Software Bus

Basecamp

August 2025

Audience & Prerequisites

- **Objectives**
  - Provide a comprehensive description of the cFS Framework Executive Service
- **Intended audience**
  - Software engineers developing with the cFS
- **Prerequisites**
  - cFS Framework Introduction
- **Refer to the Executive Services module for a list of topics covered in each cFE service training module**

Blue screens contain hands on cFS Basecamp exercises

- Basecamp is a lightweight environment with built-in tutorials for learning the cFS
- https://github.com/cfs-tools/cfs-basecamp

Software Bus Services Overview

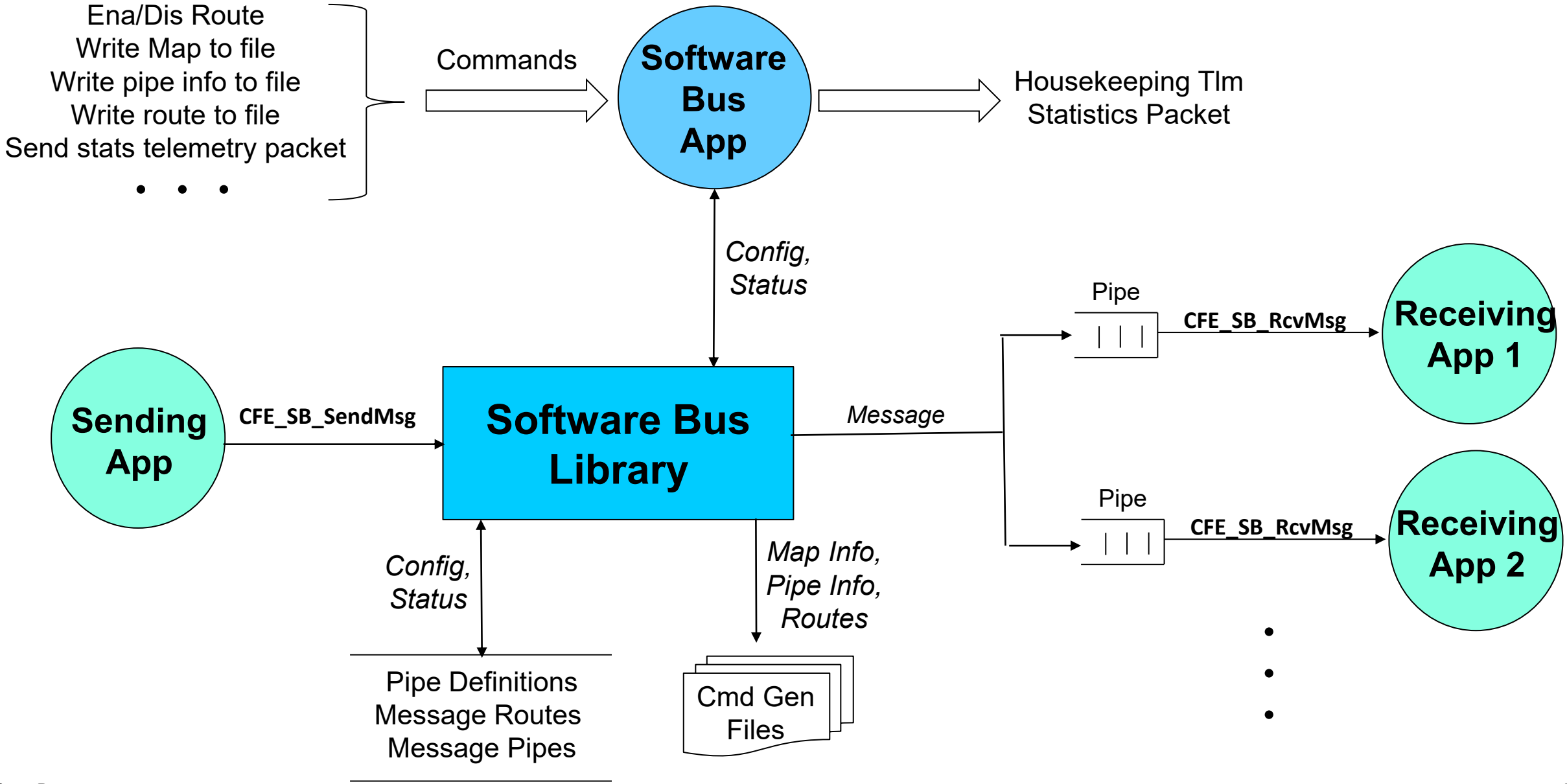
- **Provides an inter-application message service using a publish/subscribe model**
- **Routes messages to all applications that have subscribed to the message (i.e. broadcast model)**
  - Subscriptions are done at application startup
  - Message routing can be added/removed at runtime
  - Sender does not know who subscribes to their messages (i.e. connectionless)
- **Reports errors detected during the transferring of messages**
- **Outputs Statistics Packet and the Routing Information when commanded**
- **The Softwrae Bus Network application (not part of SB Services) can be used to extend the Software Bus across multiple processors**

