

**PRAKTIKUM SISTEM OPERASI  
PENGENALAN SISTEM PENGEMBANGAN OS DENGAN PC  
SIMULATOR 'BOCHS'**



**DISUSUN OLEH :  
MUHAMMAD SOFIYAN HADI  
L200190199**

**INFORMATIKA  
FAKULTAS KOMUNIKASI DAN INFORMATIKA  
UNIVERSITAS MUHAMMADIYAH SURAKARTA  
2020/2021**

## A. Menuju ke direktori kerja.

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

```
Command Prompt
Microsoft Windows [Version 10.0.19041.508]
(c) 2020 Microsoft Corporation. All rights reserved.

C:\Users\Sofiyana>cd \os
```

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

```
Command Prompt
Microsoft Windows [Version 10.0.19041.508]
(c) 2020 Microsoft Corporation. All rights reserved.

C:\Users\Sofiyana>cd \os

C:\OS>dir
Volume in drive C is WINDOWS
Volume Serial Number is 3CF2-5B73

Directory of C:\OS

23/09/2020  10:14    <DIR>          .
23/09/2020  10:14    <DIR>          ..
23/09/2020  10:14    <DIR>          Bochs-2.3.5
23/09/2020  10:14    <DIR>          Dev-Cpp
17/12/2008  00:08             1.096.291 i386.pdf
23/09/2020  10:14    <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 winima81.exe
               4 File(s)                2.659.809 bytes
               5 Dir(s)  31.270.125.568 bytes free
```

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

```
Command Prompt
Microsoft Windows [Version 10.0.19041.508]
(c) 2020 Microsoft Corporation. All rights reserved.

C:\Users\Sofiyana>cd \os

C:\OS>dir
Volume in drive C is WINDOWS
Volume Serial Number is 3CF2-5B73

Directory of C:\OS

23/09/2020  10:14    <DIR>          .
23/09/2020  10:14    <DIR>          ..
23/09/2020  10:14    <DIR>          Bochs-2.3.5
23/09/2020  10:14    <DIR>          Dev-Cpp
17/12/2008  00:08             1.096.291 i386.pdf
23/09/2020  10:14    <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 winima81.exe
               4 File(s)                2.659.809 bytes
               5 Dir(s)  31.270.125.568 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>
```

Untuk melihat script yang terdapat di dalam file 'setpath.bat' dapat digunakan perintah 'type setpath.bat'.

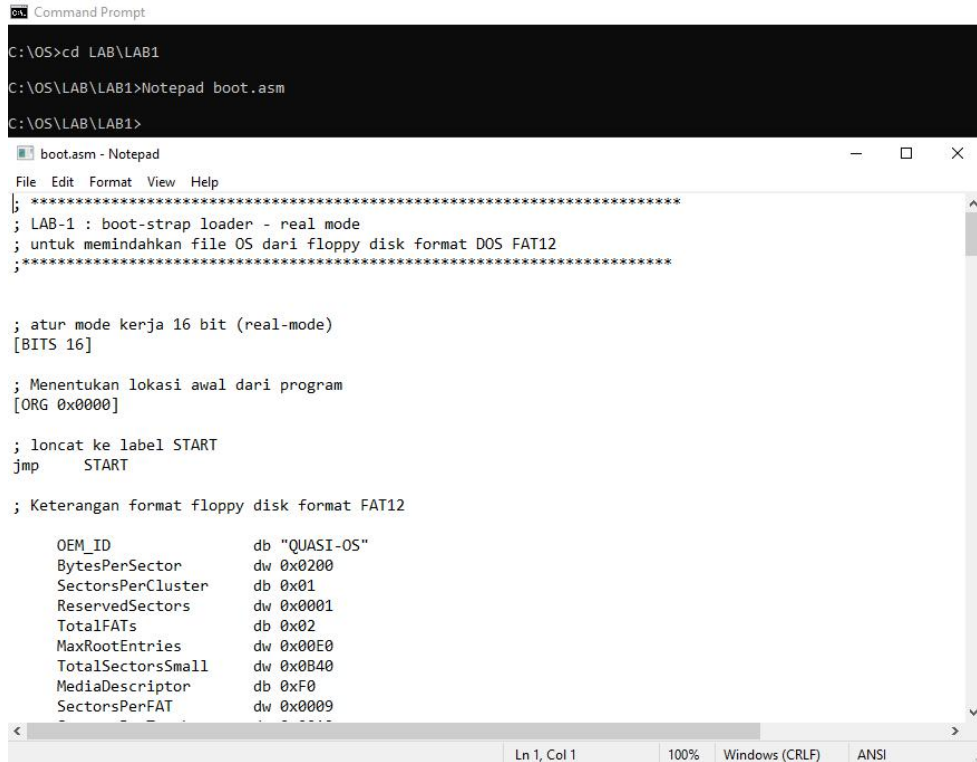
## B. Melihat isi direktori kerja.

- Jalankan command prompt.
- Masuk ke direktori kerja pada 'C:\OS\LAB\LAB1'.



```
cmd Command Prompt
C:\OS>cd LAB\LAB1
```

- Cobalah untuk membuka file tersebut dengan perintah berikut, dari 'COMMAND PROMPT', ketikkan 'Notepad boot.asm' dan tekan <ENTER>.



```
cmd Command Prompt
C:\OS>cd LAB\LAB1
C:\OS\LAB\LAB1>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]

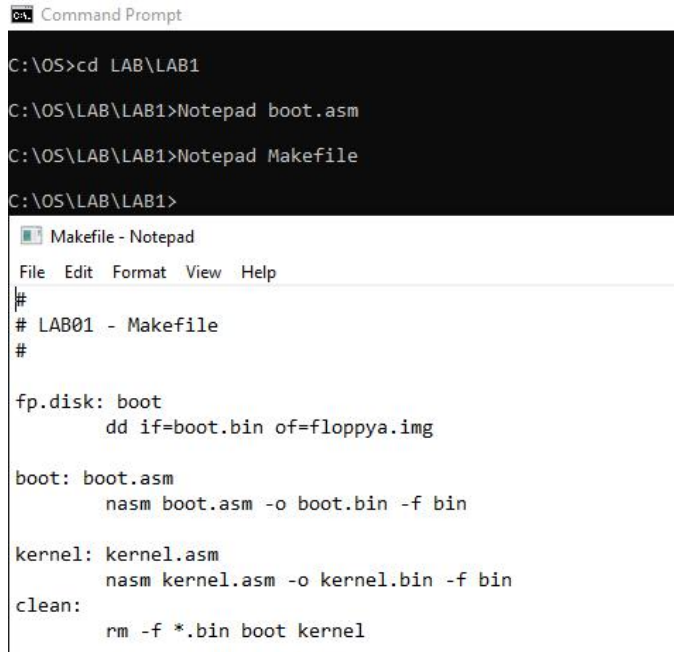
; loncat ke label START
jmp     START

; Keterangan format floppy disk format FAT12

    OEM_ID             db "QUASI-OS"
    BytesPerSector     dw 0x0200
    SectorsPerCluster  db 0x01
    ReservedSectors    dw 0x0001
    TotalFATs          db 0x02
    MaxRootEntries     dw 0x00E0
    TotalSectorsSmall  dw 0x0B40
    MediaDescriptor    db 0xF0
    SectorsPerFAT      dw 0x0009
```

