RomWBW

Getting Started

Wayne Warthen

Monday 30 March 2020

Contents

RomWBW

Z80/Z180 System Software

Version 3.1 Pre-release
Monday 30 March 2020

Wayne Warthen wwarthen@gmail.com

Download

- RomWBW Distribution Package

Related Pages

- RomWBW Architecture Document
- RomWBW Applications
- RomWBW Errata

Overview

RomWBW provides a complete software system for a wide variety of hobbyist Z80/Z180 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
- Real time clock drivers including DS1322, BQ4845

- Multiple OS support including CP/M 2.2, ZSDOS, CP/M 3, ZPM3
- 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 optimal 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 are included – no need to install assemblers, etc. Any modern computer running Windows, Linux, or MacOS can be used.

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.

Installation

The latest RomWBW distribution downloads are maintained on GitHub in the RomWBW Repository. The fully-built distributions are found on the releases page of the repository. On this page, you will probably see both pre-releases as well as normal releases. Unless you have a specific reason, I suggest you stick to the most recent normal release (not pre-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 are Source Code only and do not have the pre-built ROM or disk images.

The pre-built ROM images will automatically detect and support a reasonable range of devices 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.

In general, you will just program your system's ROM chip with the appropriate ROM image from the RomWBW distribution. Depending on how you got your system, 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 an included ROM Flashing tool (Will Sowerbutts' FLASH application) as described in the Upgrading section below.

Looking at the extracted distribution archive, You will see that the distribution is broken up into a few sub-directories. The Binary directory 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
RC Z80	RCZ80_std.rom	115200	RC2014 w/ Z80 CPU, requires 512K RAM/ROM module
RC Z180*	RCZ180_ext.rom	115200	RC2014 w/ Z180 CPU & 512K banked RAM/ROM module
RC Z180*	RCZ180_nat.rom	115200	RC2014 w/ Z180 CPU & 512K native RAM/ROM module
Easy Z80	EZZ80_std.rom	115200	Sergey Kiselev's Easy Z80
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
Dyno	DYNO_std.rom	38400	Steve Garcia's Z180 Dyno Computer

*The RC2014 Z180 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 "ext" ROM supports Spencer's official 512K RAM/ROM banked memory module. The "nat" ROM supports any of the third-party Z180 native memory modules.

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. All pre-built ROM images are simple 512KB binary images. If your system utilizes a larger ROM chip, you can just program the image into the first 512KB of 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.

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.

Initially, you should try the ROM boot options. By selecting either CP/M 2.2 or Z-System, the selected operating system will be loaded from ROM and you will see the a `B>` disk prompt. In this scenario, A: will be an empty RAM disk and B: will refer to your ROM disk containing some common applications. This provides a simple environment for learning to use your system. Be aware that files saved to the RAM disk (A:) will disappear at the next power on (RAM is generally not persistent). Also note that attempts to save files to the ROM disk (B:) will fail because ROM is not writable.

General Usage

Each of the operating systems and ROM applications included with RomWBW are sophisticated tools in their own right. It is not reasonable to

document their usage here. However, you will find complete manuals in PDF format in the Doc directory of the distribution. The intention of this section is to document the RomWBW specific enhancements to these operating systems.

Inbuilt ROM Applications

In addition to CP/M 2.2 and Z-System, there are several ROM applications that can be launched directly from ROM. These applications are not hosted by an operating system and so they are unable to save files to disk devices.

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

Application	
Monitor	Z80 system debug monitor w/ Intel Hex loader
Forth	Brad Rodriguez's ANSI compatible Forth language
Basic	Nascom 8K BASIC language
Tasty BASIC	Dimitri Theuling's Tiny BASIC implementation
Play	A simple video game (requires ANSI terminal emulation)

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

Space is available in the ROM image for the inclusion of other software. Any inbuilt application can be set up to launch automatically at startup.

Devices and Units

In order to support a wide variety of hardware, RomWBW HBIOS uses a modular approach to implementing device drivers and presenting devices to the operating system. In general, all devices are classified as one of the following:

- Disk (Hard Disk, CF Card, SD Card, RAM/ROM Disk, 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 the operating system does not need to understand the difference between a floppy drive and a ROM disk.

