Rtfs Version Six- Revision History

Releases are identified by:

- Version major.minor.revison
- Major version number – Starting at zero the minor release number is incremented when new functionality is added.
- Minor release number – Starting at zero the minor release number is incremented when new functionality is added.
- Revision - Starting at ‘a’, the revision is incremented with each release. The revision history below provides detailed descriptions of each change made for each revision.

Version 6.2.f

Release date: August 2012.

Major changes:

Added #define INCLUDE_REVERSEDIR to conditionally include reverse directory scanning to cut down on code size if RDIR is not needed.

Added pc_rtfs_register_poll_devices_ready_handler() API call and rtfs_devi_poll_request_vector type data type. Device drivers supporting removable media must call this routine to register a subroutine to poll the ready state of the media. (See drwindev.c for an example)

Bug fixes:

Version 6.2.e

Release date: August 2012.

Major changes:

Stress-tested write support on exFAT volumes and made corrections.
Added support for EXFAT in the Failsafe and Rtfs ProPlus modules.
Re-implemented extended attributes and data start offsets for circular file extract.
Added a basic CheckDisk utility for exFAT.
Created a new define INCLUDE_FAT64OREXFAT which branches to code that supports EXFAT and will support the EBS FAT64 proprietary 64 bit format when completed.
Added new features to circular file extract.
Rewrote some portions of the circular file extract and linear extract to support FAT and N=non-FAT formats.

Bug fixes:

All bug fixes for this release have a comment block containing the words “July 2012”, search for this pattern to identify them.

These problems were identified that effect FAT and exFAT releases.
==========
prfsjournal.c – Several bug fixes in the code when the library is configured for (user_index_buffer_size_sectors > 2).
Prfsjournal.c – Bug fix in code that was improperly setting the freespace in every frame and causing the restore module to mistakenly identify a freespace mismatch.
pc_unicode_validate_filename() – Fixed bug that wrongly considered a leading space in a filename an error. Same fix was applied to ASCI and UNICODE in an earlier realease but the change was missed for Unicode.

These problems were identified that effect exFAT only.
==========
_pc_diskflush() – added missing call to flush for exFAt volumes.
fatop_page_get_frag() – Fixed error in exFAT support detecting end of chain.
Exfatformat.c - Fixed error that was under-calculating bam size cfor small volumes.

pc_realloc_fat_blk()- Fixed code to swap BAM sectors properly. Previously write support would corrupt the FAT if the BAM was not fully buffered.
Rtexfatbam.c - Fixed several problems in exFAT write support
rtexfat_i_dskopen –Fixed problem with undercounting free sectors.

Version 6.2.d

Release date: August 2012.

Major changes:

Corrected error in VFAT alias file name creation routine pc_multi_get_entry_number() that caused the Rtfs to produce duplicate 8.3 aliases for certain VFAT file names under certain conditions.

Note: The regression test, tests for this condition and identified it. It is not clear why the problem has not been caught in earlier releases when the regression test was ran. The error is position dependent so the layout the test table in earlier version may have had some effect.

The following filenames all produce the same alias value _~1.
“;”,’[“,”]” and “,”

It was testing the wrong buffer when checking for more than one tildeesulted in duplicate alias names being created when more than 512 files are found in the file.

Detailed changes:
In rtfsvfatrd.c at approximately line 500:
Change

```
      for (j = i; j < 8; j++)
```

To:

```
      for (j = i; j < 8 && alias_image[j]; j++)
```

Version 6.2.c

Release date: July 2012.

Major changes:

Added mutex protection when allocating user structures. The critical section semaphore can't be used so a new semaphore was added.

Corrected error in VFAT alias file name creation routine pc_scan_alias_map() when it is configured to use a bit rather than a byte map for used alias names. The error resulted in duplicate alias names being created when more than 512 files are found in the file.

Detailed changes:

Rtfstypes.h

Added a userlist_semaphore element to struct rtfs_cfg

Rtkernfn.c

Changed rtfs_resource_init to create the new userlist_semaphore mutex.

Changed rtfs_get_system_user to use the userlist_semaphore when allocating a user structure.

Rtvfatwr.c.

Corrected error in VFAT alias file name creation routine pc_scan_alias_map() at line 998.

```
if (!(*p & b))
{
   /* Changed from July 2012  return(base+i*8+j);
      Previous version had a logic error at 512 files*/
   return((base+i)*8+j);
}
```

Version 6.2.b

Release date: April 2012.

Major changes.

.. Fixed bug in which Rtfs stripped leading spaces from VFAT file names.

.. apiregrs.c – Fixed test expectations for file creation test with a leading space, and added a new test to verify the behavior when trailing spaces are present.

.. apisetwd.c – Fixed pc_setcwd() was skipping path components whose first character was a space.

.. Fixed pc_mpath bug. pc_mpath was stopping processing of a path component as soon as it hit a space. This caused pc_enumerate to loop endlessly when it came upon a path component with a leading space.

.. Added buffer overwrite protection to pc_mpath. pc_mpath is not used in many places but it did lack buffer overwrite protection. Code was added to limit the amount of data copied to the buffer to EMAXPATH_BYTES, the buffer size used by all code using the routine.

.. rtvfatrd.c – Removed source code from pc_parsepath that stripped leading white space from filenames.

.. Fixed bug in fatop_page_continue_check_freespace. If the number of sectors in the fat are 1 sector larger than a multiple of buffer_size_sectors, the last sector in the FAT is ignored.

a multiple of buffer_size_sectors, the last sector in the FAT is ignored..

Version 6.2.a

Release date: March 2012.

Major changes.

.. Removed FAT64 variants
.. Added EXFAT conditional include sections, disbled or AI releases.
.. Added New functions to perform reverse directory scans. (pc_glast, pc_gprev).
.. Added WINIMPORT and WINSCAN command shells to import Windows sudirectories into an Rtfs volume.
.. Updated timestamping to support XP specifications.
New shell commands:
.... HACKWIN7   - Toggle making WinDev device writable on Vista and later versions of Windows.
.... SAVESECTORS and RESTORE do raw sector save and restore block copies from an RTfs device to a disk file. Use to save and restore Vendor formatted media.

.... WINSCAN - Scan a Windows subtree and caculate the Rtfs volume size required to contain it. (source code for the actual command are in the windows hostdisk device driver).
.... WINIMPORT - Copy a Windows subtree to an Rtfs volume. The Rtfs volume may be used for experimentation or burned into rom for rom disk applications.
.... RDIR - Perform a directory scan using backwards enumeration. Displays most recently created directory enteries first.
.... FILLHUGEFILE - Performs a patterned write to a file which may be very large.
.... READHUGEFILE - Performs a read a a patterned file, which may be very large.

Detailed changes.

