NASA GSFC FLIGHT SOFTWARE SYSTEMS BRANCH

FSW VERSION DESCRIPTION DOCUMENT

CFS DATA STORAGE (DS) APPLICATION

BUILD: DS 2.5.2

RELEASE DATE: 10/05/2020

1.0 FSW VERSION DESCRIPTION

1.1 PURPOSE AND SUMMARY

This build is a minor build of the Data Storage (DS) application to resolve bugs with the DS application and ensure compatibility with cFS Bootes. This build does not include any new functionality.

1.2 NEW/CHANGED FUNCTIONALITY IN THIS VERSION

Table 1.2-1 identifies the DCRs that have been implemented in this FSW version. For each DCR the "Key" column shows the corresponding DCR in the GSFC cFS tracking system.

Table 1.2-1 – DCRs Implemented in this Version

Key	Summary	Description
GSFCCFS-742	Data smashing via strncpy in DS unit test	I think a DS unit test function has an incorrect parameter. In function DS_CmdSetDestExt_Test_FileTableNotLoaded() there is a line that reads: strncpy (CmdPacket.Extension, "txt", DS_SET_DEST_EXT_CC); I'm fairly certain the length of the extension was intended, not the command code (10).

GSFCCFS-743	Uninitialized memory in DS unit test	I get the following error in the DS unit tests:
		
		FAIL: DS_AppData.AppEnableState == 0, File: /home/vagrant/Desktop/airliner/apps/ds/fsw/unit_test/ds_table_test.c, Line: 1394
		
		I think I have a bug report and a recommendation. First, the CFE_ES_RestoreFromCDSHook() function is returning a chunk of memory that is cleared with the following:
		
		memset(RestoreToMemory, 0, DS_DEST_FILE_CNT*4);
		
		That chunk of memory is actually DS_DEST_FILE_CNT+1 words in size, so the last uint32 element is not cleared with memset. The variable is allocated on stack so I see garbage in that element. The unit test is expecting DS_AppData.AppEnableState is stored in that element, so it fails due to the garbage. I recommend the hook be changed to:
		
		memset(RestoreToMemory, 0, (DS_DEST_FILE_CNT+1)*4);
		
		Second, its testing code that is actually conditionally compiled with the "#if (DS_CDS_ENABLE_STATE == 1)" macro. I recommend that the line 1038 be changed to read:
		
		#if (DS_CDS_ENABLE_STATE == 1)
		/* Only restore enable/disable state if configured */
		DS_AppData.AppEnableState = (uint8) DataStoreBuffer[DS_DEST_FILE_CNT];
		#endif
		
		So it will match whatever configuration is used.

GSFCCFS-750	Data Storage References GPM Spacecraft Header Types	ds_appdefs.h allows a file header type to be defined that is specific to the Global Precipitation Measurement (GPM). This should either be removed or generalized to a non-mission specific feature.
GSFCCFS-940	DS Unit Test must be updated to avoid seg faults	The current DS unit test configuration results in a segmentation fault when executed. The unit test must be updated to configure the test conditions so a seg fault does not occur. The FM unit test can be referenced as an example of how to avoid segfaults.
GSFCCFS-945	Fix DS compilation warnings with cFE 6.6	Building DS with cFE 6.6 generates a warning
GSFCCFS-972	DS CMakeLists.txt file should include table	Need an add_cfe_table() for the file and filter tables.
GSFCCFS-973	Issues buliding DS unit tests	Need to add comment in either the unit test makefile or the README.txt to define the following environment variables: UT_ASSERT_SRC, CFS_MISSION, CFE_DIR, OSAL_DIR, PSP_DIR & APP_DIR Need to update include path to osconfig.h COPT and LOPT Won't build without -m32 Suggest we use the following options to show line & branch coverages: @gcov -f -b $(FLIGHT_OBJS:.o=.gcda)
GSFCCFS-974	DS functions have uninitialized variables	The following DS functions have uninitialized variables. DS_TableInit DS_TableManageDestFile DS_TableManageFilter DS_TableVerifyDestFile DS_CmdsSetDestAge DS_CmdsSetDestCount DS_CmdAddMID DS_CmdSetFilterParams DS_CmdSetDestType DS_CmdSetFilterType DS_CmdSetFilterFile DS_AppProcessMsg DS_AppMain DS_FileStorePacket
GSFCCFS-975	DS has padding at the end of many structures	The DS_AppStateCmd_t seems to have padding that doesn't even pad to 32-bit boundaries.
GSFCCFS-976	DS Missing Doxygen comments for some prototypes	The function DS_FileConvertGPM is missing doxygen.
GSFCCFS-977	DS_file.c relies on hard-coded constants	The function DS_FileConvertGPM includes a number of hard-coded constants (so-called "magic numbers").
GSFCCFS-978	DS Value never used	In file ds_cmds.c, function DS_CmdAddMID, the value of HashTableIndex is never used after assignment. (Static code analysis result)
GSFCCFS-979	Sprintf in DS_AppProcessHK	sprintf is used in the function DS_AppProcessHK - potential for buffer overflow. Should be replaced with snprintf