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. This information is displayed in the unit summary table at startup. 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 possible devices regardless of whether they have actual media installed at boot time.

Note that Character Unit 0 is **always** the initial system console by definition.

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.

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` do display them.

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.

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 a single 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.

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 allows you to view or change the 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 large 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 I wanted to use drive letter L: to refer to the fourth slice of IDE0, I 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. Said another way, you cannot have multiple drive letters referring to a single 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.

Unlike MS-DOS partitions, slices are not allocated – there is no partitioning for slices. Think of every hard disk type device as having a pre-allocated set of 256 8MB slices at the start of the media. You can refer to any of them simply by assigning a drive letter. RomWBW will not check to see if there is anything else on the hard disk in the slice you are referring to, nor will it verify that the hard disk media is large enough to have a slice at the location you refer to. If you attempt to write past the end of your media, you will get an I/O error displayed, so you will know if you make a mistake. There is no tracking of your use of slices – you will need to keep track of your use of slices yourself.

Nothing automatically initializes a slice as a file system. You must do that yourself using `CLRDIR`. Since `CLRDIR` works on drive letters, make absolutely sure you know what media and slice are assigned to that drive letter before using `CLRDIR`.

While it is probably obvious, you cannot use slices on any media less than 8MB in size. Specifically, you cannot slice RAM disks, ROM disks, floppy disks, etc.

RomWBW Custom Applications

The operation of the RomWBW hosted operating systems is enhanced through several custom applications. These applications are functional on all of the OS variants included with RomWBW.

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.

Application	Description
RTC	Manage and test the Real Time Clock hardware.
TIMER	Display value of running periodic system timer.
INTTEST	Test interrupt vector hooking.

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).

Additional documentation on all of these applications can be found in "RomWBW Applications.pdf" in the Doc directory of the distribution.

Using Disks

ROM & RAM Disks

RomWBW utilizes a portion of the ROM and RAM memory in your system to implement small memory-based disks.

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.

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 on ZSDOS.

Disk Devices

While the RAM/ROM disks provide a functional system, they are not useful in the long term because you cannot save data across power cycles. They are also constrained by limited space.

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. 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, you can boot an operating system and the operating system will have access to the media. At the boot loader prompt, select either either CP/M 2.2 or Z-System to boot from ROM. 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 above 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 above in the Slices section.

Once you are seeing drive letters referring to your disk media, you can follow the instructions below to begin using the disk media with the operating system. Your disk media **must** be initialized prior to being used. There are two ways to initialize your media for use.

One option is to initialize the media in-place using your RomWBW system. 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).

Alternatively, you can use your modern Windows, Linux, or Mac computer to copy a disk image onto the disk media. 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.

Disk Initialization

To use a disk device, you will need to initialize the directory of the filesystem. On RomWBW, the initialization is done using the CLRDIR application. For example if your C: drive has been assigned to 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 device 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 adding an MS-DOS FAT filesystem to your media in addition to the CP/M area. Do not use FDISK80 unless you are specifically attempting to add an MS-DOS FAT filesystem to your media.

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.

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 drive(s). The directory is prepared and the disk is already bootable, if it is an operating system boot disk image.

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

Floppy	Hard	Description
fd_nzcom.img	hd_nzcom.img	NZCOM boot disk
fd_cpm3	hd_cpm3.img	DRI CP/M 3 boot disk
fd_zpm3	hd_zpm3.img	ZPM3 boot disk
fd_ws4	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 system and put it in a different 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.

Booting Disks

When starting your system, following the hardware initialization, you will see the Boot Loader prompt. In addition, to the ROM boot options, you will see another line listing the Disk boot options. This line lists the disk devices that you can choose to boot directly.