Merge verified AI provided changes since 10-01-2009

rtfs.h - Conditional section for ExFat and flash manager.
rtfsarch.h - Condition compilation change from INCLUDE_RTFSPROPLUS to INCLUDE_MATH64
rtfsarch.h - Condition compilation change from INCLUDE_RTFSPROPLUS to INCLUDE_MATH64
rtfsdevio.h - Change, non-interchagability of BOOLEAN ond int.
rtfsconf.h -
.. added #define INCLUDE_EXFAT, INCLUDE_FLASH_MANAGER
.. added EXRAM definition, use to place static memory buffers into external ram on microcontrollers.
.. removed #INCLUDE_FAT64
.. removed EXTENDED_ATTRIBUTES option.
.. added #define FINODEFILESIZE(f) f->fsizeu.fsize
/* File size is a Union wtih 32 bit values for FAT and 64 bit for exFAT, to support FAT and exFAT for ProPlus and failsafe accesses are done using this macro */
rtfserr.h - Removed PE64NOT64BITFILE and PERESOURCEFINODEEX64, added PERESOURCEEXFAT
rtfsprotos.h - Changes to support EXFAT and removed FAT64, added pc_get_volume,pc_set_volume,pc_glast, pc_gprev.

... Added RTFS_CBS_GETEXFATBUFFERS,RTFS_CBS_RELEASEEXFATBUFFERS, RTFS_CBS_UTCOFFSET and , RTFS_CBS_UTCOFFSET and RTFS_CBS_10MSINCREMENT.
... Added the following prototypes for new functions.
.... BOOLEAN pc_get_volume_cs(byte *driveid, byte *volume_label,int use_charset);
.... BOOLEAN pc_set_volume_cs(byte *driveid, byte *volume_label,int use_charset);
.... BOOLEAN pc_glast_cs(DSTAT *statobj, byte *name, int use_charset);
.... BOOLEAN pc_gprev_cs(DSTAT *statobj, int use_charset);
.... BOOLEAN pc_get_volume(byte *driveid, byte *volume_label);
.... BOOLEAN pc_set_volume(byte *driveid, byte *volume_label);
.... BOOLEAN pc_glast(DSTAT *statobj, byte *name);
.... BOOLEAN pc_gprev(DSTAT *statobj)
..... BOOLEAN pcexfat_format_volume(byte *path);
..... remove FAT64 prototypes accept for ddword pc_efilio_lseek64(int fd, ddword offset, int origin);
..... changed function BOOLEAN _pc_efinode_chsize(FINODE *pefinode, dword new_size);

rtfstype.h -
.. removed FAT64 declarations.
.. Added EXFAT declarations.
.. Changed finode structure.
.... File size is now a union.
.... removed reserved fields, replaced with creation and last accessed time and date values.
.... Changed ddrive_info structure voume_label filed to accomidate 22 byte UNICODE value
.. Changed ddrive structure.
.... Added partition type field to ddrive.
.. Changed dstat structure.
.... Support creation date and traverse directory backwards.

appcmdfs.c -
.. Using global variable fs_flush_behavior to configure Failsafe flush behavior.
appcmdshformat.c
.. Added conditional call to exFATformat

.. Added dohackwin7() command, toggles the ID field in the MBR to enable writing a HostDev devices unders Windows Vista and later versions.

appcmdshrd.c -

.... Added menus entries for

.... EXFATFORMAT - Foprmat exFAT volume.

.... HACKWIN7 - Toggle making WinDev device writable on Vista and later versions of Windows.

.... SAVESECTORS and RESTORE do raw sector save and restore block copies from an RTfs device to a disk file. Use to save and restore Vendor formatted media.

.... WINSCAN - Scan a Windows subtree and caculate the Rtfs volume size required to contain it. (source code for the actual command are in the windows hostdisk device driver).

.... WINIMPORT - Copy a Windows subtree to an Rtfs volume. The Rtfs volume may be used for experimentation or burned into rom for rom disk applications.

.... RDIR - Perform a directory scan using backwards enumeration. Displays most recently created directory enteries first.

.... FILLHUGEFILE - Performs a patterned write to a file which may be very large.

.... READHUGEFILE - Performs a read a a patterned file, which may be very large.

.. Integrated exFat support in certain areas and perfromed fug fixes to directory and flie cluster traversal routines.

... Added code for importing files from a windows directory and reading and writing huge files..

appcmdshwr.c -

... Bug fix in dorm(), use 8.3 file name if lfn is not present.

... Added large buffer support under linux and windows to speed copies of very large files.

... Added copymetadata only conditional code for testing on very large files without having to tranfer all data.

... Added code for importing files from a windows directory and reading and writing huge files..

prfstest.c -

... Added test to more agressively test file wrap conditions.

apickdsk.c - Removed stack test, made minor changes to code to conform to some internal API changes.
apideltr.c - ExFat code related changes only.
apidirent.c - Made minor changes to code to conform to some internal API changes.
apidirent.c - Made minor changes to code to conform to some internal API changes.
apidiskinford.c - ExFat code related changes only.
apifilmv.c - Preserves create time, and access time fileds in directory entry, plus ExFat code related changes and minor internal API call changes.
apifrmat.c - Minor declaration change.
apigetcwd.c - ExFat code related changes only.
apigfirst.c - Added support for reverse directory traversals. pc_glast(), pc_gprev().
apigread.c - ExFat code related changes only.
apigread.c - ExFat code related changes only.
apiregress.c - Factored in exFat changes and addition test scenarion, "comprehensive file IO test."
apisetattr.c - Made minor changes to code to conform to some internal API changes.
apisetvol.c - ExFat code plus changes to support Unicode volume names. Made minor changes to code to conform to some internal API changes.
apisetwd.c - ExFat code related changes only.
apistat.c - Added support for creation and access time. Also made changes or internal APIs and ExFat.
apiunlink.c - Removed FAT64 support.
apiunlink.c - Removed FAT64 support.
drdynamic.c - Changed algorithm for default volume size to fix a problem of Windows reporting too large media size.
... ??? Added supprt for dyn_partition_type field..
... Added code to protect against a race condition casued by dismount processing occuring before mount processing completed.
prbasicemurd.c - Added po_lseek64() API call to extend Rtfs API for 64 bit exfat files.
rtblockrd.c - Changed block enumerator for short to long
rtdrobjrd.c -
rtdrobjwr.c -
.. Remode FAT64 code

