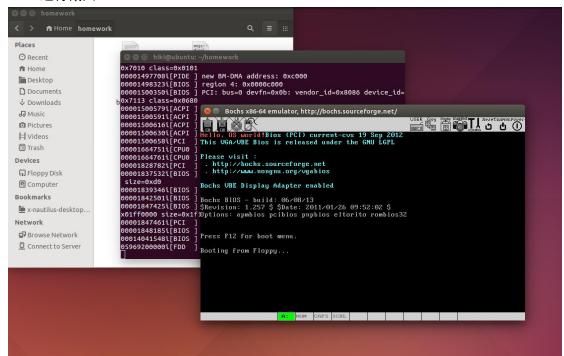
第一部分:Boot

Boot 运行截图:



第二部分: Fibonacci

Fibonacci 运行截图

```
■ ■ hiki@ubuntu: ~/Desktop/SASM
hiki@ubuntu:~/Desktop/SASM$ ./SASMprog.exe
Please enter a series of number: 0 1 4 23 201 5000
The fib numbers are below...
734544867157818093234908902110449296423351
62763028004889570860352531083496840554785287027364574390258244489279372568116632
54475883711527806250329984690249846819800648580083040107584710332687596562185073
54042228679923993261579710597471085709548734282035130747714187501217687430715601
62299658325891377797249738543627776298782295055002604771361083637090900104215369
15488632339240756987974122598603591920306874926755600361865354330444681915154695
74185196007108994401531930012857410766275705479064815275136647552912187721278548
96651017337558985803179844029638737381870001207378241931620113992005474240344408
36239726275765901190914513013217132050988064832024783370583789324109052449717186
8573272397830000207917778045039304398750686626876706788029142697848170225670880
94962311114079089533139023985296550560822285987158823657794699024656757156991872
25655878240668599547496218159297881601061923195562143932693324644219266564617042
93422789337117983238964289528540126387534264046801737892592148358011127805504425
41983822655673959464318033043043268650777429258187573706917261682286488413192314
```

汇编环境说明

汇编系统环境: Ubuntu 14.04 32 位

IDE: SASM

IDE 说明:汇编代码是用一个 Ubuntu 下一个 IDE:SASM 写的,汇编也是通过它,所以汇编过程及其产物会与传统的方法不太一样,但都是差不多的

通过 IDE 汇编的原因:先前都是用 PPT 的方法两行代码汇编运行,但是发现在顺序读内存块的时候会出现一个 bug,就是读着读着 ecx 为 0 了,原先以为是自己逻辑的原因,搞了几个小时无果,后来用 IDE 构建了一下,发现输出是正常的,后来换了环境在命令行下还是一样,索性就用 IDE 的汇编命令来进行构建了。

IDE 汇编命令格式:

| ASM Options | |
|-----------------------------|--|
| Common Colors Build | |
| Mode: | ● x86 ○ x64 |
| Assembler: | NASM ○ GAS ○ FASM ◎ MASM |
| Assembly options: | -g -f elf32 \$SOURCE\$ -l \$LSTOUTPUT\$ -o \$PROGRAM.OBJ\$ |
| Linking options: | \$PROGRAM.OBJ\$ \$MACRO.OBJ\$ -g -o \$PROGRAM\$ -m32 |
| Assembler path: | nasm |
| Linker path: | gcc |
| Object file name: | program.o |
| Build in current directory: | |
| Disable linking: | |

第三部分:代码阅读

| Write string (EGA+, meaning PC AT minimum) | AH=13h | AL = Write mode, BH = Page Number, BL = Color, CX = String length, DH = Row, DL = Column, ES:BP = Offset of string |
|--|--------|--|
|--|--------|--|

当 AH=13h 时, int 10h 中断表示写操作, mov bp, ax 指令将 ax, 即 string 的相对地址 (offset of string) 传到 bp, int 10h 即可取到 BootMessage。

ax 里面的值表示 BootMessage 所在内存中的绝对地址。

相关实验:

对 boot.bin 进行反汇编,可以看到反汇编的相应结果为 mov ax, 0x7c1e, 可知 ax 为绝对地址

Boot 反汇编代码图:

