Harlequin RIP™

Developing ProofReady Plugins

Technical Note Hqn043

April 2011

the smarter alternative™

GLOBAL GRAPHICS®

ProofReady plugins require Harlequin RIP™ version 5.1 or later. The guidelines given here apply to Harlequin PLUS Server RIP™ v8.0 (or later) in addition to those given in Technical Note Hqn042, *Developing color plugins for the Harlequin RIP*.

1 Introduction

This technical note is a supplement to the *Harlequin RIP Plugin Kit: Guide for OEMs* (and to a lesser extent, *Using Harlequin RIP Extensions*), describing how OEMs can create or modify output plugins to work as ProofReady-style plugins.

Note: This information is intended for OEMs only.

Conventions: This document uses terms in square brackets, such as [RB3] and [SWPLUGIN], to refer to external documents; all such terms and the full references are listed in Appendix C, "External references" on page 26.

1.1 Contents

The remainder of this document contains the following major sections:

- A summary of what a ProofReady-style plugin does. See Section 2.
- A review of what other, existing, plugins do. See Section 3 on page 3.
- A procedure for converting existing plugins to ProofReady operation. Parts of this procedure are also relevant for use in adding more profiles to an existing ProofReady plugin. See Section 4 on page 4.
- Customizing default color settings. See Section 5 on page 10.
- Protecting your investment in profile development. See Section 6 on page 11.
- A preview of expected future developments in the Harlequin RIP. See Section 7 on page 14.
- Appendixes containing material that will become part of the *Harlequin RIP Plugin Kit* manual and *Using Harlequin RIP Extensions* manuals. See Section A on page 14 and Section B on page 16 for details.

2 ProofReady plugins

The main intent of a ProofReady plugin is to provide a plugin that is both easy to use as supplied and easy for OEMs to extend and maintain.

The main implementation changes are:

- Naming profiles and color rendering dictionaries (CRDs) for the media and resolution in use.
- Extending the possible contents of Harlequin profiles.
- Storing color management files with the plugin.

These changes provide for users with varying needs for color fidelity and varying levels of technical competence. There are several cases:

- Using the plugin as supplied. This provides basic color management without the need for any technical procedures.
- Calibrating for the actual printer in use, but continuing to use the supplied color settings and profiles.
- Making custom color settings but continuing to use the supplied profiles.
- Creating and using profiles for new printing conditions, including new media types.

Each of these cases is supported and described in the *ProofReady Plugin for Epson Printers* manual.

2.1 Naming

In ProofReady plugins supplied by Global Graphics Software (GGS), both profiles and CRDs are named for the media and resolution. This is both easy for the user to understand and easy to manage from software. There are some limits on the length of name that the Harlequin RIP dialog boxes can display, as noted in the procedures that follow. (These limits are specific to the Classic Harlequin RIP; GGS expects to relax or remove these limits in other GUI products).

Note: It is not essential for a CRD to be named this way, but it is convenient for ease of tracking related files. The user sees a CRD displayed in the GUI only in terms of the rendering intent that it provides.

2.2 New content in Harlequin profiles

The new entries in Harlequin profiles can declare profiles as available for use with ColorPro, prevent profiles from being uninstalled, declare the use of CRDs held in external files, and associate an installed profile with the default calibration or golden state for which the profile was prepared.

The new entries are:

- **`%%ICC Profile`**
- **`%%Creator`**
- **`ExternalCRDs`**
- **`LinearizationName`**

For details, see Section B.4 on page 24.

2.3 Storing color management data with the plugin

It is much more convenient for installation of a plugin if all the data required by the plugin is stored in subdirectories of the plugin directory. The main change is the addition of a **`colorrenderings`** directory and the use of this directory from profiles supplied with the plugin.

See Section A.1 on page 15 for details of the new structure.

3 Existing plugins

This description is based on the methods used by GGS. If you have developed additional techniques, you may have to adapt them.

Good color output requires appropriate profiles and calibration. The most direct way to create a ProofReady pull-down menu item is to install an ICC profile. The result of installing ICC profiles is:

- CRDs are stored in the locations outlined Section 4.1.1 on page 6.

- CSAs are stored in folders within the **`SW/colorspaces`** folder.
- Device link profiles are stored within the **`SW/colorspaces`** folder.

The difficulties arising from this arrangement include:

- The naming system must be unique across all plugins.
- Device link profiles are visible for all output plugins when they are, by definition, suited only to one combination of input and output devices.
- The need to install files outside the plugin folder and to possibly require other user actions when delivering color management with a plugin.

4 Converting a plugin to ProofReady operation

Importing an ICC profile, as described above, by default uses the ICC Colorimetric intent (if present), and the profile associated with it is immediately available in the ProofReady pull down menu.

