



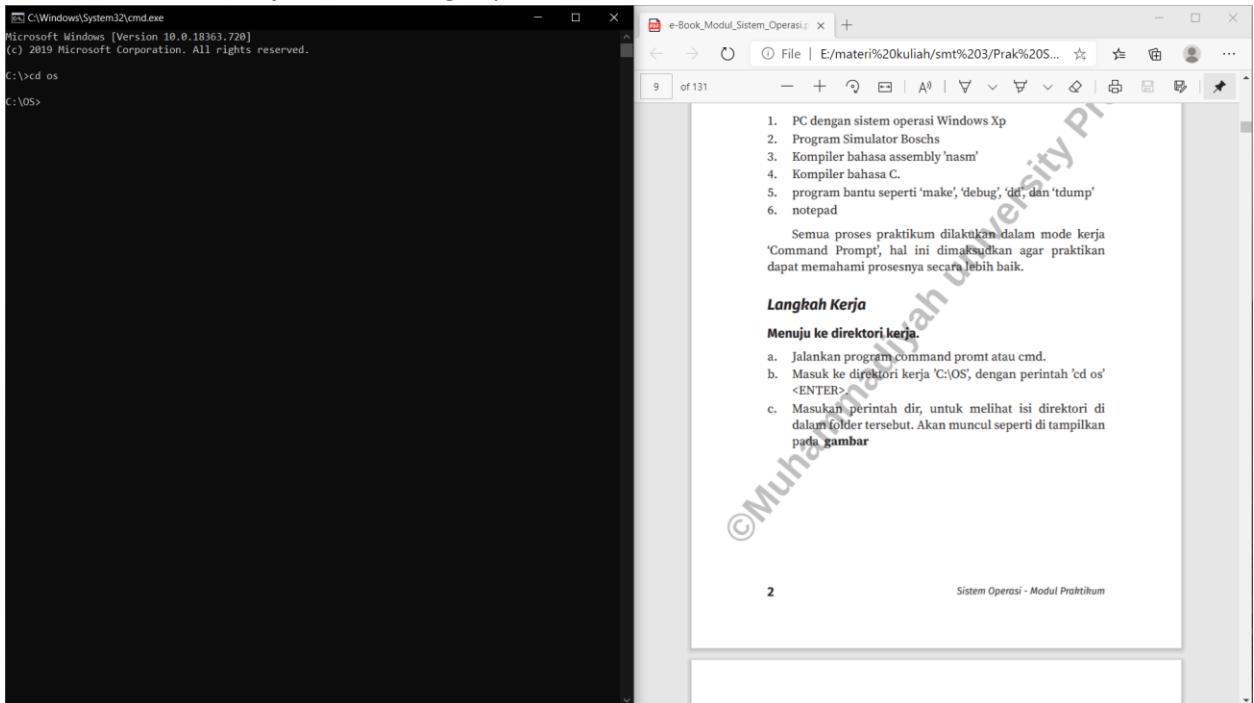
LAPORAN KEGIATAN PRAKTIKUM SISTEM OPERASI

PENGENALAN SISTEM PEMGEMBANG OS

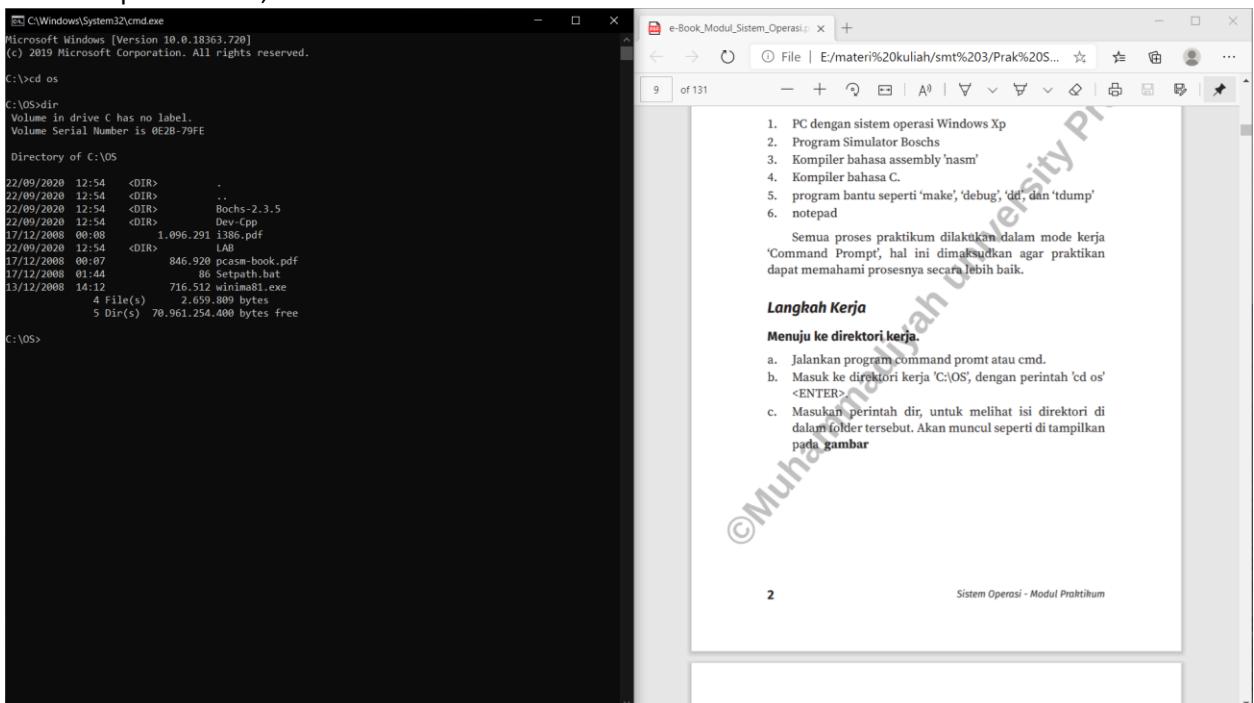


NAMA : MUHAMAD YUSUF FATKURRAHAN
NIM : L20019026
KELAS : G

1. Menjalankan program command prompt atau cmd
2. Masuk ke direktori kerja 'C:\OS', dengan perintah 'cd os' .<ENTER>



3. Masukan perintah dir, untuk melihat isi direktori di dalam folder tersebut



4. Jalankan file setpath, untuk menjalankannya ketik 'setpath' tekan<ENTER>

C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.18363.720]
(c) 2019 Microsoft Corporation. All rights reserved.
C:\>cd os
C:\OS>dir
Volume in drive C has no label.
Volume Serial Number is 0E2B-79FE
Directory of C:\OS
22/09/2020 12:54 <DIR> .
22/09/2020 12:54 <DIR> ..
22/09/2020 12:54 <DIR> Bochs-2.3.5
22/09/2020 12:54 <DIR> Dev-Cpp
17/12/2008 00:08 1,096,291 i386.pdf
22/09/2020 12:54 <DIR> .
17/12/2008 00:07 846,920 pcasm-book.pdf
17/12/2008 01:44 86 Setpath.bat
13/12/2008 14:12 716,512 winimage01.exe
4 File(s) 2,659,809 bytes
5 Dir(s) 70,961,254,400 bytes free
C:\OS>setpath.bat
C:\OS>Path=C:\OS\Dev-Cpp\bin;C:\OS\Bochs-2.3.5;c:\OS\Perl;C:\Windows;C:\Windows\System32
C:\OS>

e-Book_Modul_Sistem_Operasi x
10 of 131
File | E/materi%20kuliah/smt%203/Prak%20S...
14/12/2008 03:46 <DIR> .
14/12/2008 03:46 <DIR> ..
14/12/2008 03:44 <DIR> Bochs-2.3.5
14/12/2008 03:44 <DIR> Dev-Cpp
14/12/2008 03:49 Setpath.bat
5 File(s) 15,977,971,712 bytes free
C:\OS>

©Muhammad Syaiful University Press

Gambar 1.1 Struktur direktori kerja

- d. Jalankan file setpath, untuk menjalankannya ketik 'setpath' tekan<ENTER>
File 'setpath.bat' digunakan untuk mengatur lingkungan kerja ('path') selama anda melakukan praktikum, anda harus menjalankan program ini sebelum memulai setiap sesi praktikum anda, untuk menjalankannya ketik 'setpath' tekan <ENTER>. Perhatikan teks yang muncul di layar, seperti ditampilkan pada Gambar 1.1.

Untuk melihat script yang terdapat di dalam file 'setpath.bat' dapat digunakan perintah 'type setpath.bat', cobalah 'PATH' dapat berisi banyak daftar lokasi, di antara item lokasi di batasi dengan karakter ';' (titik komma). Variabel 'PATH' ini akan digunakan oleh windows untuk mencari lokasi file yang dipanggil oleh 'user'. Urutan pencarian dimulai dari daftar lokasi yang paling awal.

5. Masuk ke direktori kerja pada 'C:\OS\LAB\LAB1'

C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.18363.720]
(c) 2019 Microsoft Corporation. All rights reserved.
C:\>cd os
C:\OS>dir
Volume in drive C has no label.
Volume Serial Number is 0E2B-79FE
Directory of C:\OS
22/09/2020 12:54 <DIR> .
22/09/2020 12:54 <DIR> ..
22/09/2020 12:54 <DIR> Bochs-2.3.5
22/09/2020 12:54 <DIR> Dev-Cpp
17/12/2008 00:08 1,096,291 i386.pdf
22/09/2020 12:54 <DIR> LAB
17/12/2008 00:07 846,920 pcasm-book.pdf
17/12/2008 01:44 86 Setpath.bat
13/12/2008 14:12 716,512 winimage01.exe
4 File(s) 2,659,809 bytes
5 Dir(s) 70,961,254,400 bytes free
C:\OS>setpath.bat
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
C:\OS\LAB>d lab1
C:\OS\LAB\LAB1>

e-Book_Modul_Sistem_Operasi x
12 of 131
File | E/materi%20kuliah/smt%203/Prak%20S...
Melihat isi direktori kerja
Untuk masuk ke direktori kerja untuk modul ini pertama adalah:
a. Jalankan command prompt
b. Masuk ke direktori kerja pada 'C:\OS\LAB\LAB1'
c. Cobalah untuk membuat file tersebut dengan perintah berikut, dari 'COMMAND PROMPT', ketikkan 'Notepad boot.asm' dan tekan <ENTER>. Saat ini sebaiknya anda tidak melakukan modifikasi terhadap program tersebut. Tutuplah kembali program 'notepad'-nya. Source code prototype program bootloader disimpan dalam file 'boot.asm' sedangkan untuk source code prototype program kernel disimpan dalam file 'kernel.asm' listing lengkap keduaanya bisa dibaca di lampiran. Sebaiknya anda memahami jalannya kedua program sebelum melakukan praktikum. Kedua program ditulis dalam bahasa assembly, anda dapat membukanya dengan menggunakan program aplikasi teks-editor sembarang, termasuk 'Notepad.exe' milik windows juga dapat digunakan.
d. Selain kedua file source code ada sebuah file 'image floppy' dengan nama file 'floppy.img' file ini yang akan digunakan untuk menyimpan hasil kompilasi kedua source code, kemudian digunakan sebagai 'boot disk' pada PC simulator 'Bochs'. Sebagai kompiler digunakan program 'nasm.exe' yang terletak dalam subdirektori 'C:\OS\Dev-Cpp\bin' dan pada direktori tersebut juga ada program 'dd.exe' untuk memindahkan byte data dari satu file ke file yang lain, program 'make.exe' untuk mempercepat proses kompilasi. Pada saat ini kita akan menggunakan ketiga program tersebut yang dikemas dalam script 'Makefile' sehingga proses kompilasi menjadi lebih cepat, cukup dengan memanggil 'make'.