### C. Sekilas tentang Makefile

- a. Membuka file 'Makefile' lewat cmd dengan cara masuk ke direktori 'C:\OS\LAB\LAB1' selanjutnya ketik 'Notepad M' tekan tombol 'TAB' sehingga muncul 'Notepad Makefile' dan tekan <ENTER>.



```
Command Prompt
C:\OS>cd LAB\LAB1
C:\OS\LAB\LAB1>Notepad boot.asm
C:\OS\LAB\LAB1>Notepad Makefile
C:\OS\LAB\LAB1>

Makefile - Notepad
File Edit Format View Help
#
# LAB01 - Makefile
#

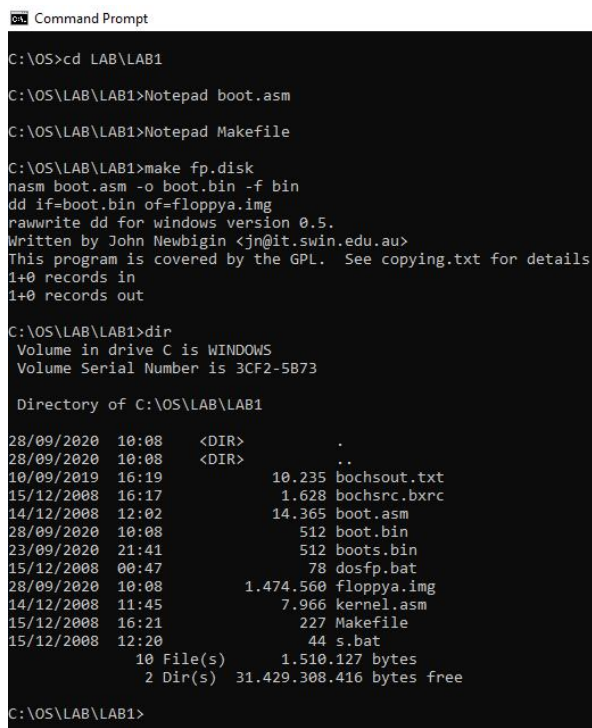
fp.disk: boot
    dd if=boot.bin of=floppya.img

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
```

- b. Bukalah 'Command Prompt' dan buka direktori kerja 'LAB1' selanjutnya ketik 'make fp.disk'. Periksa hasil kompilasi dengan memasukkan perintah 'dir', sekarang pada direktori kerja terdapat tambahan file baru, yaitu 'boot.bin' dan isinya sudah disalin ke dalam bootsector 'floppya.img'.



```
Command Prompt
C:\OS>cd LAB\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
rawwrite dd for windows version 0.5.
Written by John Newbigin <jn@it.swin.edu.au>
This program is covered by the GPL. See copying.txt for details
1+0 records in
1+0 records out

C:\OS\LAB\LAB1>dir
Volume in drive C is WINDOWS
Volume Serial Number is 3CF2-5B73

Directory of C:\OS\LAB\LAB1

28/09/2020 10:08 <DIR> .
28/09/2020 10:08 <DIR> ..
10/09/2019 16:19      10.235 bochsout.txt
15/12/2008 16:17      1.628 bochsrc.bxrc
14/12/2008 12:02     14.365 boot.asm
28/09/2020 10:08      512 boot.bin
23/09/2020 21:41      512 boots.bin
15/12/2008 00:47       78 dosfp.bat
28/09/2020 10:08    1.474.560 floppya.img
14/12/2008 11:45     7.966 kernel.asm
15/12/2008 16:21      227 Makefile
15/12/2008 12:20       44 s.bat
                10 File(s)    1.510.127 bytes
                2 Dir(s)    31.429.308.416 bytes free

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

#### D. Mengenal '*BOOT DISK*'

a. Hapuslah file 'floppya.img' jika sudah ada pada direktori kerja anda, dari 'Command Prompt' (lakukan dari direktori kerja) ketik 'del floppya.img /P' lanjutkan dengan tekan 'Y' dan <ENTER>. Pastikan bahwa file udah benar- benar terhapus dengan perintah 'dir'.

```
CA: Command Prompt
C:\OS\LAB\LAB1>del floppya.img /P
C:\OS\LAB\LAB1>dir
Volume in drive C is WINDOWS
Volume Serial Number is 3CF2-5B73

Directory of C:\OS\LAB\LAB1

28/09/2020  10:46    <DIR>          .
28/09/2020  10:46    <DIR>          ..
10/09/2019  16:19             10.235 bochsout.txt
15/12/2008  16:17             1.628 bochsrc.bxrc
14/12/2008  12:02             14.365 boot.asm
28/09/2020  10:08              512 boot.bin
23/09/2020  21:41              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)  31.428.902.912 bytes free

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

Selanjutnya panggil 'bximage' sehingga ditampilkan window seperti pada gambar.

```
CA: Command Prompt - bximage
C:\OS\LAB\LAB1>bximage
=====
                        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]
```

b. Kita akan membuat floppy image (karena ukuran filenya lebih kecil), selanjutnya ketikan 'fd' dan tekan <ENTER>.

```
CA: Command Prompt - bximage
C:\OS\LAB\LAB1>bximage
=====
                        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] _
```



c. Pilih tipe floppy dengan kapasitas '1.44MB', ditunjukkan oleh angka [1.44], selanjutnya tekan <ENTER>.

```
ca. Command Prompt - bxiimage
C:\OS\LAB\LAB1>bxiimage
=====
                        bxiimage
                Disk Image Creation Tool for Bochs
                $Id: bxiimage.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]
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]
```

