Mission Name

Core Flight System (CFS)
Limit Checker Extended (LCX) Version 2.0.0.0
Supplement to LC Version 2.2.1.0
APPLICATION USER'S GUIDE SUPPLEMENT

Flight Software Systems Branch – Code 582
Release Version 1.0 – 9/19/12
582-2012-005

Goddard Space Flight Center

Greenbelt, Maryland

National Aeronautics and
Space Administration

FORWORD

This Core Flight Software (CFS) Limit Checker Extended (LC) Application User's Guide *Supplement* provides guidance for the Flight Operations Team (FOT) for the CFS LC Application.

Sections not included in this supplement that appear in the LC User's Guide also apply to LCX.

AUTHOR

9/19/2012

Gary M Smith
Technical Writer

Date

APPROVALS

9/26/2012

Susanne Strege / 582
Core Flight Executive (cFE) Core Flight System (CFS) / PDL

Date

9-28-2012

Charles Wildermann / 582
Flight Software Systems Branch / Head

Date

UPDATE HISTORY

Version	Date	Description	Affected Pages
Draft 0.1	09/04/12	1st Draft	All
1.0	09/19/12	Release Version	All

CONTENTS

TABLE OF FIGURES

TABLES

This page intentionally blank.

Chapter 1. Introduction to the CFS LCX User's Guide Supplement

1.1 *Purpose and Scope of this Guide Supplement*

The purpose of this Application User's Guide *Supplement* is to help the Flight Operations Team (FOT) understand the Core Flight System (CFS) Limit Checker Extended (LCX) application.

While Limit Checker (LC) is not covered in this document, LCX and LC are very close in design. This *Supplement* shows the differences between LCX and LC, and is intended to be used along with the LC Application User's Guide. Sections not included in this *Supplement* that appear in the LC User's Guide also apply to LCX.

1.1.1 Changes from LC User's Guide

Summarized below are selected, key changes between the figures and tables appearing in the LC User's Guide and the LCX User's Guide *Supplement*. LC figures not appearing in the LCX Users Guide are unchanged and remain valid for LCX.

1.1.1.1 Changes to Figures from the LC Document

- In Figure 1, LC/LCX - Flow Control, on page 2-3, references to "Not Measured" were replaced with "Stale".
- In Figure 4, LCX - Flow Detail (B1) – Actionpoint Command on page 2-6, before the start of the loop for each actionpoint, (B2) was inserted.
- A new chart, Figure 5, LCX - Flow Detail (B2) – Update WP Stale Counters, on page 2-7, was added to update the age of WP Results.
- In all figures, the color yellow means a change in program flow between the LC and LCX applications.

1.1.1.2 Changes to Tables from the LC Document

Summarized below are selected changes between the tables appearing in the LC User's Guide and the LCX User's Guide *Supplement*. LC tables not appearing in the LCX Users Guide are unchanged and remain valid for LCX. A few minor supporting document tables included for convenience are not mentioned in this section.

1.1.1.2.1 *Internal Messaging*

- To the AP Sample Command, an option was added to update the age of WP results. ***See also:*** Table 7, Internal Messages, on page A-2.

1.1.1.2.2 *Watchpoint Definition Table*

- To the Watchpoint Definition Table, was inserted "Age When Watchdog Comparison Result Becomes Stale".
- ***See also:*** Section 2.3.1, LCX Tables in Support of Watchpoints and Actionpoints, on page 2-2; Section 2.3.2.1, Watchpoint Table Enhancements, on page 2-8; and Table 9, Watchpoint Definition Table (WDT) Entry, on page A-3.

1.1.1.2.3 *Watchpoint Results Table*

- In the Watchpoint Results Table, "Not Measured" references were replaced with "Stale".
- ***See also:*** Section 2.3.2.1, Watchpoint Table Enhancements, on page 2-8; Question 5.2.1, When would a watchpoint evaluate to Stale? on page 5-2; Table 12, Watchpoint Results Table (WRT) Entry, on page A-5; Table 3 Nomenclature Differences: Watchpoint (WP) and Actionpoint (AP) on page A-1; and Table 4 Nomenclature Differences: Housekeeping (WP and AP Results) on page A-1.

1.1.1.2.4 *Actionpoint Results Table*

- In the Actionpoint Results Table, the term "Not Measured" was replaced with the term "Stale".
- ***See also:*** Section 2.3.2.4, Actionpoint Result Transition to Stale, on page 2-9; Question 5.2.3, When would an actionpoint evaluate to Stale? on page 5-2; Table 13, Actionpoint Results Table (ART) Entry, on page A-6; Table 3 Nomenclature Differences: Watchpoint (WP) and Actionpoint (AP) on page A-1; and Table 4 Nomenclature Differences: Housekeeping (WP and AP Results) on page A-1.

1.1.1.2.5 *AP Sample Request*

- To "AP Sample Request" was added the option to update age of WP results.
- ***See also****:* Table 6, Watchpoint (WP) Age Results Option in Actionpoint (AP) Sample Command, on page A-2; and Table 13, Actionpoint Results Table (ART) Entry, on page A-6.

1.2 *Acknowledgements*

This Application User's Guide *Supplement* relies heavily on the content of LCX design presentations and requirements documents, interviews with flight software engineers, and earlier heritage LC publications and presentations. Appendix A is based on information from CFS LCX source code.

1.3 *Conventions*

- In this document, the percent sign (%), when followed by a string, may indicate variable text. See Appendix A for references to the text that may be substituted in each case.
- Core Flight Executive is abbreviated cFE (lower case "c" with uppercase "FE").

- In this document, the following are used interchangeably: "Result table" and "Results table"; flight controller, ground controller, and the Flight Operations Team (FOT); Limit Checker Application User's Guide, and LC User's Guide.
- *See Appendix C for more.*

1.4 *Related Documents*

Documents used in the preparation of this Guide *Supplement* are listed in the table below.

Table 1 Related Documents

Item No.	Document ID	Document Source
1	N/A	*Core Flight System (CFS) Limit Checker (LC) Application User's Guide Draft.* Greenbelt, MD: Goddard Space Flight Center, Code 582 (Flight Software Branch), 4 Sept 2012. DOCX.
2	N/A	Walling, Scott. *CFS_LCX_Design_2012_07_25 [LCX Design Presentation].* Greenbelt, MD: Goddard Space Flight Center, Code 582 (Flight Software Branch), 25 July 2012. PPTX.
3	N/A	*CFS-LCX-ReqWorkingDraft-D [CFS LCX Requirements – Draft D].* Greenbelt: NASA Goddard Space Flight Center, Code 582, Flight Software Systems Branch, 21 June 2012. DOCX.
4	N/A	Strege, Susanne L. *CFS-LCX-DoxUsersGuide*. Greenbelt: NASA Goddard Space Flight Center, Code 582, CFS Product Development Team, 23 August 2012. Doxygen Output via RTF.
5	N/A	Walling, Scott. *Diff [diff file of LC and LCX .h and .c files]*. Greenbelt, MD: Goddard Space Flight Center, Code 582 (Flight Software Branch), 24 August 2012. TXT.

1.5 *Assumptions*

1.5.1 Personnel

This Application User's Guide *Supplement* assumes the primary reader is either a new or long term member of the Flight Operations Team (FOT) at NASA or the equivalent elsewhere.

1.5.2 CFS LC Application

The following list summarizes the assumptions made about the CFS Limit Checker Application as documented in this Guide *Supplement*:

- The mission is using LCX instead of LC.
- All other assumptions listed in the CFS LC User's Guide remain unchanged.

