RomWBW
User Guide

Version 3.1 Pre-release
29 Jan 2023

RetroBrew Computers Group
www.retrobrewcomputers.org

Wayne Warthen
wwarthen@gmail.com

Contents

This document is a general usage guide for the RomWBW software and is generally the best place to start with RomWBW. There are several companion documents you should refer to as appropriate:

- RomWBW System Guide discusses much of the internal design and construction of RomWBW. It includes a reference for the RomWBW HBIOS API functions.
- RomWBW Applications is a reference for the OS-hosted proprietary command line applications that were created to enhance RomWBW.
- RomWBW ROM Applications is a reference for the ROM-hosted applications provided with RomWBW including the monitor, programming languages, etc.
- RomWBW Disk Catalog is a reference for the contents of the disk images provided with RomWBW. It is not entirely up-to-date.
- RomWBW Errata is updated as needed to document issues or anomalies discovered in the current software distribution.

Since RomWBW is purely a software product for many different platforms, the documentation does **not** cover hardware construction, configuration, or troubleshooting – please see your hardware provider for this information.

RomWBW exists primarily to host preexisting operating systems and applications. As such, the documentation of these is not repeated here. Please refer to the Doc directory of the RomWBW distribution where you will find specific documents covering these components.

Each of the operating systems and ROM applications included with RomWBW are sophisticated tools in their own right. It is not reasonable to fully document their usage here. However, you will find complete manuals in PDF format in the Doc directory of the distribution. The intention of this document is to describe the operation of RomWBW and the ways in which it enhances the operation of the included applications and operating systems.

Section 1

Overview

RomWBW provides a complete software system for a wide variety of hobbyist Z80/Z180/Z280 CPU-based systems produced by these developer communities:

- RetroBrew Computers
- RC2014
- retro-comp

General features include:

- Banked memory services for several banking designs
- Disk drivers for RAM, ROM, Floppy, IDE, CF, and SD
- Serial drivers including UART (16550-like), ASCI, ACIA, SIO
- Video drivers including TMS9918, SY6545, MOS8563, HD6445
- Keyboard (PS/2) drivers via VT8242 or PPI interfaces
- Real time clock drivers including DS1302, BQ4845
- Multiple OS support including CP/M 2.2, ZSDOS, CP/M 3, ZPM3, QPM, & p-System
- Built-in VT-100 terminal emulation support

RomWBW is distributed as both source code and pre-built ROM and disk images. Some of the provided software can be launched directly from the

ROM firmware itself:

- System Monitor
- Operating Systems (CP/M 2.2, ZSDOS)
- ROM BASIC (Nascom BASIC and Tasty BASIC)
- ROM Forth

A dynamic disk drive letter assignment mechanism allows mapping operating system drive letters to any available disk media. Additionally, mass media devices (IDE Disk, CF Card, SD Card) support the use of multiple slices (up to 256 per device). Each slice contains a complete CP/M filesystem and can be mapped independently to any drive letter. This overcomes the inherent size limitations in legacy OSes and allows up to 2GB of accessible storage on a single device.

The pre-built ROM firmware images are generally suitable for most users. However, it is also very easy to modify and build custom ROM images that fully tailor the firmware to your specific preferences. All tools required to build custom ROM firmware under Windows are included – no need to install assemblers, etc. The firmware can also be built using Linux or MacOS after confirming a few standard tools have been installed.

Multiple disk images are provided in the distribution. Most disk images contain a complete, bootable, ready-to-run implementation of a specific operating system. A “combo” disk image contains multiple slices, each with a full operating system implementation. If you use this disk image, you can easily pick whichever operating system you want to boot without changing media.

By design, RomWBW isolates all of the hardware specific functions in the ROM chip itself. The ROM provides a hardware abstraction layer such that all of the operating systems and applications on a disk will run on any RomWBW-based system. To put it simply, you can take a disk (or CF/SD Card) and move it between systems transparently.

A tool is provided that allows you to access a FAT-12/16/32 filesystem. The FAT filesystem may be coresident on the same disk media as RomWBW slices or on stand-alone media. This makes exchanging files with modern OSes such as Windows, MacOS, and Linux very easy.

Section 2

Getting Started

2.1 Acquiring RomWBW

The RomWBW Repository on GitHub is the official distribution location for all project source and documentation. The fully-built distribution releases are available on the RomWBW Releases Page of the repository. On this page, you will normally see a Development Snapshot as well as recent stable releases. Unless you have a specific reason, I suggest you stick to the most recent stable release. Expand the “Assets” drop-down for the release you want to download, then select the asset named RomWBW-vX.X.X-Package.zip. The Package asset includes all pre-built ROM and Disk images as well as full source code. The other assets contain only source code and do not have the pre-built ROM or disk images.

All source code and distributions are maintained on GitHub. Code contributions are very welcome.

Distribution Directory Layout

The RomWBW distribution is a compressed zip archive file organized in a set of directories. Each of these directories has it’s own ReadMe.txt file

describing the contents in detail. In summary, these directories are:

Directory	Description
Binary	The final output files of the build process are placed here. Most importantly, the ROM images with the file names ending in ".rom" and disk images ending in .img.
Doc	Contains various detailed documentation, both RomWBW specifically as well as the operating systems and applications.
Source	Contains the source code files used to build the software and ROM images.
Tools	Contains the programs that are used by the build process or that may be useful in setting up your system.

2.2 Installation

In general, installation of RomWBW on your platform is very simple. You just need to program your ROM with the correct ROM image from the RomWBW distribution. Subsequently, you can write disk images on your disk drives (IDE disk, CF Card, SD Card, etc.) which then provides even more functionality.

The pre-built ROM images will automatically detect and support typical devices for their corresponding platform including serial ports, video adapters, on-board disk interfaces, and PropIO/ParPortProp boards without building a custom ROM. The distribution is a .zip archive. After downloading it to a working directory on your modern computer (Windows/Linux/Mac) use any zip tool to extract the contents of the archive.

Depending on how you got your hardware, you may have already been provided with a pre-programmed ROM chip. If so, use that initially. Otherwise, you will need to use a ROM programmer to initially program your ROM chip. Please refer to the documentation that came with your ROM programmer

for more information. Once you have a running RomWBW system, you can generally update your ROM to a newer version in-situ with the included ROM Flashing tool (Will Sowerbutts' FLASH application) as described in the Upgrading section of this document.

The Binary directory of the distribution contains the pre-built ROM and disk images. The ROM image files all end in ".rom". Based on the table below, **carefully** pick the appropriate ROM image for your hardware.

Platform	ROM Image File	Baud	Description
SBC	SBC_std.rom	38400	RetroBrew SBC v1 or v2 ECB Z80
Zeta V1	ZETA_std.rom	38400	RetroBrew Zeta V1 Z80, ParPortProp (optional)
Zeta V2	ZETA2_std.rom	38400	RetroBrew Zeta V2 Z80, ParPortProp (optional)
N8	N8_std.rom	38400	RetroBrew N8 Z180, date code >= 2312
Mark IV	MK4_std.rom	38400	RetroBrew Mark IV ECB Z180
RC2014 Z80	RCZ80_std.rom	115200	RC2014 w/ Z80 CPU, requires 512K RAM/ROM module
RC2014 Z80	RCZ80_duart.rom	115200	RC2014 w/ Z80 CPU w/ DUART serial module
RC2014 Z80	RCZ80_mt.rom	115200	RC2014 w/ Z80 CPU w/ MT011 network/SPI module
RC2014 Z80	RCZ80_kio.rom	115200	RC2014 w/ Z80 CPU, requires 512K RAM/ROM module
RC2014 Z180*	RCZ180_ext.rom	115200	RC2014 w/ Z180 CPU & 512K banked RAM/ROM module
RC2014 Z180*	RCZ180_nat.rom	115200	RC2014 w/ Z180 CPU & 512K native RAM/ROM module

Platform	ROM Image File	Baud	Description
RC2014 Z280*	RCZ180_ext.rom	115200	RC2014 w/ Z280 CPU & 512K banked RAM/ROM module
RC2014 Z280*	RCZ180_nat.rom	115200	RC2014 w/ Z280 CPU & 512K native RAM/ROM module
Easy Z80	EZZ80_std.rom	115200	Sergey Kiselev's Easy Z80
Tiny Z80	EZZ80_tz80.rom	115200	Sergey Kiselev's Tiny Z80
Z80-512K	EZZ80_skz.rom	115200	Sergey Kiselev's Z80+RAM+ROM+CPLD module
SC126	SCZ180_126.rom	115200	Stephen Cousin's SC126 Z180
SC130	SCZ180_130.rom	115200	Stephen Cousin's SC130 Z180
SC131	SCZ180_131.rom	115200	Stephen Cousin's SC131 Z180
SC140	SCZ180_140.rom	115200	Stephen Cousin's SC140 Z180
Dyno	DYNO_std.rom	38400	Steve Garcia's Z180 Dyno Computer
MBC	MBC_std.rom	38400	Andrew Lynch's Nhyodyne Multi Board Computer
RPH	RPH_std.rom	38400	Andrew Lynch's Rhyophyre Single Board Computer
ZRC	RCZ80_zrc.rom	115200	Bill Shen's Z80 CPU w/ CPLD for RC2014 bus
ZZRC	RCZ280_nat_zzr.rom	115200	Bill Shen's Z280 CPU w/ CPLD for RC2014 bus
ZZ80MB	RCZ280_nat_zz.rom	115200	Bill Shen's Z280 CPU Motherboard w/ RC2014 slots

*The RC2014 Z180 & Z280 requires a separate RAM/ROM memory module. There are two types of these modules and you must pick the ROM for your type of memory module. The first type of RAM module includes bank switching logic – this is called external ("ext") because the bank switching is performed externally from the CPU. The second type of RAM module has

no bank switching logic – this is called native (“nat”) because the CPU itself provides the bank switching logic. Only Z180 and Z280 CPUs have the ability to do bank switching in the CPU, so the ext/nat selection only applies to them. Z80 CPUs have no bank switching logic, so they are always configured for external bank switching.

All pre-built ROM images are pure binary files (they are not “hex” files). They are intended to be programmed starting at the very start of the ROM chip (address 0). All of the pre-built images are exactly 512KB in size. If your system utilizes a larger ROM, you can just program the image into the first 512KB of the ROM for now.

