REXX FULL SCREEN SYSTEM (RXFS)

USER DOCUMENTATION

2 Feb 2015

Document Resides on VMA:NEALE.RXFS.DOC
Author: David Tobin, Monica Hurdal, Neale Ferguson

Table of Contents

1.0 Introduction

This document describes the use of the BiCentenary Project (BCP) / ReXx Full Screen (RXFS) system. It is written in the context of the re-written system, and extensively details the new features and enhancements of the re-written system. It uses terms and concepts which would be meaningful a programmer familiar with the TAB computer environment.

This documents covers the following:

- An overview of the function of each of the programs in the system.
- Documention on how to use each of the programs in the system.
- A guide to the Advanced Features of the RXFS system.
- An appendix containing:
    - A list of the various settings for the ACTION_KEY variable.
    - A list of the system return codes.
    - A description of the various files that are used by the RXFS system.
    - A table of each RXFS keyword, along with the valid representations and abbreviations for each.
    - A summary of the major differences between the new RXFS system and the old system.

2.0 Overview of RXFS programs

The RXFS system consists of several programs. The purpose of each of these programs is briefly outlined below.

2.1 Overview of RXFSOPEN

RXFSOPEN sets up memory so that it can be used by other programs in the RXFS system. When RXFSOPEN is called, it sets up a **pageset** (which is a set of 10 screen buffers) to be used. The first time that an RXFSOPEN that is issued for a particular device causes that device to be set up for a fullscreen environment.

As many pagesets as required may be created by issuing further RXFSOPEN commands, either within the same EXEC, or from another EXEC, although only the pageset for the most recent RXFSOPEN for a device may be used.

There are some other more technical features of RXFSOPEN, which are outlined in detail in "RXFSOPEN Usage" on page 4.

2.2 Overview of RXFSWRIT

This program adds fields to pages for the most recent pageset (the one most recently RXFSOPENed). If a "PANEL" keyword appears in the arguments to RXFSWRIT, then a CMSPANEL file is parsed, and all fields defined in the file are added to the current page. Otherwise, if a field is defined in the arguments to RXFSWRIT, then this field is added to the current page. By using the "PAGE" keyword, the current page can be changed.

For a complete description of the use of RXFSWRIT, see "RXFSWRIT Usage" on page 5.

2.3 Overview of RXFSREAD

The RXFSREAD program will display the current page of the pageset. (i.e. it displays the fields that have been put into the buffer by RXFSWRIT). All variable fields on the page displayed will contain the current REXX values for those variables (if defined).

If the **NOWAIT** argument is **not** present in the arguments to RXFSREAD, then it will wait for a PF Key, a PA Key, Clear, Enter, or other special interrupt to occur, before updating the REXX variables for which there were modified variable fields on the screen.

RXFSREAD also sets some other REXX variables such as SCR_DATA, PFKEY, PAKEY, ACTION_KEY and CURSOR_ADDRESS.

For a complete description of the processing of RXFSREAD, see "RXFSREAD Usage" on page 8.

2.4 Overview of RXFSCLOS

RXFSCLOS causes the current pageset to be removed from the system, freeing all of the memory that was set up by the RXFSOPEN for it, and making the previous pageset for that device active. If there are no more pagesets active for the device, then the fullscreen environment set up by RXFSOPEN for that device is closed.

For a complete description of the processing of RXFSCLOS, see "RXFSCLOS Usage" on page 9.

2.5 Overview of RXFSPURG

RXFSPURG will close all pagesets currently active for every device. This means that every RXFSOPEN that is currently active will be closed. This is the same as doing an RXFSCLOS for *every* RXFSOPEN that is still active.

2.7 Overview of RXFSDUMP

RXFSDUMP produces a disk file which contains a "screen dump" of the current page (i.e. the screen that would be displayed if RXFSREAD were done). The file produced is of the type BCPDUMP.

For more information on the functioning of RXFSDUMP, see “RXFSDUMP Usage” on page 9.

2.8 Overview of RXFSHELP

RXFSHELP will display help for the given return code from an RXFS program. Help is available for return codes from all RXFS programs.

3.0 How to Use the RXFS system

3.1 RXFSOPEN Usage

RXFSOPEN will set up a new pageset buffer for the given device. RXFSOPEN will also load any requested programmed symbol set files (if possible on the device of the RXFSOPEN).

There are several REXX variables that get set by RXFSOPEN. These are:

- **SCR_WIDTH** - This variable contains the number of columns that are available on the device for which the RXFSOPEN was performed.
- **SCR_HEIGHT** - This variable contains the number of rows that are available on the device for which the RXFSOPEN was performed.
- **DEVICE_NAME** - This contains a unique name for the device that has been opened. It only gets set on the first RXFSOPEN for the device. This name is used by the other RXFS programs as a value for the "DEV" keyword.

The following keywords are valid for RXFSOPEN:

Keyword	**Description**
DEVice	Changes the device for which the pageset is set up. The value of this keyword is the address of a GRAF device that has previously been defined. e.g. To do an RXFSOPEN for GRAF 020, the DEV keyword would be DEV=020. Default: DEV=FFFF - First RXFSOPEN for any device. If there are any previous RXFSOPENs, then the default will be the device address of the previous RXFSOPEN. *See "Using Devices Other than the Console" on page 12 for more information on how to use the the DEV keyword.*
PSA, PSB, PSC, PSD, PSE, PSF, PSG, or PSH	This keyword is used to specify the names of RXFSPSS files to be loaded as programmed symbol sets (character sets). The character sets are only loaded if the terminal has an available slot for a character set. PSA is the first available slot, PSB the second, and so on. *For more information on how to use character sets see "Character Sets" on page 13.*

Field definitions in CMSPANELs are exactly the same as a field definition from the arguments line. All keywords that are valid from command line arguments are valid in CMSPANELs (except for the **CLEAR** and **PANEL** keywords). *For further information on CMSPANEL files, see "CMSPANEL Filetypes" on page 19.*

Below is a list of the keywords which are valid for RXFSWRIT (and keywords which are valid in CMSPANELs).

