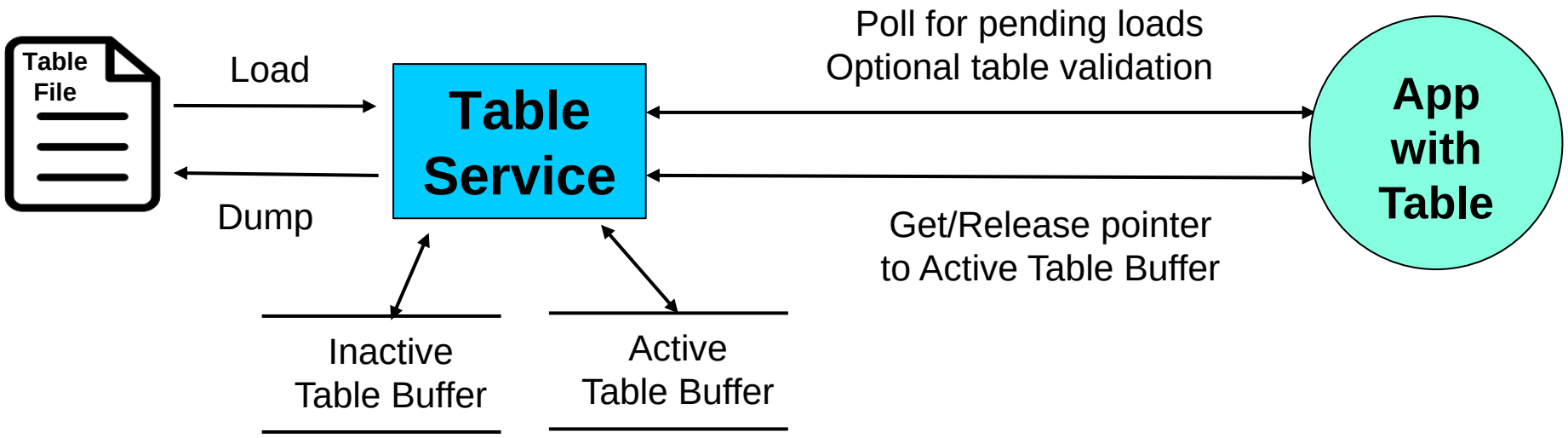

cFE
Table Service (TBL)
Tutorial
OSK v3.1

Table (TBL) Service Overview

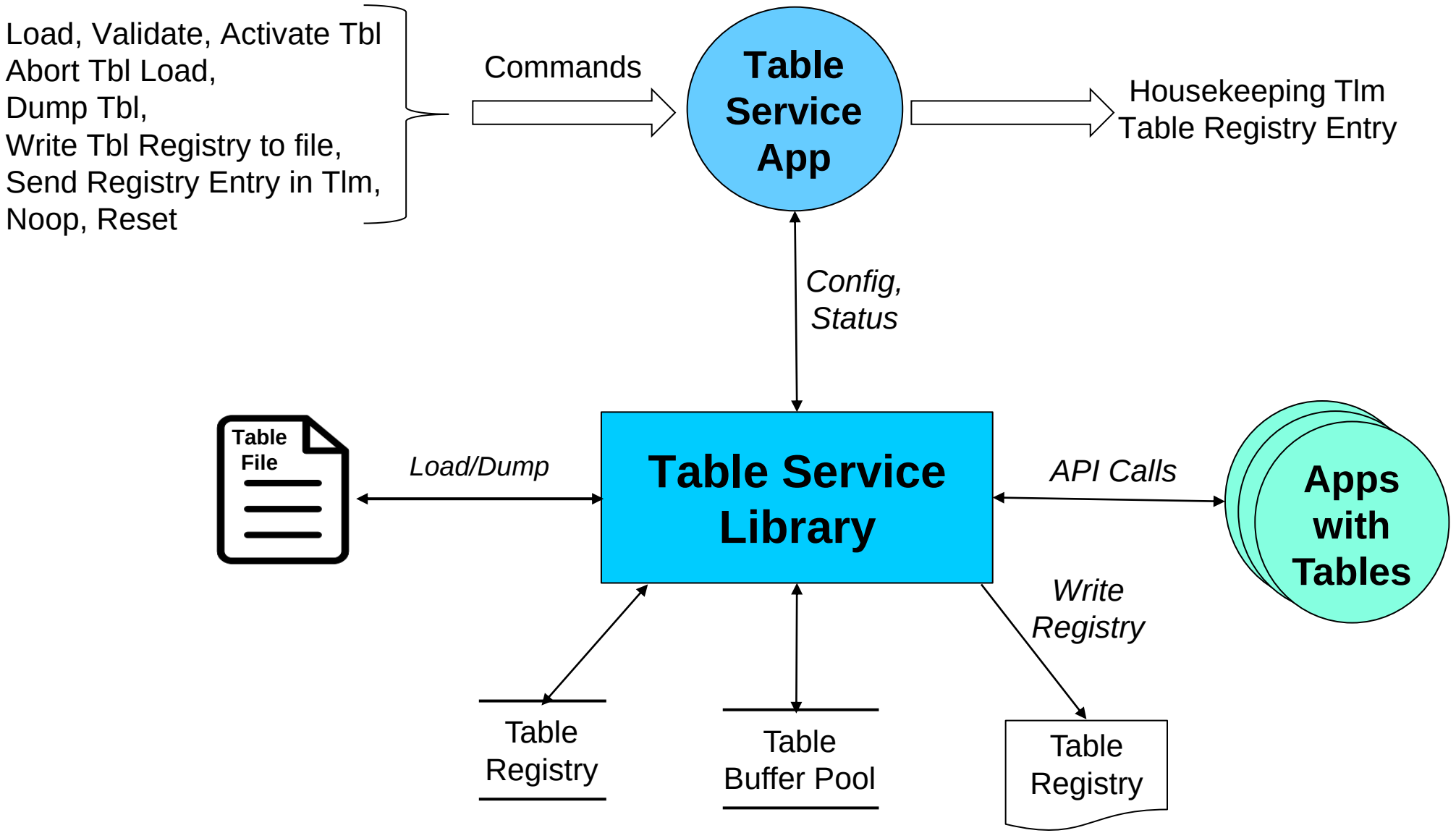
- **What is a table?**
  - Tables are logical groups of parameters that are managed as a named entity
- **Parameters typically change the behavior of a FSW algorithm**
  - Examples include controller gains, conversion factors, and filter algorithm parameters
- **Tables service provides ground commands to load a table from a file and dump a table to a file**
  - Table loads are synchronized with applications
- **Tables are binary files**
  - Ground support tools are required to create and display table contents
- **The cFE can be built without table support**
  - Note the cFE applications don't use tables