Initially, don’t worry about trying to write a disk image to any disk (or CF/SD) devices you have. This will be covered later. You will be able to boot and check out your system with just the ROM.

Connect a serial terminal or computer with terminal emulation software to the primary serial port of your CPU board. You may need to refer to your hardware provider’s documentation for details. A null-modem connection may be required. Set the baud rate as indicated in the table above. Set the line characteristics to 8 data bits, 1 stop bit, no parity, and no flow control. If possible, select VT-100 terminal emulation.

RomWBW will automatically attempt to detect and support typical add-on components for each of the systems supported. More information on the required system configuration and optional supported components for each ROM is found in the file called “RomList.txt” in the Binary directory.

2.3 System Startup

Upon power-up, your terminal should display a sign-on banner within 2 seconds followed by hardware inventory and discovery information. When hardware initialization is completed, a boot loader prompt allows you to choose a ROM-based operating system, system monitor, application, or boot

from a disk device.

Here is an example of a fairly typical startup. Your system will have different devices and configuraion, but the startup should look similar.

```
RomWBW HBIOS v3.1.1-pre.183, 2022-10-04

RC2014 [RCZ80_kio] Z80 @ 7.372MHz
0 MEM W/S, 1 I/O W/S, INT MODE 2, Z2 MMU
512KB ROM, 512KB RAM
ROM VERIFY: 00 00 00 00 PASS

KIO: IO=0x80 ENABLED
CTC: IO=0x84 TIMER MODE=TIM16
AY: MODE=RCZ80 IO=0xD8 NOT PRESENT
SIO0: IO=0x89 SIO MODE=115200,8,N,1
SIO1: IO=0x8B SIO MODE=115200,8,N,1
DSRTC: MODE=STD IO=0xC0 NOT PRESENT
MD: UNITS=2 ROMDISK=384KB RAMDISK=256KB
FD: MODE=RCWDC IO=0x50 NOT PRESENT
IDE: IO=0x10 MODE=RC
IDE0: NO MEDIA
IDE1: NO MEDIA
PPIDE: IO=0x20
PPIDE0: LBA BLOCKS=0x00773800 SIZE=3815MB
PPIDE1: NO MEDIA

Unit        Device      Type              Capacity/Mode
----------  ----------  ----------------  --------------------
Char 0      SIO0:       RS-232            115200,8,N,1
Char 1      SIO1:       RS-232            115200,8,N,1
Disk 0      MD0:        RAM Disk          256KB,LBA
Disk 1      MD1:        ROM Disk          384KB,LBA
```

```
Disk 2      IDE0:       Hard Disk         --
Disk 3      IDE1:       Hard Disk         --
Disk 4      PPIDE0:     CompactFlash      3815MB,LBA
Disk 5      PPIDE1:     Hard Disk         --
```

If your system completes the ROM-based boot process successfully, you should see the RomWBW Boot Loader prompt. For example:

```
RC2014 [RCZ80_kio] Boot Loader


Boot [H=Help]:
```

If you get to this prompt, your system has completed the boot process and is ready to accept commands. Note that the Boot Loader is not an operating system or application. It is essentially the point where you choose which operating system or application you want RomWBW to execute.

The Boot Loader is explained in detail in the next section. For now, you can try a few simply commands to confirm that you can interact with the system.

At the Boot Loader prompt, you can type `H <enter>` for help. You can type `L <enter>` to list the available built-in ROM applications. If your terminal supports ANSI escape sequences, you can try the 'G' command to play a simple on-screen game.

If all of this seems fine, your ROM has been successfully programmed and you can continue below to learn how to use more of the RomWBW features.

2.4 Devices and Units

During the startup process, RomWBW displays a summary of all hardware along with a summary of the system configuration. Understanding how RomWBW deals with your hardware.

In order to support a wide variety of hardware, RomWBW HBIOS uses a

modular approach to implementing device drivers and presenting devices to an operating system. In general, all devices are classified as one of the following:

- Disk (RAM/ROM Disk, Floppy Disk, Hard Disk, CF Card, SD Card, etc.)
- Character (Serial Ports, Parallel Ports, etc.)
- Video (Video Display/Keyboard Interfaces)
- RTC/NVRAM (Real Time Clock, Non-volatile RAM)

HBIOS uses the concept of unit numbers to present a complex set of hardware devices to the operating system. As an example, a typical system might have a ROM Disk, RAM Disk, Floppy Drives, and Disk Drives. All of these are considered disk devices and are presented to the operating system as generic block devices. This means that each operating system does not need to imbed code to interact directly with all of the different hardware devices – RomWBW takes care of that.

There are 3 basic groups of information displayed during the startup process.

- Core System Information
- Hardware Discovery Log
- Device Unit Assignments & Configuration

Each of these is described below.

Core System Information

The first few lines of startup information displayed provide the most basic information on your system. In the example above, these lines are the Core System Information:

```
RomWBW HBIOS v3.1.1-pre.183, 2022-10-04

RC2014 [RCZ80_kio] Z80 @ 7.372MHz
0 MEM W/S, 1 I/O W/S, INT MODE 2, Z2 MMU
512KB ROM, 512KB RAM
```

```
ROM VERIFY: 00 00 00 00 PASS
```

The first line is a version identification banner for RomWBW. After that you see a group of 4 lines describing the basic system. In this example, the platform is the RC2014 running configuration RCZ80_kio at 7.372 MHz CPU clock speed. There are 0 memory wait states and 1 I/O wait state. Z80 interrupt mode 2 is active and the bank memory manager is the “Z2” which is standard for RC2014. There is 512KB of ROM total and 512KB of RAM total. Finally, a verification of the checksum of the critical ROM banks is shown (all 4 should be 00).

Hardware Discovery Log

The next set of messages during boot show the hardware devices as they are probed and initially configured. In the example above, these lines are:

```
KIO: IO=0x80 ENABLED
CTC: IO=0x84 TIMER MODE=TIM16
AY: MODE=RCZ80 IO=0xD8 NOT PRESENT
SIO0: IO=0x89 SIO MODE=115200,8,N,1
SIO1: IO=0x8B SIO MODE=115200,8,N,1
DSRTC: MODE=STD IO=0xC0 NOT PRESENT
MD: UNITS=2 ROMDISK=384KB RAMDISK=256KB
FD: MODE=RCWDC IO=0x50 NOT PRESENT
IDE: IO=0x10 MODE=RC
IDE0: NO MEDIA
IDE1: NO MEDIA
PPIDE: IO=0x20
PPIDE0: LBA BLOCKS=0x00773800 SIZE=3815MB
PPIDE1: NO MEDIA
```

What you see will depend on your specific system and ROM, but should match the hardware present in your system. Each device has a tag that precedes the colon. This tag identifies the driver and instance of each

device. For example, the tag “SIO0:” refers to the SIO serial port driver and specifically the first channel. The “SIO1:” tag refers to the second channel.

In many cases you will see IO=0xNN in the data following the tag. This identifies the base I/O port address of the hardware device and is useful for identifying hardware conflicts.

Note that you may see some lines indicating that the associated hardware is not present. Above, you can see that the FD driver did not find a floppy interface. Lines such as these are completely normal when your system does not have the associated hardware.

Finally, be aware that all ROMs are configured to identify specific hardware devices at specific port addresses. If you add hardware to your system that is not automatically identified, you may need to build a custom ROM to add support for it. Building a custom ROM is cover later.

Device Unit Assignments & Configuration

In the final group of startup messages, you will see the device unit summary table.

As RomWBW boots, it assigns a unit number to each device. This unit number is used by the operating system to refer to the device. It is, therefore, important to know the unit number assigned to each device. Here is an example:

```
Unit        Device      Type              Capacity/Mode
----------  ----------  ----------------  --------------------
Char 0      UART0:      RS-232            38400,8,N,1
Char 1      UART1:      RS-232            38400,8,N,1
Disk 0      MD1:        RAM Disk          384KB,LBA
Disk 1      MD0:        ROM Disk          384KB,LBA
Disk 2      FD0:        Floppy Disk       3.5",DS/HD,CHS
Disk 3      FD1:        Floppy Disk       3.5",DS/HD,CHS
```

```
Disk 4      IDE0:       CompactFlash      3815MB,LBA
Disk 5      IDE1:       Hard Disk         --
Disk 6      PRPSD0:     SD Card           1886MB,LBA
Video 0     CVDU0:      CRT               Text,80x25
```

In this example, you can see that the system has a total of 7 Disk Units numbered 0-6. There are also 2 Character Units and 1 Video Unit. The table shows the unit numbers assigned to each of the devices. Notice how the unit numbers are assigned sequentially regardless of the specific device.

There may or may not be media in the disk devices listed. For example, the floppy disk devices (Disk Units 2 & 3) may not have a floppy in the drive. Also note that Disk Unit 4 shows a disk capacity, but Disk Unit 5 does not. This is because the PPIDE interface of the system supports up to two drives, but there is only one actual drive attached. A unit number is assigned to all available devices regardless of whether they have actual media installed at boot time.

Note that Character Unit 0 is normally the initial system console.

If your system has an RTC/NVRAM device, it will not be listed in the unit summary table. Since only a single RTC/NVRAM device can exist in one system, unit numbers are not required nor used for this type of device.

Section 3

Boot Loader Operation

Once your system has completed the startup process, it presents a Boot Loader command prompt. The purpose of the Boot Loader is to select and launch a desired application or operating system. It also has the ability to configure some aspects of system operation.

After starting your system, following the hardware initialization, you will see the RomWBW Boot Loader prompt. Below is an example:

```
Mark IV [MK4_wbw] Boot Loader


Boot [H=Help]:
```

From the Boot Loader prompt, you can enter commands to select and launch any of the RomWBW operating systems or ROM applications. It also allows you to manage some basic settings of the system. To enter a command, just enter the command followed by ***<enter>***.

