课程设计2报告

设计思路

1. int 19 将 0 道 0 面 1 扇区（512） 的内容复制道 **0:7c00h** 处， CS:IP 改为 **0:7c00h**
    1. 所以 0 道 0 面 1 扇区功能为 复制 0 道 0 面 2 扇区 开始的3个扇区 中的内容(**Boot_end - Boot**)道 **0:7E00h** 处,CS:IP 改为 **0:7E00h**
2. 选功能的话即通过键盘中断来执行
    3. 一号功能 把 CS:IP 改为 **FFFF:0**
    4. 二号功能 把 c 盘 0 道 0 面 1 扇区（512） 的内容读到 **0:7c00h** 处
    5. 三号功能 从 CMOS 读出时间
    6. 四号功能 调用键盘中断读取键盘输入，并修改 CMOS 对应位置
        1. 新增退出功能
        2. 新增日期格式判断
            1. 没有对每个月是31,30还是28,29天判断，全部统一判断31天
            2. 年份没有判断
            3. 0<月<=12,0<小时<=24,0<=秒（分）<=60

前置准备

1. 安装virtual Box && win xp
2. 为 win xp 添加 软盘 && 共享空间

实验流程

1. 将编译的 asm 文件的可执行程序 放入 共享空间 并在 win xp 中的CMD终端中运行
2. 重启计算机，测试相应功能

设计代码

```
1    assume cs:code,ds:data,ss:stack
```

```
2
3   stack segment
4    db 128 dup (0)
5   stack ends
6
7   data segment
8   ; begin        db 512 dup (0) ;一个扇区
9   ; begin_boot   db 512 dup (0)
10  ;              db 512 dup (0)
11  ;              db 512 dup (0)
12  data ends
13
14  code segment
15  start:
16          mov ax,stack
17          mov ss,ax
18          mov sp,128
19
20          call copy_introduce
21          call copy_boot_disk
22
23
24          mov ax,4c00h
25          int 21h
26  ;---------------------------------
27  introduce:                      ;引导程序，将程序复制到0:7c00处，
28           mov bx,0
29      mov ss,bx
30      mov sp,7c00h
31
32      call save_old_int9
33      call copy_Boot_from_disk
34
35      mov bx,0
36      push bx
37      mov bx,7e00h            ;设置cs：ip为0:7e00h执行Boot程序
38      push bx
39      retf
40          ;-----------------------------
41          copy_Boot_from_disk:
42                  mov bx,0
43                  mov es,bx
44                  mov bx,7e00h
45
46                  mov al,2
47                  mov ch,0
48                  mov cl,2
49                  mov dl,0
50                  mov dh,0
51                  mov ah,2
52                  int 13h
53
54                  ret
55          ;---------------------------------
56          save_old_int9:
57                  mov bx,0
58                  mov es,bx
59
```

```
                push es:[9*4]
                pop es:[200h]
                push es:[9*4+2]
                pop es:[202h]
                ret
        ;-----------------------
                db 512 dup (0)
        introduce_end:nop
 ;=================================
copy_introduce:
       mov bx,cs
       mov es,bx
       mov bx,offset introduce

       mov al,1
       mov ch,0
       mov cl,1
       mov dl,0
       mov dh,0
       mov ah,3
       int 13h
       ret
;----------------------
copy_boot_disk:
       mov bx,cs
       mov es,bx
       mov bx,offset Boot

       mov al,2
       mov ch,0
       mov cl,2
       mov dl,0
       mov dh,0
       mov ah,3
       int 13h
       ret
;--------------------------------
Boot:
        jmp Boot_start
        ;xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
        option1   db '(1) reset pc',0
        option2   db '(2) start system',0
        option3   db '(3) clock',0
        option4   db '(4) set clock',0

        address_option  dw offset option1 - offset Boot + 7e00h
                        dw offset option2 - offset Boot + 7e00h
                        dw offset option3 - offset Boot + 7e00h
                        dw offset option4 - offset Boot + 7e00h
        timestyle   db '00/00/00 00:00:00',0
        timeadress  db 9,8,7,4,2,0
        string_stack   db 12 dup ('0'),0
        error_string db 'time format error!!!!',0
        ;xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

        Boot_start:
                call init_reg
                call clear_screen
```

```
118                 call show_option
119
120                 jmp short choose_option
121
122                 mov ax,4c00h
123                 int 21h
124         ;-----------------------
125                 choose_option:
126                         call clear_buff
127
128                         mov ah,0
129                         int 16h
130
131                         cmp al,'1'
132                         je choose1
133                         cmp al,'2'
134                         je choose2
135                         cmp al,'3'
136                         je choose3
137                         cmp al,'4'
138                         je choose4
139
140                         jmp choose_option
141
142                 choose1:mov di,160*3
143                         mov byte ptr es:[di],'1'
144                         mov bx,0ffffh
145                         push bx
146                         mov bx,0
147                         push bx
148                         retf
149                         jmp choose_option
150
151                 choose2:mov di,160*3
152                         mov byte ptr es:[di],'2'
153                         call start_old_system
154                         jmp choose_option
155
156                 choose3:mov di,160*3
157                         mov byte ptr es:[di],'3'
158                         call show_clock
159                         jmp Boot_start
160
161                 choose4:mov di,160*3
162                         mov byte ptr es:[di],'4'
163                         call set_clock
164                         jmp Boot_start
165         ;-----------------------
166         start_old_system:
167                 mov bx,0
168                 mov es,bx
169                 mov bx,7c00h
170
171                 mov al,1
172                 mov ch,0
173                 mov cl,1
174                 mov dl,80h      ;80h代表C盘
175                 mov dh,0
```

```
176                 mov ah,2
177                 int 13h
178 
179                 mov bx,0
180                 push bx
181                 mov bx,7c00h
182                 push bx
183                 retf
184         ;------------------------
185         set_clock:
186                 ; call clear_screen
