National Aeronautics and Space Administration

cFE Release 6.6

A Summary of cFE 6.6

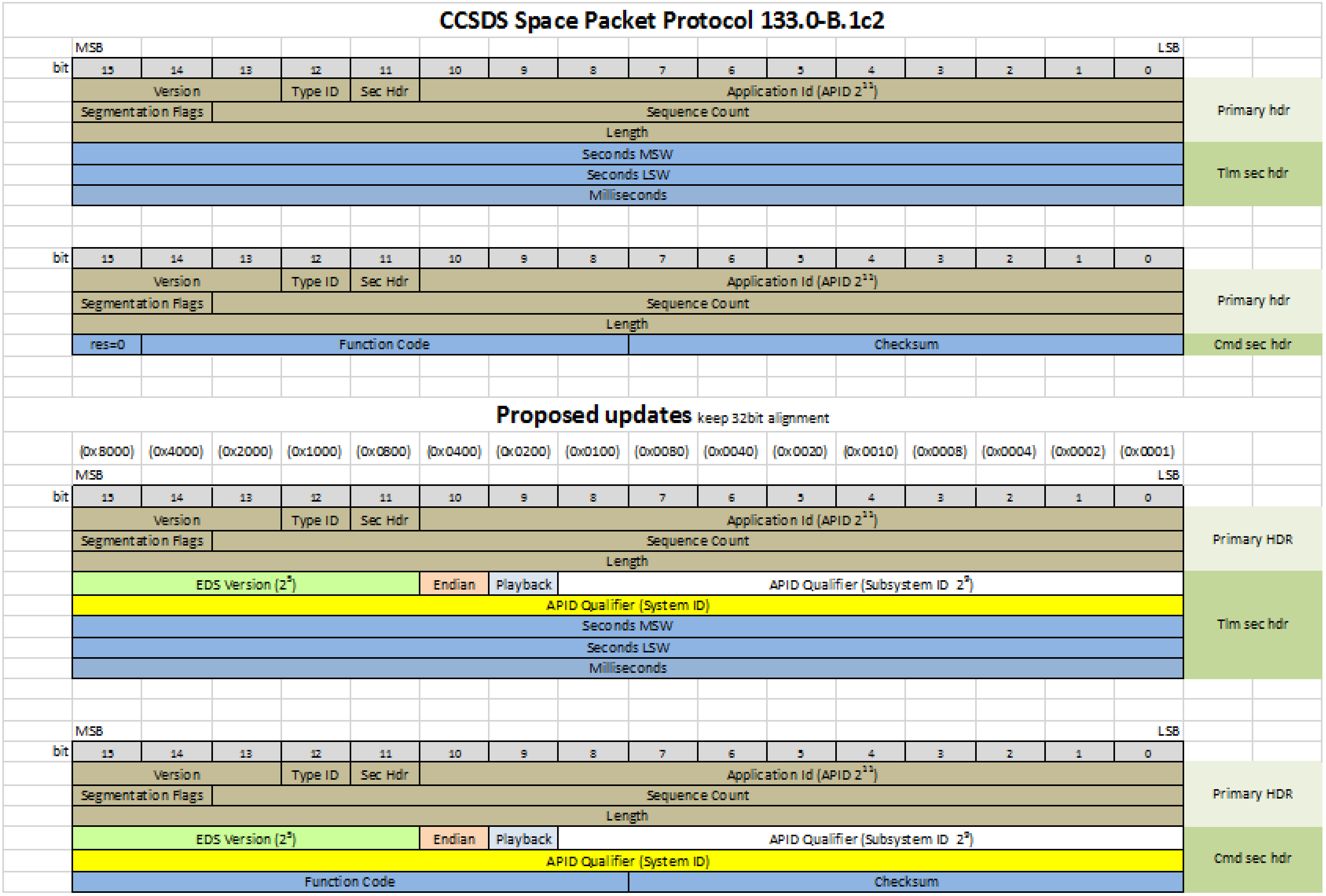
- All qualification testing and documentation is now complete and the release has been tagged and approved as “ready to ship”
  - Just finalized last Friday, Dec 1 2017, ~17 months of development since 6.5.0 (released June 2016)
  - NASA users can get it immediately from the git server
  - Should hit all the regular public distribution channels in the near future (i.e. sourceforge etc).
- Major target areas:
  - Support expanding number of software bus endpoints beyond 2^{11}.
  - Improve support for multi-core CPUs (SMP)
  - Introduction of CCSDS electronic data sheet (EDS) files to accompany the source files.
  - And a significant number of other bug fixes and improvements….

