# CFS Memory Dwell (MD) User's Guide

* [CFS Memory Dwell Introduction](#AAAAAAAAHY)
* [CFS Memory Dwell Overview](#AAAAAAAAHZ)
* [CFS Memory Dwell Operation](#AAAAAAAAIC)
* [CFS Memory Dwell Requirements](#AAAAAAAAIG)
* [CFS Memory Dwell Deployment Guide](#AAAAAAAAIH)
* [CFS Memory Dwell Commands](#AAAAAAAAII)
* [CFS Memory Dwell Telemetry](#AAAAAAAAIJ)
* [CFS Memory Dwell Table Definitions](#AAAAAAAAIK)
* [CFS Memory Dwell Configuration Parameters](#AAAAAAAAIL)
* [CFS Memory Dwell Constraints](#AAAAAAAAIT)
* [CFS Memory Dwell Frequently Asked Questions](#AAAAAAAAJB)
* [CFS Memory Dwell Event Message Reference](#AAAAAAAAEI)

*This guide was generated using Doxygen.*

# CFS Memory Dwell Introduction

### Scope

This document provides a complete specification for the commands and telemetry associated with the CFS Memory Dwell (MD) application software. The document is intended primarily for users of the software (operations personal, test engineers, and maintenance personal). The deployment guide section, is intended for mission developers when deploying and configuring the MD application software for a mission flight software build environment.

### Applicable Documents

|  |  |
| --- | --- |
| **Document ID** | **Document Title** |
| TBD | CFS Memory Dwell Application Requirements Document |
| TBD | CFS Memory Dwell Heritage Analysis Document |
| TBD | CFS Memory Dwell Design Document |
| TBD | *Need Reference to OSAL Document* |

### Acronyms

|  |  |
| --- | --- |
| **Acronym** | **Description** |
| API | Application Programming Interface |
| ATP | Absolute Time Processor |
| ATS | Absolute Time tagged command Sequence |
| CCSDS | Consultative Committee for Space Data Systems |
| C&DH | Command and Data Handling |
| CFE | Core Flight Executive |
| CFS | Core Flight System |
| CI | Command Ingest |
| Cmd | Command |
| CPU | Central Processing Unit |
| EDAC | Error Detection and Correction |
| FDS | Flight Data System |
| FM | File Manager |
| FSW | Flight Software |
| GN&C | Guidance Navigation & Control |
| GSFC | Goddard Space Flight Center |
| HK | Housekeeping |
| HW, H/W | Hardware |
| ICD | Interface Control Document |
| ISR | Interrupt Service Routine |
| OS | Operating System |
| OSAL | Operating System Abstraction Layer |
| Pkts | Packets |
| RAM | Random-Access Memory |
| RTOS | Real Time Operating System |
| RTP | Relative Time Processor |
| RTS | Relative Time tagged command Sequence |
| SB | Software Bus Service |
| SBC | Single Board Computer |
| SC | Stored Commands task |
| SW, S/W | Software |
| TBD | To Be Determined |
| TBL | Table |
| TLM | Telemetry |
| UTC | Universal time code |

# CFS Memory Dwell Overview

The Memory Dwell (MD) application enables an operator to create and control telemetry streams. These dwell telemetry streams can be defined either by table loads or by jam commands. If the operating system supports symbolic addressing, Memory Dwell supports specifying memory addresses using a symbolic name. Memory addresses to be sampled are specified in terms of a numerical "offset" address, and a symbolic address. Dwell stream definitions and state information (i.e. enable vs. disable) for each dwell that are provided via table load are saved across processor resets. An optional ability to insert string signatures into dwell packets is provided.

## Design Overview

MD is a command driven, single threaded application. It will pend indefinitely on the software bus message queue until messages arrive and then processes them sequentially. Memory Dwell utilizes OSAL routines to:

* Read memory
* Translate symbolic addresses to their numerical equivalents
* Determine if a memory address is in a valid range

The fastest rate at which memory values are collected by the Memory Dwell application is determined by the rate at which the Scheduler issues wakeup messages. Collection rates are specified in terms of multiples of wakeup calls. Memory Dwell specifications that are initiated via Table Loads are saved to the Critical Data Store (CDS). Memory Dwell specifications that are initiated via Jam Commands are not saved to the CDS. Similarly, enabled/disabled state information for each Memory Dwell included in Table Loads is saved to the CDS. Enabled/disabled state information set by commands is not saved to the CDS.

