Well, that's it! Load the file HP.ASM, and press Apple-A to assemble the file. When the listing of the source file begins, things will go faster if you press Control-D to turn off the screen display while the program assembles. (You could also just add a LST OFF to the beginning of HP.ASM).

This assembly should proceed with no errors. Version 4.0 of HodgePodge with the changes indicated here should assemble to a total length of 8627 bytes. If you do get any errors, note the lines that they occur on, and check the corrections to the files made in these instructions. You will probably have to load HP.ASM to check the line the put file was used on to identify the actual file the error occurred in.

The final step to generate the system file is to do the linking process by first loading the file HP.ASM. Then go to the Command Box with Apple-O, and type NEW to clear the editor. Then type LINK and press Return. Merlin will assemble the HP.ASM file again, and automatically generate the ProDOS 16 system file named HP. This can then be launched from the Finder, DeskTop, or even the Main Menu of Merlin 16 if you have already booted on a ProDOS 16 startup disk. When testing the HP file, be sure that you have all the necessary tools and driver files on your startup disk. If you are unsure, just startup on your Apple IIGS System disk before launching HP.

In testing the HP program, you may notice that the "About HodgePodge" window has the text "&SysDate" in it. Merlin 16 does not currently have a system date pseudo-op, but you can edit line 257 in DIALOG.ASM to use a fixed date if you wish.

## Final Notes...

Although these instructions may seem fairly long, when you have gone through the conversion process once, you'll find that converting any file goes very quickly. For example, using a RAM disk with the folders HP.ASM and HP.MERLIN as described in this document, all the conversion steps described here can be done in about 15 minutes, including test assemblies, use of the FIND utility, etc. Other interesting facts are that the Merlin source files take up only 75% of the space of the APW version (not counting the Equates and Macros files), and that Merlin can assemble, link and save the final HP system file on a RAM disk in just 56 seconds, compared to almost 3 minutes (2:50) for the APW assembler to do the same thing.

# Merlin 8/16 Supplement

This documentation describes a number of additional utilities and program files on your Merlin 8/16 disk that we think you will be interested in.

## New Merlin 16 Assembler Features:

**Easier-Reading Binary Numbers:** Merlin 16 now lets you break up binary numbers in an operand for easier reading. AND #%0101_1101 instead of AND #%01011111, for example.

**Longer Operand/Comment Lines:** The maximum size for the operand in Merlin 16 has been increased from 64 to 80 characters. This means fewer problems when converting comments from APW, and also gives you more room for your own operands and comments.

**Improved MON command:** When using the MON command to test a routine you've just assembled, the MON command in Merlin 16 now properly sets the softswitches so that the "Go" command can be used (i.e., 8000G).

**Merlin.Sys16** This is a ProDOS 16 Merlin startup program that makes testing ProDOS 16 programs directly from Merlin 16 easier. Instead of launching Merlin.System from your program launcher, startup Merlin.Sys16 instead. This program must be in the same directory as Merlin.System. When Merlin.Sys16 is started, it puts itself in the quit stack and runs Merlin.System.

To test a ProDOS 16 system file after assembly and linking, just type in -FILENAME as a Disk Command from the Main Menu of Merlin 16. When your program quits, it will automatically re-start Merlin.Sys16 which in turn will re-run Merlin.system. When you want to quit Merlin 16 back to your program selector such as the Finder, just hold down the Option key when you quit Merlin 16, and continue to hold it down while Merlin.Sys16 returns you to the Finder.

For Merlin.Sys16 to work, you *must* startup through ProDOS 16, such as with the Apple System disk. Presumably you are launching Merlin.Sys16 with the Finder, DeskTop, Program Launcher, or some other ProDOS 16 program selector. (ProSel can also be used as long as you have started up using ProDOS 16)

**Linker** The Merlin 16 Absolute Linker has four new commands.

**FASt:** *Doubles* link speed of Linker in those files that have only one SAV.

**LEN LABEL:** Puts "LABEL" in symbol dictionary as an ENTry whose value is equal to the number of bytes of the *last* linked file.

**POS LABEL:** Similar to LEN, but sets LABEL equal to the number of bytes in the linked file from the start (or from last zero-point; see below).

**POS:** When used by itself with no label, POS resets the byte counter to zero for the next POS LABEL command.

The upshot of these last three commands is that the Linker can communicate the length of various linked files with later modules in the link process . This would be useful, for example, if the size of one module in a program had to be used by a later module.