©Muhammad Syaiful University Press

6. Mengetik perintah ‘Notepad boot.asm’ dan tekan <ENTER>

```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.18363.720]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\>cd os

C:\OS>dir
Volume in drive C has no label.
Volume Serial Number is 0E2B-79FE

Directory of C:\OS

22/09/2020 12:54 <DIR> .
22/09/2020 12:54 <DIR> ..
22/09/2020 12:54 <DIR> Bochs-2.3.5
22/09/2020 12:54 <DIR> Dev-Cpp
17/12/2008 00:08 1.096.291 i386.pdf
22/09/2020 12:54 <DIR> LAB
17/12/2008 00:07 846.928 pcasm-book.pdf
17/12/2008 01:44 86 Setpath.bat
13/12/2008 14:12 716.512 winim81.exe
               4 File(s) 2.659.809 bytes
               5 Dir(s) 70.961.254.400 bytes free

C:\OS>setpath.bat

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

C:\OS\LAB>>notepad boot.asm
C:\OS\LAB\LAB1>
```

```
boot.asm - Notepad
File Edit Format View Help
*****
; LAB-1 : boot-strap loader - real mode
; untuk memindahkan file OS dari floppy disk format DOS FAT12
; *****
; atur mode kerja 16 bit (real-mode)
[BITS 16]

; Menentukan lokasi awal dari program
[ORG 0x0000]

; locat ke label START
jmp START

; Keterangan format floppy disk format FAT12
        OEM_ID          db "QUASI-OS"
        BytesPerSector   dw 0x0200
        SectorsPerCluster dw 0x01
        ReservedSectors  dw 0x0001
        TotalFATs        db 0x02
        MaxRootEntries    dw 0x00E0
        TotalSectorsSmall dw 0x0840
        MediaDescriptor   db 0xF0
        SectorsPerFAT     dw 0x0009
        SectorsPerTrack   dw 0x0012
        NumHeads          dw 0x0002
        HiddenSectors     dd 0x00000000
        TotalSectorsLarge dd 0x00000000
```

©Muhammad Firdaus

Ln 1, Col 1 100% Windows (CRLF) ANSI

‘C:\OS\Dev-Cpp\bin’ dan pada direktori tersebut juga ada program ‘dd.exe’ untuk memindahkan byte data dari satu file ke file yang lain, program ‘make.exe’ untuk mempercepat proses komplilasi. Pada saat ini kita akan menggunakan ketiga program tersebut yang dikemas dalam script ‘Makefile’ sehingga proses komplilasi menjadi lebih cepat, cukup dengan memanggil ‘make’.

7. Mengetik perintah ‘Notepad M’ tekan tombol ‘TAB’ sehingga muncul ‘Notepad Makefile’ dan tekan <ENTER>

```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.18363.720]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\>cd os

C:\OS>dir
Volume in drive C has no label.
Volume Serial Number is 0E2B-79FE

Directory of C:\OS

22/09/2020 12:54 <DIR> .
22/09/2020 12:54 <DIR> ..
22/09/2020 12:54 <DIR> Bochs-2.3.5
22/09/2020 12:54 <DIR> Dev-Cpp
17/12/2008 00:08 1.096.291 i386.pdf
22/09/2020 12:54 <DIR> LAB
17/12/2008 00:07 846.928 pcasm-book.pdf
17/12/2008 01:44 86 Setpath.bat
13/12/2008 14:12 716.512 winim81.exe
               4 File(s) 2.659.809 bytes
               5 Dir(s) 70.961.254.400 bytes free

C:\OS>setpath.bat

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

C:\OS\LAB>>notepad boot.asm
C:\OS\LAB\LAB1>>notepad Makefile
C:\OS\LAB\LAB1>
```

```
Makefile - Notepad
File Edit Format View Help
#
# LAB01 - Makefile
#
fp.disk: boot
        dd if=boot.bin of=floppy.img

boot: boot.asm
        nasm boot.asm -o boot.bin -f bin

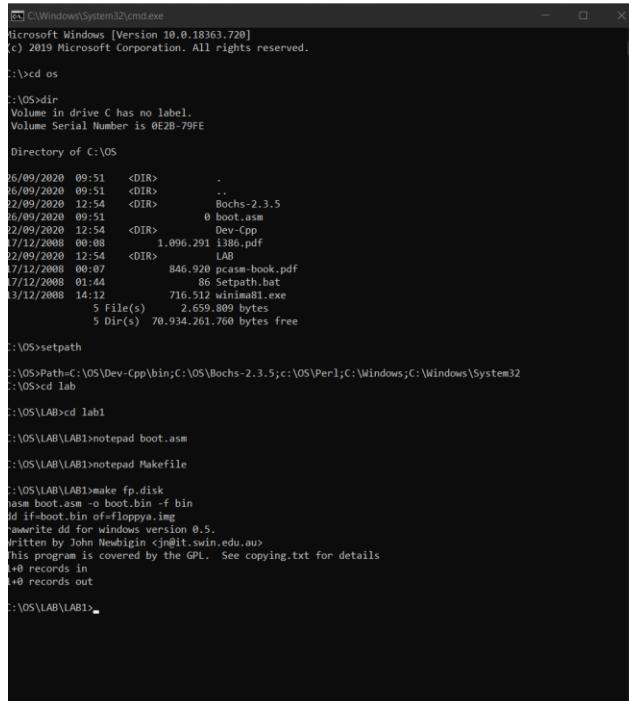
kernel: kernel.asm
        nasm kernel.asm -o kernel.bin -f bin
clean:
        rm -f *.bin boot kernel
```

©Muhammad Firdaus

Ln 1, Col 1 100% Windows (CRLF) UTF-8

‘C:\OS\Dev-Cpp\bin’ dan pada direktori tersebut juga ada program ‘dd.exe’ untuk memindahkan byte data dari satu file ke file yang lain, program ‘make.exe’ untuk mempercepat proses komplilasi. Pada saat ini kita akan menggunakan ketiga program tersebut yang dikemas dalam script ‘Makefile’ sehingga proses komplilasi menjadi lebih cepat, cukup dengan memanggil ‘make’.

8. Sekarang membuka ‘Command Prompt’ dan buka direktori kerja ‘LAB1’ selanjutnya ketik ‘make fp.disk’



```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.18363.729]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\>cd os

C:\OS>dir
Volume in drive C has no label.
Volume Serial Number is 0E2B-79FE

Directory of C:\OS

26/09/2020 09:51 <DIR> .
26/09/2020 09:51 <DIR> ..
22/09/2020 12:54 <DIR> Bochs-2.3.5
16/09/2020 09:51 0 boot.asm
22/09/2020 12:54 <DIR> Dev-Cpp
7/12/2008 00:08 1,996,291 i386.pdf
22/09/2020 12:54 <DIR> LAB
7/12/2008 00:07 846,920 piasm-book.pdf
7/12/2008 01:44 86 Setpath.bat
3/12/2008 14:12 716,512 winimage81.exe
5 File(s) 2,659,809 bytes
5 Dir(s) 70,934,261,760 bytes free

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

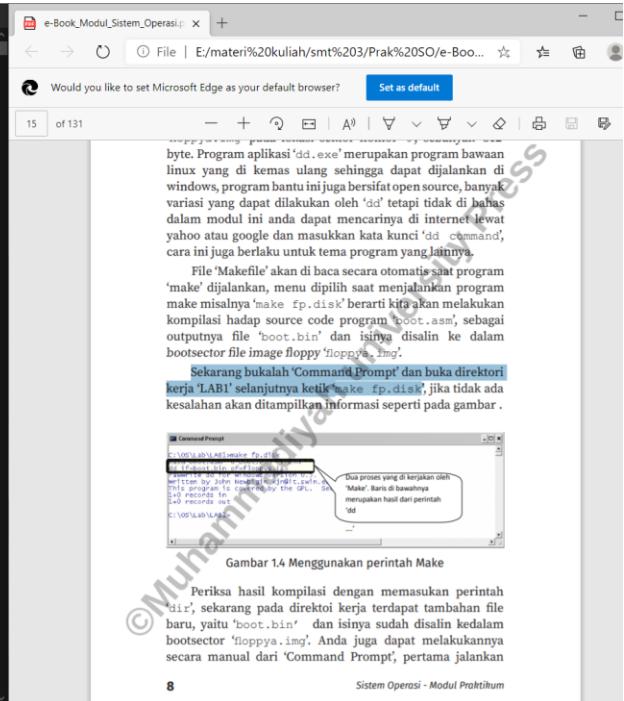
C:\OS\LAB>cd lab1

C:\OS\LAB\LAB1>notepad boot.asm

C:\OS\LAB\LAB1>notepad Makefile

C:\OS\LAB\LAB1>make fp.disk
nasm boot.asm -o boot.bin -f bin
dd if=boot.bin of=floppya.img
rawrite dd for windows version 0.5.
written by John Newbigin <jn@t.swin.edu.au>
This program is covered by the GPL. See copying.txt for details
40 records in
40 records out

C:\OS\LAB\LAB1>
```



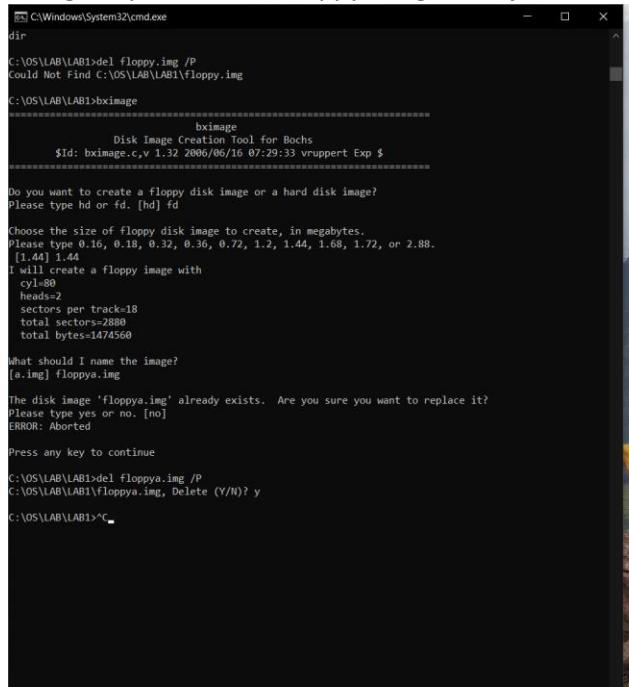
8

Sistem Operasi - Modul Praktikum

Gambar 1.4 Menggunakan perintah Make

Periksa hasil komplilasi dengan memasukan perintah ‘dir’, sekarang pada direktori kerja terdapat tambahan file baru, yaitu ‘boot.bin’ dan isinya sudah disalin kedalam bootsector ‘floppya.img’. Anda juga dapat melakukannya secara manual dari ‘Command Prompt’, pertama jalankan

9. Mengetik perintah ‘del floppya.img /P’ lanjutkan dengan tekan ‘Y’ dan <ENTER>



```
C:\Windows\System32\cmd.exe
del
C:\OS\LAB\LAB1>del floppya.img /P
Could Not Find C:\OS\LAB\LAB1\floppya.img

C:\OS\LAB\LAB1>bximage
=====
Disk Image Creation Tool for Bochs
$Id: bximage.c,v 1.32 2006/06/16 07:29:33 vruppert Exp $

Do you want to create a floppy disk image or a hard disk image?
Please type hd or fd. [hd] fd

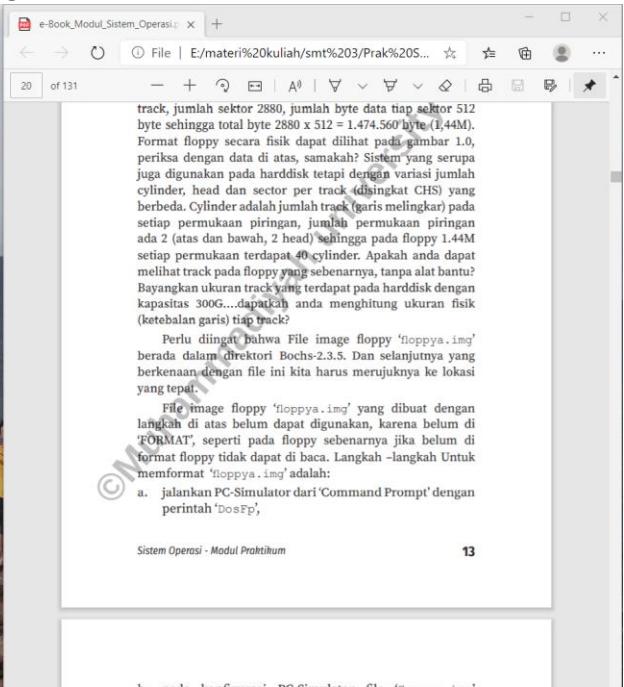
Choose the size of floppy disk image to create, in megabytes.
Please type 0.16, 0.18, 0.32, 0.36, 0.72, 1.2, 1.44, 1.68, 1.72, or 2.88.
[1.44] 1.44
I will create a floppy image with
  cyl=80
  heads=2
  sectors per track=18
  total sectors=2880
  total bytes=1474560

What should I name the image?
[a.img] floppya.img

The disk image 'floppya.img' already exists. Are you sure you want to replace it?
Please type yes or no. [no]
ERROR: Aborted

Press any key to continue

C:\OS\LAB\LAB1>del floppya.img /P
C:\OS\LAB\LAB1\floppya.img, Delete (Y/N)? y
C:\OS\LAB\LAB1>
```

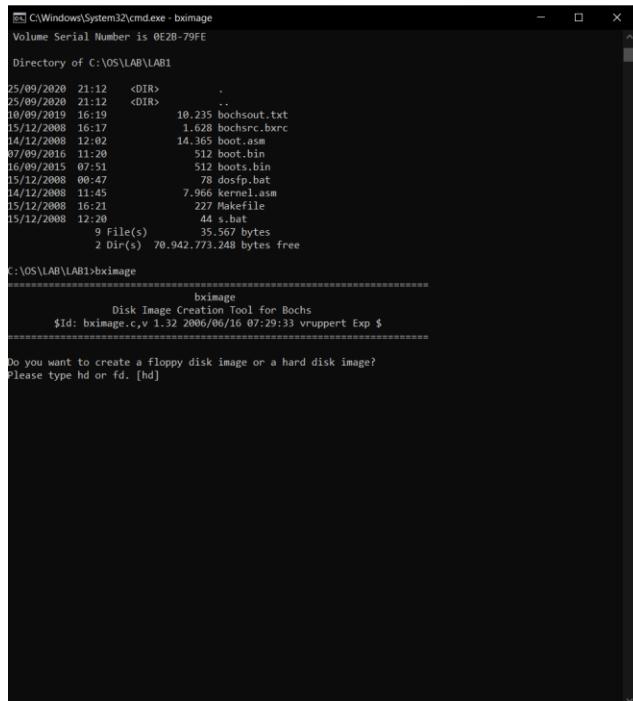


Sistem Operasi - Modul Praktikum

13

b. pada konfigurasi PC-Simulator file ‘floppya.img’
tersimpan pada ‘drive D’.