This section discusses ICC profiles and CRDs, and need only be performed if Perceptual or Saturation intents are going to be used.

The steps in converting a plugin are:

- Obtain a CRD, then move or rename it. See Section 4.1 on page 5.
- Make the Harlequin profile use the CRD in its new location. See Section 4.2 on page 6.
- Optionally, protect the color management data from copying. See Section 4.3 on page 10.

Repeat these steps for all relevant profiles and CRDs.

The end result is a plugin where the plugin folder contains all the relevant color management data. This makes it easier to package and distribute the plugin without requiring complex procedures in each Harlequin RIP installation.

Note: The following description is *not* true for a device link profile. For version 5.3r1 and later, installing a device link profile produces data in the plugin folder in a form that is immediately usable, for that plugin only.

4.1 Obtain a CRD and rename it

One way of obtaining a CRD is to install an ICC profile as *media_name*, by entering *media_name* in the **Name** text box in the Install ICC Profile dialog box.

Note: If you are using a supplied ICC profile, ensure that you have permission to redistribute it and that the permission applies to derived files. A ProofReady plugin contains derived files.

media_name represents a name including both the name of the media and the resolution for which it is valid. You may need to abbreviate the media name to make the resolution visible in the **Calibration** or **ProofReady** menu of the Edit Page Setup dialog box. An example of a suitable name is **`D-E Com Matte 1440`**.

If you intend to use different screens with the same plugin device, you should check to see how much the screens affect the profile. If the profiles are very similar, you can use *media_name* as just suggested. If the profiles for the different screens are significantly different, you should add the screening name to *media_name* so that you have a name including the name of the media, the screening, and the resolution. In this case, an example of a suitable name is **`D-E Com Matte HDS Super F 1440`**.

Note: The limit for these names is 31 characters and it is always possible to see all these characters somewhere in the GUI, but in some dialog boxes the Harlequin RIP Classic can display only 22 characters.

Installing an ICC profile as a device profile produces the profile called **`media_name`** and, typically, three related CRD files, with names of the general form **`media_name`**-*uniquenumber*.

Note: These three files each contain one ICC rendering intent (labeled Absolute Colorimetric, Perceptual, or Saturation), which together are sufficient for the Harlequin RIP to generate all four conventional rendering intents. The fourth intent is possible because ColorPro and ProofReady plugins can generate both Absolute Colorimetric and Relative Colorimetric intents from the file holding the Absolute Colorimetric rendering intent, shown in the Create Color Setup dialog boxes as rendering intent **`ICC colorimetric`**.

You maintain maximum flexibility of usage by keeping all the rendering intents available.

4.1.1 Locations

Version note: If you are using version 5.3r1 (or later), the CRD files are already stored in the plugin folder. You have completed this step, and the profile already uses the CRD location within the plugin folder so you do not need to edit the profile.

If you are using version 5.3r0 or earlier, copy all of the installed files from:

SW/icccrd/*media_name-uniquenumber*

If the plugin supports only one device, copy the files to the folder:

SW/Devices/*plugin_name***/colorrenderings/**

If the plugin supports multiple devices, copy the files to the folder:

SW/Devices/*plugin_name***/colorrenderings/***device_name***/**

4.2 Make the Harlequin profile use the renamed CRD

Note: This step is required only for version 5.3r0 or earlier, but the requirement applies both to cases where you have used these versions of the RIP to import the ICC profiles and to cases where you require your plugin to operate with these versions.

To make the plugin self-contained, you must make profiles supplied with the plugin use CRDs stored with the plugin. For each profile, you can ensure this by editing the value of the **ExternalCRDs** key, to point to the location where you installed or copied the relevant renamed CRD.

For a multiple-device plugin, if you installed the ICC profile with name *media_name*, the profile that you must edit is:

SW/Devices/*plugin_name***/Profiles/CMYK/***device_name***/***media_name*

For a single device plugin, if you installed the ICC profile with name *media_name*, the profile that you must edit is:

SW/Devices/*plugin_name***/Profiles/CMYK/***media_name*

The value for the **ExternalCRDs** key is an array that can contain one or more dictionaries, each dictionary giving a style name to be shown in user menus and a file that holds the style definition.

In most usages you only need to change the value for **CRDFileName**. See the definition of the **ExternalCRDs** key on page 24 for details.

The details of what you see and what you may wish to edit vary with the kind of profile and the versions of the RIP. The distinct cases are:

- ICC profile installed into version 5.3r0 or earlier
- ICC profile installed into version 5.3r1 or later

There is another case, which should not require editing but which it is convenient to describe and compare here:

- GGS-supplied profiles

The following subsections describe these cases.