Examples:
#1:

```
    ORG $2000      ; ProDOS 8 System file
    FAS            ; Fast linker mode
    ASM FILE       ; Assemble FILE.S
    LNK FILE.L     ; Link output file
    TYP $FF        ; System (SYS) filetype
    SAV FILE       ; Save object file
```

#2:

```
    LNK FILE1
    LNK FILE2      ; Link 1st & 2nd file
    LEN LABEL1     ; Set LABEL1 equal to length of FILE2
                   ; i.e., last linked file
    POS LABEL2     ; Set LABEL2 equal to length of FILE1 + FILE2
    LNK FILE3      ; Misc. file in our program...
    POS            ; No operand sets byte counter to zero.
    LNK FILE4
    LNK FILE5      ; Link files 4 & 5 now.
    POS LABEL3     ; Set LABEL3 to length of FILE4 + FILE5
    SAV PROGRAM    ; Save final object file.
```

**Find Utility,** by Sean Nolan: This file will automatically search an entire ProDOS directory of source files for any given string, and display all the source file names that string is found in, including the actual line of source code in each file. FIND can be set up to be run from within Merlin 16, or as a stand-alone utility, and can be set to be case-sensitive or not. If you're working with a number of source files and want to know which file had the PRINT routine in it, you can search for "PRINT", and FIND will display the filename and the line of source code, for example:

```
FILENAME.S    872:    JSR PRINT ;  Print something
PARTB.S       984:    PRINT LDA CHR ; Get character to print
```

FIND is used by moving the file FIND to the directory with the source files to be searched. This is most easily done from within Merlin 16 by actually loading the FIND.S source file, setting the prefix to the target directory, and then re-assembling the source file. The object code is then saved in the directory you will be working in. When you assemble the file, you will be given the option of a stand-alone version vs. a Merlin version, and also whether the utility should be case-dependent in its search. For Merlin use, the final file is used by just typing -FIND as a Disk Command from the Main Menu. When prompted, enter the search string.

If used as a stand-alone system file, FIND (FIND.SYSTEM) will ask you what pathname you wish to use for the directory to search. In this mode FIND need not be in the same directory as the files to be searched. The system-file version of FIND will search APW source (filetype SRC = $B0) files as well as TXT type files.

**BIN.LNK:** This utility converts a BIN type data file, such as a Hi-Res picture, character set, data table, etc., to a LNK type file which can be linked into Merlin 16 object files with any of the Merlin 16 Linkers. The LNK file is given one entry at the start with an entry name of your choosing. This name can then be referred to as an external by other segments. Run BIN.LNK for instructions on how to use it. The Commands directory holds some machine language routines used by the BIN.LNK utility.

**Macros,** by Dave Klimas: When the Apple IIGS first came out, and Apple distributed their standard Toolset Macros, many people wondered why they only used a minimal macro that only combined the LDX #$XXXX instruction with the JSL $E10000, when they could have created truly useful macros that also included the variables to be passed as well. The good news is that Dave Klimas has now completely edited the Apple IIGS Toolset Macros to include the ability to pass all relevant labels within the toolcall. Best of all, the new macro library is completely compatible with the original Apple macro library. That is to say, at any time, you have your choice of either the standard macro, or the new, improved "Super-Macro."

Thus, instead of using the old syntax:

```
MENU    PushLong $0000 ;  (space for result)
        PushLong BLKSZ ;  Size of Memory Block
        PushWord ID    ;  Our USER ID
        PushWord ATTR  ;  Attributes for memory block
        PushLong PTR   ;  Low Word of pointer to memory block
        _NewHandle     ;  Do NewHandle Tool Call
        PullWord HDL   ;  Store new Handle
```

You can now use:

```
MENU    ~NewHandle BLKSZ;ID;ATTR;PTR;HDL
```

What a difference, and time-saver! Replace all of the old definition files on your Merlin 16 disk with these new ones. The files are 100% compatible with the old files, and if you don't use the new form, Macgen only puts the part of the definition you use in the macro file for a particular program.

To use the new form of a macro, you'll need to look at your Toolbox Reference books, which you'd need to do anyway to use the old form too. By looking at the chart of what is pushed on the stack before the call, and retrieved after the call, you can tell what variables to include in the new macro call. The new macros automatically push any needed space on a stack for a call. This may seem a little brief in the way of explanation, but if you look at a few of the defined macros compared to the reference books, we think you'll see their use is quite obvious once you look at the tool-call documentation. The tilda (~) character is used in place of the underscore to signify the new form of any macro. The same names are used for the new macros as would be used for the old tool call.