Table Service Functional Overview

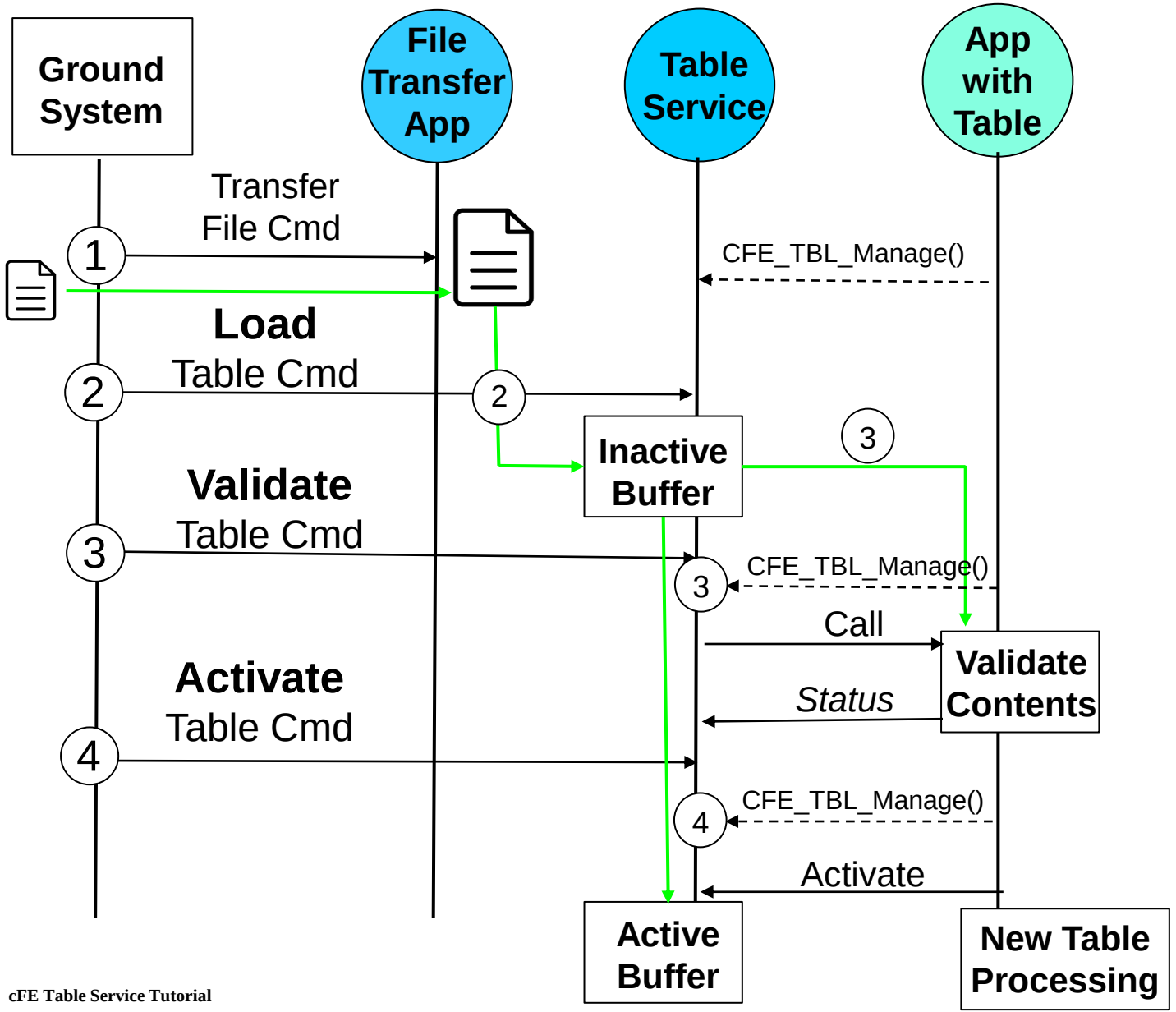


- **Table service contains buffers that hold tables for all applications**
  - Active Table Buffer - Image accessed by app while it executes
  - Inactive Table Buffer - Image manipulated by ops (could be stored commands)
- **"Table Load" is a sequence of activities to transfer data from a file to the Active Table Buffer**
- **"Table Dump" is a sequence of activities to transfer data from a either Table Buffer to a file**
- **Table operations are synchronous with the application that owns the table to ensure table data integrity**

