* 1. 实验目的：
* *实现键盘输入*
* *继续使用10h号中断来完成一些内容*
* *熟悉使用16h号中断—键盘控制中断的功能，包括获取键盘输入状态，以及如何处理。*

**bootasm.S 文件代码流程：**

1.清空屏幕（在原先的基础上做了一部分改动）

2.显示一个字符串

3.键盘输入（设置了一些控制字符的汇编处理）

1：清空屏幕（只介绍新添加的内容）

movb $0x07, %ah #功能号07，这里调用07和06结果一样，上卷下卷都是一样的。

movb $0, %al #上卷全部行，即清屏

movb $0, %ch #左上角行

movb $0, %cl #左上角列

movb $24, %dh #右下角行

movb $79, %dl #右下角列

movb $0x07, %bh #卷入行的属性

int $0x10

movb $0x02, %ah #这时候光标的位置是在中间的位置，需要重置一下位置

movw $0x00, %dx #光标的行与列都是0和0

movb $0x00, %bh #页号是0

int $0x10

ret

一开始时候清屏结束以后，光标停留在中间的位置。调用10h中断的02h号功能来重置光标。

设置dx=0，表示光标的行号与列号为0，设置bh=0表示页号为0。重置结束以后可以在屏幕的左上角开始显示光标。

另外有关清屏，调用06h与07h号功能获取一样的结果。

2：显示字符串

movw $BootMessage, %ax

movw %ax, %bp #ES:BP = 串地址

movw $16, %cx #CX = 串长度

movw $0x1301, %ax #AH = 13, AL = 01h,这里面是调用了10h号中断的13号功能

movw $0x00c, %bx #页号为0(BH=0) 黑底红字(BL=0Ch,高亮)

movb $0, %dl #表示起始行号

int $0x10 #调用10h号中断

call Fill

ret

BootMessage: .ascii "hello, BLCU OS,My name is ChenPeng"

详细参见实验0.1

3：键盘输入Typing:

movb $0x00, %ah #ah = 00h，从键盘读取字符，读取的内容保存到al中

int $0x16 #调用BIOS16h中断，是键盘控制中断

#比较字符的ascii码

cmpb $0x08, %al #比较输入的字符是不是Backspace键

jz Back

cmpb $0x0D, %al #比较输入的字符是不是Enter键

jz Enter

#上下左右键, 包括其他的光标移动键和功能键, 都是没有 ascii 码的, 只有通过扫描码来判别究竟是哪个键

cmpb $0x48, %ah #比较输入的字符是不是Up键

jz Up

cmpb $0x50, %ah #比较输入的字符是不是Down键

jz Down

cmpb $0x4b, %ah #比较输入的字符是不是Left键

jz Left

cmpb $0x4d, %ah #比较输入的字符是不是Right键

jz Right

movb $0x0e, %ah #ah = 0eh，在光标处显示字符。光标随着字符而移动

int $0x10 #调用BIOS10h中断，显示字符

call Typing

ret

有关键盘的BIOS中断服务程序是16h号中断。INT $0x15h来进入。

首先先从键盘读取一个字符。功能号是00H，读取的数据（ASCII码）保存到了AL寄存器中。

之后判别一下寄存器中保存的输入是不是特殊控制字符的ASCII码

Backspace ：跳转到Back

Enter：跳转到Enter

若不符合以上情况。先将AH置为0x0e，表示准备调用10h中断的0Eh号服务：在光标处显示字符。

再通过INT $0x10h，调用显示中断。

最终再call Typing 实现循环输入。

在Back中：

movb $0x0e, %ah #ah = 0eh，在光标处显示字符。光标随着字符而移动

movb $0x08, %al #先后退一个字符

int $0x10

movb $0x20, %al #输出空格

int $0x10

movb $0x08, %al #再后退一个字符

int $0x10

call Typing

首先后退一个字符。让光标停留在准备删除的字符前面

在输出一个空格，让空格来覆盖准备删除的字符，此处光标停留在空格处。

为了能够实现连续删除， 需要将光标停留在空格前面。

最后再call Typing 实现循环输入。

在Enter中：

movb $0x0e,%ah #ah = 0eh，在光标处显示字符。光标随着字符而移动

movb $0x0a,%al #换行，这时候光标在垂直位置上保持当前值。

int $0x10

movb $0x0d,%al #归位

int $0x10

call Typing

首先先将0x0a存入到al中，实现换行。但是这个时候换行后的光标在垂直位置上没有改变。

在将0x0d存入到al中，实现归位，将光标停留在这一行的最初始的位置。

最后再call Typing 实现循环输入。

这里只是实现了Backspace与Enter，但是其他一些键盘控制没有实现，up,down,left,right等等。

**练习：**

实现键盘控制，如 up,down,left,right。

**扩展练习（可选）：**

实现键盘输入ctrl+alt+delete时重启。

**补充知识：**

1.**INT 16H 中断简介**

|  |  |  |  |
| --- | --- | --- | --- |
| AH | 功 能 | 调用参数 | 返回参数 / 注释 |
| 00H和10H | 从键盘读入字符 | AH＝00H——读键盘  ＝10H——读扩展键盘，可根据0000:0496H单元的内容判断：扩展键盘是否有效 | AH＝键盘的扫描码 AL＝字符的ASCII码 |
| 01H和11H | 读取键盘状态 | AH＝01H——检查普通键盘  ＝11H——检查扩展键盘 | ZF＝1——无字符输入，否则，AH＝键盘的扫描码，AL＝ASCII码。 |
| 02H和12H | 读取键盘标志 | AH＝02H——普通键盘的移位标志  ＝12H——扩展键盘的移位标志 | AL＝键盘标志(02H和12H都有效)，其各位之值为1时的含义如下：  位7—INS开状态  位3—ALT键按下 位6—CAPS LOCK开状态位2—CTRL键按下 位5—NUM LOCK开状态位1—左SHIFT键按下 位4—SCROLL LOCK开状态  位0—右SHIFT键按下 AH＝扩展键盘的标志(12H有效)，其各位之值为1时的含义如下： 位7—SysReq键按下  位3—右ALT键按下 位6—CAPS LOCK键按下位2—右CTRL键按下 位5—NUM LOCK键按下位1—左ALT键按下 位4—SCROLL键按下  位0—左CTRL键按下 |
| 03H | 设置重复率 | AH＝03H 对于PC/AT和PS/2：  AL＝05H BH＝重复延迟 BL＝重复率 对于PCjr：  AL＝00H——装入缺省的速率和延迟  ＝01H——增加初始延迟  ＝02H——重复频率降低一半  ＝03H——增加延迟和降低一半重复频率  ＝04H——关闭键盘重复功能 |  |
| 04H | 设置键盘点击 | AH＝04H  AL＝00H——关闭键盘点击功能  ＝01H——打开键盘点击功能 |  |
| 05H | 字符及其扫描码进栈 | AH＝05H CH＝字符的描述码 CL＝字符的ASCII码 | CF＝1——操作成功，AL＝00H，否则，AL＝01H |

