Computer Boot

x86

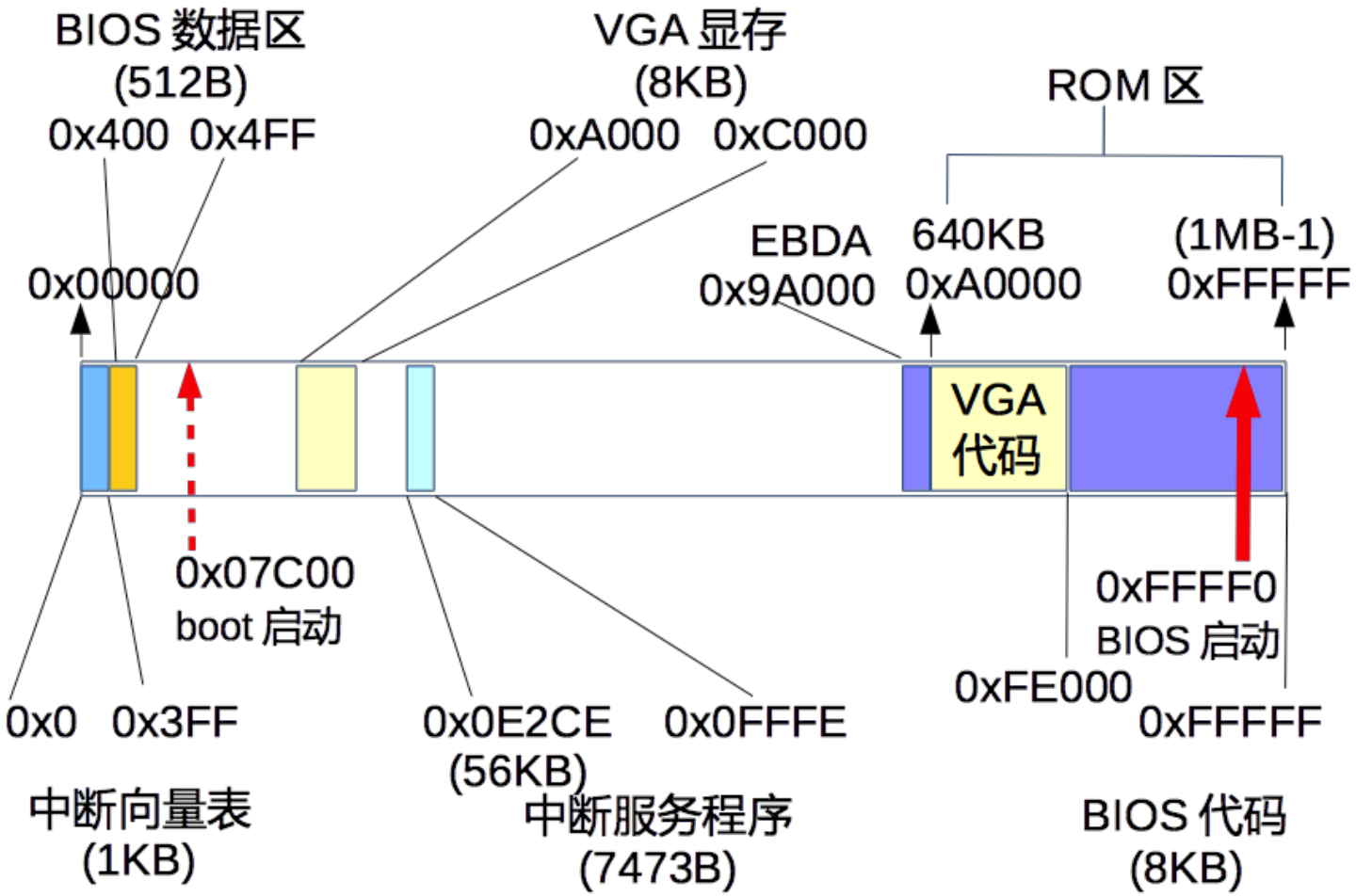
Environment

- Linux: ubuntu 18.04 LTS
- GCC
  - sudo apt-get -y install build-essential libelf-dev binutils-dev
- Bochs
  - sudo apt-get install bochs
  - sudo apt-get install bochs-x

Booting The Computer

- an IBM-compatible personal computer's x86 CPU executes
- Power On, real mode
- the instruction located at reset vector (the physical memory address FFFF0h on 16-bit x86 processors and FFFFFFF0h on 32-bit and 64-bit x86 processors, i.e. BIOS entry point
- BIOS: POST, power-on self-test
- BIOS: goes through a pre-configured list of non-volatile storage devices ("boot device sequence") until it finds one that is bootable
- BIOS:load the bootstrap (i.e. MBR, Master Boot Record) from bootable storage device
- MBR:load the OS Kernel
- OS Kernel: OS services and shell

Memory Layout



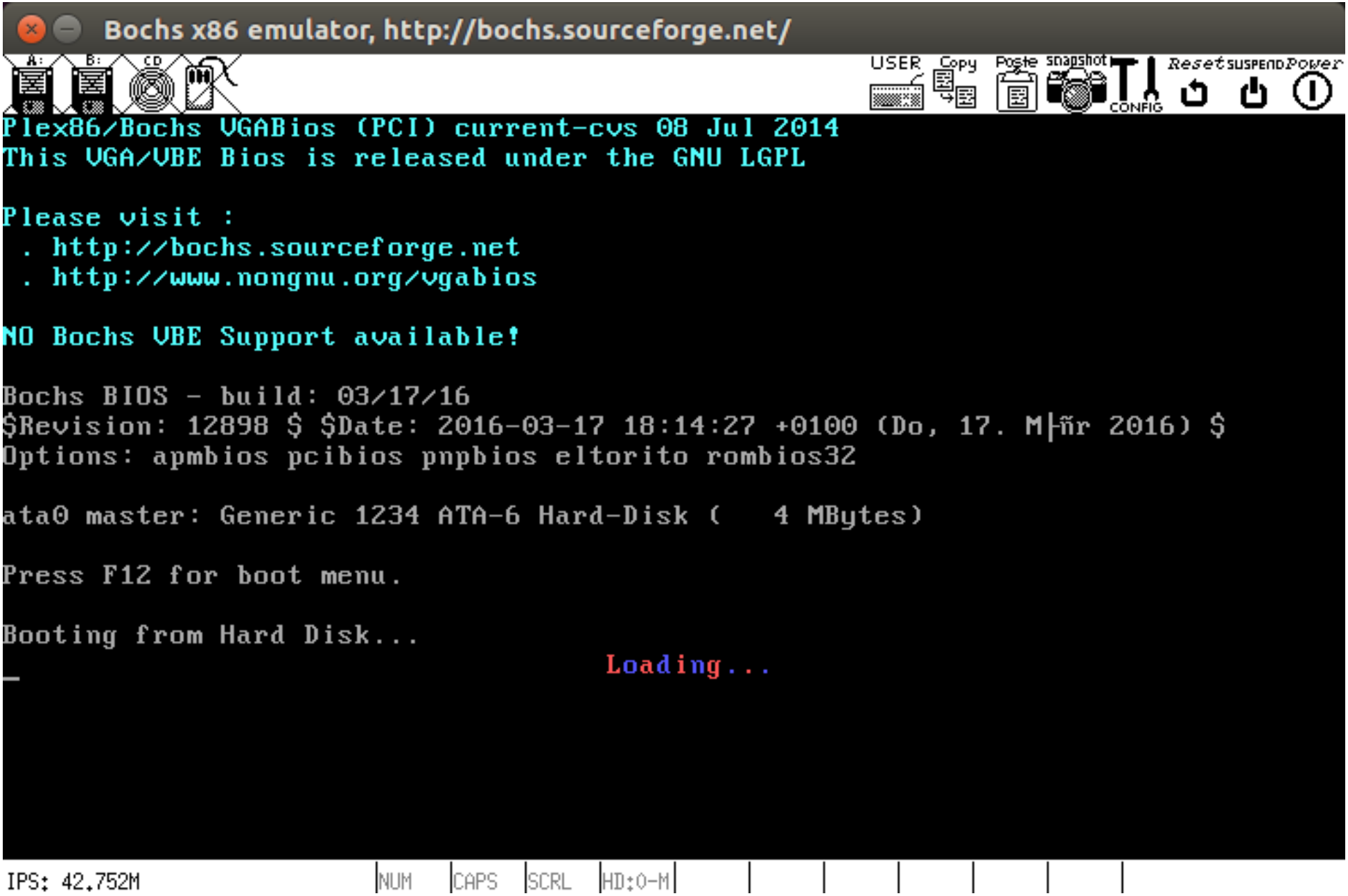
bootstrap

0x000~0x002 <A jump instruction to 0xttt>

0x003~… Disk parameters(used by BIOS)