Keyword	**Description**
Alarm	Will sound the alarm when RXFSREAD is done. Y - alarm sounds N - alarm doesn't sound Default - N
Border (NEW)	Indicates the type of window border to be used. R - reverse video as borders L - lines as borders N - no borders Default - R but if exthilight is not available then L.
CHaracter (NEW)	User defined and stored character set or character set provided with the device A, B, ..., H - loadable character set (Loaded by RXFSOPEN). 0 - nonloadable character set (device provided) Default - 0. Note that if A, ..., H is given, but not available on the current terminal, then they will also default to 0.
CLear	Erase all fields from the current screen (page) Y - erase data N - do not erase data Default - N **NOT VALID IN CMSPANELs.**
COLOR	The colour of a field on the screen. BLU- blue, GRE- green, TUR- turquoise, RED- red, PIN- pink, WHI- white, YEL- yellow, NEU- neutral, DBL- deep blue (defaults to BLU), PGR- pale green (defaults to GRE), PTU- pale turquoise (defaults to TUR), PUR- purple (defaults to RED), NWH- neutral (white on displays), ORA- orange (defaults to YEL), GRY- grey (defaults to WHI), BLA- black (defaults to NEU). Default - TUR
Column	The column at which a field starts Default Value - Not Applicable Valid values are integers. Negative values are taken from the right side of the screen.
CUrsor	Will position cursor at the given row/column Y - position cursor N - do not position cursor Default - N
Detectable (NEW)	Field detected by selector pen or cursor select key Y - field detectable N - field not detectable Default - Y *See "Using Detectable Fields" on page 13.*
DEVice (NEW)	Changes the device for which subsequent fields are added. The value of this keyword is a character string which represents the name of a device that has been RXFSOPENed (this name is returned by RXFSOPEN when a device is first opened in the **DEVICE_NAME** REXX variable). Fields are added to the current page of the current pageset of the device given.

	Default - B
Exthilight	Parameters U - underscore = each character or field is underlined B - blink = each character or field affected flashes on and off R - reverse video = effect is white on black becoming black on white N - normal video (i.e. no extended highlighting) Default - N
Fill	If the length of a field (TEXT or VNAME) is larger than its value, then the remaining characters after the value will be the fill character specified for that field. Default Value - N (NULL) Note: To have the space character for fill, use FILL=B.
Height (NEW)	Indicates the height (number of rows) of a window. Default Value - the screen height.
Length	The length of a field Default Value - (Text Fields) - length of 'TEXT' initialisation. (Vname Fields) - Length of REXX Variable Value (it will be the length of the Vname if not defined). Valid values are positive integers
Outlining (NEW)	Field outlining. 16 kinds of outlining can be defined by the combinations of the 4 horizontal and vertical lines. O,U,L,R - A combination of O, U, L and R can be used to specify outlining Over, Under, Left and Right of a field respectively. N - none Default - N ***Note: At the time of writing, there are no terminals at the TAB which support field outlining.***
PAge	Changes the current page in the pageset. This means that all fields added will go onto the new page, and also, when RXFSREAD is done, the new page will be shown. Note: Any fields or panels defined on the same RXFSWRIT as a PAGE keyword will go into the page given by the PAGE keyword. Also, the PAGE keyword can appear in a CMSPANEL file, so that one panel defines several pages. Values - Integer 0 through 9 Default - Previous page number given (0 if none given yet).
Panel	Read and parse a CMSPANEL. The file name is arbitrary but the panel must be of filetype CMSPANEL. All fields added will go onto the current page. (Unless a PAGE keyword within the CMSPANEL changes the current page). Default - Not Applicable **NOT VALID IN CMSPANELs.**
PRotection	Type of field protection required. Y - protection required N - no protection required I - invisible and unprotected field required Default - Y
Row	The row (line number) at which a field starts Default Value - Not Applicable Valid values are integers. Positive values are from the top of the screen, negative are from the bottom.
Skip	Skip at the end of an unprotected field to the next unprotected field. Y - skip N - no skip Default = Y
Text	Text to be displayed in the field. If a field is a TEXT field, then the value cannot be changed be the program. The text must be enclosed by ' (quote) characters. If the text keyword is specified, the VNAME keyword cannot be specified. (e.g. TEXT = 'This is some text.'). Default - Not Applicable

(used for windows)	Y - yes = tranparent = underlying presentation space can be viewed Default - Y ***Currently not fully supported.***
TYpe (Formerly CAse)	Type or case of field required M - mixed case (alphanumeric field) U - upper case (alphanumeric field) N - numeric field (new feature). RXFSREAD sets a return code if non-numeric data entered into a numeric field. Valid characters in numeric fields are '0', '1', ..., '9', ',', '.', and ' '. Default - M
VAlidation (NEW)	E - mandatory entry = field must be modified before before data can be transmitted from the display F - mandatory fill = a field that if modified, must be filled with characters other than null before the cursor can be moved out of the field or the data can be transmitted from the display T - trigger = a field that if modified, is transmitted as soon as the cursor is moved out of the field. (This allows the application to receive and modify fields one by one). ACTION_KEY will be 'TRIGGER' for these. N - none Default - N ***Note: At the time of writing, there are no terminals at the TAB which support field validation.***
Vname	The name of a REXX variable. The name must be enclosed by ' (quote) characters. If the variable keyword is specified, the TEXT keyword cannot be specified. (e.g. VNAME = 'Variable.1'). When this keyword is used, RXFSREAD will automatically update the screen with the value of the variable. When the variable is displayed on the screen data may be entered in the field to change the variable. After an RXFSREAD, values of variable fields which have been modified are automatically updated to the values on the screen. Any occurances of the FILL character for a modified field will be replaced by spaces in the REXX variable value that gets updated by RXFSREAD (unless FILL=N). Default - Not Applicable
WINdow (NEW)	Read and parse a CMSPANEL. The file name is is arbitrary and can be anything but 'CLOSE'. This is used to indicate the name of a CMSPANEL which is to be used as a window. All fields added will go into the current window. When a 'WINDOW=CLOSE' command is issued, the current window is closed and the screen underneath it will be re-displayed.
Width (NEW)	Indicates the width (number of columns) of a window.