d. Berikan nama file, ketikan 'floppya.img' dan <ENTER>.

```
ca. Command Prompt - bxiimage
C:\OS\LAB\LAB1>bxiimage
=====
                        bxiimage
                Disk Image Creation Tool for Bochs
                $Id: bxiimage.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]
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.

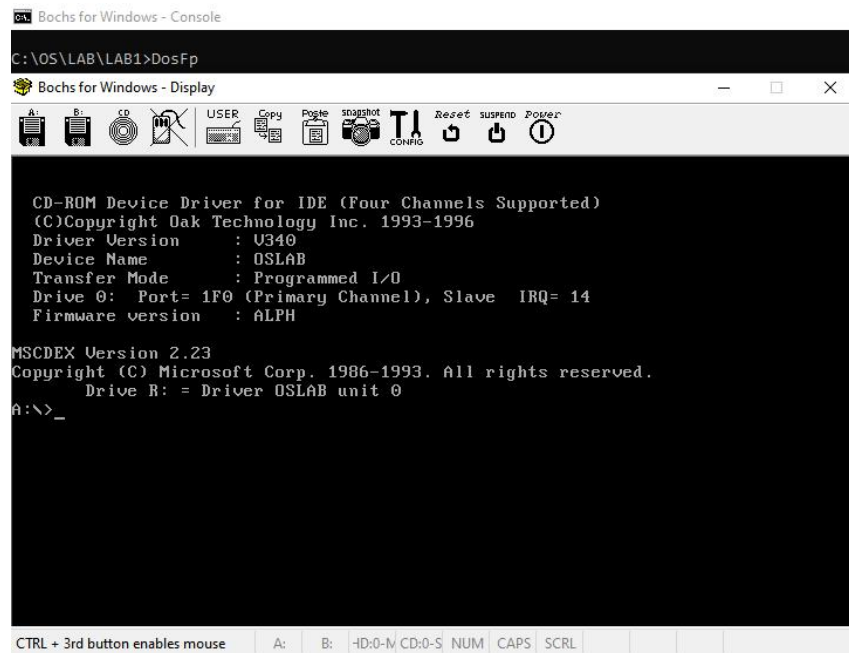
I wrote 1474560 bytes to floppya.img.

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)

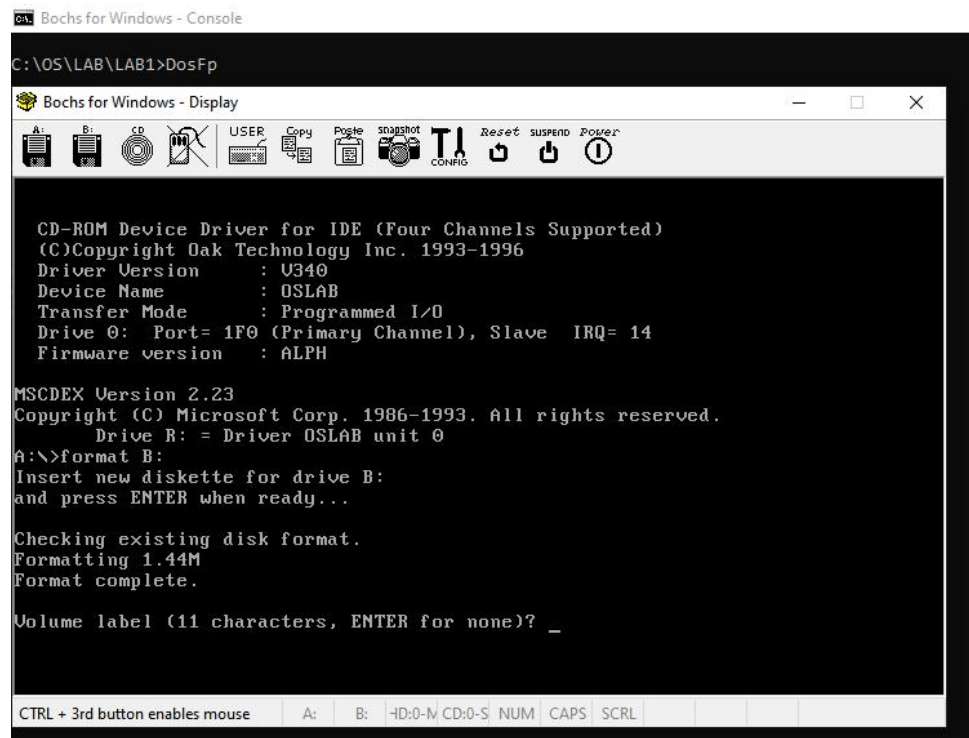
Press any key to continue
```

File image floppy 'floppya.img' yang dibuat dengan langkah di atas belum dapat digunakan, karena belum di 'FORMAT', seperti pada floppy sebenarnya jika belum di format floppy tidak dapat di baca. Langkah –langkah untuk memformat 'floppya.img' adalah:

- a. Jalankan PC-Simulator dari 'Command Prompt' dengan perintah 'DosFp'.



- b. Pada konfigurasi PC-Simulator file 'floppya.img' terpasang pada 'drive B:'. Selanjutnya dari prompt 'A:>' ketikkan 'Format B:' <ENTER> [2x].



- c. Tutup kembali PC-Simulator dengan klik pada tombol power, di bagian menu atas-kanan.

## E. Melihat data dalam boot sector

- a. Copy 512 byte data bootsektor ke dalam sebuah file terpisah, caranya dari 'Command Prompt' ketik 'dd if=floppya.img of=boots.bin count=1' maksud dari perintah ini adalah menyalin byte data dari file 'floppya.img' ke dalam file 'boots.bin' sebanyak satu sektor mulai dari sektor 0.

```
C:\> Command Prompt

C:\OS\LAB\LAB1>dd if=floppya.img of=boots.bin count=1
rawwrite dd for windows version 0.5.
Written by John Newbigin <jn@it.swin.edu.au>
This program is covered by the GPL. See copying.txt for details
1+0 records in
1+0 records out

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

Untuk melihat isinya jalankan 'Command Prompt' dan ketik 'debug boots.bin' dan <ENTER> (Perlu diketahui untuk mengakhiri debug ketikkan 'quit' kemudian enter).

- b. Dari 'Command Prompt' ketikkan 'tdump boots.bin' <ENTER>.

```
C:\> Command Prompt

C:\OS\LAB\LAB1>tdump boots.bin
Turbo Dump Version 5.0.16.12 Copyright (c) 1988, 2000 Inprise Corporation
Display of File BOOTS.BIN

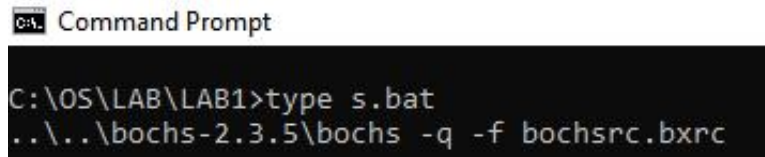
000000: EB 3C 90 4D 53 57 49 4E 34 2E 31 00 02 01 01 00 .<.MSWIN4.1....
000010: 02 E0 00 40 0B F0 09 00 12 00 02 00 00 00 00 00 ...@.....
000020: 00 00 00 00 00 00 29 06 13 4C 14 4E 4F 20 4E 41 .....).L.NO NA
000030: 4D 45 20 20 20 20 46 41 54 31 32 20 20 20 33 C9 ME FAT12 3.
000040: 8E D1 BC FC 7B 16 07 BD 78 00 C5 76 00 1E 56 16 ....{...x..v..V.
000050: 55 BF 22 05 89 7E 00 89 4E 02 B1 0B FC F3 A4 06 U."..~..N.....
000060: 1F BD 00 7C C6 45 FE 0F 38 4E 24 7D 20 8B C1 99 ...|.E..8N$} ...
000070: E8 7E 01 83 EB 3A 66 A1 1C 7C 66 3B 07 8A 57 FC .~...:f..|f;..W.
000080: 75 06 80 CA 02 88 56 02 80 C3 10 73 ED 33 C9 FE u....V....s.3..
000090: 06 D8 7D 8A 46 10 98 F7 66 16 03 46 1C 13 56 1E ..}.F...f..F..V.
0000A0: 03 46 0E 13 D1 8B 76 11 60 89 46 FC 89 56 FE B8 .F....v..".F..V.
0000B0: 20 00 F7 E6 8B 5E 0B 03 C3 48 F7 F3 01 46 FC 11 ....^...H...F..
0000C0: 4E FE 61 BF 00 07 E8 28 01 72 3E 38 2D 74 17 60 N.a....(r>8-t.`
0000D0: B1 0B BE D8 7D F3 A6 61 74 3D 4E 74 09 83 C7 20 ....}.at=Nt...
0000E0: 3B FB 72 E7 EB DD FE 0E D8 7D 7B A7 BE 7F 7D AC ;.r.....}{...}.
0000F0: 98 03 F0 AC 98 40 74 0C 48 74 13 B4 0E BB 07 00 .....@t.Ht.....
000100: CD 10 EB EF BE 82 7D EB E6 BE 80 7D EB E1 CD 16 .....}. ....}.
000110: 5E 1F 66 8F 04 CD 19 BE 81 7D 8B 7D 1A 8D 45 FE ^f.....}. ..E.
000120: 8A 4E 0D F7 E1 03 46 FC 13 56 FE B1 04 E8 C2 00 .N....F..V.....
000130: 72 D7 EA 00 02 70 00 52 50 06 53 6A 01 6A 10 91 r....p.RP.Sj.j..
000140: 8B 46 18 A2 26 05 96 92 33 D2 F7 F6 91 F7 F6 42 .F..&...3.....B
000150: 87 CA F7 76 1A 8A F2 8A E8 C0 CC 02 0A CC B8 01 ...v.....
000160: 02 80 7E 02 0E 75 04 B4 42 8B F4 8A 56 24 CD 13 ..~..u..B...V$..
000170: 61 61 72 0A 40 75 01 42 03 5E 0B 49 75 77 C3 03 aar.@u.B.^Iuw..
000180: 18 01 27 0D 0A 49 6E 76 61 6C 69 64 20 73 79 73 ..'..Invalid sys
000190: 74 65 6D 20 64 69 73 6B FF 0D 0A 44 69 73 6B 20 tem disk...Disk
0001A0: 49 2F 4F 20 65 72 72 6F 72 FF 0D 0A 52 65 70 6C I/O error...Repl
0001B0: 61 63 65 20 74 68 65 20 64 69 73 6B 2C 20 61 6E ace the disk, an
0001C0: 64 20 74 68 65 6E 20 70 72 65 73 73 20 61 6E 79 d then press any
0001D0: 20 6B 65 79 0D 0A 00 00 49 4F 20 20 20 20 20 20 key....IO
0001E0: 53 59 53 4D 53 44 4F 53 20 20 20 53 59 53 7F 01 SYSMSDOS SYS..
0001F0: 00 41 BB 00 07 60 66 6A 00 E9 3B FF 00 00 55 AA .A...`fj...;...U.