1.6 How to Use this Document

Use this *Supplement* alongside the Core Flight System (CFS) Limit Checker (LC) Application User's Guide to show the differences between the LCX and LC applications.

1.7 Acronyms and Abbreviations

Acronyms and abbreviations in this publication are shown in Table 2 below. Telemetry and command mnemonics and similar terms, are not shown here.

Table 2 Acronyms and Abbreviations

Acronym/Abbreviation	Description
ADT	Actionpoint Definition Table
AP	Actionpoint
ART	Actionpoint Results Table
ASIST	Advanced Spacecraft Integration & System Test software
CC	Command Code
cFE	Core Flight Executive
CFS	Core Flight System
CI	Command Ingest Application
CM	Configuration Management
DWord	32 bit signed double word
FOT	Flight Operations Team
FSSE	Flight Software Sustaining Engineering
FSW	Flight Software
HK	Housekeeping Application
LC	Limit Checker Application
LCX	Limit Checker Extended Application
MID	Message ID
MMS	Magnetospheric Multiscale Mission
MOT	Mission Operations Team
MS	Microsoft
Oper	Operator
PDF	Portable Document Format
PDL	Project Development Lead
RDL	Record Definition Language
RPN	Reverse Polish Notation; also known as Postfix notation
RTF	Rich Text Format

Acronym/Abbreviation	Description
RTS	Relative Time tagged command Sequence
SC	Stored Command Application
SCH	Scheduler Application
SDO	Solar Dynamics Observatory
TBL	Table Services application
Tbl	Table
TLM	Telemetry
UByte	8 bit unsigned byte
UDWord	32 bit unsigned double word
uint16	Unsigned 16-bit integer
uint32	Unsigned 32-bit integer
uint8	Unsigned 8-bit integer
UWord	16 bit signed word
WDT	Watchpoint Definition Table
WP	Watchpoint
WRT	Watchpoint Results Table

This page intentionally blank.

Chapter 2. Introduction to the CFS LC Extended (LCX) Application

2.1 *Heritage*

In the spring and summer of 2012, the Magnetospheric Multiscale Mission (MMS) requested additional functionality to the CFS LC application. Those changes were deemed significant enough to give the application a different name, Limit Checker Extended (LCX).

There are separate/independent code paths for LC and LCX. Future missions that ask for LC will take the code from the LC repository. Future missions that ask for LCX will take the code from the LCX repository.

For other heritage details, please see the LC User's Guide.

2.2 *Differences with Limit Checker (LC)*

2.2.1 LCX is the Same Design as LC, with One Enhancement.

- LCX defaults to the application name LC.
- Its mnemonics are the same as LC.
- LCX cannot be used in addition to LC; LCX must replace LC.

2.2.2 LCX Adds Support for Stale Watchpoint/Actionpoint Results

- LCX monitors the age of each watchpoint result. The WP result is changed to Stale when the age of the result exceeds the table defined limit. The WP result age is measured as the number of Sample Actionpoints commands.
- LCX introduces a new argument to the internal Sample Actionpoints command. The existing argument defines which actionpoints to sample (all vs. some). The new argument signals LCX to update watchpoint result stale counters. The design separates the WP result age from selected actionpoint samples.
- In LCX, the LC watchpoint and actionpoint results term "Not Measured" has been replaced with the LCX term "Stale".

2.3 *CFS Limit Checker Extended (LCX) Overview*

The CFS Limit Checker Extended (LCX) application monitors telemetry data points in the flight system and compares the values against predefined or computed threshold limits. When a threshold condition is encountered, an event message is issued and a Relative Time Sequence command script may be initiated to respond to the threshold violation.

CFS LCX is responsible for monitoring watchpoints and evaluating actionpoints. LCX is table driven but also responds to ground commands.

Each **watchpoint** compares a telemetry data value with a predefined constant threshold limit in the normal case. By using a custom function the threshold can be a computed value.

The comparison result may be:

- True
- False
- Error, or
- Stale

Each **actionpoint** analyzes the results of one (or more) watchpoints. Analysis result may be:

- Pass
- Fail
- Error, or
- Stale

If the number of consecutive fails exceeds a preset limit, then LCX sends an event and optionally invokes an RTS. No RTS or event is issued until the Actionpoint expression is evaluated based on the current watchpoint states.

2.3.1 LCX Tables in Support of Watchpoints and Actionpoints

LCX is a table driven application but can also accept commands. LCX uses Watchpoint and Actionpoint **definition** tables for characterizing mission specific telemetry limits and actions, and Watchpoint and Actionpoint **result** tables for statistics.

Definition Tables

1. Watchpoint Definition Table (WDT) – defines the data to be evaluated, such as, for example, message ID, offset, comparison value, **and age when watchpoint comparison result becomes stale.**
2. Actionpoint Definition Table (ADT) – defines the equations used to evaluate watchpoint states and the actions to be taken

Result Tables

1. Watchpoint Results Tables – contain the results of the watchpoint evaluations defined in the WDT.
2. Actionpoint Results Tables – contain the results of the actionpoint evaluations defined in the ADT.

Figure 1 below shows the overall program flow of the CFS LCX application, from start to exit. The B & C circles are highlighted in yellow to indicate a difference in flow control between the LCX and LC applications.

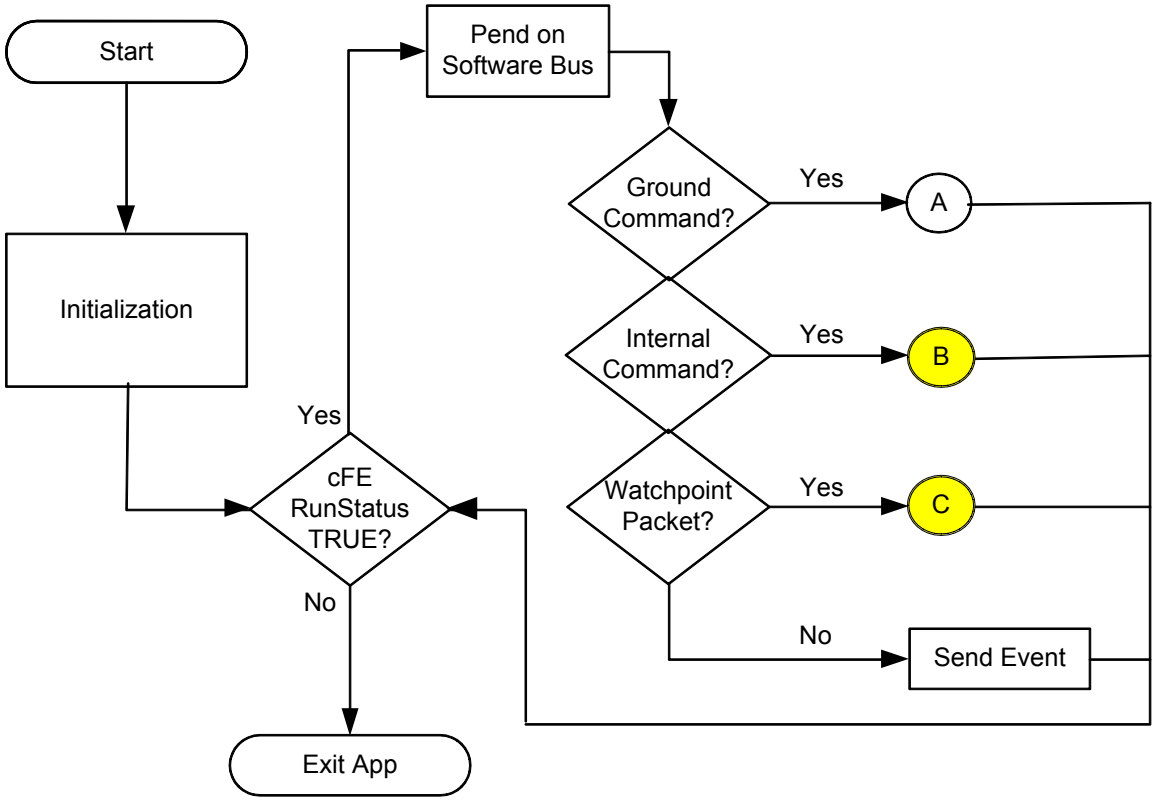

Figure 1 LC/LCX - Flow Control

Figure 2 below shows the LC/LCX flow detail (A) for ground commands. This figure is presented for clarity; there is no change in this figure from LC.

