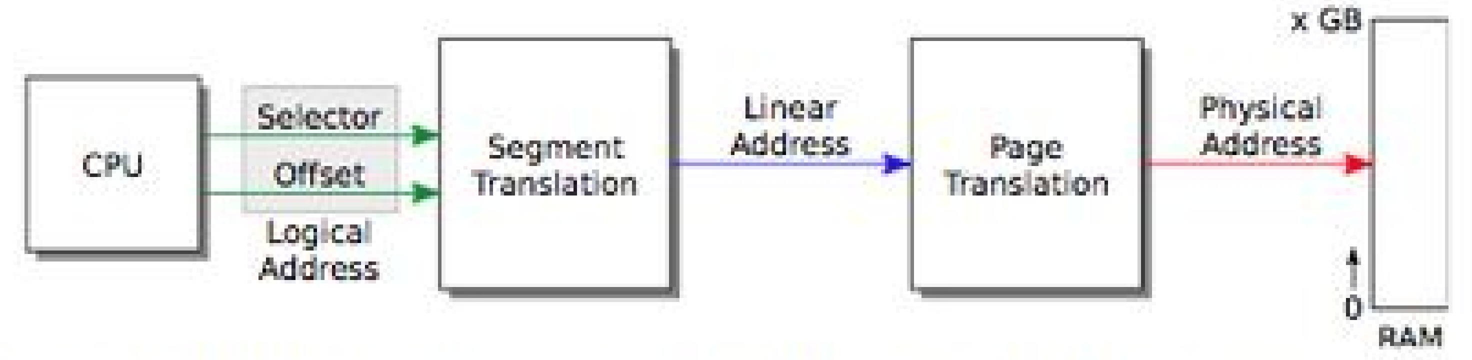
Computer Boot To Protected Mode

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

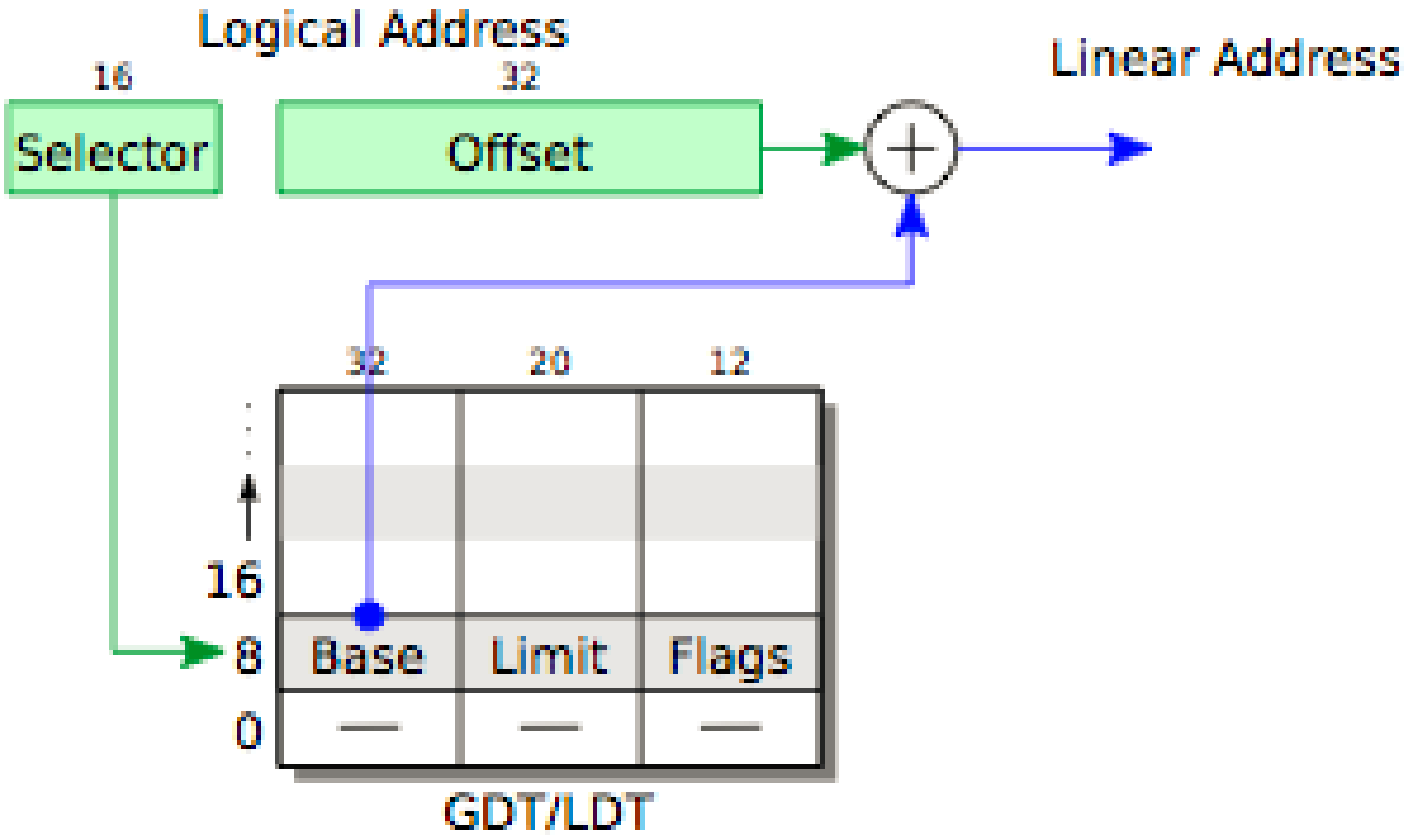
Memory Address

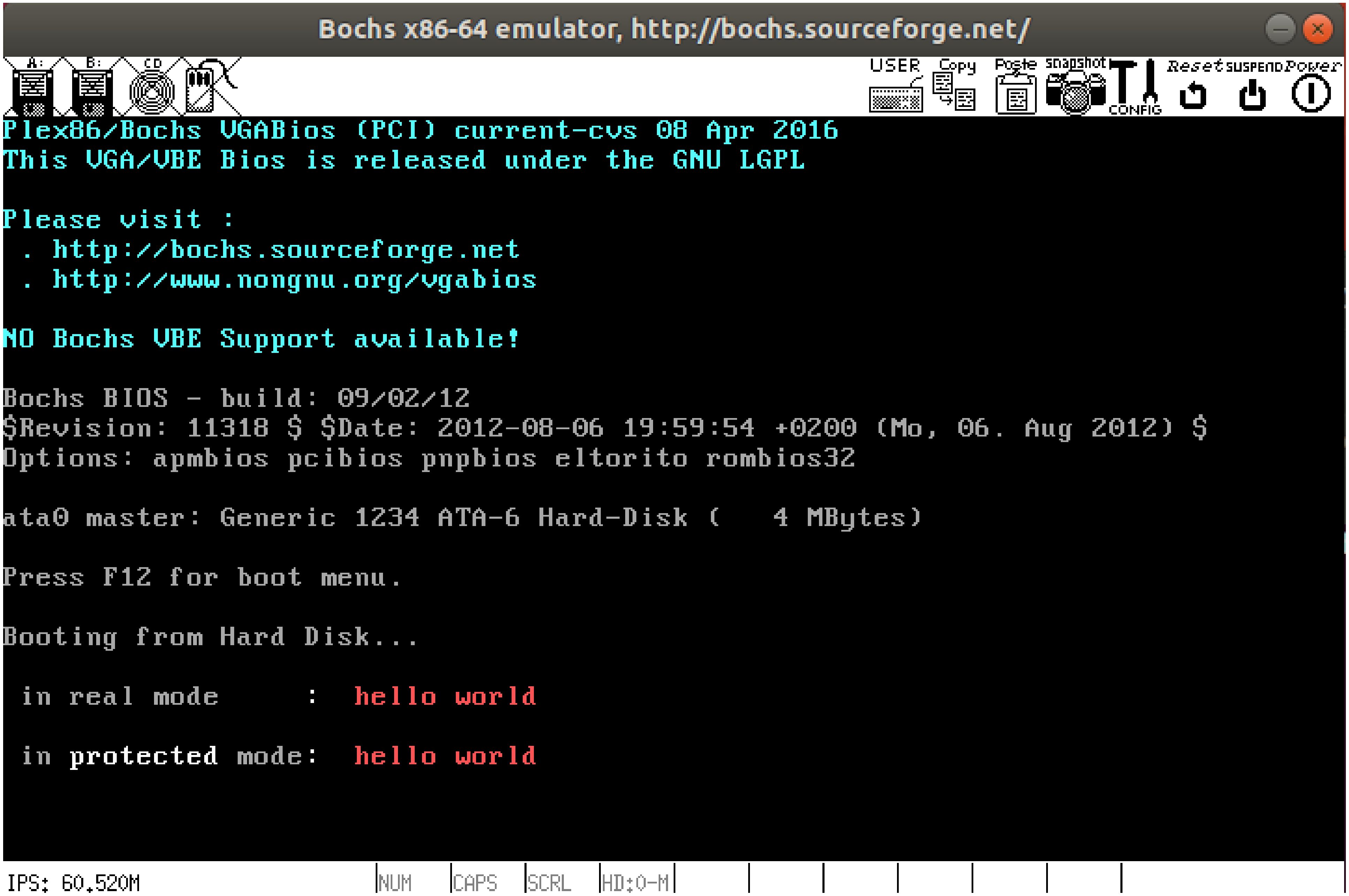


