

Computer Boot

x86

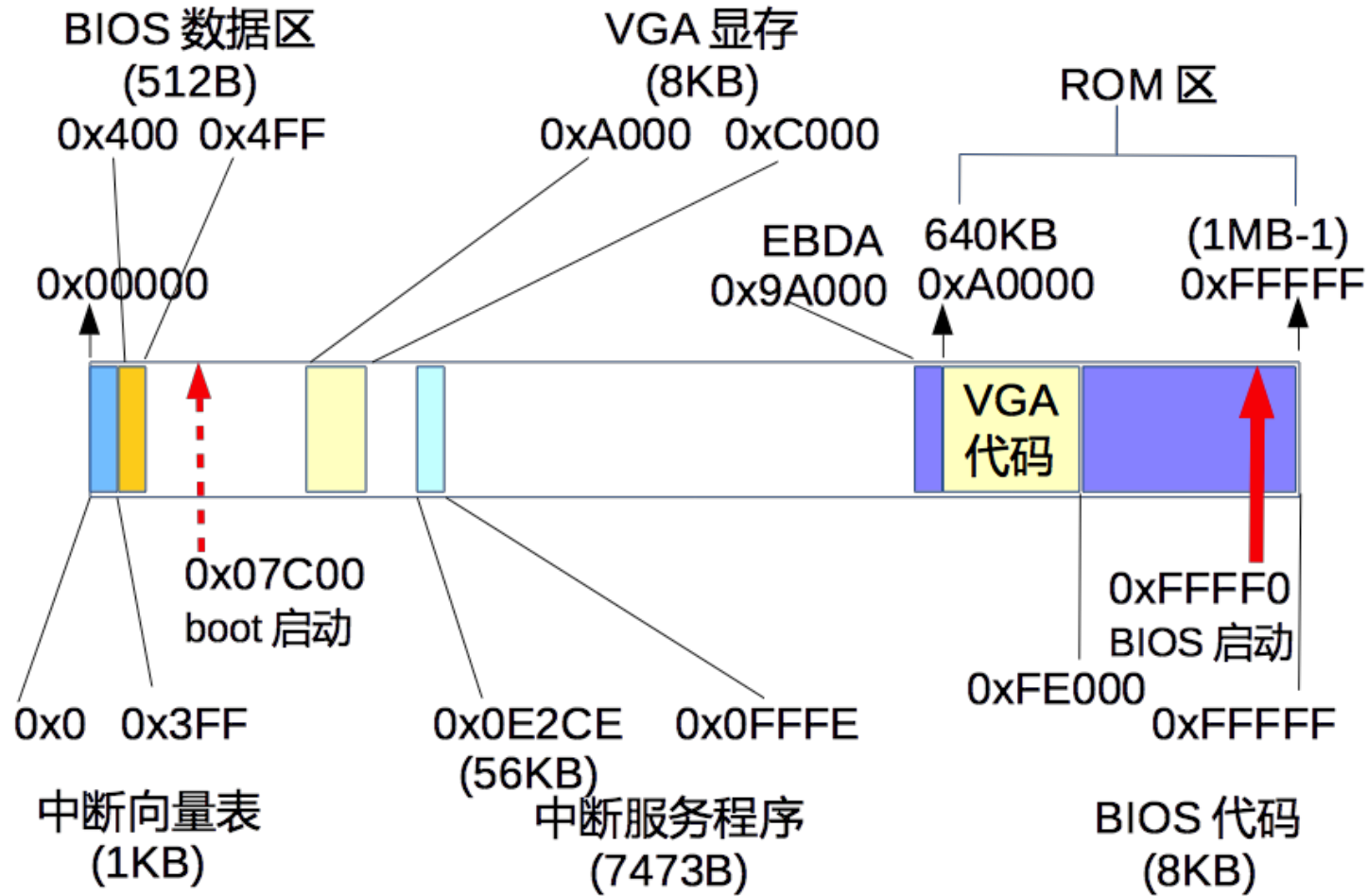
Environment

- 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 FFFFFFFF0h 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