GSFCCFS-980	DS possible overflow in strncpy	In DS_AppProcessHK: Static code analyzer - possible buffer overflow in string operation, where there's no place for string NULL terminator.
GSFCCFS-981	Potential out bounds in DS_FileCreateName	if (Workname[TotalLength - 1] != DS_PATH_SEPARATOR) Static code analyzer - Potential array out of bound if TotalLength is 0.
GSFCCFS-982	DS potential overflow in strcpy	in ds_file.c, function DS_FileCreateName: Static code analyzer - potential buffer overflow in unbound string copy. Should use strncpy instead.
GSFCCFS-998	DS table filenames need modification	DS table filenames (DS_DEF_DEST_FILENAME and DS_DEF_FILTER_FILENAME) need modification from "cf/apps/..." "cf/..." in order to work out-of-the-box with cFE 6.6.
GSFCCFS-1060	Remove GPM-Specific Code from DS	Remove the GPM-specific code from DS.
GSFCCFS-1068	DS Readme file needs updates	Readme file contains out of date information on sources for cFE and OSAL.
GSFCCFS-1122	Update DS Doxygen User Guide	Suggest updating the doxygen user guide file and adding a doxygen configuration file to allow users to successfully generate the doxygen guide themselves.
GSFCCFS-1150	DS should build with Werror and OMIT_DEPRECATED	DS should build against the latest cFE with -Werror and OMIT_DEPRECATED true.
GSFCCFS-1176	DS has no space for null terminator	In analysis on 7/10/2020, CodeSonar flagged the following red finding: No Space For Null Terminator help strncpy() may not null-terminate Workname. strncpy() may write a non-null value into the very last byte of the string. The string's capacity is 128. The last character written by strncpy() could be character WorknameLen - 1, which evaluates to 127. If a subsequent statement definitely null-terminates the string, you can safely ignore this warning. The issue can occur if the highlighted code executes.
GSFCCFS-1234	DS may have alignment problems on some platforms	DS uses uint8[] for command and telemetry packet headers. This can cause alignment issues (this has been experienced with other apps). Instead of the uint8[], the command and telemetry packets should use the actual header types to ensure alignment.
GSFCCFS-1242	DS Version is not correct for Release Candidate 2.5.2	The ds_version.h file still contains 2.5.1 as the version in the rc2.5.2 branch.

No new functionality was added in this build.

1.3 MISSING PLANNED FEATURES AND KNOWN PROBLEMS

Table 1.3-1 identifies currently open DCRs that are not addressed in this build.

Any workarounds that may apply are identified.

Table 1.3-1 – Currently open DCRs

Key	Summary	Description
GSFCCFS-1179	DS has static code analysis findings	In analysis on 7/10/2020, CodeSonar flagged the attached findings.
GSFCCFS-1087	Migrate DS unit tests to distributed UT Assert	
GSFCCFS-1018	DS should use CFE_FS_InitHeader	DS initializes and populates the CFE FS header itself (see ds_file.c, lines 273-278) This requires DS to understand the details of the header structure and could break if that structure changes. It should instead use the CFE_FS_InitHeader function which is designed to do exactly this.
GSFCCFS-759	DS file header should include additional metadata	DS currently stores a number of fields in the DS file header (DS_FileHeader_t), namely the time the file was closed, the file name, the file table index, and the file name type. When reading DS-created files on other platforms with other configurations, it is possible to tease apart platform/mission-specific information but it would be easier to store the configuration in the header for easier analysis of DS files. Additional information should include, at least, the CCSDS time format (CFE_SB_PACKET_TIME_FORMAT), the CCSDS secondary header (CC, timestamp) endian-ness, byte alignment, even the DS file header endian-ness for the close timestamp. Also, as this will break compatibility with any DS-generated files from previous versions, including a DS header format version # should be included.
GSFCCFS-737	DS file header values should be big-endian	As with CCSDS, which is standardizing on big-endian for message headers, the fields in the DS file header (close time, FileTableIndex, FileNameType) should be stored in big-endian order. (Should the same be true for the CFS file header?)

2.0 DELIVERED PRODUCTS

Table 2-1 identifies the locations of FSW products relevant to this FSW Build. The version or date of the Build and where the product can be located are provided. Changes from a previous VDD are identified.

Table 2-1 – Delivered Products and their Locations

Software Element	Changed with this Version?	New Version or Date	Location
Source Code of this FSW Build	Yes	2.5.2	https://github.com/nasa/ds
Doxygen Documentation	Yes	N/A	https://github.com/nasa/ds
Unit Test Data	Yes	2.5.2	https://github.com/nasa/ds
FSW Make Files	Yes	2.5.2	https://github.com/nasa/ds

3.0 INSTALLATION PROCEDURES

In order to build and install the DS application, it must be added to the cFE CMake build system. This is done by modifying the TGTX_APPLIST in the cFE targets.cmake file. This is shown in the trivial example below.

```
SET(TGT1_NAME cpu1)
SET(TGT1_APPLIST cfs_lib ds)
SET(TGT1_FILELIST cfe_es_startup.scr)
```

After DS is added to the targets.cmake file, it is built and installed using the standard cFE CMake build instructions. These instructions are available in cFE CMake documentation:

https://github.com/nasa/cFE/blob/main/cmake/README.md

4.0 CONFIGURATION SUMMARY AND VERSION IDENTIFICATION

This software can be found in the DS GitHub repository (https://github.com/nasa/DS) under the tag “2.5.2”.

Verification of the version can be done by sending a DS NOOP command which produces an event message containing the version information. In addition, the initialization event message generated during the application startup provides the version information.

ACRONYMS

ACS ..... Attitude Control System
C&DH ..... Command and Data Handling
cFS ..... Core Flight System
CM ..... Configuration Management
COTS ..... Commercial Off-The-Shelf
CPU ..... Central Processing Unit
DCR ..... Discrepancy/Change Request
DS ..... Data Storage
ETU ..... Engineering Test Unit
FSB ..... Flight Software Branch
FSW ..... Flight Software
GSFC ..... Goddard Space Flight Center
I&T ..... Integration & Test
JSC ..... Johnson Space Center
POSIX ..... Portable Operating System Interface
RTOS ..... Real-Time Operating System
SMP ..... Symmetric Multiprocessing
T&C ..... Telemetry and Command
TBD ..... To Be Determined
URL ..... Universal Resource Locator
VDD ..... Version Description Document