

**TUGAS**  
**PRAKTIKUM SISTEM OPERASI**  
**MENGENAL CARA ‘DEBUGGING’ PROGRAM BOOTSTRAP-LOADER**



**Disusun Oleh :**

**DIAN NUR HAYATI**

**L200210229**

**Kelas E**

**PROGRAM STUDI TEKNIK INFORMATIKA**  
**FAKULTAS KOMUNIKASI DAN INFORMATIKA**  
**UNIVERSITAS MUHAMMADIYAH SURAKARTA TAHUN 2022/2023**

NIM	: L200210229	Nilai :
Nama	: Dian Nur Hayati	
Kelas	: E	
Dosen pengampu	: Heru Setya Nugraha, S.T.,M.Kom	Tanda tangan :
Tanggal Praktikum	: 27 September 2022	

## Langkah Kerja

1. Mengatur 'path' dan pergi ke direktori kerja

```

C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\LABRPL-22>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>

```

2. Mengetikkan perintah 'type s.bat'

```

C:\OS\LAB\LAB3>type s.bat
..\..\bochs-2.3.5\bochsdbg -q -f bochsrc.bxrc

C:\OS\LAB\LAB3>

```

3. Mulai melakukan 'debugging'; masukkan perintah 'S' <ENTER>.

```

C:\OS\LAB\LAB3>s

C:\OS\LAB\LAB3>..\..\bochs-2.3.5\bochsdbg -q -f bochsrc.bxrc
00000000000i[APIC?] local apic in  initializing
=====
                Bochs x86 Emulator 2.3.5
                Build from CUS snapshot, on September 16, 2007
=====
00000000000i[      ] reading configuration from bochsrc.bxrc
00000000000i[      ] installing win32 module as the Bochs GUI
00000000000i[      ] using log file bochs.log
Next at t=0
<0> [0xffffffff] f000:fff0 <unk. ctxt>: jmp far f000:e05b          ; ea5be000f0
<bochs:1>

```

4. Melihat isi register CS dan IP dengan perintah 'r'.

```
<bochs:1> r
rax: 0x00000000:00000000 rcx: 0x00000000:00000000
rdx: 0x00000000:00000f20 rbx: 0x00000000:00000000
rsp: 0x00000000:00000000 rbp: 0x00000000:00000000
rsi: 0x00000000:00000000 rdi: 0x00000000:00000000
r8 : 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>
```

5. Selanjutnya kita suruh PC untuk mengeksekusi perintah tersebut, ketikkan 's' <ENTER> kemudian lanjutkan dengan perintah 'r' <ENTER>.

```
<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: 0x00000000:00000f20 rbx: 0x00000000:00000000
rsp: 0x00000000:00000000 rbp: 0x00000000:00000000
rsi: 0x00000000:00000000 rdi: 0x00000000:00000000
r8 : 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>
```

6. Masukkan perintah 'vb 0:7xC00' <ENTER>, selanjutnya masukkan perintah 'c' <ENTER>

```
<bochs:4> vb 0:0x7c00
<bochs:5> c
<0> Breakpoint 2683464, in 0000:7c00 <0x00007c00>
Next at t=2082128
<0> [0x00007c00] 0000:7c00 <unk. ctxt>: jmp .+0x003b <0x00007c3e> ; e93b00
<bochs:6>
```

7. Sekarang PC mulai memasuki tahapan 'BOOTSTRAP-LOADER', selanjutnya bandingkan 10 instruksi berikutnya akan dieksekusi oleh PC dengan program yang terdapat pada 'boot.asm'

```
<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> q
# In bx_win32_gui_c::exit(void)!
Bochs is exiting. Press ENTER when you're ready to close this window.
```

