Nama : Luqman Hanung Asidiq
NIM : L200180035
Kelas : A

Tugas Praktikum Sistem Operasi Modul 3

Proses Debugging

```
Microsoft Windows [Version 10.0.17763.107]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Windows 10>cd..

C:\Users>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.bat

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 CVS 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) [0xfffffff0] f000:fff0 (unk. ctxt): jmp far f000:e05b         ; ea5be000f0
<bochs:1> s
Next at t=1
(0) [0x000fe05b] f000:e05b (unk. ctxt): xor ax, ax                ; 31c0
<bochs:2> 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:3> r
rax: 0x00000000:00000000 rcx: 0x00000000:00000000
```

```
IOPL=0 id vip vif ac vm rf nt of df if tf sf zf af pf cf
<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> vb 0:0x7C00
<bochs:5> c
(10264512) Breakpoint 10285608, 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> q
# In bx_win32_gui_c::exit(void)!

Bochs is exiting. Press ENTER when you're ready to close this window.


C:\OS\LAB\LAB3>s

C:\OS\LAB\LAB3>..\..\bochs-2.3.5\bochsdbg -q -f bochsrc.bxrc
00000000000i[APIC?] local apic in  initializing
========================================================================
```

```
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 CVS 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) [0xfffffff0] f000:fff0 (unk. ctxt): jmp far f000:e05b         ; ea5be000f0
<bochs:1> vb 0x0100:0x0000
<bochs:2> c
(10264512) Breakpoint 10285608, 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> 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            ; bcffff
<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
```

```
Bochs for Windows - Console
<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>
```

Tugas!!

1. Table pemetaan memori pada pc

a) Pemetaan Langsung (Direct Mapping)

Item	Keterangan
Panjang alamat	(s+w) bits
Jumlah unit yang dapat dialamati	2s+w words or bytes
Ukuran Bloks sama dengan ukuran Line	2w words or bytes
Jumlah blok memori utama	2s+ w/2w = 2s
Jumlah line di chace	M = 2r
Besarnya tag	(s - r) bits

b) Pemetaan Asosiatif (Associative Mapping)

Item	Keterangan
Panjang alamat	(s+w) bits
Jumlah unit yang dapat dialamati	2s+w words or bytes
Ukuran Bloks sama dengan ukuran Line	2w words or bytes
Jumlah blok memori utama	2s+ w/2w = 2s

Jumlah line di chace	Undetermined
Besarnya tag	s bits

c) Pemetaan Asosiatif Set (Set Associative Mapping)

Item	**Keterangan**
Panjang alamat	(s+w) bits
Jumlah unit yang dapat dialamati	2s+w words or bytes
Ukuran Bloks sama dengan ukuran Line	2w words or bytes
Jumlah blok memori utama	2d
Jumlah line dalam set	k
Jumlah set	V=2d
Jumlah line di chace	Kv = k*2d
Besarnya tag	(s – d)bits

2. Pembedaan antara real mode dan protect mode pada pc ibm compatible

a) Real Mode

Real-Mode adalah sebuah modus di mana prosesor Intel x86 berjalan seolah-olah dirinya adalah sebuah prosesor Intel 8086 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.

Prosesor yang dirilis setelah 8086, semacam Intel 80286 juga dapat menjalankan instruksi 16-bit, tetapi jauh lebih cepat dibandingkan 8086. Dengan kata lain, Intel 80286 benar-benar kompatibel dengan prosesor Intel 8086 yang didesain sebelumnya. Sehingga prosesor Intel 80286 pun dapat menjalankan program-program 16-bit yang didesain untuk 8086 (IBM PC), dengan tentunya kecepatan yang jauh lebih tinggi.

Dalam Real-mode, tidak ada proteksi ruang alamat memori, sehingga tidak dapat melakukan *multi-tasking*. Inilah sebabnya, mengapa program-program DOS bersifat *single-tasking*. Jika dalam modus real terdapat *multi-tasking*, maka kemungkinan besar antara dua program yang sedang berjalan, terjadi tabrakan (*crash*) antara satu dengan lainnya.

b) Protect 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.

Prosesor Intel 80286 memang dilengkapi kemampuan masuk ke dalam modus terproteksi, tetapi tidak dapat keluar dari modus tersebut tanpa harus mengalami reset (*warm boot* atau *cold boot*). Kesalahan ini telah diperbaiki oleh Intel dengan merilis prosesor Intel 80386 yang dapat masuk ke dalam modus terproteksi dan keluar darinya tanpa harus melakukan reset. Inilah sebabnya mengapa Windows 95/Windows 98 dilengkapi dengan modus **Restart in MS-DOS Mode**, meski sebenarnya sistem operasi tersebut merupakan sistem operasi yang berjalan dalam modus terproteksi.