For example, typing `H<enter>` will display a short command summary:

```
Boot [H=Help]: h

  L             - List ROM Applications
```

```
  D            - Disk Device Inventory
  R            - Reboot System
  I <u> [<c>] - Set Console Interface/Baud code
  V [<n>]      - View/Set HBIOS Diagnostic Verbosity
  <u>[.<s>]    - Boot Disk Unit/Slice
```

Likewise the `L` command will display the list of ROM Applications that you can launch right from the Boot Loader:

```
Boot [H=Help]: L

ROM Applications:

  M: Monitor
  Z: Z-System
  C: CP/M 2.2
  F: Forth
  B: BASIC
  T: Tasty BASIC
  P: Play a Game
  N: Network Boot
  X: XModem Flash Updater
  U: User App
```

3.1 Launching from ROM

To start a ROM application you just enter the corresponding letter at the Boot Loader prompt. In the following example, we launch the built-in Micrsosoft BASIC interpreter. From within BASIC, we use the `BYE` command to return to the Boot Loader:

```
Boot [H=Help]: b
```

```
Loading BASIC...
Memory top?
Z80 BASIC Ver 4.7b
Copyright (C) 1978 by Microsoft
55603 Bytes free
Ok
bye


Mark IV [MK4_wbw] Boot Loader

Boot [H=Help]:
```

ROM Hosted Applications & OSes

The following ROM applications and OSes are available at the boot loader prompt:

Application	Description
Monitor	Z80 system debug monitor w/ Intel Hex loader
CP/M 2.2	Digital Research CP/M 2.2 OS
Z-System	ZSDOS 1.1 w/ ZCPR 1 (Enhanced CP/M compatible OS)
Forth	Brad Rodriguez's ANSI compatible Forth language
Tasty BASIC	Dimitri Theuling's Tiny BASIC implementation
Play	A simple video game (requires ANSI terminal emulation)
Network Boot	Boot system via Wiznet MT011 device
Flash Update	Upload and flash a new ROMWBW image using xmodem

Each of the ROM Applications is documented in RomWBW ROM Applications. Some of the applications (such as BASIC) also have their own independent manual in the Doc directory of the distribution. The OSes in-

cluded in the ROM (CP/M 2.2 & Z-System) are described in the Operating Systems section of this document.

In general, the command to exit any of these applications and restart the system is `BYE`. The exceptions are the Monitor which uses `B` and Play which uses `Q`.

Two of the ROM Applications are, in fact, complete operating systems. Specifically, “CP/M 2.2” and “Z-System” are provided so that you can actually start either operating system directly from your ROM without having any disk drives attached to your system. This is a great way to experiment with an operating system before you have disk drives attached and initialized with content.

3.2 Launching from Disk

In order to make use of the more sophisticated operating systems available with RomWBW, you will need to boot an operating system from a disk. Setting up disks is described in detail later. For now, we will just go over the command line for performing this type of boot.

From the Boot Loader prompt, you can enter a number (***<diskunit>***) and optionally a dot followed by a second number (***<slice>***). The ***<disk unit>*** unit number refers to a disk unit that was displayed when the system was booted – essentially it specifies the specific physical disk drive you want to boot. The ***<slice>*** numbers refers to a portion of the disk unit to boot. If no slice is specified, then it is equivalent to booting from the first slice (slice 0). Disk units and slices are described in more detail later.

Following this, you should see the operating system startup messages. Your operating system prompt will be `A>` and when you look at the drive letter assignments, you should see that A: has been assigned to the disk and slice you selected to boot.

If you receive the error message “Disk not bootable!”, you have either failed to properly initialize the disk and slice requested or you have selected the wrong disk/slice.

The following example shows a disk boot into the first slice of disk unit 4 which happens to be the CP/M 2.2 operating system on this disk. This is accomplished by entering just the number ‘4’ and pressing ***<enter>***.

```
Boot [H=Help]: 4

Booting Disk Unit 4, Slice 0, Sector 0x00000800...

Volume "Unlabeled" [0xD000-0xFE00, entry @ 0xE600]...

CBIOS v3.1.1-pre.194 [WBW]

Formatting RAMDISK...

Configuring Drives...

        A:=IDE0:0
        B:=MD0:0
        C:=MD1:0
        D:=FD0:0
        E:=FD1:0
        F:=IDE0:1
        G:=IDE0:2
        H:=IDE0:3
        I:=PRPSD0:0
        J:=PRPSD0:1
        K:=PRPSD0:2
        L:=PRPSD0:3
```

```
        1081 Disk Buffer Bytes Free

CP/M-80 v2.2, 54.0K TPA

A>
```

Here is another example where we are booting disk unit 4, slice 3 which is the CP/M 3 operating system on this disk:

```
Boot [H=Help]: 4.3

Booting Disk Unit 4, Slice 3, Sector 0x0000C800...

Volume "Unlabeled" [0x0100-0x1000, entry @ 0x0100]...

CP/M V3.0 Loader
Copyright (C) 1998, Caldera Inc.

 BNKBIOS3 SPR  F600  0800
 BNKBIOS3 SPR  4500  3B00
 RESBDOS3 SPR  F000  0600
 BNKBDOS3 SPR  1700  2E00

 60K TPA

CP/M v3.0 [BANKED] for HBIOS v3.1.1-pre.194

A>
```

Note that the Boot Loader simply launches whatever is in the disk unit/slice you have specified. It does not know what operating system is at that location. The layout of operating systems on disk media is described in the Using Disks section of this document.

3.3 System Configuration

Listing Disk Device Inventory

The disk device units available in your system are listed in the boot messages. However, if that list has scrolled off of your screen, you can use the 'D' command to display a list of them at any time from the Boot Loader prompt.

```
Boot [H=Help]: d

Disk Devices:

  Disk Unit 0 on MD0:
  Disk Unit 1 on MD1:
  Disk Unit 2 on FD0:
  Disk Unit 3 on FD1:
  Disk Unit 4 on IDE0:
  Disk Unit 5 on IDE1:
  Disk Unit 6 on IDE2:
  Disk Unit 7 on IDE3:
  Disk Unit 8 on IDE4:
  Disk Unit 9 on IDE5:
  Disk Unit 10 on SD0:
  Disk Unit 11 on PRPSD0:
```

Rebooting the System

The 'R' command within the Boot Loader performs a software reset of the system. It is the software equivalent of pressing the reset button.

There is generallhy no need to do this, but it can be convenient when you want to see the boot messages again or ensure your system is in a clean state.

```
Boot [H=Help]: r

Restarting System...
```

Changing Console and Console speed

Your system can support a number of devices for the console. They may be VDU type devices or serial devices. If you want to change which device is the console, the ***I*** menu option can be used to choose the unit and it's speed.

The command format is `I <unit> [<baudrate>]`

where ***<unit>*** is the character unit to select and ***<baudrate>*** is the optional baud rate.

Supported baud rates are:

```
 75   450  1800   7200  38400  115200   460800  1843200
150   600  2400   9600  28800  153600   614400  2457600
225   900  3600  14400  57600  230400   921600  3686400
300  1200  4800  19200  76800  307200  1228800  7372800
```

Here is an example of changing the console to unit #1 (the second serial port) and switching the port to 9600 baud:

```
Boot [H=Help]: i 1 9600

  Change speed now. Press a key to resume.

  Console on Unit #1
```

At this point, the Boot Loader prompt will be displayed on character unit #1.

Note that not all character devices support changing baud rates and some only support a limited subset of the baud rates listed. If you attempt to select an invalid baud rate for your system, you will get an error message.

HBIOS Diagnostic Verbosity

The ‘V’ command of the Boot Loader allows you to view and optionally change the level of diagnostic messages that RomWBW will produce. The normal verbosity level is 4, which means to display only fatal errors. You can increase this level to see more warnings when function calls to RomWBW HBIOS detect problems.

The use of diagnostic levels above 4 are really intended only for software developers. I do not recommend changing this under normal circumstances.

Section 4

Disk Devices

The systems supported by RomWBW all have the ability to use persistent disk media. A wide variety of disk devices are supported including floppy drives, hard disks, CF Cards, and SD Cards. RomWBW also supports the use of extra RAM and ROM memory as pseudo-disk devices.

RomWBW supports a variety of storage devices which will be discussed in more detail later.

- ROM Disk
- RAM Disk
- Floppy Disk
- Hard Disk (includes CF Cards and SD Cards)

We will start by discussing each of these types of storage devices and how to prepare them so files can be stored on them. Subsequently, we will describe how to install the pre-built disk images with bootable operating systems and ready-to-run content.

Some systems have disk interfaces built-in, while others will require add-in cards. You will need to refer to the documentation for your system for your specific options.

In the RomWBW boot messages, you will see hardware discovery messages. If you have a disk drive interface, you should see messages listing device types like FD:, IDE:, PPIDE:, SD:. Additionally, you will see messages indicating the media that has been found on the interfaces. As an example, here are the messages you might see if you have an IDE interface in your system with a single CF Card inserted in the primary side of the interface:

```
IDE: IO=0x80 MODE=MK4
IDE0: 8-BIT LBA BLOCKS=0x00773800 SIZE=3815MB
IDE1: NO MEDIA
```

The messages you see will vary depending on your hardware and the media you have installed. But, they will all have the same general format as the example above.

Once your your system has working disk devices, they will be accessible from any operating system you choose to run. Disk storage is available whether you boot your OS from ROM or from the disk media itself.

Refering back to the Boot Loader section on “Launching from ROM”, you could start CP/M 2.2 using the ‘C’ command. As the operating system starts up, you should see a list of drive letters assigned to the disk media you have installed. Here is an example of this:

```
Configuring Drives...

   A:=MD1:0
   B:=MD0:0
   C:=IDE0:0
   D:=IDE0:1
```

You will probably see more drive letters than this. The drive letter assignment process is described below in the Drive Letter Assignment section. Be aware that RomWBW will only assign drive letters to disk interfaces that actually have media in them. If you do not see drive letters assigned as expected,

refer to the prior system boot messages to ensure media has been detected in the interface. Actually, there is one exception to this rule: floppy drives will be assigned a drive letter regardless of whether there is any media inserted at boot.

Notice how each drive letter refers back to a specific disk hardware interface like IDE0. This is important as it is telling you what each drive letter refers to. Also notice that mass storage disks (like IDE) will normally have multiple drive letters assigned. The extra drive letters refer to additional “slices” on the disk. The concept of slices is described below in the Slices section.

4.1 Drive Letter Assignment

In legacy CP/M-type operating systems, drive letters were generally mapped to disk drives in a completely fixed way. For example, drive A: would **always** refer to the first floppy drive. Since RomWBW supports a wide variety of hardware configurations, it implements a much more flexible drive letter assignment mechanism so that any drive letter can be assigned to any disk device.

