Notes

# x86体系结构

## 8086指令集

|  |  |
| --- | --- |
| 指令 | 作用 |
| pushad | 好多参数压栈 |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

# Linux环境搭建

sudo apt-get install dosemu

sudo apt-get install nasm

在Ubuntu中执行dosemu，C盘的位置在home目录下.dosemu中。

编译的命令可以使用：

nasm hello.asm –fbin –o hello.exe

然后直接在dosemu中执行即可

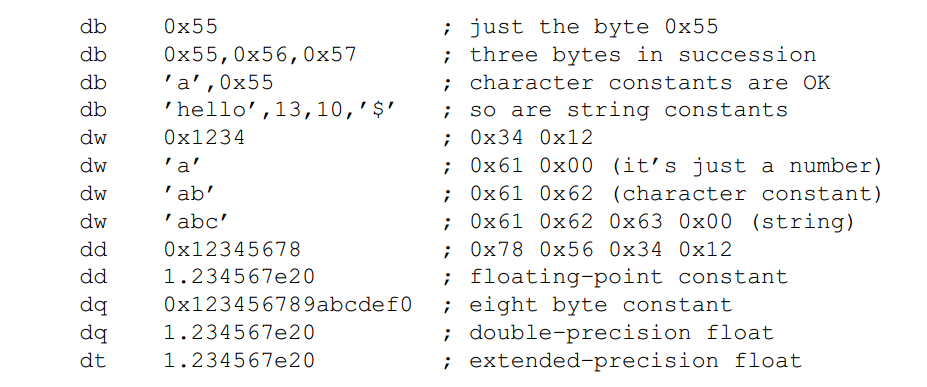
# 参考链接

* <http://stanislavs.org/helppc/idx_interrupt.html> Interrupt Service DOS/BIOS/EMS/Mouse
* <http://spike.scu.edu.au/~barry/interrupts.html> DOS Function
* <http://cs.lmu.edu/~ray/notes/nasmtutorial/> NASM Examples
* <https://en.wikipedia.org/wiki/Intel_8086> 8086处理器介绍

