# Building a Custom ROM

At present, the build environment assumes you are running a current version of Microsoft Windows (either 32-bit or 64-bit). Additionally, you will need Microsoft PowerShell. PowerShell is included in all distributions of Microsoft Windows starting with Vista. It is available as a free download for Windows XP from the Microsoft website.

Other than PowerShell, all required tools are included in the distribution. You should not need anything other than what comes as part of Windows or as part of the distribution.

## Overview

Creating a custom ROM involves running a scripted “build” of the source code. There is a configuration file that controls the options for the build. A set of generic configuration files are supplied that can be used as examples. The same generic configuration files are the ones used to produce the pre-build ROM images distributed with RomWBW.

There are many options and combinations allowed by RomWBW. The generic configurations should work well to get you started, but ultimately you will probably want to build your own, custom ROM to optimize your system for your specific hardware and preferences.

I strongly recommend that you initially try an appropriate pre-built ROM image from the distribution (refer to RomList.txt) before trying to build a custom ROM. This will ensure that your hardware is working properly.

## Caveats

There is an almost infinite combination of configuration options, not to mention the possible hardware variations. It is very easy to create configurations that will not build or, even worse, ones that build OK, but do not work. If something does not work, there is little recourse but to dive into the code and figure it out by adding your own diagnostics.

Any issues with the generic configurations or pre-built ROMs will be supported by me on a best effort basis. I am happy to try and help with custom builds, but cannot guarantee any specific level of support.

## Summary

This is a brief summary of the steps you will need to complete. These steps are described in greater detail in the sections that follow.

The following steps are required to prepare your build environment. Once the steps above have been completed, you do not need to repeat them.

1. Download the RomWBW distribution archive (.zip file)
2. Extract the distribution files
3. Configure PowerShell to allow running unsigned scripts

The following steps apply to each cycle of changing options/code and creating a new ROM.

1. Create/update configuration file
2. Update/Add/Delete any files you want incorporated in the ROM Disk
3. Run the build scripts and confirm there are no errors.
4. Burn the resultant ROM image and try it.

You can (and should) skip steps 1 & 2 until you are able to run steps 3 & 4 smoothly. This will ensure you are able to successfully create new ROM images before you start changing the configuration or code.

## Preparing the Build Environment

### Acquire the Distribution Package

The primary source of the distribution package (.zip file) is the N8VEM Wiki. The Wiki home page is found at <http://n8vem-sbc.pbworks.com/>. The RomWBW Project directory is found at <http://n8vem-sbc.pbworks.com/w/browse/#view=ViewFolder&param=RomWBW>. The stable release will normally be called RomWBW.zip. You may see other variants out there when prereleases are being tested, but RomWBW.zip would be the most stable version available.

Simply download the zip file using your browser.

### Extract the Distribution

The distribution zip file can be extracted using any of the commonly available zip file processing tools. Newer versions of Windows will allow you to right-click on it and “Extract All…”. Note that the zip file contains multiple levels of subdirectories, so your extraction tool must preserve the directory structure. The extracted files do not need to be placed in any special location as long as the directory structure under the main directory is maintained.

### Configure PowerShell Execution Policy

By default, PowerShell is configured to not run any scripts which are not signed. The RomWBW build scripts are not signed, so you will need to configure PowerShell to run unsigned scripts. Specifically, you need to change the PowerShell ExecutionPolicy to “Unrestricted”. This may be considered a security risk, so you will need to decide if the risk is something you do not want to incur.

A command file is provided to update the PowerShell configuration. This will change the Execution-Policy to Unrestricted. The command file **must**  be run with Administrator permissions. To perform this operation, right-click on “FixPowerShell.cmd” in the main directory of the distribution and select “Run as administrator”. You may be prompted with a “User Account Control” dialog and will need to select “Yes” in order to continue. When it proceeds, the script will display the current setting and ask you if you want to proceed with changing the Execution-Policy to Unrestricted. Press ‘Y’ to complete the change. Below is an example of this the screen output from this procedure.