At boot, you will notice that RomWBW automatically assigns drive letters to the available disk devices. These assignments are displayed during the startup of the selected operating system. Additionally, you can review the current drive assignments at any time using the `ASSIGN` command. CP/M 3 and ZPM3 do not automatically display the assignments at startup, but you can use `ASSIGN` to display them. Refer to RomWBW Applications for more information on use of the `ASSIGN` command.

Here is an example of the list of drive letter assignments made during the startup of Z-System:

```
Loading Z-System...


CBIOS v3.1.1-pre.194 [WBW]
```

```
Formatting RAMDISK...

Configuring Drives...

        A:=MD0:0
        B:=MD1:0
        C:=FD0:0
        D:=FD1:0
        E:=IDE0:0
        F:=IDE0:1
        G:=IDE0:2
        H:=IDE0:3

        1081 Disk Buffer Bytes Free

ZSDOS v1.1, 54.0K TPA
```

Above you can see that drive A: has been assigned to MD0 which is the RAM Disk device. Drives C: and D: have been assigned to floppy drives. Drives E: thru L: have been assigned to the IDE0 hard disk device. The 4 entries for IDE0 are referring to 4 slices on that disk. Slices are discussed later.

The drive letter assignments **do not** change during an OS session unless you use the `ASSIGN` command yourself to do it. Additionally, the assignments at boot will stay the same on each boot as long as you do not make changes to your hardware configuration. Note that the assignments **are** dependent on the media currently inserted in hard disk drives. So, notice that if you insert or remove an SD Card or CF Card, the drive assignments will change. Since drive letter assignments can change, you must be careful when doing destructive things like using `CLRDIR` to make sure the drive letter you use is referring to the desired media.

When performing a ROM boot of an operating system, note that A: will be

your RAM disk and B: will be your ROM disk. When performing a disk boot, the disk you are booting from will be assigned to A: and the rest of the drive letters will be offset to accommodate this. This is done because most legacy operating systems expect that A: will be the boot drive.

4.2 ROM & RAM Disks

A typical RomWBW system has 512KB of ROM and 512KB of RAM. Some portions of each are dedicated to loading and running applications and operating system. The space left over is available for an operating system to use as a pseudo-disk device.

The RAM disk provides a small CP/M filesystem that you can use for the temporary storage of files. Unless your system has a battery backed mechanism for persisting your RAM contents, the RAM disk contents will be lost at each power-off. However, the RAM disk is an excellent choice for storing temporary files because it is very fast. You will notice that the first time an operating system is started after the power was turned off, you will see a message indicating that the RAM disk is being formatted. If you reset your system without turning off power, the RAM disk will not be reformatted and it's contents will still be intact.

Like the RAM disk, the ROM disk also provides a small CP/M filesystem, but it's contents are static – they are part of the ROM. As such, you cannot save files to the ROM disk. Any attempt to do this will result in a disk I/O error. The contents of the ROM disk have been chosen to provide a core set of tools and applications that are helpful for either CP/M 2.2 or ZSDOS. Since ZSDOS is CP/M 2.2 compatible, this works fairly well. However, you will find some files on the ROM disk that will work with ZSDOS, but will not work on CP/M 2.2. For example, `LDDS`, which loads the ZSDOS date/time stamper will only run under ZSDOS.

Unlike other types of disk devices, ROM and RAM Disks do not contain an

actual operating system and are not “bootable”. However, they are accessible to any operating system (whether the operating system is loaded from ROM or a different disk device).

Neither RAM not ROM disks require explicit formatting or initialization. ROM disks are pre-formatted and RAM disks are formatted automatically with an empty directory when first used.

Flash ROM Disks

The limitation of ROM disks being read only can be overcome on some platforms with the appropriate selection of Flash ROM chip and system configuration. In this case the flash-file system can be enabled which will allow the ROM disk to be read and written to. Flash devices have a limited write lifespan and continual usage will eventually wear out the device. It is not suited for high usage applications. Enabling ROM disk writing requires building a custom ROM.

4.3 Floppy Disks

If your system has the appropriate hardware, RomWBW will support the use of floppy disks. The supported floppy disk formats are generally derived from the IBM PC floppy disk formats:

- 5.25” 360K Double-sided, Double-density
- 5.25” 1.2M Double-sided, High-density
- 3.5” 720K Double-sided, Double-density
- 3.5” 1.44M Double-sided, High-density

When supported, RomWBW is normally configured for 2 3.5” floppy drives. If a high-density drive is used, then RomWBW automatically detects and adapts to double-density or high-density media. It cannot automatically detect 3.5” vs. 5.25” drive types – the ROM must be pre-configured for the

drive type.

Floppy media must be physically formatted before it can be used. This is normally accomplished by using the supplied Floppy Disk Utility (FDU) application. This application interacts directly with your hardware and therefore you must specify your floppy interface hardware at startup. Additionally, you need to specify the floppy drive and media format to use for formatting.

Below is a sample session using FDU to format a 1.44M floppy disk in the first (primary) floppy disk drive:

```
B>fdu

Floppy Disk Utility (FDU) v5.8, 26-Jul-2021 [HBIOS]
Copyright (C) 2021, Wayne Warthen, GNU GPL v3

SELECT FLOPPY DISK CONTROLLER:
  (A) Disk IO ECB Board
  (B) Disk IO 3 ECB Board
  (C) Zeta SBC Onboard FDC
  (D) Zeta 2 SBC Onboard FDC
  (E) Dual IDE ECB Board
  (F) N8 Onboard FDC
  (G) RC2014 SMC (SMB)
  (H) RC2014 WDC (SMB)
  (I) SmallZ80 Expansion
  (J) Dyno-Card FDC, D1030
  (K) RC2014 EPFDC
  (L) Multi-Board Computer FDC
  (X) Exit
=== OPTION ===> D-IDE

===== D-IDE ===========<< FDU MAIN MENU >>======================
(S)ETUP: UNIT=00  MEDIA=720KB DS/DD    MODE=POLL        TRACE=00
```

```
---------------------------------------------------------------
(R)EAD          (W)RITE         (F)ORMAT        (V)ERIFY
(I)NIT BUFFER   (D)UMP BUFFER   FDC (C)MDS      E(X)IT
=== OPTION ===> SETUP
ENTER UNIT [00-03] (00):
00: 3.5" 720KB - 9 SECTORS, 2 SIDES, 80 TRACKS, DOUBLE DENSITY
01: 3.5" 1.44MB - 18 SECTORS, 2 SIDES, 80 TRACKS, HIGH DENSITY
02: 5.25" 320KB - 8 SECTORS, 2 SIDES, 40 TRACKS, DOUBLE DENSITY
03: 5.25" 360KB - 9 SECTORS, 2 SIDES, 40 TRACKS, DOUBLE DENSITY
04: 5.25" 1.2MB - 15 SECTORS, 2 SIDES, 80 TRACKS, HIGH DENSITY
05: 8" 1.11MB - 15 SECTORS, 2 SIDES, 77 TRACKS, DOUBLE DENSITY
06: 5.25" 160KB - 8 SECTORS, 1 SIDE, 40 TRACKS, DOUBLE DENSITY
07: 5.25" 180KB - 9 SECTORS, 1 SIDE, 40 TRACKS, DOUBLE DENSITY
08: 5.25" 320KB - 8 SECTORS, 1 SIDE, 80 TRACKS, DOUBLE DENSITY
09: 5.25" 360KB - 9 SECTORS, 1 SIDE, 80 TRACKS, DOUBLE DENSITY
ENTER MEDIA [00-09] (00): 01
00: POLLING (RECOMMENDED)
01: INTERRUPT (!!! READ MANUAL !!!)
02: FAST INTERRUPT (!!! READ MANUAL !!!)
03: INT/WAIT (!!! READ MANUAL !!!)
04: DRQ/WAIT (!!! NOT YET IMPLEMENTED!!!)
ENTER MODE [00-04] (00):
ENTER TRACE LEVEL [00-01] (00):

===== D-IDE ===========<< FDU MAIN MENU >>======================
(S)ETUP: UNIT=00  MEDIA=1.44MB DS/HD   MODE=POLL        TRACE=00
---------------------------------------------------------------
(R)EAD          (W)RITE         (F)ORMAT        (V)ERIFY
(I)NIT BUFFER   (D)UMP BUFFER   FDC (C)MDS      E(X)IT
=== OPTION ===> FORMAT (T)RACK, (D)ISK ===> DISK
ENTER INTERLEAVE [01-12] (02):
```

```
RESET DRIVE...
PROGRESS: TRACK=4F HEAD=01 SECTOR=01

===== D-IDE ===========<< FDU MAIN MENU >>======================
(S)ETUP: UNIT=00  MEDIA=1.44MB DS/HD   MODE=POLL        TRACE=00
-------------------------------------------------------------
(R)EAD          (W)RITE         (F)ORMAT        (V)ERIFY
(I)NIT BUFFER   (D)UMP BUFFER   FDC (C)MDS      E(X)IT
=== OPTION ===> EXIT
```

Since the physical format of floppy media is the same as that used in a standard MS-DOS/Windows computer, you can also physicall format floppy media in a modern computer. However, the directory format itself will not be compatible with CP/M OSes. In this case, you can use the `CLRDIR` application supplied with RomWBW to reformat the directory area.

Once a floppy disk is formatted, you can read/write files on it using any of the RomWBW operating systems. The specific commands will depend on the operating system or application in use – refer to the appropriate OS/application documentation as needed.

WARNING: Some of the operating systems provided with RomWBW require that a soft-reset be performed when swapping floppy disk media. For example, under CP/M 2.2, you must press control-C at the CP/M prompt after inserting a new floppy disk.

4.4 Hard Disks

Under RomWBW, a hard disk is similar to a floppy disk in that it is considered a disk unit. However, RomWBW has multiple features that allow it's legacy operating systems to take advantage of modern mass storage media.

To start with, the concept of a hard disk in RomWBW applies to any storage device that provides at least 8MB of space. The actual media can be a real spinning hard disk, a CompactFlash Card, a SD Card, etc. In this document, the term hard disk will apply equally to all of these.

RomWBW uses Logical Block Addressing (LBA) to interact with all hard disks. The RomWBW operating systems use older Cylinder/Head/Sector (CHS) addressing. To accommodate the operating systems, RomWBW emulates CHS addressing. Specifically, it makes all hard disks look like they have 16 sectors and 16 heads. The number of tracks varies with the size of the physical hard disk.

