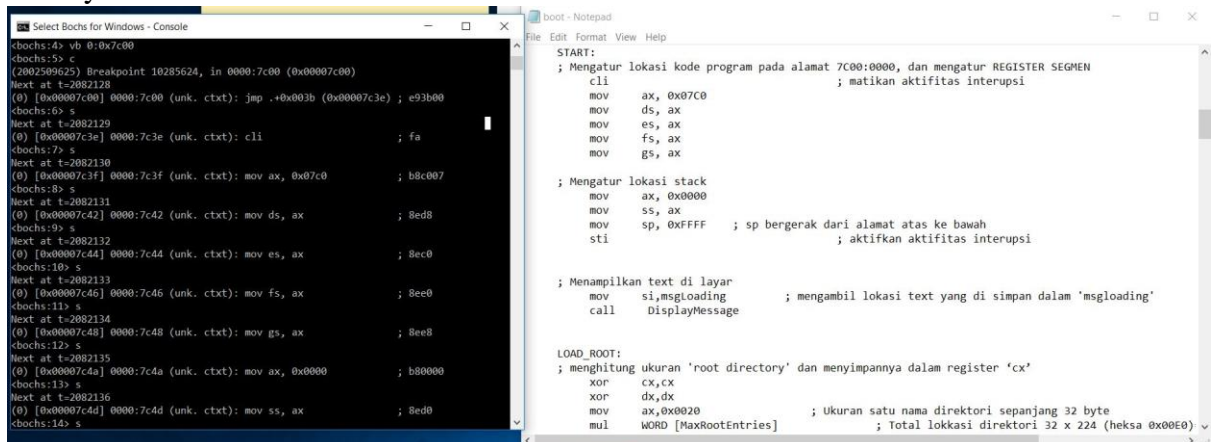
call _shell ; call the shell

int 0x19 ; reboot
;END #####

_int0x21:
_int0x21_ser0x01: ; service 0x01
cmp al, 0x01 ; periksa apakah service 0x01 yang diinginkan
jne _int0x21_end ; jika bukan service no 0x01 EXIT