In the Macros folder is also a folder called Example, that shows what the SIMP demo program (located on your Merlin 16 disk in the Sample.M16 folder) looks like using the new macros.

**NDA Examples, by Mark Larson:** This is a set of three example New Desk Accessory source files, specifically, Clock, See Prefixes, and Display Memory Status. There is also a demonstration ProDOS 16 System program that shows the minimum structure required for a P16 program that uses the menu bar and NDAs. It also shows an example use of the Standard File Operations function of the GS Toolset, which is used to select a file from any of the online volumes.

Load the file READ.ME.FIRST for a brief description of the files. Simply put, DEMO.CMD is the Linker command file that generates an example ProDOS 16 program that uses menus, the standard file operations tool, and also any desk accessories. Just type LINK "DEMO.CMD" from the Merlin 16 Editor Command Box to assemble and link the file. Demo can then be launched from Merlin 16, the Finder, or any program selector as long as you have started up through ProDOS 16. The other three command files, CLOCK.CMD, MEMORY.CMD, and PREFIX.CMD are the Linker command files for three New Desk Accessories. After assembling and linking the files, you will want to move CLOCK, MEMORY and PREFIX to the Desk.Accs folder in the System folder of your ProDOS 16 startup disk. These desk accessories should then appear in any program which has the Apple menu (such as the Demo program).

# HodgePodge:
## Converting the Source Files
## from APW to Merlin 16

This document discusses the steps required to convert the source files for the Apple IIGS demonstration file "HodgePodge" from the APW assembler format to files usable by Merlin 16. This conversion process requires that you have the Merlin 16 assembler, and the utilities MACGEN and CONVERTER from the Merlin 16 disk. The HodgePodge program is a demonstration program that is included with the book "Programmer's Introduction to the Apple IIGS" from Addison-Wesley. This program is not included on the Merlin Extras disk (there just isn't room), but these notes are provided to help those Merlin 16 users that do have the program.

Although these instructions deal specifically with the files for the program HodgePodge, they are presented with the larger objective of illustrating how to convert any APW source file to that usable by Merlin 16. In addition to the specific advantages of the Merlin 16 operating environment, conversion to Merlin 16 from APW offers two immediate advantages. First, Merlin 16 can assemble, link and save a file such as HodgePodge over two times as quickly as the APW assembler. In addition, source files for Merlin 16 take up much less space on a disk than their APW counterparts.

## Conversion Instructions

1) First, make a backup copy of the files that make up the APW Assembler HodgePodge source files. The original disk is presumably the one that came with the book "Programmer's Introduction to the Apple IIGS". Format a new 3.5" disk, and copy the folder HP.ASM and the files PIC1 through PIC4 to the new disk.

NOTE: Although not essential, there are enough file operations to be done that you will find things will go a lot faster if you use a RAM disk while you convert the files. If you have sufficient memory for a RAM disk, you can do all the following operations on your RAM disk, which will save a considerable amount of time in the conversion process. After completing the conversion process, you can then move the final folder with your converted files to a 3.5" disk.

2) Now run the program Converter on the Merlin 16 disk. This can be launched from the Finder, ProSel, or any other program launcher. You can alternatively go to Applesoft BASIC, set the prefix to /MERLIN.16/UTILITIES, and then type RUN CONVERTER. Press Return at the title screen, and insert the disk with the files to be converted in a second disk drive.

If you have only one disk drive, and no RAM disk, you can move the files Converter, Merlin.Hdr, Dictionary from the /Merlin.16/Utilities directory and the file Converter.Obj from the /Merlin.16/Utilities/Conv.Source directory to the disk with the files to be converted, and run the Converter program from that disk.

3) When you run the Converter program, the first thing to do is to choose "Select File From Disk" from the Main Menu. This will display a list of all online disk volumes. Select the disk volume that has the APW files to be converted. You should then see the directory HP.ASM. Select this.

4) Now you will see a list of all the files in the HP.ASM directory. You will want to convert each filename that ends with ".ASM". To do the first conversion, choose HP.ASM.