.. Removed extended attributes.
.. Support for access and creation time.
.. Added support for backward directory traversal.
.. Updated to internal APIs.
.. Added Exfat support.
rteraseblock.c -
.. One bug fix and some minor changes
rtfatdrvrd.c - Logic changes to support exFAT, also integrates the changes supporting cluster masking and correct end of chain marker use.
rtfatdrvwr.c - Logic changes to support exFAT, also integrates the changes supporting cluster masking and correct end of chain marker use.
rtfblockrd.c - Logic changes support passing buffer structure seperately for the drive so the buffer can be shared between FAT and exFAT modules.
rtfragmtrd.c - Added abstracted pc_grow_basic_fragment() and pc_fraglist_alloc_frag_clipped() routines that are shared by FAT and exFAT.
rtfsbasicrd.c - Support for last accessed time stamp 64 bit file seek and necessary internal API changes to support exFat.
rtfsbasicwr.c - Algorithm change for last modified time stamp, latching the time value when the changed occured, not when the flush occurred. Plus necessary internal API changes to support exFat.
rtfsgluerd.c - Updated timestamp algorithm, added exFat stub functions.
rtfsgluewr.c - Updated timestamp algorithm.
rtfsgluewr.c - Updated timestamp algorithm.
rtfskern.c   - Minor changes plus exFat support.
rtlowl.c
.. New code loads the partition type from the value stored in the device sruture when the device was inserted.
.. New code recognizes the purposely modified MBR signature created by HACKWIN7 and allows the mout to proceed with the invlid MBR signature.
.. New code recognizes exFAT and calls exFAt mount routines.
.. New code recognizes exFAT and calls exFAt mount routines.
.. Changed pc_sec2index() routine to return a 32 but value.
.. Logic change in auto fialsfe support.
rtutil.c -
.. Include 64 bit math routines, previously included with Rtfs Pro plus only.

drwindev.c - Has a lot of changes, the driver underwent restructuring has been used heavilly in this update cycle so changes well tested. Recent fixes reported by AI are integrated.

drhostdisk.c -

.. In addition to host disk support this file now includes support for new shell commands that rely on Windows file system access.

... Dumping dumping raw blocks from an Rtfs device to a WIndows file.

... Restoring raw blocks to an Rtfs device from a Windows file.

... Importingfiles and subdirectories from a windows Volume to an Rtfs volume.

prfstrio.c - Uses macros to access the file size field.

rtfspackages/apps -

... All files in this subdirectory are unchanged except to removes FAT64 support.

rtfspackages/rtfsproplus -

... prefi64.c - File was removed.

... All other files were modified to remove FAT64 support, in some case FAT64 conditional inclusion is replaced by EXFAT based inclusion, but the exFAT based logic is not complete.

rtfspackages/rtfsproplusdvr -

... Files were modified to remove FAT64 support, in some case FAT64 conditional inclusion is replaced by EXFAT based inclusion, but the exFAT based logic is not complete.

Sep 19, 2011 - Updated Conexant and Alps electronics.

===

Fixed an exFAT bug caused a failure on media with large cluster sizes over 32768 K sectors per cluster or more. Changed the secpalloc field in driveing form word to dword.

===

Fixed an exFAT bug that resulted in duplicated files when a file was renamed.

The directory entry is cloned but the original is not deleted and thus they are duplicated.

The problem is with pcexfat_mvnode(), I changed calls to pcexfat_update_by_finode() when I added more precise time stamping and something slipped through the cracks.

rtexfatdirwrite.c: approximately lines 756:775

The solution is to change:
        ret_val = pcexfat_update_by_finode(old_obj->finode, old_obj->finode->s.segindex, FALSE, FALSE, 0);
to:
        ret_val = pcexfat_update_by_finode(old_obj->finode, old_obj->finode->s.segindex, FALSE, 0, TRUE);
===

Version 6.1.e

Release date: June 2009.

include\rtfsarch.h – Add support for #ifdef _LINUX
include\rtfsconf.h - Change EMAXPATH_CHARS from 260 to 255.
rtfscommon\source\apifrmat.c - Changed _pc_calculate_fat_size_sectors algorithm to eliminate formatting errors on certain media sizes at the boundaries of FAT12 and FAT16 sizes.
NOTE: - Needs editing.. and comparison with version1.0
rtfscommon\source\rtfsbasicwr.c- bfilio_chsize() - Make sure errno is clear if the function succeeded.
rtfscommon\source\rtfatdrvwr – Changed _fatop_find_contiguous_free_clusters() to set errno to PENOSPC.

   rtfscommon\apps\appcmdfs.c   - Minor cleanup.
    rtfscommon\source\apideltr.c - Make sure errno is clear if the function succeeded.

rtfscommon\source\rtnvfatrd.c - Added maximum length test for path names to guard against.
rtfsfailsafe\prfsjournal.c   - Fix so can build without INCLUDE_RTFS_FREEMANAGER
rtfsfailsafe\prfsjournal.c   - Fixed bug that was causing a drop in the apparent file size by one sector each time the journal wrapped the file at exactly the last sector of the journal.
rtfsfailsafe\prfsnvio.c   - Fix so can build without INCLUDE_RTFS_FREEMANAGER

rtfscommon\source\apifilmv.c - Fixed bug processing ".." that caused moving a directory to fail when using unicode API.

rtfscommon\source\apiinit.c - Add code to autoformat ram disk.
rtfscommon\source\apimkdir.c - Make sure errno is clear if the function succeeded.
rtfscommon\source\apirun.c - Removed compiler errors when building with Failsafe.
rtfscommon\source\apiunlink.c- Make sure errno is clear if the function succeeded.
rtfscommon\source\rtfatdrvrd.c-Changed to recognize >= 0x0ffffff8 not >= 0xffffff8 as FAT32 eof
rtfscommon\source\rtfsbasicwr.c- _pc_bfilio_write() - Add support for RTFS_MAX_FILE_SIZE limit
rtfsdrivers\ramdisk\drramdsk.c - Modified configuration method so HCN values are always legal.
rtfsfailsafe\prfsapi.c - Minor change does not effect operations
rtfsfailsafe\prfscb.c - Minor change does not effect operations
rtfspro\rtfat32rd.c - Minor bug fix do not assert error, just fix it if the FAT32 info block is uninitialized.
rtfspro\rtvfatrd.c - Added maximum length test for path names to guard against.
rtfspro\rtvfatrd.c - Added test double backslashes in the path name.
rtfspro\rtvfatwr.c - BUG FIX !! Removed bug that could cause adjacent file name segments to
be corrupted if DOS inode is at the beginning of a sector the number of
segments in the LFN is >= 16.

Version 6.1.e

Release date: June 2009.

The following sections are provided:

Read me first - This describes the changes that are provided with this release and describes changes to project files and new tests and shell commands that are available with this release.

This release description contains four sections.

- Read me first
- Major functional changes - Bulk description of code changes.
- Glossary - A list of categories of code changes made in the release.
- File Change List - A list of files that have been changed.

Read me first

This release can be configured to build the following libraries:

Major functional changes

Modified Rtfs ProPlus test code and Rtfs Failsafe Test code to build and execute with the new run time configuration and device driver attachment methodologies introduced in the previous release.
Modified Failsafe Test code so it can build and execute with RtfsPro. Previously the test could run only if RtfsProPlus was enabled.
Added Frame level buffering for Failsafe. This improves performance, especially when using nand flash.