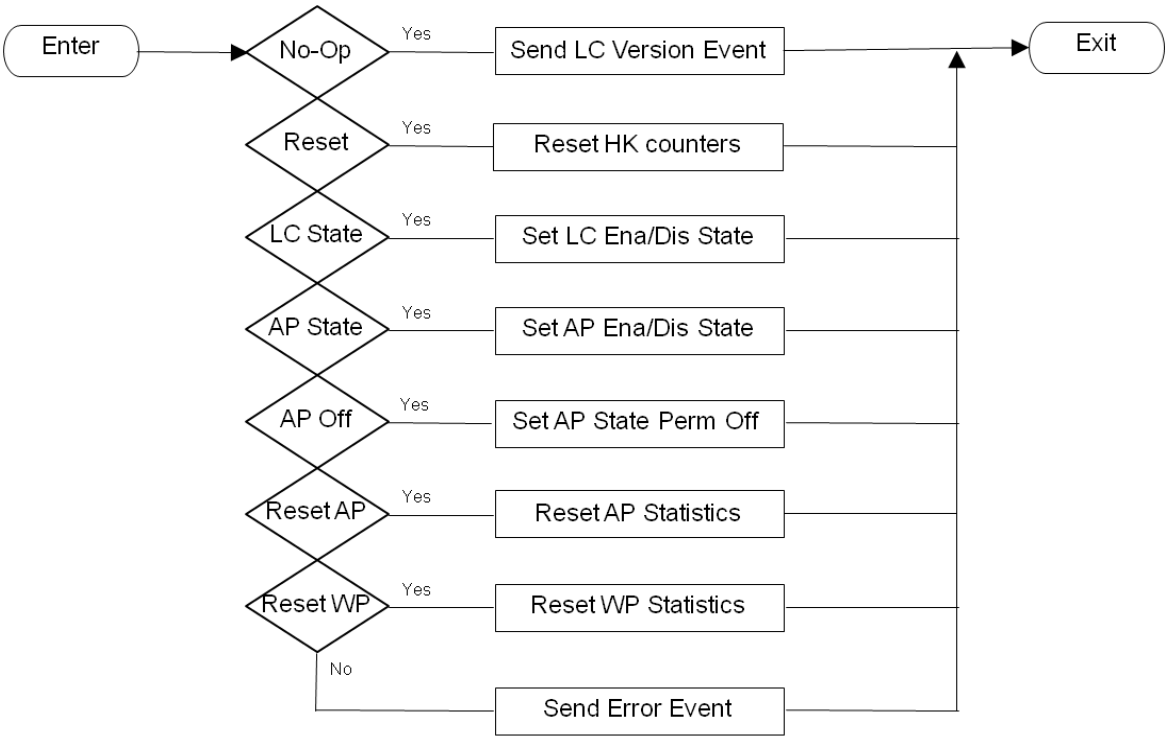

Figure 2 LC/LCX - Flow Detail (A) – Ground Command

Figure 3 below shows the LCX flow detail B for internal commands. The B1 circle is highlighted in yellow to indicate a difference in flow control between the LCX and LC applications.

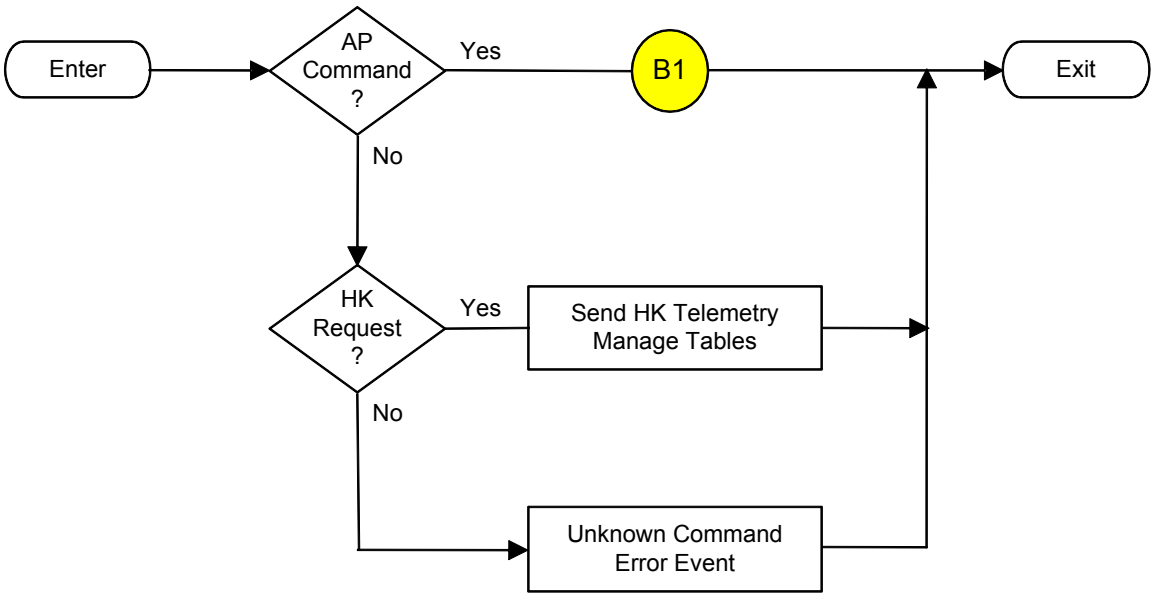

Figure 3 LCX - Flow Detail (B) – Internal Command

Figure 4 below shows the flow control for Actionpoint Commands. The B2 circle is highlighted in yellow to indicate a difference in flow control between the LCX and LC applications.

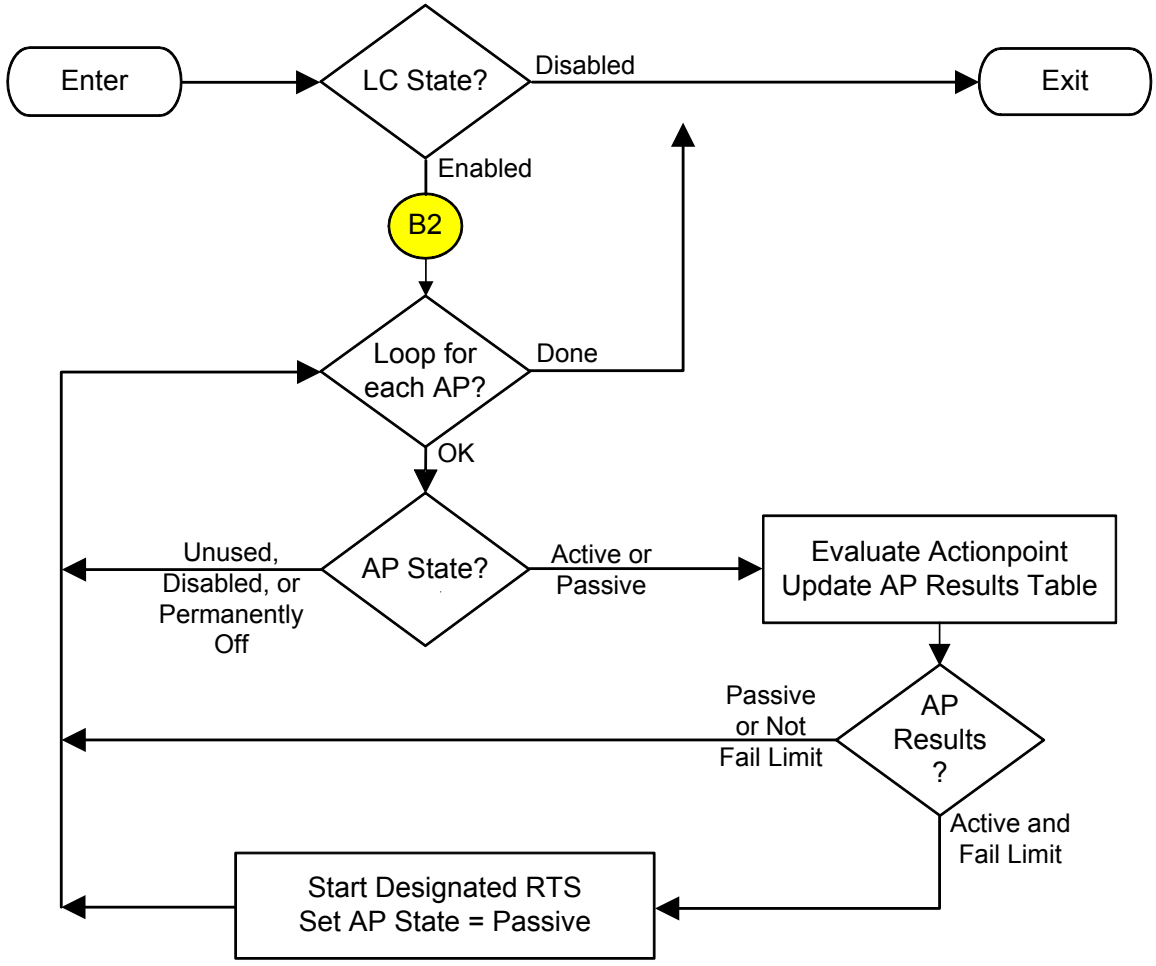

Figure 4 LCX - Flow Detail (B1) – Actionpoint Command

Figure 5 below shows LCX Flow Detail (B2) to update the WP stale counters.

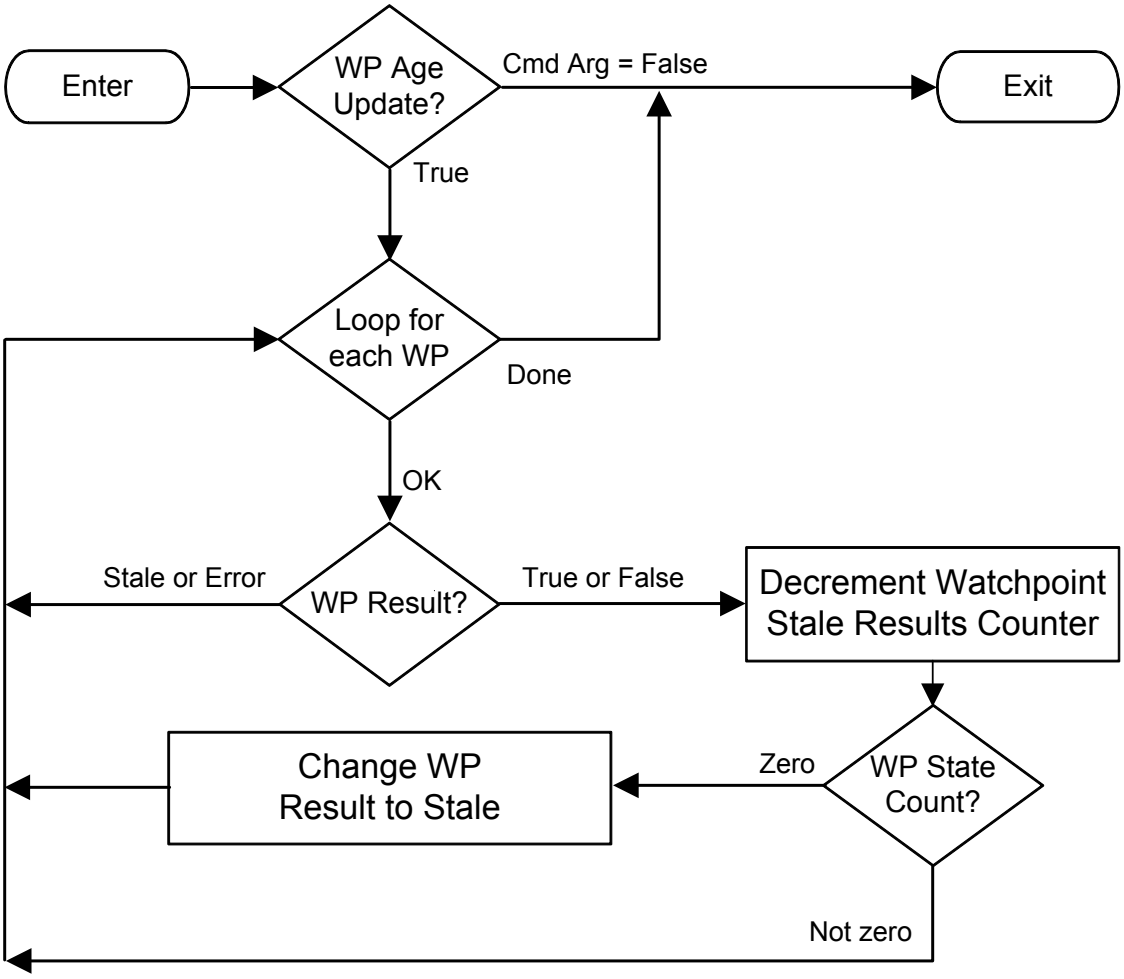

Figure 5 LCX - Flow Detail (B2) – Update WP Stale Counters

Figure 6 below shows Flow Detail (C) – Watchpoint Packet.