- Logical Address
- Linear Address
- Physical Address

- Real Mode
  - Segment + Offset
  - Address: 20 bits

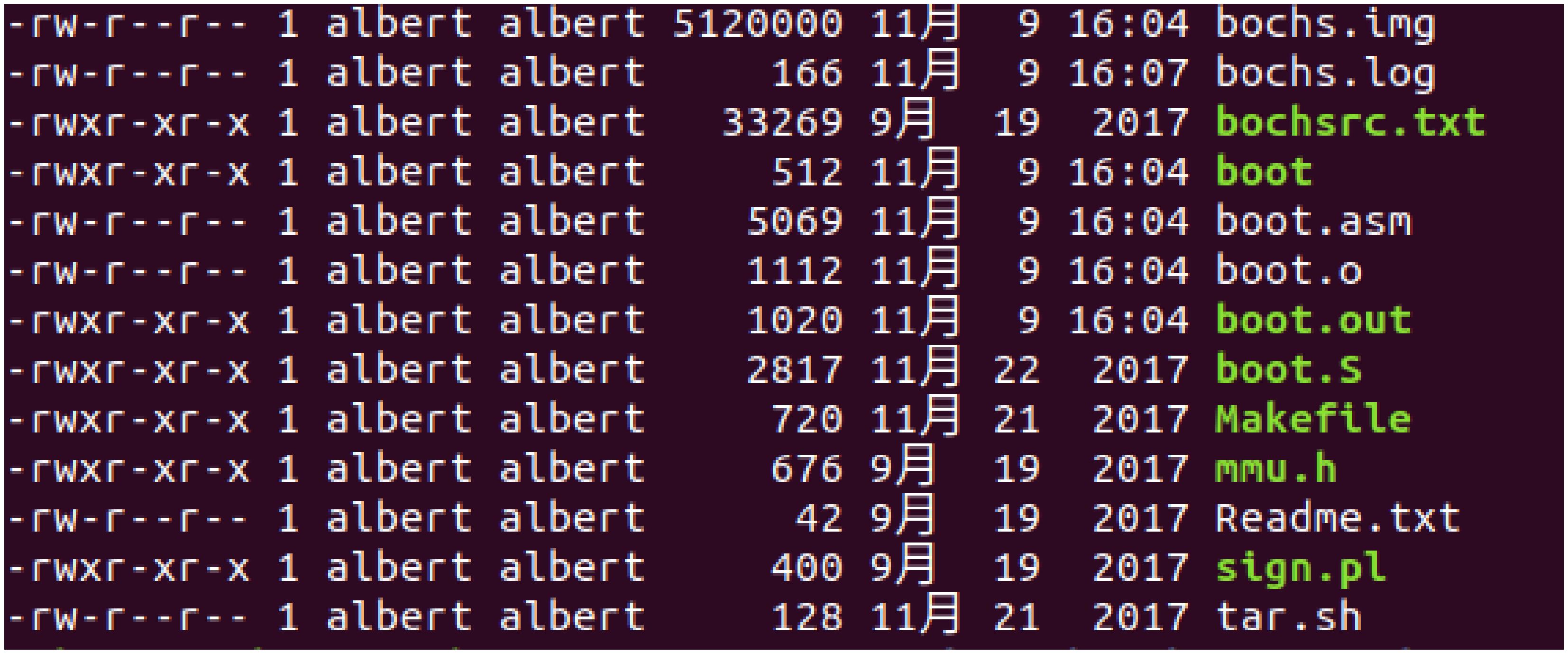
Protected Mode





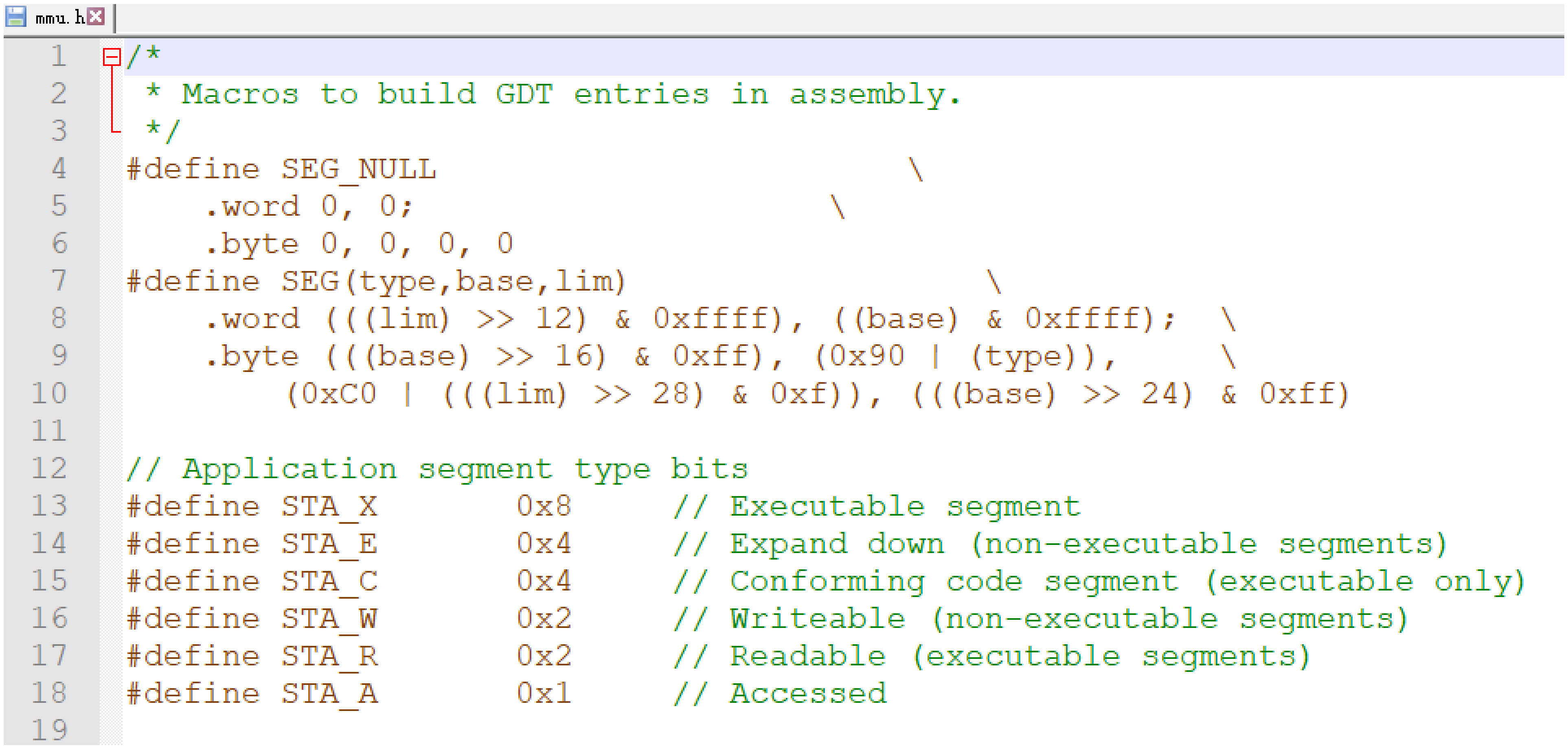
```
Plex86/Bochs VGABios (PCI) current-cvs 08 Apr 2016
This VGA/VBE Bios is released under the GNU LGPL

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...

 in real mode      :  hello world

 in protected mode:  hello world
```