Table Service Context



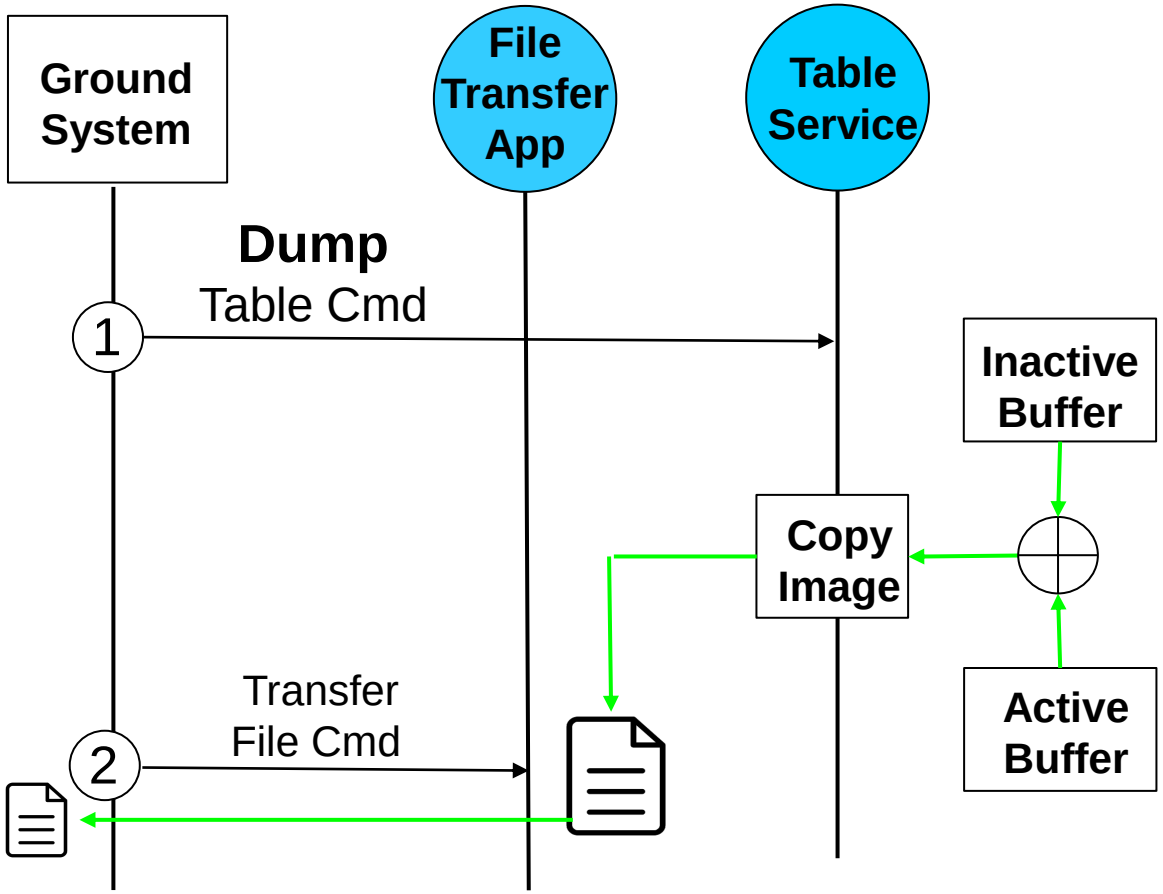
Load Table Sequence Diagram



1. **Transfer File Command**
   a. Transfer table image file from ground to flight
   b. Multiple table files can be stored onboard
2. **Load Table Command**
   a. Table Service copies file image to Inactive Buffer
   b. Loads can be partial or complete
   c. For partial loads Active Buffer contents copied to Inactive Buffer prior to updates from file (not shown)
3. **Validate Table Command***
   a. Apps validate the contents of a table image prior to table service accepting an activate command
4. **Activate Table Command***
   a. Apps initiate copy from Inactive to Active Buffer
   b. Apps may need to perform one-time functions when a new table is loaded