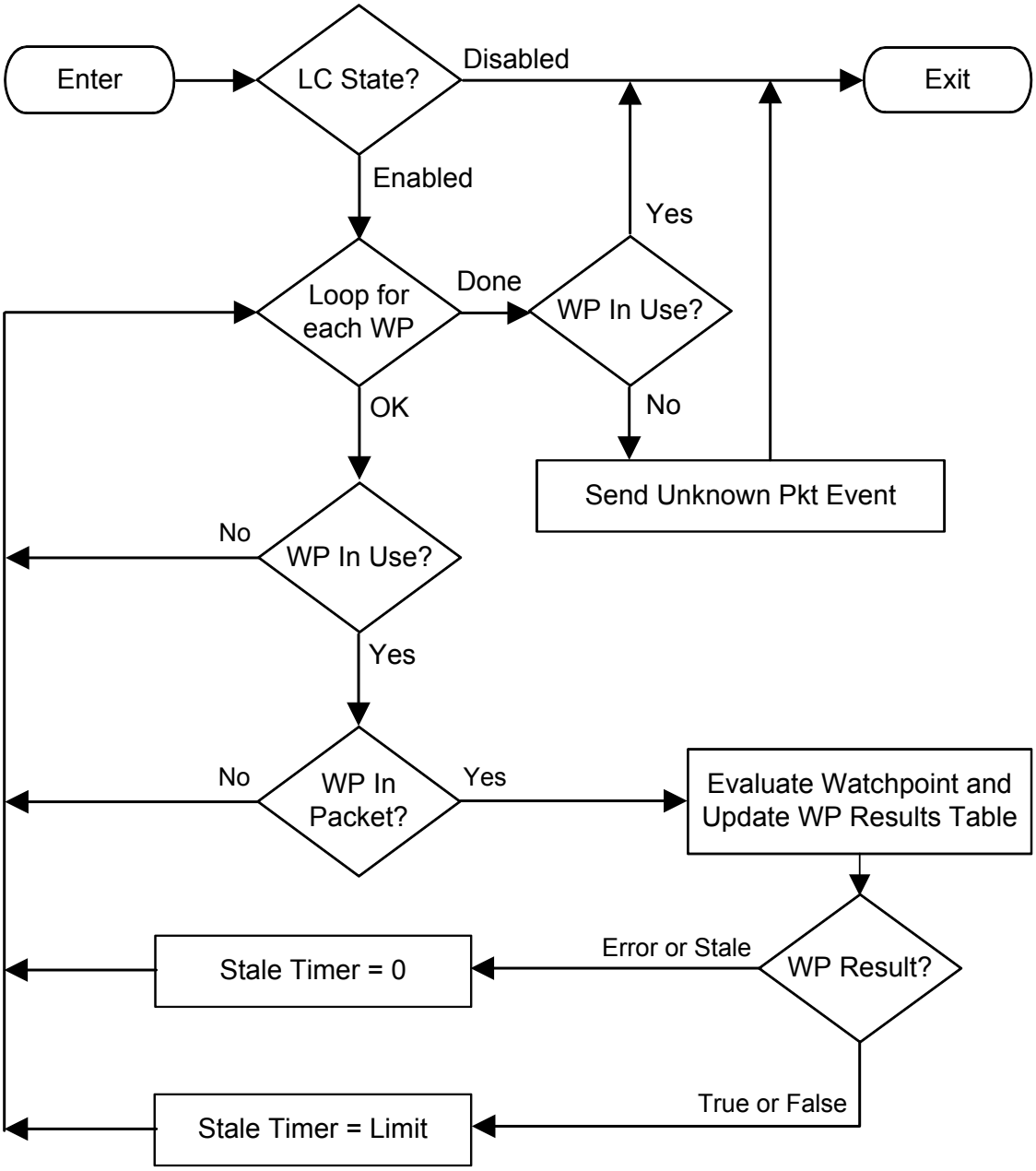

Figure 6 LCX - Flow Detail (C) – Watchpoint Packet

2.3.2 LCX – Result Transition to Stale

2.3.2.1 Watchpoint Table Enhancements

The LCX watchpoint definition table specifies a Stale Limit for each watchpoint. The Stale Limit specifies how long after evaluation before the watchpoint result becomes stale.

The watchpoint results table specifies a Stale Counter for each watchpoint. The Stale Counter specifies how much longer before the watchpoint result becomes stale.

2.3.2.2 Watchpoint and Actionpoint Result Initialization

All watchpoint and actionpoint results are initialized to Stale at startup.

2.3.2.3 Watchpoint Result Transition to Stale

The Stale Counter is set to equal the Stale Limit as specified in the WDT when the watchpoint evaluates to True or False.

Stale Counters decrement upon receipt of a scheduled command (assuming a Sample Actionpoints command with "update stale counters" set to true.)

The Watchpoint Result is set to stale when the Stale Counter becomes zero.

Note: The use of zero as the stale limit (ResultAgeWhenStale = 0) will result in bypassing the stale process for the watchpoint. When using zero as the stale limit the WatchResult will never be changed from true or false to stale, since the CountdownToStale will never decrement if ResultAgeWhenStale is already zero at the start of the process.

ResultAgeWhenStale is the age when a Watchpoint comparison result becomes stale. This value is contained in the WDT, as shown in Table 9, Watchpoint Definition Table (WDT) Entry, on page A-3.

2.3.2.4 Actionpoint Result Transition to Stale

The Actionpoint Result is set to Stale when the sampled actionpoint has stale watchpoints.

2.3.3 LCX – Actionpoint Equations

This section discusses AP Equations.

AP equations consist of operators and operands. The AP operators are AND, OR, XOR, NOT, and EQUAL. AP operands are a single WP result or the result from a previous operator in the equation.

Equations are expressed in Reverse Polish Notation (RPN), also known as postfix notation. For example (23, 24, OR, EQUAL).

- Step 1 – OR the results from WP 23 and WP 24
- Step 2 – compare the result from Step 1 to TRUE
- Equation result = the result from Step 2
- Note: all AP equations must terminate with the EQUAL operator

AP equations use a true result to indicate an error condition.

- AP result = Fail if result of equation = True
- AP result = Pass if result of equation = False
- AP result = Error if result of equation = Error
- AP result = Stale if result of equation = Stale

2.3.4 LCX – Actionpoint Operators

Actionpoint Operators consist of AND, OR, XOR, NOT, and EQUAL. Details are discussed below.

2.3.4.1 AP Operator = AND

When the AP Operator is AND there are two operands.

- If either operand evaluates to False, the operator result evaluates to False, even if the other operand is Error or Stale.
- If either operand evaluates to Error, the operator result evaluates to Error.
- If either Operand evaluates to Stale, the Operator result evaluates to Stale.
- Otherwise, the Operator result evaluates to True.

2.3.4.2 AP Operator = OR

When the AP Operator is OR there are two operands.

- If either operand evaluates to True, the operator result evaluates to True, even if the other operand is Error or Stale.
- If either operand evaluates to Error, the operator result evaluates to Error.
- If either Operand evaluates to Stale, the Operator result evaluates to Stale.
- Otherwise, the Operator result evaluates to False.

2.3.4.3 AP Operator = XOR

When the AP Operator is XOR there are two operands.

- If either operand evaluates to Error, the operator result evaluates to Error.
- If either operand evaluates to Stale, the operator result evaluates to Stale.
- If operand 1 equals operand 2, the operator result evaluates to False.
- Otherwise, the Operator result evaluates to True.

2.3.4.4 AP Operator = NOT

When the AP Operator is NOT there is one operand.

- If the operand evaluates to Error, the operator result evaluates to Error.
- If the operand evaluates to Stale, the operator result evaluates to Stale.
- If the operand evaluates to True, the operator result evaluates to False.
- Otherwise, the Operator result evaluates to True.

2.3.4.5 AP Operator = EQUAL

When the AP Operator is EQUAL there is one operand.

- If the operand evaluates to Error, the operator result evaluates to Error.
- If the operand evaluates to Stale, the operator result evaluates to Stale.