# CFS Memory Dwell Operation

The operational interface of the MD application consists of ground commands, dwell table loads, and housekeeping telemetry. The following items detail the operational interface:

1. **The application version number**  - The application version number is displayed in the [Initialization event message](#AAAAAAAAEJ)  and the [No-op event message](#AAAAAAAAEQ). Both events are 'informational' type and are unfiltered by default.

2. **Loading a Dwell Table** - Memory Dwell supports a configurable number of dwell tables. Each of these dwell tables can be loaded by way of a file. A new dwell table may be loaded at any time. The steps to load a new Copy Table are:

a) Transfer the table file to the on-board file system

b) Send the CFE Table load command

c) Send the CFE Table validate command

d) Send the CFE Table activate command

3. **Sending the No-op Command** - To verify connectivity with the Memory Dwell application, the ground may send an [MD No-op command](#AAAAAAAABK) . If the packet length field in the command is set to the value expected by the MD app, then the command counter will increment and a [No-op informational event message](#AAAAAAAAEJ)  will be sent. This no-op event will show the version number of the MD application.

4. **Sending the reset counters command** - The reset counters command will reset only the telemetry points listed here:

a) [MD\_HkTlm\_t.ValidCmdCntr](#AAAAAAAAIE) - Number of accepted commands

b) [MD\_HkTlm\_t.InvalidCmdCntr](#AAAAAAAAIF) - Number of rejected commands

5. **Monitoring the command counter** - The command counter, MD\_CMDPC, will increment when the MD application receives any valid command.

6. **Monitoring the command error counter** - The command error counter will increment under the following conditions:

a) Invalid command code

b) Unexpected packet length field for ['MD Wakeup' command](#AAAAAAAAAI)

c) Unexpected packet length field for ['Send MD Housekeeping' command](#AAAAAAAAAH)

d) Unexpected packet length field for a ground command

e) Invalid command argument

7. **Enabling and Disabling Dwell Packet Streams** - The enabled/disabled state of dwell packet streams can be modified using the start and stop ground commands. The enabled/disabled state is also controlled by the enabled flag in the dwell load table.Note that a dwell packet stream will not actively operate if the total delay count is zero, no matter what the enabled/disabled state is.

8. **Specifying a memory dwell stream** - Each Memory Dwell Table consists of [MD\_DWELL\_TABLE\_SIZE](#AAAAAAAAAM) entries. Each entry serves either as an individual dwell definition or a null entry. The MD will build packets beginning with the dwell specified in the first slot, adding additional dwells until it reaches either a null entry or the end of the table. Each individual memory dwell definition is specified by a memory address, the number of delay counts before the next read, and the dwell length. The number of delay counts is the number of wakeup calls the application will receive before the next memory dwell read. A valid Dwell Table entry which is an individual memory dwell definition must have:

a) a valid memory address (Symbol name, if any, must be in operating system Symbol Table, and resolved address i.e. symbol plus offset must be in a valid range),

b) a value of 1, 2, or 4 for dwell length.

A valid Dwell Table entry which is a null entry must have a zero value for dwell length.

# CFS Memory Dwell Requirements

For detailed MD requirements, please see the "Core Flight System (CFS) Memory Dwell Application Requirements" Document.

# CFS Memory Dwell Deployment Guide

In order to generate dwell streams, MD must receive Wakeup messages, typically from the CFS Scheduler Application. The rate at which these Wakeup messages are sent will determine the fastest rate at which Memory Dwell can sample memory addresses.

In order to generate housekeeping data, MD must receive [Send Housekeeping messages](#AAAAAAAAAH), which are also typically sent from the [CFS Scheduler Application](file:///C:\Users\ejtimmon\Documents\CFE\CFS_Apps\MD\sch\index.html).

MD sends [housekeeping telemetry messages](#AAAAAAAAGO) and MD sends [Dwell Packet telemetry streams.](#AAAAAAAAGP)

The number of dwell packet telemetry streams is determined by the [MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL) configuration parameter and can be a number between 1 and 8 (inclusive).

MD requires one message pipe from CFE Software Bus.

[CFE Table Services](file:///C:\Users\ejtimmon\Documents\CFE\CFS_Apps\MD\cfe\cfetblovr.html)  must be configured so that its [maximum size](#AAAAAAAAID) is large enough to accommodate a single dwell table.

Memory Dwell assumes sufficient pool memory from Table Services for the tables to be single-buffered.

Enable or disable the MD Signature capability by setting the [MD\_SIGNATURE\_OPTION](#AAAAAAAAAO) platform configuration parameter. Note that if the option is set to zero, the command to set the signature will not be compiled, and the field will not be present in dwell packets. When enabled (non-zero), the length of the signature field is specified by the [MD\_SIGNATURE\_FIELD\_LENGTH](#AAAAAAAAAP) platform configuration parameter.