(unprotected text and variable fields which had information typed into them), and also **SCR_DATA.x** variables are set for each modified field describing the field location, length, and modified contents (x is an integer between 1 and the contents of SCR_DATA.0).

- The **CURSOR_ADDRESS** variable contains a string which tells the position of the cursor when the key was struck to complete the RXFSREAD. The format of this string is "**ROW rr COLUMN cc**", where 'rr' is the row and 'cc' is the column.
- **BCP_TOKEN** is set if the cursor was on a field which was given a token between 1 and 255. The value of the token is put into the BCP_TOKEN variable. It is dropped if the cursor was not on a field which had a token associated with it.
- **PFKEY** contains the PF key number if a PF key was the key that was struck to complete the RXFSREAD. The value of PFKEY will be between 1 and 24. The PFKEY variable will be dropped if a PF key was not struck.
- **PAKEY** contains the PA key number if a PA key was the key that was struck to complete the RXFSREAD. The value of PAKEY will be between 1, 2, or 3. The PAKEY variable will be dropped if a PA key was not struck.
- **ACTION_KEY** contains a description of the key (or other Attention Identifier) that completed the RXFSREAD. *For a list of possible settings of the action_key variable, see "Settings for the ACTION_KEY Variable" on page 15.*

It is possible to change several of the attributes of variable fields before they are displayed by RXFSREAD by using **dynamic attribute settings**. *For information on how to do this, see "Using Dynamic Attribute Settings" on page 11.*

Below is a list of keywords which are valid for RXFSREAD:

Keyword	**Description**
DEVice	Changes the device for which the current page is displayed. The value of this keyword is a character string which represents the name of a device that has been RXFSOPENed (this name is returned by RXFSOPEN when a device is first opened in the **DEVICE_NAME** REXX variable). The current page of the current pageset of the device given is displayed.
NOWAIT	This keyword has no setting (i.e. it appears by itself with no '='). When this is present, it will prevent RXFSREAD from stopping and waiting for a PF Key, PA Key, etc to be struck. RXFSREAD will refresh the screen with the current REXX variable values, and then finish.

Note that if the character sets which are currently loaded are different to the character sets which are loaded for the RXFSOPEN which will become active, then RXFSCLOS will re-load any necessary character sets for the RXFSOPEN which will become active.

Valid keywords for RXFSCLOS are:

Keyword	**Description**
DEVice	Changes the device for which the current pageset closed. The value of this keyword is a character string which represents the name of a device that has been RXFSOPENed (this name is returned by RXFSOPEN when a device is first opened in the **DEVICE_NAME** REXX variable). The current pageset for the device given is closed.

3.5 RXFSPOSN Usage

RXFSPOSN sets up the REXX array **BCP_FIELDS**, which describes the positions of all variable fields on the current page. The format of this array is:

BCP_FIELDS.0 = *the number of variable fields on the page,*
BCP_FIELDS.1 = "*<VName> <Row> <Col> <Length>***",**
BCP_FIELDS.2 = "*<VName> <Row> <Col> <Length>***",**
BCP_FIELDS.3 = ..., and so on.

Any values that were previously held in BCP_FIELDS are dropped. **Note that no arguments are required by RXFSPOSN.**

Note that the RXFS_LEVEL variable is also set by RXFSPOSN to tell the current version of RXFS that is running. RXFS_LEVEL contains a string of the form:

"Version *v.rr.fff* Feb 05 1992 11:13:05"

Where:

v is the current version (1 digit),
rr is the current release (2 digits),
fff is the fix number (3 digits).
The date and time in RXFS_LEVEL is the time that the current level of RXFS was compiled.

3.6 RXFSPURG Usage

RXFSPURG closes all active pagesets (it is the same as RXFSCLOSing every active RXFSOPEN for every device). **RXFSPURG requires no arguments**.

3.7 RXFSDUMP Usage

RXFSDUMP requires the "FILE" keyword to be present in the arguments to it. It produces a file containing a "screen dump" of the current page. The setting of the "FILE" keyword specifies the name of the file into which the screen dump is to be put. The file is of the type **BCPDUMP**.

e.g. "RXFSDUMP FILE=MYSCREEN" would produce MYSCREEN BCPDUMP.

Note that the file produced will have the same dimensions as the terminal on which RXFSDUMP was called. (i.e. If the screen has 80 columns and 32 lines, then the RXFSPSS file produced will have a fixed record length of 80 characters, and it will be 32 lines long).

The image put into the BCPDUMP file is the same image that would be displayed if an RXFSREAD we done. Note that fields which have either "PROTECTION=I" (Invisible) or "DISPLAY=N" (Non-display) are not put into the screen dump file.

RXFSDUMP checks *dynamic attribute settings* (color_..., prot_..., exthi_..., and cursor_...) for each variable field on the screen. This is useful when it is required to make some variable fields not appear on the screen dump by making them invisible (i.e. prot_vname = "I").

Any fields which have extended highlighting of 'U' (Underline), and have either Fill=B or Fill=N are shown on the screen dump file as having FILL=_. This means that any boxes, etc which have been constructed using underlined fields will be shown on the screen dump.

Below is a list of valid keywords for RXFSDUMP:

	of the BCPDUMP file into which the screen dump will be put.

3.8 RXFSHELP Usage

The RXFSHELP program will display a description on the cause of a given RXFS return code. Return codes from all RXFS programs have help available.