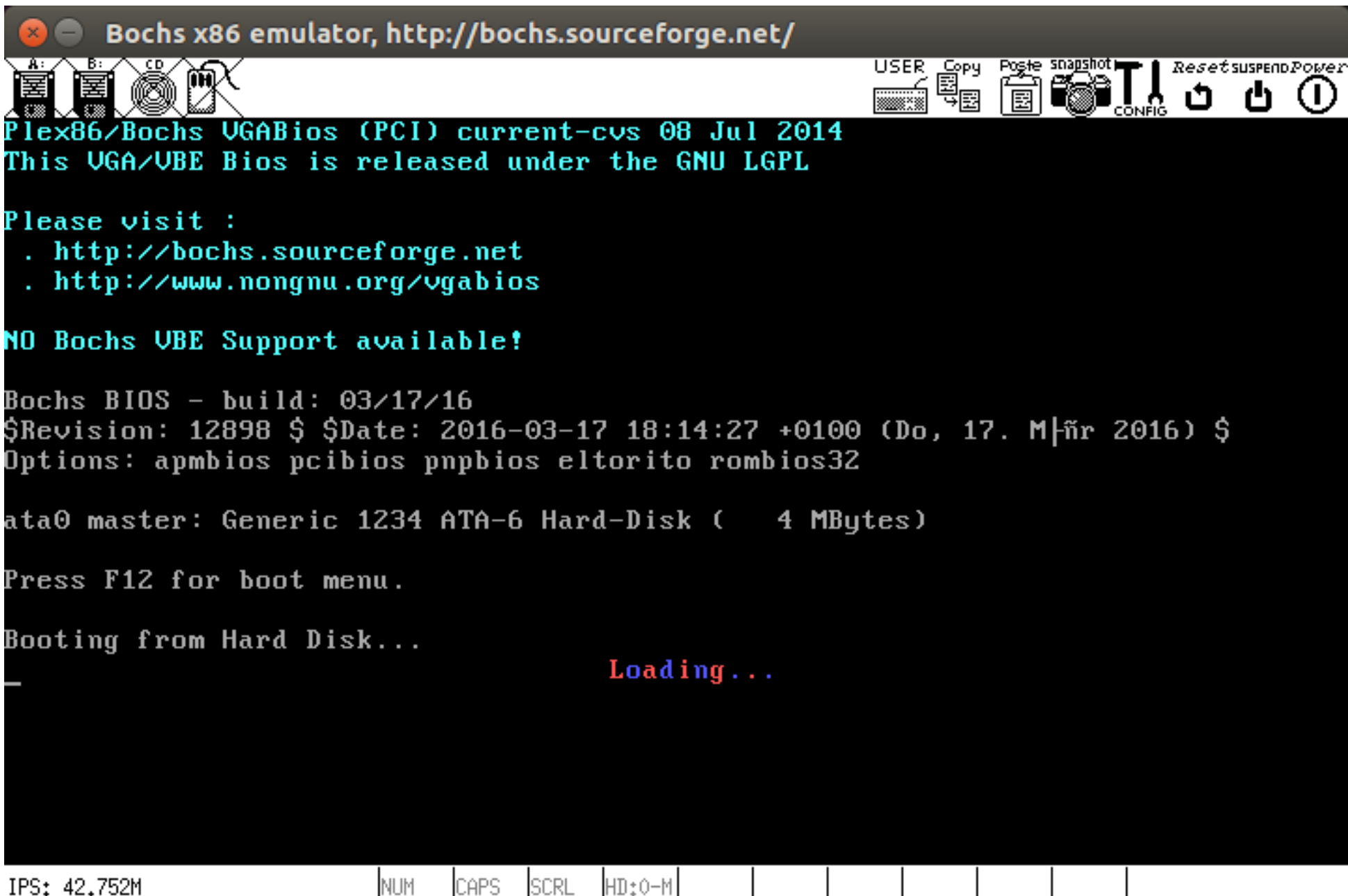
bootstrap

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

0x003~... Disk parameters(used by BIOS)

