# RomWBW Getting Started

Wayne Warthen

March 17, 2020

## Contents

# RomWBW

## Z80/Z180 System Software

Version 2.9.2 of March 17, 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 providing 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 (not pre-release) 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 called Source Code 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 Prop-IO/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:

| Platform | ROM Image File | Baud | Description |
| --- | --- | --- | --- |
| SBC V1/V2 | 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 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 |
| 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 thrid-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 1MB ROM, 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 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 typical 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.

# 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 keep 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 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. You may find that you are unable to load the .com file because it is too large to fit in available application RAM (TPA). Unfortunately, in this case, you will not be able to use the .com file to start your system.

If you do not have easy access to a ROM programmer, it is entirely 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. The ROM image will be 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:

```
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 use a disk or 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 would 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
- FORMAT.COM
- OSLDR.COM
- SYSCOPY.COM
- TALK.COM
- FDU.COM (was FDTST.COM)
- XM.COM
- MODE.COM
- RTC.COM

- TIMER.COM
- INTTEST.COM

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

# Using Disks

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.

If your system has working disk devices, then you should notice some drive letters assigned at startup. The specific drive letters assigned will depend on your system configuration. Note that there **must** be media installed in IDE, CF, SD interfaces in order for drive letters to be assigned at boot.

To use a disk device, you will need to initialize the directory of the filesystem. This 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:`.

If you are using a floppy drive, you will need to 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.

## 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 on the directory of the drive letter(s).

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 bootable disk |
| fd_zsdos.img | hd_zsdos.img | ZSDOS 1.1 bootable disk |
| fd_nzcom.img | hd_nzcom.img | NZCOM bootable disk |
| fd_cpm3 | hd_cpm3.img | DRI CP/M 3 bootable disk |
| fd_zpm3 | hd_zpm3.img | ZPM3 bootable 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 (see below for information on disk 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 bootable disk |
| Slice 1 | ZSDOS 1.1 bootable disk |
| Slice 2 | NZCOM bootable disk |
| Slice 3 | DRI CP/M 3 bootable disk |
| Slice 4 | ZPM3 bootable disk |

| Slice | Description |
| --- | --- |
| 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 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.

# 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 OSes.

## ROM Disk

In addition to the ROM-based operating systems and applications, the ROM also contains a ROM disk with a small CP/M filesystem. The contents have been optimized to provide a core set of tools and applications that are helpful for either CP/M 2.2 and 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.

## 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 useable space on a single media. You can think of slices as a way to refer to 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 provided to allow 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 :n where n 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 message meaning the first IDE device. To refer to the second slice of IDE0, you would type "IDE0:1". So, if I wanted to use drive letter L: to refer to the second slice of IDE0, I could use the command `ASSIGN L:=IDE0:1`.

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

# Inbuilt ROM Applications

In addition to CP/M 2.2 and Z-System, there are several additional 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 options 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.

# 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 RomWBW 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. |
| FDU | Format and test floppy disks. Menu driven interface. |

| Application | Description |
| --- | --- |
| OSLDR | Load a new OS on the fly. For example, you can switch to Z-System when running CP/M. Use OSLDR with no parms for usage instructions. |
| FORMAT | Will someday be a command line tool to format floppy disks. Currently does nothing! |
| MODE | Reconfigures serial ports dynamically. |
| XM | XModem file transfer program adapted to hardware. Automatically uses primary serial port on system. |
| FDISK80 | John Coffman's Z80 hard disk partitioning tool. See documentation in Doc directory. |
| FAT | Access MS-DOS FAT filesystems from RomWBW (based on FatFs). |
| FLASH | Will Sowerbutts' in-situ ROM programming utility. |
| CLRDIR | Format the directory areas of a CP/M disk. |

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). |

There is additional documentation on some of these applications at the RomWBW Applications Page.

# 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 as result, a formatted filesystem will be accessible to any operating

system. The only possible issue is that if you turn of 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.

## 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").

## 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.

To use, NZCOM, you must run through a simple configuration process. This is well documented in the NZCOM manual in the "NZCOM Users Manual.pdf" file in the RomWBW Doc directory. Additionally, there are instructions for automatically launching NZCOM when the disk is booted under the host OS via an auto command submission process.

### 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 better 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.

### 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 CPM3.SYS on the system tracks of the disk.

### FreeRTOS

Note that Phillip Stevens has also ported FreeRTOS to run under RomWBW. FreeRTOS is not provided in the RomWBW distribution, but is available from Phillip.

# 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 image. 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.

# RomWBW Distribution

## 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. |
| 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. |

## Source Code Respository

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

RomWBW GitHub Repository

# Acknowledgements

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.
- Curt Mayer contributed the Linux / MacOS build process.
- UNA BIOS is a product of John Coffman.

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.