The format of RXFSHELP is:

RXFSHELP <*retcode*>

Where, <*retcode*> is the return code for which help is required.

For a list of RXFS return codes, see "Return codes from the RXFS System" on page 16.

4.0 Advanced Use of RXFS

4.1 Using Dynamic Attribute Settings

Once you have declared a variable field (with RXFSWRIT), you are able to change the attributes assigned to the field by changing the value of certain special REXX variables. When these values are changed, RXFSREAD will display the fields with their new settings. Take note that it is not possible to reset the old attributes by simply dropping the special REXX variable.

The special REXX variables are:
"**Color_...**" to change the colour of a field,
"**Exthi_...**" to change the extended highlighting of a field,
"**Prot_...**" to change the protection type of a field.

For example, suppose you code:

```
"RXFSWRIT VNAME='TEST' ROW=1 COLUMN=1 COLOR=BLU EXTHI=N PROT=N"
```

This will cause the value of the REXX variable "TEST" to be shown in the top left corner, with no extended highlighting, blue in colour, and unprotected.

To change any of the three attributes, you (in your exec) code
"Color_TEST=xxx", or, "Exthi_TEST=yyy", or, "Prot_TEST=zzz",
where,
"xxx" is a valid setting for the COLOR keyword,
"yyy" is a valid setting for the EXTHI keyword, and
"zzz" is a valid setting for the PROT keyword.

Also, RXFSREAD will check for a "**CURSOR_....**" variable for each variable field on the page to be displayed. If the value of the "CURSOR_...." is "Y", then the cursor is positioned at the start of the given variable field. This is the same as doing an RXFSWRIT with "CURSOR=Y" for the row and column of the variable field.

See “RXFSWRIT Usage” on page 5 for a summary of the valid settings for COLOR, EXTHI and PROT.

4.2 Using Tokens with Fields

When a field is added to a screen, it is possible to associate a **token** with that field. This means that when RXFSREAD is completed, if the cursor was on a field that had a token associated with it, then the REXX variable **BCP_TOKEN** is set to contain the value of the token of the field which the cursor was on.

This feature is useful for choosing fields using the cursor. For example, there could be a screen with 10 different options on it, with each option having a field to select it (each of these selection fields having a different token. The EXEC will be able to tell which option has been selected by checking the value of BCP_TOKEN. If BCP_TOKEN is undefined, then the EXEC will know that no options have been selected.

Fields can have a token between 1 and 255. If the token for a field is 0, then this is taken to mean that the field does not have a token.

To give a field a token, use the TOKEN keyword with RXFSWRIT.
e.g. To give a TEXT field a token value of 2, use:
"**RXFSWRIT R=10 C=10 L=7 TOKEN=2 T='A Field'**"

If the cursor is positioned anywhere on this field when an RXFSREAD is completed, then the BCP_TOKEN variable will be set to 2. If the cursor is not on a field which has a token (i.e. on a field which has a token of 0, or not on any field), then the BCP_TOKEN variable is dropped.

A virtual device must be defined to be used. The command to do this is:
" **DEF GRAF <*devaddr*>**".

4.3.1.2 *Step 2 - Dial to the GRAF Device.*

Before full screen I/O can be performed on the virtual device which was defined in Step 1, a real device must be connected to it. To do this, another terminal must *dial* the user which is running RXFS.
This is done by typing " **DIAL <*userid*>**" on the "COMMAND" line of the login screen of the terminal.

4.3.1.3 *Step 3 - Open the Device.*

The device is now ready to be RXFSOPENed. The device address which was used to define the virtual device (in Step 1) must be used with RXFSOPEN. e.g. To open GRAF 020, type:
" **RXFSOPEN DEV=020**".

When this is done, the "**DEVICE_NAME**" REXX variable will be set. This name should be noted, since all future references to this device are now made using the name of the device, rather than the address.

All subsequent RXFS commands will now refer to this device (until a different device is used by any RXFS program).

Note that the device address of the terminal is FFFF.

4.3.1.4 *Step 4 - Build Display Screen.*

The device is now ready to be used by the other RXFS programs. All RXFSWRITs, RXFSREADs, etc will be for the new device until one of the RXFS programs specifies a valid DEV keyword for a different device, or until RXFSCLOS closes all RXFSOPENs that are active for the device.

e.g. To display a field on the device that has just been RXFSOPENed:
" **RXFSWRIT R=1 C=1 L=9 T='The Field' DEV=**<*devname*> "
" **RXFSREAD DEV=**<*devname*>"
Where <*devname*> is the DEVICE_NAME string which was set by the RXFSOPEN for the device.

4.3.1.5 *Step 5 - Close The Device.*

When no more full screen I/O is required for the device, RXFSCLOS is called. This will close the fullscreen environment for the device, which will "drop" the dialed terminal. To detach the GRAF device which was defined in step 1, use:
" **DET <*devaddr*>**"
Where <*devaddr*> is the device address given in step 1.

file of type RXFSPSS. Files of the type RXFSPSS define character sets, and must be generated in a special way. *See "RXFSPSS Filetypes" on page 19 for information on how to generate RXFSPSS files.*

A terminal may have several loadable character sets available. The PSA keyword refers to the first character set, PSB the second, and so on. In the case where a filename is given for a character set, but the terminal does not have a character set available, the device-provided (normal) character set is used.

e.g. To open a pageset which will have the first character set loaded with the character set file "LXBOLDC RXFSPSS", the command would be:
" **RXFSOPEN PSA=LXBOLDC**"

Note that more than one RXFSPSS file may be loaded by an RXFSOPEN.