Glossary

The following "tags" describe reasons for changes between the latest source files from EBS and the previous release. The file change list below lists files that are different and with tags identifying why the files were changes.

<FSINDEXBUFFER> - Frame level buffering for Failsafe. Journaled data may be buffered in memory until the frame is flushed.
<CLEANUP> - Code clean up, removal of prototype code, routine edit, or eliminate obolete conditional compilation.
<TEST> - Code modifications or conviguration changes made to enable the test environment

<MOVEF> - To run the Failsafe test code from the standard release some source code was rearranged and moved from the rtfspackages\apps subdirectory to the common\apps subdirectory.
<MOVEH> - To run the Failsafe test code from the standard release some source code was rearranged and moved from the rtfspackages\apps subdirectory to the common\apps subdirectory.
<BUGFAILSAFE> - A bug fix but only if Failsafe is being used.
<BUGPROPLUS> - A bug fix but only if ProPlus is being used.

File Change List - The following files have changed or moved.
==========

- **rtfscommon\include - (<MOVEH> All files were moved from src\rtfscommon\include to src\include)**
- **include\fsadaptmem.h - (<FSINDEXBUFFER> <MOVEH>)**
- **include\rtfs.h - (<CLEANUP> <MOVEH>)**
- **include\rtfsblkmedia.h - (<FSINDEXBUFFER> <MOVEH>)**
- **include\rtfsconf.h - (<TEST>, <CLEANUP> <MOVEH>)**
- **include\rtfsfailsafe.h - (<FSINDEXBUFFER>, <CLEANUP> <MOVEH>)**
- **include\rtfsprotos.h - (<CLEANUP> <MOVEH>)**
- **include\rtfstypes.h - (<FSINDEXBUFFER>, <CLEANUP> <MOVEH>)**
- **include\rtfsversion.h - (<CLEANUP> <MOVEH>)**
- **rtfstargets\dspbios\portkern.c - (<INIT>)**
- **rtfscommon\apps\appcmdfs.c - (<TEST>)**
- **rtfscommon\apps\prfstest.c - (<TEST>,<MOVEF> moved to rtfscommon\apps from rtfspackages\apps)**
- **rtfscommon\apps\protests.h - (<TEST>,<MOVEF> moved to rtfscommon\apps from rtfspackages\apps)**
- **rtfscommon\apps\protestrd.c - (<TEST>,<MOVEF> moved to rtfscommon\apps from rtfspackages\apps)**
- **rtfscommon\source\apiregrs.c - (<CLEANUP>)**
- **rtfscommon\source\rtdeviord.c - (<TEST>)**
- **rtfscommon\source\rtdeviowr.c - (<BUGFAILSAFE>)**
- **rtfscommon\source\rtfilebuffer.c - (<CLEANUP>)**
- **rtfscommon\source\rtfsbasicwr.c - (<BUGFAILSAFE>)**
- **rtfscommon\source\rtleakcheck.c - (<CLEANUP>)**
- **rtfscommon\source\rtlowl.c - (<CLEANUP>)**

- **rtfsfailsafe* - (<TESTS> <BUGFAILSAFE> <FSINDEXBUFFER>)**
- **rtfspackages\apps* - (<TESTS> <MOVEF>)**
- rtfspackages\rtfsproplus* - (<CLEANUP>)

Version 6.1.b

Release date: November 2008.

Rtfs 6.1.b

Removed obsolete files telserv.c and rtsockets.c from source trees and project files.
Updated several device drivers to the new device driver interface first released in Rtfs 6.1.b. The status of device drivers is now:

hostdev   - Upgraded
hostdisk   - Upgraded
norflash   - Upgraded
ramdisk   - Upgraded
romdisk   - Upgraded
ata   - Not Upgraded
mmc   - Not Upgraded
pcmcia   - Not Upgraded
smartmedia - Not Upgraded
floppy   - Not Upgraded

Version 6.1.a

Release date: November 6 2008.

Rtfs 6.1.a provides several important new features and some user interface improvements.

- An application layer callback method is provided for system wide configuration and optional dynamic memory support.
- New callback methods are provided that add additional features and simplify earlier callback interfaces.
- A new device driver model supports variable sector sizes, NAND erase blocks, dynamically configured operating and buffering

policies, more flexible support for removable media, on the fly drive letter and buffer assignment and optional dynamic memory support.

- Internal optimizations for NAND flash are provided which minimize block replacements by favoring erase block aligned buffering and write data transfers.

Note: Some parts of Rtfs have not yet been updated to the latest architecture, they are included with the release but are not functional. These include:

Device drivers – In this release only the host disk and ram disk are updated to the new device driver architecture, Conversion of the other device drivers is in process and will be available shortly.

Rtfs Pro Plus unit tests – These modules provide coverage testing for the Rtfs Pro Plus feature set. The tests are currently disabled because they rely on **pc_diskio_configure()** which is now obsolete and replaced by the new device layer configuration method. The tests will be upgrade in a future release.

Changes by file:

New header files:

- **rtfsblkmedia.h and rtfsfailsafe.h**, These are new files in the rtfscommon\include directory.
- **rtfsconfig.h** This is a new file in the project directory.
- Header files that have been removed or moved.
- **prfs.h, prfsint.h, ,rfsjournal.h, prfspriv.h, prfsrestore.h,** These header files have been removed and replaced by a single file named **rtfsfailsafe.h**.
- **portconf.h, rtfsproplus.h, rtfsint.h, rtleakcheck.h**, These header files have been removed and their contents merged into the header files named **rtfs.h** and **rtfsconf.h**..
- **rtfscfgdeclare.h and rtfscfgassign.h** These header files have been removed, their functionality is replaced by **rtfsconfig.h** in the project directory..
- **protests.h** This file has been moved to the ProPlus apps directory.

New source files:

- **rtfscommon\apps\appcmdshformat.c** This is a separate file for format related shell commands. Having the file segregated makes it easier as a reference for integrating format capabilities in an application.
- **rtfscommon\source\apipartition.c** Partition management code has been isolated to this module.
- **rtfscommon\source\drdynamic.c** This file provides the necessary API entry points and internal logic needed to implement new dynamic configuration and device attachment features.
- **rtfscommon\source\rteraseblock.c** – This file provides erase block aware internal functions that are needed for optimized NAND functioning.
- **rtfscommon\source\rtfilebuffer.c** – This file provides internal functions that process all file IO data transfers. When NAND media is installed erase block aware functions control and implement file buffering policies.
- **rtfsprojects\msvc.net\source\rtfscallbacks.c** – This file implements a set of model callback functions for Rtfs. This file should be copied to your project directory and modified as required for your environment and application needs.
- **rtfsprojects\msvc.net\source\rtfsconfig.c** – This file implements a model callback functions for Rtfs configuration. This file and its companion file , rtfsconfig.h should be copied to your project directory and modified as required for your application needs.
- **rtfsprojects\msvc.net\source\runrtfsdemo.c** – This file implements a model Rtfs session entry point that initializes Rtfs and selected device drivers. This file should be copied to your project directory and modified as required for your application needs.