5) The disk will come on as the file is converted, and then in a moment, the menu of files in the subdirectory will be re-displayed. Continue to choose each file that ends in ".ASM". Because there are many files, after converting the first group of files, it will be necessary to select item #0 "See Page 2 of this Menu" to see the remaining filenames available for conversion. The converted files will also show up in this menu with the suffix ".ASM.S". Do not re-convert any of these files, but instead, end your converting process with the last ".ASM" file (probably GLOBALS.ASM) that appears in the menu. Ignore the files README, MAKE, LINK, HP.MACROS, and any other files that do not end with ".ASM" in the menu.

6) When you have converted the last file, press Apple-Escape to return to the Main Menu, and then choose "Quit to Program Selector" to return to your program selector. If you are not using a program selector such as the Finder or ProSel, you can choose to exit to Applesoft BASIC.

7) Now that the files are converted, you can delete all the files in the directory HP.ASM that do not end in the suffix ".S", that is, all those files that are not Merlin 16 source files. Alternatively, you can create a new subdirectory named HP.MERLIN, and copy all those source files ending in the ".S" suffix to that new subdirectory. There should be 10 Merlin files at this point.

8) As will be seen shortly, the HodgePodge program also requires three "equates" files, specifically E16.Window, E16.Dialog, and E16.Memory. These files are found on the Merlin 16 disk in the directory Tool.Equates. Copy these three files to the directory with the Merlin source files. There is also a utility program called FIND on the "Merlin Extras" disk that you should now copy to the HP.MERLIN directory. This will come in useful later.

9) Now run the Merlin 16 assembler, and from the Main Menu, use the Disk Command, and type in -UTILITIES/MACGEN to activate the Macro Generator utility. Then use the Disk Command again to set the prefix to your volume and subdirectory with the Merlin HodgePodge files.

NOTE: As mentioned in the Merlin 16 manual's discussion of APW and Merlin 16 source files, most APW source files are written without regard to case-sensitivity. That is, you often find references to the same label using different cases, for example, PushLong, pushlong, Pushlong, etc. For Merlin 16 to assemble this file, you must set the flag in the Merlin PARMS file (at approx. line #120) to set Merlin 16 to be *case insensitive*. Remember, this changed PARMS file does not take effect until you re-run Merlin.System with the new PARMS file.

10) The main source file for the HodgePodge program is the file HP.ASM(.S). Load this file into the Merlin editor. HP.ASM uses a macro file with the Apple IIGS tool calls in it. The line that references this library will have to be temporarily deactivated while Macgen scans the file. Look in the vicinity of line 96 in the file HP.ASM, and find the statement "use HP.MACROS". Put an asterisk at the beginning of the line for the moment to make this a comment. You'll also want to change the line that indicates the output link file from "dsk HP" to "dsk HP.L" (line 95).

11) For MACGEN to operate, the editor buffer must be empty, so save the file HP.ASM back to the disk now. Then return to the editor, type Apple-O to open the command box, and type NEW to clear the editor. Then type Apple-O again, and type MAC "HP.ASM".

12) As Macgen starts to read the file, you should get an error message referring to an invalid pathname "7/E16.WINDOW.S" in the file HP.ASM. APW uses a numbered pathname system, and these references will need to be edited. Press a key to return to the Main Menu of Merlin 16, and load the file HP.ASM.

13) Use the Find command (Apple-F), and search for the string 7/E16. This will display two references on lines 180, 181 (depending on your version of HodgePodge) that will need to be edited. The E16 files are "equates" files that are found on your Merlin 16 disk in the subdirectory Tool.Equates, and which you've presumably already moved to the disk with the Merlin HodgePodge files. Edit the two lines in HP.ASM to read:

```
put     E16.WINDOW
put     E16.DIALOG
```

In the normal course of events, you could also change the "7/" prefix to the prefix for the location of your own Equates library, for example on a hard disk, or on the Merlin.16 disk if you have more than one disk drive.

Although you would normally discover this next item on your second attempt at running Macgen, we'll tell you now that the source file INIT.ASM also has a "put" command using E16.MEMORY (at about line #77). Since one Merlin "put" file cannot have another nested within it, you will need to move this use of the equates file to HP.ASM. Therefore, add the line:

```
put    E16.MEMORY
```

to the HP.ASM source immediately following the other two equates-file puts. Now save HP.ASM back to the disk, and load the file INIT.ASM. Clear the line with "put E16.MEMORY" (about line #77), and save it back to the disk also.