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



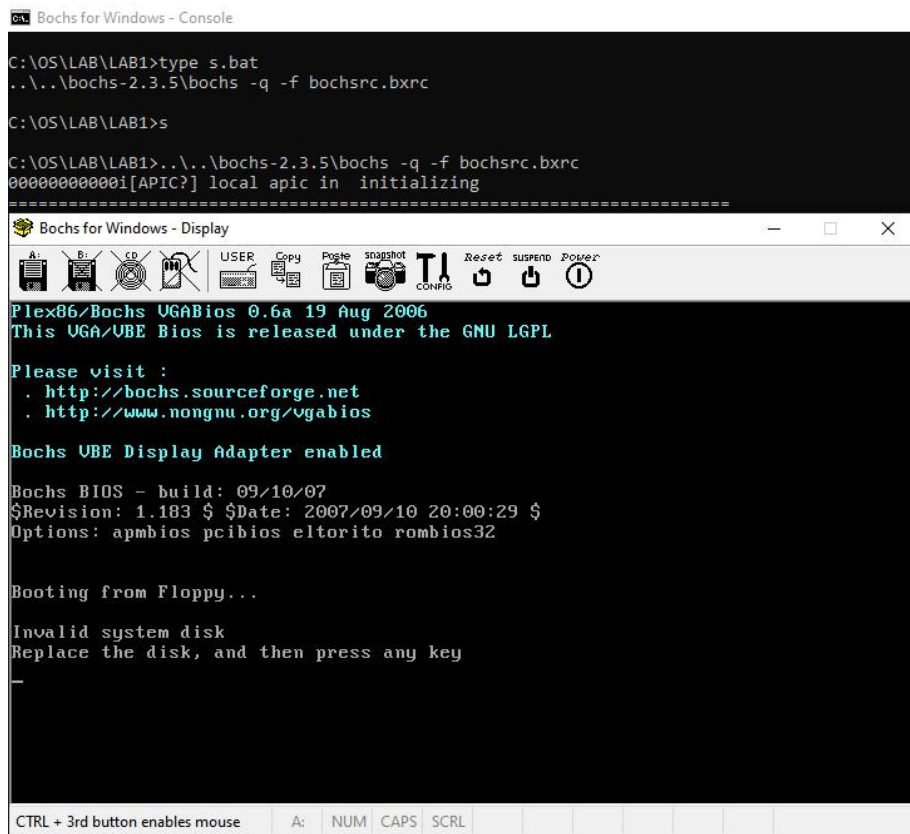
## F. 'Boot' PC-Simulator dengan file image 'floppya.img'

- a. Lihat isi file 's.bat' dengan perintah 'type s.bat' <ENTER>



```
Command Prompt
C:\OS\LAB\LAB1>type s.bat
..\..\bochs-2.3.5\bochs -q -f bochsrc.bxrc
```

- b. Selanjutnya masukan perintah 's' <ENTER>, akan ditampilkan windows 'Bochs for windows - display' yang sedang melakukan proses 'booting' namun tidak berhasil karena tidak menemukan diskboot.



'Floppya.img' belum terisi 'system file' atau 'kernel' sehingga proses boot gagal.

- c. Format 'floppya.img' dan tambahkan 'system file' ke dalamnya. Panggil 'DosFp' <ENTER>. Pada windows 'Bochs' masukan perintah 'A:>format B:/S' <ENTER>, selesaikan proses format dan berikan nama label disk.

```
Bochs for Windows - Console
C:\OS\LAB\LAB1>dosfp
C:\OS\LAB\LAB1>cd "..\..\Bochs-2.3.5\dos"

Bochs for Windows - Display
MSCDEX Version 2.23
Copyright (C) Microsoft Corp. 1986-1993. All rights reserved.
Drive R: = Driver OSLAB unit 0
A:\>format B:/S
Insert new diskette for drive B:
and press ENTER when ready...

Checking existing disk format.
Verifying 1.44M
Format complete.
System transferred

Volume label (11 characters, ENTER for none)?

1,457,664 bytes total disk space
221,696 bytes used by system
1,235,968 bytes available on disk

512 bytes in each allocation unit.
2,414 allocation units available on disk.

Volume Serial Number is 344E-13F3
Format another (Y/N)?_

CTRL + 3rd button enables mouse  A:  B:  HD:0-M CD:0-S  NUM  CAPS  SCRL
```

