Agilent Technologies

The Life of a 5DX Inspection C# File

The {Hashed Panel Name}.C# file is basically cad information about a panel program. It is used in conjunction with various utilities to overlay cad data on images or provide cad data for filters. With the increased use of multiple 5DX systems sending result information to a single result server for PLR processing, many confusing situations can occur. First I will outline the generation of the {Hashed Panel Name}.C# file. After that we will discuss what happens in multiple 5DX scenarios.

MIT.exe, the 5DX application program compiler coverts Neutral Data Files (NDFs) files into Run-Time Files (RTFs). These RTFs are loaded into memory when an inspection is initiated. This allows the 5DX to know how and where to inspect solder joints for a particular panel program.

During the compilation process a file called rptcad.rtf is generated. This file contains binary information about the cad data. It provides a blueprint of what the cad should look like when compared to the results of the inspection.

When a board is inspected with the 5DX, the IAS program will create the result file (. RES file) and check for a {Hashed Panel Name}.C# file. If the file is missing, such as during the first inspection, it will make a copy of the RPTCAD.RTF file and give it the hashed panel name with a C0 extension. The files are sent to the RESROOT subdirectory set by the environment variables in Windows NT

If the data for a panel program has changed because of a different number of views or a change in components, MIT.exe generates a new rptcad.rtf. When the IAS program is run again, it will go out to the RESROOT and look for the {Hashed Panel Name}.C# file, make a comparison and if needed generate a new cad information file. For Example: If the data in the rptcad.rtf file is different from than in the {Hashed Panel Name}. C0 file, the IAS will generate a {Hashed Panel Name}.C1 file. If a {Hashed Panel Name}.C1 file does not match the rptcad.rtf file, a {Hashed Panel Name}.C2 file is generated and on and on. In this way, the most current cad for a panel program is maintained for use with the Result file.

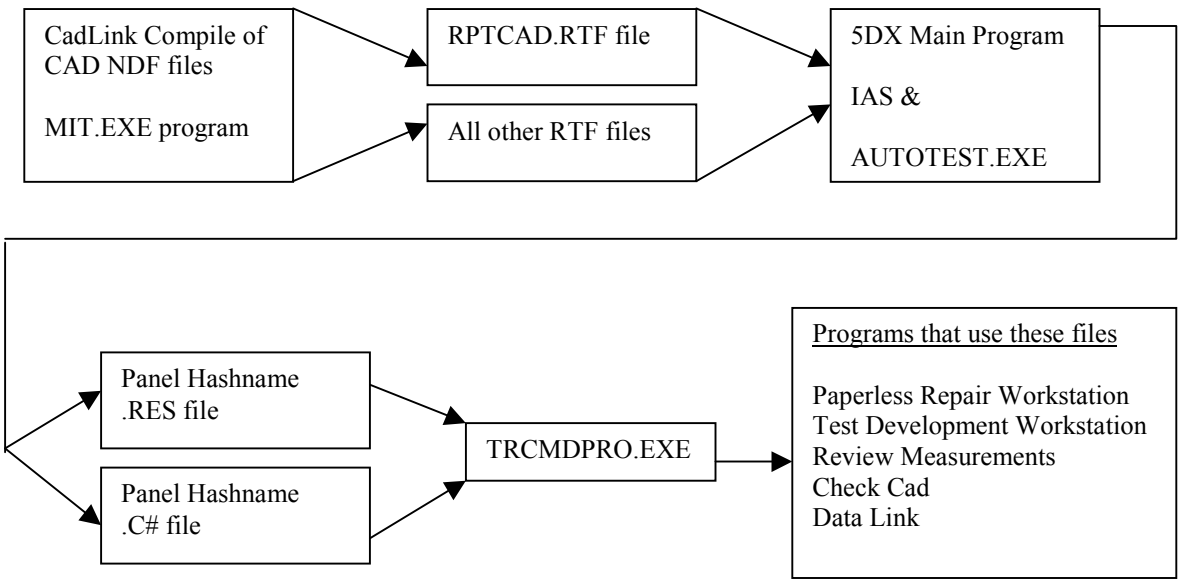

Figure 1

Where we sometimes run into trouble is in a multiple 5DX environment. Most of our customers use a central result file server for the PLR workstations. If by chance the same panel type is being inspected on two different 5DX's, there's a good chance that the C# files can cause the wrong cad to be displayed on the workstation or not to be displayed at all.

.

Example:

A customer has multiple 5DX-inspection systems and networks all of the result files to a central PLR server. It can be extremely easy to get the C# files out of sync with the result files on the server. System #501 is used to develop a panel program creating multiple C# files during the process. Once the program is complete, the program would be ported over to system #502. With further tuning on system #502, the system would generate it's own C# files (See figure 2). As you can see, one 5DX can overwrite the C# file from another 5DX with a common RES directory, creating a synchronization problem with the Result files and the C# files.

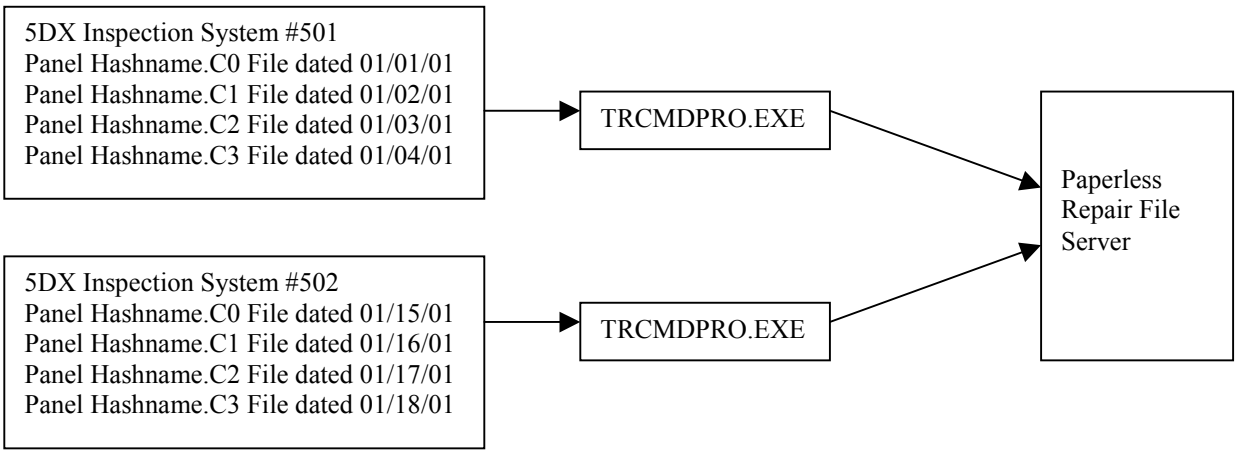

Figure 2

The only way to correct this problem is to update all 5DXs with the current panel program and delete all C# files on the 5DXs and the Paperless Repair file server.

Archiving

This brings up the problem of archiving result files. As you can see from the example above, any archive process that doesn't retain the synchronization between the result files and the C# file will render the archive completely useless. Keep this in mind when you are developing your own archiving process. Another option would be to include the RPTCAD data in the RES files. This does increase the size of each RES file but is a safer option.

October 16, 2001