10. Mengetik perintah perintah 'dir'. Selanjutnya panggil 'bximage'



C:\Windows\System32\cmd.exe - bximage
Volume Serial Number is 0E2B-79FE
Directory of C:\OS\LAB\LAB1
25/09/2020 21:12 <DIR> .
25/09/2020 21:12 <DIR> ..
10/09/2019 16:19 10.235 bochsout.txt
15/12/2008 16:17 1.628 bochsrc.brc
14/12/2008 12:02 14.365 boot.asm
07/09/2016 11:20 512 boot.bin
16/09/2015 07:51 512 boots.bin
15/12/2008 00:47 78 dosfp.bat
14/12/2008 11:45 7.966 kernel.asm
15/12/2008 16:21 227 Makefile
15/12/2008 12:20 44 s.bat
9 File(s) 35.567 bytes
2 Dir(s) 70.942.773.248 bytes free
C:\OS\LAB\LAB1>bximage
=====
Disk Image Creation Tool for Bochs
\$Id: bximage.c,v 1.32 2006/06/16 07:29:33 vruppert Exp \$
=====
Do you want to create a floppy disk image or a hard disk image?
Please type hd or fd. [hd]

e-Book_Modul_Sistem_Operasi.pptx - 16 of 131

nama-nama file dipisahkan dengan 'spasi', anda dapat mencobanya dengan perintah 'make clean', periksa isi file dalam direktori kerja, sekarang 'boot.bin' sudah terhapus.

Mengenal 'BOOT DISK'

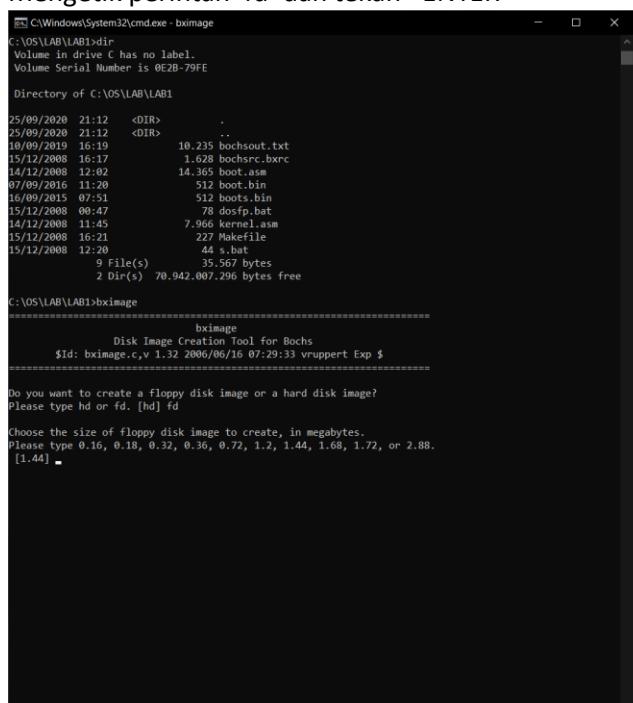
File 'floppya.img' adalah file image (bukan gambar), sebuah file yang isinya diformat seperti format floppy disk kapasitas 1.44M yang akan digunakan sebagai 'bootdisk' pada PC-Simulator, anda juga dapat memindahkan isi file image ini ke dalam floppy disk sebenarnya dengan menggunakan program bantu 'rawrite' atau yang lainnya. File image ini dapat di buat dengan menggunakan progam bantu bawaan 'Bochs' yaitu 'bximage' atau menggunakan program aplikasi 'Virtual PC' milik Windows (free juga) atau dengan program 'Magiciso'. Sekarang kita coba membuat file image floppy baru dengan menggunakan program aplikasi 'bximage.exe', lakukan urutan perintah berikut.

- a. Hapuslah file 'floppya.img' jika sudah ada pada direktori kerja anda, dari 'Command Prompt' (lakukan dari direktori kerja) ketik 'del floppya.img /f' lanjutkan dengan tekan 'Y' dan <ENTER>. Pastikan bahwa file sudah benar-benar terhapus dengan perintah 'dir'. Selanjutnya panggil 'bximage' sehingga ditampilkan window seperti pada gambar 1.5.

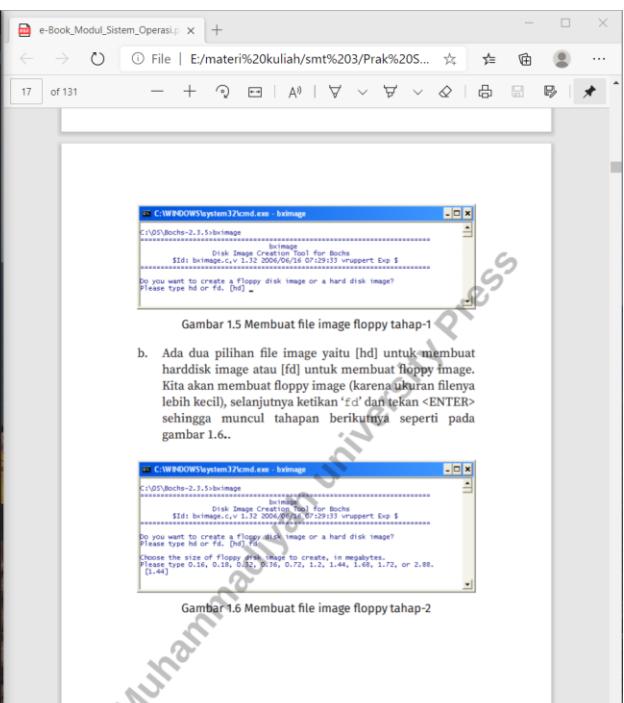
Sistem Operasi - Modul Praktikum

9

11. Mengetik perintah 'fd' dan tekan <ENTER>



C:\Windows\System32\cmd.exe - bximage
C:\OS\LAB\LAB1>dir
Volume in drive C has no label.
Volume Serial Number is 0E2B-79FE
Directory of C:\OS\LAB\LAB1
25/09/2020 21:12 <DIR> .
25/09/2020 21:12 <DIR> ..
10/09/2019 16:19 10.235 bochsout.txt
15/12/2008 16:17 1.628 bochsrc.brc
14/12/2008 12:02 14.365 boot.asm
07/09/2016 11:20 512 boot.bin
16/09/2015 07:51 512 boots.bin
15/12/2008 00:47 78 dosfp.bat
14/12/2008 11:45 7.966 kernel.asm
15/12/2008 16:21 227 Makefile
15/12/2008 12:20 44 s.bat
9 File(s) 35.567 bytes
2 Dir(s) 70.942.007.296 bytes free
C:\OS\LAB\LAB1>bximage
=====
Disk Image Creation Tool for Bochs
\$Id: bximage.c,v 1.32 2006/06/16 07:29:33 vruppert Exp \$
=====
Do you want to create a floppy disk image or a hard disk image?
Please type hd or fd. [hd]
Choose the size of floppy disk image to create, in megabytes.
Please type 0.16, 0.18, 0.32, 0.36, 0.72, 1.2, 1.44, 1.68, 1.72, or 2.88.
[1.44] -



C:\Windows\System32\cmd.exe - bximage
C:\OS\LAB\LAB1>bximage
=====
Disk Image Creation Tool for Bochs
\$Id: bximage.c,v 1.32 2006/06/16 07:29:33 vruppert Exp \$
=====
Do you want to create a floppy disk image or a hard disk image?
Please type hd or fd. [hd]
Choose the size of floppy disk image to create, in megabytes.
Please type 0.16, 0.18, 0.32, 0.36, 0.72, 1.2, 1.44, 1.68, 1.72, or 2.88.
[1.44]

e-Book_Modul_Sistem_Operasi.pptx - 17 of 131

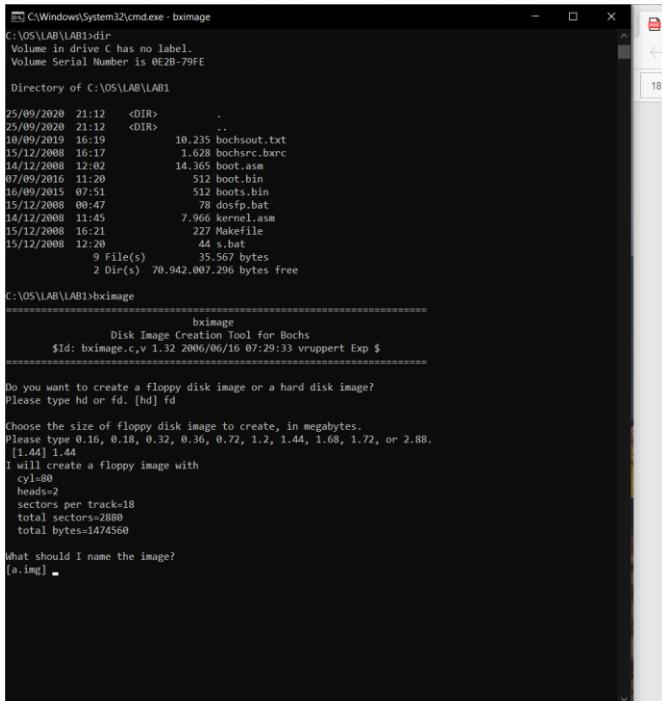
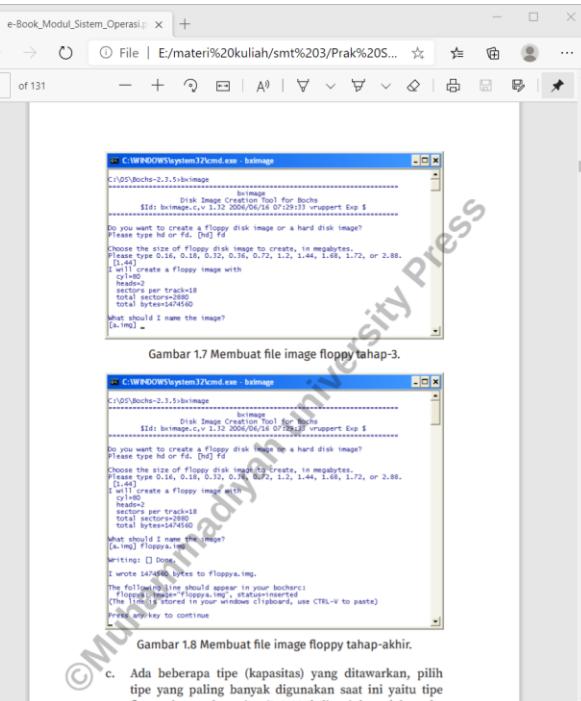
Gambar 1.5 Membuat file image floppy tahap-1

b. Ada dua pilihan file image yaitu [hd] untuk membuat harddisk image atau [fd] untuk membuat floppy image. Kita akan membuat floppy image (karena ukuran filenya lebih kecil), selanjutnya ketikan 'fd' dan tekan <ENTER> sehingga muncul tahapan berikutnya seperti pada gambar 1.6..