Untuk memastikan bahwa floppy pada drive 'B:' terisi dengan 'system file',  
periksa dengan perintah berikut 'A:>dir B:' <ENTER>.

```
Bochs for Windows - Console
C:\OS\LAB\LAB1>dosfp
C:\OS\LAB\LAB1>cd "..\..\Bochs-2.3.5\dos"

Bochs for Windows - Display
1,457,664 bytes total disk space
221,696 bytes used by system
1,235,968 bytes available on disk

512 bytes in each allocation unit.
2,414 allocation units available on disk.

Volume Serial Number is 344E-13F3
Format another (Y/N)?N
Format another (Y/N)?n

A:\>dir B:

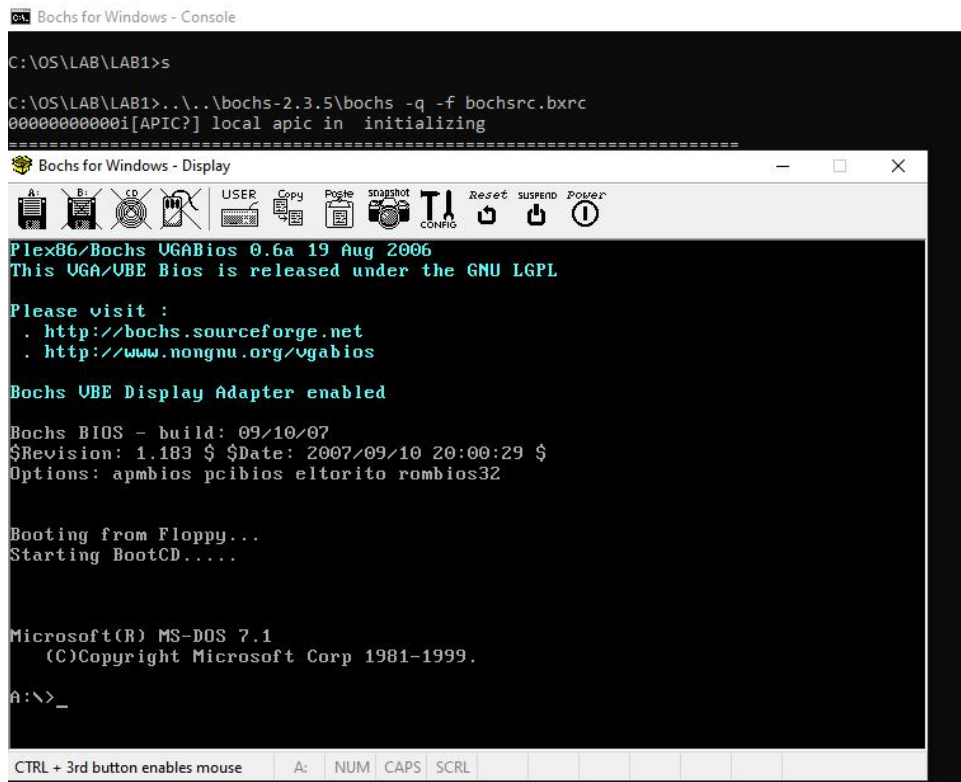
Volume in drive B has no label
Volume Serial Number is 344E-13F3
Directory of B:\

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

A:\>

CTRL + 3rd button enables mouse  A:  B:  HD:0-M CD:0-S  NUM  CAPS  SCRL
```

d. Coba untuk menggunakan 'floppya.img' sebagai 'boot disk' lagi, ketik 'S' <ENTER>.



The image shows a screenshot of a Bochs virtual machine environment. At the top, a console window titled 'Bochs for Windows - Console' displays the following text:

```
C:\OS\LAB\LAB1>s
C:\OS\LAB\LAB1>..\..\bochs-2.3.5\bochs -q -f bochsrc.bxrc
0000000000i[APIC?] local apic in initializing
=====
```

Below the console is a 'Bochs for Windows - Display' window. It features a toolbar with icons for A: (floppy), B: (floppy), CD (CD-ROM), USER, Copy, Paste, Snapshot, CONFIG, Reset, Suspend, and Power. The main display area shows the following text:

```
Plex86/Bochs VGABios 0.6a 19 Aug 2006
This UGA/VBE Bios is released under the GNU LGPL

Please visit :
. http://bochs.sourceforge.net
. http://www.nongnu.org/vgabios

Bochs VBE Display Adapter enabled

Bochs BIOS - build: 09/10/07
$Revision: 1.183 $ $Date: 2007/09/10 20:00:29 $
Options: apmbios pcibios eltorito rombios32

Booting from Floppy...
Starting BootCD.....

Microsoft(R) MS-DOS 7.1
(C)Copyright Microsoft Corp 1981-1999.

A:\>_
```

At the bottom of the display window, a status bar indicates 'CTRL + 3rd button enables mouse' and shows the current keyboard state: 'A: NUM CAPS SCRL'.

## TUGAS

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.

Jawab:

(**ASCII**) singkatan dari American Standard **Code** for Information Interchange atau **Kode** Standar Amerika untuk Pertukaran Informasi (/ˈæski/ ( simak) ASS-kee), adalah standar pengkodean karakter untuk alat komunikasi. **Kode ASCII** mewakili teks dalam komputer, peralatan telekomunikasi, dan perangkat lainnya. (sumber: wikipedia.)

Karakter	Hexadecimal	Decimal	Binary	Keterangan
NUL	0000	0	0	Null (tidak tampak)
SOH	0001	1	1	Start of heading (tidak tampak)
STX	0002	2	10	Start of text (tidak tampak)
ETX	0003	3	11	End of text (tidak tampak)
EOT	0004	4	100	End of transmission (tidak tampak)
ENQ	0005	5	101	Enquiry (tidak tampak)
ACK	0006	6	110	Acknowledge (tidak tampak)
BEL	0007	7	111	Bell (tidak tampak)
BS	0008	8	1000	Menghapus satu karakter di belakang kursor (Backspace)
HT	0009	9	1001	Horizontal tabulation
LF	000A	10	1010	Pergantian baris (Line feed)
VT	000B	11	1011	Tabulasi vertikal
FF	000C	12	1100	Pergantian baris (Form feed)
CR	000D	13	1101	Pergantian baris (carriage return)
SO	000E	14	1110	Shift out (tidak tampak)
SI	000F	15	1111	Shift in (tidak tampak)
DLE	0010	16	10000	Data link escape (tidak tampak)
DC1	0011	17	10001	Device control 1 (tidak tampak)
DC2	0012	18	10010	Device control 2 (tidak tampak)
DC3	0013	19	10011	Device control 3 (tidak tampak)
DC4	0014	20	10100	Device control 4 (tidak tampak)
NAK	0015	21	10101	Negative acknowledge (tidak tampak)
SYN	0016	22	10110	Synchronous idle (tidak tampak)
ETB	0017	23	10111	End of transmission block (tidak tampak)
CAN	0018	24	11000	Cancel (tidak tampak)
EM	0019	25	11001	End of medium (tidak tampak)
SUB	001A	26	11010	Substitute (tidak tampak)
ESC	001B	27	11011	Escape (tidak tampak)
FS	001C	28	11100	File separator
GS	001D	29	11101	Group separator
RS	001E	30	11110	Record separator
US	001F	31	11111	Unit separator
SP	0020	32	100000	Spasi
!	0021	33	100001	Tanda seru (exclamation)
"	0022	34	100010	Tanda kutip dua
#	0023	35	100011	Tanda pagar (kres)
\$	0024	36	100100	Tanda mata uang dolar
%	0025	37	100101	Tanda persen
&	0026	38	100110	Karakter ampersand (&)
'	0027	39	100111	Karakter Apostrof