4.2.1 ICC profile installed into RIP 5.3r0 or earlier

This listing from the installed profile shows how the profile refers to the external color rendering dictionaries, as installed by the RIP in the **SW/icccrd** folder.

```
/ExternalCRDs
[
<<
/CRDStyle (\(->ICC perceptual\))
/CRDFileName (icccrd/media_name-uniquenumber1)
>>

<<
/CRDStyle (\(->ICC saturation\))
/CRDFileName (icccrd/media_name-uniquenumber2)
>>

<<
/CRDStyle (\(->ICC colorimetric\))
/CRDFileName (icccrd/media_name-uniquenumber3)
>>

]
```

The next lines of code show how to write the reference to each external color rendering dictionary, once moved to the plugin folder. These examples are for the **-> ICC perceptual** style and *media_name-uniquenumber1*.

If the plugin supports only one device, edit the name to be:

```
/CRDFileName
(%os%Devices/plugin_name/colorrenderings/media_name-uniquenumber1)
```

If the plugin supports multiple devices, edit the name to be:

```
/CRDFileName
(%os%Devices/plugin_name/colorrenderings/device_name/media_name-
uniquenumber1)
```

Note: Because of the length of the example name, the listing here is longer than one line. Do not create a line break in the names you edit.

4.2.2 ICC profile installed into RIP version 5.3r1 or later

This listing shows how version 5.3r1 uses the **%fs%ICCcrd** name server construction to refer to color rendering dictionaries, as installed by the Harlequin RIP in the plugin's **colorrenderings** folder.

```
/ExternalCRDs
[
<<
/CRDStyle (\(->ICC perceptual\))
/CRDFileName (%fs%ICCcrd/device_name/media_name-uniquenumber1)
>>

<<
/CRDStyle (\(->ICC saturation\))
/CRDFileName (%fs%ICCcrd/device_name/media_name-uniquenumber2)
>>

<<
/CRDStyle (\(->ICC colorimetric\))
/CRDFileName (%fs%ICCcrd/device_name/media_name-uniquenumber3)
>>

]
```

These entries and the referenced files are local to the plugin, so the plugin is easy to package for installation elsewhere. However, the **%fs%ICCcrd** naming style is only usable in version 5.3r1 or later, so you may wish to edit these

entries to match those at the end of Section 4.2.1 making the plugin compatible with all Harlequin RIP versions capable of supporting a ProofReady plugin.

4.2.3 GGS-supplied profiles

This example shows what is necessary for the **`Stylus9000`** device of the multiple-device **`epsonplg`** plugin, using a CRD file called **`D-E Com Matte 1440`**.

```
/ExternalCRDs [
<<
  /CRDStyle (\(->Default\))
  /CRDFileName
  (%os%Devices/epsonplg/colorrenderings/Stylus9000/D-E Com Matte
1440)
>>
]
```

This form requires no editing to be usable with all versions of the Harlequin RIP capable of supporting a ProofReady plugin.

Note: There is one difference that users will see if they create a new color setup with ColorPro, and choose a GGS-supplied profile as the output profile. In the New Color Setup dialog box, they will see an entry for **`(-> Default)`** in place of the conventional ICC rendering intents.

When using a RIP generated CRD two lines of code must be removed from the CRD for it to work properly. In the following CRD the lines identified as strike through characters should be removed:

```
%!PS Harlequin Generated Color Rendering Dictionary

% Generated from the following parameters:
%  ColorantFamily: DeviceCMYK
%  KFactor: 1.0
%  RGRange: 20.0
%  Emulation:
%  ManualOverrideRG: 0
%  ManualOverrideBY: 0
%  PluginCharacteristics: 0
%  PluginDate: Wed Dec 12 05:05:56 2001
%  KTableSize: 128
%  MappingBias: 0.0
```

```
%  SaturationMapLStart: 0.25
%  Hull: 1
%  LDRange: 20.0
%  Lighting: Graphic Arts D50
%  ManualOverrideLD: 0
%  MaximumInk: 4.0
%  PageBufferType: Stylus9000_Roll
%  Style: Absolute Colorimetric
%  BYRange: 20.0
%  CRDType: 2
%  Profile: D-E Com Matte 720
%  ProfileDate: 2000/3/10/10/10/00
%  BlackThreshold: 70.0
%  SaturationMapLEnd: 0.5
%  SaturationMapType: 2
%  ProfileDevice: Stylus9000_Roll
%  BlackGeneration: From Profile

/10101000.AAA
currentfile eexec
...
ce1b9f42d8aebebeb3db3ab2ec5faf9732ad26acaf5a01298c12cfc9992b2935d
c2f4c5bdfd4a7a670791ae947df75e3a7eea8db32108f4b27de7bb95bd479f169
31a8c49a7eea61eb8bbe8f74231647
...
/ColorRendering defineresource pop
```

4.3 Protect the color management data

This is an optional procedure but it is an important one if you have created a number of high quality profiles. It is possible to encrypt the data in a number of ways, depending upon the level of security you wish and the way in which you wish to key the encryption to specific users or installations.