8. Selanjutnya anda dapat memerintahkan PC Simulator untuk melanjutkan pekerjaannya.

```
C:\OS\LAB\LAB3>s
C:\OS\LAB\LAB3>..\..\bochs-2.3.5\bochsdbg -q -f bochsrc.bxrc
000000000000i[APIC?] local apic in initializing
=====
                Bochs x86 Emulator 2.3.5
                Build from CVS snapshot, on September 16, 2007
=====
000000000000i[      ] reading configuration from bochsrc.bxrc
000000000000i[      ] installing win32 module as the Bochs GUI
000000000000i[      ] using log file bochs.log
Next at t=0
<0> [0xffffffff] f000:fff0 <unk. ctxt>: jmp far f000:e05b    ; ea5be000f0
```

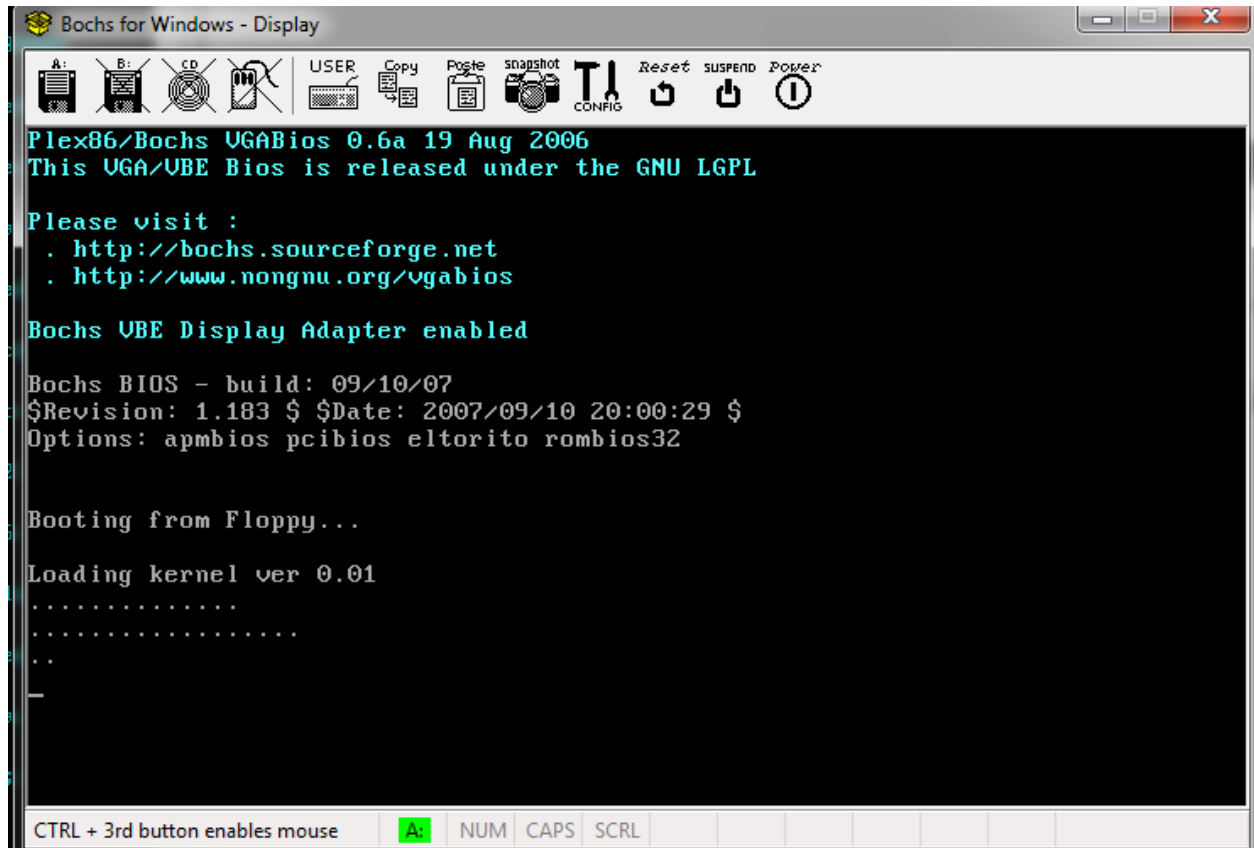
9. Menghentikan PC Simulator pada saat akan menjalankan program 'kernel.bin'

```
<bochs:1> vb 0x0100:0x0000
<bochs:2> c
<0> Breakpoint 2683464, in 0100:0000 <0x00001000>
Next at t=2945013
<0> [0x00001000] 0100:0000 <unk. ctxt>: mov ax, 0x0100      ; b80001
```