(	0028	40	101000	Tanda kurung buka
)	0029	41	101001	Tanda kurung tutup
*	002A	42	101010	Karakter asterisk (bintang)
+	002B	43	101011	Tanda tambah (plus)
,	002C	44	101100	Karakter koma
-	002D	45	101101	Karakter hyphen (strip)
.	002E	46	101110	Tanda titik
/	002F	47	101111	Garis miring ( <i>slash</i> )
0	0030	48	110000	Angka nol
1	0031	49	110001	Angka satu
2	0032	50	110010	Angka dua
3	0033	51	110011	Angka tiga
4	0034	52	110100	Angka empat
5	0035	53	110101	Angka lima
6	0036	54	110110	Angka enam
7	0037	55	110111	Angka tujuh
8	0038	56	111000	Angka delapan
9	0039	57	111001	Angka sembilan
:	003A	58	111010	Tanda titik dua
;	003B	59	111011	Tanda titik koma
<	003C	60	111100	Tanda lebih kecil
=	003D	61	111101	Tanda sama dengan
>	003E	62	111110	Tanda lebih besar
?	003F	63	111111	Tanda tanya
@	0040	64	1000000	A keong (@)
A	0041	65	1000001	Huruf latin A kapital
B	0042	66	1000010	Huruf latin B kapital
C	0043	67	1000011	Huruf latin C kapital
D	0044	68	1000100	Huruf latin D kapital
E	0045	69	1000101	Huruf latin E kapital
F	0046	70	1000110	Huruf latin F kapital
G	0047	71	1000111	Huruf latin G kapital
H	0048	72	1001000	Huruf latin H kapital
I	0049	73	1001001	Huruf latin I kapital
J	004A	74	1001010	Huruf latin J kapital
K	004B	75	1001011	Huruf latin K kapital
L	004C	76	1001100	Huruf latin L kapital
M	004D	77	1001101	Huruf latin M kapital
N	004E	78	1001110	Huruf latin N kapital
O	004F	79	1001111	Huruf latin O kapital
P	0050	80	1010000	Huruf latin P kapital
Q	0051	81	1010001	Huruf latin Q kapital
R	0052	82	1010010	Huruf latin R kapital
S	0053	83	1010011	Huruf latin S kapital
T	0054	84	1010100	Huruf latin T kapital
U	0055	85	1010101	Huruf latin U kapital
V	0056	86	1010110	Huruf latin V kapital
W	0057	87	1010111	Huruf latin W kapital
X	0058	88	1011000	Huruf latin X kapital
Y	0059	89	1011001	Huruf latin Y kapital
Z	005A	90	1011010	Huruf latin Z kapital
[	005B	91	1011011	Kurung siku kiri
\	005C	92	1011100	Garis miring terbalik ( <i>backslash</i> )
]	005D	93	1011101	Kurung siku kanan
^	005E	94	1011110	Tanda pangkat
_	005F	95	1011111	Garis bawah ( <i>underscore</i> )

`	0060	96	1100000	Tanda petik satu
a	0061	97	1100001	Huruf latin a kecil
b	0062	98	1100010	Huruf latin b kecil
c	0063	99	1100011	Huruf latin c kecil
d	0064	100	1100100	Huruf latin d kecil
e	0065	101	1100101	Huruf latin e kecil
f	0066	102	1100110	Huruf latin f kecil
g	0067	103	1100111	Huruf latin g kecil
h	0068	104	1101000	Huruf latin h kecil
i	0069	105	1101001	Huruf latin i kecil
j	006A	106	1101010	Huruf latin j kecil
k	006B	107	1101011	Huruf latin k kecil
l	006C	108	1101100	Huruf latin l kecil
m	006D	109	1101101	Huruf latin m kecil
n	006E	110	1101110	Huruf latin n kecil
o	006F	111	1101111	Huruf latin o kecil
p	0070	112	1110000	Huruf latin p kecil
q	0071	113	1110001	Huruf latin q kecil
r	0072	114	1110010	Huruf latin r kecil
s	0073	115	1110011	Huruf latin s kecil
t	0074	116	1110100	Huruf latin t kecil
u	0075	117	1110101	Huruf latin u kecil
v	0076	118	1110110	Huruf latin v kecil
w	0077	119	1110111	Huruf latin w kecil
x	0078	120	1111000	Huruf latin x kecil
y	0079	121	1111001	Huruf latin y kecil
z	007A	122	1111010	Huruf latin z kecil
{	007B	123	1111011	Kurung kurawal buka
	007C	124	1111100	Garis vertikal (pipa)
}	007D	125	1111101	Kurung kurawal tutup
~	007E	126	1111110	Karakter gelombang (tilde)
DEL	007F	127	1111111	Delete
	0080	128	10000000	Dicadangkan
	0081	129	10000001	Dicadangkan
	0082	130	10000010	Dicadangkan
	0083	131	10000011	Dicadangkan
IND	0084	132	10000100	Index
NEL	0085	133	10000101	Next line
SSA	0086	134	10000110	Start of selected area
ESA	0087	135	10000111	End of selected area
	0088	136	10001000	Character tabulation set
	0089	137	10001001	Character tabulation with justification
	008A	138	10001010	Line tabulation set
PLD	008B	139	10001011	Partial line down
PLU	008C	140	10001100	Partial line up
	008D	141	10001101	Reverse line feed
SS2	008E	142	10001110	Single shift two
SS3	008F	143	10001111	Single shift three
DCS	0090	144	10010000	Device control string
PU1	0091	145	10010001	Private use one
PU2	0092	146	10010010	Private use two
STS	0093	147	10010011	Set transmit state
CCH	0094	148	10010100	Cancel character
MW	0095	149	10010101	Message waiting
	0096	150	10010110	Start of guarded area
	0097	151	10010111	End of guarded area

	0098	152	10011000	Start of string
	0099	153	10011001	Dicadangkan
	009A	154	10011010	Single character introducer
CSI	009B	155	10011011	Control sequence introducer
ST	009C	156	10011100	String terminator
OSC	009D	157	10011101	Operating system command
PM	009E	158	10011110	Privacy message
APC	009F	158	10011110	Application program command
	00A0	160	10100000	Spasi yang bukan pemisah kata
ı	00A1	161	10100001	Tanda seru terbalik
¢	00A2	162	10100010	Tanda sen (Cent)
£	00A3	163	10100011	Tanda Poundsterling
¤	00A4	164	10100100	Tanda mata uang ( <i>Currency</i> )
¥	00A5	165	10100101	Tanda Yen
¦	00A6	166	10100110	Garis tegak putus-putus ( <i>broken bar</i> )
§	00A7	167	10100111	Section sign
¨	00A8	168	10101000	Diaeresis
©	00A9	169	10101001	Tanda hak cipta (Copyright)
ª	00AA	170	10101010	Feminine ordinal indicator
«	00AB	171	10101011	Left-pointing double angle quotation mark
¬	00AC	172	10101100	Not sign
	00AD	173	10101101	Tanda strip ( <i>hyphen</i> )
®	00AE	174	10101110	Tanda merk terdaftar
—	00AF	175	10101111	Macron
°	00B0	176	10110000	Tanda derajat
±	00B1	177	10110001	Tanda kurang lebih (plus-minus)
²	00B2	178	10110010	Tanda kuadrat (pangkat dua)
³	00B3	179	10110011	Tanda kubik (pangkat tiga)
´	00B4	180	10110100	Acute accent
µ	00B5	181	10110101	Micro sign
¶	00B6	182	10110110	Pilcrow sign
·	00B7	183	10110111	Middle dot

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’.

Jawab :

### 1. ACALL (Absolute Call)

ACALL berfungsi untuk memanggil sub rutin program

### 2. ADD (Add Immediate Data)

ADD berfungsi untuk menambah 8 bit data langsung ke dalam isi akumulator dan menyimpan hasilnya pada akumulator.