Changes by file: - Some changes were made to almost all files to support the new feature set, the following list points out changes to be aware of.

rtfscommon\include\rtfsconf.h – Removed FAILSAFE_MODE_AUTOMATIC, RTFS_CFG_SHARE_BUFFERS, RTFS_CFG_ALLOC_FROM_HEAP, RTFS_CFG_NUM_USERS,

INCLUDE_DEBUG_VERBOSE_ERRNO, INCLUDE_DEBUG_SIM_ASSERT, INCLUDE_TELNET_TERMINAL, INCLUDE_SYS_TELNET_TERMINAL and STORE_DEVICE_NAMES_IN_DRIVE_STRUCT options. These options are now either obsolete or are provided by other means.

rtfscommon\source\apiregrs.c – Added a new test named **do_more_long_file_tests()** that verifies correct file/directory name handling and proper filename alias creation, including 1 byte and 2 byte Shift JIS characters, illegal characters and reserved names over a range of approximately 300 test cases.

rtfsdrivers\hostdisk\drhostdisk.c - Added a NAND simulator feature. Added the interface elements needed to support the new dynamic device driver methodology for simulated fixed disks and simulated NAND devices.

BLK_DEV_hostdisk_Mount() and **BLK_DEV_nandsim_Mount()** – These routines are called from startup code to mount the simulated fixed disk and simulated NAND devices. These routines call **pc_rtfs_media_insert()** to register the simulated devices with Rtfs. They may be used as a model when implementing other device drivers.

BLK_DEV_VIRT_device_configure_media() and **BLK_DEV_VIRT_device_configure_volume()** - These are the configuration callback functions for the simulated media drivers. They provide media and volume buffer configuration and allocation for the simulated devices and may be used as models when implementing other device drivers. *Note: These functions instruct Rtfs to dynamically allocate buffers, rtfsdrivers\ramdisk\drramdisk.c provides an example using static allocation of buffer pools.*

rtfs_media_insert_args - This structure is filled in for individual devices and passed to **pc_rtfs_media_insert()** by **BLK_DEV_hostdisk_Mount()** and **BLK_DEV_nandsim_Mount()**. This configures media wide buffering for the simulated devices and may be used as a model used by the host disk and that by host disk driver

The target specific porting file, portkern.c, has been updated with three new optional functions.

rtfs_port_set_task_env() and **rtfs_port_get_task_env()** provide an optimized method for Rtfs to map the current thread to it's Rtfs user context structure.

rtfs_port_set_task_exit_handler()provides a method to automatically schedule release of Rtfs user structures when a thread exits.

Version 6.0.f

Release date: August 1 2008.

- Release descriptions -
- Has several bug fixes that were made as a result of quality control review using removable media on embedded hardware platforms.
- Has an improved windev driver for accessing removable media from Rtfs running on a PC.
- Includes a dynamic device driver interface which allows intelligent device drivers, variable sector size and efficient support for NAND flash. This source code is included but it is not supported in this release. The code is excluded from compilation by the declaration in rtfsconf.h: #define INCLUDE_DYNAMIC_DRIVER 0. Do not enable enable INCLUDE_DYNAMIC_DRIVER in this release.
- Contains architectural changes that are necessary to support dynamic device drivers. Most changes are the result of introducing logic necessary to support variable sized sectors.
- Note: Variable sector size support requires INCLUDE_DYNAMIC_DRIVER , which is not supported in this release.

Bug fixes :

- \rtfscommon\source\apideltr.c – pc_deltree() changed exit code to be sure to clear errno if no error is being reported.
- rtfscommon\source\apifrmat.c – Changed _pc_mkfs() Changed FAT32 format to put the correct eight byte sequence in the first two cluster entries

- Changed from: MEDIADESC,FF,FF,FF , FF,FF,FF,0F, FF,FF,FF,FF
- Changed to: MEDIADESC,FF,FF,FF , FF,FF,FF,FF, 0F,FF,FF,FF
- rtfscommon\source\apimkdir.c – pc_mkdir()changed exit code to be sure to clear errno if no error is being reported.
- rtfscommon\source\apiunlink.c – pc_unlink() changed exit code to be sure to clear errno if no error is being reported.
- rtfscommon\source\rtfatdrvrd.c - fatop_next_cluster() fixed bug where Rtfs was not using the correct cluster chain terminator resulting in incompatabilities when expanding a FAT32 directory created with windows.
- Changed: (pdr->drive_info.fasize == 8 && nxt >= 0xffffff8))
- to: (pdr->drive_info.fasize == 8 && nxt >= 0x0ffffff8))
- \rtfspro\rtfat32rd.c - pc_init_drv_fat_info32() – Fixed code that generated an assert if the free sectors field in the info block was 0xffff ffff.. This is a legal value indicating “unknown”.
- Changed: ERTFS_ASSERT(pdr->drive_info.known_free_clusters < pdr->drive_info.maxfindex)
- To: ERTFS_ASSERT(pdr->drive_info.known_free_clusters == 0xffffffff || pdr->drive_info.known_free_clusters < pdr->drive_info.maxfindex)
- rtfspro\rtvfatwr.c - pc_insert_inode() – Changed code that assumed that extending a subdirectory by one cluster would always provide enough directory segments for any file size and cluster size. The code can now allocates additional clusters if one additional cluster is not enough.
- DROBJ *pobj , DROBJ *pmom, byte attr, dword initcluster, byte *filename, byte *fileext, int use_charset)

New features:

- rtfsdrivers\hostdev\drwindev.c – Several improvements including a background thread that monitors card removal events.
- The windev device driver now provides a reliable way to test Rtfs with removable media like SD/MMC and USB stick on a PC using Visual C. To use it enable INCLUDE_WINDEV in portconf.h.

Architectural changes to support variable sector sizes and dynamic device drivers:

Changes by file:

- rtfsconf.h
- Added new constant named RTFS_CFG_DEFAULT_BLOCK_SIZE
- Removed constants RTFS_MAX_BLOCKSIZE and RTFS_MAX_SECTORSIZE
- apirun.c
  - Eliminated use of RTFS_MAX_BLOCKSIZE, non 512 byte block device must now provide buffering
- apicnfig.c
  - Changed configuration for new block buffering scheme
- rtdblock.c
  - Changed pc_initialize_block_pool for new block buffering scheme.
  - Changed scratch block handling code to support allocating sector sized scratch buffers on media with non 512 byte sectors
- rtkernfn.c
  - Changed pc_memory_init() for new block buffering scheme.
- Many Files
  - Changed all instances of &buff->data[X] to buff->data+X
  - Changed all instances of &buff->data[0] to buff->data
  - Changed all calls to pc_scratch_blk() to pass a drive structure if a sector sized buffer is required

---

Version 6.0.e

Release date: April 2008 – (released for internal review)

Synopsis –

Bug fixes:

- Changed format routine to build the volume sized by the original block count, not by the block count calculated from H*C*N, which can be truncated.
- Added PO_APPEND file mode support to the basic file IO layer provided with RtfsBasic and RtfsPro
- Changed po_lseek() behavior to more closely emulate po_lseek() behavior of version 44.