4.4.1.2 Defining fields which use the character set.

To define a field on the screen which will appear in the character set which is loaded into the first character set (PSA), the "char" keyword is used in the definition for that field. The setting of the "char" keyword is the letter of the character set (e.g. for PSA, the setting would be "CHAR=A").
e.g. " **RXFSWRIT R=10 C=1 V='Errmsg' CHAR=A**"

4.4.1.3 Notes

- When an RXFSOPEN is done which loads RXFSPSS files, all subsequent RXFSOPENs for that device will default to having the same symbol sets. If a subsequent RXFSOPEN loads a different symbol set, then when it is closed, any necessary RXFSPSS files are re-loaded so that the original symbol sets are there.
- It is assumed that the device onto which character sets loaded will loaded is capable of supporting the characteristics of the PSS that is loaded. The maximum recommended character size of PSS is 9 x 16.

4.5 Using Detectable Fields

Detectable fields are fields which are selectable with the **cursor select key** or the **selector pen**, subject to the use of a *designator character* The definition of a detectable field includes a "DETECT=Y" keyword.

For a field to be detectable, it must have a *designator character* associated with it. The designator characters are the first character in the field. There are 4 types of designator characters:

- **<space> or <NULL>** - Selection of fields with a SPACE or NULL designator character causes RXFSREAD to complete, and set the REXX ACTION_KEY variable to "**SELPEN_ATTN**".
- **&** - When a field is selected which has an ampersand designator character, RXFSREAD will proceed as though the ENTER key was struck at the current cursor location.
- **>** - Selecting a field with a '>' designator character will cause the designator character will change to a '?'. The field which is selected will **not** be marked as being changed.
- **?** - Selecting a field with a '?' designator character will cause the designator character will change to a '>'. The field which is selected will be marked as being changed. It will be returned in the SCR_DATA array, and if it is a variable field, its value will be updated.

Take note that when fields are selected which have SPACE or NULL designator characters, then the SCR_DATA array will contain the address of all modified fields on the screen, but NOT the correct modified contents. RXFSREAD will think that all modified fields have been deleted, so the values of any modified variable fields will be cleared. **Care should be taken when using detectable fields which have SPACE or NULL designator characters.**

4.6 Using Windows

This version of the new system allows windows to be used. By specifying 'RXFSWRIT WINDOW=panel R=a C=b W=c H=d COLOUR=e B=b will display the cmspanel defined in 'panel CMSPANEL *' (where 'panel' is a CMSPANEL) on top of the existing panel. The keywords R, W, H are required where R & C specify the top left hand corner of where the panel (window) is to be displayed, H specifies the height (number of rows or width) of the window, and W specifies the width (length or number of columns) of the window. B can be one of REVERSE for reverse video borders, LINES for line boerders or NONE for no borders. Note that these window dimensions INCLUDE the borders of the window which are displayed as reverse video highlighting in colour COLOUR. Thus (R + H) & (C + W) specify the bottom right hand then the file "TEST CMSPANEL" defines the fields in the window, and the window borders are red reverse video highlighting with the top border on row 5 (since R=5), the left border on column 8 (since C=8), the bottom border on row 19 (since there 15 rows in the window starting at row 5 = R+H-1 = 19), and the right border on column 17 (since there are 10 columns in the window starting at column 8 = C+W-1 = 17). Thus the top left corner of the window is located at (R,C) and the bottom right corner of the window is located at (R+H-1,C+W-1). There are 13 usuable rows in this window (15 rows minus 2 borders = H-2) and 8 usable columns in this window (10 columns - 2 borders= W-2).

If the file "TEST CMSPANEL" contains the field "R=4 C=6 T='Hello'", then the text 'Hello' would be displayed on row 4, column 6 of the window. This field corresponds to row 9 of the screen (since the window row = 5 including the top window border, and the field row = 4 ie. 5+4=9) and column 14 of the screen (since the window column = 8 including the left window border, and the field column = 6 ie. 8+6=14).

If the file "TEST CMSPANEL" contains the field "R=0 C=-5 T='Bye'", then the text 'Hello' would be displayed on window row 0 which is the same as window row 13 (ie. 13 usable rows in window minus 0). This window row corresponds to row 18 of the screen (since the window row = 5 including the top window border, and the field row = 13 (ie. 13+5=18) or, the bottom window border is on screen row 19 which makes last usuable window row=field row 0 equal to screen row 18). Similarly, the text 'Bye' would be displayed on column -5 which is the same as col 3 of the window (ie. 8 usuable cols in window minus 5) and this corresponds to col 13 of the screen (since the window col = 10 including the top window border, and the field col = 3 (ie. 10+3=13) or, right window border on screen row 17 which makes last usuable window row=field row 0 equal to screen row 16, field row -1 equal to screen row 15, field row -2 equal to screen row 14 and field row -3 equal to screen row 13. All succeeding RXFS commands will apply to the window until a "RXFSWRIT WINDOW=CLOSE" is issued at which point the current window is closed and the screen is redisplayed as it was before the window was opened.

5.0 Appendix

5.1 Settings for the ACTION_KEY Variable

Below is a list of the various settings for the ACTION_KEY REXX variable which is set upon completion of RXFSREAD.