14) Now let's try Macgen again. Type NEW in the command box again, and then type MAC "HP.ASM" to scan for macros again. This time Macgen should successfully scan all the HodgePodge files without any error messages.

15) After scanning the file, Macgen will display a short menu. Press "D" to display all the unresolved Macro names. If the labels are not all in upper case letters, then you did not set Merlin 16 to be case insensitive in the PARMS file, as discussed above, and you will need to do this, then re-run Macgen.

16) Now it is time to scan the existing macro libraries on your Merlin 16 disk to create the optimized macro definition file that will be used by HP.ASM. To start this, press "S" to "Search directory for macros". At the prompt, enter /MERLIN.16/TOOL.MACROS and press Return.

NOTE: This assumes you have copied all the macro files from the "Merlin.Extras" disk that accompanied this document to your working Merlin 16 disk!

17) Now press "D" to display any remaining unresolved Macros. There should not be any remaining macros listed. If there is, try to identify the reason, and re-scan a directory that contains the appropriate macro files. The only reason any macros should remain here (in the HodgePodge conversion) is if you do not have all the necessary macro definition files in your Tool.Macros directory on your Merlin 16 disk. On other files, there may be APW-specific assembler pseudo-ops that have not been handled by the Converter program. If there is an apparent macro call that you wish to examine in the context of the source file being converted, you can use the FIND utility, described elsewhere in this document, to search all the source files for the particular text in question.

18) At the Main Menu, save the new macro file in the directory with the source files under the name HP.MACROS.

19) Now the real fun begins! It's time to try a test assembly of the converted files, and fix any remaining problems. The first thing to do, however, is to re-activate the line in HP.ASM that loads the macro definitions. Do that now by removing the asterisk at the beginning of the line with the statement "use HP.MACROS" (probably line 96 of the converted file). When you remove the asterisk, the line formatting probably won't look right with just the asterisk gone. To tell Merlin to reformat the line, type Apple-O for the Command Box, and type FIX. There will be a pause and the line should be reformatted. You could also just re-type the line manually to have Merlin automatically tab as you enter the text.

20) Now try a test assembly. To make the listing of errors that you need to fix easier, first turn on the printer by typing PRTR1"" in the Command Box. Then press Apple-A to start the assembly. The most common errors remaining are duplicate label errors, and a few syntax differences between APW and Merlin that the Converter did not handle. As each error is encountered, the line will be printed on the printer, and the assembly will pause waiting for you to press a key. Press the space bar after each error to advance the listing.

IMPORTANT: You only need to print out the errors that occur on the first pass of the assembler, that is, those messages printed before the assembly listing itself is printed. When you see the text of the source code starting on your printer, press Control-C to stop the assembly, and remove the printed listing from your printer.

## Specific Corrections...

The following text annotates the correction of the duplicate labels and other errors in the conversion of HodgePodge version 4.0 from the APW format to the Merlin 16 format. The line numbers given should be accurate for this particular version, but may vary for other versions. (Note: the line numbers given assume that line 184 is the line with "put INIT.ASM").

The real point of this text, however, is to illustrate how to handle the various errors you may encounter in converting a program, and to to show how to correct them without changing the logic of the original program. Although the number of errors may seem large, they are almost entirely "duplicate label" errors, caused by the way APW treats local vs. global labels. In Merlin, a local label is indicated with the syntax :Label. In APW, all labels are assumed local to a particular segment ("put" file). This is fine as far as it goes, but also means there is no such thing as a local label within a particular routine itself.

The easiest way to fix most duplicate error labels is to just change them to the :Label form. In some cases, where the use of the local label is mixed with global labels, changing the name from Done to Done2, for example, may be required.

Without further delay, here then is the listing of corrections which are required to finish the conversion:

The first error is an **"Illegal char in operand"** in **line 184 >483**. The > symbol indicates that the error is on line 483 of the file actually referenced in line 184 of HP.ASM. Looking at this line, you can see it is the line with INIT.ASM. To make further work easier, you may wish to use a pen or pencil to block off each set of errors and which file is involved. For example, there should be three errors on line 185 (EVENT.ASM), four on 187 (WINDOW.ASM), etc. Writing the file names on your printed listing will make it unnecessary to return to HP.ASM as you work through the other files.