_int0x21_ser0x01_start:
lodsb ; ambil karakter berikutnya
or al, al ; cek apakah berupa karakter NULL (0x00)
```

6. Membandingkan hasil yang dieksekusi dengan yang ada di *file boot.asm*. Dan membandingkan hasil eksekusi dengan fdengan yang ada di dalam *kernel.asm* ternyata hasilnya sama.



The screenshot shows the Select Bochs for Windows - Console window on the left and the boot.asm Notepad window on the right. The console window displays the execution state, including the current instruction and the next instruction. The Notepad window shows the assembly code for boot.asm, which includes comments and instructions for setting up the interrupt service and displaying the boot information.

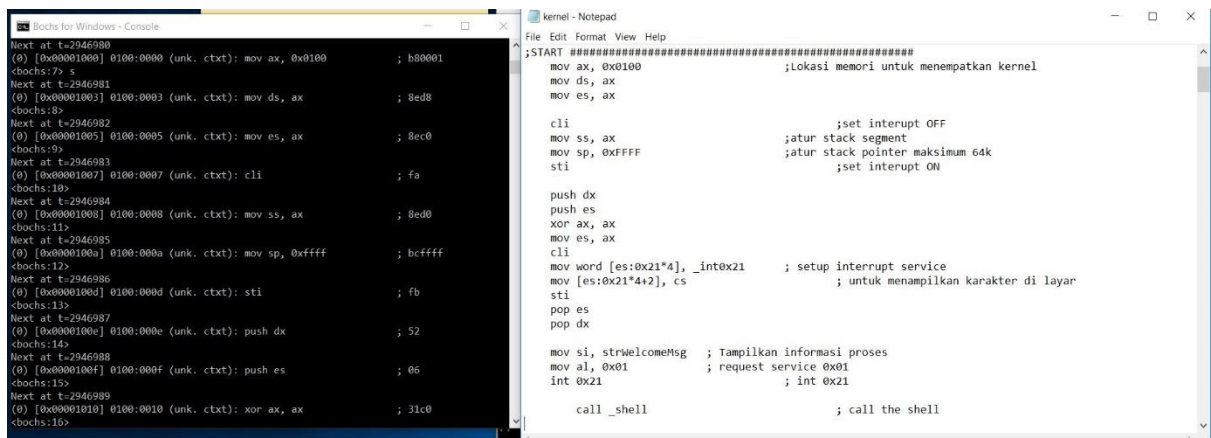
```
Select Bochs for Windows - Console
<bochs:4> vb 0:0x7c00
<bochs:5> c
(200250025) Breakpoint 10285624, in 0000:7c00 (0x00007c00)
Next at t=2082128
(0) [0x00007c00] 0000:7c00 (unk. ctxt): jmp .+0x003b (0x00007c3e) ; e93b00
<bochs:6> s
Next at t=2082129
(0) [0x00007c3e] 0000:7c3e (unk. ctxt): cli ; fa
<bochs:7> s
Next at t=2082130
(0) [0x00007c3f] 0000:7c3f (unk. ctxt): mov ax, 0x07c0 ; b8c007
<bochs:8> s
Next at t=2082131
(0) [0x00007c42] 0000:7c42 (unk. ctxt): mov ds, ax ; 8ed8
<bochs:9> s
Next at t=2082132
(0) [0x00007c44] 0000:7c44 (unk. ctxt): mov es, ax ; 8ec0
<bochs:10> s
Next at t=2082133
(0) [0x00007c46] 0000:7c46 (unk. ctxt): mov fs, ax ; 8ee0
<bochs:11> s
Next at t=2082134
(0) [0x00007c48] 0000:7c48 (unk. ctxt): mov gs, ax ; 8ee8
<bochs:12> s
Next at t=2082135
(0) [0x00007c4a] 0000:7c4a (unk. ctxt): mov ax, 0x0000 ; b80000
<bochs:13> s
Next at t=2082136
(0) [0x00007c4d] 0000:7c4d (unk. ctxt): mov ss, ax ; 8ed0
<bochs:14> s
```

```
boot.asm - Notepad
File Edit Format View Help
;START:
; Mengatur lokasi kode program pada alamat 7C00:0000, dan mengatur REGISTER SEGMENT
cli ; matikan aktifitas interupsi
mov ax, 0x07C0
mov ds, ax
mov es, ax
mov fs, ax
mov gs, ax

; Mengatur lokasi stack
mov ax, 0x0000
mov ss, ax
mov sp, 0xFFFF ; sp bergerak dari alamat atas ke bawah
sti ; aktifkan aktifitas interupsi

; Menampilkan text di layar
mov si, msgLoading ; mengambil lokasi text yang di simpan dalam 'msgloading'
call DisplayMessage

LOAD_ROOT:
; menghitung ukuran 'root directory' dan menyimpannya dalam register 'cx'
xor cx, cx
xor dx, dx
mov ax, 0x0020 ; ukuran satu nama direktori sepanjang 32 byte
mul WORD [MaxRootEntries] ; Total lokasi direktori 32 x 224 (heksa 0x00E0)
```



The screenshot shows the Bochs for Windows - Console window on the left and the kernel.asm Notepad window on the right. The console window displays the execution state, including the current instruction and the next instruction. The Notepad window shows the assembly code for kernel.asm, which includes comments and instructions for setting up the interrupt service and displaying the kernel information.