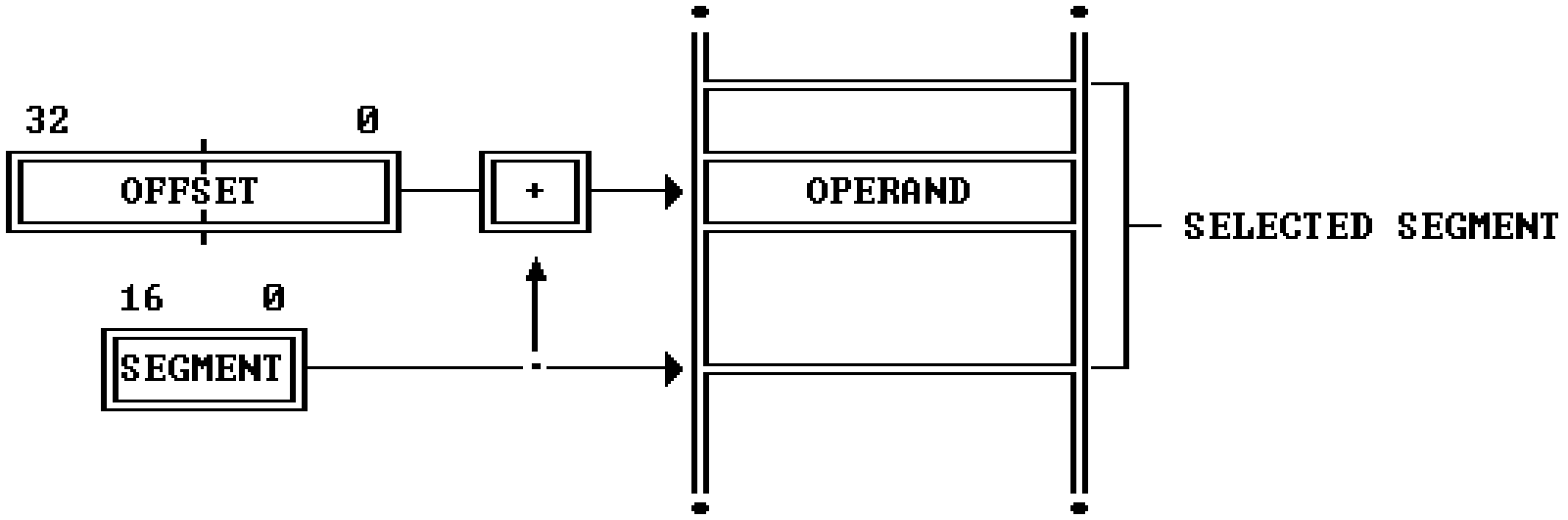
0xttt~0x1fd Bootstrap program

0x1ff~0x1fe 0xaa55


Bochs x86 emulator, http://bochs.sourceforge.net/
A:
B:
CD
USER
Copy
Paste
snapshot
CONFIG
Reset
suspend
Power
Plex86/Bochs VGABios (PCI) current-cvs 08 Jul 2014
This VGA/VBE Bios is released under the GNU LGPL
Please visit :
. http://bochs.sourceforge.net
. http://www.nongnu.org/vgabios
NO Bochs VBE Support available!
Bochs BIOS - build: 03/17/16
$Revision: 12898 $ $Date: 2016-03-17 18:14:27 +0100 (Do, 17. M├ñr 2016) $
Options: apmbios pcibios pnpbios eltorito rombios32
ata0 master: Generic 1234 ATA-6 Hard-Disk (4 MBytes)
Press F12 for boot menu.
Booting from Hard Disk...
Loading...
IPS: 42.752M
NUM
CAPS
SCRL
HD:0-M