Software Bus Terminology

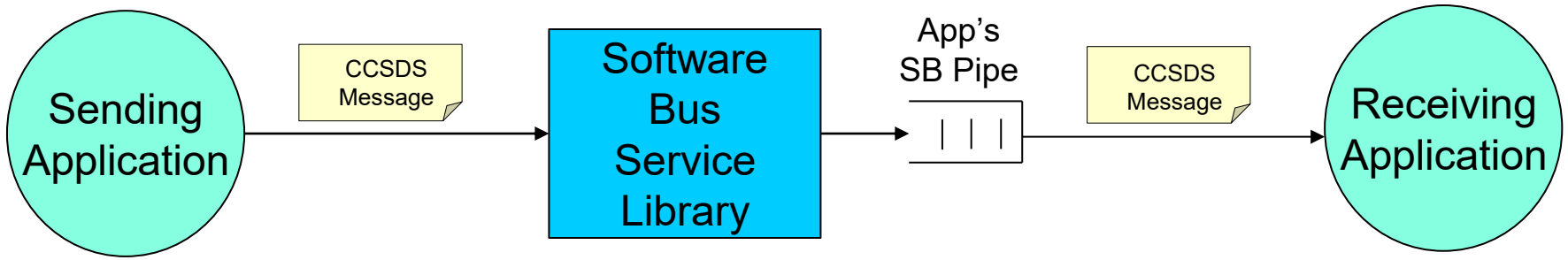
- **Pipe**
  - Destination to which SB Messages are sent
  - Queues that can hold SB Messages until they are dequeued and processed
- **Message**
  - A collection of typed data treated as a single entity
- **Buffer**
  - A contiguous block of binary data that is routed between applications
- **Applications receive *Buffers* and cast them to a specific *Message* type to use the data**

Software Bus Context

Basecamp



Software Bus Application Communication



- **Messages defined using the Consultative Committee for Space Data Systems (CCSDS) packet standard**

- **Applications create *SB Pipe (a FIFO queue)* and subscribe to receive messages**
  - Typically performed during application initialization
  - However apps can subscribe and unsubscribe to messages at any time

- ***SB Pipes* used for application data and control flow**
  - Poll and pend for messages

cFS Message Module

- **The Caelum release (September 2021) introduced the Message Module**
- **The Message Module encapsulates the definition of Messages that are routed by the Software Bus**
- **The Message Module provides an API for accessing message header information**
  - This design decouples messages from Software Bus routing buffers
  - The Software Bus queries Messages for information
  - The Message payload data is not interpreted by the Message Module

