HandyROM Program Specification
27 Feb 1989
Last Modified: 19 April 1989
- RJ Mical

Confidential and Proprietary

Get HR000 and HR0001 notes in here.

* means the current ROM address.

The HandyROM program merges various types of binary files into one large binary file, the image of the Handy ROM. HandyROM optionally allows you the ability to create a directory structure at the beginning of the ROM, which structure contains information describing where in the ROM the data of the files can be found.

HandyROM accepts as input arguments found in a *HandyROM specifications file*. This file contains directives to HandyROM regarding the type of ROM that should be built and the files that should be included in the ROM.

The command-line arguments:
-v causes verbose diagnostics to be printed to the CLI
Any other tokens in the command-line will be regarded as the names of specifications files, which files will be processed sequentially in the order that they appear in the command line.

The output from the HandyROM program will go by default to a .rom file using the name of the first specifications file in the command line according to the following rules:

- if the name has a dot extension (if it ends with, say, .src, .txt, .anything) then the dot extension will be stripped off
- the extension .rom then will be appended to the name, and this will be used as the output file name

Note that an alternate output name can be specified using the OUT directive described below.

The Handy game ROM is comprised of 256 pages, and each page is comprised of a fixed number of bytes. The size of the ROM is declared in the specifications file. The number of bytes per page is derived by dividing ROM size by 256.

ROM Directory structure

There can be up to 256 files in a ROM, and therefore up to 256 file entries. You won't known until you've gone through all the specification files how many file entries there actually.

The directory entry contains some or all of these components:

Where the file is in the ROM (options: page and byte offset or just page). The page number is always 8 bits. The byte offset is as many bits as necessary
Flag byte (optional, included if any of the FLAG directives is given)
RAM destination (optional)
Byte size of this file (optional)

HandyROM Directives

These occur before the first FILE directive:

FILEALIGN
When this directive is used, the first byte of all files should be aligned to an even multiple of this value *and* the insignificant bits should be discarded when creating the directory entry describing the location the file in the ROM. This will allow using fewer bytes per entry. For example, a 128K ROM would normally require 17 bits to describe the page and offset of a given file, but if FILEALIGN 2 is used then all files will be aligned on even byte boundaries and only 16 bits are required per directory entry.

FILEDEST
This directive is used to specify that you want RAM destination to be included in the directory entry. If you want this, the FILEDEST directive must be given before the first file is included. Also see RAMDEST for setting the RAMDEST for a given file.

FILEPAGE
The FILEPAGE directive is used to specify that you want directory entries with positions using page offsets only (the default is both page and byte offsets). If you want page offsets only, this argument must be given before the first file is included.

FILESIZE
With this FILESIZE directive, you specify that you want the size (in bytes) of a file to be part of the file's directory entry. If you want FILESIZE, this directive must be given before the first file is included.

NODIR
If this directive is used, no directory structure will be built into the ROM. If you don't want a directory, this argument must be given before the first file is included.

OUT pathname
If this directive is given, the output will be sent to the named file rather than to the default file. If you're going to use this directive it should be used before most other directives, should probably be one of the first directives in the specifications file.

PACKFLAG n
If this directive is given, then each directory entry will include a flag byte which byte will have the PACKFLAG set if the file has been designated as a PACKED file. The argument to PACKFLAG should be a value wherein only one bit is set. If PACKFLAG is going to be specified, it must be given before the first file is included.

ROMSIZE n
You use this to define the total size of the ROM. You must specify a ROMSIZE before the first file is included.

These occur as of or after the first FILE directive:

ALIGN n
This directive causes the ROM address to be aligned to the next multiple of the specified argument. This works identically to the assembler's .ALIGN directive.

BIN pathname
This directive causes a normal assembler output file (.bin) to be included in the ROM. This data will the next sequential data of the current ROM file.

FILE <file number>

Create a new directory entry for the next file. If a numeric argument is provided, then the number defines a file number. If no argument is given, the file gets the next sequential file number (starting from zero).

ORG page,offset
This directive causes the next data to go to the specified ROM page and offset.

PACKED
If you have specified PACKFLAG then you can specify whether a given file is packed by specifying PACKED immediately prior to the inclusion of the file.

RAMDEST n
You can specify where a particular file should be loaded in RAM. This works in conjunction with the FILEDEST directive, with which you declare that directory entries will have RAM destination information. You must use RAMDEST to declare the RAM destination for a raw binary file. You may use RAMDEST to change the RAM destination of an assembler output file.

RAW pathname
This directive causes a raw binary file to be included in the ROM. This data will the next sequential data of the current ROM file.

Diagnostics

Columnated filenumber, offset in ROM, size, RAM destination, etc.
Unused data space in ROM
Size of ROM

Output

HandyROM outputs a file with a binary ROM image.

- 512 bytes of zero
- ROM Directory, which is (Directory entry count) * (Directory entry size) bytes long
- ROM data

Optional 6502 assembly text output of directory structure. This includes a constant describing the byte size of each directory entry:

```
DIR_ENTRY_SIZE     .EQU  number of bytes in each directory entry
```

And then, for each directory entry, a comment describing the file number and then the directory entry data declared as .BYTE's and .WORD's as appropriate:

```
; File 0
        .BYTE and .WORD declarations
```

Optional 6502 assembly text output of directory structure, same as above, except that directory entry numbers will be output in the form of equates. This allows the information to be used in the program without requiring a data table to be built.

Implementation Notes

When building the ROM image, leave the first 512 bytes blank.

For each included assembler output (.bin) file:

- Clear a 64K buffer
- Read in each data packet, recording the lowest and highest data block seen
- Create ROM data for all data between the lowest and highest data inclusive

A line that starts with ';' (semi-colon) is comment that should be disregarded.

If a number starts with '$' (dollar sign) it's a hex value, otherwise it's a decimal value.