- If the operand evaluates to True, the operator result evaluates to True.
- Otherwise, the Operator result evaluates to True.

Chapter 3. CFS LCX Normal Operations

3.1 *LCX Modes of Operation*

The Limit Checker Extended application has three operating modes that can be set via the "Set LC State" ground command: active, passive, and disabled, as explained in the sections below.

No counters are reset when changing mode. (Counters may be reset if the application is reset or if the counters are reset by ground command.)

The three operating modes are explained in the sections below.

3.1.1 Active Mode

Active Mode is the normal operation mode. LCX performs all limit tests defined in the watchpoint definition table and invokes stored command sequences as defined in the actionpoint definition table when an actionpoint fails.

3.1.2 Passive Mode

In Passive mode, LCX performs all limit tests as in Active mode, but no stored command sequences are invoked as the result of actionpoint failures. Event messages are still generated as they are in the Active mode.

3.1.3 Disabled Mode

In Disabled mode, no watchpoint or actionpoint evaluations take place.

3.2 *Timing Issues*

When LCX subscribes to a packet it will get that packet from the software bus at whatever frequency that packet is being generated by the source of the data. LCX doesn't control the rate in any way.

Watchpoint states are updated when a packet that contains the watchpoint is received by LCX. Actionpoints are evaluated when LCX receives a Sample Actionpoint Command, typically from the CFS Scheduler (SCH) application or corresponding scheduler application. If the sample actionpoint command is at a slower rate than the watchpoint packets, LCX could potentially be processing "stale" data.

3.3 *Housekeeping Packet Structure*

Housekeeping reference material is located in Appendix A. See Table 14 Watchpoint (WP) Results Housekeeping Telemetry on page A-7; and Table 15 Actionpoint (AP) Results Housekeeping Telemetry on page A-7.

3.4 *Custom Functions*

A custom function is a way to do more complicated processing on a watchpoint when the standard comparison operators are not adequate. A custom function can be used in place of a standard comparison operator in watchpoint definitions.

A custom function returns **True** or **False** as the result of the comparison for the watchpoint that triggered the call. LC can have as many custom functions as monitor points.

Custom functions cannot be created or changed by the FOT. In flight updates to custom functions require rebuilding, testing, and reloading the LCX application.

Mission developers need to modify LCX LC_CustomFunction source code, though changes are limited to a specific file. This means minimal overhead for missions that don't need custom functions.

3.5 *Task: Creating a Limit Checker Table*

The first step in deploying the LCX application is to size and construct default WDT and ADT tables. LCX requires these default table images to be on the filesystem when the application is started or it will fail to load.

Many different personnel need to know how to create a Limit Checker table. Mission Developers or testers may want to use Limit Checker itself to verify that LCX is working and then initiate checks of other systems. Flight controllers and post-launch software maintenance developers may routinely use tables to initiate or modify Limit Checker tables for their own purposes.

Missions may maintain their own documentation on how to create Limit Checker tables. Web based ground software tools at NASA Goddard may be used to help with part of the formatting task of creating a Limit Checker table.

Step by step instructions of creating a Limit Checker table from scratch are beyond the scope of this Guide *Supplement*.

See also: CFS Limit Checker Deployment Guide in "Doxygen" html documentation.

Chapter 4. Understanding LCX Operational Constraints

All items in this chapter not appearing below remain unchanged from the LC User's Guide.

4.1 *Replacements*

In the LC User's Guide, in Chapter 4, Understanding LC Operational Constraints, in the section titled "Unused Entries in WDT" the third paragraph is replaced with the following for LCX:

When either the WDT or ADT are updated, the corresponding results table (WRT or ART) is reset to initialization values. For each entry in the WRT, WatchResult is set to LC_WATCH_STALE and all other values are zeroed. For each entry in the ART, ActionResult is set to LC_ACTION_STALE, the CurrentState is set to the value of the actionpoint's DefaultState specified in the ADT, and all other values are zeroed.

This page intentionally blank.

Chapter 5. Frequently Asked Questions (FAQs)

All FAQs that appear in the LC User's Guide and do not appear below also apply to LCX.

5.1 *FOT Questions*

5.1.1 How is stale data defined?

The goal of tracking stale data by LCX is to prevent LCX from taking action on old or inaccurate information. LCX is designed with stale data defined as datum for which a packet update has not been received and/or has not been received in a requisite interval specified by the age value. In most cases, it indicates that a subsystem is turned off or some other condition has prevented packet generation.

5.1.2 On cFE power-on, what values does LCX use to initialize the result of the last Actionpoint Sample?

Upon cFE Power-On, LC initializes the result of the last Actionpoint Sample is set to Stale.

5.1.3 In LCX, what value are Watchpoint and Actionpoint Results assigned at Initialization?

For each Watchpoint specified in the Watchpoint Definition Table (WDT) LCX specifies an age value which indicates when the data becomes "stale". However, in LCX, all watchpoint and actionpoint results are initialized to Stale at startup.

5.1.4 How do I activate or deactivate the Stale process for individual watchpoints?

To activate the stale process on individual watchpoints, use nonzero values for the stale limit. That is, for individual watchpoints, set ResultAgeWhenStale to a nonzero value at the start of the process. *ResultAgeWhenStale* is the age when a Watchpoint comparison result becomes stale. This value is contained in the WDT, as shown in Table 9, Watchpoint Definition Table (WDT) Entry, on page A-3.

To turn off the stale process for individual watchpoints, use zero as the stale limit. That is, for individual watchpoints, set ResultAgeWhenStale to zero (0) at the start of the process. This shuts down the stale process for this watchpoint, because the WatchResult never changes from true or false to stale. Why? Because the CountdownToStale never decrements if ResultAgeWhenStale is already zero at the start of the process.

5.1.5 How does LCX track stale data?

For each watchpoint specified in the Watchpoint Definition Table (WDT) LCX maintains the age of the data.

5.2 *Questions for Creators of Watchpoints and Actionpoints*

While the questions in this section generally do not apply to the FOT (the FOT generally do not build watchpoint or actionpoint tables), they are included to help the FOT better understand those tables.

5.2.1 When would a watchpoint evaluate to Stale?

LC_WATCH_STALE is an initialization value for the Watchpoint Results Table. If a watchpoint has this WatchResult then the watchpoint is unused (the DataType in the WDT is set to LC_WATCH_NOT_USED), or a message that contains the watchpoint has not yet been received by LC and evaluated. For more, see the next question below.

5.2.2 How exactly does a watchpoint result transition from True or False to Stale?

To fully understand how an actionpoint evaluates to LC_ACTION_STALE, one needs to understand how a watchpoint result transitions from LC_WATCH_TRUE or LC_WATCH_FALSE to LC_WATCH_STALE.

When any watchpoint evaluates to true or false, the corresponding WatchResult is set to LC_WATCH_TRUE or LC_WATCH_FALSE and CountdownToStale is set to ResultAgeWhenStale.

Thereafter, each time a Sample Actionpoints command is processed, each non-zero watchpoint CountdownToStale is decremented.

If the decremented CountdownToStale becomes zero, then that WatchResult is set to LC_WATCH_STALE.

5.2.3 When would an actionpoint evaluate to Stale?

LC_ACTION_STALE is an initialization value for the Actionpoint Results Table. If an actionpoint has this ActionResult then one of three possible conditions are true:

1. The actionpoint is unused (the DefaultState in the ADT is set to LC_ACTION_NOT_USED).
2. An actionpoint sample request (LC_SAMPLE_AP_MID) targeting the AP has not yet been received by LC so the AP has not yet been evaluated.
3. One or more of the watchpoints that this AP depends on (as defined by the RPN expression) has a current WatchResult of LC_WATCH_STALE so the AP can't be evaluated.