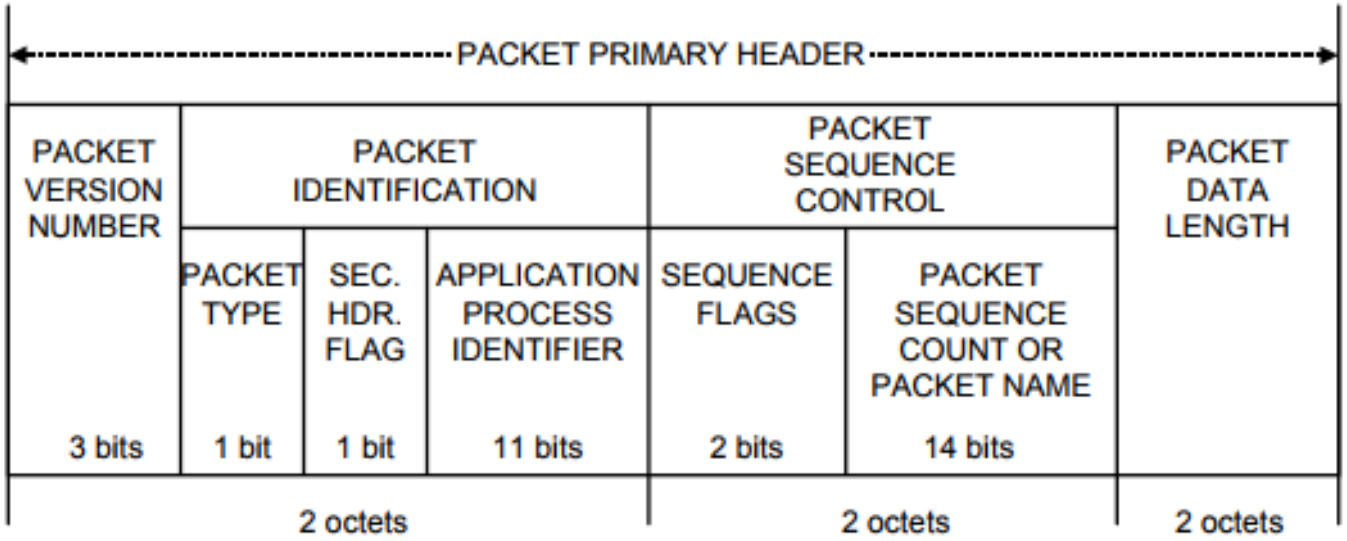
SB Messages (1 of 4)

- **By default, messages conform to the Consultative Committee for Space Data Systems (CCSDS) space packet standard**
  - In theory non-CCSDS formats could be used but that has not occurred in practice
  - Using CCSDS packets onboard simplifies data management when CCSDS standards are used for the flight-ground interface

- **Messages are routed by a *"MessageID"***
  - Message IDs should be treated as an abstract data type and referenced through the Message Module's API and not manipulated directly

- **By default, the Message Module provides two message implementations**
  - MISSION_MSG_V1 - Maps directly to the CCSDS Stream ID
  - MISSION_MSG_V2 – TODO (related to cFE 6.6 supports CCSDS extended APID that significantly increases the APID range)

CCSDS Packet Primary Header (Always big endian)



- **Packet Type**
  - 0: Telemetry
  - 1: Command
- **Secondary Header Flag**
  - 1: Secondary Header Present

- **CFE_MISSION_ES_CMD_MSG = 0x1806**
  - Cmd packet with secondary header
  - AppId = 6
- **CFE_MISSION_ES_HK_TLM_MSG = 0x0800**
  - Tlm packet with secondary header
  - AppId = 0

SB Messages (3 of 4)

- **“Packet” often used instead of “message” but not quite synonymous**
  - “Message ID” (first 16-bits) used to uniquely identify a message (also known as Stream ID)
  - “App ID” (11-bit) CCSDS packet identifier

- **CCSDS Command Packets**
  - Secondary packet header contains a command function code (or command code) and checksum
  - Apps define a single ground command message and use the function code to identify each command and dispatch a command processing function
  - Commands can originate from the ground or from onboard applications

- **CCSDS Telemetry Packets**
  - Secondary packet header contains a time stamp of when the data was produced
  - Telemetry is sent on the software bus by apps and can be ingested by other apps, stored onboard and sent to the ground

SB Messages (4 of 4)