10. Selanjutnya teruskan langkah PC simulator step by step minimal sebanyak 10x, ketik 's' <ENTER> ,selanjutnya menekan <ENTER> secara langsung.

```
<bochs:3> s
Next at t=2945014
<0> [0x00001003] 0100:0003 (unk. ctxt): mov ds, ax ; 8ed8
<bochs:4> s
Next at t=2945015
<0> [0x00001005] 0100:0005 (unk. ctxt): mov es, ax ; 8ec0
<bochs:5> s
Next at t=2945016
<0> [0x00001007] 0100:0007 (unk. ctxt): cli ; fa
<bochs:6> s
Next at t=2945017
<0> [0x00001008] 0100:0008 (unk. ctxt): mov ss, ax ; 8ed0
<bochs:7> s
Next at t=2945018
<0> [0x0000100a] 0100:000a (unk. ctxt): mov sp, 0xffff ; bfffff
<bochs:8> s
Next at t=2945019
<0> [0x0000100d] 0100:000d (unk. ctxt): sti ; fb
<bochs:9> s
Next at t=2945020
<0> [0x0000100e] 0100:000e (unk. ctxt): push dx ; 52
<bochs:10> s
Next at t=2945021
<0> [0x0000100f] 0100:000f (unk. ctxt): push es ; 06
<bochs:11> s
Next at t=2945022
<0> [0x00001010] 0100:0010 (unk. ctxt): xor ax, ax ; 31c0
<bochs:12> s
Next at t=2945023
<0> [0x00001012] 0100:0012 (unk. ctxt): mov es, ax ; 8ec0
<bochs:13>
Next at t=2945024
<0> [0x00001014] 0100:0014 (unk. ctxt): cli ; fa
<bochs:14>
Next at t=2945025
<0> [0x00001015] 0100:0015 (unk. ctxt): mov word ptr es:0x84, 0x0030 ; 26c706840
03000
<bochs:15>
```

## 11. Program 'kernel.asm'



## Tugas

1. Buatlah tabel pemetaan memori pada PC selengkap mungkin.

**Jawab:**

Blok Memori	Alokasi Pemakaian
<b>F 0 0 0 0</b>	ROM BIOS, Diagnostic, BASIC
<b>E 0 0 0 0</b>	ROM program
<b>D 0 0 0 0</b>	ROM program
<b>C 0 0 0 0</b>	Perluasan BIOS untuk hardisk XT
<b>B 0 0 0 0</b>	Monokrom Monitor
<b>A 0 0 0 0</b>	Monitor EGA, VGS, dll
<b>9 0 0 0 0</b>	Daerah kerjapemakai s/d 640 KB
<b>8 0 0 0 0</b>	Daerah kerjapemakai s/d 576 KB
<b>7 0 0 0 0</b>	Daerah kerjapemakai s/d 512 KB
<b>6 0 0 0 0</b>	Daerah kerjapemakai s/d 448 KB
<b>5 0 0 0 0</b>	Daerah kerjapemakai s/d 384 KB
<b>4 0 0 0 0</b>	Daerah kerjapemakai s/d 320 KB
<b>3 0 0 0 0</b>	Daerah kerjapemakai s/d 256 KB
<b>2 0 0 0 0</b>	Daerah kerjapemakai s/d 192 KB
<b>1 0 0 0 0</b>	Daerah kerjapemakai s/d 128 KB
<b>0 0 0 0 0</b>	Daerah kerjapemakai s/d 64 KB

2. Baca buku referensi, jelaskan perbedaan antara mode kerja 'Real-Mode' dan mode kerja 'Protect-mode' pada PC IMB Compatible.

**Jawab:**

a. 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.

b. 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.