C:\Windows\System32\cmd.exe - bximage
C:\OS\LAB\LAB1>bximage
=====
Disk Image Creation Tool for Bochs
\$Id: bximage.c,v 1.32 2006/06/16 07:29:33 vruppert Exp \$
=====
Do you want to create a floppy disk image or a hard disk image?
Please type hd or fd. [hd]
Choose the size of floppy disk image to create, in megabytes.
Please type 0.16, 0.18, 0.32, 0.36, 0.72, 1.2, 1.44, 1.68, 1.72, or 2.88.
[1.44]

Gambar 1.6 Membuat file image floppy tahap-2

12. Memilih tipe yang paling banyak digunakan saat ini yaitu tipe floppy dengan kapasitas '1.44MB'

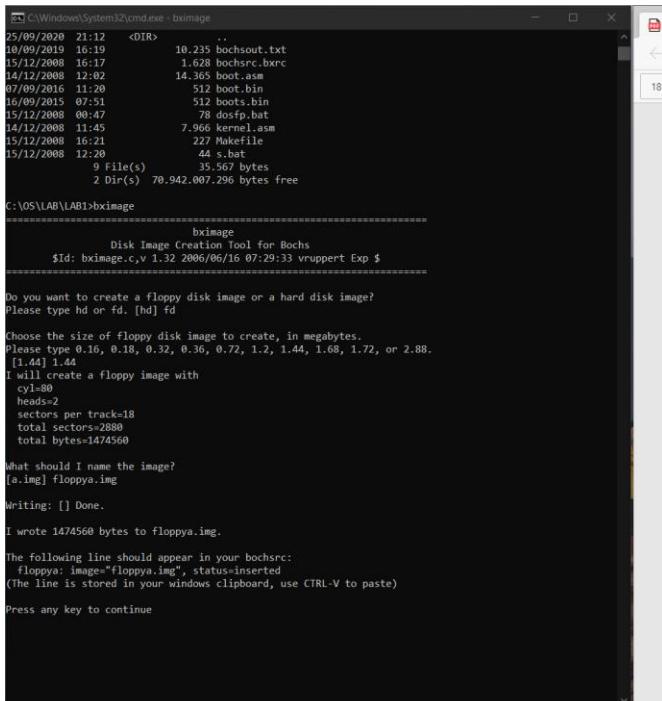
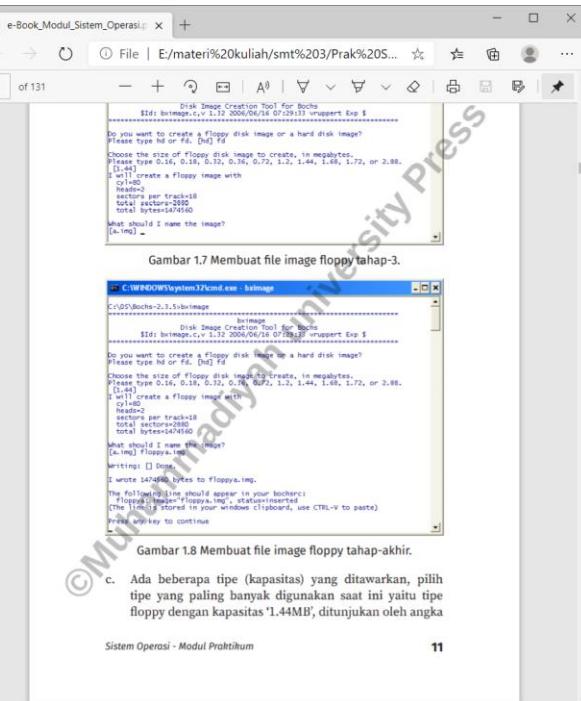



Gambar 1.7 Membuat file image floppy tahap-3.

Gambar 1.8 Membuat file image floppy tahap-akhir.

C. Ada beberapa tipe (kapasitas) yang ditawarkan, pilih tipe yang paling banyak digunakan saat ini yaitu tipe floppy dengan kapasitas '1.44MB', ditunjukkan oleh angka

13. Terakhir memberikan nama file, ketikan 'floppya.img' dan <ENTER>

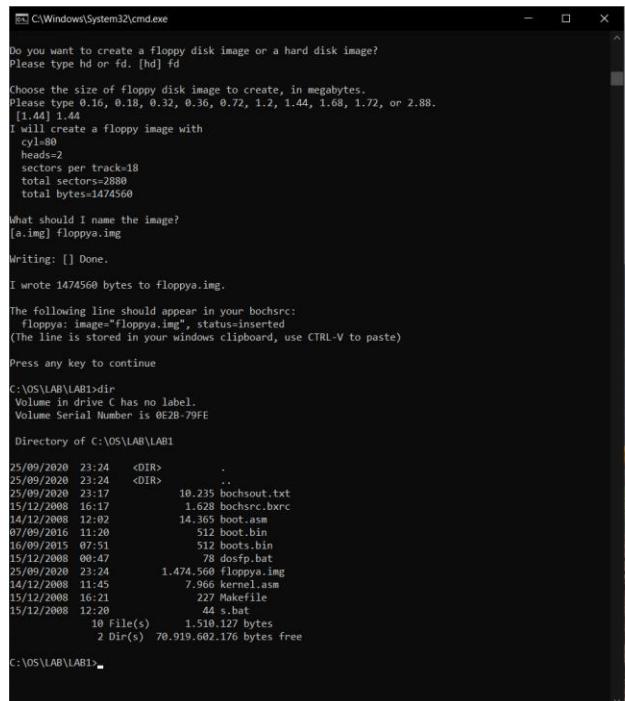



Gambar 1.7 Membuat file image floppy tahap-3.

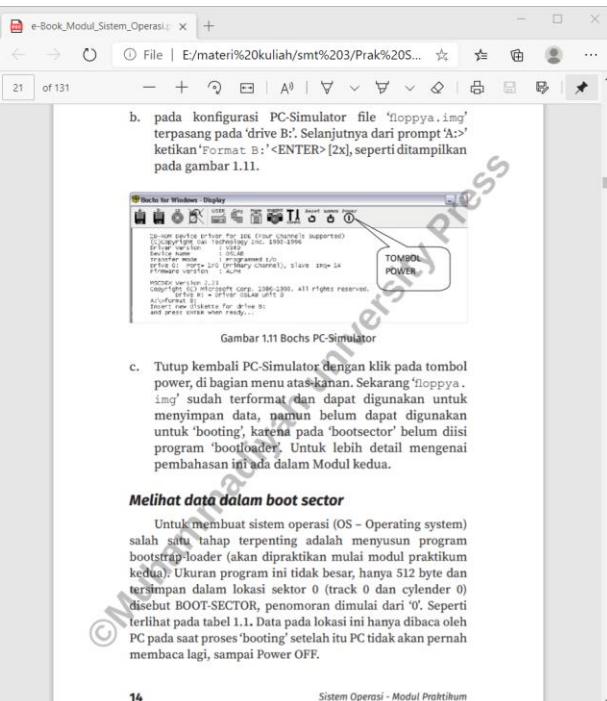
Gambar 1.8 Membuat file image floppy tahap-akhir.

C. Ada beberapa tipe (kapasitas) yang ditawarkan, pilih tipe yang paling banyak digunakan saat ini yaitu tipe floppy dengan kapasitas '1.44MB', ditunjukkan oleh angka

14. Memastikan keberadaan file image tersebut dengan perintah ‘dir’



C:\Windows\System32\cmd.exe
Do you want to create a floppy disk image or a hard disk image?
Please type hd or fd. [hd] fd
Choose the size of floppy disk image to create, in megabytes.
Please type 0.16, 0.18, 0.32, 0.36, 0.72, 1.2, 1.44, 1.68, 1.72, or 2.88.
[1.44] 1.44
I will create a floppy image with
cyl=80
heads=2
sectors per track=18
total sectors=2880
total bytes=1474560
What should I name the image?
[a.img] floppya.img
Writing: [] Done.
I wrote 1474560 bytes to floppya.img.
The following line should appear in your bochsrc:
floppya: image="floppya.img", status=inserted
(The line is stored in your windows clipboard, use CTRL-V to paste)
Press any key to continue
C:\OS\LAB\LAB1>dir
Volume in drive C has no label.
Volume Serial Number is 0E2B-79FE
Directory of C:\OS\LAB\LAB1
25/09/2020 23:24 <DIR> .
25/09/2020 23:24 <DIR> ..
25/09/2020 23:17 10,235 bochsout.txt
15/12/2008 16:17 1,628 bochsrc.brcx
14/12/2008 12:02 14,365 boot.asm
07/09/2016 11:20 512 boot.bin
16/09/2015 07:51 512 boots.bin
15/12/2008 08:47 78 dosfp.bat
25/09/2020 23:24 1,474,560 floppya.img
14/12/2008 11:45 7,966 kernel.asm
15/12/2008 16:21 227 Makefile
15/12/2008 12:26 44 s.bat
10 File(s) 1,510,127 bytes
2 Dir(s) 70,919,662,176 bytes free
C:\OS\LAB\LAB1>



e-Book_Modul_Sistem_Operasi... x +
21 of 131
b. pada konfigurasi PC-Simulator file 'floppya.img' terpasang pada 'drive B:'. Selanjutnya dari prompt 'A:>' ketikan 'Format B:<ENTER>' [2x], seperti ditampilkan pada gambar 1.11.

Bochs for Windows - Display
CD-ROM Device Driver for IDE (Four Channels Supported)
(C)Copyright Oak Technology Inc. 1993-1996
Driver Version : U340
Device Name : OSLAB
Transfer Mode : Programmed I/O
Drive 0: Ports: 1F0 (Primary Channel), Slave IRQ= 14
Firmware version : ALPH
MSCDEX Version 2.23
Copyright (C) Microsoft Corp. 1986-1993. All rights reserved.
Drive N: -> Driver OSLAB unit 0
R:>_

Gambar 1.11 Bochs PC-Simulator

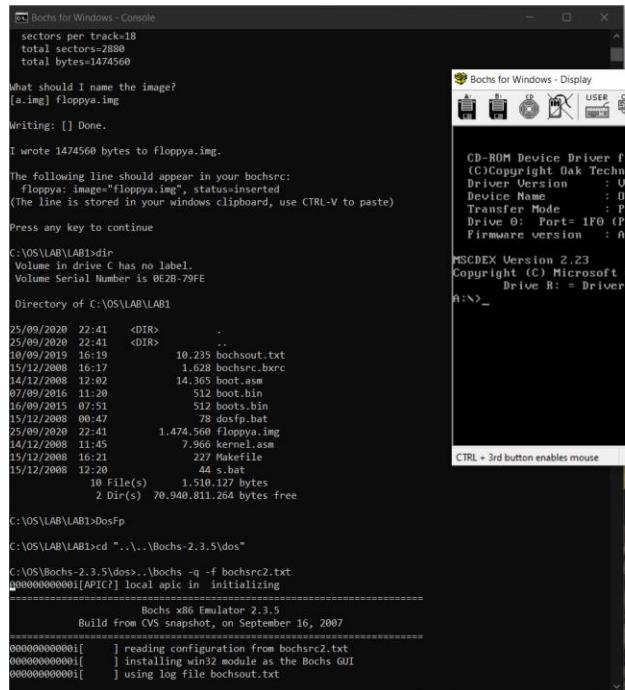
c. Tutup kembali PC-Simulator dengan klik pada tombol power, di bagian menu atas kanan. Sekarang 'floppya.img' sudah terformat dan dapat digunakan untuk menyimpan data, namun belum dapat digunakan untuk 'booting', karena pada 'bootsector' belum diisi program 'bootloader'. Untuk lebih detail mengenai pembahasan ini ada dalam Modul kedua.

Melihat data dalam boot sector

Untuk membuat sistem operasi (OS – Operating system) salah satu tahap terpenting adalah menyusun program bootstrap-loader (akan dipraktikkan mulai modul praktikum kedua). Ukuran program ini tidak besar, hanya 512 byte dan tersimpan dalam lokasi sektor 0 (track 0 dan cylinder 0) disebut BOOT-SECTOR, nomorannya dimulai dari '0'. Seperti terlihat pada tabel 1.1. Data pada lokasi ini hanya dibaca oleh PC pada saat proses 'booting' setelah itu PC tidak akan pernah membacanya lagi, sampai Power OFF.