### 3. ADDC (Add Carry Plus Immediate Data to Accumulator)

ADDC berfungsi untuk menambahkan isi carry flag (0 atau 1) ke dalam isi akumulator. Data langsung 8 bit ditambahkan ke akumulator.

### 4. AJMP (Absolute Jump)

AJMP adalah perintah jump mutlak. Jump dalam 2 KB dimulai dari alamat yang mengikuti perintah AJMP. AJMP berfungsi untuk mentransfer kendali program ke lokasi

dimana alamat dikalkulasi dengan cara yang sama dengan perintah ACALL. Konter program ditambahkan dua kali dimana perintah AJMP adalah perintah 2-byte. Konter program di-load dengan a10 – a0 11 bits, untuk membentuk alamat tujuan 16-bit.

#### **5. ANL (logical AND memori ke akumulator)**

ANL berfungsi untuk mengAND-kan isi alamat data dengan isi akumulator.

#### **6. CJNE (Compare Indirect Address to Immediate Data)**

CJNE berfungsi untuk membandingkan data langsung dengan lokasi memori yang dialamati oleh register R atau Akumulator A. apabila tidak sama maka instruksi akan menuju ke alamat kode.

Format : CJNE R,#data,Alamat kode.

#### **7. CLR (Clear Accumulator)**

CLR berfungsi untuk mereset data akumulator menjadi 00H.

Format : CLR A

#### **8. CPL (Complement Accumulator)**

CPL berfungsi untuk mengkomplemen isi akumulator.

#### **9. DA (Decimal Adjust Accumulator)**

DA berfungsi untuk mengatur isi akumulator ke padanan BCD, steleah penambahan dua angka BCD.

#### **10. DEC (Decrement Indirect Address)**

DEC berfungsi untuk mengurangi isi lokasi memori yang ditujukan oleh register R dengan 1, dan hasilnya disimpan pada lokasi tersebut.

#### **11. DIV (Divide Accumulator by B)**

DIV berfungsi untuk membagi isi akumulator dengan isi register B. Akumulator berisi hasil bagi, register B berisi sisa pembagian.

#### **12. DJNZ (Decrement Register And Jump Id Not Zero)**

DJNZ berfungsi untuk mengurangi nilai register dengan 1 dan jika hasilnya sudah 0 maka instruksi selanjutnya akan dieksekusi. Jika belum 0 akan menuju ke alamat kode.

#### **13. INC (Increment Indirect Address)**

INC berfungsi untuk menambahkan isi memori dengan 1 dan menyimpannya pada alamat tersebut.

#### **14. JB (Jump if Bit is Set)**

JB berfungsi untuk membaca data per satu bit, jika data tersebut adalah 1 maka akan menuju ke alamat kode dan jika 0 tidak akan menuju ke alamat kode.

#### **15. JBC (Jump if Bit Set and Clear Bit)**

Bit JBC, berfungsi sebagai perintah rel menguji yang terspesifikasikan secara bit. Jika bit di-set, maka Jump dilakukan ke alamat relatif dan yang terspesifikasi secara bit di dalam perintah dibersihkan. Segmen program berikut menguji bit yang kurang signifikan (LSB:



Least Significant Byte), dan jika ditemukan bahwa ia telah di-set, program melompat ke READ lokasi. JBC juga berfungsi membersihkan LSB dari akumulator.

#### **16. JC (Jump if Carry is Set)**

Instruksi JC berfungsi untuk menguji isi carry flag. Jika berisi 1, eksekusi menuju ke alamat kode, jika berisi 0, instruksi selanjutnya yang akan dieksekusi.

#### **17. JMP (Jump to sum of Accumulator and Data Pointer)**

Instruksi JMP berfungsi untuk memerintahkan loncat kesuatu alamat kode tertentu.

Format : JMP alamat kode.

#### **18. JNB (Jump if Bit is Not Set)**

Instruksi JNB berfungsi untuk membaca data per satu bit, jika data tersebut adalah 0 maka akan menuju ke alamat kode dan jika 1 tidak akan menuju ke alamat kode.

Format : JNB alamat bit,alamat kode.

#### **19. JNC (Jump if Carry Not Set)**

JNC berfungsi untuk menguji bit Carry, dan jika tidak di-set, maka sebuah lompatan akan dilakukan ke alamat relatif yang telah ditentukan.

#### **20. JNZ (Jump if Accumulator Not Zero)**

JNZ adalah mnemonik untuk instruksi jump if not zero (lompat jika tidak nol). Dalam hal ini suatu lompatan akan terjadi bilamana bendera nol dalam keadaan “clear”, dan tidak akan terjadi lompatan bilamana bendera nol tersebut dalam keadaan set. Andaikan bahwa JNZ 7800H disimpan pada lokasi 2100H. Jika Z=0, instruksi berikutnya akan berasal dari lokasi 7800H: dan bilamana Z=1, program akan turun ke instruksi urutan berikutnya pada lokasi 2101H.

#### **21. JZ ( Jump if Accumulator is Zero )**

JZ berfungsi untuk menguji konten-konten akumulator. Jika bukan nol, maka lompatan dilakukan ke alamat relatif yang ditentukan dalam perintah.

#### **22. LCALL ( Long Call )**

LCALL berfungsi untuk memungkinkan panggilan ke subrutin yang berlokasi dimanapun dalam memori program 64K. Operasi LCALL berjalan seperti berikut:

- Menambahkan ke dalam konter program sebanyak 3, karena perintahnya adalah perintah 3-byte.
- Menambahkan penunjuk stack sebanyak 1.
- Menyimpan byte yang lebih rendah dari konter program ke dalam stack.
- Menambahkan penunjuk stack.
- Menyimpan byte yang lebih tinggi dari program ke dalam stack.
- Me-load konter program dengan alamat tujuan 16-bit.

#### **23. LJMP ( Long Jump )**

Long Jump berfungsi untuk memungkinkan lompatan tak bersyarat kemana saja dalam lingkup ruang memori program 64K. LCALL adalah perintah 3-byte. Alamat tujuan 16-bit ditentukan secara langsung dalam perintah tersebut. Alamat tujuan ini di-load ke dalam konter program oleh perintah LJMP.

#### **24. MOV ( Move From Memory )**

MOV berfungsi untuk memindahkan isi akumulator/register atau data dari nilai luar atau alamat lain.

#### **25. MOVC ( Move From Codec Memory )**

Instruksi MOVC berfungsi untuk mengisi accumulator dengan byte kode atau konstanta dari program memory. Alamat byte tersebut adalah hasil penjumlahan unsigned 8 bit pada accumulator dan 16 bit register basis yang dapat berupa data pointer atau program counter. Instruksi ini tidak mempengaruhi flag apapun juga.

#### **26. MOVX (Move Accumulator to External Memory Addressed by Data Pointer)**

MOVX berfungsi untuk memindahkan isi akumulator ke memori data eksternal yang alamatnya ditunjukkan oleh isi data pointer.

#### **27. MUL ( Multiply )**

MUL AB berfungsi untuk mengalikan unsigned 8 bit integer pada accumulator dan register B. Byte rendah (low order) dari hasil perkalian akan disimpan dalam accumulator sedangkan byte tinggi (high order) akan disimpan dalam register B. Jika hasil perkalian lebih besar dari 255 (0FFh), overflow flag akan bernilai '1'. Jika hasil perkalian lebih kecil atau sama dengan 255, overflow flag akan bernilai '0'. Carry flag akan selalu dikosongkan.