187                 call clear_string_stack
188                 call show_string_stack
189                 call get_string
190 
191                 cmp ah,01h
192                 je set_clock_ret
193 
194                 call check_time_fromat
195                 call set_time
196 
197                 set_clock_ret:
198                 ret
199                 ;--------------------
200                 check_time_fromat:
201                         mov bx,offset timeadress - offset Boot + 7e00h
202                         mov si,offset string_stack - offset Boot +7e00h
203                         mov cx,6
204                         ctf_lp1:
205                                 mov dx,ds:[si]
206                                 sub dh,30h
207                                 sub dl,30h
208                                 shl dl,1
209                                 shl dl,1
210                                 shl dl,1
211                                 shl dl,1
212                                 and dh,00001111b
213                                 or dl,dh
214 
215                                 ;6Y 5M 4D 3H 2m 1S
216                                 cmp cx,6
217                                 je check_year
218                                 cmp cx,5
219                                 je check_month
220                                 cmp cx,4
221                                 je check_day
222                                 cmp cx,3
223                                 je check_hour
224                                 cmp cx,2
225                                 je check_min
226                                 cmp cx,1
227                                 je check_sec
228                                 jmp continue_check
229 
230                                         ;------------------------------
231                                         check_year:
232                                                 jmp continue_check
233                                         check_month:
```

```
                                        cmp dl,12h
                                        ja print_error
                                        cmp dl,0h
                                        je print_error
                                        jmp continue_check
                                check_day:
                                        cmp dl,31h
                                        ja print_error
                                        cmp dl,0h
                                        je print_error
                                        jmp continue_check
                                check_hour:
                                        cmp dl,24h
                                        ja print_error
                                        jmp continue_check
                                check_min:
                                        cmp dl,60h
                                        ja print_error
                                        jmp continue_check
                                check_sec:
                                        cmp dl,60h
                                        ja print_error
                                        jmp continue_check

                        ;-------------------------------
                        continue_check:
                                add si,2
                                inc bx
                        loop ctf_lp1
                        jmp check_time_fromat_ret

                                ;-------------------------------
                                print_error:
                                        push si
                                        push di
                                        mov si,offset error_string -
    offset Boot + 7e00h
                                        mov di,160*20
                                        call showstr
                                        pop si
                                        pop di
                                        call delay

                                        mov cx,7

                                        jmp check_time_fromat_ret
        check_time_fromat_ret:
                                ret
                ;-------------------------------
                delay:
                        push ax
                        push dx

                        mov dx,10000h
                        mov ax,0

                        s1: sub ax,1
                                sbb dx,0
```

```
291                                     cmp ax,0
292                                     jne s1
293                                     cmp dx,0
294                                     jne s1
295
296                                     pop dx
297                                     pop ax
298
299                                     ret
300
301                 ;--------------------
302                 set_time:
303
304                         cmp cx,7
305                         je set_time_ret
306
307                         mov bx,offset timeadress - offset Boot + 7e00h
308                         mov si,offset string_stack - offset Boot +7e00h
309                         mov cx,6
310                         settime:
311
312                                 mov dx,ds:[si]
313                                 sub dh,30h
314                                 sub dl,30h
315                                 shl dl,1
316                                 shl dl,1
317                                 shl dl,1
318                                 shl dl,1
319                                 and dh,00001111b
320                                 or dl,dh
321                                 mov al,ds:[bx]
322                                 out 70h,al
323                                 mov al,dl