80386: Address

Figure 2-1. Two-Component Pointer



80386: Fundamental Data Types

Figure 2-2. Fundamental Data Types

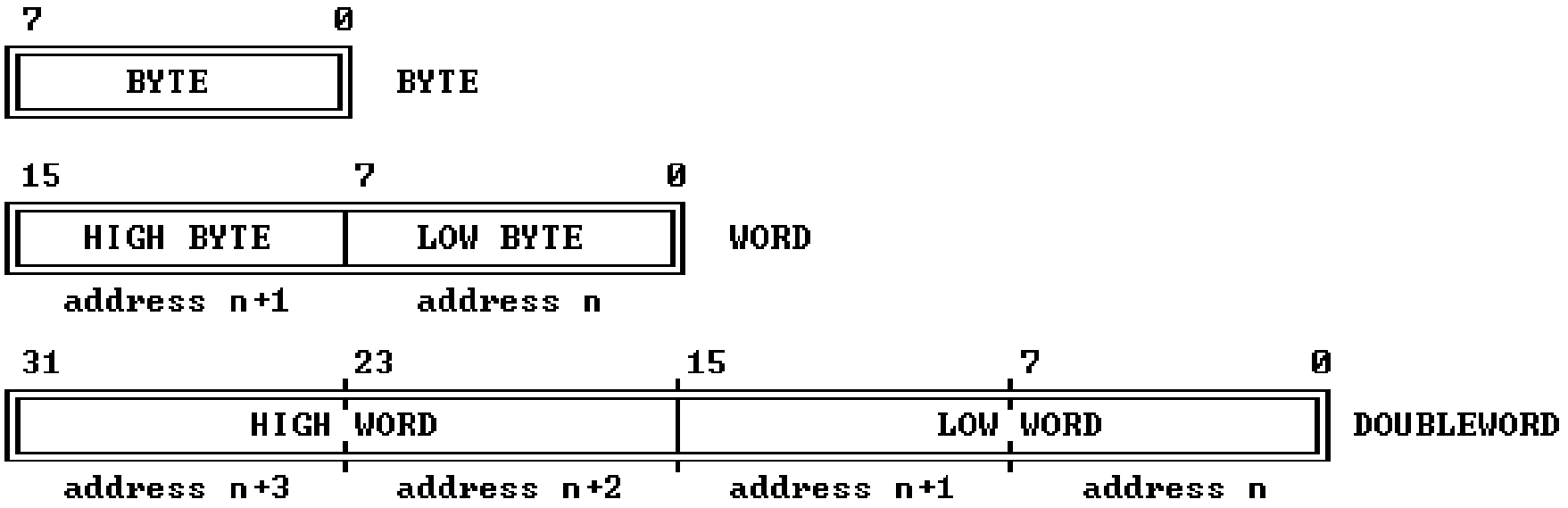
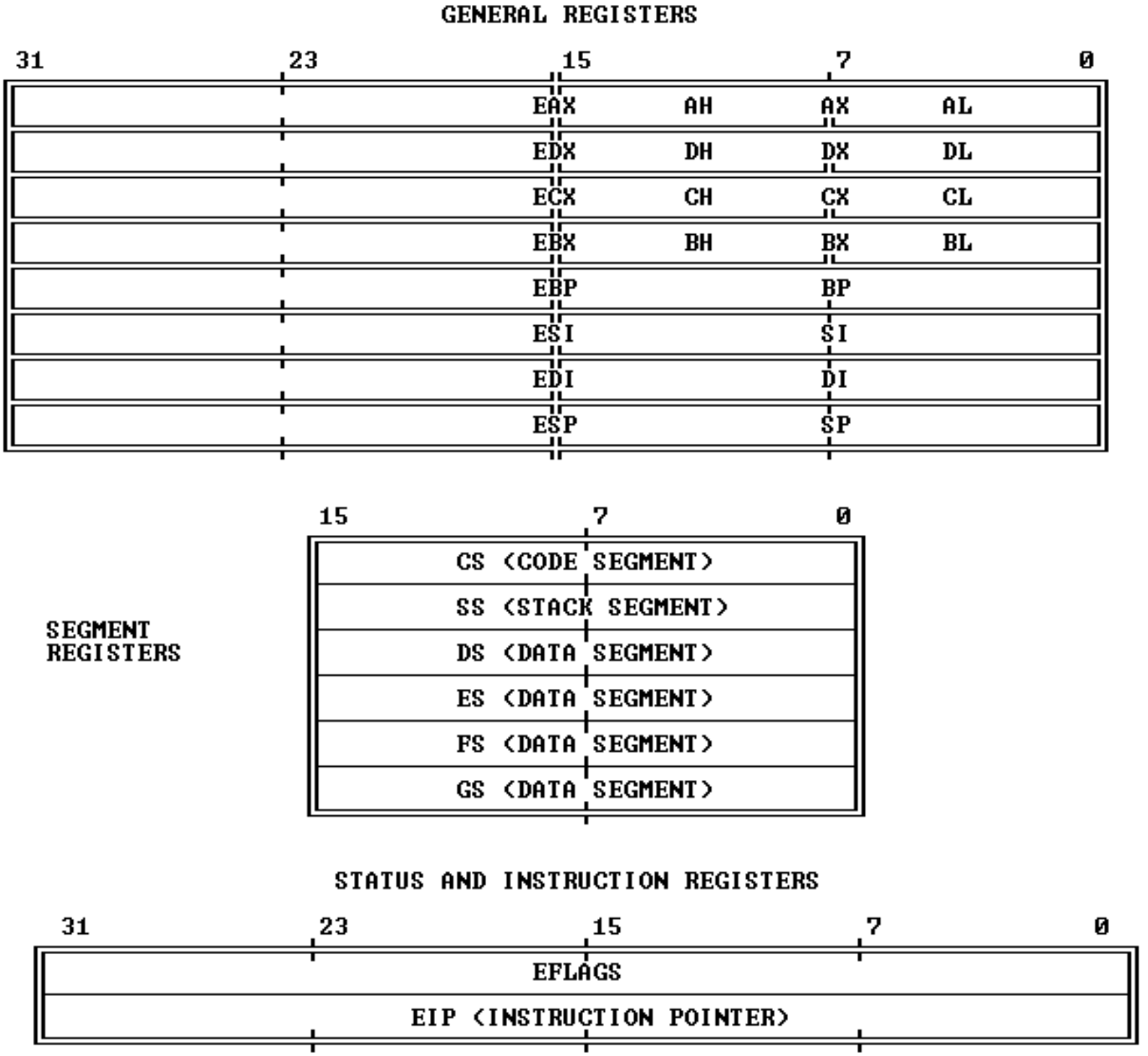
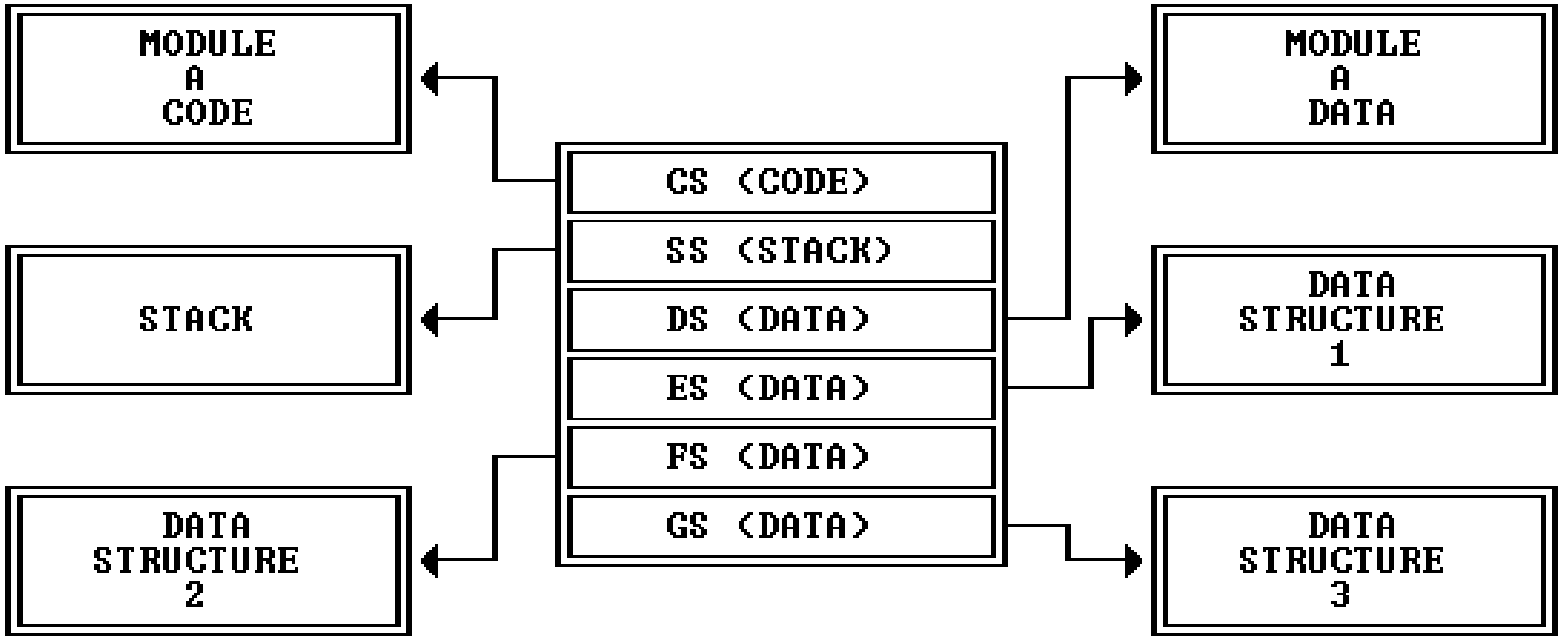