   c. Non-Blocking table updates allow tables to be used in Interrupt Service Routines

* Apps typically poll table services during their "housekeeping" execution cycle

Dump Table



1. **Dump Table Command**
   a. Copy either Inactive Buffer or Active Buffer to a file
2. **Transfer File**
   a. Transfer the file from flight to ground

Table Buffering Options

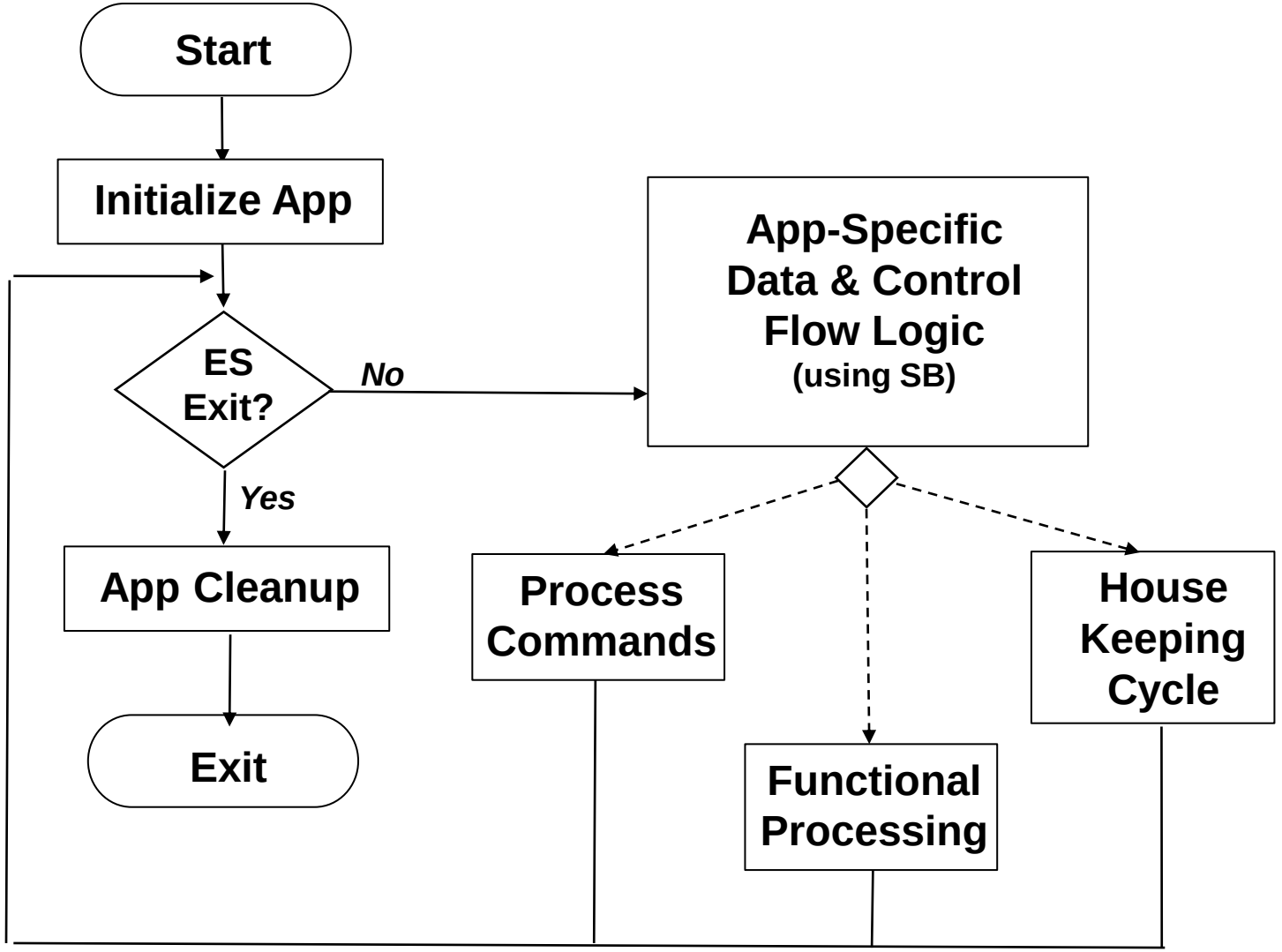
- **Single Buffer**
  - The Active Buffer is the only buffer dedicated to an application's table
  - Table service shares Inactive Buffers to service multiple app's with single buffer tables
    - Pool of fixed sized buffers that must accommodate the largest single buffer image
    - CFE_TBL_MAX_SIMULTANEOUS_LOADS defines the number of concurrent table load sessions
  - Most efficient use of memory and adequate for most situations

