

```

;*****;
;*          C O N D R V          *;
;*-----*
;* Task      : This program represents a normal Console *;
;*            Driver (Keyboard and Display Monitor). It should *;
;*            serve as a framework for a driver in the form of *;
;*            an ANSI.SYS driver. *;
;*-----*
;* Author    : MICHAEL TISCHER *;
;* Developed on : 08/04/87 *;
;* Last update : 01/07/92 *;
;*-----*
;* Assembly  : MASM CONDRV; *;
;*            LINK CONDRV; *;
;*            EXE2BIN CONDRV CONDRV.SYS *;
;*                                     or *;
;*            TASM CONDRV *;
;*            LINK CONDRV; *;
;*            EXE2BIN CONDRV CONDRV.SYS *;
;*-----*
;* Call      : Copy into root directory, copy the command *;
;*            DEVICE=CONDRV.SYS into the file CONFIG.SYS *;
;*            and then boot the system. *;
;*****;

```

```
code segment
```

```
assume cs:code,ds:code
```

```
org 0 ;Program has no PSP therefore start
;at offset address 0
```

```
== Constants =====
```

```
cmd_fld equ 2 ;Offset command field in data block
status equ 3 ;Offset status field in data block
end_adr equ 14 ;Offset driver end addr. in data block
num_db equ 18 ;Offset number in data block
b_adr equ 14 ;Offset buffer address in data block
```

```
KEY_SZ equ 20 ;Size of keyboard buffer
num_cmd equ 16 ;Subfunctions 0-16 are supported
```

```
== Data =====
```

```
-- Device driver header -----
```

```
dw -1,-1 ;Link to next driver
dw 1010100000000011b ;Driver attribute
dw offset strat ;Pointer to strategy routine
dw offset intr ;Pointer to interrupt routine
db "CONDRV " ;New console driver
```

```
-- Jump table for functions -----
```

```
fct_tab dw offset init ;Function 0: Initialization
dw offset dummy ;Function 1: Media check
dw offset dummy ;Function 2: Create BPB
dw offset no_sup ;Function 3: I/O control read
dw offset read ;Function 4: Read
dw offset read_b ;Function 5: Non-destructive read
dw offset dummy ;Function 6: Input status
dw offset del_in_b ;Function 7: Delete input buffer
dw offset write ;Function 8: Write
dw offset write ;Function 9: Write & verify
dw offset dummy ;Function 10: Output status
dw offset dummy ;Function 11: Delete output buffer
dw offset no_sup ;Function 12: I/O control write
dw offset dummy ;Function 13: Open (Ver. 3.0 and up)
dw offset dummy ;Function 14: Close
dw offset dummy ;Function 15: Changeable medium
dw offset write ;Function 16: Output until busy
```

```
db_ptr dw (?), (?) ;Address of data block passed
```

```
key_a dw 0 ;Pointer to next character in KEY_SZ
```


[illegible]

```

        mov  ah,1             ;Function number for BIOS interrupt
        int  16h             ;Call BIOS keyboard interrupt
        je   read_p1         ;No character present --> READ_P1

        mov  es:[di+13],al    ;Store character in data block
        xor  ax,ax           ;Everything O.K.
        ret                  ;Return to caller

read_p1 label near

        mov  ax,0100h        ;Set busy bit (no character)
        ret                  ;Return to caller

read_b  endp

;-----

del_in_b proc near          ;Clear input buffer

        mov  ah,1            ;Still characters in the buffer?
        int  16h            ;Call BIOS keyboard interrupt
        je   del_e          ;No character in the buffer --> END

        xor  ah,ah           ;Remove character from buffer
        int  16h            ;Call BIOS keyboard interrupt
        jmp  short del_in_b  ;Test for additional characters

del_e:   xor  ax,ax          ;Everything O.K.
        ret                  ;Return to caller

del_in_b endp

;-----

write proc near            ;Write a specified number of
                           ;characters on the display screen

        mov  cx,es:[di+num_db] ;Number of characters read
        jcxz write_e        ;Test for equality to 0
        lds  si,es:[di+b_adr] ;Address of character buffer to DS:SI
        cld                  ;Increment on LODSB

        mov  ah,3           ;Read current display page
        int  16h           ;Call BIOS video interrupt

        mov  ah,14          ;Function number for BIOS interrupt

write_1: lodsb              ;Read character to be output to AL
        int  10h            ;Call BIOS video interrupt
        loop write_1        ;Repeat until all characters output

write_e: xor  ax,ax         ;Everything O.K.
        ret                  ;Return to caller

write endp

;-----

init      proc near        ;Initialization routine

        mov  word ptr es:[di+end_adr],offset init ;Set end address of
        mov  es:[di+end_adr+2],cs                ;the driver

        xor  ax,ax         ;Everything O.K.
        ret                  ;Return to caller

init      endp

;=====

code      ends
end

```