Figure 2-5. 80386 Applications Register Set



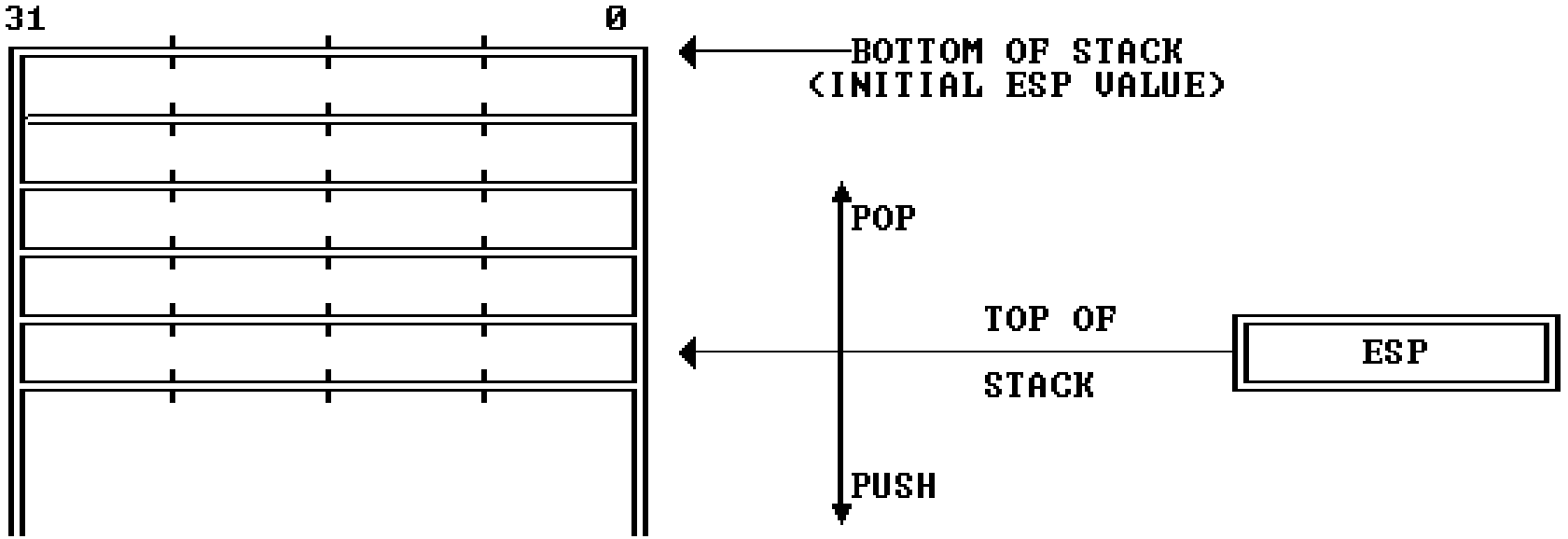
80386: Memory Segmentation

Figure 2-6. Use of Memory Segmentation



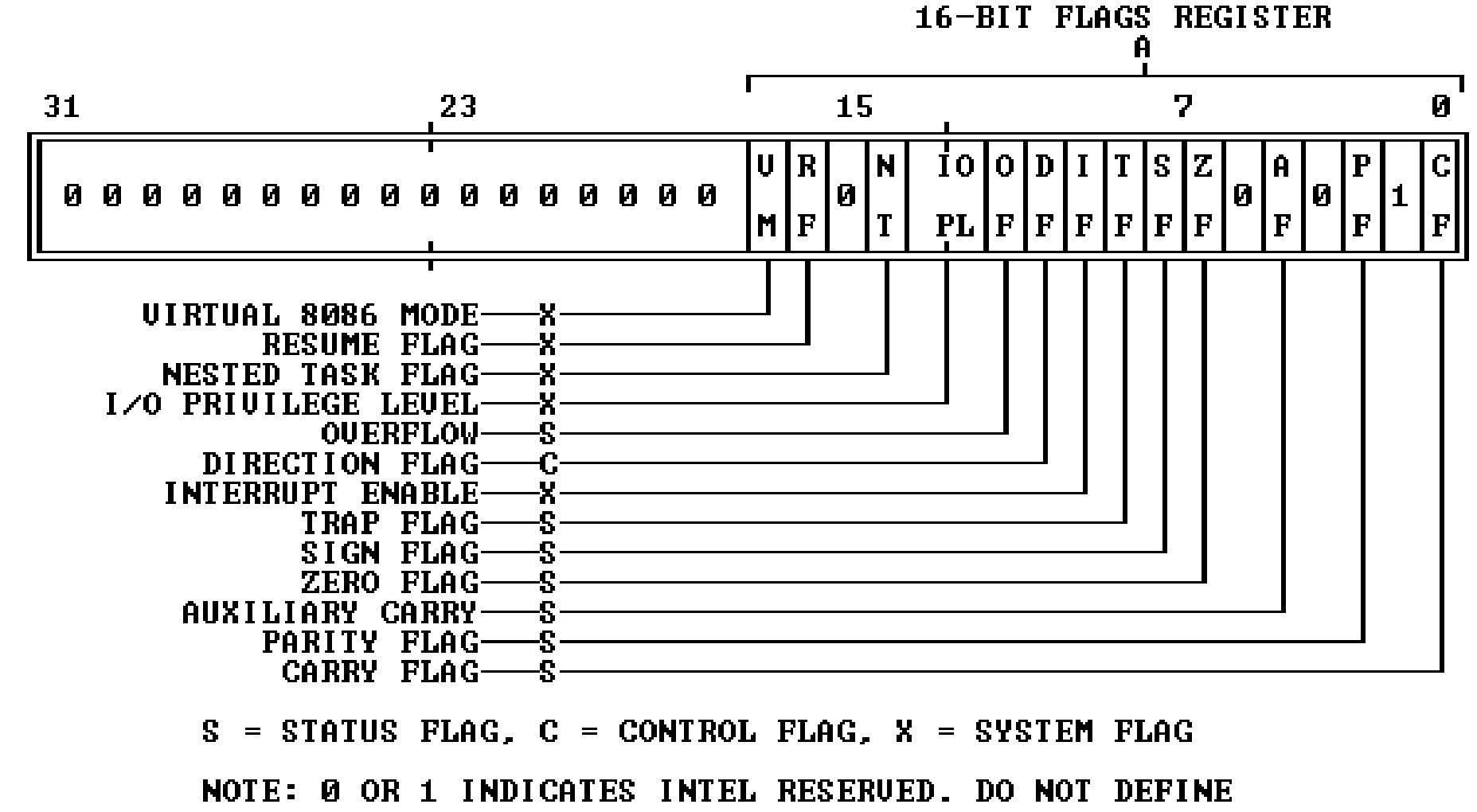
80386: Stack

Figure 2-7. 80386 Stack



80386: EFLAGS Register

Figure 2-8. EFLAGS Register

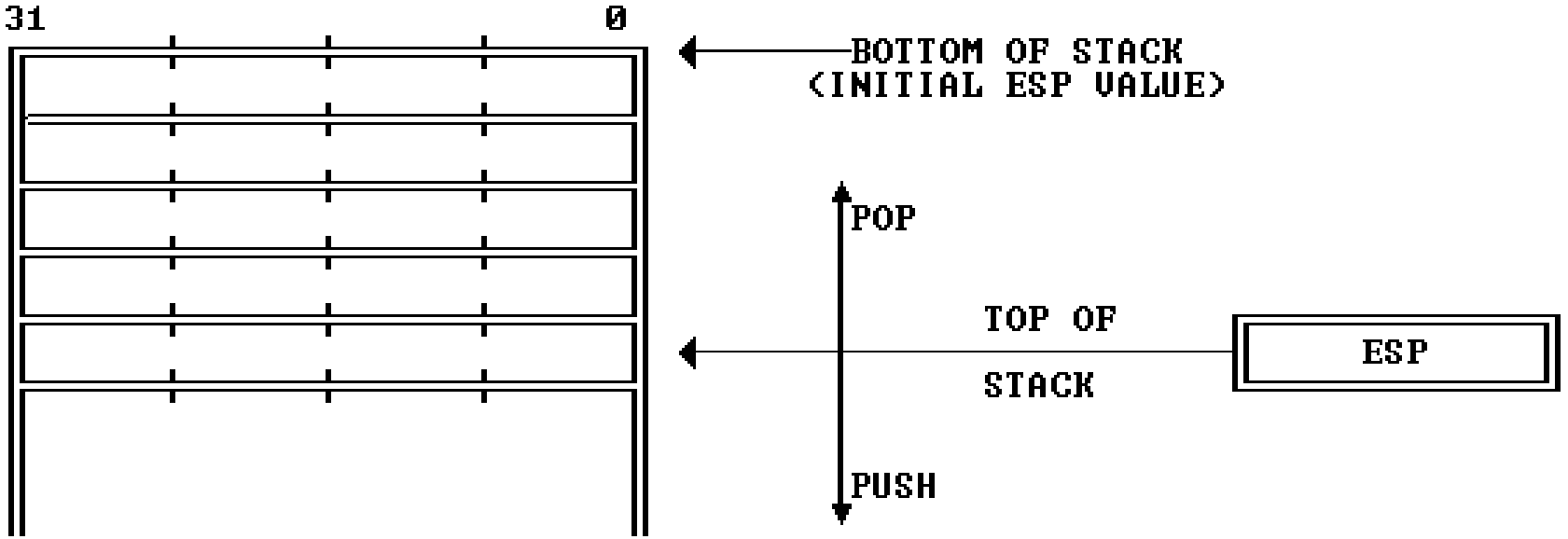


S = STATUS FLAG, C = CONTROL FLAG, X = SYSTEM FLAG

NOTE: 0 OR 1 INDICATES INTEL RESERVED. DO NOT DEFINE

80386: Stack

Figure 2-7. 80386 Stack



Source Code of BOOT

```
-rw-r--r-- 1 albert albert 5120000 11月  9 13:58 bochs.img
-rw-r--r-- 1 albert albert     166 11月  9 14:36 bochs.log
-rwxrwx--- 1 albert albert   33269 9月  19  2017 bochsrc.txt
-rwxr-xr-x 1 albert albert     512 11月  9 13:58 boot
-rw-r--r-- 1 albert albert    1072 11月  9 13:58 boot.o
-rwxr-xr-x 1 albert albert    1016 11月  9 13:58 boot.out
-rwxrwx--- 1 albert albert     831 9月  19  2017 boot.S
-rwxrwx--- 1 albert albert     709 9月  19  2017 Makefile
-rwxrwx--- 1 albert albert     202 9月  29 22:11 Readme.txt
-rwxrwx--- 1 albert albert   21682 9月  19  2017 Screenshot2017-09-19-09-34-32.png
-rwxrwx--- 1 albert albert     399 9月  19  2017 sign.pl
-rwxrwx--- 1 albert albert     533 9月  19  2017 snapshot.txt
```