Files



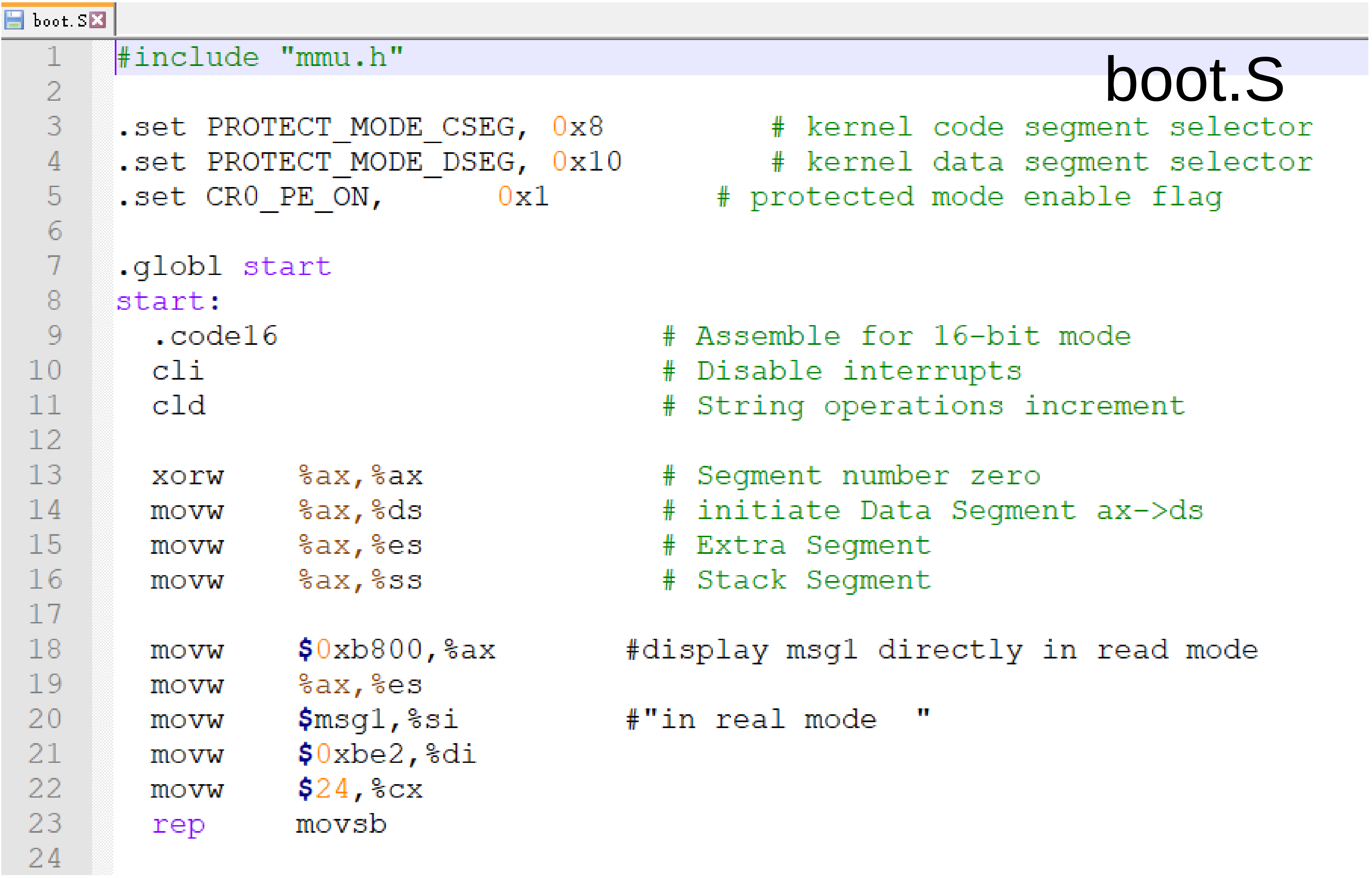
```
-rw-r--r-- 1 albert albert 5120000 11月  9 16:04 bochs.img
-rw-r--r-- 1 albert albert     166 11月  9 16:07 bochs.log
-rwxr-xr-x 1 albert albert   33269 9月  19  2017 bochsrc.txt
-rwxr-xr-x 1 albert albert     512 11月  9 16:04 boot
-rw-r--r-- 1 albert albert    5069 11月  9 16:04 boot.asm
-rw-r--r-- 1 albert albert    1112 11月  9 16:04 boot.o
-rwxr-xr-x 1 albert albert    1020 11月  9 16:04 boot.out
-rwxr-xr-x 1 albert albert    2817 11月 22  2017 boot.S
-rwxr-xr-x 1 albert albert     720 11月 21  2017 Makefile
-rwxr-xr-x 1 albert albert     676 9月  19  2017 mmu.h
-rw-r--r-- 1 albert albert      42 9月  19  2017 Readme.txt
-rwxr-xr-x 1 albert albert     400 9月  19  2017 sign.pl
-rw-r--r-- 1 albert albert     128 11月 21  2017 tar.sh
```