- **Double Buffer**
  - Dedicated Inactive image for each double buffered table
  - Useful for fast table image swaps (.e.g. high rate app and/or very large table) and delayed activation of table's content (e.g. ephemeris)
  - E.g. Stored Command's Absolute Time Command table

Table Attributes

- **Validation Function**
  - Applications register validation functions during initialization
  - Table activates for tables with validation functions will be rejected if the validation has not been performed
  - Mission critical data table values are usually verified
- **Critical Tables**
  - Table data is stored in a Critical Data Store
  - Contents updated for each table activate command
- **User Defined Address**
  - Application provides the memory address for the active table buffer
  - Typically used in combination with a dump-only table
- **Dump-Only**
  - Contents can’t be changed via the load/validate/activate sequence
  - The dump is controlled by the application that owns the table so it can synchronize the dump and avoid dumps that contain partial updates

Common Application Table Service Usage



- Initialize App
  - CFE_TBL_Register()
  - CFE_TBL_Load()
  - CFE_TBL_GetAddress()
- Command/Functional Processing
  - CFE_TBL_Modified()
- Housekeeping Cycle
  - CFE_TBL_ReleaseAddress()
    - CFE_TBL_Manage()

      or
    - CFE_TBL_GetStatus()
    - CFE_TBL_Validate()
    - CFE_TBL_Update()
  - CFE_TBL_GetAddress()
- App Cleanup
  - CFE_TBL_Unregister()

Table Service API

Application Functions	Purpose
CFE_TBL_Register	Registers a new table
CFE_TBL_Unregister	Unregister a table and release its resources
CFE_TBL_Load	Initialize or update the contents of a table from memory or a file
CFE_TBL_Share	Get a handle to a table that was created by another application
CFE_TBL_GetAddress	Get the address of a table (locks the table)
CFE_TBL_GetAddresses	Get the address of a collection of tables (locks the tables)
CFE_TBL_ReleaseAddress	Release a table address (unlocks the table). Must be done periodically by the cFE Application that owns the table in order to allow updates to the tables
CFE_TBL_ReleaseAddresses	Release an array of table address (unlocks the tables)
CFE_TBL_GetStatus	Returns the status on the specified table regarding validation or update requests
CFE_TBL_Validate	Performs the registered validation function for the specified table and reports the success/failure to the operator via Table Services Housekeeping Telemetry and Event Messages.
CFE_TBL_Update	Update table contents with new data if an update is pending
CFE_TBL_Manage	Performs routine actions to manage the specified table. This includes performing any necessary table updates or table validations
CFE_TBL_GetInfo	Provides information about the specified table including size, last time updated etc.
CFE_TBL_DumpToBuffer	Copy Dump Only table to buffer for later dump to file by table services
CFE_TBL_Modified	Notify TBL Services that the contents of the table has been modified by the application
CFE_TBL_NotifyByMessage	Instruct TBL Services to notify calling application whenever the specified table requires management.

