The H8-Z180 RAM Programmer's Manual
02/09/23

Table of Contents

Initialization....2
CP/M 3 Banked Memory....2
RAM Disk....3

The Z180 can directly access 1M of RAM/ROM using the built-in MMU. The Z180 CPU is still restricted to a 64K address space, but the MMU presents a sliding window partitioning of the 64K logical address space mapped to the 1M physical address space. The windows have a granularity of 4K.

There are three, maximum, windows which have fixed order and relationship to each other. The first window is “Common Area 0” and is always mapping logical addresses starting at 0000 to physical addresses starting at 00000, however this window may be shrunk to 0-bytes length – essentially disabling it. The second window is “Banked Area” and always immediately follows “Common Area 0” in the logical address space. This window may also be shrunk to 0-bytes length. The last window is “Common Area 1” and always immediately follows “Banked Area” and always extends to the end of the logical address space.

This windowing is configured using the Common/Bank Area Register (CBAR). The low 4 bits of the CBAR define the starting address (x000) of the “Banked Area” – which also defines the end of “Common Area 0”. The high 4 bits of the CBAR define starting address of “Common Area 1” – and thus define the end of the “Banked Area”.

Here are some example CBAR values and how they affect the MMU windows:

BAR	Windows
FF (default)	0000-EFFF = Common Area 0 (60K)
	F000-F000 = Banked Area (disabled)
	F000-FFFF = Common Area 1 (4K)
C0 – Traditional 48K banked, 16K common	0000-0000 = Common Area 0 (disabled)
	0000-BFFF = Banked Area (48K)
	C000-FFFF = Common Area 1 (16K)
C4 – Used for accessing 32K EEPROM	0000-3FFF = Common Area 0 (16K)
	4000-BFFF = Banked Area (32K)
	C000-FFFF = Common Area 1 (16K)

“Common Area 0” is always directly mapped to physical address 00000, but “Banked Area” and “Common Area 1” may be mapped to physical memory starting at any 4K boundary. The Bank Base

Register (BBR) defines the physical address base (xx000) for the “Banked Area” window. The Common Base Register (CBR) defines the physical address base for the “Common Area 1” window.

Note that the Z180 MMU adds the logical address to the relevant base address to produce the physical address. Therefor, the CBAR configuration must be taken into account when determining/interpreting the base address registers.

The Z180 maps all windows to physical address 00000 by default (BBR = CBR = 00). The H8-Z180 places RAM at physical address 00000, and exposes the 32K EEPROM at physical addresses F8000-FFFFF. Since the tradition Heath ROM mapping scheme is required by software, after RESET the H8-Z180 overlays the logical address space 0000-1FFF with the first 8K of the EEPROM, allowing for the normal Heath H8/H89 startup procedures.

As with all banked memory schemes, care must be taken to ensure that the code which is executing is in memory that does not change with the mapping. In addition, the program stack must be preserved, or must not be used. It is also necessary to consider interrupts and whether the necessary code/stack is available to service an interrupt. For this reason, interrupts are often disabled during critical bank switch operations.

Note that the Z180 DMA engine uses physical memory addresses, and completely bypasses CBAR, BBR, and CBR. This is a benefit at times, but can also create complications for things like CP/M 3 when trying to access a user DMA (as defined by CP/M) block that spans the banked and common boundary.

Initialization

CP/M 3 Banked Memory

Here is a typical CP/M 3 Bank Switching table for the first 208K (four 56K banks plus 8K common) of the RAM:

```
CBAR = E0 (56K banked, 8K common)
CBR = 00 (physical memory 0E000-0FFFF in Common Area 1)
bank0:      BBR = 00 (physical memory 00000-0DFFF)
bank1:      BBR = 10 (physical memory 10000-1DFFF)
bank2:      BBR = 1E (physical memory 1E000-2BFFF)
bank3:      BBR = 2C (physical memory 2C000-39FFF)
```

Because of the DMA engine using physical addresses, the CP/M 3 "XMOVE" feature may be implemented. This allows for more flexible placement of buffers and can also improve performance of the Directory HASH Buffers and warm boot reloading of the CCP.

The reference implementation (memz180.asm) demonstrates all of the above. An example bank select routine, where the logical bank number is passed in A, is:

```
bank$sel:
        lxi     h,table ; table of bank schemes
        mov     c,a     ; logical bank number * 4
        mvi     b,0     ;
        dad     b       ; index to desired bank
        mov     a,m     ; get base addresses
        out0    mmu$bbr ; change base addr reg – map memory
```

RAM Disk

Since CP/M 3 can rarely take advantage of more that 3 banks (176K), the additional memory beyond what CP/M 3 can use may be organized into a "RAM disk".

Because of the DMA engine, I/O to/from the RAM Disk can be done directly to the CP/M DMA Buffers. This greatly improves performance and simplifies the driver code. However, care must be taken when the CP/M DMA buffer spans the Common Area 1 boundary.

The reference RAM Disk implementation (rdz180.asm) computes the physical address which represents the desired disk data, then programs the DMA engine to transfer to/from the data area. Note, the DMA buffer may be for BDOS buffers or the user's program space so a specific bank cannot be assumed. The DMA bank and address must be translated to a physical address, and also must be checked for location below/above the Common Area 1 boundary.