mmu.h



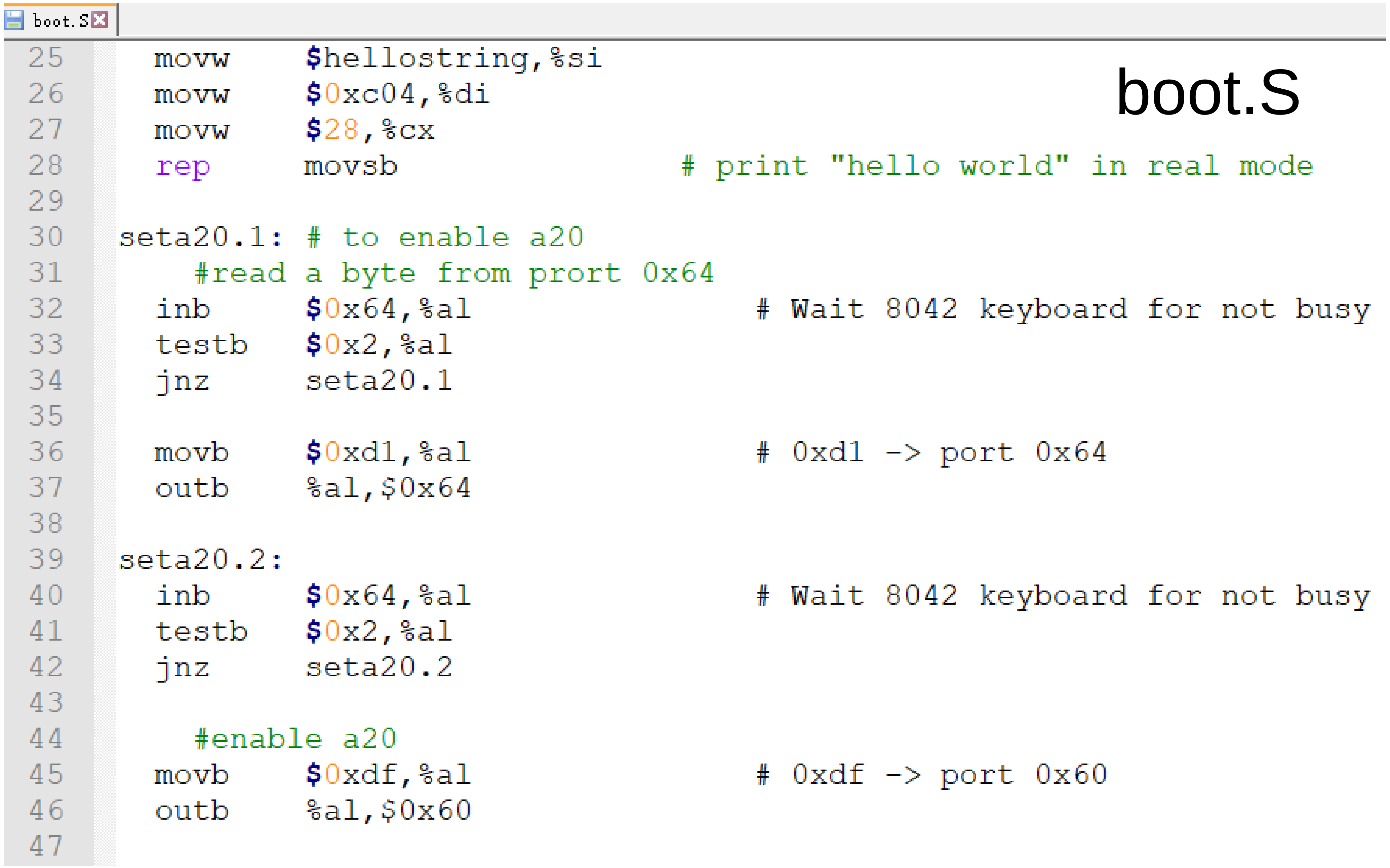
```
/*
 * Macros to build GDT entries in assembly.
 */
#define SEG_NULL                                \
    .word 0, 0;                             \
    .byte 0, 0, 0, 0
#define SEG(type,base,lim)                          \
    .word (((lim) >> 12) & 0xffff), ((base) & 0xffff);  \
    .byte (((base) >> 16) & 0xff), (0x90 | (type)),     \
        (0xC0 | (((lim) >> 28) & 0xf)), (((base) >> 24) & 0xff)

// Application segment type bits
#define STA_X       0x8     // Executable segment
#define STA_E       0x4     // Expand down (non-executable segments)
#define STA_C       0x4     // Conforming code segment (executable only)
#define STA_W       0x2     // Writeable (non-executable segments)
#define STA_R       0x2     // Readable (executable segments)
#define STA_A       0x1     // Accessed

```

boot.S



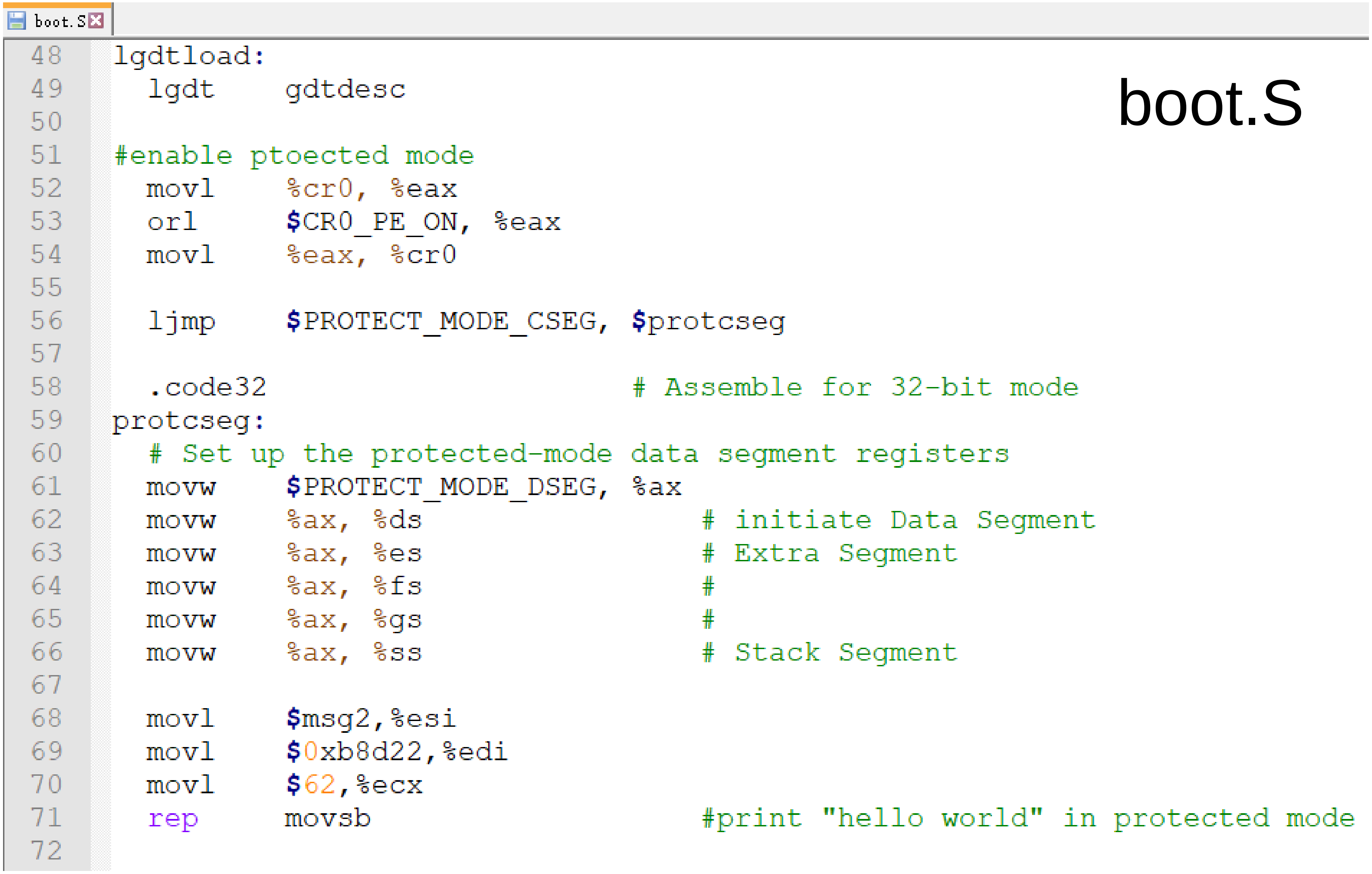
```
#include "mmu.h"

.set PROTECT_MODE_CSEG, 0x8           # kernel code segment selector
.set PROTECT_MODE_DSEG, 0x10          # kernel data segment selector
.set CR0_PE_ON,       0x1           # protected mode enable flag

.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 ax->ds
  movw    %ax,%es                   # Extra Segment
  movw    %ax,%ss                   # Stack Segment

  movw    $0xb800,%ax             #display msg1 directly in read mode
  movw    %ax,%es
  movw    $msg1,%si               #"in real mode  "
  movw    $0xbe2,%di
  movw    $24,%cx
  rep     movsb