It is recommended that hard disk media used with RomWBW be 1GB or greater in capacity. The reasons for this are discussed later, but it allows you to use the recommended disk layout for RomWBW that accommodates 64 CP/M filesystem slices and a 384KB FAT filesystem.

> Although we have not yet discussed how to get content on your disk units, it is necessary to have a basic understanding of how RomWBW handles disk devices as background. The following sections explain how disk units are managed within the operating systems. We will subsequently discuss how to actually setup disk devices with usable content.

4.5 Slices

The vintage operating systems included with RomWBW were produced at a time when mass storage devices were quite small. CP/M 2.2 could only handle filesystems up to 8MB. In order to achieve compatibility across all of the operating systems supported by RomWBW, the hard disk filesystem format used is 8MB. This ensures any filesystem will be accessible to any of

the operating systems.

Since storage devices today are quite large, RomWBW implements a mechanism called slicing to allow up to 256 8MB filesystems on a single large storage device. This allows up to 2GB of usable space on one media. You can think of slices as a way to refer to any of the first 256 8MB chunks of space on a single media.

Note that although you can use up to 256 slices per physical disk, this large number of slices is rarely used. The recommended RomWBW disk layout provides for 64 slices which is more than enough for most use cases.

Of course, the problem is that CP/M-like operating systems have only 16 drive letters (A:-P:) available. Under the covers, RomWBW allows you to use any drive letter to refer to any slice of any media. The `ASSIGN` command is used to view or change the current drive letter mappings at any time. At startup, the operating system will automatically allocate a reasonable number of drive letters to the available storage devices. The allocation will depend on the number of mass storage devices available at boot. For example, if you have only one hard disk type media, you will see that 8 drive letters are assigned to the first 8 slices of that media. If you have two large storage devices, you will see that each device is allocated four drive letters.

Referring to slices within a storage device is done by appending a : where is the device relative slice number from 0-255. For example, if you have an IDE device, it will show up as IDE0: in the boot messages meaning the first IDE device. To refer to the fourth slice of IDE0, you would type “IDE0:3”. Here are some examples:

`IDE0:0`	First slice of disk in IDE0
`IDE0:`	First slice of disk in IDE0
`IDE0:3`	Fourth slice of disk in IDE0

So, if you wanted to use drive letter L: to refer to the fourth slice of IDE0,

you could use the command `ASSIGN L:=IDE0:3`. There are a couple of rules to be aware of when assigning drive letters. First, you may only refer to a specific device/slice with one drive letter at a time. Said another way, you cannot have multiple drive letters referring to a the same device/slice at the same time. Second, there must always be a drive assigned to A:. Any attempt to violate these rules will be blocked by the `ASSIGN` command.

In case this wasn't already clear, you **cannot** refer directly to slices using CP/M. CP/M only understands drive letters, so to access a given slice, you must assign a drive letter to it first.

While it may be obvious, you cannot use slices on any media less than 8MB in size. Specifically, you cannot slice RAM disks, ROM disks, floppy disks, etc. All of these are considered to have a single slice and any attempt to ASSIGN a drive letter to a slice beyond that will result in an error message.

It is very important to understand that RomWBW slices are not individually created or allocated on your hard disk. RomWBW uses a single, large chunk of space on your hard disk to contain the slices.
You should think of slices as just an index into a sequential set of 8MB areas that exist in this large chunk of space. The next section will go into more detail on how slices are located on your hard disk.

Although you do not need to allocate slices individually, you do need to initialize each slice for CP/M to use it. This is somewhat analogous to doing a FORMAT operation on other systems. With RomWBW you use the `CLRDIR` command to do this. This command is merely "clearing out" the directory space of the slice referred to by a drive letter and setting up the new empty directory. Since `CLRDIR` works on drive letters, make absolutely sure you know what media and slice are assigned to that drive letter before using `CLRDIR` because CLRDIR will wipe out any pre-existing contents of the slice.

Here is an example of using `CLRDIR`. In this example, the `ASSIGN` command is used to show the current drive letter assignments. Then the `CLRDIR` command is used to initialize the directory of drive 'G' which is slice 2 of hard

disk device IDE0 (“IDE0:2”).

```
B>assign

   A:=MD0:0
   B:=MD1:0
   C:=FD0:0
   D:=FD1:0
   E:=IDE0:0
   F:=IDE0:1
   G:=IDE0:2
   H:=IDE0:3

B>clrdir G:
CLRDIR Version 1.2 April 2020 by Max Scane

Warning - this utility will overwrite the directory sectors of Drive: G
Type Y to proceed, any key other key to exit. Y
Directory cleared.
B>
```

4.6 Hard Disk Layout

As previously discussed, when RomWBW uses a hard disk, it utilizes a chunk of space for a sequential series of slices that contain the actual CP/M filesystems referred to by drive letters.

Originally, RomWBW always used the very start of the hard disk media for the location of the slices. In this layout, slice 0 referred to the first chunk of ~8MB on the disk, slice 1 referred to the second chunk of ~8MB on the disk, and so on. The number of slices is limited to the size of the disk media – if you attempted to read/write to a slice that would exceed the disk size,

you would see I/O errors. This is considered the “legacy” disk layout for RomWBW.

RomWBW has subsequently been enhanced to support the concept of partitioning. The partition mechanism is entirely compliant with Master Boot Record (MBR) Partition Tables introduced by IBM for the PC. The Wikipedia article on the Master Boot Record is excellent if you are not familiar with them. This is considered the “modern” disk layout for RomWBW. RomWBW uses the partition type id 0x2E. RomWBW does not support extended partitions – only a single primary partition can be used.

Both the legacy and modern disk layouts continue to be fully supported by RomWBW. There are no plans to deprecate the legacy layout. In fact, the legacy format takes steps to allow a partition table to still be used for other types of filesystems such as DOS/FAT. It just does not use a partition table entry to determine the start of the RomWBW slices.

There is one more difference between the legacy and modern disk layouts that should be highlighted. The CP/M filesystem in the slices of the legacy disk layout contain 512 directory entries. The modern disk layout filesystems provide 1024 directory entries. In fact, you will subsequently see that the prefixes “hd512” and “hd1k” are used to identify disk images appropriate for the legacy and modern format.

You **cannot** mix disk layouts on a single disk device. To say it another way, the existence of a partition table entry for RomWBW on a hard disk makes it behave in the modern mode. The lack of a RomWBW partition table entry will cause legacy behavior. Adding a partition table entry on an existing legacy RomWBW hard disk will cause the existing data to be unavailable and/or corrupted. Likewise, removing the RomWBW partition entry from a modern hard disk layout will cause the same problems. It is perfectly fine for one system to have multiple hard disks with different layouts – each physical disk device is handled separately.

If you are setting up a new disk, the modern (hd1k) layout is recommended

for the following reasons:

- Larger number of directory entries per filesystem
- Simplifies creation of coresident FAT filesystem
- Reduces chances of data corruption

Section 5

Disk Initialization

With some understanding of how RomWBW presents disk space to the operating systems, we need to go over the options for actually setting up your disk(s) with content.

Since it would be quite a bit of work to transfer over all the files you might want initially to your disk(s), RomWBW provides a much easier way to get initial contents on your disks. You can use your modern Windows, Linux, or Mac computer to copy a disk image onto the disk media, then just move the media over to your RomWBW computer. RomWBW comes with a variety of disk images that are ready to use and have a much more complete set of files than you will find on the ROM disk. This process is covered below under Disk Images.

If you do not want to start with pre-built disk images, you can alternatively initialize the media in-place using your RomWBW system.
Essentially, this means you are creating a set of blank directories on your disk so that files can be saved there. This process is described below under Disk Initialization. In this scenario, you will need to subsequently copy any files you want to use onto the newly initialized disk (see Transferring Files).

5.1 Using Disk Images

As mentioned previously, RomWBW includes a variety of disk images that contain a full set of applications for the operating systems supported. It is generally easier to use these disk images instead of copying all the files over using XModem. You use your modern computer (Windows, Linux, MacOS) to place the disk image onto the disk media, then just move the media over to your system. In this scenario you **do not** run `CLRDIR` or `SYSCOPY` on the slices that contain the data. When using this method, the disk will be partitioned and setup with 1 or more slices containing ready-to-run bootable operating systems.

To copy the disk image files onto your actual media (floppy disk, CF Card, SD Card, etc.), you need to use an image writing utility on your modern computer. Your modern computer will need to have an appropriate interface or slot that accepts the media. To actually copy the image, you can use the `dd` command on Linux or MacOS. On Windows, in the "Tools" directory of the distribution there are two tools you can use. For floppy media, you can use RawWriteWin and for hard disk media, you can use Win32DiskImager. In all cases, the image file should be written to the media starting at the very first block or sector of the media. This will **destroy** any other data on the media.

The disk image files are found in the Binary directory of the distribution. Floppy disk images are prefixed with "fd_" and hard disk images are prefixed with "hd_". The floppy images are specifically for 1.44M floppy media only. Each disk image has the complete set of normal applications and tools distributed with the associated operating system or application suite.

The following table shows the disk image files available. Note that the images in the "Hard" column are fine for use on CF Cards, SD Cards, as well as real spinning hard disks.

Floppy	Hard	Description
fd_cpm22.img	hd_cpm22.img	DRI CP/M 2.2 boot disk
fd_zsdos.img	hd_zsdos.img	ZSDOS 1.1 boot disk
fd_nzcom.img	hd_nzcom.img	NZCOM boot disk
fd_cpm3.img	hd_cpm3.img	DRI CP/M 3 boot disk
fd_zpm3.img	hd_zpm3.img	ZPM3 boot disk
fd_ws4.img	hd_ws4.img	WordStar v4 application disk

In addition to the disk images above, there is also a special hard disk image called hd_combo.img. This image contains all of the images above, but in a single image with 6 slices. At the boot loader prompt, you can choose a disk with the combo image, then select the specific slice you want. This allows a single disk to have all of the possible operating system options.

This is the layout of the hd_combo disk image:

Slice	Description
Slice 0	DRI CP/M 2.2 boot disk
Slice 1	ZSDOS 1.1 boot disk
Slice 2	NZCOM boot disk
Slice 3	DRI CP/M 3 boot disk
Slice 4	ZPM3 boot disk
Slice 5	WordStar v4 application disk

Note that unlike the ROM firmware, you do **not** need to choose a disk image specific to your hardware. Because the RomWBW firmware provides a hardware abstraction layer, all hard disk images will work on all hardware variations. Yes, this means you can remove an SD Card from one RomWBW system and put it in a different RomWBW system. The only constraint is that the applications on the disk media must be up to date with the firmware on the system being used.