Message formats are defined in the Message Module, along with functions to access message header fields (CFE_MSG_GetApId, CFE_MSG_GetSequenceCount, etc.)

```
union CFE_MSG_Message {
    CCSDS_SpacePacket_t CCSDS;
    uint8               Byte[sizeof(CCSDS_SpacePacket_t)];
}
struct CFE_MSG_CommandHeader {
    CFE_MSG_Message_t                Msg;
    CFE_MSG_CommandSecondaryHeader_t Sec;
}
struct CFE_MSG_TelemetryHeader {
    CFE_MSG_Message_t                  Msg;
    CFE_MSG_TelemetrySecondaryHeader_t Sec;
    uint8                              Spare[4];
}
```

Software Bus Reset Behavior

- **No data is preserved for either a Power-on or Processor Reset**
  - All routing is reestablished as applications create pipes and subscribe to messages
  - Any packet in transit at the time of the reset is discarded
  - All packet sequences are reset to 1

Retrieving Onboard State

- **Housekeeping Telemetry**
  - Counters: No subscribers, send errors, pipe overflows, etc.
  - Memory Stats
- **Telemetry packets generated by command**
  - Statistics
  - Subscription report
- **Files generated by command**
  - Routing information
  - Pipe information
  - Message ID to Routes

System Design: Guidelines

- **A system analysis and tuning effort should be performed to understand the message routing memory usage/margins and timing margins****
  - Many factors influence performance, such as SB configuration parameters, scheduler app's schedule, task priorities, pipe subscription depths, etc.
- **Helpful SB telemetry for tuning includes**
  - Housekeeping packet counters (No subscribers, send errors, pipe overflows, etc.) and memory stats
  - The telemetry statistics message sent via command shows pipe high water marks
- **Commands available to dump pipe, message and routing information to files that can be analyzed**
- **"Zero Copy" can be used for high speed message transfers typically between two apps**
  - Apps must manage acquiring and releasing message pointers
- **Telemetry message time stamps can be used monitored time between packets**
  - This is helpful when one app is using device data sent by an interrupt handler or another app

** The cFS System Engineering training module addresses system tuning

System Design : Message ID Management

- **TODO**
  - Discuss strategies for managing message IDs on a single or multiple processors
  - Message ID uniqueness
  - Topic IDs and message ID relationships (where are things going)