See Section 6 on page 11 for details.

5 Customizing default color settings

The settings are supplied with ProofReady plugins, in the file:

```
SW/Devices/epsonplg/Misc/JOBSTART.PS
```

Warning: The **JOBSTART.PS** file is supplied for compatibility of ProofReady plugins with versions 5.1 through 5.3r1. version 5.3r2, and later versions, and it is expected to have the same code provided in its procsets. If the flag

SetColorFromCalibration is provided in **HqnPageSetupConf**, then the version in **JOBSTART.PS** file is not used. You must then use a technique that redefines the defaults supplied by the procsets.

In the **JOBSTART.PS** file, this section defines the default color settings.

```
/CRDFileName get run setcolorrendering
<<
  /Black false
  /DeviceCMYK (DeviceCMYK/SWOP (CGATS TR001)) cvn
    /ColorSpace findresource
  /DeviceRGB (DeviceRGB/Trinitron) cvn /ColorSpace findresource
>> setinterceptcolorspace
```

In summary, this section treats CMYK colors as SWOP (CGATS TR001), RGB colors as Trinitron, and uses relative white and black points. The relative white point is the media color (not the illuminant) and the relative black point is the darkest color that the device can print.

These settings defined in PostScript code correspond closely to settings that a user might make in the ColorPro dialog boxes. For example, the line starting **/DeviceCMYK** has the same effect as when a user chooses **SWOP (CGATS TR001)** as the setting for **CMYK data** in the New Color Setup dialog box.

See Section B.3 on page 21 for details of **setinterceptcolorspace**.

6 Protecting profiles

Protecting profiles (and other pieces of information) involves encryption.

There are at least two options for protecting the important parts of profiles from copying or use with unintended RIP installations.

- Using FireWorks to encrypt data, either in an original ICC profile or in a directly installed CRD.
- Writing a custom filter used as part of an input plugin. This allows the encryption and decryption to be performed against something other than the customer number, as obtained from the dongle.

6.1 Using FireWorks and _hqxrun

This is a simple method, using facilities provided by GGS and involving little programming.

A possible limitation is that the encryption can only be done against the OEM number assigned by GGS or the dongle security number. If this limitation is a real problem, see Section 6.2 for a method that is more flexible but involves creating a plugin.

For a ProofReady plugin, the best place to encrypt data is in the installed CRD.

6.1.1 FireWorks

To encode a file, you can use FireWorks. (The technical note for FireWorks mentions fonts only, but FireWorks is able to encrypt any PostScript-language file.)

Note: You must edit the CRD file before encryption, because the **_hqxrun** operator used to decode and execute files adds **systemdict** to the dictionary stack when it runs, which is read-only. Otherwise, operations in the CRD requiring write-access will fail. For example, you can add **end** at the beginning of the CRD file and **systemdict begin** at the end to restore the situation.

6.1.2 _hqxrun

To decode the CRD file, use **_hqxrun**. This operator decodes and executes referenced files. Firstly, you should rename the encrypted file with an easily recognizable name. You must then create a new file with the same name as the original CRD file containing the following line of PostScript code:

```
(encrypted_CRD_file) {1183615869 internaldict /_hqxrun get exec}
stopped pop
```

The string *encrypted_CRD_file* must include the path to the encrypted CRD file. You should also modify the PostScript code to implement appropriate error handling rather than using **pop**.You must then place this file along with the encrypted CRD file in the correct RIP directory, as described in Section 4.1 on page 5.

When the RIP tries to load a CRD, it runs this file which uses **_hqxrun** to decode the referenced CRD file.

For further details of **_hqxrun**, see the description of encryption in [SWEXTN].

6.2 Using a custom filter

There is background information on creating and using filters in [RB3] and both [SWPLUGIN] and [SWEXTN].

This is an example skeleton of the PostScript-language code for using a filter.

```
%!PS-Adobe-3.0
%%Title: OEMname_decrypt filter wrapper for inline decryption
%%Version 1.0
%%EndComments

currentfile /OEMname_decrypt filter cvx exec
<<encrypted data inserted here>>
<eod>
```

A real usage would have real encrypted data and an end of data marker in place of **<eod>**. (Also, **OEMname_decrypt** is a place holder for a more meaningful name.)

It is possible to test the mechanism in stages.