By default, PowerShell is configured to block the  
execution of unsigned scripts on your local system.  
This command file will attempt to modify your  
PowerShell ExecutionPolicy to "Unrestricted"  
which means that local scripts can be run without  
being signed. This is required to use the RomWBW  
build process.

Your PowerShell ExecutionPolicy is currently set to: ' Restricted '

In order to modify the ExecutionPolicy, this command  
file \*MUST\* be run with administrator privileges.  
Generally, this means you want to right-click the  
command file called FixPowerShell.cmd and choose  
"Run as Administrator". If you attempt to continue  
without administrator privileges, the modification  
will fail with an error message, but no harm is done.

Do you want to proceed [Y,N]?Y

Attempting to change Execution Policy...

Your new PowerShell ExecutionPolicy is now set to: ' Unrestricted '

Press any key to continue . . .

## Build Process

### Create/Update Configuration File

The settings for a build are primarily controlled by a configuration file that is included in the build process. In order to customize your settings, you need to modify an existing configuration file or create your own.

If you look in the Source directory, you will see a series of files named config\_xxxx\_yyyy.asm. Each of them corresponds to one of the standard configurations listed in the ROMList.txt file.

You have two choices. You can simply modify the existing configuration file that is closest to your situation, or you can copy it to a new config\_xxxx\_yyyy.asm file and modify that. I recommend that you copy one to your own name so that you will always have the unmodified standard configuration files left in place. So, for example, you could just copy config\_ZETA\_std.asm to config\_ZETA\_wayne.asm. You MUST name your config file as config\_xxxx\_yyyy.asm. The xxxx's must match your platform (N8VEM, ZETA, N8, S2I, or S100). The yyyy's can be whatever you want.

The config files are simply text files with various settings. Open your target config file with your favorite text editor and modify the settings as desired.

Unfortunately, I have not yet documented each of the settings in detail; that will be a separate document provided in the future. However, there are comments in the config file that will probably be sufficient for the most part.

### Prepare ROM Disk Contents

The files that are included on the ROM Disk of your ROM are copied from a set of directories during the build process. This allows you to have complete flexibility over the files you want included in your ROM.

If you look at the RomDsk directory, you will see a variety of subdirectories. These subdirectories contain the files that will be included in the ROM disk. The build process will determine which subdirectories to include files from based on the following rules:

First, all files from either std\_512 or std\_1024 will be included depending on the size of the ROM you are building. If you are building a 512KB ROM, then all the files from std\_512KB will be included. If you are building a 1MB ROM, then all the files from std\_1024KB will be included. Essentially, the files in std\_1204KB are a superset of the ones in std\_512KB because there is more space available for the ROM drive.

Second, all files from the directory that corresponds to your configuration file will be included. If you build the "ZETA\_std" configuration, all files in cfg\_ZETA\_std will be added. Note that these files will be in addition to the files from the std\_XXXKB directory.

If you created your own config file (like config\_ZETA\_wayne.asm described above), you MUST create a subdirectory within the RomDsk directory and populate it with the files you want added. Normally, you would include the files from the original standard config. So, if you created config\_ZETA\_wayne.asm from config\_ZETA\_std.asm, then you would create a subdirectory in RomDsk called cfg\_ZETA\_wayne and copy all the files from cfg\_ZETA\_std to cfg\_ZETA\_wayne.

### Run Build Scripts

The build involves running commands at the command prompt. From a Command Prompt window, you will need to change to the high level directory for the build. Normally, you would be changing to the RomWBW directory unless you renamed it.

First, you will need to build the components that are common to all configurations. These components do not require any configuration. To build these, run the “BuildCommon” command and ensure there are no errors.

To run the main build and be prompted for required information, just enter "Build". You will be prompted for the information described below and the build should run. If an error is encountered, the build should stop and display an error in red text.

If you immediately receive the error "the execution of scripts is disabled on this system", then you have not successfully changed the PowerShell Execution-Polcy to "Unrestricted". Refer to the “Configure PowerShell Execution Policy” section above to correct this.