324                                 out 71h,al
325
326                                 add si,2
327                                 inc bx
328
329
330                                 loop settime
331                         set_time_ret:
332                                 ret
333                 ;-------------------------
334                 get_string:
335                         mov si,offset string_stack - offset Boot + 7e00h
336                         mov bx,0
337                         getstring:
338                                 call clear_buff
339                                 mov ah,0
340                                 int 16h
341                                 cmp al,'0'
342                                 jb notnumber
343                                 cmp al,'9'
344                                 ja notnumber
345                                 call char_push
346                                 call show_string_stack
347
348                                 jmp getstring
```

```
349                             getstringret:
350                                     ret
351                             notnumber:
352                                     cmp ah,0eh      ;backspace
353                                     je isbackspace
354                                     cmp ah,01h      ;ese
355                                     je getstringret
356                                     cmp ah,1ch
357                                     je getstringret ;enter
358                                     jmp getstring
359                             isbackspace:
360                                     call char_pop
361                                     call show_string_stack
362                                     jmp getstring
363                             ;-------------------------
364                             char_pop:
365                                     cmp bx,0
366                                     je charpopret
367                                     dec bx
368                                     mov byte ptr ds:[si+bx],'0'
369                                     charpopret:
370                                             ret
371                             ;-------------------------
372                             char_push:
373                                     cmp bx,11
374                                     ja charpushret
375                                     mov ds:[si+bx],al
376                                     inc bx
377                                     charpushret:
378                                             ret
379                             ;-------------------------
380
381                     ;-------------------
382                     show_string_stack:
383                             push si
384                             push di
385                             mov si,offset string_stack - offset Boot + 7e00h
386                             mov di,160*4
387                             call showstr
388                             pop di
389                             pop si
390                             ret
391                     ;--------------------
392                     clear_string_stack:
393                             push bx
394                             push cx
395                             push es
396                             push si
397                             push di
398
399                             mov si,offset string_stack - offset Boot + 7e00h
400                             mov dx,3030h
401
402                             mov cx,6
403                             clearstringstack:
404                                     mov ds:[si],dx
405                                     add si,2
406                                     loop clearstringstack
```

```
407
408                                 pop di
409                                 pop si
410                                 pop es
411                                 pop cx
412                                 pop bx
413                                 ret
414         ;----------------------
415         show_clock:
416                 call show_style
417                 call set_new_int9
418
419                 mov bx,offset timeadress - offset Boot + 7e00h
420                 showtime:
421                         mov si,bx
422                         mov di,160*20
423                         mov cx,6
424                         showdate:
425                                 mov al,ds:[si]
426                                 out 70h,al
427                                 in al,71h
428
429                                 mov ah,al
430                                 shr ah,1
431                                 shr ah,1
432                                 shr ah,1
433                                 shr ah,1
434                                 and al,00001111b
435                                 add ah,30h
436                                 add al,30h
437                                 mov es:[di],ah
438                                 mov es:[di+2],al
439                                 add di,6
440                                 inc si
441                                 loop showdate
442
443                                 jmp showtime
444                         show_clockret:
445                                 call set_old_int9
446                                 ret
447                 ;----------------------
448                 show_style:
449                         mov si,offset timestyle - offset Boot + 7e00h
450                         ;mov si,offset error_string - offset Boot + 7e00h
451                         mov di,160*20
452                         call showstr
453                         ret
454                 ;--------------------
455                 set_old_int9:
456                                 push bx