All of the disk images that indicate they are bootable (boot disk) will boot from disk as is. You do not need to run `SYSCOPY` on them to make them bootable. However, if you upgrade your ROM, you should use `SYSCOPY` to update the system tracks.

A full implementation of the UCSD p-System for Z80 under RomWBW is also provided. This is a completely separate and standalone disk image called psys.img. It contains 6 p-System filesystem slices, but these are not interoperable with the CP/M slices described above.

5.2 Raw Disk Initialization

To use a disk device, you will need to initialize the individual directories of each slice you want to use. On RomWBW, slice initialization is done using the CLRDIR application. For example if your C: drive has been assigned to a slice on a storage device, you would use `CLRDIR C:` to initialize C: and prepare it hold files. Note that CLRDIR will prompt you for confirmation and you must respond with a **capital** 'Y' to confirm. Once `CLDIR` has completed, you can copy files onto the drive, for example `COPY *.* C:`. Be very careful to pay attention to your drive letter assignments prior to running `CLRDIR` to avoid accidentally wiping out a filesystem that has data on it.

Running `CLRDIR` on a disk slice is roughly equivalent to running FORMAT on MS-DOS. Note that unlike MS-DOS you do **not** partition your mass storage device. CP/M knows nothing about disk partitions. You may notice a partitioning application on your ROM disk (FDISK80), but this is strictly for an advanced technique of managing the RomWBW slices container or MS-DOS FAT filesystem on your media. Do not use FDISK80 unless you are specifically attempting to change the existing partitions (not slices).

If you are using a floppy drive, you will need to physically format your floppy disk prior to use. This is only required for floppy disks, not hard disk, CF Cards, or SD Cards, etc. To format a floppy drive, you can use the interactive

application FDU. FDU is not terribly user friendly, but is generally documented in the file "FDU.txt" found in the Doc directory of the distribution. It is not necessary to run CLRDIR on a floppy disk after physically formatting it – the directory is cleared as part of the formatting.

Once you have initialized a disk device and copied your desired files onto it, you may want to make the disk bootable. On CP/M filesystems, you must perform one additional step to make a disk bootable. Specifically, you need to place a copy of the operating system on the system tracks of the disk. This is done using the SYSCOPY command. Let's say you have prepared drive C: by initializing it with CLRDIR and copied some files onto it. You can now make C: bootable by running the following command:

```
B>SYSCOPY C:=B:ZSYS.SYS
```

This command means: copy the Z-System operating system onto the system tracks of drive C:. In this example, it is assumed that you have booted from ROM, so B: is the ROM disk drive. Additionally, this example assumes you want the Z-System operating system to be booted from C:. If you want CP/M 2.2 instead, you would replace B:ZSYS.SYS with B:CPM.SYS. Here is a full example of this process.

```
B>SYSCOPY C:=B:ZSYS.SYS

SYSCOPY v2.0 for RomWBW CP/M, 17-Feb-2020 (CP/M 2 Mode)
Copyright 2020, Wayne Warthen, GNU GPL v3

Transfer system image from B:ZSYS.SYS to C: (Y/N)? Y
Reading image... Writing image... Done
```

Once this process succeeds, you will be able to boot directly to the disk from the boot loader prompt. See the instructions in Booting Disks for details on this.

Section 6

Operating Systems

One of the primary goals of RomWBW is to expose a set of generic hardware functions that make it easy to adapt operating systems to any hardware supported by RomWBW. As a result, there are now 5 operating systems that have been adapted to run under RomWBW. The adaptations are identical for all hardware supported by RomWBW because RomWBW hides all hardware specifics from the operating system.

Note that all of the operating systems included with RomWBW support the same basic filesystem format. As a result, a formatted filesystem will be accessible to any operating system. The only possible issue is that if you turn on date/time stamping using the newer OSes, the older OSes will not understand this. Files will not be corrupted, but the date/time stamps will not be maintained.

The following sections briefly describe the operating system options currently available.

6.1 Digital Research CP/M 2.2

This is the most widely used variant of the Digital Research operating system. It has the most basic feature set, but is essentially the compatibility metric for all other CP/M-like operating systems including all of those listed below. The Doc directory contains a manual for CP/M usage ("CPM Manual.pdf"). If you are new to the CP/M world, I would recommend using this CP/M variant to start with simply because it is the most stable and you are less likely to encounter problems.

Notes

- You can change media, but it must be done while at the OS command prompt and you **must** warm start CP/M by pressing ctrl-c. This is a CP/M 2.2 constraint and is well documented in the DRI manuals.
- The original versions of DDT, DDTZ, and ZSID used the RST 38 vector which conflicts with interrupt mode 1 use of this vector. The DDT, DDTZ, and ZSID applications in RomWBW have been modified to use RST 30 to avoid this issue.
- Z-System applications will not run under CP/M 2.2. For example, the `LDDS` date stamper will not run.

6.2 ZSDOS 1.1

ZSDOS is the most popular non-DRI CP/M "clone" which is generally referred to as Z-System. Z-System is intended to be an enhanced version of CP/M and should run all CP/M 2.2 applications. It is optimized for the Z80 CPU (as opposed to 8080 for CP/M) and has some significant improvements such as date/time stamping of files. For further information on the RomWBW implementation of Z-System, see the wiki page Z-System Notes. Additionally, the

official documentation for Z-System is included in the RomWBW distribution Doc directory (“ZSDOS Manual.pdf” and “ZCPR Manual.pdf”).

Notes

- Although most CP/M 2.2 applications will run under Z-System, some may not work as expected. The best example is PIP which is not aware of the ZSDOS paths and will fail in some scenarios (use `COPY` instead).
- Although ZSDOS can recognize a media change in some cases, it will not always work. You should only change media at a command prompt and be sure to warm start the OS with a ctrl-c.

6.3 NZCOM Automatic Z-System

NZCOM is a much further refined version of Z-System (ZCPR 3.4). NZCOM was sold as an enhancement for existing users of CP/M 2.2 or ZSDOS. For this reason, (by design) NZCOM does not provide a way to boot directly from disk. Rather, it is loaded after the system boots into a host OS. On the RomWBW NZCOM disk images, the boot OS is ZSDOS 1.1. After you configure NZCOM, you can add a `PROFILE.SUB` file to automatically launch NZCOM at boot.

NZCOM is not pre-configured. You must run through a simple configuration process before loading it. Run MKZCM to do this.

NZCOM has substantially more functionality than CP/M or basic Z-System. It is important to read the the “NZCOM Users Manual.pdf” file in the RomWBW Doc directory.

Notes

- There is no `DIR` command, you must use `SDZ` instead. If you don't like this, look into the ALIAS facility.

6.4 Digital Research CP/M 3

This is the Digital Research follow-up product to their very popular CP/M 2.2 operating system. While highly compatible with CP/M 2.2, it features many enhancements. It makes direct use of banked memory to increase the user program space (TPA). It also has a new suite of support tools and help system.

Note that to make a CP/M 3 boot disk, you actually place CPMLDR.SYS on the system tracks of the disk. You do not place CPM3.SYS on the system tracks. `CPMLDR.SYS` chain loads `CPM3.SYS`.

Notes

- The `DATE` command cannot yet be used to **set** the RTC. The RTC is used to read the current date/time for file stamping, etc. You can use the `RTC` app to set the RTC clock.
- The `COPYSYS` command described in the DRI CP/M 3 documentation is not provided with RomWBW. The RomWBW `SYSCOPY` command is used instead.
- Although CP/M 3 is generally able to run CP/M 2.2 programs, this is not universally true. This is especially true of the utility programs included with the operating system. For example, the `SUBMIT` program of CP/M 3 is completely different from the `SUBMIT` program of CP/M 2.2.

6.5 Simeon Cran's ZPM3

ZPM3 is an interesting combination of the features of both CP/M 3 and ZCPR 3. Essentially, it has the features of and compatibility with both.

Like CP/M 3, to make ZPM3 boot disk, you put ZPMLDR.SYS on the system tracks of the disk.

Notes

- `ZPMLDR` is equivalent to CPMLDR. Both are included. Previously, ZPMLDR had issues that prevented it from properly booting RomWBW ZPM3. However, those issues have been resolved.
- The ZPM operating system is contained in the file called CPM3.SYS which is confusing, but this is as intended by the ZPM3 distribution. I believe it was done this way to make it easier for users to transition from CP/M 3 to ZPM3.

6.6 QPM

QPM is another OS providing compatibility with and enhancements to CP/M 2.2. It is provided as bootable disk images for RomWBW.

The following documentation files from the original QPM distribution are included in the RomWBW Doc directory:

- QP/M 2.7 Features and Facilities (qcp27.pdf)
- QP/M 2.7 Interface Guide (qdos27.pdf)
- QP/M 2.7 Installation Guide and Supplements (qpm27.pdf)

Refer to the ReadMe.txt file in Source/Images/d_qpm for more details regarding the RomWBW adaptation and customizations.

Notes

- QPM is not available as source. This implementation was based on the QPM binary distribution and has been minimally customized for RomWBW.
- QINSTALL is used to customize QPM. It is included on the disk image. You should review the notes in the ReadMe.txt file in Source/Image/d_qpm before making changes.

6.7 UCSD p-System

This is a full implementation of the UCSD p-System IV.0 for Z80 running under RomWBW. Unlike the OSes above, p-System uses it's own unique filesystem and is not interoperable with other OSes.

It was derived from the p-System Adaptable Z80 System. Unlike some other distributions, this implements a native p-System Z80 Extended BIOS, it does not rely on a CP/M BIOS layer.

The p-System is provided on a hard disk image file called psys.img. This must be copied to it's own dedicated hard disk media (CF Card, SD Card, etc.). It is booted by selecting slices 0 of the corresponding hard disk unit at the RomWBW Boot Loader prompt.

The p-System Users Manual is included in the Doc directory of the distribution as "UCSD p-System Users Manual.pdf".

Refer to the ReadMe.txt file in Source/pSys for more details.

Notes

- There is no floppy support at this time.
- The hard disk image contains 6 p-System slices which are assigned to p-System unit numbers 4, 5, 9, 10, 11 which is standard for p-System.