The build script will prompt you for the following information which you will need to provide (don't worry, it is simple):

|  |  |
| --- | --- |
| Platform | Respond with the name of the platform that you are targeting. It must be one of N8VEM, ZETA, N8, MK4, S2I, or S100. |
| Configuration | Respond with the name of the configuration you wish to build. A list of all available configurations is displayed for your convenience. For example, if you are building the provided ZETA\_std configuration, just enter "std". If you have created a custom configuration as described above, you would enter "wayne". |
| ROM Size | Respond with either "512" for a 512KB ROM build or "1024" for a 1MB ROM build. Only the two choices are possible at this time. It is important that you choose a ROM size that is no larger than the szie of the ROM you will ultimately be burning. This is dependant on your hardware. |

If you don't want to be prompted for the options to the "Build" command, you can specify the options right on the command line.

For example:

Build ZETA std 512

In this case, you will not be prompted. This is useful if you wish to automate your build process.

At this point, the build should run and you will see output related to the assembler runs and some utility invocations. Just review the output for any obvious errors. Most errors will cause the build to stop immediately and display an error message in red. However, there are situations where an error will occur and the build will not halt. Review the output of the build carefully to ensure no errors are reported.

### Program ROM Image

If you look in the Output directory. You should find the following files:

|  |  |
| --- | --- |
| <platform>\_<config>.rom | Binary ROM image to burn to EEPROM |
| <platform>\_<config>\_CPM.sys | CP/M 2.2 system image that can be written to the start of a disk to enable boot from disk functionality |
| <platform>\_<config>\_ZSYS.sys | ZSystem system image that can be written to the start of a disk to enable boot from disk functionality |
| <platform>\_<config>.bio | A BIOS image that can be loaded and tested without actually burning a new ROM – see Advanced Topics below |

The actual ROM image is the file ending in .rom. It should be exactly 512KB or 1MB depending on the ROM size you chose. Simply burn the .rom image to your PROM and install it in your hardware. The process of burning the image to your PROM and installing in your system will depend on your PROM Programming hardware and the specific PROM chip you are using. Refer to the documentation for your PROM Programmer for help with this.

Specifying Build Options on Command Line

----------------------------------------

If you don't want to be prompted for the options to the "Build"

command, you can specify the options right on the command line.

For example:

Build ZETA std 512

In this case, you will not be prompted. This is useful if you

wish to automate your build process.

Example Build Run

-----------------

C:\Users\WWarthen\Projects\N8VEM\Build\RomWBW>Build.cmd

Platform [N8VEM|ZETA|N8|S2I|S100]: ZETA

Configurations available:

> ppp

> std

Configuration: std

ROM Size [512|1024]: 512

Building ZETA\_std: 512KB ROM configuration std for Z80...

tasm -t80 -g3 ccpb03.asm cp.bin

TASM Z80 Assembler. Version 3.2 September, 2001.

Copyright (C) 2001 Squak Valley Software

tasm: pass 1 complete.

tasm: pass 2 complete.

tasm: Number of errors = 0

tasm -t80 -g3 bdosb01.asm dos.bin

TASM Z80 Assembler. Version 3.2 September, 2001.

Copyright (C) 2001 Squak Valley Software

tasm: pass 1 complete.

tasm: pass 2 complete.

tasm: Number of errors = 0

tasm -t80 -g3 syscfg.asm syscfg.bin

TASM Z80 Assembler. Version 3.2 September, 2001.

Copyright (C) 2001 Squak Valley Software

tasm: pass 1 complete.

Configuration: ZETA Z80 SBC, FLOPPY (AUTOSIZE), PPIDE (STD)

tasm: pass 2 complete.

tasm: Number of errors = 0

tasm -t80 -g3 -dBLD\_SYS=SYS\_CPM cbios.asm cbios.bin

TASM Z80 Assembler. Version 3.2 September, 2001.

Copyright (C) 2001 Squak Valley Software

tasm: pass 1 complete.

Configuration: ZETA Z80 SBC, FLOPPY (AUTOSIZE), PPIDE (STD)

INFOLIST occupies 18 bytes.

