CP/M 2.2 Memory Layout

Complete Memory Map and System Architecture

CP/M 2.2 uses a well-defined memory layout that enables portability across different hardware platforms while providing efficient access to system services. This document describes the complete memory organization from address 0x0000 through to the top of memory, including the critical low-memory structures that programs use to interface with BDOS and BIOS.

Memory Organization Overview

```
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ Top of Memory (e.g., 0xFFFF for 64K)
■ BIOS Jump Table ■ Hardware-specific routines
■ (17+ JMP instructions) ■
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
■ BIOS Code ■ Machine-specific I/O handlers
■ ■
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ FBASE (typically 0xE400 for 64K)
■ BDOS ■ File system & I/O services
■ (~3.5K in CP/M 2.2) ■
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ CBASE (typically 0xDC00 for 64K)
■ CCP ■ Console Command Processor
■ (~2K bytes) ■
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ TBASE (0x0100)
■ ■
■ TPA ■ Transient Program Area
■ (User program space) ■ Programs load and execute here
■ ■
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ 0x0100
■ Default DMA Buffer (128 B) ■ 0x0080-0x00FF
■ & Command Line Tail ■
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ 0x0080
■ Default FCB #2 (optional) ■ 0x006C-0x007F
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ 0x006C
■ Default FCB (36 bytes) ■ 0x005C-0x007F
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ 0x005C
■ Unused/BIOS Scratch (88 bytes) ■ 0x0040-0x005B
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ 0x0040
■ RST 7 (0x0038): Debugger Jump ■ 0x0038-0x003A (3 bytes)
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ 0x0038
■ RST Vectors 1-6 (interrupts) ■ 0x0008-0x0037
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ 0x0008
■ JMP BDOS (C3 xx xx) ■ 0x0005-0x0007
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ 0x0005
■ Current Disk & User Number ■ 0x0004 (bits 0-3: disk, 4-7: user)
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ 0x0004
■ IOBYTE (device mapping) ■ 0x0003
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ 0x0003
■ JMP WBOOT (C3 xx xx) ■ 0x0000-0x0002
■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ 0x0000
```

Low Memory Map (0x0000 - 0x00FF)

Address	Size	Contents	Description
0x0000	3 bytes	`C3 xx xx`	JMP WBOOT - Jump to warm boot routine in BIOS. Always C3H (JMP opcode) followed by 16-bit address at 0x0001-0x0002.
0x0001-0x0002	2 bytes	`Address`	Warm boot vector - Points to WBOOT entry in BIOS jump table. Programs use this to find BIOS base (subtract 3).
0x0003	1 byte	`IOBYTE`	I/O byte for device mapping. Controls logical-to-physical device assignments (CON:, RDR:, PUN:, LST:).
0x0004	1 byte	`Disk/User`	Bits 0-3: Current default disk (0=A:, 1=B:, etc). Bits 4-7: Current user number (0-15).
0x0005	3 bytes	`C3 xx xx`	JMP BDOS - Primary entry point to BDOS. C3H (JMP) followed by BDOS entry address at 0x0006-0x0007.
0x0006-0x0007	2 bytes	`FBASE`	Address of BDOS entry point. Also indicates top of TPA (FBASE-1 is highest usable byte).
0x0008	3 bytes	`RST 1`	Restart vector 1. Available for user programs or system extensions. 8-byte intervals.
0x0010	3 bytes	`RST 2`	Restart vector 2. Can be used for custom interrupt handlers or fast call routines.
0x0018	3 bytes	`RST 3`	Restart vector 3. Often unused in standard CP/M systems.
0x0020	3 bytes	`RST 4`	Restart vector 4. May be used by operating system extensions.
0x0028	3 bytes	`RST 5`	Restart vector 5. Available for system or application use.
0x0030	3 bytes	`RST 6`	Restart vector 6. Can be programmed for interrupt service routines.
0x0038	3 bytes	`RST 7/INT`	Restart vector 7 - Reserved for debuggers (DDT, SID, etc). Creates break to debugger.
0x0040-0x004F	16 bytes	`Scratch`	BIOS scratch area. Used by BIOS for internal variables. Contents undocumented and system-dependent.
0x0050-0x005B	12 bytes	`Reserved`	Generally unused by CP/M 2.2. May be used by MP/M or system extensions. Safe for user data.
0x005C-0x007F	36 bytes	`FCB #1`	Default File Control Block. CCP parses first filename from command line into this FCB.
0x006C-0x007F	20 bytes	`FCB #2`	Second default FCB (overlaps). CCP parses second filename from command line (for COPY, REN, etc).
0x0080	1 byte	`Length`	Command line tail length. Number of characters in command line following program name.