One way to start would be to use a standard PostScript-language filter and unencrypted data as in this example:

```
currentfile 0 (%%EndSubFile) /SubFileDecode filter cvx exec
<<plain text data inserted here>>
%%EndSubFile
```

This mimics the operation without having to encrypt the data or use the custom plugin to provide a filter.

The next stage would be to use the plugin to just pass through the data, and replace the **SubFileDecode** line with the skeleton at the beginning of this section. At this point you should write the plugin-based filter to detect end of data as **(%%EndSubFile)** or whatever you chose in the **SubFileDecode** example.

Finally the sensitive section of the profile can be encrypted; the end of data marker replaced with whatever marker text you want to use; and the filter enabled to perform its decryption.

7 Actual and expected changes to the Harlequin RIP

There are some changes to the RIP that can provide greater convenience of use with ProofReady plugins. Several changes have taken place during the development of the first ProofReady plugins. Other changes are proposed but have no schedule for completion.

The following changes appear in version 5.3r1 and later:

- Storage of CRDs within the plugin folder, when the user installs an output profile.
- Storage of color spaces within the plugin folder, when the user installs a device link profile. (Color spaces derived from input profiles, will continue to be stored in folders within the **`SW/colorspaces`** folder.)
- A modified search order for CRDs. This will mean that locations specific to the plugin are searched first, before looking for CRDs in other locations.
- The installation of color space data from device link profiles in a plugin-based folder.

Amongst the proposed changes is:

- Movement of the default color settings into a stand-alone file, separate from **`JOBSTART.PS`** or the **`HqnPageSetupConf`** procset.

The anticipated result of these changes is that it becomes easier to supply color resources specific to a plugin in locations specific to that plugin, and that normal use and reasonable user customization of a plugin will not affect other plugins.

Appendix A Supplement to the Plugin Kit Guide

The following items are not documented in the Plugin Kit Guide for version 5.1r2:

- Additions to the plugin directory structure.

A.1 Additions to the plugin directory structure

The plugin directory structure for a freshly installed ProofReady plugin contains some extra directories not found in earlier plugins. The main addition is the **colorrenderings** directory, but there are some minor changes to lower level folders, as shown by the thick lines to the left of the listing that follows.

- In version 5.3r0 and earlier, these directories exist only if the plugin installation procedure generates them.
- In version 5.3r1 and later, some user actions in the GUI add directories or files to this structure. Installing an ICC profile as an output profile is one such action at version 5.3r1. Other actions may be added later.

The ability of the plugin to use files in directories local to the plugin is dependent on writing PostScript-language code to do so. In the ProofReady plugin, the main use of this ability is in profiles — to refer to external CRDs stored elsewhere but also within the plugin directory.

```
SW
   Devices
      <plugin directory>
         <plugin-executable>
         <plugin>.map
         <any other arbitrary files for plugin or installation use>
         colorrenderings
            <directories named by device type>
               < files named by usage, for example, paper and resolution>

         colorspaces
               <directories named by color space, such as DeviceLinkRGB>
                  <directories named by device type>
                     < files named by the user, defaulting to profile's filename>

         Setups
            Setup
            <or several setup files named by device type>
         Misc
            <any other folders/files to be accessible from the PostScript language>

         Targets
            Target
            <or several targets named by device type>
            <or folders named by colorant family, holding target files>
         targeteps
            <one or several target EPS files named arbitrarily or by function>
```

Profiles
    <*directories named by device type***>**
        < ***Harlequin RIP color profile files*****>**

Screen Names
    Screen Names
    <*or several screen names files named by device type***>**

Factory Settings
    <*duplicate of rest of tree for factory reset***>**
<*UNIX Motif X resource file***>**

Appendix B Supplement to Using Harlequin RIP Extensions

There are several items that do not appear in the *Using Harlequin RIP Extensions* manual. These are:

- Support for relative neutral white and black points. See Section B.1 on page 16.
- New content for Harlequin profile files. See Section B.4 on page 24.

B.1 Relative neutral colors

The PostScript-language color model includes provision for a chromatic adaptation transform using a **`WhitePoint`** from both the CSA and CRD. There are, however, two candidate values for the white point: the Illuminant and the Media White point.

The **Simulate paper color of job** option in the RIP UI controls whether the RIP uses **`WhitePoint`** or the **`RelativeWhitePoint`** for color conversions. Please refer to *Using Harlequin RIP Extensions, section 16.8* for richer methods of controlling color.

B.2 Extensions to CRDs and CSAs

The Harlequin RIP supports additional keys in CRDs and CSAs. These keys are present in CRDs and CSAs generated by the RIP, whether the origin of the data is a profile created by the HFCS color management option or an installed ICC profile.

The following tables describe the additional keys for CRDs and CSAs.