ACTION_KEY Value	Description
PFKEY 1, PFKEY2, ..., PFKEY 24	A **PF Key** was the key which was struck to finish the RXFSREAD.The number of the PF Key that was used follows the "PFKEY" word.
PAKEY 1, PAKEY 2, or PAKEY 3.	A **PA Key** was the key which was struck to finish the RXFSREAD. The number of the PA Key that was used follows the "PAKEY" word.
ENTER	The **ENTER** key was struck to finish the RXFSREAD.
CLEAR	The **CLEAR** key was struck to finish the RXFSREAD. In this case, there will be no REXX variables for modified variable fields updated. The SCR_DATA array will be empty.
TRIGGER	This happens when data has been entered into a **trigger field** and then the cursor is moved out of that field. It will only work on terminals that support *field validation*. At the time of writing, there are no terminals at the TAB which support this feature.
CLEAR_PART	The **Clear Partition** key was used to finish the RXFSREAD. At the time of writing, no terminals at the TAB had Clear Partition Keys.
SELPEN_ATTN	The **Selector Pen** or **Cursor Select** Key was used to select a detectable field which had a space or null designator character. *See "Using Detectable Fields" on page 13 for a full explanation on the use of detectable fields.*
TEST_SYS_REQ	The **Test Req** or **Sys Req** Key was used to finish the RXFSREAD. Note that pressing the Sys Req Key will often cause ACTION_KEY to be set to"OTHER".
OPID_RDR	**Magnetic Stripe Reader (OPerator ID ReaDeR)** was used to finish the RXFSREAD.
MAG_RDR_NUM	**Magnetic Stripe Reader (MAGnetic ReaDeR NUMber)** was used to finish the RXFSREAD.
OTHER	This will occur in a situation where none of the interrupts above are used to finish the RXFSREAD. Examples of situations where this can occur are when the user gets a warning from another user, or when the "SysRq" key is struck on some terminals.

Return Code	Possible Cause
50	Could not open CMSPANEL - The CMSPANEL requested from a 'PANEL' argument to RXFSWRIT could not be opened (generally means file not found).
51	No RXFSOPEN was issued before calling RXFSWRIT.
52	No RXFSOPEN was issued before calling RXFSREAD.
53	No RXFSOPEN was issued before calling RXFSCLOS. There are no RXFSOPENS active for any device.
54	No RXFSWRIT was issued for the current page before calling RXFSREAD (i.e. no fields on current page).
55	Bad arguments for RXFSWRIT. The call to RXFSWRIT had no parameters.
56	A call was made to an RXFS program which attempted to work on a device for which no RXFSOPENs are current. This means that the device given by a 'DEV' keyword is not currently open. There are RXFSOPENs active for other device(s).
57	No RXFSOPEN was issued before calling RXFSPURG. There are no RXFSOPENS active for any device.
58	No RXFSOPEN was issued before calling RXFSPOSN. There are no RXFSOPENS active for any device.
59	No RXFSWRIT was issued for the current page before calling RXFSPOSN (i.e. no fields on current page).
60	No RXFSOPEN was issued before calling RXFSDUMP. There are no RXFSOPENS active for any device.
61	No RXFSWRIT was issued for the current page before calling RXFSDUMP (i.e. no fields on current page).
62	There was an error opening the output file for RXFSDUMP. This is usually caused by the filename given to RXFSDUMP (with FILE=....) being either not present, or invalid.
63	Arguments were given to RXFSPOSN. RXFSPOSN does not require any arguments, so the arguments given were ignored.
64	Arguments were given to RXFSPURG. RXFSPURG does not require any arguments, so the arguments given were ignored.
65	Invalid PSS Filename. The filename given in either the PSA, PSB, PSC, ..., or PSH keyword was not a valid filename. The filename of the RXFSPSS file to be loaded as a Programmed Symbol Set (character set) is expected.
66	RXFSPSS File Not found. The file containing a preprocessed programmed symbol set (filetype RXFSPSS) could not be found. This occurs when a PSA, PSB, ..., or PSH keyword is given to RXFSOPEN, but the filename given could not be found. This situation is only detected if there is a loadable PSS available on the current terminal.

Return codes in the range 201 - 400 indicate a problem in a panel definition, or in the arguments to a program. The problem is caused by bad keyword.

Return Code	Possible Cause
201	Unrecognisable keyword. There was a keyword specified in the CMSPANEL, or in the arguments to an RXFS program which could not be which could not be recognised. Examples of keywords are ROW, PANEL, EXTHI and COLOR.
202	Keyword given was too long. The given keyword was longer than the maximum length allowed (currently 50 characters).
203	Keyword setting was too long. The value of a keyword setting was longer than the maximum length allowed (currently 300 characters).
204	Arguments too long. The length of the arguments given exceeded the maximum length allowed (currently 450 characters). No arguments will be processed.

Return codes in the range 401 - 600 indicate a problem in a panel definition, or in the arguments to a program. The problem is caused by a bad initialisation for a keyword.