    - Return -1 and set errno to PEINVALIDPARMS if origin is PSEEK_SET and offset < 0
    - Return -1 and set errno to PEINVALIDPARMS if origin is PSEEK_END and offset > 0
- Fixed command shell command line processing bug, which did not allow passing file names that start with a number.
- Made bug fixes to vfat file creation code.
    - If mixed case short file name is provided, create a mixed case lfn, but only if an alias does not already exist.
    - Fixed bug that allowed illegal characters in the first character of a file name when using the Unicode interface.
- Added exhaustive vfat file naming compliance test to apiregrss.c.
-
- New features:
    - Added new options to format to force FAT32 format, to force only one fat copy and to force a specific cluster size.
    - Added a section to the manual application notes describing Rtfs tests
    - Added XML formatted code to comments in test programs to extract test commentary from source code.
    - Added code to API run that demonstrates configuring more than 1 drive when dynamic allocation is disabled.
    - Added new compile time constant, **RTFS_CFG_MAX_DIRENTS**, that determines the maximum number of directory entries that may be created or scanned in an individual subdirectory.
    - Added support for **RTFS_CFG_MAX_DIRENTS** in directory entry scan and create procedures.
    - Added diagnostic leak checking procedure for testing.

Changes by file:

- rtfscommon\source\prbasicemurd.c
    - Changed po_lseek() behavior.
- rtfscommon\source\apifrmat.c
- rtfscommon\include\rtfstypes.h
- rtfscommon\apps\appcmdsh.c
    - Added new options to format that allow greater user control over how a volume is formatted.
        - force_reserved_sectors – Renamed from “reserved_sectors” and update user’s manual.

        - force_fat32 - Added a new option to force formatting FAT32, regardless of the volume size.
        - force_one_fat - Added a new option to force formatting with only one fat copy.
        - force_cluster_size - Added a new option to force the cluster size to a specific number of blocks regardless of the volume size.
        - total_sectors - Added a new field **total_sectors** to the fmtparms structure that holds the 32 bit volume size before HCN values are truncated to legal sizes. When the volume is formatted this value is used as the total volume size, instead of H*C*N, which may be truncated truncated.
    - Added new optional LEAKCHECK command.
- rtfspro\source\csunicodewr.c – Fixed bug in pc_unicode_validate_filename() that allowed a file to created with an illegal character in the first character.
- rtfscommon\source\rtfsbasicrd.c
- rtfscommon\source\rtfsbasiwr.c
    - Created new function named pc_bpefile_ulseek(). This internal function is called by seek and by write when in append mode.
- rtfscommon\source\apirun.c
    - Added code to API run that demonstrates configuring more than 1 drive when dynamic allocation is disabled.
- rtfscommon\apps\apiregrs.c
    - Added test for open append mode.
    - Added exhaustive vfat file naming compliance test
- rtfscommon\apps\apputil.c
    - Fixed command shell command line processing bug.
- rtfscommon\source\cscommon.c
    - Modified pc_cs_valid_sfn() to optionally identify lower case characters as invalid.
- rtfscommon\source\csjiswr.c
    - Modified jis_valid_sfn () to optionally identify lower case characters as invalid.

- rtfscommon\source\csasciiwr.c
  - Modified ascii_valid_sfn () to optionally identify lower case characters as invalid.
- rtfspro\source\rtvfatwr.c
  - Several changes to fix alias creation problems
  - Added support for **RTFS_CFG_MAX_DIRENTS**
- rtfspro\source\rtvfatrd.c
  - Added support for **RTFS_CFG_MAX_DIRENTS**
- rtfscommon\source\rtnvfatwr.c
  - Added support for **RTFS_CFG_MAX_DIRENTS**
- rtfscommon\source\rtvnfatwr.c
  - Added support for **RTFS_CFG_MAX_DIRENTS**
- rtfscommon\source\rtleakcheck.c
- rtfscommon\include\rtleakcheck.h
  - New leak checking code.

Version 6.0.d

Release date: January 2008

Synopsis –

New features:

- Command shell menus were changed to be easier to understand.
- Added API routines that directly manipulate clusters in linear.
  - pc_cluster_to_sector
  - pc_sector_to_cluster
  - pc_efilio_extract
  - pc_efilio_swap
  - pc_efilio_remove
- Added new API routines that perform 64 bit arithmetic on HI:LO.
  - pc_subtract_64
  - pc_add_64
- Manuals were rearranged and revised with formatting improvements, pages for new functions, updated command shell reference guides and documentation of 64 bit arithmetic macros.
- A new implementation of the posix like file IO routines (po_open, po_close et al.) has been implemented for use in version 6.x

RtfsBasic and RtfsPro releases. This is a lightweight library providing basic file IO without using the extended file IO package.

- Some restructuring has been performed to support creating separate releases for RtfsBasic, RtfsPro, RtfsProPlus and RtfsProPlusDvr and failsafe variants from the version 6.x source tree.
  - Rtfs Pro includes FAT12, FAT16, FAT32 and VFAT, but it does not include extended file IO.
  - Rtfs Basic includes FAT12 and FAT16. It excludes FAT32 and VFAT, only supporting 8.3 file names. It, like RtfsPro excludes extended file IO.
  - When Rtfs Pro and Rtfs Basic are created, extended IO files in rtfspackages/source are omitted.
  - RtfsProPlus includes extended fileIO but not 64 bit files and circular files.
  - Rtfsconf.h includes a file named rtfspackages.h and adjust compile time constants to reflect what packages are included.
  - A new directory named rtfspro was created and source files not required for RtfsBasic were moved from rtfscommon\source to the new directory.
  - A new directory named rtfspro was created and source files not required for RtfsBasic were moved from rtfscommon\source to the new directory.
  - A new directory named rtfspackages\rtfsproplus was created and extended file IO were moved from rtfspackages\source.
  - A new directory named rtfspackages\rtfsproplusdvr was created and 64 bit file and circular file IO were moved from rtfspackages\source.
  - Minor modifications were made to several test files, shell files, configuration files and header files to support the new code partitioning scheme.

Changes by file:

- rtfscommon\source\rtfsbasicrd.c
- rtfscommon\source\rtfsbasicwr.c
  - These are new files containing source code that provides a lightweight file IO alternative to extended IO routines.

