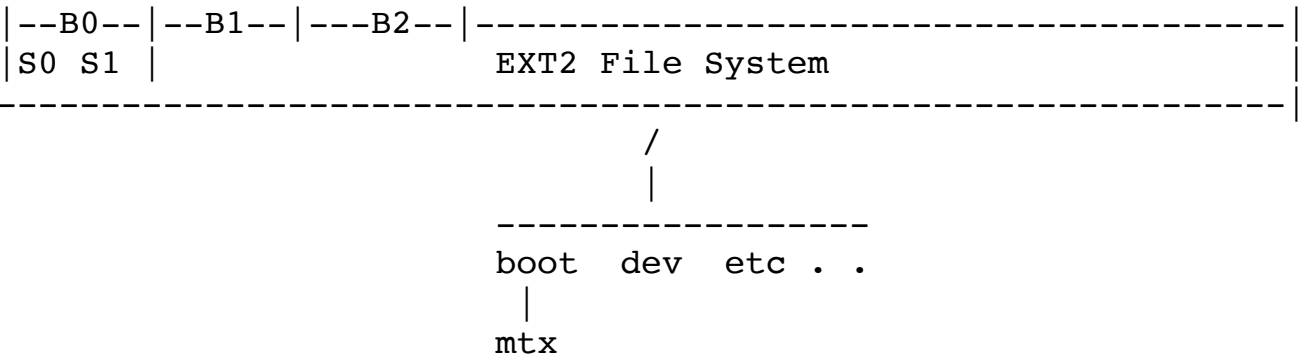
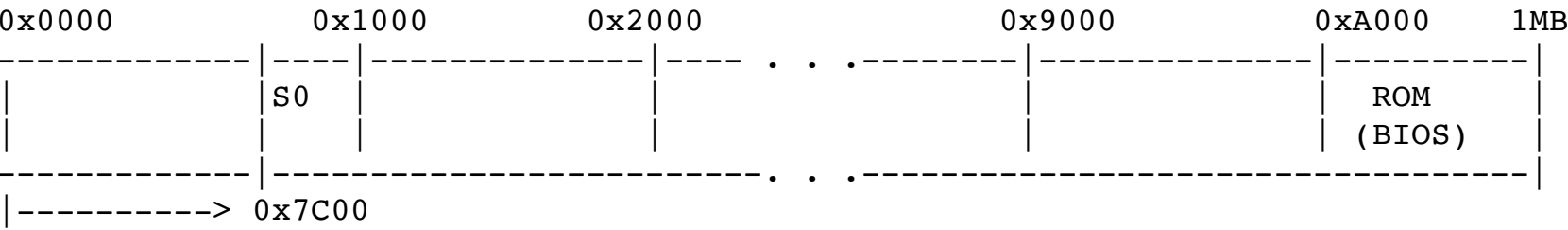


1. diskimage:



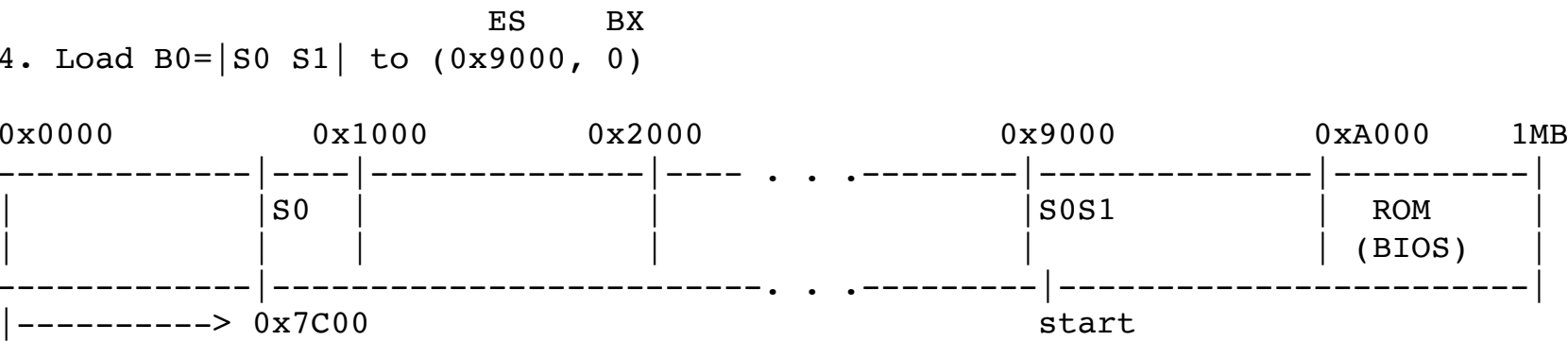
2. Memory



3. BOOTing: BIOS loads 512 bytes (S0) into (segment, offset)=(0x0000, 0x7C00)
Execute this piece of code at (0x0000, 0x7C00)

```
!-----
! Only one SECTOR loaded at (0000,7C00). Get entire BLOCK in
!-----
mov ax,#BOOTSEG      ! set ES to 0x9000
mov es,ax
xor bx,bx             ! clear BX = 0

!-----
! Call BIOS int13 to load boot BLOCK to (ES, BX)=(0x9000,0)
!-----
xor dx,dx             ! drive 0, head 0
xor cx,cx
incb cl              ! cyl 0, sector 1
mov ax, #0x0202      ! READ 1 block from (Cyl, Head, Sector)=(0,0,1)
int 0x13             ! to (ES=0x9000, BX=0) in memory
```

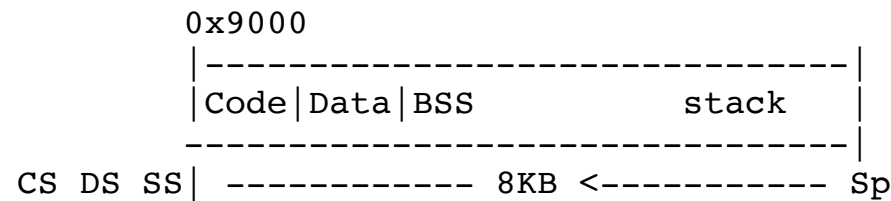


```
jmp    start,BOOTSEG      ! CS=BOOTSEG, IP=start
```

5. CPU jumps to (0x9000, start) to continue execution:

```
start:
mov     ax,cs             ! Set segment registers for CPU
mov     ds,ax             ! we know ES,CS=0x9000. Let DS=CS
mov     ss,ax             ! SS = CS ==> all point at 0x9000
mov     es,ax
mov     sp,#SSP           ! SP = 8KB above SS=0x9000
```

6. Set CS, DS, SS all point at 0x9000 => Execution image at 0x9000



```
mov     ax,#0x0012        ! set display to 640x480 color mode
int     0x10

call    _main             ! call main() in C
```

7. main() returns non-zero for success, 0 for failure

```
test ax, ax
je _error

jmp     0,0x1000          ! jump to (0x1000, 0) to execute kernel
```