Table: CRD keys

`RelativeWhitePoint` *array*

(Optional) An array of 3 numbers, expressing a color in CIE XYZ values. In the case of print media, the **`RelativeWhitePoint`** is the color of the media, whereas the **`WhitePoint`** is the color of the illuminant.

If this key is not supplied, the default is **`WhitePoint`**.

`XUID` *array* **`of`** *integers*

A unique identifier for the CRD, produced when the CRD is generated. The Harlequin RIP uses this identifier to make efficient use of color caching. This can be any unique array. GGS uses the following conventions.

The array contains 8 elements, whether derived from a Harlequin profile or imported from an ICC profile. (The values of the resource category and the rendering intent elements *do* vary according to the source of the CRD.) Here is an example:

```
/XUID [124 11 3 744171 67030 742837 42846 0]
```

In order, the values are:

1. Adobe organization ID. For GGS OEMs, this is always 124.
2. A customer (OEM) number, assigned by GGS.
3. Resource category, described in Section B.2.1 on page 19.
4. CRD creation date.
5. CRD creation time.
6. Output profile date.
7. Output profile time.
8. CRD rendering intent number, described in Section B.2.3 on page 20.

All dates and times are encoded as described in Section B.2.2 on page 19.

Table: CSA keys

The Harlequin RIP supports additional CSA keys.

`RelativeWhitePoint` *array*

(Optional) An array of 3 numbers, expressing a color in CIE XYZ values. In the case of print media, the **`RelativeWhitePoint`** is the color of the media, whereas the **`WhitePoint`** is the color of the illuminant.

If this key is not supplied, the default is **`WhitePoint`**.

`XUID` *array* **`of`** *integers*

A unique identifier for the CSA, produced when the CSA is generated. The RIP uses this identifier to make efficient use of color caching. This can be any unique array. GGS uses the following conventions. The length of the array is different depending upon the source of the CSA. CSAs can be generated by the Harlequin RIP or imported from an ICC profile.

For a CSA generated by GGS, the array contains 5 elements. Here is an example:

```
/XUID [124 0 0 19961128 13150]
```

In order, the values are:

1. Adobe organization ID. For GGS OEMs, this is 124.
2. A customer (OEM) number, if appropriate. Assigned by GGS.
3. Resource category, always value 0. Described in Section B.2.1.
4. CSA creation date.
5. CSA creation time.

Dates and times are encoded as described in Section B.2.2.

For a CSA imported from an ICC profile, the array contains 7 elements. Here is an example:

```
/XUID [124 11 2 744150 70471 742985 64062]
```

In order, the values are:

1. Adobe organization ID. For GGS OEMs, this is 124.
2. A customer (OEM) number, if appropriate. Assigned by GGS.
3. Resource category, always value 2. Described in Section B.2.1.
4. CSA creation date.
5. CSA creation time.
6. ICC profile date.
7. ICC profile time.

Dates and times are encoded as described in Section B.2.2.

Note: The format of XUIDs for CSAs imported from ICC profiles is likely to change.

B.2.1 XUID resource categories

The resource categories are coded as integers. Table 1 shows the values and their meanings.

Value	Meaning
0	Harlequin color space
1	Harlequin CRD
2	ICC imported color space
3	ICC imported CRD

Table 1 Resource category values

There is no GGS support for other resource categories.

B.2.2 XUID date and time encoding

The date encoding is:

```
yyyy*(12*31) + mm*31 + dd
```

where:

dd	is the day of the month in the range 1 through 31 as applicable to the month.
mm	is the month of the year with January being 0 and December being 11.
yyyy	is the full year number including century. For example, use 2000, not 00.

The time encoding is:

```
hh*(60*60) + mm*60 + ss
```

where hh:mm:ss is in 24 hour format, that is, between 00:00:00 and 23:59:59.

B.2.3 XUID CRD rendering intents

CRD rendering intent numbers correspond to a particular rendering intent. The values used by rendering intents in Harlequin profiles are:

Value	Meaning
32768	Perceptual
32769	Relative colorimetric
32770	Saturation
32771	Absolute colorimetric

Table 2 Rendering intents from Harlequin profiles

In a CRD installed from an ICC profile, the rendering intent takes one of the values shown in Table 3.

Value	Meaning
0	ICC Perceptual
1	ICC RelativeColorimetric

Table 3 Rendering intents from ICC profiles

Value	Meaning
2	ICC Saturation
3	ICC AbsoluteColorimetric

Table 3 Rendering intents from ICC profiles

GGS reserves the range of values 32768 (2^{15}) through 8388607 (2^{23} - 1) for CRDs.

OEMs who generate their own CRD rendering intents can use values in the range 2^{23} through 2^{31} - 1.