5.2.4 When do results tables get cleared?

When either the WDT or ADT are updated, the corresponding results table (WRT or ART) is reset to initialization values. For each entry in the WRT, WatchResult is set to LC_WATCH_STALE and all other values are zeroed. For each entry in the ART, ActionResult is set to LC_ACTION_STALE, the CurrentState is set to the value of the actionpoint's DefaultState specified in the ADT, and all other values are zeroed.

These are also the values used (for the entries specified in the command) when the LC_RESET_AP_STATS_CC or LC_RESET_WP_STATS_CC ground command is received.

Appendix A CFS LCX Reference

Tables not included in this appendix that appear in the LC User's guide also apply to LCX.

A.1 Internal Name Differences between LC and LCX

This section summarizes the difference in selected #Defined names between LC and LCX for optional use by FSSE.

These values are defined in `lc_tbldefs.h` and show the differences between LC and LCX:

Table 3 Nomenclature Differences: Watchpoint (WP) and Actionpoint (AP)

LC Name and Value		LCX Name and Value	
LC_WATCH_NOT_MEASURED	0xFF	LC_WATCH_STALE	0xFF
LC_ACTION_NOT_MEASURED	0xFF	LC_ACTION_STALE	0xFF

These values are defined in `lc_msgdefs.h` and show the differences between LC and LCX:

Table 4 Nomenclature Differences: Housekeeping (WP and AP Results)

LC Name, Value, and Description			LCX Name, Value, and Description		
LC_HKWR_NOT_MEASURED	0x03	Brief Two bit value used for Not Measured	LC_HKWR_STALE	0x03	Brief Two bit value used for STALE
LC_HKAR_NOT_MEASURED	0x03	Brief Two bit value used for Not Measured	LC_HKAR_STALE	0x03	Brief Two bit value used for STALE

These values are defined in `lc_tbl.h` and appear only in LCX:

Table 5 Stale Counters for Watchpoint (WP) Definition and Result tables

Type	Name	Description
uint32	ResultAgeWhenStale	Brief Number of LC Sample Actionpoint commands that must be processed after comparison before result goes stale

Type	Name	Description
uint32	CountdownToStale	Brief Number of LC Sample Actionpoint commands still to be processed before WatchResult becomes stale

The values in Table 6 below are defined in `lc_msg.h` and appear only in LCX. The other items in this command are the command header, EndIndex, and StartIndex.

Table 6 Watchpoint (WP) Age Results Option in Actionpoint (AP) Sample Command

Type	Name	Description
uint16	UpdateAge	Brief Update WP results age (True or False)

A.2 Internal Messaging

Table 7 below shows Internal Messages. AP Sample Request is expanded in LC to optionally update the age of the WP results.

Table 7 Internal Messages

Message	Flow	Description
Housekeeping Request	Input to LC	Send housekeeping data. (See LC_SEND_HK_MID in Table 8 below.)
AP Sample Request	Input to LC	Sample one, all, or a range of actionpoints. Source of request is transparent to LC. Normally sent internally but can also be sent from the ground. [Optionally update age of WP results. ***See also:*** Table 6, Watchpoint (WP) Age Results Option in Actionpoint (AP) Sample Command, above.] ***See also:*** LC_SAMPLE_AP_MID in Table 8 below.
Start RTS	Output from LC	Sent when an actionpoint failure needs to initiate an RTS.

Table 8 below shows Message IDs. The table appears for convenience; the information remains unchanged from LC.

Table 8 Message IDs

Name	Value	Description
LC_HK_TLM_MID	0x08A7	LC Housekeeping Telemetry.
LC_SAMPLE_AP_MID	0x18A6	Msg ID to request actionpoint sample. This is not a ground command but an internal message that will normally come from the CFS scheduler (SCH) application.
LC_SEND_HK_MID	0x18A5	Msg ID to request LC housekeeping.

A.3 Watchpoints and Actionpoints

A.3.1 Watchpoint Definitions

Table 9 below shows a full definition of a single watchpoint. (The Watchpoint Definitions Table (WDT) is a series of watchpoint definitions. There is only one WDT.)

Table 9 Watchpoint Definition Table (WDT) Entry

Element	Type	Name	Description
Watchpoint Data Type	uint8	DataType	Watchpoint Data Type: Unused Byte UByte Word UWord DWord UDWord Float Endian byte order (Big or Little)
Watchpoint Operator ID	uint8	OperatorID	Comparison type (enumerated) - ***See also:*** Table 10 below for Operator ID Comparison Types.
Watchpoint Message ID	uint16	MessageID	Message ID for the message containing the watchpoint
Watchpoint Data Offset	uint32	WatchpointOffset	Zero based byte offset from the beginning of the message (including any headers) to the first byte of the watchpoint data
Watchpoint Bit Mask	uint32	BitMask	Value to be masked with watchpoint data prior to comparison. Use the constant LC_NO_BITMASK when no masking is desired. ***See also:*** Table 11 below for Watchpoint Definition Table (WDT) Enumerated Types.
Watchpoint Comparison Value	28 byte struct	ComparisonValue	Value against which watchpoint data is compared. This field uses the LC_MultiType_t union to store different data types in a fixed 32-bit field. ***See also:*** lc_def_wdt.c in the CFS LC source code for examples of how to set this value.

Element	Type	Name	Description
Age When Watchpoint Comparison Result Becomes Stale	uint32	ResultAgeWhenStale	Units are the number of LC "sample actionpoints" commands since comparison.
Custom Function Argument	uint32	CustomFuncArgument	Optional 32 bit data to be passed to the custom function when Operator ID is set to Custom. Can be used for any mission-defined purpose. Must be set up before program is compiled. Generally not changeable by the FOT.

Table 10 below shows the enumerated choices for the Watchpoint Operator ID comparison type. The table appears for convenience; the information remains unchanged from LC.

Table 10 Operator ID Comparison Types

Element	CFS LC Value	Description	
LC_NO_OPER	0xFF	Can be used for unused entries (optional)	Unused (optional)
LC_OPER_LT	1	Less Than	<
LC_OPER_LE	2	Less Than or Equal	<=
LC_OPER_NE	3	Not Equal	!=
LC_OPER_EQ	4	Equal	=
LC_OPER_GE	5	Greater Than or Equal	>=
LC_OPER_GT	6	Greater Than	>
LC_OPER_CUSTOM	7	No compare, call custom function	Custom

Table 11 below shows the enumerated choices for Watchpoint Bit Mask. The table appears for convenience; the information remains unchanged from LC.

Table 11 Watchpoint Definition Table (WDT) Enumerated Types

Internal Name	CFS LC Default Value	Description
LC_NO_BITMASK	0xFFFFFFFF	Use for no bitmasking.
LC_NO_OPER	0xFF	Use for empty entries.
LC_OPER_CUSTOM	7	Use custom function.

A.3.2 Watchpoint Results

Table 12 below shows a full definition of a single watchpoint results table entry; a Watchpoint Results Table (WRT) is a series of watchpoint results definitions. The WRT is an array of single watchpoint results table entries sized by the configuration parameter LC_MAX_WATCHPOINTS. The index into this table is the same Watchpoint ID used for the corresponding definition table entry.