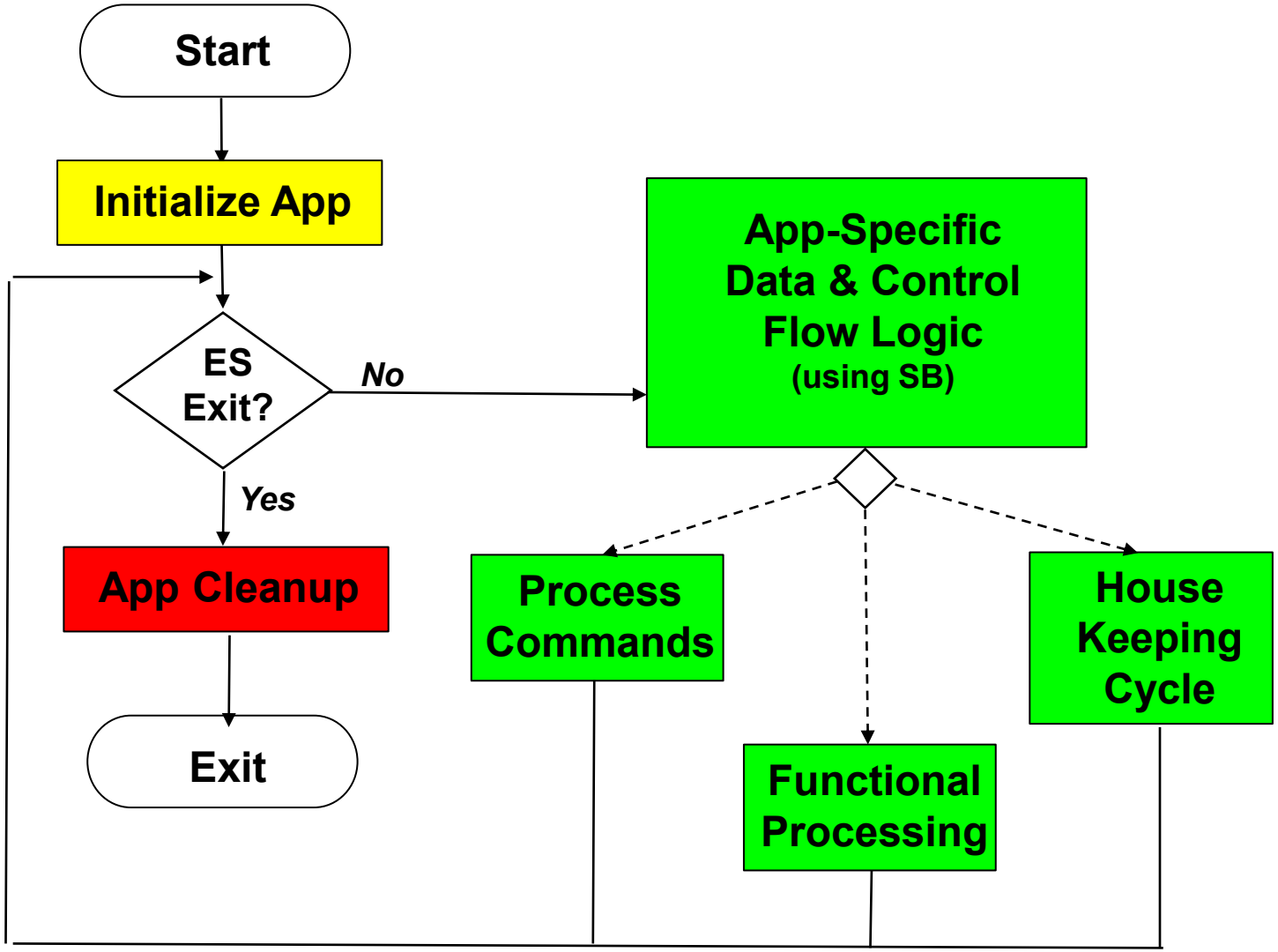
Application Development: Guidelines

- **SB message and buffer data structures should not be directly accessed**
  - API accessor functions should be used to read from and write to these structures

- **Applications can send commands**
  - Use *CFE_MSG_SetFcnCode()* to set the command function codes; requires the receiving app's header file with function code definitions
  - Use *CFE_MSG_GenerateChecksum()* to generate and load the command's checksum
  - For an example see Basecamp KIT_TO's event playback in file evt_plbk.c function *EVT_PLBK_Constructor()*

- **Packet sequence count increment/rollover is managed by *CFE_SB_TransmitMsg()* and is not an app's responsibility**

App Development: Example Control Flow



App Development: Example Control Flow SB Calls

Initialize App

CFE_SB_CreatePipe () – Create a pipe to receive messages; multiple pipes can be created

CFE_SB_Subscribe(), CFE_SB_SubscribeEx() – Subscribe to messages; **Ex** specifies Quality of Service & Message Limit

CFE_SB_InitMsg() – Initialize telemetry messages that are sent by the app

App-Specific Flow Logic

CFE_SB_ReceiveBuffer() – Poll, pend w/ timeout, or pend indefinitely

Process Commands

CFE_SB_GetUserDataLength() – Use if want to verified received length equals expected length

CFE_MSG_ValidateChecksum() – Use to verify command checksum; Assumes command source computed a checksum

CFE_MSG_GetFcnCode() – Obtain command function code (sometimes called command code) & dispatch corresponding function

House Keeping Cycle

CFE_SB_TimeStampMsg() – Set telemetry time to current system time

CFE_SB_TransmitMsg() – Send the message on the software bus

App Cleanup

CFE_ES_ExitApp() – Deallocates SB resources; special cases exist like releasing zero copy pointers/buffers

Software Bus APIs (1 of 3)

Pipe Management APIs	Purpose
CFE_SB_CreatePipe	Creates a new software bus pipe.
CFE_SB_DeletePipe	Delete a software bus pipe.
CFE_SB_PipeId_ToIndex	Obtain an index value correlating to an SB Pipe ID
CFE_SB_SetPipeOpts	Set options on a pipe.
CFE_SB_GetPipeOpts	Get options on a pipe.
CFE_SB_GetPipeName	Get the pipe name for a given id.
CFE_SB_GetPipeIdByName	Get pipe id by pipe name.