Example code to do this could use the following hash define statements:

```
#define INTENT_USER_BASE  (1 << 23)

#define USER_INTENT_A     (INTENT_USER_BASE)
#define USER_INTENT_B     (INTENT_USER_BASE + 1)
```

B.3 Intercepting color spaces

It is sometimes useful to intercept definitions made in one color space and route the equivalent color to another color space. The RIP can perform this interception at the point of encountering a color space in a job. Additionally, The RIP can treat solid black as a special case color.

The relevant operators are **`setinterceptcolorspace`** and **`currentinterceptcolorspace`**.

Some examples and the definitions follow.

B.3.1 Examples of color space interceptions

The RIP can make interceptions of complete color spaces separately for PostScript-language color spaces — DeviceRGB and so on. For DeviceCMYK, it can also distinguish between CMYK used for images and for other elements in a job. The RIP can also intercept color spaces or profiles attached to Photoshop images.

For example, this usage demonstrates no interception for CMYK colors used outside images:

```
<<
  /DeviceCMYK null
>> setinterceptcolorspace
```

In the dictionary supplied to **setinterceptcolorspace**, the key **/DeviceCMYK** identifies the input case and the value **null** specifies no interceptions. (If other keys are supplied in the dictionary, there may be interceptions for other cases.) Because there is no interception, any one of the following color definitions and implied color spaces found in a job is treated as defined by [RB3]:

c m y k **setcmykcolor**

/DeviceCMYK setcolorspace *c m y k* **setcolor**

This next example sets a significant interception:

```
  << /DeviceCMYK [/CIEBasedDEFG params ] >>
setinterceptcolorspace
```

This example says that the same (CMYK) color space definitions in a job are to be converted to the output space using the given **CIEBasedDEFG** color space.

Similar interceptions are possible for RGB, Gray, and named color spaces. For example:

```
<< /DeviceRGB [/CIEBasedABC params ] >> setinterceptcolorspace
```

says that:

r g b **setrgbcolor**

would divert the *r*, *g*, and *b* values through the specified **CIEBasedABC** space.

The final class of interception for color spaces is for color definitions supplied with an attached color space or profile. The interceptions are controlled by the **PhotoshopCMYK**, **PhotoshopGray**, and **PhotoshopRGB** keys. These interceptions are valid for any form of file produced by Photoshop where the color space or profile is included. These file types include EPS, and TIFF/JPEG files

An example of their use is:

```
<<
  /PhotoshopCMYK /DeviceCMYK
>> setinterceptcolorspace
```

B.3.2 Examples for 100% Black

The case of a color defined as 100% black is treated specially to allow control of cases, such as text and fine line work, where it is preferable to use a solid black. This interception can avoid a color-managed equivalent which may involve several components with potential for problems with screening or registration.

Note: The RIP does not intercept all colors that appear 100% black. Two important exceptions are superblack (or rich black) and CMY black. A superblack is deliberately used to have non-zero components in channels other than black. A CMY black such as **`1 1 1 0 setcmykcolor`** is *not* intercepted as 100% black, because there is a more exact and reasonable way to define black: **`0 0 0 1 setcmykcolor`**. There are similar exclusions in other color spaces.

The interception of black takes place for reasonable definitions in most color spaces. These are the intercepted cases:

- **`0 0 0 1 setcmykcolor`**
- **`0 setgray`**
- **`0 0 0 setrgbcolor`**
- **`/DeviceCMYK setcolorspace 0 0 0 1 setcolor`**
- **`[/Indexed 100 /DeviceCMYK {...}] setcolorspace`** *`n`* **`setcolor`**

  … where the color indexed by *n* has CMYK values **`0 0 0 1`**.
- **`[/Separation /Black ... {...}] setcolorspace 1 setcolor`**

B.3.3 Operator definitions

Please refer to *Using Harlequin RIP Extensions, section 16.1.6* for the full list of operator definitions.

B.4 New content in Harlequin profiles

There are two new comments, **%%ICC Profile**, and **%%Creator** and one new key, **ExternalCRDs**, to support ProofReady plugins.

There is one new key, **LinearizationName**, to remove the spurious display of an error message. Only version 5.3r1 and later is able to use this key to avoid the spurious message.

All these additions are optional. The RIP will continue to be able to use profiles without these keys via ColorPro.

%%ICC Profile

If present in the comments section of the profile, this comment makes the profile selectable for use with ColorPro. Typically, this line follows the **$$Title** comment.

%%Creator

This comment allows the creator of the profile to be identified. This is only done atuomatically when profiles are made using SetGold.