- rtfscommon\source\prbasicemurd.c
- rtfscommon\source\prbasicemuwr.c
  - These files were moved from rtfspackages\source to rtfscommon\source.
  - They were modified to call either basic or extended file IO routines to provide file IO for the **INCLUDE_BASIC_POSIX_EMULATION** configuration.
- rtfspackages\rtfsproplus
  - This subdirectory was created and several files were moved from rtfspackages\source to it.
  - rtfspackages\rtfsproplus\rtfsproplusglue.c
    - Added these new subroutines
      - pc_bytes_to_clusters
      - pc_clusters_to_bytes
      - pc_subtract_64
      - pc_add_64
    - Moved these subroutines from apiinfo.c to rtfsproplusglue.c
      - pc_cluster_to_sector
      - pc_sector_to_cluster
  - rtfspackages\rtfsproplus\prapilinext.c
    - Created this new file which provides the following new API subroutines
      - pc_efilio_extract
      - pc_efilio_swap
      - pc_efilio_remove
- rtfspackages\apps\prlinextest.c
  - Created this new file containing test code for pc_efilio_extract, pc_efilio_swap, and pc_efilio_remove
- rtfspackages\apps\efishellrd.c
  - rearranged menus
- rtfspackages\rtfsproplusdvr
  - This subdirectory was created and several files were moved from rtfspackages\source to it.
- rtfspackages\source
  - This subdirectory was eliminated
- rtfscommon\includes\rtfsconf.h

        - rtfsconf.h includes the file rtfspackages.h and reconfigures itself by disabling options that were not purchased.
        - Added new configuration constant named **INCLUDE_BASIC_POSIX_EMULATION**
    - rtfscommon\includes\rtfs.h
    - rtfscommon\includes\rtfsproplus.h
    - rtfscommon\includes\rtfstypes.h
    - rtfscommon\includes\rtfsprotos.h
        - Minor changes to support **packages** and to prototype new code.
    - rtfscommon\apps\appcmdshrd.c
        - Minor changes to support **packages.**
        - Added a disk flush command
        - re-arranged menus
    - rtfscommon\apps\appcmdshwr.c
        - Changed partitioning dialogue to fill the media with a single partition if no size is specified.
        - Minor changes to support **packages.**
        - Added a disk flush command
    - rtfscommon\source\rtdrobj.c
    - rtfscommon\source\apicnfig.c
    - rtfscommon\source\rtfragmtrd.c
    - rtfscommon\source\rtkernfn.c
    - rtfspackages\source\prfragmtrd.c
        - These files include minor changes to support **packages.**
    - rtfspackages\source\prfragmtrd.c
        - Changed pc_fraglist_coalesce. Was leaving certain fragments un-coalesced if 3 fragments in a row were contiguous. Did not cause problems previously but the change is required for pc_efilio_extract.
    - rtfscommon\source\apidiskinfowr.c
    - rtfscommon\source\apirawrd.c
    - rtfscommon\source\apirawwr.c
    - rtfscommon\source\csunicodrd.c
    - rtfscommon\source\csunicodwr.c
    - rtfscommon\source\rtfat32rd.c
    - rtfscommon\source\rtfat32wr.c

- rtfscommon\source\rtfreemanager.c
- rtfscommon\source\rtvfatrd.c
- rtfscommon\source\rtvfatwr.c
    - These files were moved from rtfscommon\source to a new subdirectory name rtfspro.
    - No other changes were made to these files
- rtfscommon\source\rtfragmtwr.c
    - This file was moved from rtfscommon\source rtfspro.
        - Minor changes to support **packages.**

Version 6.0.c

Release date: November 26, 2007

Synopsis –

Failsafe: new features

- Changed the hidden journal file feature so that the Journal blocks are stored in free-space but no Failsafe file is created in the root directory. Failsafe reserves free clusters when it starts journaling and stores that location in the second entry in the second FAT copy.
- Automatically reduce the Journal file size when the disk becomes too full to contain both the current Journal file and new content.
- Added a new callback mechanism fs_api_cb_min_journal_size() that allows the application to specify the smallest allowable Journal file.
- Added a new callback mechanism fs_api_cb_disable_on_full() that allows the application to direct Rtfs to automatically disable Failsafe when the disk becomes too full to contain both the current Journal file and new content.
- Added a new callback mechanism fs_api_cb_journal_fixed() that allows the application to specify the exact placement of the journal file on the media. This may be in hidden sectors, on a separate partition, or in blocks reserved by the device driver.
- Added journal_max_used to failsafe statistics. Tracks worst case journal block consumption.

- Major update to the Failsafe Reference Manual
- Removed references to INCLUDE_EXPERIMENTAL_HIDDEN_JOURNAL. Journaling to free-space with no Journal file in the root is now the default behavior.
- Updated regression tests to test automatic journal file resizing on disk full conditions.
- Updated regression tests to test exact journal file placement.

Other changes:

- Added a new compile time configuration option. – The new constant, INCLUDE_SYS_TELNET_TERMINAL, instructs Rtfs to perform console IO using telnet but not to include the mini-telnet server code. If this option is enabled the following IO routines must be provided by the runtime library:
  - void telnet_gets(byte *buffer)
  - void telnet_puts(byte *buffer)
- Added a new disk configuration option, DRVPOL_DISABLE_FREEMANAGER. This option disables the memory based free manager for the drive. It reduces ram consumption and can be used for read-only or infrequently written to drives. It can also be disabled for drives that are written to frequently. IN this case real time performance is not possible the performance is still adequate for many applications.
- Added a Hostdisk mode that will run in any environment that supports ANSI file IO.

Following manual sections have changed.

- FailsafeTechnicalReferenceManual
  - Revised several sections
  - Moved Failsafe API manual pages from the Rtfs Users Guide to the Failsafe manual.
  - Document new hidden journal feature
  - Documented fs_api_cb_journal_fixed() callback
  - Documented automatic resizing.
- ApiReferenceGuide
  - Update pc_diskio_configure manual page to include new DRVPOL_DISABLE_FREEMANAGER policy option.
  - Moved Failsafe API manual pages from the Rtfs Users Guide to the Failsafe manual.

- ConfigurationGuide
  - Update compile time configuration section to include new INCLUDE_SYS_TELNET_TERMINAL option.