Slices 0-5 are assigned sequentially to these p-System unit numbers and it is not possible to reassign them.

- p-System relies heavily on the use of a full screen terminal. This implementation has been setup to expect an ANSI or DEC VT-100 terminal or emulator. The screen output will be garbled if no such terminal or emulator is used for console output.
- There is no straightforward mechanism to move files in and out of p-System. However, the .vol files in Source/pSys can be read and modified by CiderPress. CiderPress is able to add and remove individual files.

6.8 FreeRTOS

Phillip Stevens has ported FreeRTOS to run under RomWBW. FreeRTOS is not provided in the RomWBW distribution. FreeRTOS is available under the MIT licence and further general information is available at FreeRTOS.

You can also contact Phillip for detailed information on the Z180 implementation of FreeRTOS for RomWBW. feilipu

Section 7

RomWBW Custom Applications

The operation of the RomWBW hosted operating systems is enhanced through several custom applications. You have already read about one of these – the `ASSIGN` command. These applications are functional on all of the OS variants included with RomWBW.

The applications discussed here are **not** the same as the built-in ROM applications mentioned previously. These applications run as commands within the operating systems provided by RomWBW. So, these commands are only available at an operating system prompt after an operating system has been loaded.

All of the RomWBW Custom Applications are built to function under all of the RomWBW Operating Systems (except for p-System). In general, the applications will automatically adapt as needed to the currently running operating system. One exception is `FDU` – the Floppy Disk Utility. This application requires that you pick the floppy disk interface you want to interact with.

There is more complete documentation of all of these applications in the related RomWBW manual "RomWBW Applications" found in the Doc directory of the distribution.

The following custom applications are found on the ROM disk and are, therefore, globally available.

Application	Description
ASSIGN	Add, change, and delete drive letter assignments. Use ASSIGN /? for usage instructions.
SYSCOPY	Copy system image to a device to make it bootable. Use SYSCOPY with no parms for usage instructions.
MODE	Reconfigures serial ports dynamically.
FDU	Format and test floppy disks. Menu driven interface.
FORMAT	Will someday be a command line tool to format floppy disks. Currently does nothing!
XM	XModem file transfer program adapted to hardware. Automatically uses primary serial port on system.
FLASH	Will Sowerbutts' in-situ ROM programming utility.
FDISK80	John Coffman's Z80 hard disk partitioning tool. See documentation in Doc directory.
TALK	Direct console I/O to a specified character device.
RTC	Manage and test the Real Time Clock hardware.
TIMER	Display value of running periodic system timer.
CPUSPD	Change the running CPU speed and wait states of the system.

Some custom applications do not fit on the ROM disk. They are found on the disk image files or the individual files can be found in the Binary\Apps directory of the distribution.

Application	Description
TUNE	Play .PT2, .PT3, .MYM audio files.
FAT	Access MS-DOS FAT filesystems from RomWBW (based on FatFs).
INTTEST	Test interrupt vector hooking.

Section 8

Transferring Files

Transferring files between your modern computer and your RomWBW system can be achieved in a variety of ways. The most common of these are described below. All of these have a certain degree of complexity and I encourage new users to use the available community forums to seek assistance as needed.

8.1 Serial Port Transfers

RomWBW provides an serial file transfer program called XModem that has been adapted to run under RomWBW hardware. The program is called `XM` and is on your ROM disk as well as all of the pre-built disk images.

You can type `XM` by itself to get usage information. In general, you will run `XM` with parameters to indicate you want to send or receive a file on your RomWBW system. Then, you will use your modern computers terminal program to complete the process.

The `XM` application generally tries to detect the hardware you are using and adapt to it. However, you must ensure that you have a reliable serial connection. You must also ensure that the speed of the connection is not too

fast for XModem to service. Alternatively, you can ensure that hardware flow control is working properly.

There is an odd interaction between XModem and partner terminal programs that can occur. Essentially, after launching `XM`, you must start the protocol on your modern computer fairly quickly (usually in about 20 seconds or so). So, if you do not pick a file on your modern computer quickly enough, you will find that the transfer completes about 16K, then hangs. The interaction that causes this is beyond the scope of this document.

8.2 Disk Image Transfers

It is possible to pass disk images between your RomWBW system and your modern computer. This assumes you have an appropriate media slot on your modern computer for the media you want to use (CF Card, SD Card, or floppy drive).

The general process to get files from your modern computer to a RomWBW computer is:

1. Use `cpmtools` on your modern computer to create a RomWBW CP/M filesystem image.
2. Insert your RomWBW media (CF Card, SD Card, or floppy disk) in your modern computer.
3. Use a disk imaging tool to copy the RomWBW filesystem image onto the media.
4. Move the media back to the RomWBW computer.

This process is a little complicated, but it has the benefit of allowing you to get a lot of files over to your RomWBW system quickly and with little chance of corruption.

The process can be run in reverse to get files from your RomWBW computer

to a modern computer.

The exact use of these tools is a bit too much for this document, but the tools are all included in the RomWBW distribution along with usage documents.

Note that the build scripts for RomWBW create the default disk images supplied with RomWBW. It is relatively easy to customize the contents of the disk images that are part of RomWBW. This is described in more detail in the Source\Images directory of the distribution.

8.3 FAT Filesystem Transfers

RomWBW provides a mechanism that allows it to read and write files on a FAT formatted disk. This means that you can generally use your modern computer to make an SD Card or CF Card with a standard FAT32 filesystem on it, then place that media in your RomWBW computer and access the files.

When formatting the media on your modern computer, be sure to pick the FAT filesystem. NTFS and other filesystems will not work.

On your RomWBW computer you can use the `FAT` application to access the FAT media. The `FAT` application allows you to read files, write files, list a directory, and erase files on the FAT media. It can handle subdirectories as well. It will only see 8.3 character filenames however. Longer filenames will show up as a truncated version.

The `FAT` application is not on your ROM disk because it is too large to fit. You will find it on all of the pre-built disk images as well as in the Binary\Apps directory of the distribution.

For advanced users, it is possible to create a hybrid disk that contains CP/M slices at the beginning and a FAT filesystem after. Such a hybrid disk can be used to boot an operating system and still have access to FAT files on the FAT portion of the disk. David Reese has prepared a document describing how to do this. It is called “SC126_How-

To_No_2_Preparing_an_SD_Card_for_Use_with_SC126_Rev_1-5.pdf" and can be found in the Doc\Contrib directory of the distribution.

Section 9

Customizing RomWBW

9.1 Startup Command Processing

Most of the operating systems supported by RomWBW provide a mechanism to run commands at boot. This is similar to the AUTOEXEC.BAT files from MS-DOS.

With the exception of ZPM3 and p-System, all operating systems will look for a file called `PROFILE.SUB` on the system drive at boot. If it is found, it will be processed as a standard CP/M submit file. You can read about the use of the SUBMIT facility in the CP/M manuals included in the RomWBW distribution. Note that the boot disk must also have a copy of `SUBMIT.EXE`.

Note that the automatic startup processing generally requires booting to a disk drive. Since the ROM disk is not writable, there is no simple way to add/edit a `PROFILE.SUB` file there. If you want to customize your ROM and add a `PROFILE.SUB` file to the ROM Disk, it will work, but is a lot harder than using a boot disk.

In the case of ZPM3, the file called `STARTZPM.COM` will be run at boot. To customize this file, you use the ZCPR ALIAS facility. You will need to refer to ZCPR documentation for more information on the ALIAS facility.

p-System has it's own startup command processing mechanism that is covered in the p-System documentation.

9.2 ROM Customization

The pre-built ROM images are configured for the basic capabilities of each platform. Additionally, some of the typical add-on hardware for each platform will be automatically detected and used. If you want to go beyond this, RomWBW provides a very flexible configuration mechanism based on configuration files. Creating a customized ROM requires running a build script, but it is quite easy to do.

Essentially, the creation of a custom ROM is accomplished by updating a small configuration file, then running a script to compile the software and generate the custom ROM and disk images. There are build scripts for Windows, Linux, and MacOS to accommodate virtually all users. All required build tools (compilers, assemblers, etc.) are included in the distribution, so it is not necessary to setup a build environment on your computer.

RomWBW can be built on modern Windows, Linux, or MacOS computers. The process for building a custom ROM is documented in the ReadMe.txt file in the Source directory of the distribution.

For those who are interested in more than basic system customization, note that all source code is provided (including the operating systems). Modification of the source code is considered an expert level task and is left to the reader to pursue.

Note that the ROM customization process does not apply to UNA. All UNA customization is performed within the ROM setup script that is built into the ROM.

Section 10

UNA Hardware BIOS

John Coffman has produced a new generation of hardware BIOS called UNA. The standard RomWBW distribution includes it's own hardware BIOS. However, RomWBW can alternatively be constructed with UNA as the hardware BIOS portion of the ROM. If you wish to use the UNA variant of RomWBW, then just program your ROM with the ROM image called "UNA_std.rom" in the Binary directory. This one image is suitable on **all** of the platforms and hardware UNA supports.

UNA is customized dynamically using a ROM based setup routine and the setup is persisted in the system NVRAM of the RTC chip. This means that the single UNA-based ROM image can be used on most of the RetroBrew platforms and is easily customized. UNA also supports FAT file system access that can be used for in-situ ROM programming and loading system images.

While John is likely to enhance UNA over time, there are currently a few things that UNA does not support:

- Floppy Drives
- Terminal Emulation
- Zeta 1, N8, RC2014, Easy Z80, and Dyno Systems

- Some older support boards

The UNA version embedded in RomWBW is the latest production release of UNA. RomWBW will be updated with John's upcoming UNA release with support for VGA3 as soon as it reaches production status.

Please refer to the UNA BIOS Firmware Page for more information on UNA.

Section 11

Upgrading

Upgrading to a newer release of RomWBW is essentially just a matter of updating the ROM chip in your system. If you have spare ROM chips for your system and a ROM programmer, it is always safest to retain your existing, working ROM chip and program a new one with the new firmware. If the new one fails to boot, you can easily return to the known working ROM.

Prior to attempting to reprogram your actual ROM chip, you may wish to "try" the update to ensure it will work on your system. With RomWBW, you can upload a new ROM image executable and load it from the command line. For each ROM image file (.rom) in the Binary directory, you will find a corresponding application file (.com). For example, for SBC_std.rom, there is also an SBC_std.com file. You can upload the .com file to your system using XModem, then simply run the .com file. You will see your system go through the normal startup process just like it was started from ROM. However, your ROM has not been updated and the next time you boot your system, it will revert to the system image contained in ROM.