# Nasm

Nasm汇编语言

<http://jingtao.freeshell.org/assembly/Nasm.html> NASM源码阅读笔记



Nasm功能需求

1. nasm多文件编程 √
2. 函数封装调用

# tinyOS

Linux-0.11

MIT

# Check List

1. 申请时钟中断来实时更新屏幕信息。

# Issue List

## CPUID

<http://www.cnblogs.com/zyl910/archive/2012/05/14/dos16_getcpuid.html>

## E820

<http://wiki.osdev.org/Detecting_Memory_(x86)>

<http://blog.csdn.net/gxfan/article/details/2962236>

detect\_memory\_e820() –arch/x86/boot/memory.c

intcall() --arch/x86/boot/bioscall.S

## DOS CPUID

<http://www.cnblogs.com/zyl910/archive/2012/05/14/dos16_getcpuid.html>

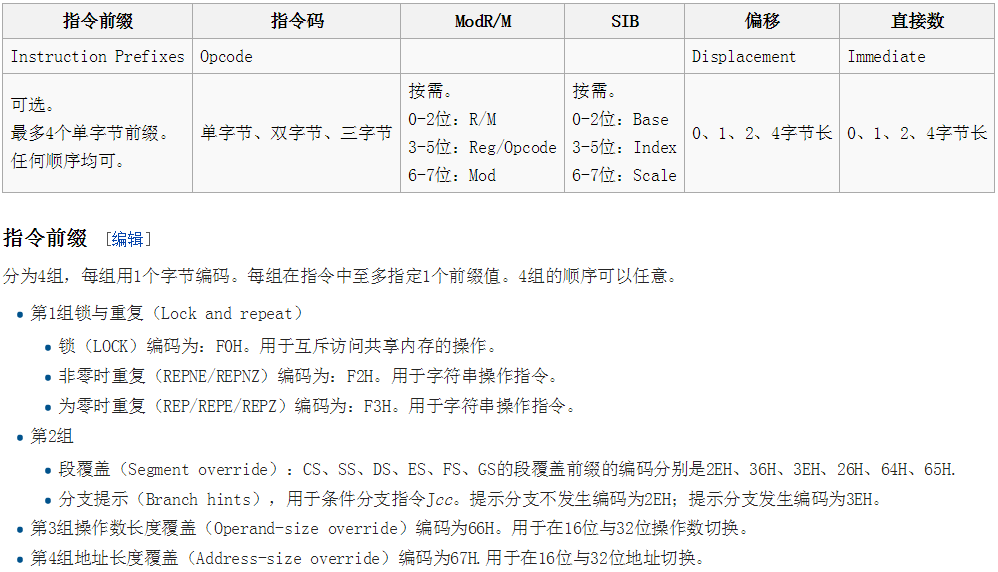
operand size override: prefix 66H

<http://www.mouseos.com/x64/doc3.html>

<https://zh.wikipedia.org/wiki/X86>

<http://wiki.osdev.org/X86-64_Instruction_Encoding>

<http://www.mouseos.com/x64/doc3.html>



原子操作指令与指令前缀有关：

<http://www.cnblogs.com/fanzhidongyzby/p/3654855.html>

## DOS菜单状态机设计

void update\_alt\_f\_menu()

{

flag=1;

  key=up;

do{

//update flag

switch(key)

{

         case up:

                   If(flag!=0) flag--;   else flag=3;

         case down:

                   If(flag!=3) flag++;  else flag=0;

}

     // update current line

if( flag != 0 )

  set\_current\_background();

else

set\_last\_line();

//update next line

if( flag != 3 )

  set\_next\_background();

else

set\_first\_line();

} while( key=read\_key() != “ESC” )

return;

}

## 菜单循环滚动

* F6中需要处理上翻下翻的边界问题，
* ALT+F中需要处理菜单上下翻边界的问题，
* ALT+R中需要处理菜单上下翻边界的问题，
* 主界面中需要处理上下左右的边界问题。

这4个问题的处理模型基本相同，以下面的模型为例。

假设有一个flag记录当前“坐标”，1,2,3是我们要显示的菜单。

|  |
| --- |
| Flag = 1 |
| Flag = 2 |
| Flag = 3 |

菜单每一次交互分为3步：

1. 更新Flag
2. 设置当前行的背景
3. 设置下一行的背景

可以拓展一下想象力，将菜单上下各拓展一格，

其中第0，4是真实不存在的，只是逻辑存在，菜单变为下图。

|  |
| --- |
| Flag = 0 |
| Flag = 1 |
| Flag = 2 |
| Flag = 3 |
| Flag = 4 |

可以将伪代码写为

void update\_alt\_f\_menu()

{

flag=1;

  key=up;

do{

//update flag

switch(key)

{

         Case up:

                   If(flag!=0) flag--;

         Case down:

                   If(flag!=3) flag++;

}

     // update current line

if( flag != 0 )

  set\_current\_background()

//update next line

if( flag != 3 )

  set\_next\_background()

} while( key=read\_key() != “ESC” )

return;

}

假设是2维的

|  |  |  |  |
| --- | --- | --- | --- |
| y=0, x=0; | 0,1 | 0,2 | 0,3 |
| 1,0 | 1,1 | 1,2 | 1,3 |
| 2,0 | 2,1 | 2,2 | 2,3 |
| 3,0 | 3,1 | 3,2 | 3,3 |

假设显示数据为红色字体所示：

#define up 1

#define down 2

#define left 3

#define right 4

Struct flag

{int x, int y};

void update\_main\_window()

{

         Struct Struct flag = {1,1};

         Int key = left;

         do{

         //update flag & direction

         switch(key)

         {

                   Case up:

                            If(flag.y != 0)

                                     flag.y --;

                            direction = up;

Case down:

                            If(flag.y != 2)

                                     flag.x ++;

                            direction = down;

                   Case left:

                            If(flag.x !=0)

                              flag.x--;

                            direction = left;

                   Case right:

                            If(flag.x !=2)

                              flag.x++;

                            direction = right;

}

//update current field

If( (flag.x != 0 && direction != left)  ||  flag.y !=0 && direction!=up)

           set\_current\_field()

         //set next field

If( (flag.x != 2 && direction != right)  ||  flag.x !=2 && direction!=down)

set\_next\_field()

}while(key=read\_key() != “ESC” | “ALT+R” | “ALT+F”)

}