0xttt~0x1fd Bootstrap program

0x1ff~0x1fe 0xaa55



```
-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
```

```
1 CC      := gcc -pipe
2 AS      := as
3 AR      := ar
4 LD      := ld
5 OBJCOPY := objcopy
6 OBJDUMP := objdump
7 NM      := nm
8 LDFLAGS := -m elf_i386
9
10 all: image
11 boot_objs := boot.o
12 boot.o: boot.S
13     $(CC) -nostdinc -m32 -Os -c -o $@ $<
14 boot: $(boot_objs)
15     $(LD) $(LDFLAGS) -N -e start -Ttext 0x7C00 -o $@.out $^
16     $(OBJCOPY) -S -O binary $@.out $@
17     perl sign.pl $@
18 bochs.img: boot
19     dd if=/dev/zero of=./.bochs.img~ count=10000 2>/dev/null
20     dd if=./boot of=./.bochs.img~ conv=notrunc 2>/dev/null
21     mv ./bochs.img~ ./bochs.img
22 image: bochs.img
23 bochs: image
24     bochs -f bochsrc.txt
25 run: bochs
26 # For deleting the build
27 clean:
28     rm *.o *.out *.asm boot *.log -fr
29     rm bochs.img -fr
30 .PHONY: clean
31
```

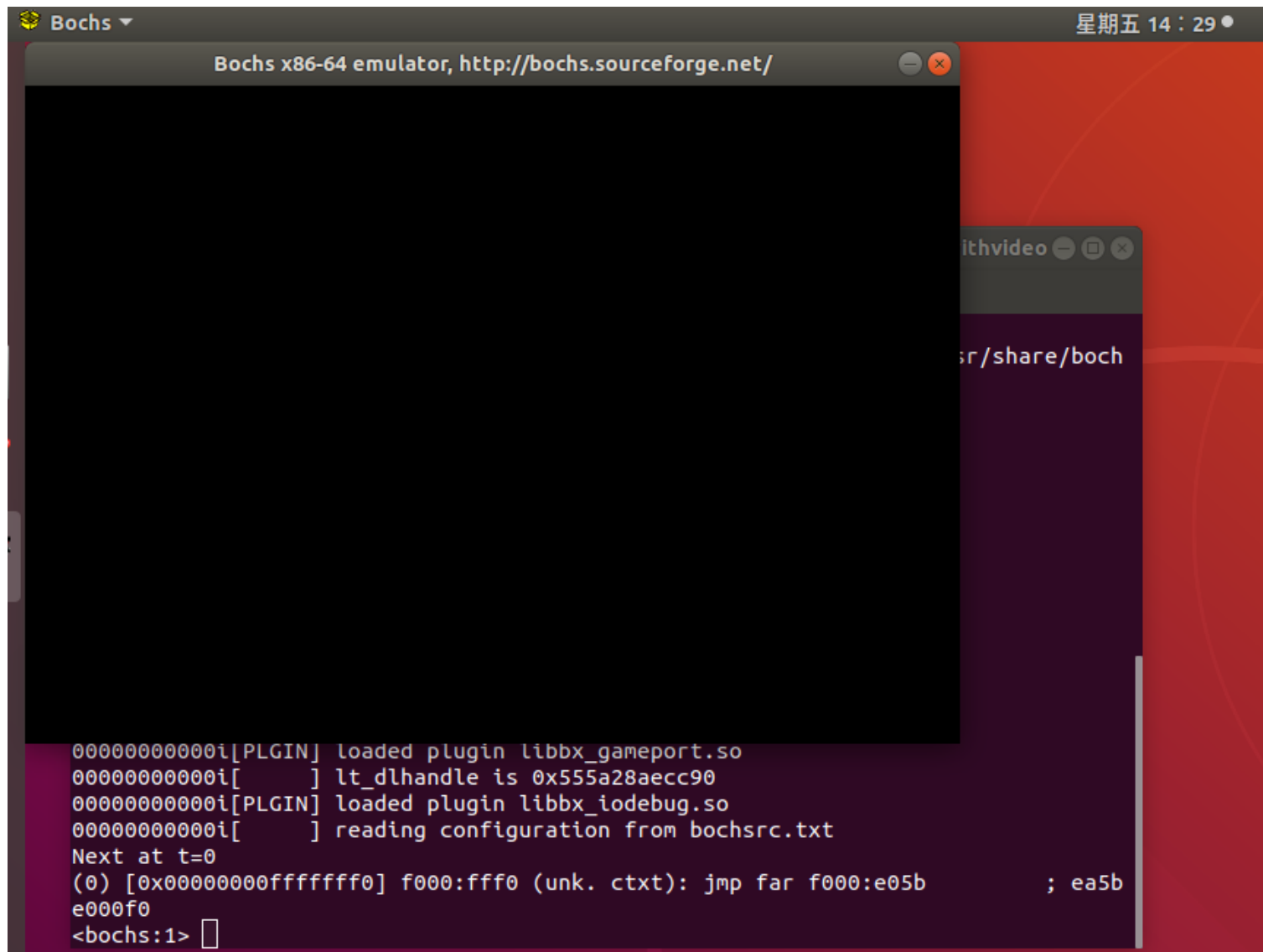

boot.S

```
1  #
2  #the first prog:boot
3  #directly write to video memory 0xb8000~0xbffff
4  #
5  .globl start
6  start:
7      .code16                # Assemble for 16-bit mode
8      cli                    # Disable interrupts
9      cld                    # String operations increment
10
11     xorw    %ax,%ax         # Segment number zero
12     movw    %ax,%ds         # initiate Data Segment
13     movw    %ax,%es         # Extra Segment
14     movw    %ax,%ss         # Stack Segment
15
16     movw    $0xb800,%ax     #text display address
17     movw    %ax,%es
18     movw    $msg1,%si       #source addr:msg1
19     movw    $0xb88,%di      #dest addr:0xb8000+0xb88
20     movw    $0x14,%cx       #count of msg1
21     rep     movsb           #print msg1
22
23 spin:    #loop forever
24     jmp spin
25
26 msg1:    #Loading...
27     .byte 'L',0xc,'o',0x9,'a',0xc,'d',0x9,'i',0xc,'n',0x9,'g',0xc
28     .byte '.',0x9,'.',0xc,'.',0x9
29
30     .org 510
31     .word 0xAA55
```

boot.S

```
4  #
5  .global start
6  start:
7      .code16                # Assemble for 16-bit mode
8      cli                    # Disable interrupts
9      movw    %cs,%ax
10     movw    %ax,%ds        # initiate Data Segment
11     movw    %ax,%es        # Extra Segment
12     movw    %ax,%ss        # Stack Segment
13
14     #get current cursor pos
15     movb    $0x03, %ah      # read cursor pos
16     xor     %bh, %bh
17     int     $0x10
18
19     #pring msg1
20     movw    $19, %cx
21     movw    $0x000c, %bx     # page 0, attribute 0x0c (red)
22                     #0x07 mormal, 0x0c red
23     movw    $msg1, %bp
24     movw    $0x1301, %ax     # write string, move cursor
25     int     $0x10
26
27     spin:    #loop forever
28     jmp spin
29
30     msg1:    #Loading System...
31     .ascii "I am booting system..."
32     .byte 13,10
33
34     .org 510
35     .word 0xAA55
```

make run



The screenshot shows the Bochs x86-64 emulator window. The title bar reads "Bochs x86-64 emulator, http://bochs.sourceforge.net/". The main window is a dark terminal with a purple border. The terminal output shows the following text:

```
000000000000i[PLGIN] loaded plugin libbx_gameport.so
000000000000i[      ] lt_dlhandle is 0x555a28aecc90
000000000000i[PLGIN] loaded plugin libbx_iodebug.so
000000000000i[      ] reading configuration from bochsrc.txt
Next at t=0
(0) [0x00000000ffffffff] f000:fff0 (unk. ctxt): jmp far f000:e05b      ; ea5b
e000f0
<bochs:1> █
```

The prompt is "<bochs:1>".

Press "c"

Bochs x86-64 emulator, <http://bochs.sourceforge.net/>



Plex86/Bochs VGABios (PCI) current-cvs 08 Apr 2016
This UGA/VE 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