	'LENGTH' keyword. The length for the field in which this happened is taken to be the number of characters specified in the 'TEXT' or the 'VNAME' initialisation for the field.
404	Bad Value for COLOUR. This happens if a the colour given is not recognised as a valid colour. The colour of this field is set to the default (turquoise).
405	Bad Value for VNAME. This happens if the quotes around the initialisation for 'VNAME' are not present, or not closed.
406	Bad Value for TEXT. This happens if the quotes around the initialisation for 'TEXT' are not present, or not closed.
407	Bad Value for ALARM. This happens if the value for the 'ALARM' keyword is not Y or N. 'ALARM=N' was assumed.
408	Bad Value for CURSOR. This happens if the value for the 'CURSOR' keyword is not Y or N. 'CURSOR=N' was assumed.
409	Bad Value for PROTECTION. This happens if the value for the 'PROT' keyword is not Y, N or I. 'PROT=Y' was assumed.
410	Bad Value for EXTHI. This happens if the value for the 'EXTHI' keyword does not begin with U, B, R or N. 'EXTHI=N' was assumed.
411	Bad Value for FILL. This happens if what appears after a 'FILL' keyword is more than a single character long. If this happens, 'FILL=N' (NULL fill character) will be assumed.
412	Bad Value for SKIP. This happens if the value for the 'SKIP' keyword is not Y or N. 'SKIP=Y' was assumed.
413	Bad Value for TYPE (or CASE). This happens if the value for the 'TYPE' (or 'CASE') keyword is not U, M or N. 'TYPE=M' (or 'CASE=M') was assumed.
414	Bad Value for PAGE. This happens if the value given for the 'PAGE' keyword was not 0, 1, ..., 9. The current page remains the same if this happens.
415	Bad Value for CLEAR. This happens if the value for the 'CLEAR' keyword is not Y or N. 'CLEAR=N' was assumed.
416	Bad Value for VALIDATION. Valid settings for the 'VALIDATION' keyword are EITHER 'VALIDATION=N', or 'VALIDATION=xxx' where xxx is a combination of E, F, and T. 'VALIDATION=N' was assumed.
417	Bad Value for DETECTABLE. This happens if the value for the 'DETECT' keyword is not Y or N. 'DETECT=Y' was assumed.
418	Bad Value for DISPLAY. This happens if the value for the 'DISPLAY' keyword is not N, I or D. 'DISPLAY=D' was assumed.
419	Bad Value for TRANSPARENT. This happens if the value for the 'TRANSPARENT' keyword is not Y or N. 'TRANSPARENT=Y' was assumed.
420	Bad Value for OUTLINE. Valid settings for the 'OUTLINE' keyword are EITHER 'OUTLINE=N', or 'OUTLINE=xxxx' where xxxx is a combination of O, U, L and R. 'OUTLINE=N' was assumed.
421	Bad Value for CHARACTER. This happens if the value for the 'CHARACTER=' keyword is not 0, or A, B, ..., H. 'CHARACTER=0' was assumed.
422	Bad Value for DEV. This will happen in several situations: • If the value given for the 'DEV' keyword in RXFSOPEN is not a valid hexadecimal value (upto 4 hex digits). • If the value given for the 'DEV' keyword in RXFSWRIT, RXFSREAD, or RXFSCLOS is not a valid device name (i.e. 1-8 characters).
423	Invalid variable name. This will happen if the name specified for a variable field (in RXFSWRIT) is not a valid REXX variable name.
424	Bad token value. The value for a 'TOKEN' keyword must be an integer in the range 0..255. A 'TOKEN=0' was assumed (i.e. it is assumed that no token is associated with that field).

Return codes in the range 601 - 800 indicate that there were conflicting keyword settings present in a CMSPANEL, or in the arguments to RXFSWRIT.

	arguments to RXFSWRIT. It is not processed within CMSPANELs, and it is ignored.
604	Value for COLUMN is off the screen. Wraparound has occurred.
605	Value for ROW is off the screen. Wraparound has occurred.
606	'ROW' Keyword was specified, but 'COLUMN' keyword was not. A field must have a Column AND a Row. Fields with this problem are ignored.
607	'COLUMN' Keyword was specified, but 'ROW' keyword was not. A field must have a Column AND a Row. Fields with this problem are ignored.
608	A 'TEXT' Keyword was present, but neither a 'COL' or a 'ROW' keyword was given for that field. Fields with this problem are ignored.
609	A 'VNAME' Keyword was present, but neither a 'COL' or a 'ROW' keyword was given for that field. Fields with this problem are ignored.
610	More than one 'TEXT' keyword was given for a field. TEXT can only be specified once for any field. The first value given is used.
611	A 'TEXT' keyword was given for a field after a 'VNAME' keyword was given. TEXT cannot be specified for a VNAME field. The first value given is used.
612	A 'VNAME' keyword was given for a field after a 'TEXT' keyword was given. VNAME cannot be specified for a TEXT field. The first value given was used.
613	More than one 'VNAME' keyword was given for a field. VNAME can only be specified once for any field. The first value given was used.
614	The length that was specified for a 'TEXT' field was not big enough to hold the initial value (e.g. initialising a field with a length of 5, and giving it the TEXT value of 'some text'). The length of this field is taken from the LENGTH setting (unless it's not valid).

Return codes in the range 801 - 1000 are for general problems.

Return Code	**Possible Cause**
801	Overlaps Occurred. The last RXFSWRIT issued defined fields that overlap with other fields previously added (either from previous RXFSWRITs, or from earlier field definitions in the CMSPANEL which is being processed). When a field is added to a page, all fields which it overlaps with are removed from that page.
802	Numeric Field Error. This occurs when non-numeric data is entered into a field which is TYPE=N. Valid characters in numeric fields are '0', '1', ..., '9', ',', '.', and ' '.
803	Insufficient Virtual Storage. This can be caused by several factors including: • Too Many programs loaded into nucleus extension (using NUCXLOAD). • Too many pages in memory. Each panel that is used takes up virtual memory. Excessive buffering of pages can cause this memory to fill up. Possible Solutions: • Re-Ipl CMS to remove any 'garbage' from virtual memory, or, • Increasing virtual storage (if authorised).
804	No Device Currently Connected. A path to the requested virtual device has been defined, but no real device is currently connected to that device. This means that the device has not been dialed from a terminal.
805	No Device Currently Defined. The path to the requested virtual device was either never opened, or it has been detached.
806	The virtual device is not supported for full-screen I/O.
807	Retry Limit Exceeded. The maximum number of retries for terminal I/O was exceeded. This is often caused by attempting to load a Program Symbol Set (character set) that is incompatible with the current terminal.
808	Too many fields were on the screen. The buffer used by RXFSREAD to display the current screen filled up because there were too many fields on that screen. Some fields on the screen were not shown.

(such as XEDIT) to build up a file. Each line of a CMSPANEL contains a definition for a field which will be added to the current page. Lines which are marked with an asterisk (*) in the first column are ignored.

CMSPANEL files may define fields for several pages on the one pageset (by using the PAGE keyword). This means that several different screens can be defined within a single CMSPANEL.

Field definitions in CMSPANEL files use the same keywords as a field definition from the RXFSWRIT arguments, although the **PANEL** and **CLEAR** keywords are not valid with CMSPANELs.

For more information on field definitions see “RXFSWRIT Usage” on page 5.

5.3.2 BCPDUMP Filetypes

Files of the type **BCPDUMP** are the output of the **RXFSDUMP** program. These files contain a screen dump image.

For a full description of the purpose of BCPDUMP files, see “RXFSDUMP Usage” on page 9.