Changes by file:

- Changes to Failsafe files
  - Failsafe related files were changed to add the following features.
  - Validated the hidden Journal file technique first introduced in version 6.00b.
  - Corrected vulnerability in hidden file technique. The hidden journal file was vulnerable to overwrite because it could be placed in a sector that already contained entries. No the journal file starts a sector and all other entries in that sector are copies of it.
  - Removed previous journal file code that used a visible named journal file.
  - Removed references to INCLUDE_EXPERIMENTAL_HIDDEN_JOURNAL
  - Modified some internal calling sequences and added a routine named fs_recover_free_clusters() that automatically frees clusters by resizing the Journal file when the disk fills.
  - Added a new function named fs_api_cb_journal_fixed() to allow strict fixed placement of the Journal file.
  - Added Failsafe compile time option RTFS_CFG_MIN_JOURNAL_SIZE
  - Added Failsafe compile time option RTFS_CFG_FAILSAFE_DISABLE_WHEN_FULL
  - Added test code for new features
    - rtfsfailsafe\prfs.h
    - rtfsfailsafe\prfscb.c
    - rtfsfailsafe\prfsjournal.c
    - rtfsfailsafe\prfsjournal.h
    - rtfsfailsafe\prfsnvio.c
    - rtfsfailsafe\prfsrestore.c
    - rtfspackages\apps\prfstest.c
- Changes to other files
  - rtfscommon\include\rtfsconf.h

        - Removed INCLUDE_EXPERIMENTAL_HIDDEN_JOURNAL option
        - Added RTFS_CFG_MIN_JOURNAL_SIZE option for Failsafe
        - Added RTFS_CFG_FAILSAFE_DISABLE_WHEN_FULL option for Failsafe
        - Added INCLUDE_SYS_TELNET_TERMINAL option
    - rtfscommon\include\rtfsproplus.h
        - Added new DRVPOL_DISABLE_FREEMANAGER flag parameter for pc_diskio_configure
    - rtfscommon\include\rtfstypes.h
        - Removed INCLUDE_EXPERIMENTAL_HIDDEN_JOURNAL reference
    - rtfscommon\source\apiinit.c
        - Minor change to select a default host disk file name when Windows or Linux are not defined but the host disk driver is enabled.
    - rtfspackages\source\preficomwr.c
        - Added calls to fs_recover_free_clusters() when the disk is full and rtfs would like Failsafe to release clusters to be used for allocations.
    - rtfscommon\source\rtfatdrvwr.c –
        - Removed references to INCLUDE_EXPERIMENTAL_HIDDEN_JOURNAL.
        - Added calls to fs_recover_free_clusters() when the disk is full and rtfs would like Failsafe to release clusters to be used for allocations.
        - Removed the function named fatop_alloc_contiguous_chain(). This was used to allocate the Failsafe file but the new method no longer requires it.
    - rtfscommon\source\rtfreemanager.c
        - Removed references to INCLUDE_EXPERIMENTAL_HIDDEN_JOURNAL.
        - Modified free_manager_attach() to disable the free manager for the drive if the DRVPOL_DISABLE_FREEMANAGER flag parameter was passed to pc_diskio_configure
    - The following files were modified to remove references and source code that placed a Journal file in a deleted

directory entry. There is no longer any journal file, the journal blocks are found in freespace.

- rtfscommon\includes\rtfstypes.h
- rtfscommon\includes\rtfsprotos.h
- rtfscommon\source\rtnvfatrd.c
- rtfscommon\source\rtvfatrd.c
- rtfscommon\source\rtfsgluerd.c
- rtfscommon\source\rtnvfatwr.c
- rtfscommon\source\rtvfatwr.c

- rtfscommon\source\rtkernfn.c
  - Minor changes to support the new INCLUDE_SYS_TELNET_TERMINAL option
- rtfscommon\source\rttermin.c
  - Cleaned up 64 bit integer formatting with sprintf.
- rtfspackages\apps\efishellrd.c
  - Minor changes
- targets/portkern.c
  - A minor change to target specific portkern.c files excludes terminal support if either INCLUDE_TELNET_TERMINAL or INCLUDE_SYS_TELNET_TERMINAL are enabled.
- rtfsdrivers\hostdisk\drhostdsk.c
  - Added support for using ANSI file IO routines for the Host disk when neither Linux nor Windows are defined.

Version 6.0.b

Release date: October 25, 2007

Synopsis – This release provides the ability to journal Failsafe data to blocks in free space. This functionality was omitted from version 6.0.a and conditionally excluded in certain places with the compile the pre-processor variable INCLUDE_EXPERIMENTAL_HIDDEN_JOURNAL

Note: In this release the compile time define "INCLUDE_EXPERIMENTAL_HIDDEN_JOURNAL" is enabled and the code sections enabled by this are included. The Failsafe Technical reference manual

has not been updated to describe the new Journal file placement. In the next release the manual will be updated and instances of "#if (INCLUDE_EXPERIMENTAL_HIDDEN_JOURNAL)" will be removed.

The following changes are included in release 6.0.b

- Implemented a new method to store the Failsafe journal file on disk as a deleted file.
- Minor bug fix regarding asynchronous mount applied to a 12 bit volume.
- Updated tests in rtfspackages\apps to run on 12 bit volumes.
- This is a minor release

Changes by File:

- Bug Fixes:
  - rtfspackages/source/prapiasyrd.c – Corrected a problem in pc_diskio_async_mount_start that was setting the async state wrong for mounts of FAT12 devices, causing async mount to fail on FAT12.
  - rtfscommon/source/apiinit.c – Corrected a problem with the default name of the HostDisk file pointing to a subdirectory on a separate drive. It is now in the default directory.
- New features:
  - Completed testing new Failsafe code that places the Journal in free clusters.
    - rtfscommon/include/rtfsconf.h – Set INCLUDE_EXPERIMENTAL_HIDDEN_JOURNAL to 1.
    - rtfsfailsafe/prfsnvio.c – Made necessary code changes for INCLUDE_EXPERIMENTAL_HIDDEN_JOURNAL
  - Changed the following files to support testing extended API on 12 bit FATS.
    - rtfspackages/apps/prasytest.c
    - rtfspackages/apps/protests.h
    - rtfspackages/apps/protestrd.c
  - The following manuals were revised because of formatting problems.
    - ApiReferenceGuide.pdf

- InstallationGuide.pdf

Version 6.0.a
October 20, 2007

Initial release – Please review the documents in the manual section.

Known problems with version 6.0.a –

The open option combination PCE_FORCE_CONTIGUOUS|PCE_KEEP_PREALLOC does not work for 64 bit files.
A new journal placement algorithm is under development but it was not completed in time for official inclusion in this release. The new method uses unallocated clusters for Journaling and uses a file marked deleted to store the position of the Journal. This solves several problems including:

- The journal file is not present and reserves no space when removable media is placed in another PC.
- The journal file can be smaller because cluster boundary requirements are relaxed. This provides a pronounced improvement on very small FAT16 and FAT12 volumes.

The code is mostly developed and is currently in testing. The source code for this feature is included in this release but it is conditionally excluded by the compile time variable, INCLUDE_EXPERIMENTAL_HIDDEN_JOURNAL. We expect to release version 6.0.b within days with INCLUDE_EXPERIMENTAL_HIDDEN_JOURNAL enabled. Only a few files will be changed.