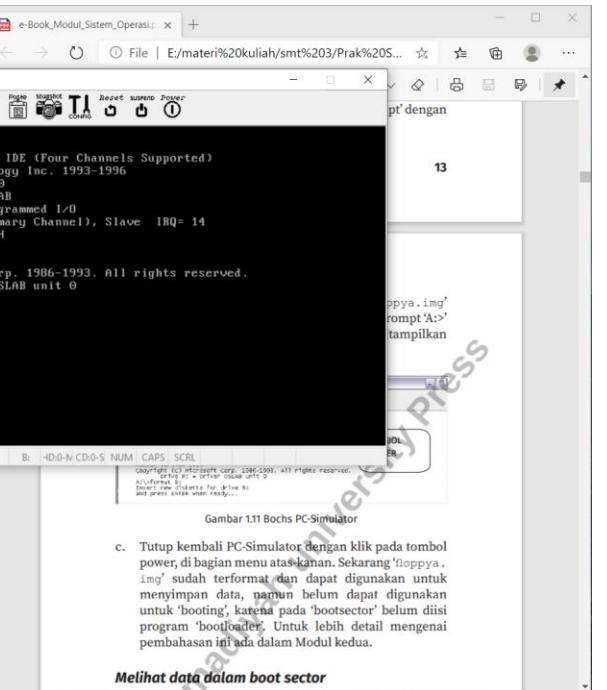
14 Sistem Operasi - Modul Praktikum

15. Menjalankan PC-Simulator dari ‘Command Prompt’ dengan perintah ‘DosFp’



Bochs for Windows - Console
sectors per track=18
total sectors=2880
total bytes=1474560
What should I name the image?
[a.img] floppya.img
Writing: [] Done.
I wrote 1474560 bytes to floppya.img.
The following line should appear in your bochsrc:
floppya: image="floppya.img", status=inserted
(The line is stored in your windows clipboard, use CTRL-V to paste)
Press any key to continue
C:\OS\LAB\LAB1>dir
Volume in drive C has no label.
Volume Serial Number is 0E2B-79FE
Directory of C:\OS\LAB\LAB1
25/09/2020 22:41 <DIR> .
25/09/2020 22:41 <DIR> ..
10/09/2019 16:19 10,235 bochsout.txt
15/12/2008 16:17 1,628 bochsrc.brcx
14/12/2008 12:02 14,365 boot.asm
07/09/2016 11:20 512 boot.bin
16/09/2015 07:51 512 boots.bin
15/12/2008 08:47 78 dosfp.bat
25/09/2020 22:41 1,474,560 floppya.img
14/12/2008 11:45 7,966 kernel.asm
15/12/2008 16:21 227 Makefile
15/12/2008 12:26 44 s.bat
10 File(s) 1,510,127 bytes
2 Dir(s) 70,940,811,264 bytes free
C:\OS\LAB\LAB1>DosFp
C:\OS\LAB\LAB1>cd "...\\Bochs-2.3.5\\dos"
C:\OS\Bochs-2.3.5\\dos>..\\bochs -q -f bochsrc2.txt
00000000000001[APIC] local apic in initializing
Bochs x86 Emulator 2.3.5
Build from CVS snapshot on September 16, 2007

00000000000001[1 reading configuration from bochsrc2.txt
00000000000001[1 installing win32 module as the Bochs GUI
00000000000001[1 using log file bochsout.txt



Bochs for Windows - Display
CD-ROM Device Driver for IDE (Four Channels Supported)
(C)Copyright Oak Technology Inc. 1993-1996
Driver Version : U340
Device Name : OSLAB
Transfer Mode : Programmed I/O
Drive 0: Ports: 1F0 (Primary Channel), Slave IRQ= 14
Firmware version : ALPH
MSCDEX Version 2.23
Copyright (C) Microsoft Corp. 1986-1993. All rights reserved.
Drive N: -> Driver OSLAB unit 0
R:>_

Gambar 1.11 Bochs PC-Simulator

c. Tutup kembali PC-Simulator dengan klik pada tombol power, di bagian menu atas kanan. Sekarang 'floppya.img' sudah terformat dan dapat digunakan untuk menyimpan data, namun belum dapat digunakan untuk 'booting', karena pada 'bootsector' belum diisi program 'bootloader'. Untuk lebih detail mengenai pembahasan ini ada dalam Modul kedua.

Melihat data dalam boot sector

16. Selanjutnya dari prompt 'A:>' ketikan 'Format B:' [2x]

```
C:\ Bochs for Windows - Console
>rawrite dd for windows version 0.5.
Written by John Newbegin <jn@it.swin.edu.au>
This program is covered by the GPL. See copying.txt for details
Error opening input file: 2 The system cannot find the file specified

C:\OS\LAB\LAB1>DosP

C:\OS\LAB\LAB1>cd ..\..\Bochs-2.3.5\dos"
C:\OS\Bochs-2.3.5\dos>..>bochs -q -f bochsrc2.txt
00000000000000000000000000000000[ APIC ] local apic in initializing
=====
Bochs X86 Emulator 2.3.5
Build from CVS snapshot, on September 16, 2007
=====
000000000000[ ] reading configuration from bochsrc2.txt
000000000000[ ] installing win32 module as the Bochs GUI
000000000000[ ] using log file bochout.txt
# In bx_win32_gui.c:exit(void)
Bochs is exiting with the following message:
[NGUI ] Window closed, exiting!
=====
Bochs is exiting. Press ENTER when you're ready to close this window.
dir
C:\OS\Bochs-2.3.5\dos>d "C:\os\lab\lab1"
C:\OS\LAB\LAB1>DosP

C:\OS\LAB\LAB1>cd ..\..\Bochs-2.3.5\dos"
C:\OS\Bochs-2.3.5\dos>..>bochs -q -f bochsrc2.txt
00000000000000000000000000000000[ APIC ] local apic in initializing
=====
Bochs X86 Emulator 2.3.5
Build from CVS snapshot, on September 16, 2007
=====
000000000000[ ] reading configuration from bochsrc2.txt
000000000000[ ] installing win32 module as the Bochs GUI
000000000000[ ] using log file bochout.txt

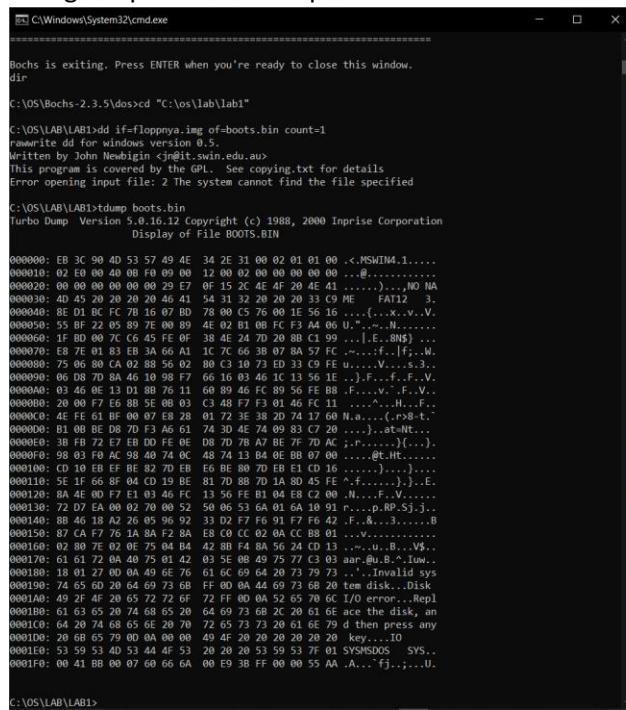
e-book_Modul_Sistem_Operasi x + Bochs for Windows - Display
File | E:/materi%20kuliah/smt%203/Prak%205... Star
Drive 0: Port: IDE (Primary Channel), Slave IRQ=14
Firmware version : ALPH
MSCDEX Version 2.23
Copyright (C) Microsoft Corp. 1986-1993. All rights reserved.
Drive B: = Driver OSLAB unit 0
A:>>Format B:
Insert new diskette for drive B:
and press ENTER when ready...
Checking existing disk format.
Formatting 1.44M
Format complete.

Volume label (11 characters, ENTER for none)?
1,457,664 bytes total disk space
1,457,664 bytes available on disk
512 bytes in each allocation unit.
2,847 allocation units available on disk.
Volume Serial Number is 352C-1E1F
Format another (Y/N)?
```

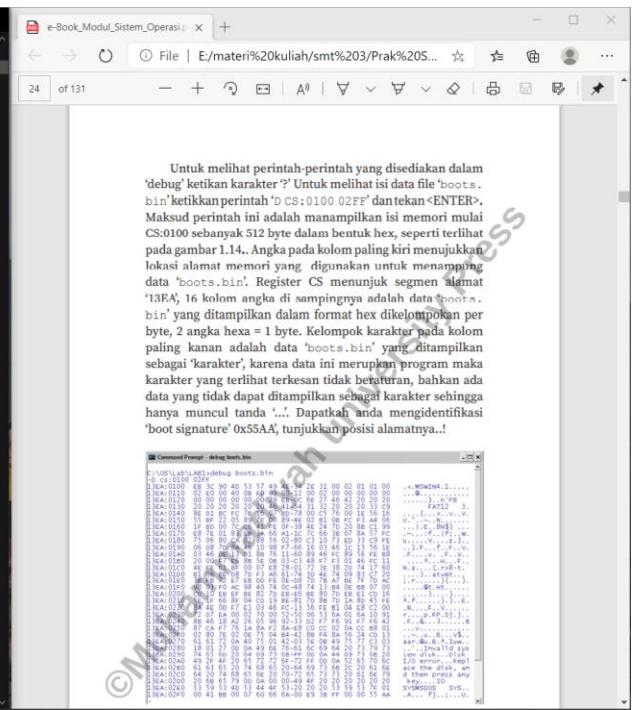
17. Menutup kembali PC-Simulator dengan klik pada tombol power, lalu copy 512 byte data bootsektor ke dalam sebuah file terpisah, caranya dari ‘Command Prompt’ ketik ‘dd if=floppya.img of=boots.bin count=1’

The screenshot shows a Windows desktop with two windows open. The left window is a Command Prompt titled 'cmd.exe' showing Bochs configuration and boot process. The right window is a browser titled 'e-book_Modul_Sistem_Operasi' displaying a page about memory mapping. A tooltip from the browser is overlaid on the desktop, pointing to the word 'debug'. The tooltip text reads: 'Maksud perintah ini adalah memindah data dalam file "boots.bin" ke dalam memori kerja "debug" mulai dari alamat '0000:0100'. Debug hanya dapat bekerja dengan ukuran data/file maksimum sebesar 64KB, jadi dengan debug kita tidak bisa membuat file image floppy yang berukuran 1,4MB. file "boots.bin" berukuran 512 B (atau 0x01FF) dan oleh debug data ini akan diletakkan di memori mulai dari alamat CS:0100 sampai dengan alamat CS:02FF. Simbol CS merupakan simbol register "CODE SEGMENT", nilai CS tergantung pada kondisi memori komputer saat itu.'

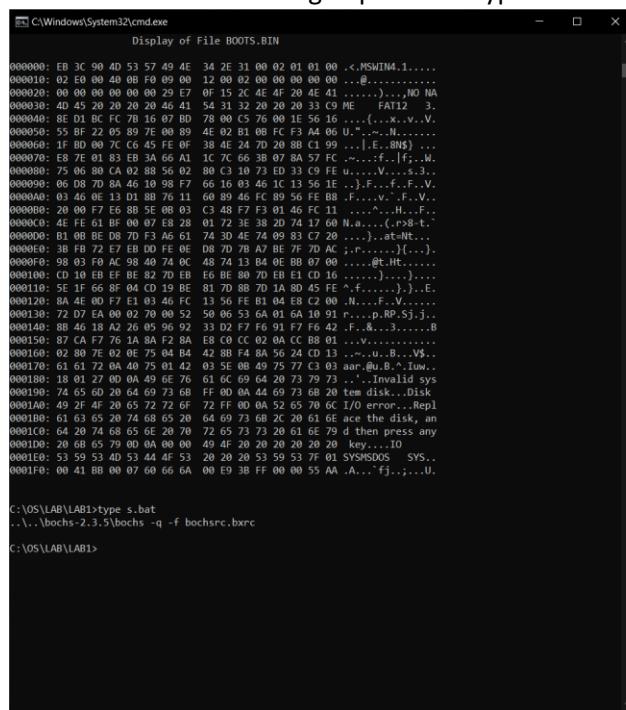
18. Mengetik perintah ‘tdump boots. bin’<ENTER>



Untuk melihat perintah-perintah yang disediakan dalam 'debug' ketikan karakter '?' Untuk melihat isi data file 'boots. bin' ketikan perintah 'D CS:0100 02FF' dan tekan <ENTER>. Maksud perintah ini adalah manampilkan isi memori mulai CS:0100 sebanyak 512 byte dalam bentuk hex, seperti terlihat pada gambar 1.14. Angka pada kolom paling kiri menunjukkan lokasi alamat memori yang digunakan untuk menampung data 'boots. bin'. Register CS menunjuk segmen alamat '13FA', 16 kolom angka di sampingnya adalah data 'boots. bin' yang ditampilkan dalam format hex dikelompokan per byte, 2 angka hex = 1 byte. Kelompok karakter pada kolom paling kanan adalah data 'boots. bin' yang ditampilkan sebagai 'karakter', karena data ini merupakan program maka karakter yang terlihat terkesan tidak benar, bahkan ada data yang tidak dapat ditampilkan sebagai karakter sehingga hanya muncul tanda '..'. Dapatkah anda mengidentifikasi 'boot signature' 0x55AA, tunjukkan posisi alamattanya..!