Message Subscription Control APIs	Purpose
CFE_SB_SubscribeEx	Subscribe to a message on the software bus
CFE_SB_Subscribe	Subscribe to a message on the software bus with default parameters
CFE_SB_SubscribeLocal	Subscribe to a message while keeping the request local to a CPU
CFE_SB_Unsubscribe	Remove a subscription to a message on the software bus
CFE_SB_UnsubscribeLocal	Remove a subscription to a message on the software bus on the current CPU

Software Bus APIs (2 of 3)

Send/Receive Message APIs	Purpose
CFE_SB_TransmitMsg	Transmit a message
CFE_SB_ReceiveBuffer	Receive a message from a software bus pipe

Zero Copy APIs	Purpose
CFE_SB_AllocateMessageBuffer	Get a buffer pointer to use for "zero copy" SB sends.
CFE_SB_ReleaseMessageBuffer	Release an unused "zero copy" buffer pointer.
CFE_SB_TransmitBuffer	Transmit a buffer

Message Characteristics APIs	Purpose
CFE_SB_SetUserDataLength	Sets the length of user data in a software bus message.
CFE_SB_TimeStampMsg	Sets the time field in a software bus message with the current spacecraft time.
CFE_SB_MessageStringSet	Copies a string into a software bus message
CFE_SB_GetUserData	Get a pointer to the user data portion of a software bus message.
CFE_SB_GetUserDataLength	Gets the length of user data in a software bus message.
CFE_SB_MessageStringGet	Copies a string out of a software bus message

Software Bus APIs (3 of 3)

Message ID APIs	Purpose
CFE_SB_IsValidMsgId	Identifies whether a given CFE_SB_MsgId_t is valid
CFE_SB_MsgId_Equal	Identifies whether two #CFE_SB_MsgId_t values are equal
CFE_SB_MsgIdToValue	Converts a #CFE_SB_MsgId_t to a normal integer
CFE_SB_ValueToMsgId	Converts a normal integer into a #CFE_SB_MsgId_t

Message Module APIs (1 of 3)

Generic Message APIs	Purpose
CFE_MSG_Init	Initialize a message

Message Primary Header APIs	Purpose
CFE_MSG_GetSize	Gets the total size of a message.
CFE_MSG_SetSize	Sets the total size of a message.
CFE_MSG_GetType	Gets the message type.
CFE_MSG_SetType	Sets the message type.
CFE_MSG_GetHeaderVersion	Gets the message header version.
CFE_MSG_SetHeaderVersion	Sets the message header version.
CFE_MSG_GetHasSecondaryHeader	Gets the message secondary header boolean
CFE_MSG_SetHasSecondaryHeader	Sets the message secondary header boolean
CFE_MSG_GetApId	Gets the message application ID
CFE_MSG_SetApId	Sets the message application ID
CFE_MSG_GetSegmentationFlag	Gets the message segmentation flag
CFE_MSG_SetSegmentationFlag	Sets the message segmentation flag
CFE_MSG_GetSequenceCount	Gets the message sequence count
CFE_MSG_SetSequenceCount	Sets the message sequence count
CFE_MSG_GetNextSequenceCount	Gets the next sequence count value (rolls over if appropriate)

Message Module APIs (2 of 3)

Message Extended Header APIs	Purpose
CFE_MSG_GetEDSVersion	Gets the message EDS version
CFE_MSG_SetEDSVersion	Sets the message EDS version
CFE_MSG_GetEndian	Gets the message endian
CFE_MSG_SetEndian	Sets the message endian
CFE_MSG_GetPlaybackFlag	Gets the message playback flag
CFE_MSG_SetPlaybackFlag	Sets the message playback flag
CFE_MSG_GetSubsystem	Gets the message subsystem
CFE_MSG_SetSubsystem	Sets the message subsystem
CFE_MSG_GetSystem	Gets the message system
CFE_MSG_SetSystem	Sets the message system

Message Secondary Header APIs	Purpose
CFE_MSG_GenerateChecksum	Calculates and sets the checksum of a message
CFE_MSG_ValidateChecksum	Validates the checksum of a message.
CFE_MSG_SetFcnCode	Sets the function code field in a message.
CFE_MSG_GetFcnCode	Gets the function code field from a message.
CFE_MSG_GetMsgTime	Gets the time field from a message.
CFE_MSG_SetMsgTime	Sets the time field in a message.