System Considerations (1 of 2)

- **Commands are typically used to initiate an action; not tables**
  - For example, commands are used to change the spacecraft control mode and control mode gains are defined in a table
- **Sometimes convenience commands are provided to change table elements**
  - For example, scheduler app provides an enable/disable scheduler table entry
- **Tables do not typically contain dynamic data computed by the FSW**
  - The cFE doesn't preclude this and tables have been used as a convenient method to collect data, save to a file, and transfer it to the ground
  - These are defined as dump-only tables
- **The checksum app can be used to verify the contents of static tables don't change**
- **Tables can be shared between applications but this is rare**
  - Tables are not intended to be an inter-application communication mechanism

System Considerations (2 of 2)

- **Most tables can be loaded & dumped and are single buffered**
  - A convenience macro *CFE_TBL_OPT_DEFAULT* is defined for these defaults
- **The *CFE_TBL_NotifyByMessage()* API allows an application to be notified by a software bus message when a table requires managing**
  - Avoids the need for an application to poll table services
- **Double buffering is useful for fast table image swaps (.e.g. high rate app and/or very large table) and delayed activation of table's content (e.g. absolute time stored commands)**
- **Table load/dump files are binary**
  - Ground tools are required to create and display table contents
  - The binary table files contain the following three sections:

cFE File Header (cfe_fs_extern_typedefs.h : CFE_FS_Header_t)
Table Header (cfe_tbl_ extern_typedefs.h: CFE_TBL_File_Hdr_t)
Table Data (Application specific)

Table Services – Reset Behavior

- **Table registry is cleared for power-on and processor resets**
  - Applications must always register and initialize their non-critical table data during their initialization

- **Critical Tables**
  - If a table is registered as critical then during a processor reset table service will use Executive Services to locate and load the preserved table data from a critical data store

Retrieving Onboard State

- **Housekeeping Telemetry**
  - Table registry statistics (number of tables and pending loads)
  - Last table validation results (CRC, validation status, total validations)
  - Last updated table
  - Last file loaded
  - Last file dumped – Last table loaded
- **Telemeter Application Registry**
  - Telemeter the Table Registry contents for the command-specified table
- **Dump Table Registry**
  - Write the pertinent table registry information to the command-specified file.
  - For each table
    - Owner App ID, table name, size in bytes, attributes
    - Pointers to Active Buffer and Inactive Buffer (if double buffered)
    - Pointer to Validation function
    - Detailed table load and dump information

Configuration Parameters

Parameter	Purpose
CFE_PLATFORM_TBL_BUF_MEMORY_BYTES	Size of Table Services Table Memory Pool
CFE_PLATFORM_TBL_MAX_DBL_TABLE_SIZE	Maximum Size Allowed for a Double Buffered Table
CFE_PLATFORM_TBL_MAX_SNGL_TABLE_SIZE	Maximum Size Allowed for a Single Buffered Table
CFE_PLATFORM_TBL_MAX_NUM_TABLES	Maximum Number of Tables Allowed to be Registered
CFE_PLATFORM_TBL_MAX_CRITICAL_TABLES	Maximum Number of Critical Tables that can be Registered
CFE_PLATFORM_TBL_MAX_NUM_HANDLES	Maximum Number of Table Handles
CFE_PLATFORM_TBL_MAX_SIMULTANEOUS_LOADS	Maximum Number of Simultaneous Loads to Support
CFE_PLATFORM_TBL_MAX_NUM_VALIDATIONS	Maximum Number of Simultaneous Table Validations
CFE_PLATFORM_TBL_DEFAULT_REG_DUMP_FILE	Default Filename for a Table Registry Dump
CFE_PLATFORM_TBL_VALID_SCID_COUNT	Number of Spacecraft ID's specified for validation
CFE_PLATFORM_TBL_U32FROM4CHARS	Macro to construct 32 bit value from 4 chars
CFE_PLATFORM_TBL_VALID_SCID_[1-2]	Spacecraft ID values used for table load validation
CFE_PLATFORM_TBL_VALID_PRID_COUNT	Number of Processor ID's specified for validation
CFE_PLATFORM_TBL_VALID_PRID_[1-4]	Processor ID values used for table load validation