You will notice that you do not have an option to boot a drive letter here (like C:). This is because the operating system is not yet loaded. When you ran SYSCOPY previously, remember that C: was assigned to IDE0:0 which means device IDE0, slice 0. So, to boot the disk that you just setup with SYSCOPY, you would choose option 2. You will then be prompted for the slice on IDE0 that you want to boot. For now, just press enter to choose slice 0. Once you are familiar with slices, you can SYSCOPY and boot alternate slices. Here is what you would see when booting to a disk device:

```
MARK IV Boot Loader

ROM: (M)onitor (C)P/M (Z)-System (F)orth (B)ASIC (T)-BASIC (P)LAY (U)SER ROM
Disk: (0)MD1 (1)MD0 (2)IDE0 (3)IDE1

Boot Selection? 2    Slice(0-9)[0]?

Booting Disk Unit 2, Slice 0...

Reading disk information...
Loc=D000 End=FE00 Ent=E600 Label=Unlabeled Drive

Loading...
```

Following this, you would see the normal operating system startup messages. However, 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 you selected to boot.

If you receive the error message “Disk not bootable!”, you have either failed to properly run `SYSCOPY` on the target disk or you have selected the wrong disk/slice.

Note that although MD1 (RAM disk) and MD0 (ROM disk) drives are listed in the Disk boot line, they are not “bootable” disks because they have no system tracks on them. Attempting to boot to one of them, will fail with a “Disk not bootable!” error message and return to the loader prompt.

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 may be lost.

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

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

- 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 RTS 30 to avoid this issue.
- Z-System applications will not run under CP/M 2.2. For example, the `LDDS` date stamper with not run.

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).

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.

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.

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 CPMLDR.SYS on the system tracks of the disk.

Notes

- `ZPMLDR` is included with ZPM3, but it is not working correctly.
- The ZPM operating system is contained in the file called CPM3.SYS which is confusing, but it is the author's intended way of using ZPM3.

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

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.

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.

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.

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.

Startup Command Processing

Each 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, 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.

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.

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.

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.

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.

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.

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 upgrade. With RomWBW, you can upload a new system image executable and load it from the command line. For each ROM image file (.rom) in the Binary directory, you will also 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.

There are two restrictions to be aware of related to loading a system image as a .com application. First, this is only supported under Z-System and CP/M 2.2. You must boot into one of these OSes before attempting to launch the .com file. Second, you may find that you are unable to load the .com file because it is too large to fit in available application RAM (TPA). Your only recourse in this situation is to build a custom ROM with fewer features.

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.img

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.img
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.

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 tracks of the disk. This is done using a SYSCOPY command such as `SYSCOPY C:=B:ZSYS.SYS`. For a ZSDOS boot disk, use ZSYS.SYS. For a CP/M 2.2 disk, use CPM.SYS. For a CP/M 3 or ZPM3 disk, use CPMLDR.SYS. CPMLDR.SYS is not provided on the ROM disk, so you will need to upload it from the distribution.

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

RomWBW Distribution

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

RomWBW GitHub Repository

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:

Application	Description
Binary	The final output files of the build process are placed here. Most importantly, are the ROM images with the file names ending in ".rom".
Doc	Contains various detailed documentation including the operating systems, RomWBW architecture, etc.

Application	Description
Source	Contains the source code files used to build the software and ROM images.
Tools	Contains the MS Windows programs that are used by the build process or that may be useful in setting up your system.

Acknowledgments

While I have heavily modified much of the code, I want to acknowledge that much of the work is derived from the work of others in the RetroBrew Computers Community including Andrew Lynch, Dan Werner, Max Scane, David Giles, John Coffman, and probably many others I am not clearly aware of (let me know if I omitted someone!).

I especially want to credit Douglas Goodall for contributing code, time, testing, and advice. He created an entire suite of application programs to enhance the use of RomWBW. However, he is looking for someone to continue the maintenance of these applications and they have become unusable due to changes within RomWBW. As of RomWBW 2.6, these applications are no longer provided.

- David Giles contributed support for the CSIO support in the SD Card driver.
- Ed Brindley contributed some of the code that supports the RC2014 platform.
- Phil Summers contributed Forth and BASIC in ROM as well as a long list of general code enhancements.
- Phillip Stevens contributed support for FreeRTOS.
- Curt Mayer contributed the Linux / MacOS build process.
- UNA BIOS and FDISK80 is a product of John Coffman.
- FLASH4 is a product of Will Sowerbutts.

Contributions of all kinds to RomWBW are very welcome.

Getting Assistance

The best way to get assistance with RomWBW or any aspect of the RetroBrew Computers projects is via 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.