457                                 push es
458
459                                 mov bx,0
460                                 mov es,bx
461                                 cli
462                                 push es:[200h]
463                                 pop es:[9*4]
464                                 push es:[202h]
```

```
465                                     pop es:[9*4+2]
466                                     sti
467
468                                     pop es
469                                     pop bx
470                                     ret
471                             ;---------------------
472                             set_new_int9:
473                                     push bx
474                                     push es
475
476                                     mov bx,0
477                                     mov es,bx
478
479                                     cli
480                                     mov word ptr es:[9*4],offset newint9 - offset
    Boot + 7e00h
481                                     mov word ptr es:[9*4+2],0
482                                     sti
483
484                                     pop es
485                                     pop bx
486                                     ret
487                             ;------------------------
488                             newint9:
489                                     push ax
490                                     call clear_buff
491
492                                     in al,60h
493                                     pushf
494                                     call dword ptr cs:[200h]
495
496                                     cmp al,01h
497                                     je inesc
498                                     cmp al,3bh
499                                     jne int9ret
500                                     call change_time_color
501
502                                     int9ret:
503                                     pop ax
504                                     iret
505                                     inesc:
506                                     pop ax
507                                     add sp,4
508                                     popf
509                                     jmp show_clockret
510                             ;----------------------
511                             change_time_color:
512                                     push bx
513                                     push cx
514                                     push es
515
516                                     mov bx,0b800h
517                                     mov es,bx
518                                     mov cx,17
519                                     mov bx,160*20+1
520                                     change_time_colors:
521                                     inc byte ptr es:[bx]
```

```
522                                 add bx,2
523                                 loop change_time_colors
524 
525                                 pop es
526                                 pop cx
527                                 pop bx
528                         ;-------------------------
529                         clear_buff:
530                                 mov ah,1
531                                 int 16h
532                                 jz clearbuffret
533                                 mov ah,0
534                                 int 16h
535                                 jmp clear_buff
536                                 clearbuffret:
537                                         ret
538                 ;---------------------
539                 show_option:
540                         mov bx,offset address_option - offset Boot + 7e00h
541                         mov cx,4
542                         mov di,160*10 + 30*2
543                         show_options:
544                                 mov si,ds:[bx]
545                                 call showstr
546                                 add di,160
547                                 add bx,2
548                                 loop show_options
549                                 ret
550                 ;-----------------------------
551                 showstr:
552                         push cx
553                         push di
554                         showstrs:
555                                 mov cl,ds:[si]
556                                 cmp cl,0
557                                 je showstrret
558                                 mov es:[di],cl
559                                 add di,2
560                                 inc si
561                                 jmp short showstrs
562                         showstrret:
563                                 pop di
564                                 pop cx
565                                 ret
566 
567                 ;----------------
568                 init_reg:
569                         mov bx,0b800h
570                         mov es,bx
571 
572                         mov bx,0
573                         mov ds,bx
574                         ret
575                 ;------------------
576                 clear_screen:
577                         mov bx,0
578                         mov dx,0700h    ;清屏中对字符属性设置应该为07h，而不是0
579                         mov cx,2000
```

```
580                             clearscreen:
581                                     mov es:[bx],dx
582                                     add bx,2
583                                     loop clearscreen
584                                     ret
585                             ;-----------------------
586                             db 512 dup (0)
587                     Boot_end:
588                     nop
589
590     code ends
591     end start
```