The number of dwell slots in each Dwell Table is specified by the [MD\_DWELL\_TABLE\_SIZE](#AAAAAAAAAM) platform configuration parameter.

The Memory Dwell Performance Id, [MD\_APPMAIN\_PERF\_ID](#AAAAAAAAAC), must be distinct from other spacecraft application performance identifiers. [MD\_APPMAIN\_PERF\_ID](#AAAAAAAAAC) is currently defined in [md\_perfids.h](#AAAAAAAAAB).

# CFS Memory Dwell Commands

The Memory Dwell Application has the following commands:

##### Global [MD\_JAM\_DWELL\_CC](#AAAAAAAABR)

Jam Dwell

##### Global [MD\_NOOP\_CC](#AAAAAAAABK)

Memory Dwell No-Op Command

##### Global [MD\_RESET\_CNTRS\_CC](#AAAAAAAABN)

Memory Dwell Reset Counters Command

##### Global [MD\_START\_DWELL\_CC](#AAAAAAAABO)

Memory Dwell Start Dwell Command

##### Global [MD\_STOP\_DWELL\_CC](#AAAAAAAABQ)

Memory Dwell Stop Dwell Command

# CFS Memory Dwell Telemetry

The Memory Dwell Application sends the following telemetry to the ground:

##### Class [MD\_DwellPkt\_t](#AAAAAAAAGP)

Memory Dwell Telemetry Packet format

##### Class [MD\_HkTlm\_t](#AAAAAAAAGO)

Memory Dwell HK Telemetry format

# CFS Memory Dwell Table Definitions

The Memory Dwell Application uses one type of table, a Memory Dwell Load Table. One table is used for each dwell stream that MD is configured to manage. The [MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL) configuration parameter specifies the number of dwell streams MD will manage. The structure format of a single table entry is as follows:

typedef struct

{

uint16 Length; /\* Length of dwell field in bytes. 0 indicates null entry. \*/

uint16 Delay; /\* Delay before following dwell sample in terms of number of task wakeup calls \*/

CFS\_SymAddr\_t DwellAddress; /\* Dwell Address in #CFS\_SymAddr\_t format \*/

} MD\_TableLoadEntry\_t;

The overall structure format of a Memory Dwell Load Table is as follows:

typedef struct

{

uint32 Enabled; /\* Is table is enabled for dwell?

Valid values are MD\_DWELL\_STREAM\_DISABLED, MD\_DWELL\_STREAM\_ENABLED \*/

MD\_TableLoadEntry\_t Entry[MD\_DWELL\_TABLE\_SIZE]; /\* Array of individual memory dwell specifications \*/

} MD\_DwellTableLoad\_t;

# CFS Memory Dwell Configuration Parameters

The following are configuration parameters used to configure the CFS Memory Dwell Application either for each platform or for a mission as a whole.

##### Global [MD\_DWELL\_TABLE\_SIZE](#AAAAAAAAAM)

Maximum number of dwell specifications (address/delay/length) in a Dwell Table.

##### Global [MD\_ENFORCE\_DWORD\_ALIGN](#AAAAAAAAAN)

Option of whether 32 bit integers must be aligned to 32 bit boundaries. 1 indicates 'yes' (32-bit boundary alignment enforced), 0 indicates 'no' (16-bit boundary alignment enforced).

##### Global [MD\_MISSION\_REV](#AAAAAAAAAQ)

Mission specific version number for MD application

##### Global [MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL)

Number of memory dwell tables.

##### Global [MD\_SIGNATURE\_FIELD\_LENGTH](#AAAAAAAAAP)

Number of characters used in the Signature Field

##### Global [MD\_SIGNATURE\_OPTION](#AAAAAAAAAO)

Option of whether a signature field will be reserved in dwell packets. 1 indicates 'yes', 0 indicates 'no'.

##### Global [MD\_TBL\_FILENAME\_FORMAT](#AAAAAAAAAK)

Memory Dwell Base Filename

# CFS Memory Dwell Constraints

During initialization of CFE Software Bus services, the Memory Dwell task will generate a SysLog message and then terminate if any of the following steps fail:

1) Create command pipe

2) Subscribe to Housekeeping request commands

3) Subscribe to MD ground command packets

4) Subscribe to MD wakeup packets

During initialization of CFE Table Services, the Memory Dwell task will send an event message and then terminate if any of the following steps fail:

1) Register each dwell table

2) Load initial values to each dwell table whenever a recovered table is unavailable.