#### **28. NOP ( No Operation )**

Fungsi NOP adalah eksekusi program akan dilanjutkan ke instruksi berikutnya. Selain PC, instruksi ini tidak mempengaruhi register atau flag apapun juga.

#### **29. ORL (Logical OR Immediate Data to Accumulator)**

Instruksi ORL berfungsi sebagai instruksi Gerbang logika OR yang akan menjumlahkan Accumulator terhadap nilai yang ditentukan.

Format : ORL A,#data.

#### **30. POP (Pop Stack to Memory)**

Instruksi POP berfungsi untuk menempatkan byte yang ditunjukkan oleh stack pointer ke suatu alamat data.

#### **31. PUSH (Push Memory onto Stack)**

Instruksi PUSH berfungsi untuk menaikkan stack pointer kemudian menyimpan isinya ke suatu alamat data pada lokasi yang ditunjuk oleh stack pointer.

#### **32. RET (Return from subroutine)**

Intruksi RET berfungsi untuk kembali dari suatu subrutin program ke alamat terakhir

subrutin tersebut di panggil.

### **33. RETI ( Return From Interrupt )**

RETI berfungsi untuk mengambil nilai byte tinggi dan rendah dari PC dari stack dan mengembalikan kondisi logika interrupt agar dapat menerima interrupt lain dengan prioritas yang sama dengan prioritas interrupt yang baru saja diproses. Stack pointer akan dikurangi dengan 2. Instruksi ini tidak mempengaruhi flag apapun juga. Nilai PSW tidak akan dikembalikan secara otomatis ke kondisi sebelum interrupt. Eksekusi program akan dilanjutkan pada alamat yang diambil tersebut. Umumnya alamat tersebut adalah alamat setelah lokasi dimana terjadi interrupt. Jika interrupt dengan prioritas sama atau lebih rendah tertunda saat RETI dieksekusi, maka satu instruksi lagi akan dieksekusi sebelum interrupt yang tertunda tersebut diproses.

### **34. RL (Rotate Accumulator Left)**

Instruksi RL berfungsi untuk memutar setiap bit dalam akumulator satu posisi ke kiri.

### **35. . RLC ( Rotate Left through Carry )**

Fungsi : Memutar (Rotate) Accumulator ke Kiri (Left) Melalui Carry Flag. Kedelapan bit accumulator dan carry flag akan diputar satu bit ke kiri secara bersama-sama. Bit 7 akan dirotasi ke carry flag, nilai carry flag akan berpindah ke posisi bit 0. Instruksi ini tidak mempengaruhi flag lain.

### **36. RR ( Rotate Right )**

Fungsi : Memutar (Rotate) Accumulator ke Kanan (Right). Kedelapan bit accumulator akan diputar satu bit ke kanan. Bit 0 akan dirotasi ke posisi bit 7. Instruksi ini tidak mempengaruhi flag apapun juga.

### **37. RRC ( Rotate Right through Carry )**

Fungsi : Memutar (Rotate) Accumulator ke Kanan (Right) Melalui Carry Flag. Kedelapan bit accumulator dan carry flag akan diputar satu bit ke kanan secara bersama-sama. Bit 0 akan dirotasi ke carry flag, nilai carry flag akan berpindah ke posisi bit 7. Instruksi ini tidak mempengaruhi flag lain.

### **38. SETB (set Carry flag)**

Instruksi SETB berfungsi untuk menset carry flag.

### **39. SJMP (Short Jump)**

Sebuah Short Jump berfungsi untuk mentransfer kendali ke alamat tujuan dalam 127 bytes yang mengikuti dan 128 yang mengawali perintah SJMP. Alamat tujuannya ditentukan sebagai sebuah alamat relative 8-bit. Ini adalah Jump tidak bersyarat. Perintah SJMP menambahkan konter program sebanyak 2 dan menambahkan alamat relatif ke dalamnya untuk mendapatkan alamat tujuan. Alamat relatif tersebut ditentukan dalam perintah sebagai 'SJMP rel'.

### **40. SUBB ( Subtract With Borrow )**

Fungsi : Pengurangan (Subtract) dengan Peminjaman (Borrow). SUBB mengurangi variabel yang tertera pada operand kedua dan carry flag sekaligus dari accumulator dan menyimpan hasilnya pada accumulator. SUBB akan memberi nilai '1' pada carry flag jika peminjaman ke bit 7 dibutuhkan dan mengosongkan C jika tidak dibutuhkan peminjaman. Jika C bernilai '1' sebelum mengeksekusi SUBB, hal ini menandakan bahwa terjadi peminjaman pada proses pengurangan sebelumnya, sehingga carry flag dan source byte akan dikurangkan dari accumulator secara bersama-sama. AC akan bernilai '1' jika peminjaman ke bit 3 dibutuhkan dan mengosongkan AC jika tidak dibutuhkan peminjaman. OV akan bernilai '1' jika ada peminjaman ke bit 6 namun tidak ke bit 7 atau ada peminjaman ke bit 7 namun tidak ke bit 6. Saat mengurangi signed integer, OV menandakan adanya angka negative sebagai hasil dari pengurangan angka negatif dari angka positif atau adanya angka positif sebagai hasil dari pengurangan angka positif dari angka negative. Addressing mode yang dapat digunakan adalah: register, direct, register indirect, atau immediate data.

#### **41. SWAP ( Swap Nibbles )**

Fungsi : Menukar (Swap) Upper Nibble dan Lower Nibble dalam Accumulator. SWAP A akan menukar nibble (4 bit) tinggi dan nibble rendah dalam accumulator. Operasi ini dapat dianggap sebagai rotasi 4 bit dengan RR atau RL. Instruksi ini tidak mempengaruhi flag apapun juga.

#### **42. XCH ( Exchange Bytes )**

Fungsi : Menukar (Exchange) Accumulator dengan Variabel Byte. XCH akan mengisi accumulator dengan variabel yang tertera pada operand kedua dan pada saat yang sama juga akan mengisi nilai accumulator ke dalam variabel tersebut. Addressing mode yang dapat digunakan adalah: register, direct, atau register indirect.

#### **43. XCHD ( Exchange Digits )**

Fungsi : Menukar (Exchange) Digit. XCHD menukar nibble rendah dari accumulator, yang umumnya mewakili angka heksadesimal atau BCD, dengan nibble rendah dari internal data memory yang diakses secara indirect. Nibble tinggi kedua register tidak akan terpengaruh. Instruksi ini tidak mempengaruhi flag apapun juga.

#### **44. XRL ( Exclusive OR Logic )**

Fungsi : Logika Exclusive OR untuk Variabel Byte XRL akan melakukan operasi bitwise logika exclusive OR antara kedua variabel yang dinyatakan. Hasilnya akan disimpan pada destination byte. Instruksi ini tidak mempengaruhi flag apapun juga. Kedua operand mampu menggunakan enam kombinasi addressing mode. Saat destination byte adalah accumulator, source byte dapat berupa register, direct, register indirect, atau immediate data. Saat destination byte berupa direct address, source byte dapat berupa accumulator atau immediate data.