```

boot.S



```
  movw    $hellostring,%si
  movw    $0xc04,%di
  movw    $28,%cx
  rep     movsb                   # print "hello world" in real mode

seta20.1: # to enable a20
    #read a byte from prort 0x64
  inb     $0x64,%al                   # Wait 8042 keyboard for not busy
  testb   $0x2,%al
  jnz     seta20.1

  movb    $0xd1,%al                   # 0xd1 -> port 0x64
  outb    %al,$0x64

seta20.2:
  inb     $0x64,%al                   # Wait 8042 keyboard for not busy
  testb   $0x2,%al
  jnz     seta20.2

    #enable a20
  movb    $0xdf,%al                   # 0xdf -> port 0x60
  outb    %al,$0x60
```

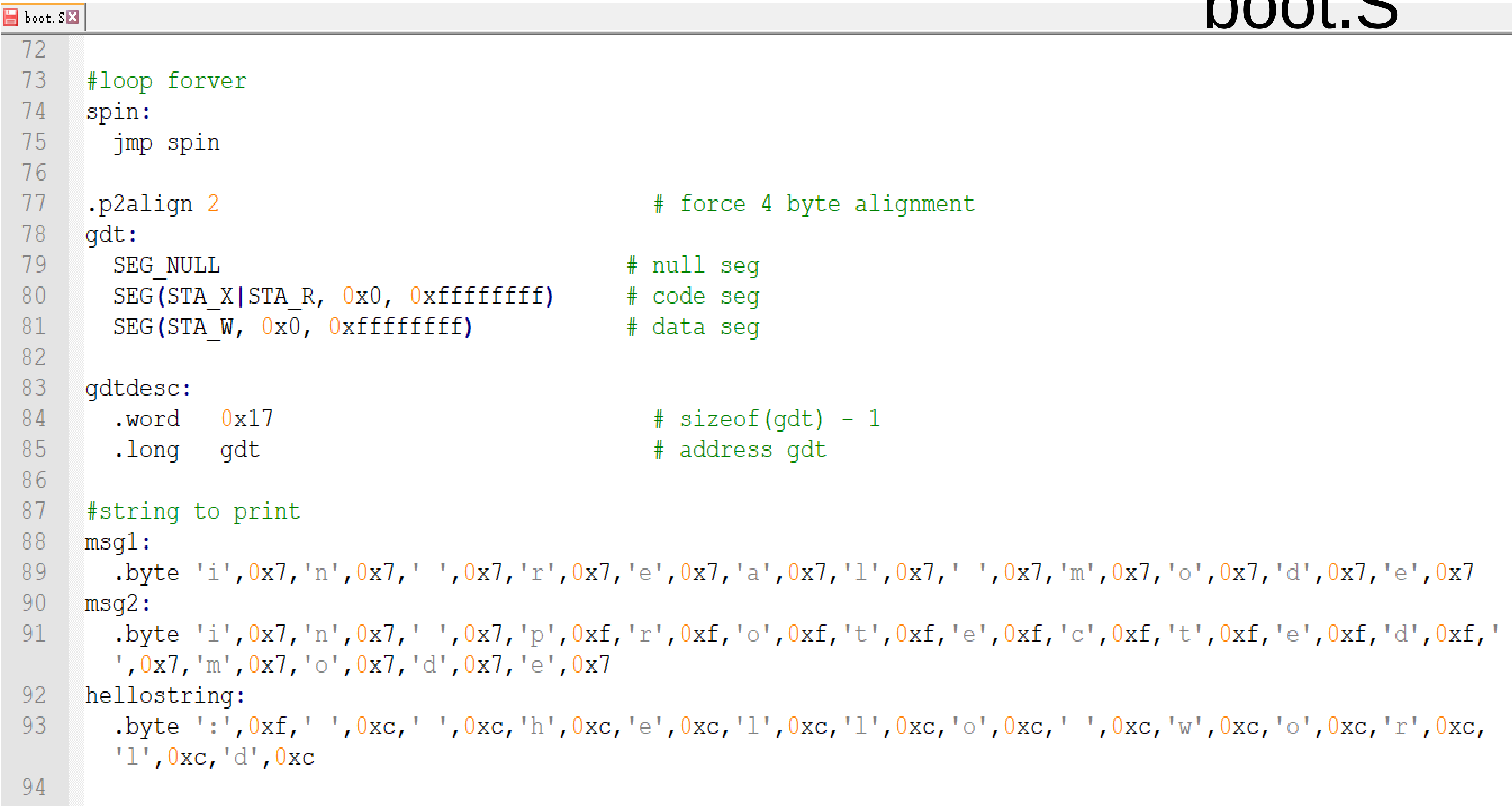
boot.S



boot.S

```
lgdtload:
  lgdt    gdtdesc

#enable ptoected mode
  movl    %cr0, %eax
  orl     $CR0_PE_ON, %eax
  movl    %eax, %cr0

  ljmp    $PROTECT_MODE_CSEG, $protcseg

  .code32                    # Assemble for 32-bit mode
protcseg:
  # Set up the protected-mode data segment registers
  movw    $PROTECT_MODE_DSEG, %ax
  movw    %ax, %ds                   # initiate Data Segment
  movw    %ax, %es                   # Extra Segment
  movw    %ax, %fs                   #
  movw    %ax, %gs                   #
  movw    %ax, %ss                   # Stack Segment

  movl    $msg2,%esi
  movl    $0xb8d22,%edi
  movl    $62,%ecx
  rep     movsb                      #print "hello world" in protected mode
```