CCSDS “Version 2” Support

Address the limitation of CCSDS packets to 2^{11} maximum endpoints (APID’s) by expanding this with an additional “APID Qualifier” header.

- By default, this is not enabled – it must be explicitly selected by setting the “`MESSAGE_FORMAT_IS_CCSDS_VER_2`” mission config
- When enabled, adds an additional 4 bytes / 32 bits to all packet headers
  - APID Qualifier starts where the command/telemetry secondary header would have been
  - Offsets the “traditional” command/telemetry secondary header by 32 bits (otherwise unchanged other than the offset)
- 25 bits of which are allocated for message routing, expanding the total message routing space to 2^{36}
  - 16 bits for “System ID”
  - 9 bits for “Subsystem ID”
  - However, not all the newly added bits are actively used by in the default, which keeps the route lookup table size in check. A hash function or similar approach could be used to gain more routing entries.

CCSDS "Version 2" Support



CCSDS Space Packet Protocol 133.0-B.1c2

	MSB															LSB	
bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
	Version			Type ID	Sec Hdr	Application Id (APID 2^{11})											Primary hdr
	Segmentation Flags		Sequence Count														
	Length																
	Seconds MSW																Tlm sec hdr
	Seconds LSW																
	Milliseconds																

bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
	Version			Type ID	Sec Hdr	Application Id (APID 2^{11})											Primary hdr
	Segmentation Flags		Sequence Count														
	Length																
	res=0	Function Code							Checksum								Cmd sec hdr

Proposed updates keep 32bit alignment

	(0x8000)	(0x4000)	(0x2000)	(0x1000)	(0x0800)	(0x0400)	(0x0200)	(0x0100)	(0x0080)	(0x0040)	(0x0020)	(0x0010)	(0x0008)	(0x0004)	(0x0002)	(0x0001)	
	MSB															LSB	
bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
	Version			Type ID	Sec Hdr	Application Id (APID 2^{11})											Primary HDR
	Segmentation Flags		Sequence Count														
	Length																
	EDS Version (2^5)					Endian	Playback	APID Qualifier (Subsystem ID 2^9)									Tlm sec hdr
	APID Qualifier (System ID)																
	Seconds MSW																
	Seconds LSW																
	Milliseconds																

	MSB															LSB	
bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
	Version			Type ID	Sec Hdr	Application Id (APID 2^{11})											Primary HDR
	Segmentation Flags		Sequence Count														
	Length																
	EDS Version (2^5)					Endian	Playback	APID Qualifier (Subsystem ID 2^9)									Cmd sec hdr
	APID Qualifier (System ID)																
	Function Code								Checksum								

CCSDS “Version 2” Support

- This naturally has an impact to existing code, but not as much as one might expect
- So long as the proper software bus type definitions, macros, and API calls are utilized by applications, it should “just work”
  - Apps must be sure to use the proper `CFE_SB_MsgId_t` and not `uint16` to store message ID values, as it may become 32 bits instead of 16 bits
  - Apps must not assume certain field positions or offsets, and should always use the SB-provided macros for reading header fields

Multi-Core (SMP) CPU Support

- Focus on fixing the known thread safety/concurrency issue areas
  - In the TIME subsystem when updating the reference time from external time signals
  - In the ES subsystem for writing to the system log
- Some of these areas were relying on interrupt locks or thread priority levels to ensure shared memory access integrity
  - These approaches only work on single processors, as it assumes only one thread is running at any given time.
  - Needed improvement in order to be safe on an SMP platform where multiple threads can be truly running concurrently, including a low priority and high priority task