Address	Size	Contents	Description
0x0081-0x00FF	127 bytes	`Command`	Command line tail buffer and default DMA. ASCII characters from command line, then DMA buffer space.

Finding and Calling BIOS Functions Directly

Although Digital Research officially recommended that programs only use BDOS calls, many applications (including Microsoft BASIC 5.21) bypass BDOS and call BIOS functions directly for better performance or to access functions not available through BDOS. Here's how programs locate and call BIOS functions:

Method to Find BIOS Jump Table

```
; Step 1: Get WBOOT address from location 0x0001-0x0002
LHLD 0001H ; Load WBOOT address into HL

; Step 2: Calculate BIOS base (WBOOT - 3 bytes)
LXI D,-3 ; DE = -3
DAD D ; HL = BIOS base (points to BOOT entry)

; Step 3: Calculate desired BIOS function address
; Each BIOS function is 3 bytes apart in jump table
; BOOT=0, WBOOT=3, CONST=6, CONIN=9, CONOUT=12, etc.

; Example: Call CONIN (console input) at offset 6
LXI D,6 ; Offset to CONIN
DAD D ; HL = address of CONIN jump
CALL HL ; Call CONIN (not supported on 8080, use PCHL trick)

; 8080-compatible method:
PCHL ; Jump to address in HL
; (or push HL, then RET to simulate CALL)
```

BIOS Jump Table Offsets (from BIOS base)

Offset	Function	Description
-3	BOOT	Cold start (BIOS base - 3)
0	WBOOT	Warm boot (address stored at 0x0001)
3	CONST	Console status
6	CONIN	Console input
9	CONOUT	Console output
12	LIST	List output (printer)
15	PUNCH	Punch output
18	READER	Reader input
21	HOME	Home disk head
24	SELDSK	Select disk
27	SETTRK	Set track
30	SETSEC	Set sector
33	SETDMA	Set DMA address
36	READ	Read sector
39	WRITE	Write sector

42	LISTST	List status (CP/M 2.2+)
45	SECTRAN	Sector translate (CP/M 2.2+)

Microsoft BASIC 5.21 Example

Microsoft BASIC 5.21 uses direct BIOS calls for performance-critical operations, particularly for the Control-C (^C) break check. Instead of calling BDOS function 11 (console status), MBASIC retrieves the BIOS CONST function address and calls it directly. This reduces overhead and provides faster keyboard response. The technique involves reading the warm boot vector at 0x0001, calculating the BIOS base, then accessing the CONST function at offset +6 from the WBOOT entry.

High Memory Organization

The top of memory contains the CP/M operating system components, loaded in reverse order from highest to lowest addresses:

BIOS (Basic Input/Output System)

Located at the very top of memory (size varies by system, typically 1.5-2K bytes). Contains hardware-specific routines for console, disk, and peripheral I/O. Begins with a jump table of JMP instructions, each 3 bytes, pointing to the actual service routines. The BIOS is the only component that must be customized for each different computer system.

BDOS (Basic Disk Operating System)

Located just below BIOS (approximately 3.5K bytes in CP/M 2.2). Provides file system services, device I/O buffering, and high-level disk operations. Entry point address is stored at 0x0006-0x0007 (FBASE). BDOS is machine-independent and identical across all CP/M systems. All BDOS functions are accessed through CALL 0005H with function number in register C.

CCP (Console Command Processor)

Located just below BDOS (approximately 2K bytes). Provides the command-line interface, parses user commands, loads and executes transient programs (.COM files). The CCP can be overlaid by transient programs that need maximum memory. Programs that overlay CCP must reload it by jumping to 0x0000 (warm boot) when exiting. Programs that don't overlay CCP can simply RET to return.

TPA (Transient Program Area)

The Transient Program Area extends from 0x0100 to CBASE-1 (base of CCP). This is the area where user programs (.COM files) are loaded and executed. Programs always start at address 0x0100H. The size of TPA determines the maximum program size and can be adjusted using MOVCPM to reconfigure CP/M for different memory sizes.