19. Melihat isi file ‘s.bat’ dengan perintah ‘type s.bat’<ENTER>

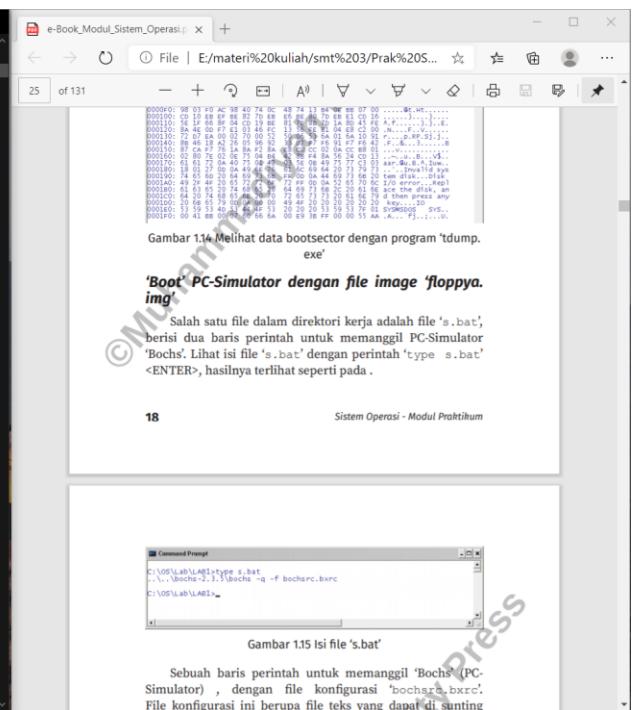


Gambar 1.14 Melihat data bootsector dengan program 'tdump. exe'

'Boot' PC-Simulator dengan file image 'floppya. img'

Salah satu file dalam direktori kerja adalah file 's.bat', berisi dua baris perintah untuk memanggil PC-Simulator 'Bochs'. Lihat isi file 's.bat' dengan perintah 'type s.bat' <ENTER>, hasilnya terlihat seperti pada .

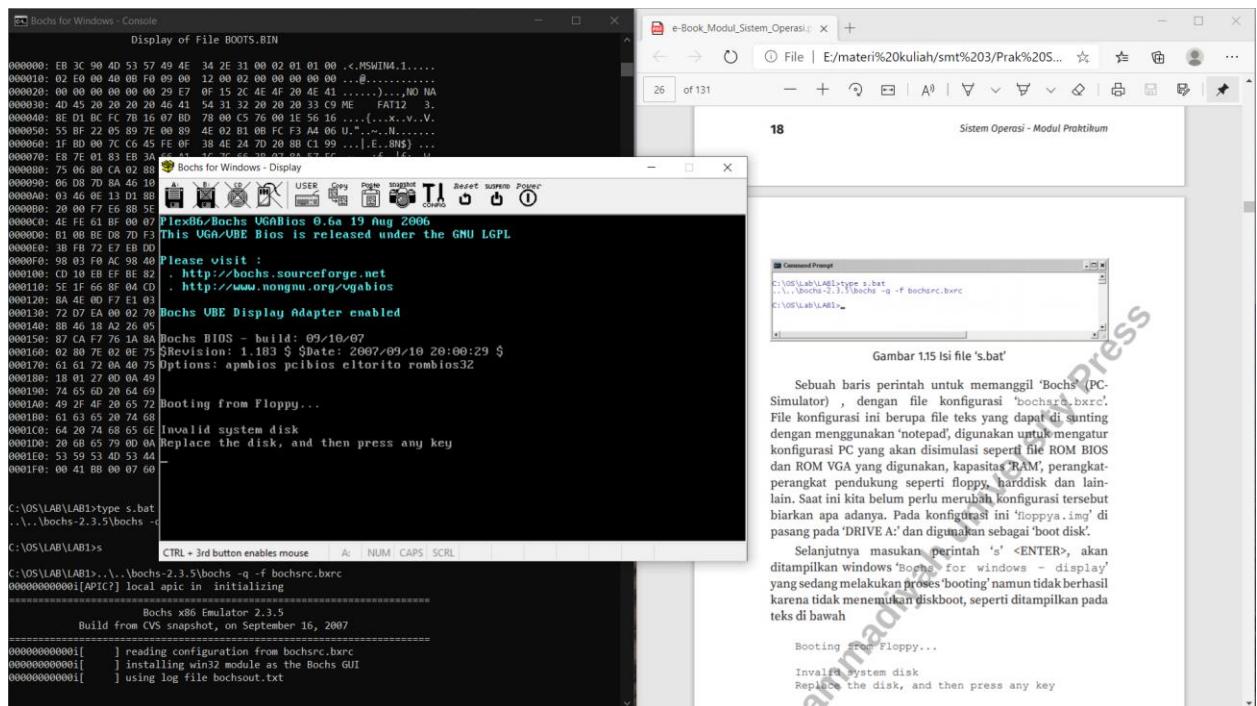
18 Sistem Operasi - Modul Praktikum



Gambar 1.15 Isi file 's.bat'

Sebuah baris perintah untuk memanggil 'Bochs' (PC-Simulator), dengan file konfigurasi 'bochsrc.bxrc'. File konfigurasi ini berupa file teks yang dapat di suntung

20. Selanjutnya masukan perintah 's' , akan ditampilkan windows 'Bochs for windows – display' yang sedang melakukan proses 'booting' namun tidak berhasil karena tidak menemukan diskboot



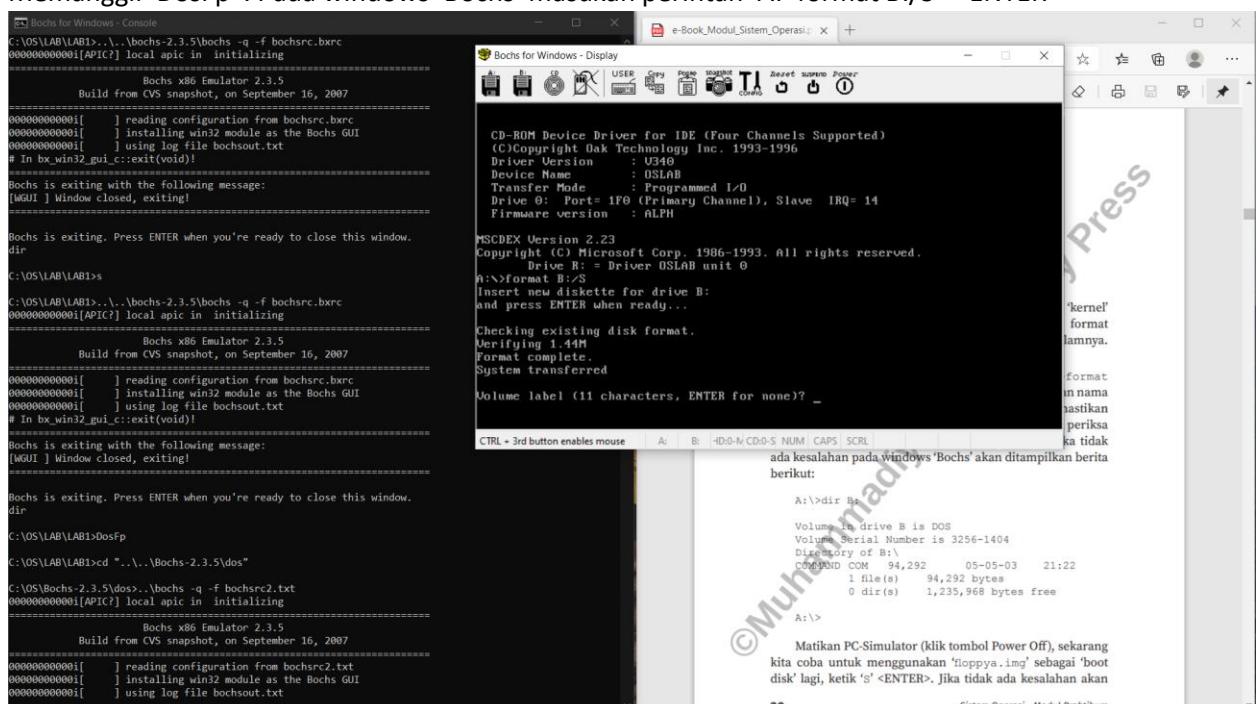
Gambar 1.15 Isi file 's.bat'

Sebuah baris perintah untuk memanggil 'Bochs' (PC-Simulator) , dengan file konfigurasi 'bochsrc.bxrc'. File konfigurasi ini berupa file teks yang dapat di sunting dengan menggunakan 'notepad', digunakan untuk mengatur konfigurasi PC yang akan dimulai seperti file ROM BIOS dan ROM VGA yang digunakan, kapasitas RAM, perangkat-perangkat pendukung seperti floppy, harddisk dan lain-lain. Saat ini kita belum perlu merubah konfigurasi tersebut biarkan apa adanya. Pada konfigurasi ini 'floppya.img' di pasang pada 'DRIVE A:' dan digunakan sebagai 'boot disk'.

Selanjutnya masukan perintah 's' <ENTER>, akan ditampilkan windows 'Bochs for windows – display' yang sedang melakukan proses 'booting' namun tidak berhasil karena tidak menemukan diskboot, seperti ditampilkan pada teks di bawah

```
Booting from Floppy...
Invalid System disk
Replace the disk, and then press any key
```

21. Memanggil 'DosFp' . Pada windows 'Bochs' masukan perintah 'A:>format B:/S' <ENTER>



ada kesalahan pada windows 'Bochs' akan ditampilkan berita berikut:

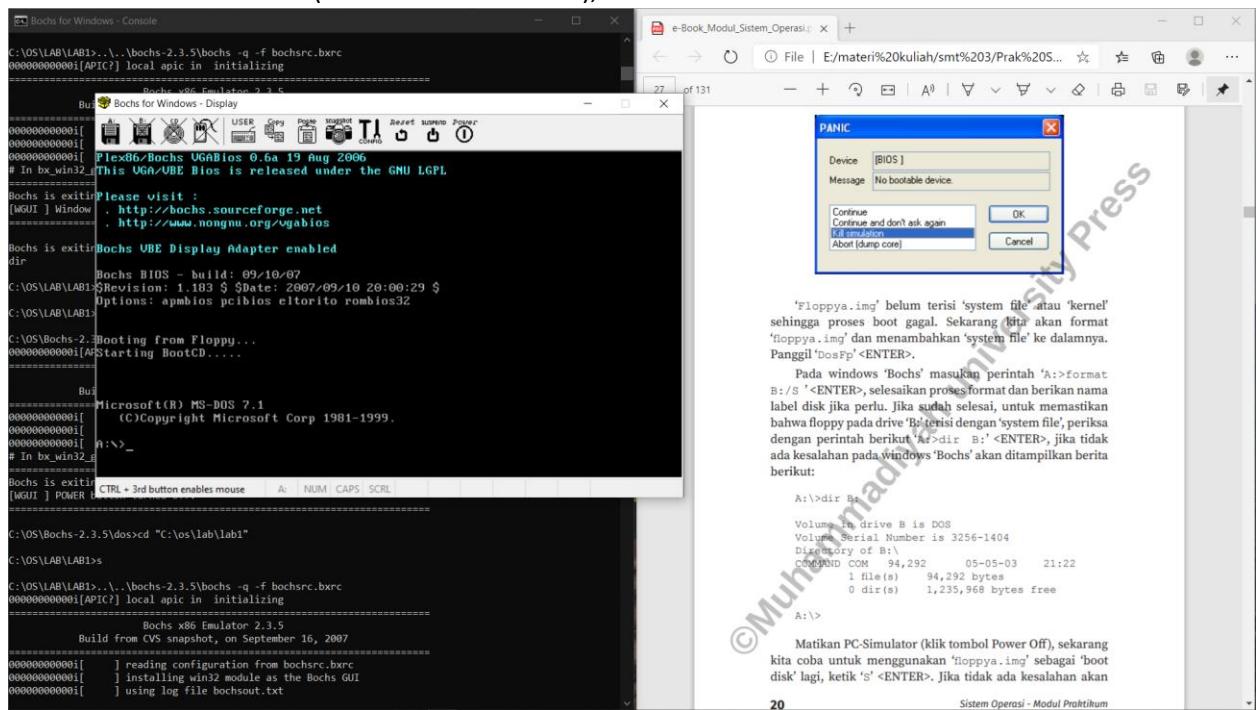
```
A:>dir B:
Volume in drive B is DOS
Volume Serial Number is 3256-1404
Directory of B:\

COMPAQ COM   94,292   05-05-03  21:22
 1 file(s)    94,292 bytes
 0 dir(s)   1,235,968 bytes free
```

Matiakan PC-Simulator (klik tombol Power Off), sekarang kita coba untuk menggunakan 'floppya.img' sebagai 'boot disk' lagi, ketik 's' <ENTER>. Jika tidak ada kesalahan akan

Sistem Operasi - Modul Praktikum

22. Mematikan PC-Simulator (klik tombol Power Off), ketik 'S'<ENTER>



TUGAS HAL 21

1. Apa yang dimaksud dengan kode ‘ASCII’, buatlah tabel kode ASCII lengkap cukup kode ASCII yang standar tidak perlu extended, tuliskan kode ASCII dalam format angka desimal, binary dan hexadesimal serta karakter dan simbol yang dikodekan.