* 1. 2. BIOS 中断向量表

|  |  |  |
| --- | --- | --- |
| 中断 | 描述 | |
| INT 00h | CPU: 除零错,或商不合法时触发 | |
| INT 01h | CPU: 单步陷阱,TF标记为打开状态时,每条指令执行后触发 | |
| INT 02h | CPU:[非可屏蔽中断](http://zh.wikipedia.org/w/index.php?title=Non-maskable_interrupt&action=edit&redlink=1), 如[开机自我测试](http://zh.wikipedia.org/wiki/開機自我測試)时发生内存错误触发。 | |
| INT 03h | CPU: 第一个未定义的中断向量, 约定俗成仅用于调试程序 | |
| INT 04h | CPU: 算数溢出。通常由INTO指令在置溢出位时触发。 | |
| INT 05h | 在按下Shift-[Print Screen](http://zh.wikipedia.org/wiki/Print_Screen)或BOUND指令检测到范围异常时触发。 | |
| INT 06h | CPU: 非法指令。 | |
| INT 07h | CPU: 没有[数学协处理器](http://zh.wikipedia.org/wiki/8087协处理器)时尝试执行浮点指令触发。 | |
| INT 08h | IRQ0: 可编程中断控制器每 55 毫秒触发一次，即每秒 18.2 次。 | |
| INT 09h | IRQ1: 每次键盘按下、按住、释放。 | |
| INT 0Ah | IRQ2: | |
| INT 0Bh | IRQ3: [COM2/COM4](http://zh.wikipedia.org/wiki/串口)。 | |
| INT 0Ch | IRQ4: [COM1/COM3](http://zh.wikipedia.org/wiki/串口)。 | |
| INT 0Dh | IRQ5: 硬盘控制器（PC/XT 下）或 [LPT2](http://zh.wikipedia.org/wiki/并口)。 | |
| INT 0Eh | IRQ6: 需要时由[软碟控制器](http://zh.wikipedia.org/wiki/軟碟控制器)呼叫。 | |
| INT 0Fh | IRQ7: [LPT1](http://zh.wikipedia.org/wiki/并口)。 | |
| [INT 10](http://zh.wikipedia.org/wiki/INT_10)h | | 显示服务 - 由BIOS或操作系统设定以供软件调用。   |  |  |  | | --- | --- | --- | | AH=00h | | 设定显示模式 | | AH=01h | 设定游标形态 | | | AH=02h | 设定游标位置 | | | AH=03h | 获取游标位置与形态 | | | AH=04h | 获取光笔位置 | | | AH=05h | 设定显示页 | | | AH=06h | 清除或卷轴画面(上) | | | AH=07h | 清除或卷轴画面(下) | | | AH=08h | 读取游标处字符与属性 | | | AH=09h | 更改游标处字符与属性 | | | AH=0Ah | 更改游标处字符 | | | AH=0Bh | 设定边界颜色 | | | AH=0Eh | 在TTY模式下写字符 | | | AH=0Fh | 取得目前显示模式 | | | AH=13h | 写字符串 | | |
| INT 11h | 返回设备列表。 | |
| INT 12h | 获取常规内存容量。 | |
| [INT 13](http://zh.wikipedia.org/w/index.php?title=INT_13&action=edit&redlink=1)h | | 低阶磁盘服务。   |  |  |  | | --- | --- | --- | | AH=00h | | 复位磁盘驱动器。 | | AH=01h | 检查磁盘驱动器状态。 | | | AH=02h | 读扇区。 | | | AH=03h | 写扇区。 | | | AH=04h | 校验扇区。 | | | AH=05h | 格式化磁道。 | | | AH=08h | 取得驱动器参数。 | | | AH=09h | 初始化硬盘驱动器参数。 | | | AH=0Ch | 寻道。 | | | AH=0Dh | 复位硬盘控制器。 | | | AH=15h | 取得驱动器类型。 | | | AH=16h | 取得软驱中盘片的状态。 | | |
| INT 14h | | 串口通信例程。   |  |  |  | | --- | --- | --- | | AH=00h | | 初始化串口。 | | AH=01h | 写出字符。 | | | AH=02h | 读入字符。 | | | AH=03h | 状态。 | | |
| INT 15h | | 其它（系统支持例程）。   |  |  |  | | --- | --- | --- | | AH=4FH | | 键盘拦截。 | | AH=83H | 事件等待。 | | | AH=84H | 读游戏杆。 | | | AH=85H | SysRq 键。 | | | AH=86H | 等待。 | | | AH=87H | 块移动。 | | | AH=88H | 获取扩展内存容量。 | | | AH=C0H | 获取系统参数。 | | | AH=C1H | 获取扩展 BIOS 数据区段。 | | | AH=C2H | 指针设备功能。 | | | AH=E8h, AL=01h (AX = E801h) | 获取扩展内存容量（自从 1944 年引入的新功能），可获取到 64MB 以上的内存容量。 | | | AH=E8h, AL=20h (AX = E820h) | 查询系统地址映射。该功能取代了 AX=E801h 和 AH=88h。 | | |
| INT 16h | | 键盘通信例程。   |  |  |  | | --- | --- | --- | | AH=00h | | 读字符。 | | AH=01h | 读输入状态。 | | | AH=02h | 读 Shift 键（修改键）状态。 | | | AH=10h | 读字符（增强版）。 | | | AH=11h | 读输入状态（增强版）。 | | | AH=12h | 读 Shift 键（修改键）状态（增强版）。 | | |
| INT 17h | | 打印服务。   |  |  |  | | --- | --- | --- | | AH=00h | | 打印字符。 | | AH=01h | 初始化打印机。 | | | AH=02h | 检查打印机状态。 | | |
| INT 18h | 执行磁带上的 BASIC 程序：“真正的”IBM 兼容机在 ROM 里内置 BASIC 程序，当引导失败时由 BIOS 调用此例程解释执行。（例：打印“Boot disk error. Replace disk and press any key to continue...”这类提示信息） | |
| INT 19h | [加电自检](http://zh.wikipedia.org/wiki/加电自检)之后载入操作系统。 | |
| INT 1Ah | | 实时钟服务。   |  |  |  | | --- | --- | --- | | AH=00h | | 读取实时钟。 | | AH=01h | 设置实时钟。 | | | AH=02h | 读取实时钟时间。 | | | AH=03h | 设置实时钟时间。 | | | AH=04h | 读取实时钟日期。 | | | AH=05h | 设置实时钟日期。 | | | AH=06h | 设置实时钟闹铃。 | | | AH=07h | 重置实时钟闹铃。 | | |
| INT 1Bh | Ctrl+Break，由 IRQ 9 自动调用。 | |
| INT 1Ch | 预留，由 IRQ 8 自动调用。 | |
| INT 1Dh | 不可调用：指向视频参数表（包含视频模式的数据）的指针。 | |
| INT 1Eh | 不可调用：指向软盘模式表（包含关于软驱的大量信息）的指针。 | |
| INT 1Fh | 不可调用：指向视频图形字符表（包含从 80h 到 FFh 的 [ASCII](http://zh.wikipedia.org/wiki/EASCII) 字符的数据）的信息。 | |
| INT 41h | 地址指针：硬盘参数表（第一硬盘）。 | |
| INT 46h | 地址指针：硬盘参数表（第二硬盘）。 | |
| INT 4Ah | 实时钟在闹铃时调用。 | |
| INT 70h | IRQ8: 由实时钟调用。 | |
| INT 74h | IRQ12: 由鼠标调用 | |
| INT 75h | IRQ13: 由数学协处理器调用。 | |
| INT 76h | IRQ14: 由第一个 IDE 控制器所呼叫 | |
| INT 77h | IRQ15: 由第二个 IDE 控制器所呼叫 | |