XRS Version 1.2: Summary of differences from Version 1.1

Page 7:

Change:
Replace the bullet point:

32-bit Windows: the application should check the registry for the `SearchPath` value in the `HKEY_CURRENT_USER\Software\XRS` key. Plug-ins should store setting information in a subkey in this key.

with the following bullet point:

32-bit Windows: the application should check the registry for the `SearchPath` value in the `HKEY_LOCAL_MACHINE\Software\XRS` key. Plug-ins should store setting information in a subkey in the `HKEY_CURRENT_USER\Software\XRS` key.

Reason:
The registry key is changed to HKEY_LOCAL_MACHINE because currently that is where the SearchPath value is set. The CURRENT_USER key is introduced as user data should not be stored under the LOCAL_MACHINE key.

Page 10:

After the paragraph:

Bit 6:
Informs the application not to show the plug-in in the plug-in list and/or menu but can still be accessed from the PM_GETNEXTPLUGIN command. (A ‘hidden’ plug-in.)
The second lowest byte specifies the class of the plug-in (this can be ORed with the above bit specifiers):

add the following:

Bit 7:
Indicates that the plugin window handle will be returned in xrsPluginStart function and allows the XRS server to manipulate it (for example to embed to the application panel).

After the paragraph:

0x0300: Decoder class – plug-ins that perform any decoding of the received signal use this category.

add the following:

0x0400: Demodulator class - plugins that perform demodulating of the received signal use this category. For proper running of all other plug-ins, demodulator plug-ins should be the last ones destroyed.
0x0500: DF class - plugins that perform direction finding of the received signal use this category.

Page 18:

After the paragraph:

Bit 6:
Informs the application not to show the plug-in in the plug-in list and/or menu but can still be accessed from the PM_GETNEXTPLUGIN command. (A ‘hidden’ plug-in.)

add the following:

Bit 7:
Indicates that the plugin window handle will be returned in xrsPluginStart function and allows the XRS server to manipulate it (for example to embed to the application panel).

Page 19

In the table, the row

PNF_RXAUDIO	No PNR_VOLUME, PNR_MUTE, PNR_BALANCE, etc notifications

replace by

PNF_RXAUDIO	No PNR_DEMODSIGNAL, PNR_VOLUME, PNR_MUTE, PNR_BALANCE, etc notifications

---

Page 29:

Before the paragraph:

DSPCAPS

insert the following:

DEMODSIGNALDATA
The DEMODSIGNALDATA structure is used to pass the samples from a digital demodulator point to a plug-in through the PNR_DEMODSIGNAL message. The XRS server receives the samples from a demodulator plug-in using the same structure and dispatches it to all other plug-ins.

```
C/C++:
  typedef struct _DEMODSIGNALDATA {
     int    iSamplingRate;
     int    iBitsPerSample;
     int    iNumChannels;
     int    iNumSamplesSets;
     BYTE   Samples[1];
  } DEMODSIGNALDATA, FAR *LPDEMODSIGNALDATA;

Delphi:
  type
    PDemodSignalData = ^TDemodSignalData;
    TDemodSignalData = record
```

```
        iSamplingRate: Integer;
        iBitsPerSample: Integer;
        iNumChannels: Integer;
        iNumSamplesSets: Integer;
        Samples: array [0..0] of Char;
    end;
```

Fields:

iSamplingRate
Specifies the sampling rate corresponding to the samples in the structure.
iBitsPerSample
Specifies the size of each sample stored in the structure in bits. It must be a multiple of 8.
iNumChannels
Specifies the number of channels for which the samples are interlaced in the structure.
iNumSamplesSets
Specifies the number of sets of samples contained in the structure. Such a set contains one sample for each channel.
Samples
The actual samples contained in the structure. The total size of this field is given by:
iNumSamplesSets * iNumChannels * iBitsPerSample / 8

Page 37:

Replace the paragraph:

DAB:

Specifies the supported digital audio broadcasting standards. Each set bit represents supported standards:

0 = Eureka 147
1 = IBOC
2 = WorldSpace

With the following:

DAB:

Specifies the supported digital audio broadcasting standards. Each set bit represents supported standards:

0 = Eureka 147
1 = IBOC
2 = WorldSpace
3 = DRM

Page 38:

Replace the paragraph:

DAB:

Specifies the supported digital audio broadcasting standards. Each set bit represents supported standards:

0 = Eureka 147
1 = IBOC
2 = WorldSpace

With the following:

DAB:

Specifies the supported digital audio broadcasting standards. Each set bit represents supported standards:

0 = Eureka 147
1 = IBOC
2 = WorldSpace
3 = DRM

Page 54:

Before the paragraph:

PM_CLOSED

insert the following:

PM_CAPABILITIES
The PM_CAPABILITIES command informs the application that the capabilities of the receiver changes due to the plug-in starting/stopping. If the command is sent while the plug-in starting phase, the new capabilities must be specified through a modified copy of the RADIODEVCAPS structure passed as argument of the xrsPluginStart exported entry point. Any changes in the content of the structure should affect only sections covered by the running plug-in (i.e. only the list of available modes when the plug-in is a demodulator one). Any change to the capabilities of the radio receiver must be changed back when the plug-in is stopped.
Parameters
dwParam
Not used
cbData
The amount of memory occupied by the new RADIODEVCAPS structure.
lpData
Pointer to the new RADIODEVCAPS. After passing the information to the XRS server the memory can be freed.