ASCII (American Standard Code for Information Interchange) merupakan Kode Standar Amerika untuk Pertukaran Informasi atau sebuah standar internasional dalam pengkodean huruf dan simbol seperti Unicode dan Hex tetapi ASCII lebih bersifat universal.

Kode ASCII

Binary	Oct	Dec	Hex	Glyph	Binary	Oct	Dec	Hex	Glyph	Binary	Oct	Dec	Hex	Glyph
010 0000	040	32	20	□	100 0000	100	64	40	@	110 0000	140	96	60	`
010 0001	041	33	21	!	100 0001	101	65	41	A	110 0001	141	97	61	a
010 0010	042	34	22	"	100 0010	102	66	42	B	110 0010	142	98	62	b
010 0011	043	35	23	#	100 0011	103	67	43	C	110 0011	143	99	63	c
010 0100	044	36	24	\$	100 0100	104	68	44	D	110 0100	144	100	64	d
010 0101	045	37	25	%	100 0101	105	69	45	E	110 0101	145	101	65	e
010 0110	046	38	26	&	100 0110	106	70	46	F	110 0110	146	102	66	f
010 0111	047	39	27	'	100 0111	107	71	47	G	110 0111	147	103	67	g
010 1000	050	40	28	(100 1000	110	72	48	H	110 1000	150	104	68	h
010 1001	051	41	29)	100 1001	111	73	49	I	110 1001	151	105	69	i
010 1010	052	42	2A	*	100 1010	112	74	4A	J	110 1010	152	106	6A	j
010 1011	053	43	2B	+	100 1011	113	75	4B	K	110 1011	153	107	6B	k
010 1100	054	44	2C	,	100 1100	114	76	4C	L	110 1100	154	108	6C	l
010 1101	055	45	2D	-	100 1101	115	77	4D	M	110 1101	155	109	6D	m
010 1110	056	46	2E	.	100 1110	116	78	4E	N	110 1110	156	110	6E	n
010 1111	057	47	2F	/	100 1111	117	79	4F	O	110 1111	157	111	6F	o
011 0000	060	48	30	0	101 0000	120	80	50	P	111 0000	160	112	70	p
011 0001	061	49	31	1	101 0001	121	81	51	Q	111 0001	161	113	71	q
011 0010	062	50	32	2	101 0010	122	82	52	R	111 0010	162	114	72	r
011 0011	063	51	33	3	101 0011	123	83	53	S	111 0011	163	115	73	s
011 0100	064	52	34	4	101 0100	124	84	54	T	111 0100	164	116	74	t
011 0101	065	53	35	5	101 0101	125	85	55	U	111 0101	165	117	75	u
011 0110	066	54	36	6	101 0110	126	86	56	V	111 0110	166	118	76	v
011 0111	067	55	37	7	101 0111	127	87	57	W	111 0111	167	119	77	w
011 1000	070	56	38	8	101 1000	130	88	58	X	111 1000	170	120	78	x
011 1001	071	57	39	9	101 1001	131	89	59	Y	111 1001	171	121	79	y
011 1010	072	58	3A	:	101 1010	132	90	5A	Z	111 1010	172	122	7A	z
011 1011	073	59	3B	;	101 1011	133	91	5B	[111 1011	173	123	7B	{

011 1100	074	60	3C	<	101 1100	134	92	5C	\	111 1100	174	124	7C	
011 1101	075	61	3D	=	101 1101	135	93	5D]	111 1101	175	125	7D	}
011 1110	076	62	3E	>	101 1110	136	94	5E	^	111 1110	176	126	7E	-
011 1111	077	63	3F	?	101 1111	137	95	5F	_					

2. Carilah daftar perintah bahasa assembly untuk mesin intel keluarga x86 lengkap (dari buku referensi atau internet). Daftar perintah ini dapat digunakan sebagai pedoman untuk memahami program ‘boot.asm’ dan ‘kernel.asm’.

Terbagi menjadi 3 bagian utama yaitu :

a. **Komentar**

Komentar diawali dengan tanda titik koma (;).

; ini adalah komentar

b. **Label**

Label diakhiri dengan tanda titik dua (:).

Contoh: main: ,loop: ,proses: ,keluar:

c. **Assembler directives**

Directives adalah perintah yang ditujukan kepada assembler ketika sedang menerjemahkan program kita ke bahasa mesin.

Directive dimulai dengan tanda titik. **.model** : memberitahu assembler berapa memori yang akan dipakai oleh program kita.

Ada model tiny, model small, model compact, model medium, model large, dan model huge.

.data : memberitahu assembler bahwa bagian di bawah ini adalah data program.

.code : memberitahu assembler bahwa bagian di bawah ini adalah instruksi program.

.stack : memberitahu assembler bahwa program kita memiliki stack.

Program EXE harus punya stack. Kira-kira yang penting itu dulu.

Semua directive yang dikenal assembler adalah: .186 .286 .286c .286p .287 .386 .386c .386p .387 .486 .486p .8086 .8087

.alpha .break .code .const .continue .cref .data .data? .dosseg .else .elseif .endif .endw .err .err1 .err2 .errb

.errdef .errdif .errdifi .erre .erridn .erridni .errnb .errndef .errnz .exit .fardata .fardata? .if .lall .lfcond .list .listall .listif .listmacro

.listmacroall .model .no87 .nocref .nolist .nolistif .nolistmacro .radix .repeat .sall .seq .sfcond .stack

.startup .tfcond .type .until .untilcxz .while .xall .xref .xlist.

Definisi data

DB : define bytes. Membentuk data byte demi byte. Data bisa data numerik maupun teks.

catatan: untuk membentuk data string, pada akhir string harus diakhiri tanda dolar (\$).

sintaks: {label} DB {data} contoh: teks1 db "Hello world \$" **DW** : define words.

Membentuk data word demi word (1 word = 2 byte).

sintaks: {label} DW {data} contoh: kucing dw ?, ?, ? ;mendefinisikan tiga slot 16-bit yang isinya don't care

(disimbolkan dengan tanda tanya)

DD : define double words. Membentuk data doubleword demi doubleword (4 byte).

sintaks: {label} DD {data} **EQU** : equals. Membentuk konstanta. sintaks: {label} EQU {data}

contoh: sepuluh EQU 10

Ada assembly yang melibatkan bilangan pecahan (floating point), bilangan bulat (integer), DF (define far words),

DQ (define quad words), dan DT (define ten bytes).

Perpindahan data

MOV : move. Memindahkan suatu nilai dari register ke memori, memori ke register, atau register ke register.

sintaks: MOV {tujuan}, {sumber}

contoh:

mov AX, 4C00h ;mengisi register AX dengan 4C00(hex).

mov BX, AX ;menyalin isi AX ke BX. mov CL, [BX] ;mengisi register CL dengan data di memori yang alamatnya ditunjuk BX.

mov CL, [BX] + 2 ;mengisi CL dengan data di memori yang alamatnya ditunjuk BX lalu geser maju 2 byte.

mov [BX], AX ;menyimpan nilai AX pada tempat di memori yang ditunjuk BX. mov [BX] – 1, 00101110b

;menyimpan 00101110(bin) pada alamat yang ditunjuk BX lalu geser mundur 1 byte.

LEA : load effective address. Mengisi suatu register dengan alamat offset sebuah data.

sintaks: LEA {register}, {sumber} contoh: lea DX, teks1 **XCHG** : exchange. Menukar dua buah register langsung.

sintaks: XCHG {register 1}, {register 2} Kedua register harus punya ukuran yang sama.

Bila sama-sama 8 bit (misalnya AH dengan BL) atau sama-sama 16 bit (misalnya CX dan DX), maka pertukaran bisa dilakukan. Sebenarnya masih banyak perintah perpindahan data, misalnya IN, OUT, LODS, LODSB, LODSW, MOVS, MOVS, LDS, LES, LAHF, SAHF, dan XLAT.

Operasi logika

- a. **AND** melakukan bitwise and. sintaks: AND {register}, {angka} AND {register 1}, {register 2} hasil disimpan di register 1.
contoh: mov AL, 00001011b mov AH, 11001000b and AL, AH ;sekarang AL berisi 00001000(bin), sedangkan AH tidak berubah.
 - b. **OR** : melakukan bitwise or. sintaks: OR {register}, {angka} OR {register 1}, {register 2} hasil disimpan di register 1.
 - c. **NOT** : melakukan bitwise not (*one's complement*) sintaks: NOT {register} hasil disimpan di register itu sendiri.
 - d. **XOR** : melakukan bitwise eksklusif or. sintaks: XOR {register}, {angka} XOR {register 1}, {register 2} hasil disimpan di register 1. Tips: sebuah register yang di-XOR-kan dengan dirinya sendiri akan menjadi berisi nol.
 - e. **SHL** : shift left. Menggeser bit ke kiri. Bit paling kanan diisi nol. sintaks: SHL {register}, {banyaknya}
 - f. **SHR** : shift right. Menggeser bit ke kanan. Bit paling kiri diisi nol. sintaks: SHR {register}, {banyaknya}
 - g. **ROL** : rotate left. Memutar bit ke kiri. Bit paling kiri jadi paling kanan kali ini. sintaks: ROL {register}, {banyaknya} Bila banyaknya rotasi tidak disebutkan, maka nilai yang ada di CL akan digunakan sebagai banyaknya rotasi.
 - h. **ROR** : rotate right. Memutar bit ke kanan. Bit paling kanan jadi paling kiri. sintaks: ROR {register}, {banyaknya} Bila banyaknya rotasi tidak disebutkan, maka nilai yang ada di CL akan digunakan sebagai banyaknya rotasi.
- Ada lagi : RCL dan RCR.

Operasi matematika

- a. **ADD** : add. Menjumlahkan dua buah register.
sintaks: ADD {tujuan}, {sumber} operasi yang terjadi: tujuan = tujuan + sumber.
carry (bila ada) disimpan di CF.
- b. **ADC** : add with carry. Menjumlahkan dua register dan carry flag (CF).
sintaks: ADC {tujuan}, {sumber} operasi yang terjadi: tujuan = tujuan + sumber + CF.
carry (bila ada lagi) disimpan lagi di CF.
- c. **INC** : increment. Menjumlah isi sebuah register dengan 1.
Bedanya dengan ADD, perintah INC hanya memakan 1 byte memori sedangkan ADD pakai 3 byte.sintaks: INC {register}
- d. **SUB** : subtract. Mengurangkan dua buah register.
sintaks: SUB {tujuan}. {sumber} operasi yang terjadi: tujuan = tujuan – sumber.
borrow (bila terjadi) menyebabkan CF bernilai 1.