11.1 Upgrading via Flash Utility

If you do not have easy access to a ROM programmer, it is usually possible to reprogram your system ROM using the FLASH utility from Will Sowerbutts. This application, called FLASH.COM, can be found on the ROM drive of any running system. In this case, you would need to transfer the new ROM image (.rom) over to your system using XModem (or one of the other mechanisms described in the Transferring Files section). The ROM image is too large to fit on your RAM drive, so you will need to transfer it to a larger storage drive. Once the ROM image is on your system, you can use the FLASH application to update your ROM. The following is a typical example of transferring ROM image using XModem and flashing the chip in-situ.

```
E>xm r rom.rom

XMODEM v12.5 - 07/13/86
RBC, 28-Aug-2019 [WBW], ASCI

Receiving: E0:ROM.IMG
7312k available for uploads
File open - ready to receive
To cancel: Ctrl-X, pause, Ctrl-X

Thanks for the upload

E>flash write rom.rom
FLASH4 by Will Sowerbutts <will@sowerbutts.com> version 1.2.3

Using RomWBW (v2.6+) bank switching.
Flash memory chip ID is 0xBFB7: 39F040
Flash memory has 128 sectors of 4096 bytes, total 512KB
Write complete: Reprogrammed 2/128 sectors.
Verify (128 sectors) complete: OK!
```

Obviously, there is some risk to this approach since any issues with the programming or ROM image could result in a non-functional system.

To confirm your ROM chip has been successfully updated, restart your system and boot an operating system from ROM. Do not boot from a disk device yet. Review the boot messages to see if any issues have occurred.

11.2 Upgrading via XModem Flash Updater

Similar to using the Flash utility, the system ROM can be updated or upgraded through the ROM based updater utility. This works by by reprogrammed the flash ROM as the file is being transfered.

This has the advantage that secondary storage is not required to hold the new image.

From the Boot Loader menu select X (Xmodem Flash Updater) and then U (Begin Update). Then initiate the Xmodem transfer of the .img or .upd file.

More information can be found in the ROM Applications document.

11.3 Post Upgrade System Image and Application Update Process

Once you are satisfied that the ROM is working well, you will need to update the system images and RomWBW custom applications on your disk drives. The system images and custom applications are matched to the RomWBW ROM firmware in use. If you attempt to boot a disk or run applications that have not been updated to match the current ROM firmware, you are likely to have odd problems.

The simplest way to update your disk media is to just use your modern computer to overwrite the entire media with the latest disk image of your

choice. This process is described below in the Disk Images section. If you wish to update existing disk media in your system, you need to perform the following steps.

If the disk is bootable, you need to update the system image on the disk using the procedure described below corresponsing to the operating system on your disk.

- **CP/M 2.2**

  Boot to CP/M 2.2 from ROM, then use `SYSCOPY` to update the system image on **all** CP/M 2.2 boot disks/slices. The CP/M 2.2 system image is called CPM.SYS and is found on the ROM disk. For example:

  ```
  B>SYSCOPY C:=CPM.SYS
  ```

- **ZSDOS**

  Boot to Z-System from ROM, then use `SYSCOPY` to update the system image on **all** ZSDOS boot disks/slices. The ZSDOS system image is called ZSYS.SYS and is found on the ROM disk. For example:

  ```
  B>SYSCOPY C:=ZSYS.SYS
  ```

- **NZCOM**

  NZCOM runs on top of either CP/M 2.2 or ZSDOS. By default, the RomWBW disk image for NZCOM uses ZSDOS. Follow the corresponding procedure above to update the system image on the NZCOM boot disks/slices.

- **CP/M 3**

  CP/M 3 uses a multi-step boot process involving multiple files. The CP/M 3 boot files are not included on the ROM disk due to space constraints. You will need to transfer the files to your system from the RomWBW distribution directory Binary\CPM3.

  After this is done, you will need to use `SYSCOPY` to place the CP/M 3

loader image on the boot tracks of all CP/M 3 boot disks/slices. The loader image is called `CPMLDR.SYS`. You must then copy (at a minimum) `CPM3.SYS` and `CCP.COM` onto the disk/slice. Assuming you copied the CP/M 3 boot files onto your RAM disk at A:, you would use:

```
A>B:SYSCOPY C:=CPMLDR.SYS
A>B:COPY CPM3.SYS C:
A>B:COPY CCP.COM C:
```

- **ZPM3**

  ZPM3 uses a multi-step boot process involving multiple files. The ZPM3 boot files are not included on the ROM disk due to space constraints. You will need to transfer the files to your system from the RomWBW distribution directory Binary\ZPM3.

  After this is done, you will need to use `SYSCOPY` to place the ZPM3 loader image on the boot tracks of all ZPM3 boot disks/slices. The loader image is called `ZPMLDR.SYS`. You must then copy (at a minimum) `CPM3.SYS`, `ZCCP.COM`, `ZINSTAL.ZPM`, and `STARTZPM.COM` onto the disk/slice. Assuming you copied the ZPM3 boot files onto your RAM disk at A:, you would use:

  ```
  A>B:SYSCOPY C:=ZPMLDR.SYS
  A>B:COPY CPM3.SYS C:
  A>B:COPY ZCCP.COM C:
  A>B:COPY ZINSTAL.ZPM C:
  A>B:COPY STARTZPM.COM C:
  ```

  You may be wondering if the reference to `CPM3.SYS` is a typo. It is not. The ZPM3 main system code file is called `CPM3.SYS` which is the same name as CP/M 3 uses, but the file contents are not the same.

Finally, if you have copies of any of the RomWBW custom applications on your hard disk, you need to update them with the latest copies. The following applications are found on your ROM disk. Use COPY to copy them over any

older versions of the app on your disk:

- ASSIGN.COM
- SYSCOPY.COM
- MODE.COM
- FDU.COM (was FDTST.COM)
- FORMAT.COM
- XM.COM
- FLASH.COM
- FDISK80.COM
- TALK.COM
- RTC.COM
- TIMER.COM
- INTTEST.COM

For example: `B>COPY ASSIGN.COM C:`

Some RomWBW custom applications are too large to fit on the ROM disk. If you are using any of these you will need to transfer them to your system and then update all copies. These applications are found in the Binary\Apps directory of the distribution and in all of the disk images.

- FAT.COM
- TUNE.COM

11.4 System Update

If the system running ROMWBW utilizes the SST39SF040 Flash chip then it is possible to do a System Update in place of a System Upgrade in some cases.

A System Update would involve only updating the BIOS, ROM applications and CP/M system.

A System Update may be more favorable than a System Upgrade in cases

such as:

- Overwriting of the ROM drive is not desired.
- Space is unavailable to hold a full ROMWBW ROM.
- To mimimize time taken to transfer and flash a full ROM.
- Configuration changes are only minor and do not impact disk applications.

The ROMWBW build process generates a system upgrade file along with the normal ROM image and can be identified by the extension ".upd". It will be 128Kb in size. In comparison the normal ROM image will have the extension ".rom" and be 512Kb or 1024Kb in size.

Transferring and flashing the System Update is accomplished in the same manner as described above in *Upgrading* with the required difference being that the flash application needs to be directed to complete a partial flash using the /P command line switch.

```
E>FLASH WRITE ROM.UPD /P
```

Section 12

Acknowledgments

I want to acknowledge that a great deal of the code and inspiration for RomWBW has been provided by or derived from the work of others in the RetroBrew Computers Community. I sincerely appreciate all of their contributions. The list below is probably missing many names – please let me know if I missed you!

- Andrew Lynch started it all when he created the N8VEM Z80 SBC which became the first platform RomWBW supported. Some of his code can still be found in RomWBW.
- Dan Werner wrote much of the code from which RomWBW was originally derived and he has always been a great source of knowledge and advice.
- Douglas Goodall contributed code, time, testing, and advice in “the early days”. He created an entire suite of application programs to enhance the use of RomWBW. Unfortunately, they have become unusable due to internal changes within RomWBW. As of RomWBW 2.6, these applications are no longer provided.
- David Giles created support for the Z180 CSIO which is now included SD Card driver.

- Ed Brindley contributed some of the code that supports the RC2014 platform.
- Phil Summers contributed the Forth and BASIC adaptations in ROM, the AY-3-8910 sound driver as well as a long list of general code enhancements.
- Phillip Stevens contributed support for FreeRTOS.
- Curt Mayer contributed the original Linux / MacOS build process.
- UNA BIOS and FDISK80 are the products of John Coffman.
- FLASH4 is a product of Will Sowerbutts.
- CLRDIR is a product of Max Scane.
- Tasty Basic is a product of Dimitri Theulings.
- Dean Netherton contributed the sound driver interface and the SN76489 sound driver.
- The RomWBW Disk Catalog document was produced by Mykl Orders.

Contributions of all kinds to RomWBW are very welcome.

Section 13

Licensing

RomWBW is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

RomWBW is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with RomWBW. If not, see https://www.gnu.org/licenses/.

Portions of RomWBW were created by, contributed by, or derived from the work of others. It is believed that these works are being used in accordance with the intentions and/or licensing of their creators.

If anyone feels their work is being used outside of it's intended licensing, please notify:

Wayne Warthen wwarthen@gmail.com

RomWBW is an aggregate work. It is composed of many individual, stan-

dalone programs that are distributed as a whole to function as a cohesive system. Each program may have it's own licensing which may be different from other programs within the aggregate.

In some cases, a single program (e.g., CP/M Operating System) is composed of multiple components with different licenses. It is believed that in all such cases the licenses are compatible with GPL version 3.

RomWBW encourages code contributions from others. Contributors may assert their own copyright in their contributions by annotating the contributed source code appropriately. Contributors are further encouraged to submit their contributions via the RomWBW source code control system to ensure their contributions are clearly documented.

All contributions to RomWBW are subject to this license.

Section 14

Getting Assistance

The best way to get assistance with RomWBW or any aspect of the RetroBrew Computers projects is via one of the community forums:

- RetroBrew Computers Forum
- RC2014 Google Group
- retro-comp Google Group

Submission of issues and bugs are welcome at the RomWBW GitHub Repository.

Also feel free to email Wayne Warthen at wwarthen@gmail.com.