Go to the Main Menu and load the file INIT.ASM. When the editor screen appears, use the Go to Label command (Apple-L). When it prompts for a label, type 483 (the line # you want) and press Return. This should show the first data entry of a table (StartTable) that has an expresssion (EndTable-StartTable)/4. Merlin 16 uses parenthesEs differently, and in this expression, they are not needed. Use Control-D (or the Delete key) to remove the two parenthesEs. Since this is the only error in this file, return to the Main Menu and save INIT.ASM back to the disk.

The next error is a **"Duplicate symbol"** in **line 185 >61**. Load EVENT.ASM, and jump to line 61 (Apple-L). Line 61 has the label AllDone, which was used elsewhere in the program. Scrolling the cursor up a little, you can see the reference to AllDone in this routine is earlier on line 43. Change AllDone to :AllDone (a Merlin local label). You might worry at this point that there might be other references in the program to AllDone that you haven't seen right away. To search for all the likely references (that is, references in this same EVENT.ASM file), use the Find command (Apple-F) to search for any other references to AllDone. In this case, you will find there are none. Note: Because of the inconsistent use of case in many APW programs, you may have to search for variations of case in the label you are looking for. That is, you may also want to try Alldone, and alldone as well. It isn't a irrevocable error if you miss a reference, however. Your next test assembly will show up any label references that you miss.

Now use the Go to Label command to examine the next **duplicate symbol** in **line 352**. Exit is another popular label, and is referenced in the lines above on line 344. Change Exit to :Exit. Note: Use the Merlin editor commands like Control-W (jump to next word) to make editing faster. Jump to the beginning of Exit on line 344 by pressing Control-W twice. Since the insert mode is "on", typing the colon (:) inserts the character you need. Then press the Escape key to drop to the next line, and then use the down arrow to move back to the label Exit, where you insert the other needed colon. Searching for Exit will show up a reference earlier to Exit1 on line 129, the first reference to the "other" Exit on line 306, and a third one (the source of the last error message in this file)

on lines 385 and 390. Although not required, you could insert colons to change Exit to :Exit on these lines as well.

You can now save EVENT.ASM back to the disk, and load WINDOW.ASM.

The **"Duplicate symbol"** in **line 85** is another Done. The first Done appeared in one of the earlier "put" files. Using the Find command reveals the only reference to Done in this file is the label on line 85, and then later on line 88. The problem here is that we can't use a local label :Done because of the (Merlin) global label OkToOpen in between our desired local labels. The easiest solution would be to just call this Done2. However, how will you know whether Done2 is used in any of the other files? You could change the name and do a test assembly, but there is an easier way, using the FIND utility. Presuming that you followed the earlier instructions to put the FIND utility in the HP.MERLIN directory, quit to the Main Menu of Merlin now, and type D for Disk Command. At the prompt, type -FIND, and when prompted for the search string, enter Done1 and press Return. FIND will search all the source files in the HP.MERLIN directory, and report any files that use that label. In this case, you'll find that Done1 isn't used anywhere else, so you can edit lines 85 and 88 to use this new label. (When you return to the editor, just use Apple-L again to jump back to line 85).

The **"Duplicate symbol"** in **line 216** is due to a 2nd use of the label OuttaHere. The Find command reveals the branch to this label on line 182. The FIND utility find this other use in the EVENT.ASM file, and no other references other than the current one in WINDOW.ASM. Change OuttaHere to OuttaHere2 on lines 182 and 216

The **"Duplicate symbol"** in **line 438** is due to a 2nd use of the label cpynm. The Find shows the only other use of this label on lines 199 and 202. Changing cpynm to :cpynm on lines 438 and 441 fixes the conflict. Note: You may also find it easier to insert the colon using the Find command while you're also searching for other uses of a label.

The last error in this file, **"Duplicate symbol"** in **line 606** is a label for a data table, Refptr. The Find command shows this label referenced on lines 121, 123, 156, 158 in this file. For this many changes, you can also use the Merlin Exchange command (Apple-E) to change Refptr to Refptr2. Note: You don't have to use a specific replacement label as long as you use the new label in all the old places. You might, for example, want to change Refptr to WRefptr to remind you this label is in the WINDOW.ASM file. However, whenever using a new name in a multi-file source listing, it's always a good idea to use the FIND utility to make sure that label is not used in another file.

That's it for this file. Save WINDOW.ASM to the disk, and load DIALOG.ASM.

The "Duplicate symbol" in line 101 is a reference to ok, which can be changed to :ok on lines 95 and 101.