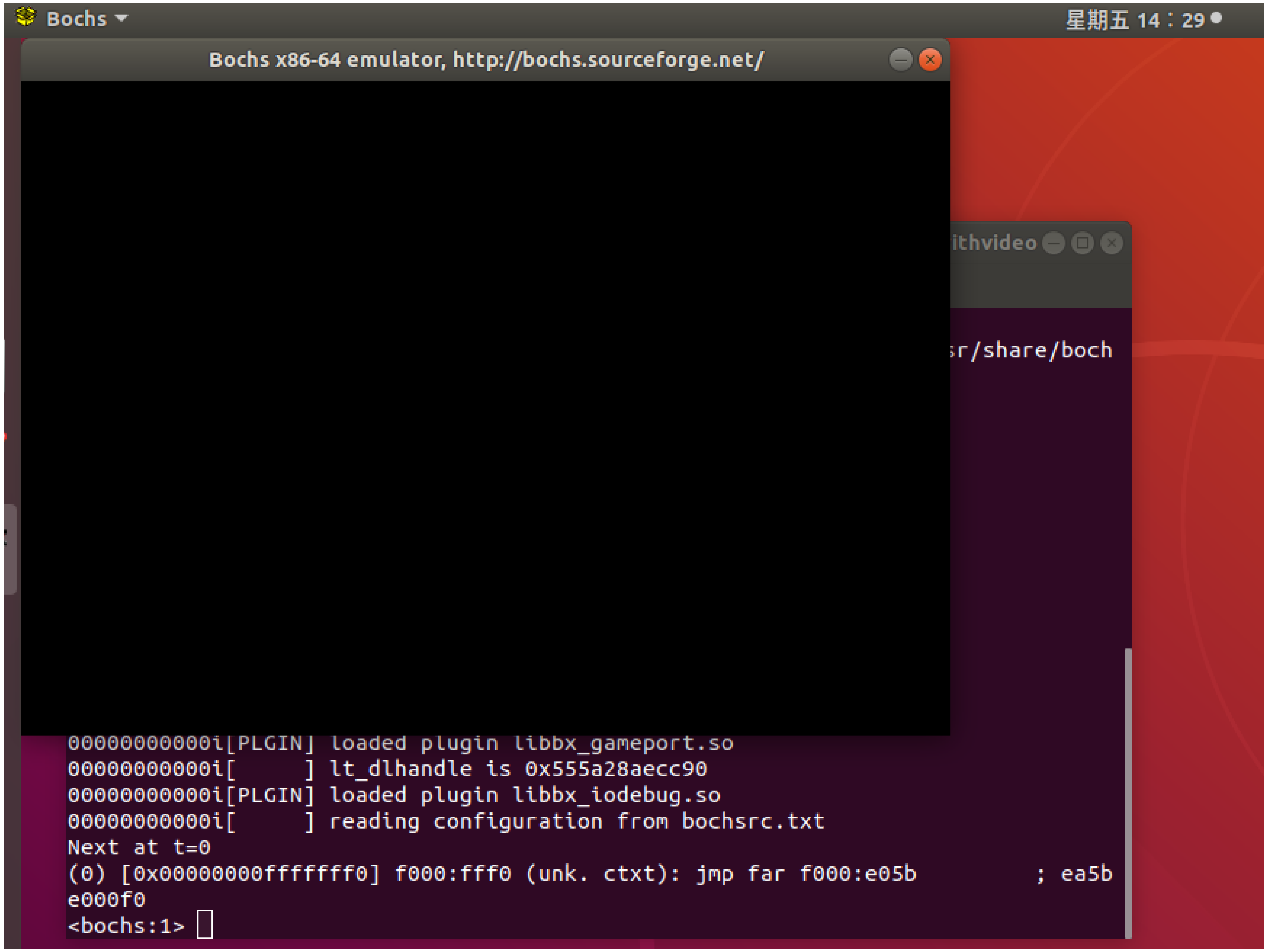
boot.S



boot.S

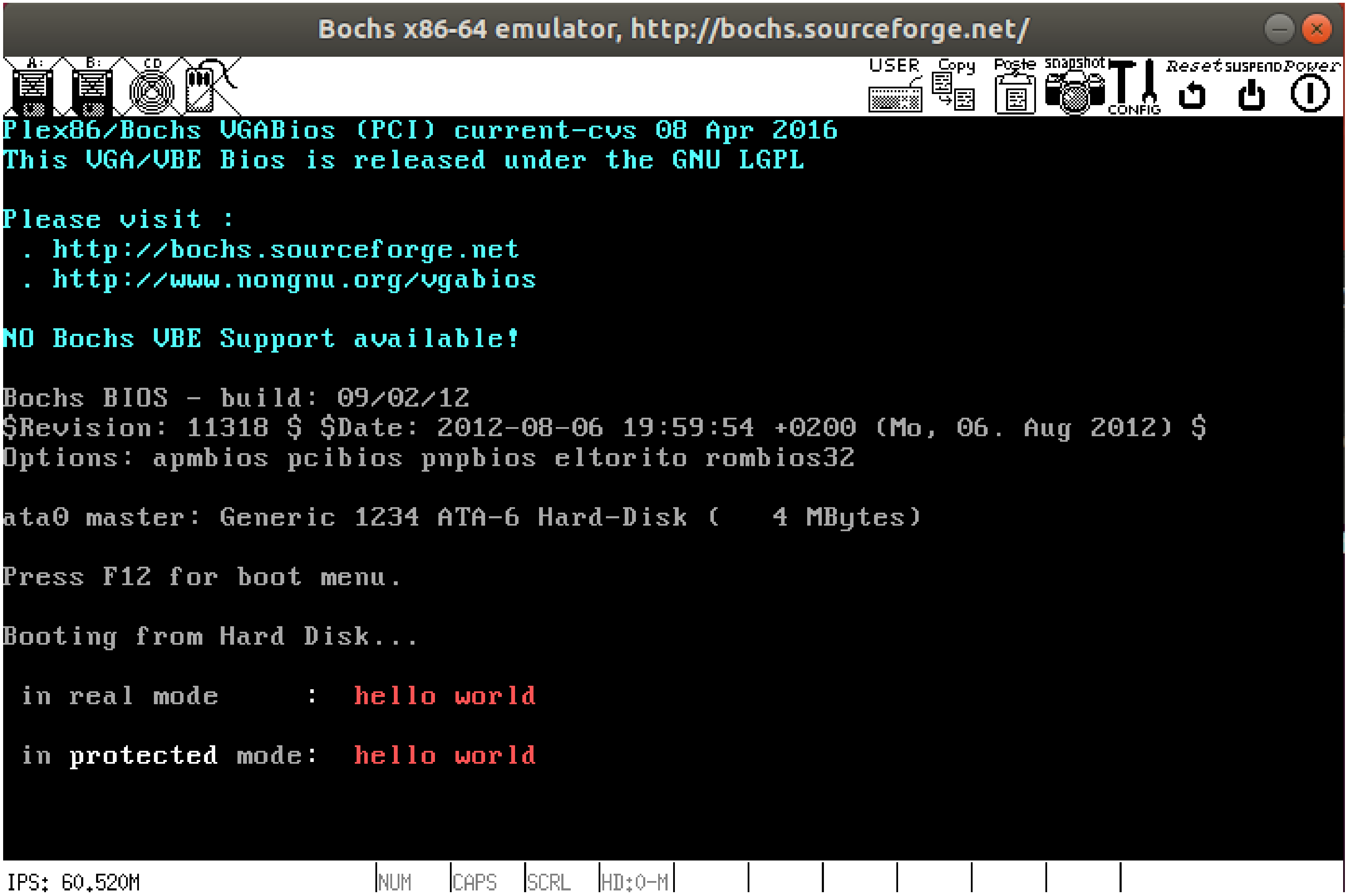
```
#loop forver
spin:
  jmp spin

.p2align 2                                # force 4 byte alignment
gdt:
  SEG_NULL                              # null seg
  SEG(STA_X|STA_R, 0x0, 0xffffffff)     # code seg
  SEG(STA_W, 0x0, 0xffffffff)           # data seg

gdtdesc:
  .word    0x17                           # sizeof(gdt) - 1
  .long    gdt                            # address gdt

#string to print
msg1:
  .byte 'i',0x7,'n',0x7,' ',0x7,'r',0x7,'e',0x7,'a',0x7,'l',0x7,' ',0x7,'m',0x7,'o',0x7,'d',0x7,'e',0x7
msg2:
  .byte 'i',0x7,'n',0x7,' ',0x7,'p',0xf,'r',0xf,'o',0xf,'t',0xf,'e',0xf,'c',0xf,'t',0xf,'e',0xf,'d',0xf,' ',0x7,'m',0x7,'o',0x7,'d',0x7,'e',0x7
hellostring:
  .byte ':',0xf,' ',0xc,' ',0xc,'h',0xc,'e',0xc,'l',0xc,'l',0xc,'o',0xc,' ',0xc,'w',0xc,'o',0xc,'r',0xc,'l',0xc,'d',0xc

```

make run



```
00000000000i[PLGIN] loaded plugin libbx_gameport.so
00000000000i[      ] lt_dlhandle is 0x555a28aecc90
00000000000i[PLGIN] loaded plugin libbx_iodebug.so
00000000000i[      ] reading configuration from bochsrc.txt
Next at t=0
(0) [0x00000000fffffff0] f000:fff0 (unk. ctxt): jmp far f000:e05b         ; ea5b
e000f0
<bochs:1>
```

Press “**c**”



```
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...

 in real mode      :  hello world

 in protected mode:  hello world
```

```
IPS: 60.520M          NUM   CAPS   SCRL   HD:0-M
```

End