Makefile

```
CC      := gcc -pipe
AS      := as
AR      := ar
LD      := ld
OBJCOPY := objcopy
OBJDUMP := objdump
NM      := nm
LDFLAGS := -m elf_i386

all: image
boot_objs := boot.o
boot.o: boot.S
    $(CC) -nostdinc -m32 -Os -c -o $@ $<
boot: $(boot_objs)
    $(LD) $(LDFLAGS) -N -e start -Ttext 0x7C00 -o $@.out $^
    $(OBJCOPY) -S -O binary $@.out $@
    perl sign.pl $@
bochs.img: boot
    dd if=/dev/zero of=./.bochs.img~ count=10000 2>/dev/null
    dd if=./boot of=./.bochs.img~ conv=notrunc 2>/dev/null
    mv ./.bochs.img~ ./bochs.img
image: bochs.img
bochs: image
    bochs -f bochsrc.txt
run: bochs
# For deleting the build
clean:
    rm *.o *.out *.asm boot *.log -fr
    rm bochs.img -fr
.PHONY: clean

```

boot.S

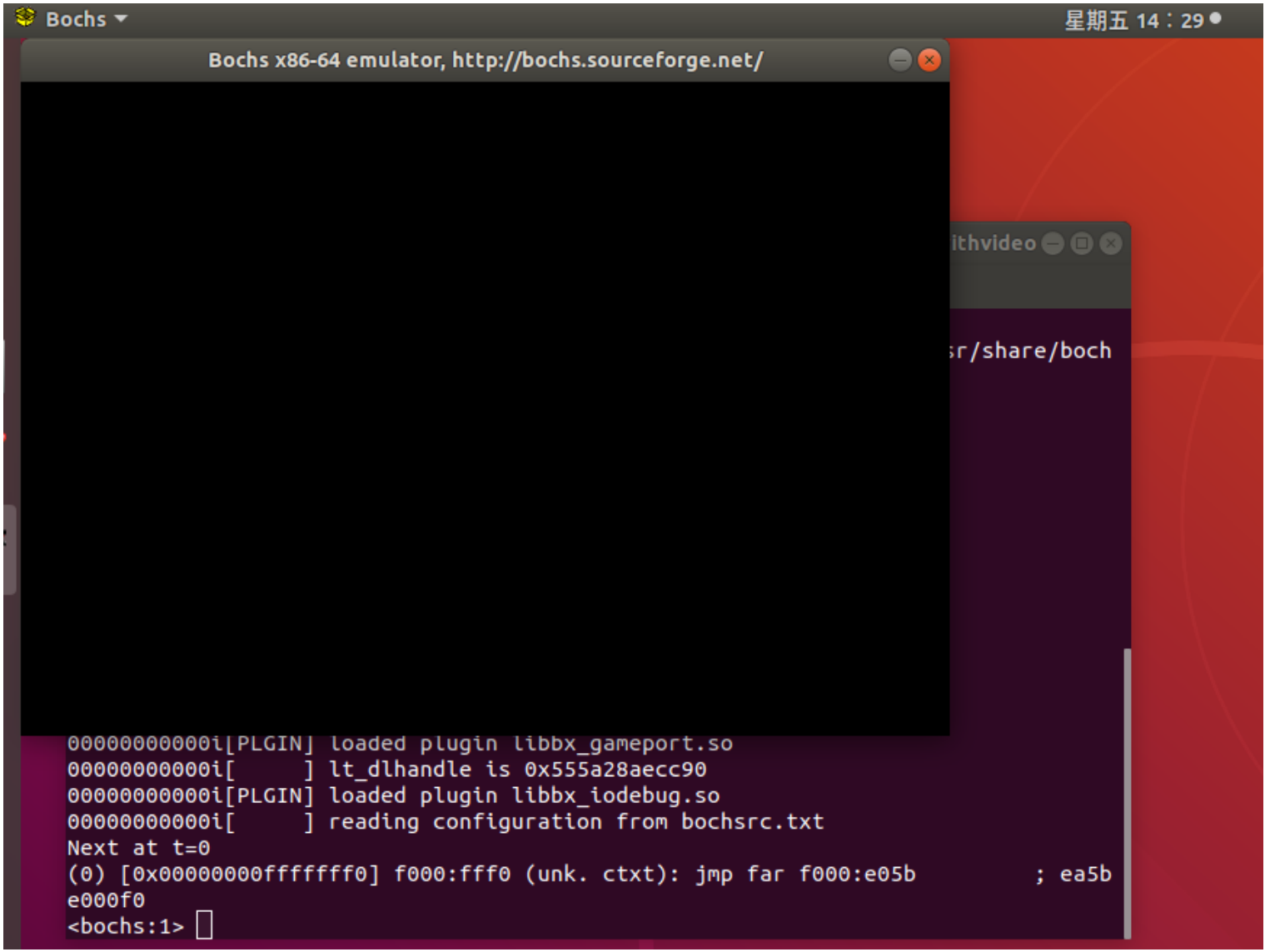
```
#
#the first prog:boot
#directly write to video memory 0xb8000~0xbffff
#
.globl start
start:
  .code16                     # Assemble for 16-bit mode
  cli                         # Disable interrupts
  cld                         # String operations increment

  xorw    %ax,%ax             # Segment number zero
  movw    %ax,%ds             # initiate Data Segment
  movw    %ax,%es             # Extra Segment
  movw    %ax,%ss             # Stack Segment

  movw    $0xb800,%ax       #text display address
  movw    %ax,%es
  movw    $msg1,%si         #source addr:msg1
  movw    $0xb88,%di        #dest addr:0xb8000+0xb88
  movw    $0x14,%cx         #count of msg1
  rep     movsb             #print msg1

spin:   #loop forever
  jmp spin

msg1:                 #Loading...
  .byte 'L',0xc,'o',0x9,'a',0xc,'d',0x9,'i',0xc,'n',0x9,'g',0xc
  .byte '.',0x9,'.',0xc,'.',0x9

  .org 510
  .word  0xAA55
```