ExternalCRDs

The value for the **ExternalCRDs** key is an array that can contain one or more dictionaries, each dictionary giving a style name to be shown in user menus and the location of a file that holds the style definition. For example:

```
/ExternalCRDs [
<<
  /CRDStyle (\(->Default\))
  /CRDFileName
  (%os%Devices/epsonplg/colorrenderings/Stylus9000/D-E Com Matte
1440)
>>
]
```

Note: A CRD produced by installing an ICC profile in version 5.3r1 or later has the following form for **/CRDFileName**. The CRD file is in the same plugin-based location as for the example just given but these versions of the RIP use the **%fs%ICCcrd** file name server key, which is more

compact (and introduces the possibility of searching **SW/icccrd** if the named CRD is not in the plugin-based location).

```
/CRDFileName   (%fs%ICCcrd/D-E Com Matte 1440)
```

A ProofReady plugin is able to use one dictionary only. When there are several dictionaries, as is likely when the profile has been created by installing an ICC profile, the search order used by a ProofReady plugin is for ICC Colorimetric (**->ICC colorimetric**), and then the first dictionary listed. The plugin uses the first match found when using this search order on the values of **CRDStyle**.

Note: If there is a ColorPro color setup in operation, any match made by a ProofReady plugin is overridden by choices made in the color setup. The match made by a ProofReady plugin is used only when the choice for **Color** in the Edit Page Setup dialog box is **(None)**. A ColorPro option is able to use a dictionary with any of the conventional labels for styles: **->ICC perceptual**, **->ICC saturation**, or **->ICC colorimetric** and uses the one(s) needed for choices made in the relevant color setup.

CRDStyle *must* include the leading string **->**, as shown in the example listing.

CRDFileName must be the name of a CRD file, accessible to the RIP. **%os%** is a convenient way of referring to the **SW** folder of the Harlequin RIP installation.

Note: Profiles containing this key are identified as ProofReady profiles and cannot be uninstalled.

LinearizationName

(Optional, but recommended.) The value of this key is a string identifying the profile from which the default linearization data was taken. Harlequin RIP, version 5.3r1 and later, uses the string to avoid spurious error messages about mismatches when the user chooses color setups or profiles and calibration sets.

For example, if a Harlequin profile is created by installing an ICC profile, the RIP sets this string to indicate which default linearization the user chose (using the GUI control **Linear Calibration From** in the Install ICC Profile dialog box). If the calibration profile has the

LinearizationName key, the RIP copies its value into the installed profile; otherwise, the RIP uses the file name of the calibration profile to create the key value in the installed profile.

Typical values for this string identify the device, media, and resolution for which the default linearization was derived. An example is:

```
/LinearizationName (Epson 3000; PhotoPaper; 1440)
```

If the default linearization is actually the raw state of the device, the value may be this:

```
/LinearizationName (Linear)
```

If **LinearizationName** is not present, there are some cases where the Harlequin RIP displays error messages about a possible mismatch between a calibration and a profile. These messages are often an unnecessary alarm.

Appendix C External references

The following books are valuable sources of supporting information.

[RB3] *PostScript Language Reference Manual, 3rd Edition* (the "Red Book"), Addison Wesley.

[HQEXTN] *Using Harlequin RIP Extensions: Guide for OEMs*, edition 7.0. Global Graphics Software Limited.

[HQHCP] *Harlequin ColorPro User's Guide*, edition 7.0. Global Graphics Software Limited.

[HQOEM] *Harlequin RIP OEM Manual*, edition 7.0. Global Graphics Software Limited.

[HQPLUGIN] *Harlequin RIP Plugin Kit: Guide for OEMs.* Global Graphics Software Limited.

There is also a draft document describing the calibration-related portions of Harlequin profiles. (*Harlequin RIP: Profiles and Calibration*, June 2001).

Change history		
v 1.0	2000.06.26	First issued, for ScriptWorks 5.3
v 1.1	2000.09.01	Add %fs% usage and minor corrections
v 1.2	2001.06.18	Updated cover page and copyright page. Removed references to ScriptWorks and replaced with Harlequin RIP. Updated Appendix.
v 1.3	2001.10.30	Modify _hqxrun details.
v 1.4	2002.09.27	Add information on how to edit a RIP generated CRD, see Section 4.2.3. Change all HIPP and HCPS references to ColorPro and rename options which have changed in name. Add **%%Creator** information.
v 1.5	2002.11.15	Changed copyright. Changed version to Eclipse Release.
v1.6	2005.02.04	Small updates to text, copyright and logos.
v1.7	2011:04:12	Replace **DeviceCMYKPicture** and **DeviceCMYKOther** with **DeviceCMYK** and remove referencces s to **RelativeBlackPoint**, **setrelativeneutrals** and **currentrelativeneutrals**.. Updated Extensions manual references.