The "Duplicate symbol" in line 326 is a reference to item1, which is used in the data structure for a dialog box. Since the "1" in item1 also has some meaning in reference to the items in the dialog box, you'll probably want to change this to something like item1d (for "Dialog") on lines 323 and 326.

The "Duplicate symbol" in line 335 is a reference to Msg, which can be changed to Msgd on lines 335 and 330.

The "Duplicate symbol" in line 466 is a reference to OurAlert, which can be changed to OurAlert2 on lines 458 and 466.

The "Duplicate symbol" in line 494 is a reference to Msg, which can be changed to Msg2 on lines 458 and 466.

The "Duplicate symbol" in line 495 is a reference to StartMsg, which can be changed to StartMsg2 on lines 494 and 495.

The "Duplicate symbol" in line 501 is a reference to EndMsg, which can be changed to EndMsg2 on lines 494 and 501.

Save DIALOG.ASM and load FONT.ASM.

The "Duplicate symbol" in line 47 is a reference to LineCounter, which is a duplicate of a similar label in WINDOW.ASM (found with the FIND utility). Thus we can change all the references in FONT.ASM from LineCounter to LineCounter2 with the Exchange command.

The "Duplicate symbol" in line 313 is a reference to LineLoop, which is also used in WINDOW.ASM. The conflict is fixed by changing LineLoop to LineLoop2 on lines 313 and 341 of FONT.ASM.

Save FONT.ASM and load PRINT.ASM.

The "Duplicate symbol" in line 99 is a reference to AlreadyThere, which can be changed to the local label :AlreadyThere on line 99. A helpful note here: One thing that will help you determine whether a reference to a label in one part of an APW-generated uses a label in another part is to look for the Start and end comments left by the Converter program. Unless a Start comment is followed by a "using Label" comment where Label is the label at the beginning of the other section you're examining, all labels in that second section were considered local by APW. For example, in this case you can see that AlreadyThere is in the section SetupDefault starting on line 85 and ending on line 105. Since the other reference to AlreadyThere is in the DoSetupItem section that goes from

line 54 to line 75, and which a) has its own AlreadyThere label, and b) had only a "Using GlobalData" (not a "Using SetupDefault"), this tells you that lines 54-75 of the APW version of HodgePodge "don't know" about the label AlreadyThere in lines 85-105.

The "Duplicate symbol" in line 240 is a reference to AllDone. We can't use a local label here, but FIND reveals no other reference to AllDone2, so you can replace AllDone on lines 229 and 240 with AllDone2.

Save PRINT.ASM, and load IO.ASM.

The "Duplicate symbol" in line 102 is a reference to cont1, which is also used earlier in this same file on lines 45 and 48. We could change cont1 to cont1a, etc. but let's explore another method. Since the only thing preventing cont1 from being changed to the local label :cont1 is the use of the label Cont0 on line 98, why not make Cont0 a local label too? The only reference to Cont0 is on line 95, so this entire section can be tidied up by changing Cont0 to :cont0 on lines 95 and 98, and cont1 to :cont1 on lines 99 and 102. Note: the casual use of case in lines 95 and 98 (Cont0 vs. cont0) is why Merlin 16 has to be set to case insensitive when assembling many APW files.

The "Duplicate symbol" in line 111 is a reference to cont2, which as we've already seen, can just be added to our group of local labels by changing cont2 to :cont2.

Save the file IO.ASM.

The last thing to do involves searching for a problem caused by a bug in the APW assembler. This does not cause an error in the Merlin assembly, so it can't be found by looking at the error messages. To add insult to injury, HodgePodge actually counts on the behavior of this bug to have the program operate properly. The problem is this: the instruction LDA #' ' should load the accumulator with the ASCII value for a space. With the high-bit clear, this would be $20. In the 16-bit mode, the accumulator should be loaded with $20 in the low byte, and $00 in the high byte. The APW assembler, however, loads *both* bytes with $20, giving the value $2020 in the accumulator - definitely *not* the value for a space.

Using the FIND program to look for a LDA #' (no space or closing quote - after all, any ASCII character could be used in this expression), finds one reference on line 413 of WINDOW.ASM. The comment on this line explains that the program requires the value $20 in the high byte for a subsequent operation. Merlin supports a double-byte load in those cases where the programmer explicitly calls it out by using *two* characters as the argument of the load-immediate instruction. You can edit line 413 to have two spaces in the load argument to the LDA instruction. Then save WINDOW.ASM back to the disk.