boot.S

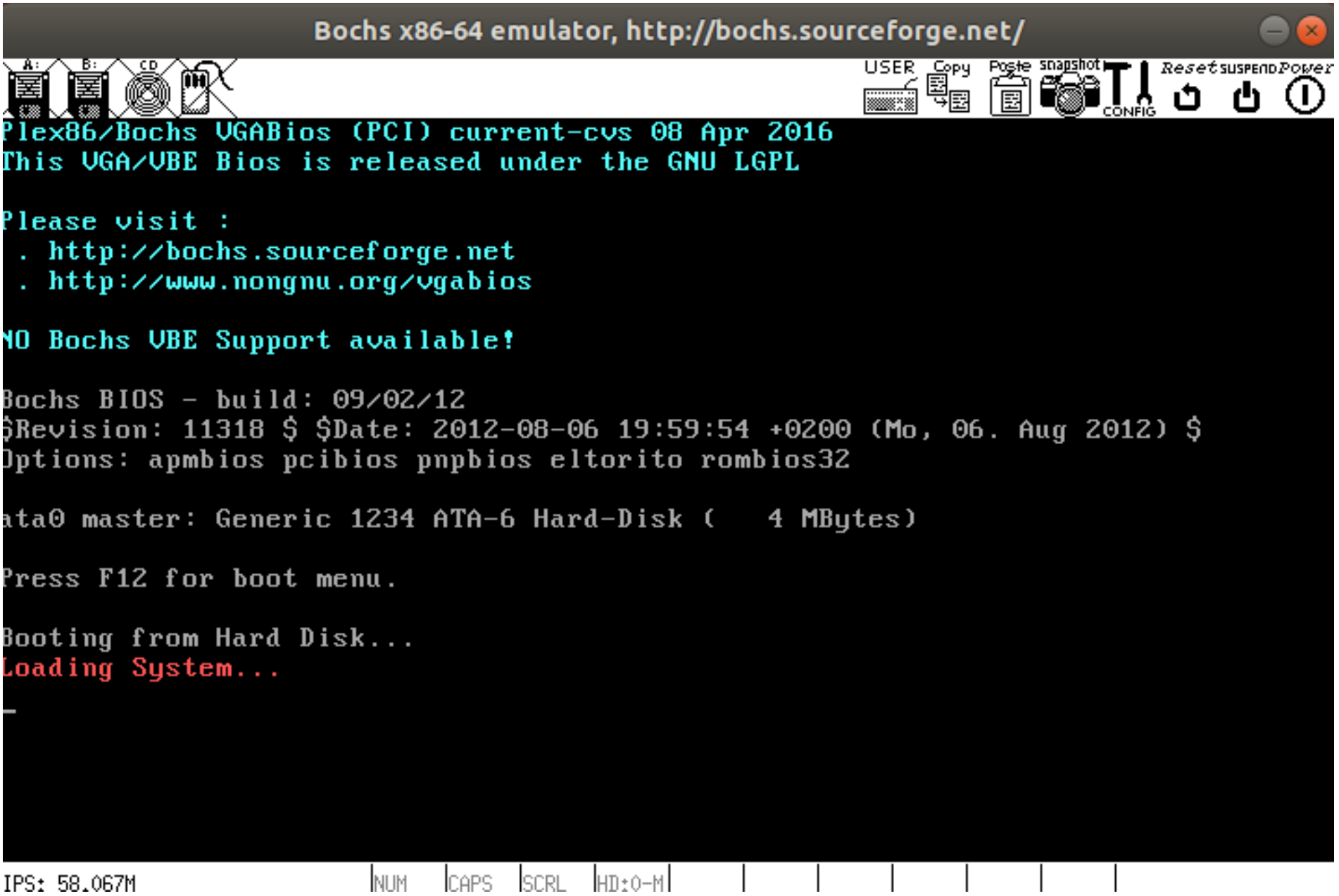
boot.S

```
#
.global start
start:
  .code16                     # Assemble for 16-bit mode
  cli                         # Disable interrupts
  movw    %cs,%ax
  movw    %ax,%ds             # initiate Data Segment
  movw    %ax,%es             # Extra Segment
  movw    %ax,%ss             # Stack Segment

#get current cursor pos
  movb    $0x03, %ah              # read cursor pos
  xor     %bh, %bh
  int     $0x10

#pring msg1
  movw     $19, %cx
  movw     $0x000c, %bx            # page 0, attribute 0x0c (red)
                   #0x07 mormal, 0x0c red
  movw     $msg1, %bp
  movw     $0x1301, %ax            # write string, move cursor
  int      $0x10

spin:   #loop forever
  jmp spin

msg1:                #Loading System...
  .ascii "I am booting system..."
  .byte 13,10

  .org 510
  .word  0xAA55
```

make run



Press “**c**”


Bochs x86-64 emulator, http://bochs.sourceforge.net/
A:
B:
CD
USER
Copy
Paste
snapshot
CONFIG
Reset
suspend
Power
Plex86/Bochs VGABios (PCI) current-cvs 08 Apr 2016
This VGA/VBE Bios is released under the GNU LGPL
Please visit :
. http://bochs.sourceforge.net
. http://www.nongnu.org/vgabios
NO Bochs VBE Support available!
Bochs BIOS - build: 09/02/12
$Revision: 11318 $ $Date: 2012-08-06 19:59:54 +0200 (Mo, 06. Aug 2012) $
Options: apmbios pcibios pnpbios eltorito rombios32
ata0 master: Generic 1234 ATA-6 Hard-Disk (4 MBytes)
Press F12 for boot menu.
Booting from Hard Disk...
Loading System...
IPS: 58.067M
NUM
CAPS
SCRL
HD:0-M

End