```
Bochs for Windows - Console
Next at t=2946980
(0) [0x00001000] 0100:0000 (unk. ctxt): mov ax, 0x0100 ; b80001
<bochs:7> s
Next at t=2946981
(0) [0x00001003] 0100:0003 (unk. ctxt): mov ds, ax ; 8ed8
<bochs:8>
Next at t=2946982
(0) [0x00001005] 0100:0005 (unk. ctxt): mov es, ax ; 8ec0
<bochs:9>
Next at t=2946983
(0) [0x00001007] 0100:0007 (unk. ctxt): cli ; fa
<bochs:10>
Next at t=2946984
(0) [0x00001008] 0100:0008 (unk. ctxt): mov ss, ax ; 8ed0
<bochs:11>
Next at t=2946985
(0) [0x0000100a] 0100:000a (unk. ctxt): mov sp, 0xffff ; bcffff
<bochs:12>
Next at t=2946986
(0) [0x0000100d] 0100:000d (unk. ctxt): sti ; fb
<bochs:13>
Next at t=2946987
(0) [0x0000100e] 0100:000e (unk. ctxt): push dx ; 52
<bochs:14>
Next at t=2946988
(0) [0x0000100f] 0100:000f (unk. ctxt): push es ; 06
<bochs:15>
Next at t=2946989
(0) [0x00001010] 0100:0010 (unk. ctxt): xor ax, ax ; 31c0
<bochs:16>
```

```
kernel.asm - Notepad
File Edit Format View Help
;START #####
mov ax, 0x0100 ;lokasi memori untuk menempatkan kernel
mov ds, ax
mov es, ax

cli ;set interrupt OFF
mov ss, ax ;atur stack segment
mov sp, 0xFFFF ;atur stack pointer maksimum 64k
sti ;set interrupt ON

push dx
push es
xor ax, ax
mov es, ax
cli
mov word [es:0x21*4], _int0x21 ; setup interrupt service
mov [es:0x21*4+2], cs ; untuk menampilkan karakter di layar
sti
pop es
pop dx

mov si, strWelcomeMsg ; Tampilkan informasi proses
mov al, 0x01 ; request service 0x01
int 0x21 ; int 0x21

call _shell ; call the shell
```

TUGAS !!**a. Membuat tabel pemetaan memori pada PC**

No.	Blok Memori	Alokasi Pemakaian
1	F 0 0 0 0	ROM BIOS, Diagnostic, BASIC
2	E 0 0 0 0	ROM program
3	D 0 0 0 0	ROM program
4	C 0 0 0 0	Perluasan BIOS untuk hardisk XT
5	B 0 0 0 0	Monokrom Monitor
6	A 0 0 0 0	Monitor EGA, VGS, dll
7	9 0 0 0 0	Daerah kerja pemakai s/d 640 KB
8	8 0 0 0 0	Daerah kerja pemakai s/d 576 KB
9	7 0 0 0 0	Daerah kerja pemakai s/d 512 KB
10	6 0 0 0 0	Daerah kerja pemakai s/d 448 KB
11	5 0 0 0 0	Daerah kerja pemakai s/d 384 KB
12	4 0 0 0 0	Daerah kerja pemakai s/d 320 KB
13	3 0 0 0 0	Daerah kerja pemakai s/d 256 KB
14	2 0 0 0 0	Daerah kerja pemakai s/d 192 KB
15	1 0 0 0 0	Daerah kerja pemakai s/d 128 KB
16	0 0 0 0 0	Daerah kerja pemakai s/d 64 KB

b. Menjelaskan perbedaan antara real mode dengan protect mode pada PC IBM Compatible

- Real-Mode

Real-Mode adalah sebuah modus di mana prosesor Intel x86 berjalan seolah-olah dirinya adalah sebuah prosesor Intel 8085 atau Intel 8088, meski ia merupakan prosesor Intel 80286 atau lebih tinggi. Karenanya, modus ini juga disebut sebagai modus 8086 (8086 Mode). Dalam modus ini, prosesor hanya dapat mengeksekusi instruksi 16-bit saja dengan menggunakan register internal yang berukuran 16-bit, serta hanya dapat mengakses hanya 1024 KB dari memori karena hanya menggunakan 20-bit jalur bus alamat. Semua program DOS berjalan pada modus ini.

- Protected Mode

Modus terproteksi (protected mode) adalah sebuah modus di mana terdapat proteksi ruang alamat memori yang ditawarkan oleh mikroprosesor untuk digunakan oleh sistem operasi. Modus ini datang dengan mikroprosesor Intel 80286 atau yang lebih tinggi. Karena memiliki proteksi ruang alamat memori, maka dalam modus ini sistem operasi dapat melakukan multitasking.