If a pipe read error is encountered from the CFE Software Bus, the Memory Dwell task will send an error message and then terminate.

# CFS Memory Dwell Frequently Asked Questions

### Questions about CFE Table Services Interface

**(Q) If a dwell stream is modified or defined using jam commands, will those changes and definition be copied to the buffer maintained by Table Services?**

*Yes.*

**(Q) Are changes to dwell tables preserved across processor resets?**

*Dwell table contents that are introduced via table loads are preserved across processor resets. Information modified via jam commands, start commands, and stop commands are also preserved across processor resets.*

### Questions about Command and Telemetry

**(Q) Which MD telemetry can be used to confirm that dwell packet processing is proceeding?**

*The telemetry point should change each time a Wakeup Message is received for all dwell streams that are being actively processed.*

**(Q) What needs to be in place for dwell packets to be issued?**

*1) The dwell table's entry slot must contain a valid entry definition, i.e. a valid memory address and field length.*

*2) The total delay, defined as the total of all individual entry delays from the first slot until the last slot before the terminal entry, must be 1 or more.*

*3) The dwell table must be enabled.*

**(Q) How long will it be before the first dwell packet is issued?**

*The first dwell always takes place when the first wakeup message is received following a* [*Start command*](#AAAAAAAABO) *. Dwelling proceeds until the first entry with a non-zero delay is reached. Thus, anytime a start command is sent for a dwell table that does not have non-zero delays for intermediate entries, a dwell packet will be issued immediately. So, for example, a dwell table that is set up to sample several addresses every 10 cycles, will actually issue dwell packets following the 1st, 11th, and 21st wakeup messages.*

**(Q) What happens if a dwell tream is** [**Started**](#AAAAAAAABO) **,** [**Stopped**](#AAAAAAAABQ) **, and then** [**Started**](#AAAAAAAABO)  **again?**

*Whenever a start command is received, whether or not there is an intervening stop command, the dwell packet processing starts over. The countdown timer is reinitialized, and the dwelling begins again with the first table entry.*

**(Q) What if an operating system does not support referencing memory addresses by symbolic names?**

*The symbol name portion of the memory address in a table load or* [*Jam command*](#AAAAAAAABR)  *must be set to a null string.*

### Questions about Specifying Memory Dwell Streams

**(Q) What confirmation is available for changes made using a** [**Jam command**](#AAAAAAAABR) **?**

*1) An information event,* [*MD\_JAM\_DWELL\_INF\_EID*](#AAAAAAAAFB)*, will be issued showing the table number and entry number affected.*

*2) The following housekeeping parameters for the corresponding dwell table may be affected:*

**(Q) Does MD support supercommutation? What is it? How do you set up a dwell table to achieve supercommutation?**

*Yes it does. Supercommutation is the insertion of multiple instances of the same dwell address in the same packet. To define a dwell table for supercommutation, associate a non-zero delay multiple for each instance of the dwell address in the table.*

**(Q) How large is a dwell table?**

*The size of a dwell table depends on:*

*1) the* [*size of a signature field,*](#AAAAAAAAAP)

*2) the* [*size of symbolic addresses*](#AAAAAAAAID)  *if enabled,*

*3) the* [*number of dwell tables*](#AAAAAAAAAL)  *which have been configured,*

*4) the* [*number of entries in a dwell table.*](#AAAAAAAAAM)

*It is also possible that a compiler may insert padding between fields for alignment purposes, which will add to the size. If #MD\_DEBUG is set to a non-zero value, the size of the dwell table will be issued via #OS\_printf on initialization.*

**(Q) What is the fastest rate at which MD can sample memory addresses?**

*The fastest that MD can sample memory address is the rate at which* [*Wakeup messages*](#AAAAAAAAAI)  *are received. To achieve this fastest rate, the dwell table must be defined to have a total delay multiple of 1.*

**(Q) How would a simple dwell table be defined to sample several memory addresses at the same time?**

*For each memory address to be sampled, define a dwell entry beginning with the first entry, entry #1. Enter the dwell field length value of 1, 2, or 4. Note that if the desired memory region is larger than 4, it must be broken down into consecutive dwell entries. Enter the memory address using the numerical value, a symbolic value, or a symbolic value and numerical offset combination. In each of the entries that are not the last dwell, enter a zero value for the delay. For the last dwell entry, enter a value corresponding to the multiple of the wakeup call rate, which is the desired issue rate for the dwell packets. If the last dwell entry is not the last entry in the table, then the next entry must be a "null entry" with a zero value for the field length to signify that this is the end of the dwell packet.*