Stack Pointer Initialization

When a transient program starts, the stack pointer (SP) is set to a small stack area maintained by the CCP, located just below the CCP code in high memory. This provides an 8-level stack with the CCP return address already pushed. However, most programs immediately reset SP to point to their own stack area within the TPA, typically near the top of their data area (but below CBASE if not overlaying CCP). The BDOS maintains its own internal stack and switches to it on every BDOS call.

Default FCB and DMA Buffer

Default FCB (0x005C-0x007F): The File Control Block is a 36-byte structure used for all file operations. The CCP automatically parses the first filename from the command line into this FCB. Structure:

```
Byte 0x005C: Drive (0=default, 1=A:, 2=B:, etc)
Bytes 0x005D-0x0064: Filename (8 bytes, space-padded)
Bytes 0x0065-0x0067: Extension (3 bytes, space-padded)
Byte 0x0068: Extent number (EX)
Bytes 0x0069-0x006A: Reserved (S1, S2)
```

```
Byte 0x006B: Record count in current extent (RC)
Bytes 0x006C-0x007F: Disk allocation map and other fields
```

Second FCB (0x006C-0x007F): For commands with two filenames (like COPY, REN), CCP parses the second filename here. This overlaps with the disk map portion of the first FCB.

DMA Buffer (0x0080-0x00FF): The Default Direct Memory Access buffer is 128 bytes. Initially contains the command line tail:

```
Byte 0x0080: Number of characters in command line (0-127)
Bytes 0x0081-0x00FF: Command line characters in uppercase ASCII
```

After parsing, this area serves as the default disk I/O buffer for file read/write operations. Programs can change the DMA address using BDOS function 26.

Memory Size Configuration

CP/M can be configured for different memory sizes using the MOVCPM utility. This adjusts the location of CCP, BDOS, and BIOS to match available RAM. Example memory layouts:

```
20K System:
BIOS: 0x4A00-0x4FFF (1.5K)
BDOS: 0x3C00-0x49FF (3.5K)
CCP: 0x3400-0x3BFF (2K)
TPA: 0x0100-0x33FF (~12.75K)

48K System:
BIOS: 0xBA00-0xBFFF (1.5K)
BDOS: 0xAC00-0xB9FF (3.5K)
CCP: 0xA400-0xABFF (2K)
TPA: 0x0100-0xA3FF (~40.75K)

64K System:
BIOS: 0xFA00-0xFFFF (1.5K)
BDOS: 0xEC00-0xF9FF (3.5K)
CCP: 0xE400-0xEBFF (2K)
TPA: 0x0100-0xE3FF (~56.75K)
```

Critical Programming Notes

1. Portability Considerations:
Programs that call BIOS directly are not portable across different CP/M systems. Each vendor's BIOS may have different implementations or extensions. Use BDOS calls for maximum portability.

2. BDOS vs BIOS:
BDOS provides buffered I/O and file services. BIOS provides raw hardware access. Direct BIOS calls bypass BDOS buffers and can cause inconsistent system state if disk functions are used carelessly.

3. Finding Available Memory:
Read the BDOS entry address from 0x0006-0x0007. Subtract 1 to get the highest usable byte of TPA. Example:

```
LHLD 0006H ; Get FBASE (BDOS entry)
DCX H ; HL = last byte of TPA
```

4. Command Line Parsing:
Always extract command line data from 0x0080-0x00FF before performing any file operations, as this area serves as the default DMA buffer and will be overwritten on the first disk read.

5. RST Vectors:
RST 0 (at 0x0000) always contains JMP WBOOT for system reset. RST 7 (at 0x0038) is reserved for debuggers. Other RST vectors (1-6) can be used by programs but should be restored before exiting if modified.

6. IOBYTE Usage:
The IOBYTE at 0x0003 allows logical device redirection (console, list, reader, punch). Not all CP/M implementations support this fully. Use BDOS functions 7 and 8 to get/set IOBYTE.

7. Memory Protection:
CP/M has no memory protection. Programs can write to any memory location, including system areas. Careless

programming can crash the system requiring a cold boot.

Additional Information

For more detailed information about CP/M system programming, consult:
• *CP/M 2.2 Operating System Manual* - Digital Research
• *CP/M Assembly Language Programming Manual* - Digital Research
• *The CP/M Handbook* by Rodney Zaks
• *CP/M Bible* by Waite Group Press

Online resources include archives at:
• www.gaby.de/cpm/
• www.seasip.info/Cpm/
• www.cpm.z80.de/