- e. **SBB** : subtract with borrow. Mengurangkan dua register dan carry flag (CF).
sintaks: SBB {tujuan}, {sumber} operasi yang terjadi: tujuan = tujuan – sumber – CF.
borrow (bila terjadi lagi) menyebabkan CF dan SF (sign flag) bernilai 1.
- f. **DEC** : decrement. Mengurang isi sebuah register dengan 1.
Jika SUB memakai 3 byte memori, DEC hanya memakai 1 byte. sintaks: DEC {register}
- g. **MUL** : multiply. Mengalikan register dengan AX atau AH.
sintaks: MUL {sumber} Bila register sumber adalah 8 bit,
maka isi register itu dikali dengan isi AL, kemudian disimpan di AX.
Bila register sumber adalah 16 bit, maka isi register itu dikali dengan isi AX,
kemudian hasilnya disimpan di DX:AX. Maksudnya, DX berisi high order byte-nya, AX berisi low
order byte-nya.
- h. **IMUL** : signed multiply. Sama dengan MUL,
hanya saja IMUL menganggap bit-bit yang ada di register sumber sudah dalam bentuk *two's complement*.
sintaks: IMUL {sumber}
- i. **DIV** : divide. Membagi AX atau DX:AX dengan sebuah register.
sintaks: DIV {sumber} Bila register sumber adalah 8 bit (misalnya: BL), maka operasi yang terjadi:
-AX dibagi BL,
-hasil bagi disimpan di AL, -sisa bagi disimpan di AH.
Bila register sumber adalah 16 bit (misalnya: CX), maka operasi yang terjadi: -DX:AX dibagi CX, -
hasil bagi disimpan di AX, -sisa bagi disimpan di DX.
- j. **IDIV** : signed divide. Sama dengan DIV, hanya saja IDIV menganggap bit-bit yang ada di register
sumber sudah dalam bentuk *two's complement*.
sintaks: IDIV {sumber}
- k. **NEG** : negate. Membuat isi register menjadi negatif (*two's complement*).
Bila mau *one's complement*, gunakan perintah NOT. sintaks: NEG {register} hasil disimpan di
register itu sendiri.

Pengulangan

- a. **LOOP** : loop. Mengulang sebuah proses. Pertama register CX dikurangi satu.
Bila CX sama dengan nol, maka looping berhenti. Bila tidak nol, maka lompat ke label tujuan.
sintaks: LOOP {label tujuan} Tips: isi CX dengan nol untuk mendapat jumlah pengulangan
terbanyak.Karena nol dikurang satu sama dengan -1, atau dalam notasi *two's complement* menjadi FFFF(hex) yang sama dengan 65535(dec).
- b. **LOOPE** : loop while equal. Melakukan pengulangan selama CX ≠ 0 dan ZF = 1. CX tetap
dikurangi 1 sebelum diperiksa.
sintaks: LOOP {label tujuan}
- c. **LOOPZ** : loop while zero. Identik dengan LOOPE.

- d. **LOOPNE** : loop while not equal.
Melakukan pengulangan selama CX $\neq 0$ dan ZF = 0. CX tetap dikurangi 1 sebelum diperiksa.
sintaks: LOOPNE {label tujuan}
- e. **LOOPNZ** : loop while not zero. Identik dengan LOOPNE.
- f. **REP** : repeat. Mengulang perintah sebanyak CX kali. sintaks: REP {perintah assembly} contoh:
mov CX, 05 rep inc BX ;register BX ditambah 1 sebanyak 5x.
- g. **REPE** : repeat while equal. Mengulang perintah sebanyak CX kali, tetapi pengulangan segera dihentikan bila didapati ZF = 1.
sintaks: REPE {perintah assembly}
- h. **REPZ** : repeat while zero. Identik dengan REPE.
- i. **REPNE** : repeat while not equal. Mengulang perintah sebanyak CX kali, tetapi pengulangan segera dihentikan bila didapati ZF = 0.
sintaks: REPNE {perintah assembly}
- j. **REPNZ** : repeat while not zero. Identik dengan REPNE.

Perbandingan

CMP : compare. Membandingkan dua buah operand. Hasilnya mempengaruhi sejumlah flag register.
sintaks: CMP {operand 1}, {operand 2}. Operand ini bisa register dengan register , register dengan isi memori, atau register dengan angka.

CMP tidak bisa membandingkan isi memori dengan isi memori. Hasilnya adalah:

Kasus	Bila operand 1 < operand 2	Bila operand 1 = operand 2	Bila operand 1 > operand 2
Signed binary	OF = 1, SF = 1, ZF = 0	OF = 0, SF = 0, ZF = 1	OF = 0, SF = 0, ZF = 0
Unsigned binary	CF = 1, ZF = 0	CF = 0, ZF = 1	CF = 0, ZF = 0

Lompat-lompat

JMP: jump. Lompat tanpa syarat. Lompat begitu saja. sintaks: JMP {label tujuan}

Lompat bersyarat sintaksnya sama dengan JMP, yaitu perintah jump diikuti label tujuan.

PERINTAH	ARTI	SYARAT	KASUS	KETERANGAN ("OP" = OPERAND)	MENGIKUTI CMP?
JA	jump if above	CF = 0 \wedge ZF = 0	unsigned	lompat bila op 1 > op 2	ya
JNBE	jump if not below or equal	= 0			
JB	jump if below	CF = 1 \wedge ZF = 0	unsigned	lompat bila op 1 < op 2	ya
JNAE	jump if not above or equal	= 0			
JAE	jump if above or equal	CF = 0 \vee ZF = 0	unsigned	lompat bila op 1 \geq op 2	ya
JNB	jump if not below	= 1			
JBE	jump if below or equal	CF = 1 \vee ZF = 0	unsigned	lompat bila op 1 \leq op 2	ya
JNA	jump if not above	= 1			
JG	jump if greater	OF = 0 \wedge ZF = 0	signed	lompat bila op 1 > op 2	ya
JNLE	jump if not less or equal	= 0			

JGE	jump if greater or equal	OF = 0 v ZF = 1	signed	lompat bila op 1 \geq op 2	ya
JNL	jump if not less than	= 1			
JL	jump if less than	OF = 1 \wedge ZF = 0	signed	lompat bila op 1 $<$ op 2	ya
JNGE	jump if not greater or equal	= 0			
JLE	jump if less or equal	OF = 1 v ZF = 1	signed	lompat bila op 1 \leq op 2	ya
JNG	jump if not greater	= 1			
JE	jump if equal	ZF = 1	keduanya	lompat bila op 1 = op 2	ya
JZ	jump if zero	ZF = 1	keduanya	lompat bila op 1 = op 2	ya
JNE	jump if not equal	ZF = 0	keduanya	lompat bila op 1 \neq op 2	ya
JNZ	jump if not zero	ZF = 0	keduanya	lompat bila op 1 \neq op 2	ya
JC	jump if carry	CF = 1	N/A	lompat bila carry flag = 1	tidak
JNC	jump if not carry	CF = 0	N/A	lompat bila carry flag = 0	tidak
JP	jump on parity	PF = 1	N/A	lompat bila parity flag = 1	tidak selalu
JPE	jump on parity even			lompat bila bilangan genap	
JNP	jump on not parity	PF = 0	N/A	lompat bila parity flag = 0	tidak selalu
JPO	jump on parity odd			lompat bila bilangan ganjil	
JO	jump if overflow	OF = 1	N/A	lompat bila overflow flag = 1	tidak
JNO	jump if not overflow	OF = 0	N/A	lompat bila overflow flag = 0	tidak
JS	jump if sign	SF = 1	N/A	lompat bila bilangan negatif	tidak
JCXZ	jump if CX is zero	CX = 0000	N/A	lompat bila CX berisi nol	tidak

Operasi stack

a. PUSH : push. Menambahkan sesuatu ke stack.

Sesuatu ini harus register berukuran 16 bit (pada 386+ harus 32 bit), tidak boleh angka, tidak boleh alamat memori.

Maka Anda tidak bisa mem-push register 8-bit seperti AH, AL, BH, BL, dan kawan-kawannya.

sintaks: push {register 16-bit sumber}

contoh: push DX push AX Setelah operasi push, register SP (stack pointer) otomatis dikurangi 2 (karena datanya 2 byte).

Makanya, "top" dari stack seakan-akan "tumbuh turun".

b. POP : pop. Mengambil sesuatu dari stack.

Sesuatu ini akan disimpan di register tujuan dan harus 16-bit. Maka Anda tidak bisa mem-pop menuju AH, AL, dkk.

sintaks: POP {register 16-bit tujuan}

contoh: POP BX Setelah operasi pop, register SP otomatis ditambah 2 (karena 2 byte), sehingga "top" dari stack "naik" lagi.

Tip: karena register segmen tidak bisa diisi langsung nilainya, Anda bisa menggunakan stack sebagai perantaranya.

Contoh kodennya: mov AX, seg teks1 push AX pop DS

- c. **PUSHF** : push flags. Mem-push **semua** isi register flag ke dalam stack.
Biasa dipakai untuk membackup data di register flag sebelum operasi matematika. Sintaks: PUSHF ;(saja).
- d. **POPF** : pop flags. Lawan dari pushf. Sintaks: POPF ;(saja).
- e. **POPA** : pop all general-purpose registers.
Adalah ringkasan dari sejumlah perintah dengan urutan:
pop DI pop SI pop BP pop SP pop BX pop DX pop CX pop AX
Urutan sudah ditetapkan seperti itu.
sintaks: POPA ;(saja). Jauh lebih cepat mengetikkan POPA daripada mengetik POP-POP-POP yang banyak itu.
- f. **PUSHA** : push all general-purpose registers. Lawan dari POPA,
dimana PUSHA adalah singkatan dari sejumlah perintah dengan urutan yang sudah ditetapkan:
push AX push CX push DX push BX push SP push BP push SI push DI

Operasi pada register flag

- a. **CLC** : clear carry flag. Menjadikan CF = 0. Sintaks: CLC ;(saja).
- b. **STC** : set carry flag. Menjadikan CF = 1. Sintaks: STC ;(saja).
- c. **CMC** : complement carry flag. Melakukan operasi NOT pada CF. Yang tadinya 0 menjadi 1, dan sebaliknya.
- d. **CLD** : clear direction flag. Menjadikan DF = 0. Sintaks: CLD ;(saja).
- e. **STD** : set direction flag. Menjadikan DF = 1.
- f. **CLI** : clear interrupt flag. Menjadikan IF = 0, sehingga interrupt ke CPU akan di-disable.
Biasanya perintah CLI diberikan sebelum menjalankan sebuah proses penting yang riskan gagal bila diganggu.
- g. **STI** : set interrupt flag. Menjadikan IF = 1.
Perintah lainnya
- h. **ORG** : origin. Mengatur awal dari program (bagian static data).
Analoginya seperti mengatur dimana letak titik (0, 0) pada koordinat Cartesius.
sintaks: ORG {alamat awal}
Pada program COM (program yang berekstensi .com), harus ditulis "ORG 100h" untuk mengatur alamat mulai dari program pada 0100(hex),
karena dari alamat 0000(hex) sampai 00FF(hex) sudah dipesan oleh sistem operasi (DOS).
- i. **INT** : interrupt. Menginterupsi prosesor.
Prosesor akan:
 1. Membackup data registernya saat itu,
 2. Menghentikan apa yang sedang dikerjakannya,
 3. Melompat ke bagian interrupt-handler (entah dimana kita tidak tahu, sudah ditentukan BIOS dan DOS),
 4. Melakukan interupsi,

5. Mengembalikan data registernya,
 6. Meneruskan pekerjaan yang tadi ditunda.
- sintaks: INT {nomor interupsi}

j. **IRET** : interrupt-handler return.

Kita bisa membuat interrupt-handler sendiri dengan berbagai cara.

Perintah IRET adalah perintah yang menandakan bahwa interrupt-handler kita selesai, dan prosesor boleh melanjutkan pekerjaan yang tadi tertunda.

k. **CALL** : call procedure. Memanggil sebuah prosedur.

sintaks: CALL {label nama prosedur}

l. **RET** : return. Tanda selesai prosedur.

Setiap prosedur harus memiliki RET di ujungnya.

sintaks: RET ;(saja)

m. **HLT** : halt. Membuat prosesor menjadi tidak aktif.

Prosesor harus mendapat interupsi dari luar atau di-reset supaya aktif kembali.

n. **Jadi, jangan gunakan perintah HLT untuk mengakhiri program!!**

Sintaks: HLT ;(saja). **NOP** : no operation.

Perintah ini memakan 1 byte di memori tetapi tidak menyuruh prosesor melakukan apa-apa selama 3 clock prosesor.

Berikut contoh potongan program untuk melakukan *delay* selama 0,1 detik pada prosesor Intel 80386 yang berkecepatan 16 MHz.

mov ECX, 533333334d ;ini adalah bilangan desimal idle: nop loop idle

SUMBER

<https://oferiachacha.blogspot.com/2012/11/bahasa-assembly-untuk-x86.html>

<https://itinternals.blogspot.com/2011/12/ascii-printable-characters.html>