5.3.3 RXFSPSS Filetypes

RXFSPSS files define *programmed symbol sets* (device-loadable character sets). They are the files which define the character sets which are are loaded in by RXFSOPEN.

RXFSPSS files are the output of SYMASM EXEC. There are several steps involved in creating a RXFSPSS file:

1. Generate a new Symbol Set, or use an existing one using the "ADMISSE" program on the GDDM disk (accessed by typing "GDDM ON"). To use "ADMISSE", you must be on a graphics screen.
2. Create an ADMDECK file for the symbol set using option 3 ("Create Object Deck") on the main menu of ADMISSE.
3. Use "SYMASM EXEC" to translate the ADMDECK file into an RXFSPSS file.

For information on how to use RXFSPSS files and character sets, see “Character Sets” on page 13.

alarm	a , al , alarm	
character set	ch , char , character	c
clear	cl , clear	c , clr
colour	clr , colour , color	c , co , col , cl
column	c , co , col , column	cl
cursor	cu , cur , curs , cursor	c
detectable	d , det , detect , detectable	
device	dev , devc , device	d, de
display	di , dis , disp , display	d
extended highlighting	e , ex , ext , exth , exthi , exthilight	
filename (RXFSDUMP)	file , filen , filename	f , fi , fil , fn
fill	f , fil , fill	
length	l , len , length	
outlining (field)	o , out , outl , outline , outlining	
page	pa , pg , page	p
panel	p , pan , pnl , panel	pa
protection	pr , prot , protect , protection	p
PSS Filename (PSA,...,PSH)	psa , psb , psc , psd , pse , psf , psg , psh	
row	r , ro , row	
skip	s , sk , skip	
text field name	t , txt , text, tname	
token for field	to, tok , token	t , tokn
transparency	tr , trans , transpar , transparent , transparency	t
type	ty , type, ca, case	t
validation (field)	va , val , valid , validation	v
variable field name	v , var , vname	va

is now the default (previously, the default was FILL=_). The advantage of using this is that it now allows the insert key to be used on fields.

- Negative row numbers (e.g. ROW=-1) are now ALWAYS shown as being from the bottom of the screen, no matter what the size of the screen. The old version would appear to do this also, but if the screen had more than 32 rows, any negative row numbers were taken to be from line 32.
  e.g. On a 43 line screen, with "RXFSWRIT R=-1 C=1 T='Hello'",
  the old programs would display 'Hello' on line 31,
  but the new programs will display it on line 42.

- The new system has return codes to assist in debugging of programs that use RXFS. The old system generally didn't return anything of usefulness. The new system will make it much more obvious if there is a problem with panels, commands, etc.

- Numeric fields (TYPE=N) are now supported. This means that if non-numeric data is entered into a field which has TYPE=N, a return code will be sent to REXX from RXFSREAD. Numeric characters are: '0', '1',..., '9', ',', '.', '-', '+' and ' '.

- Upon issuing an RXFSOPEN, REXX variables are set which tell the current screen size. These variables are:

```
SCR_WIDTH  ... Number of columns on the terminal,  and
SCR_HEIGHT ... Number of rows on the terminal.
```

- Certain things would cause RXFSREAD to return undocumented ACTION_KEY variables (e.g. receiving a warning during an RXFSREAD). The new system returns far more meaningful variables when unusual interrupts such as this occur.

- When using the new system on unusual screen sizes, the routines now work (e.g. 132 col x 27 rows now shows screens properly. The old system would not work propely on these screens).

- Upper case fields (TYPE=U) will display their initial values in upper case (in addition to converting the values in them to upper case). e.g. if the a REXX variable has the value of "abcd", and it has a variable field on the screen, then when an RXFSREAD is done, the value of this variable will appear as ABCD.

- There are several new programs in the new system (RXFSPURG, RXFSPOSN, RXFSDUMP).

- Loadable character sets are now supported (on graphics screens). *See "Character Sets" on page 13.*

- The RXFS system can now be used to control full-screen I/O on devices other than the current terminal. *See "Using Devices Other than the Console" on page 12.*

- Cursor positioning (for RXFSREAD) can now be done by setting the "CURSOR_..." REXX variable to contain the name of the variable field on which to position the cursor. RXFSREAD checks this variable in a similar way to COLOR_, PROT_, and EXTHI_.

6.0 Related Information

Below is a list of other information that is relevant to the RexX FullScreen (RXFS) system:

- Original design for re-write: BCPDESGN SCRIPT on PSS 192 Disk.
- System Documentation for re-written system (???????? SCRIPT, which resides on ????????????).
- IBM 3270 Information Display System - Data Stream Programmer's Reference, Sixth Edition (Manual Number GA23-0059-05).
- This document is called ???????? SCRIPT and resides on ???????????.

closuse	RXFSUS	9	RXFSCLOS Usage	2
posnuse	RXFSUS	9	RXFSPOSN Usage	3
purguse	RXFSUS	9	RXFSPURG Usage	
dumpuse	RXFSUS	9	RXFSDUMP Usage	3, 19
helpuse	RXFSUS	10	RXFSHELP Usage	
attrset	RXFSUS	11	Using Dynamic Attribute Settings	8
tokens	RXFSUS	11	Using Tokens with Fields	
devices	RXFSUS	12	Using Devices Other than the Console	4, 21
charset	RXFSUS	13	Character Sets	4, 19, 21
detflds	RXFSUS	13	Using Detectable Fields	5, 15
windows	RXFSUS	13	Using Windows	
actnkey	RXFSUS	15	Settings for the ACTION_KEY Variable	8
retcode	RXFSUS	16	Return codes from the RXFS System	10
panels	RXFSUS	19	CMSPANEL Filetypes	5
bcpdump	RXFSUS	19	BCPDUMP Filetypes	
pssform	RXFSUS	19	RXFSPSS Filetypes	13