Message Module APIs (3 of 3)

Message Id APIs	Purpose
CFE_MSG_GetMsgId	Gets the message id from a message.
CFE_MSG_SetMsgId	Sets the message id bits in a message.
CFE_MSG_GetTypeFromMsgId	Gets message type using message ID

Command List

SB Command List	Purpose
CFE_SB_NoopCmd	Software Bus No-Op
CFE_SB_ResetCountersCmd	Resets counters in the Software Bus housekeeping telemetry
CFE_SB_EnableSubReportingCmd	Enable Subscription Reporting Command
CFE_SB_DisableSubReportingCmd	Disable Subscription Reporting Command
CFE_SB_SendHKTlmCmd	Function to send the SB housekeeping packet
CFE_SB_EnableRouteCmd	Enable Software Bus Route
CFE_SB_DisableRouteCmd	Disable Software Bus Route
CFE_SB_SendStatsCmd	Send Software Bus Statistics
CFE_SB_WriteRoutingInfoCmd	Write Software Bus Routing Info to a File
CFE_SB_WritePipeInfoCmd	Write Pipe Info to a File
CFE_SB_WriteMapInfoCmd	Write Map Info to a File
CFE_SB_SendPrevSubsCmd	Generates a series of packets that contain information regarding all subscriptions previously received by SB.

Platform Configuration Parameters

Parameter	Purpose
CFE_PLATFORM_SB_MAX_MSG_IDS	Maximum Number of Unique Message IDs SB Routing Table can hold
CFE_PLATFORM_SB_MAX_PIPES	Maximum Number of Unique Pipes SB Routing Table can hold
CFE_PLATFORM_SB_MAX_DEST_PER_PKT	Maximum Number of unique local destinations a single MsgId can have
CFE_PLATFORM_SB_DEFAULT_MSG_LIMIT	Default Subscription Message Limit
CFE_PLATFORM_SB_BUF_MEMORY_BYTES	Size of the SB buffer memory pool
CFE_PLATFORM_SB_HIGHEST_VALID_MSGID	Highest Valid Message Id
CFE_PLATFORM_SB_DEFAULT_ROUTING_FILENAME	Default Routing Information Filename
CFE_PLATFORM_SB_DEFAULT_PIPE_FILENAME	Default Pipe Information Filename
CFE_PLATFORM_SB_DEFAULT_MAP_FILENAME	Default Message Map Filename
CFE_PLATFORM_SB_FILTERED_EVENT[1-8]	SB Event Filtering
CFE_PLATFORM_SB_FILTER_MASK[1-8]	SB Event Filtering Mask
CFE_PLATFORM_SB_MEM_BLOCK_SIZE_[01-16]	Define SB Memory Pool Block Sizes
CFE_PLATFORM_SB_MAX_BLOCK_SIZE	Defines Max SB Memory Pool Block Size
CFE_PLATFORM_SB_START_TASK_PRIORITY	SB Task Priority
CFE_PLATFORM_SB_START_TASK_STACK_SIZE	SB Task Stack Size

Mission Configuration Parameters

Parameter	Purpose
CFE_MISSION_SB_MAX_SB_MSG_SIZE	Maximum SB Message Size
CFE_MISSION_SB_MAX_PIPES	Maximum Number of pipes that SB command/telemetry messages may hold

Software Bus Exercises - 1

1. TODO: Decide whether current overview exercise should be moved here and a new simpler overview exercise created
2. TODO: SB queue over flow fix
3. TODO: Write routing map. See NASA's exercises