Table 12 Watchpoint Results Table (WRT) Entry

WRT Field Name	Description
Watchpoint Evaluation Result	Most recent comparison result for this watchpoint. Possible Values: True (1) False (0) Error (2) Stale (0xFF)
Age Countdown Until Watchpoint Result Becomes Stale	Age units are the number of LC "sample actionpoints" commands
Cumulative Evaluation Count	Total number of times this watchpoint has been evaluated
Cumulative False To True Count	Total number of times this watchpoint has transitioned from False to True
Consecutive True Count	Number of consecutive times this watchpoint has evaluated to True
Cumulative True Count	Total number of times this watchpoint has evaluated to True
Most Recent Watchpoint Result Transition From False to True	Spacecraft time and watchpoint data when transition occurred. • Watchpoint value and timestamp at the last transition from 0 (False) or Stale (0xFF) to 1 (True) • (The timestamp used for the LastFalseToTrue field is taken from the header of the message that contained the watchpoint. If the message timestamp is zero, LC will use the time returned by the CFE_TIME_GetTime function instead.)

WRT Field Name	Description
Most Recent Watchpoint Result Transition From True to False	Spacecraft time and watchpoint data when transition occurred • Watchpoint value and timestamp at the last transition from 1 (True) to 0 (False). • (The timestamp used for the LastTrueToFalse field is taken from the header of the message that contained the watchpoint. If the message timestamp is zero, LC will use the time returned by the CFE_TIME_GetTime function instead.)

A.3.3 Actionpoint Definitions

For Actionpoint definitions, please refer to the LC User's Guide.

A.3.4 Actionpoint Results

Table 13 below shows the descriptions for Actionpoint results tables.

Table 13 Actionpoint Results Table (ART) Entry

Element	Description
Actionpoint Evaluation Result	Most recent evaluation result Pass (0) Fail (1) Error (2) Stale (0xFF)
Actionpoint Current State	Current state of this actionpoint Unused Active Passive Disabled Permanently Off
Cumulative Fail To Pass Count	Total number of times this actionpoint has transitioned from Fail to Pass
Cumulative Pass To Fail Count	Total number of times this actionpoint has transitioned from Pass to Fail
Consecutive Fail Count	Number of consecutive times this actionpoint has evaluated to Fail
Cumulative Fail Count	Total number of times this actionpoint has evaluated to Fail
Cumulative RTS Exec Count	Total number of times an RTS request has been sent for this actionpoint

A.4 Housekeeping Telemetry

Table 14 below shows descriptions of the packed subset of the **Watchpoint** Results Table.

Table 14 Watchpoint (WP) Results Housekeeping Telemetry

WPResults	Description
Byte array	Two bits per watchpoint (aligned to nearest longword boundary)
Most recent watchpoint comparison result (2 bits)	0 = False (0x00) 1 = True (0x01) 2 = Error (0x02) 3 = Stale (0x03)
Ordering	(Rwp3, Rwp2, Rwp1, Rwp0), (Rwp7, Rwp6, Rwp5, Rwp4), etc...

Table 15 below shows descriptions of the packed subset of the **Actionpoint** Results Table.

Table 15 Actionpoint (AP) Results Housekeeping Telemetry

APResults	Description
Byte array	Four bits per actionpoint (aligned to nearest longword boundary)
Actionpoint current state (2 bits)	0 = Unused or Permanently Off (0x00) 1 = Active (0x01) 2 = Passive (0x02) 3 = Disabled (0x03)
Most recent actionpoint analysis result (2 bits)	0 = Pass (0x00) 1 = Fail (0x01) 2 = Error (0x02) 3 = Stale (0x03)
Ordering	(Sap1, Rap1, Sap0, Rap0), (Sap3, Rap3, Sap2, Rap2), etc...

A.5 Configuration Parameters

Configuration parameters as documented in the LC User's Guide are unchanged between LC and LCX.

A.6 LC Commands

LC Commands as documented in the LC User's Guide remain unchanged between LC and LCX.

A.7 Event Messages

Event messages as documented in the LC User's Guide are unchanged between LC and LCX.

This page intentionally blank.

Appendix B Mission-Specific Reference

Mission specific LCX-related documentation is located at (mission defined).

This page intentionally blank.

Appendix C Document Notes

C.1 Mission-Specific Conventions

- *This document presents selected information that should be removed when tailoring this document for a mission in this italic dark orange Times Roman font.*
- *Command and Telemetry mnemonics are mission-specific. This document as delivered has "suggested" names that may or may not be used by the mission when the MOT creates the ground system database. In particular, the suggested names start with $sc_$cpu_LC which indicate a global setting for spacecraft, processor selection, and the LC subsystem. This has meaning if the mission has multiple spacecraft, each with a copy of cFE/CFS apps being executed, and/or multiple processors per spacecraft, each with a copy of cFE/CFS apps being executed. Most missions have neither and they do not prepend a $sc_$cpu_ selection to the front of the command name. However it is common for missions to differentiate the spacecraft subsystem commands from instrument commands by prepending a couple of characters (e.g. pw for power system) to all the command and telemetry names for that subsystem.*
- *The nomenclature of command and telemetry mnemonics is highly mission-specific, so this document does not attempt to include the actual command and telemetry database names in advance. For example, for MMS the telemetry mnemonics are defined in a Record Definition Language (RDL) file for ASIST, and they will not exactly match the mnemonics in this Guide Supplement.*

C.2 Updating This Document

This section is for anyone updating this Guide *Supplement* in the future.

When tailoring this document for a particular mission, remove text appearing in this italic dark orange Times Roman font.

- *Review figures to be sure there is no conflict with mission configurations. Edit figures with MS Visio if necessary.*
- *Add Mission Defined values in Appendix A.*
- *Add the location of mission-specific documentation in Appendix B.*
- *Regenerate the table of contents (TOC); add Appendix page prefixes to TOC as needed.*

This Guide *Supplement* is formatted using Microsoft Word styles. When adding new sections to the Guide *Supplement*, assign paragraphs to the styles shown in the table below. Center tables and figures horizontally on the page. Use 15% grayscale in new table headings.

Table of Contents, Figures, and Tables can be updated automatically, but the letter prefix for Appendix pages must be added manually after update. To update all figure and table references in the document, when using the PC version of Word, select all, then choose F9.

Table 16 Internal Document Styles

Type	Style to be Used	Justification
Chapter titles, subtitles, and subsections.	Heading 1 through 6	Left
First level bullets	"List Bullet 1" style.	Left
Second level bullets	"List Bullet 2" style	Left
Third level bullets	"List Bullet 2" style	Left
Numbered lists	"Style List Enum 0"	Left
Names of code modules	Code	N/A
All text not otherwise tagged	"Body Text"	Full justification
Tables & Figures		
First row of tables	Table Header	Center
All other cells of tables	Table Cells	Left
Figure captions	Caption Figure	Center
Table captions	Caption Table	Center

C.3 Providing Feedback about this Document

For CM reasons, if you find any item in this Guide *Supplement* that is in error, or want to be informed of any updates, please email feedback to the cFE/CFS PDL for validation and routing. As of the date of publication, the cFE/CFS PDL is Susie Strege (susanne.L.strege@nasa.gov).

Besides corrections of any errors, the CFS team is interested in your ideas on improvements that help understanding, such as, perhaps, improvements to flow charts to show better how LCX juggles incoming commands with other program responsibilities, for example, or any other area.