Introducing CCSDS Electronic Data Sheets (EDS)

- The CFE distribution now includes electronic data sheet files adhering to the XML schema described in CCSDS book 876.0 (SEDS).
  - These are machine-readable XML files describing the external interfaces (commands and telemetry) of the software
- These files are intended to feed into the various command and data dictionary solutions floating around
  - No need to write scripts to "scrape" the header files and/or documentation files to fill the command and data dictionary anymore, the XML files are intended for this purpose and will be maintained as such.
- Employ a more consistent "message handler" pattern for task pipe threads
  - All defined messages have a handler function, with a consistent prototype: accepts const pointer to message as input, returns int32 status
  - Allows all "task pipe" switch statements to be replaced by a single unified EDS-based message dispatcher function in a future version

Noteworthy Changes: Naming Conventions

- All identifier and macro names in code has been scrubbed to ensure naming consistency and correct scoping.
  - Applies to types and enumerations used in the "external-facing" interface (i.e. software bus)
  - The intent is to "pave the way" for using generated C header files, which requires a consistent and predictable mapping for identifier names between EDS and the source code.
- Scope-based configuration macro names
  - All macro names are prefixed with either `CFE_MISSION_` or `CFE_PLATFORM_` to clearly indicate the intended scope
  - Application code should transition away from using platform definitions to remain compatible with future versions of cFE.
- All software bus interface message definitions are now based on **mission-scope** definitions only.
  - Important because other CPUs might use a different number for platform scope
  - But the internal (platform) values are still allowed to be different as needed, only the external message needs to be a fixed size.

Naming Conventions and Backward Compatibility

- As desirable as clean and consistent identifier naming is, there is a risk of breaking existing code by changing names
- To help mitigate this, the existing identifier names are still preserved wherever possible
  - These are either `typedef`'ed (for types) or `#define`'ed (for macros) from the old name to the new name
  - All are conditionally excluded by a macro called `CFE_OMIT_DEPRECATED_6_6`
  - By defining this macro in your project, it is possible to get a preview of how your apps and libraries will compile in CFE 6.7 and beyond
- This feature is intended to assist in migration of apps and libraries
  - Not only does it allow checking if your apps are using any symbols that have changed names, but it clearly records what the old name was and what the new name is, easy search and replace to update code.

Noteworthy Changes: Pool Buffer Alignment

- Alignment of buffers from CFE ES Memory Pools is improved
  - This was previously only aligning buffers to 32 bit boundaries in all cases
  - Problematic on some platforms such as PowerPC where “double” types are used, which require 8 byte alignment.
- Now the alignment of buffers is adjusted based on the actual machine requirements
  - Based on how the cross compiler in use *actually* aligns a “`long long int`”, “`long double`”, or “`void*`” C99 type (whichever is greatest)
- Add a new platform configuration option to increase this to an even larger boundary as desired
  - For instance, on some CPUs, alignment to a cache line size (such as 32 bytes) can improve performance

Noteworthy changes: Short Format Events

- CFE 6.6 contains a better implementation of “Short Format” event messages
  - The original intent was to send the event metadata without the message string
  - However the historical implementation was just sending the entire message with the first character zeroed out
- To preserve compatibility with entities subscribing to the event message telemetry, short format messages are sent on a different message ID
  - Avoids a situation where existing code is expecting a message of a certain size and getting a smaller (short format) size in its place

Summary

- CFE 6.6 has just been finalized and introduces a bunch of important new features and fixes
  - The full set of fixes and improvements is in the version description document included with the release
  - The release notes cover known issues that may arise when upgrading from a previous release and how to resolve them
- CFE 6.6 makes a number of important changes to help pave the way for an even more modular and feature-rich CFE/CFS of future