UTIL occupies 484 bytes.

FD\_DATA occupies 340 bytes.

PPIDE\_DATA occupies 1116 bytes.

CBIOS space remaining: 2092 bytes.

tasm: pass 2 complete.

tasm: Number of errors = 0

tasm -t80 -g3 dbgmon.asm dbgmon.bin

TASM Z80 Assembler. Version 3.2 September, 2001.

Copyright (C) 2001 Squak Valley Software

tasm: pass 1 complete.

Configuration: ZETA Z80 SBC, FLOPPY (AUTOSIZE), PPIDE (STD)

DBGMON space remaining: 795 bytes.

tasm: pass 2 complete.

tasm: Number of errors = 0

tasm -t80 -g3 prefix.asm prefix.bin

TASM Z80 Assembler. Version 3.2 September, 2001.

Copyright (C) 2001 Squak Valley Software

tasm: pass 1 complete.

Configuration: ZETA Z80 SBC, FLOPPY (AUTOSIZE), PPIDE (STD)

tasm: pass 2 complete.

tasm: Number of errors = 0

tasm -t80 -g3 bootrom.asm bootrom.bin

TASM Z80 Assembler. Version 3.2 September, 2001.

Copyright (C) 2001 Squak Valley Software

tasm: pass 1 complete.

Configuration: ZETA Z80 SBC, FLOPPY (AUTOSIZE), PPIDE (STD)

tasm: pass 2 complete.

tasm: Number of errors = 0

tasm -t80 -g3 bootapp.asm bootapp.bin

TASM Z80 Assembler. Version 3.2 September, 2001.

Copyright (C) 2001 Squak Valley Software

tasm: pass 1 complete.

Configuration: ZETA Z80 SBC, FLOPPY (AUTOSIZE), PPIDE (STD)

tasm: pass 2 complete.

tasm: Number of errors = 0

tasm -t80 -g3 loader.asm loader.bin

TASM Z80 Assembler. Version 3.2 September, 2001.

Copyright (C) 2001 Squak Valley Software

tasm: pass 1 complete.

Configuration: ZETA Z80 SBC, FLOPPY (AUTOSIZE), PPIDE (STD)

LOADER space remaining: 1205 bytes.

tasm: pass 2 complete.

tasm: Number of errors = 0

tasm -t80 -g3 pgzero.asm pgzero.bin

TASM Z80 Assembler. Version 3.2 September, 2001.

Copyright (C) 2001 Squak Valley Software

tasm: pass 1 complete.

tasm: pass 2 complete.

tasm: Number of errors = 0

tasm -t80 -g3 hbios.asm hbios.bin

TASM Z80 Assembler. Version 3.2 September, 2001.

Copyright (C) 2001 Squak Valley Software

tasm: pass 1 complete.

Configuration: ZETA Z80 SBC, FLOPPY (AUTOSIZE), PPIDE (STD)

UART occupies 146 bytes.

FD occupies 2071 bytes.

PPIDE occupies 809 bytes.

HBIOS space remaining: 24428 bytes.

STACK space remaining: 145 bytes.

tasm: pass 2 complete.

tasm: Number of errors = 0

tasm -t80 -g3 hbfill.asm hbfill.bin

TASM Z80 Assembler. Version 3.2 September, 2001.

Copyright (C) 2001 Squak Valley Software

tasm: pass 1 complete.

Configuration: ZETA Z80 SBC, FLOPPY (AUTOSIZE), PPIDE (STD)

tasm: pass 2 complete.

tasm: Number of errors = 0

tasm -t80 -g3 romfill.asm romfill.bin

TASM Z80 Assembler. Version 3.2 September, 2001.

Copyright (C) 2001 Squak Valley Software

tasm: pass 1 complete.

tasm: pass 2 complete.

tasm: Number of errors = 0

Building ZETA\_std output files...

Building 512KB ZETA\_std ROM disk data file...

C:\Users\WWarthen\Projects\N8VEM\Build\RomWBW>