Page 73:

Before the paragraph:

PMR_IFGAIN

insert the following:

PMR_DEMODSIGNAL
The PMR_DEMODSIGNAL command can be sent only by demodulator plug-ins. It sends a buffer of samples from a specific point in the demodulator for other plug-ins that might need it. Samples can be modified to implement extra signal processings like audio signal conditioning.

Parameters
dwParam
A constant specifying the demodulator point where the samples have been obtained.

```
        DEMODSIGNAL_IF          - IF input
        DEMODSIGNAL_IQ          - I and Q samples before filtering
        DEMODSIGNAL_IQ_FILTERED - I and Q samples after filtering
        DEMODSIGNAL_AUDIO       - audio output
```

cbData
The amount of memory occupied by the structure containing the samples.
lpData
Pointer to the structure containing the samples, DEMODSIGNALDATA.

Page 74:

Before the paragraph:

PMR_LOUD

insert the following:

PMR_IFSPECTRUM
The `PMR_IFSPECTRUM` command is sent by a digital demodulator plugin to the application to provide the spectrum resulted from the IF input signal. When receiving this command, the application, apart from using it, must send `PNR_IFSPECTRUM` notifications to all plug-ins.
Parameters
dwParam
Not used
cbData
The amount of memory occupied by the IF spectrum samples.
lpData
Pointer to the vector of IF spectrum samples. Each sample is stored using 32-bit unsigned integers with (2^32-1) corresponding to the maximum possible level.

Pages 75, 83, 99, 106

In the mode constant list, after the line

RADIOMODE_DAB - Digital Audio Broadcasting

add the following:

RADIOMODE_FM3	- Frequency modulation with 3 kHz deviation
RADIOMODE_FM6	- Frequency modulation with 6 kHz deviation
RADIOMODE_AMN	- Narrow bandwidth amplitude modulation
RADIOMODE_DSB	- Double side band amplitude modulation with supressed carrier
RADIOMODE_ISB	- Independent side band amplitude modulation with supressed carrier

---

Page 83:

Replace the paragraph:

DAB:

The *dw…Param1* field specifies the digital audio broadcasting standard:

0 = Eureka 147
1 = IBOC
2 = WorldSpace

With the following:

DAB:

The *dw…Param1* field specifies the digital audio broadcasting standard:

0 = Eureka 147
1 = IBOC
2 = WorldSpace
3 = DRM

Page 88:

Before the paragraph:

PN_CLOSE

insert the following:

PN_CAPABILITIES
The PN_CAPABILITIES message informs the plug-in that the capabilities of the receiver changed due to another plug-in starting/stopping. The plug-in must be able to handle the changes that affect it without restarting.
Parameters
dwParam
Not used
cbData
The amount of memory occupied by the new RADIODEVCAPS structure.
lpData
Pointer to the new RADIODEVCAPS.

Page 97:

Before the paragraph:

PN_FMWDATA

insert the following:

PNR_DEMODSIGNAL
The PN_DEMODSIGNAL dispatches buffers with samples from various points in digital demodulators either for

study or supplementary signal processing.
Parameters
dwParam
A constant specifying the demodulator point where the samples have been obtained.

```
        DEMODSIGNAL_IF          - IF input
        DEMODSIGNAL_IQ          - I and Q samples before filtering
        DEMODSIGNAL_IQ_FILTERED - I and Q samples after filtering
        DEMODSIGNAL_AUDIO       - audio output
```

cbData
The amount of memory occupied by the structure containing the samples.
lpData
Pointer to the structure containing the samples, DEMODSIGNALDATA.

Page 98:

Before the paragraph:

PNR_LOUD

insert the following:

PNR_IFSPECTRUM
The `PNR_IFSPECTRUM` message notifies the plug-in that a digital demodulator plugin sent to the application the spectrum of its IF input signal. The plug-in must not affect the spectrum samples.
Parameters
dwParam
Not used
cbData
The amount of memory occupied by the IF spectrum samples.
lpData
Pointer to the vector of IF spectrum samples. Each sample is stored using 32-bit unsigned integers with (2^32-1) corresponding to the maximum possible level.

Page 102:

Replace the paragraph PNR_SLEVEL with the following paragraph:

PNR_SLEVEL

The PNR_SLEVEL message notifies the plug-in what the currently received signal strength is. Typically, this is called at regular intervals to keep the plug-in updated with the latest signal level (even if it has not changed).

Parameters
dwParam
The current received signal strength. This can be an arbitrary value from 0 to a maximum or in actual dBm.
If the reading is in dBm, the RADIOCAL_SLEVEL flag is set in the *dwCalibrated* field of the RADIODEVCAPS structure.
cbData
The current received RAW signal stregth. This is 8-bit value obtained from DAC.
lpData
Not used.

Page 106:

Replace the paragraph:

DAB:

The *dw…Param1* field specifies the digital audio broadcasting standard:

0 = Eureka 147
1 = IBOC
2 = WorldSpace

With the following:

DAB:

The *dw…Param1* field specifies the digital audio broadcasting standard:

0 = Eureka 147
1 = IBOC
2 = WorldSpace
3 = DRM