# BIOS schreiben

**howto write a bios for cp/m 2.2**
**V. Pohlers, 2012**

Ich will versuchen, die Grundlagen zum Schreiben eines eigenen kleinen BIOS für CP/M 2.2 praxisgerecht darzustellen. Grundlagenwissen sollte man schon haben, z.B. was ein BIOS ist und welche Funktionen drin sind, wie eine Diskette aufgebaut ist u.a.m.
Detailwissen liefert CP/M 2.2 Interna.

- Teil 1 Warmstart, Zeichen-I/O, Diskettenfunktion
- Teil 2 Ansteuerung einer RAM-Disk

## CP/M-2.2 BIOS-Funktionen

| Funktion | Offs zu @0001 | Nr (Turbo-Pascal) | Beschreibung | in | out |
|---|---|---|---|---|---|
| BOOT | -3 | -1 | Cold start routine | - | - |
| WBOOT | 0 | 0 | Warm boot - reload command processor | - | - |
| CONST | 3 | 1 | Console status | - | A=Status (0=no key, 0FFh=key pressed) |
| CONIN | 6 | 2 | Console input, wait for key | - | A=key |
| CONOUT | 9 | 3 | Console output | C=char | - |
| LIST | 12 | 4 | Printer output, wait for ready | C=char | - |
| PUNCH | 15 | 5 | Paper tape punch output | C=char | - |
| READER | 18 | 6 | Paper tape reader input | - | A=char |
| HOME | 21 | 7 | Move disc head to track 0 | - | - |
| SELDSK | 24 | 8 | Select disc drive | C=drive(0=A,1=B,..) | HL=dph, 0000 as error indicator |
| SETTRK | 27 | 9 | Set track number | BC=track(0..nn) | - |
| SETSEC | 30 | 10 | Set sector number | BC=sector(0..nn) | - |
| SETDMA | 33 | 11 | Set DMA address | BC=dma | - |
| READ | 36 | 12 | Read a sector | - | A=0 for OK, 1 for unrecoverable error, 0FFh if media changed |

| Funktion | Offs zu @0001 | Nr (Turbo-Pascal) | Beschreibung | in | out |
|---|---|---|---|---|---|
| WRITE | 39 | 13 | Write a sector | C=0 Write can be deferred, C=1 Write must be immediate, C=2 Write can be deferred, no pre-read is necessary | A=0 for OK, 1 for unrecoverable error, 2 if disc is readonly, 0FFh if media changed |
| LISTST | 42 | 14 | Status of list device | - | A=0 (not ready) or A=0FFh (ready) |
| SECTRAN | 45 | 15 | Sector translation for skewing | BC=logical sector number (zero based) DE=address of translation table | HL = physical sector number |

# CP/M-2.2 BDOS-Funktionen

| Number | Hex | Function Name | Input | Output |
|---|---|---|---|---|
| 0 | 0 | System Reset | - | - |
| 1 | 1 | Console Input | - | A = ASCII char |
| 2 | 2 | Console Output | E = char | - |
| 3 | 3 | Reader Input | - | A = ASCII char |
| 4 | 4 | Punch Output | E = char | - |
| 5 | 5 | List Output | E = char | - |
| 6 | 6 | Direct Console I/O | E = 0FFH (input) | A = char |
| | | | E = 0FEH (status) | A = status |
| | | | E = char | - |
| 7 | 7 | Get I/O Byte | - | A = I/O byte value |
| 8 | 8 | Set I/O Byte | E = I/O byte | - |
| 9 | 9 | Print String | DE = Buffer Address | - |
| 10 | A | Read Console String | DE = Buffer | Console characters in Buffer |
| 11 | B | Get Console Status | - | A = 00/non zero |
| 12 | C | Return Version # | - | HL = Version # |
| 13 | D | Reset Disk System | - | - |
| 14 | E | Select Disk | E = Disk # | - |
| 15 | F | Open File | DE = FCB address | A = FF if not found |
| 16 | 10 | Close File | DE = FCB address | A = FF if not found |
| 17 | 11 | Search For First | DE = FCB address | A = Directory Code |
| 18 | 12 | Search For Next | - | A = Directory Code |
| 19 | 13 | Delete File | DE = FCB address | A = - |
| 20 | 14 | Read Sequential | DE = FCB address | A = Error Code |
| 21 | 15 | Write Sequential | DE = FCB Address | A = Error Code |
| 22 | 16 | Make File | DE = FCB address | A = FF if no DIR Space |
| 23 | 17 | Rename File | DE = FCB address | A = FF if not found |
| 24 | 18 | Return Login Vector | - | HL = Login Vector* |
| 25 | 19 | Return Current Disk | - | A = Current Disk Number |
| 26 | 1A | Set DMA Address | DE = DMA address | - |
| 27 | 1B | Get ADDR (ALLOC) | - | HL = ALLOC address* |

| Number | Hex | Function Name | Input | Output |
|---|---|---|---|---|
| 28 | 1C | Write Protect Disk | - | - |
| 29 | 1D | Get Read/only Vector | - | HL = ALLOC address* |
| 30 | 1E | Set File Attributes | DE = FCB address | A = - |
| 31 | 1F | Get ADDR (Disk Parms) | - | HL = DPB address |
| 32 | 20 | Set/Get User Code | E = 0FFH for Get | A = User Number |
| | | | E = 00 to 0FH for Set | - |
| 33 | 21 | Read Random | DE = FCB address | A = Error |
| 34 | 22 | Write Random | DE = FCB address | A = Error Code |
| 35 | 23 | Compute File Size | DE = FCB address | r0, r1, r2 |
| 36 | 24 | Set Random Record | DE = FCB address | r0, r1, r2 |
| 37 | 25 | Reset Drive | DE = Drive Vector | A = 0 |
| 38 | 26 | Access Drive | not supported | |
| 39 | 27 | Free Drive | not supported | |
| 40 | 28 | Write Random w/Fill | DE = FCB | A = error code |

* Note that A=L, and B=H upon return.