**(Q) Does MD accept tables with non-null entries past a terminator entry?**

*Yes, a non-null entry is accepted past a terminator entry as long as that entry itself is valid. To be valid, the entry must have valid values for dwell address, and field length.*

# MD Telemetry Mnemonic Cross Reference

##### Global [MD\_DwellPkt\_t::AddrCount](#AAAAAAAAJD)

##### Global [MD\_DwellPkt\_t::ByteCount](#AAAAAAAAJF)

##### Global [MD\_DwellPkt\_t::Data](#AAAAAAAAJH) [MD\_DWELL\_TABLE\_SIZE \*4]

##### Global [MD\_DwellPkt\_t::Rate](#AAAAAAAAJJ)

##### Global [MD\_DwellPkt\_t::TableId](#AAAAAAAAJL)

##### Global [MD\_HkTlm\_t::ByteCount](#AAAAAAAAJN) [MD\_NUM\_DWELL\_TABLES]

##### Global [MD\_HkTlm\_t::Countdown](#AAAAAAAAJP) [MD\_NUM\_DWELL\_TABLES]

##### Global [MD\_HkTlm\_t::DwellEnabledMask](#AAAAAAAAJR)

##### Global [MD\_HkTlm\_t::DwellPktOffset](#AAAAAAAAJT) [MD\_NUM\_DWELL\_TABLES]

##### Global [MD\_HkTlm\_t::DwellTblAddrCount](#AAAAAAAAJV) [MD\_NUM\_DWELL\_TABLES]

##### Global [MD\_HkTlm\_t::DwellTblEntry](#AAAAAAAAJX) [MD\_NUM\_DWELL\_TABLES]

##### Global [MD\_HkTlm\_t::InvalidCmdCntr](#AAAAAAAAIF)

##### Global [MD\_HkTlm\_t::NumWaitsPerPkt](#AAAAAAAAKA) [MD\_NUM\_DWELL\_TABLES]

##### Global [MD\_HkTlm\_t::ValidCmdCntr](#AAAAAAAAIE)

# MD Event Message Cross Reference

##### Global [MD\_CANT\_RESOLVE\_JAM\_ADDR\_ERR\_EID](#AAAAAAAAFG)

'Jam Cmd rejected because symbolic address 's' couldn't be resolved'

##### Global [MD\_CC\_NOT\_IN\_LOOP\_ERR\_EID](#AAAAAAAAEX)

'Command Code d not found in command processing loop'

##### Global [MD\_CC\_NOT\_IN\_TBL\_ERR\_EID](#AAAAAAAAEW)

'Command Code d not found in MD\_CmdHandlerTbl structure'

##### Global [MD\_CMD\_LEN\_ERR\_EID](#AAAAAAAAEZ)

'Cmd Msg with Bad length Rcvd: ID = 0x%08X, CC = d, Exp Len = d, Len = d'

##### Global [MD\_CREATE\_PIPE\_ERR\_EID](#AAAAAAAAGJ)

'Failed to create pipe. RC = d'

##### Global [MD\_DWELL\_LOOP\_GET\_DWELL\_DATA\_ERR\_EID](#AAAAAAAAGI)

'Dwell Table failed to read entry d in table d'

##### Global [MD\_DWELL\_TBL\_INF\_EID](#AAAAAAAAFW)

'MD Dwell Tbl verify results: good = %d, bad = %d, unused = %d'

##### Global [MD\_DWELL\_TBL\_TOO\_LARGE\_CRIT\_EID](#AAAAAAAAEN)

'Dwell Table(s) are too large to register: d > d bytes, d > d entries'

##### Global [MD\_EMPTY\_TBLMASK\_ERR\_EID](#AAAAAAAAEU)

's command rejected because no tables were specified in table mask (0x%04X)'

##### Global [MD\_INIT\_INF\_EID](#AAAAAAAAEJ)

'MD Initialized. Version d.d.d.d'

##### Global [MD\_INIT\_TBL\_FILENAME\_ERR\_EID](#AAAAAAAAGG)

'TblFileName could not be made. Err=0x%08X, Idx=%d"

**Global MD\_INIT\_TBL\_NAME\_ERR\_EID**

'TableName could not be made. Err=0x\%08X, Idx=\%d"

##### Global [MD\_INVALID\_ENTRY\_ARG\_ERR\_EID](#AAAAAAAAFE)

'Jam Cmd rejected due to invalid Entry Id arg = d (Expect 1.. d)'

##### Global [MD\_INVALID\_JAM\_ADDR\_ERR\_EID](#AAAAAAAAFH)

'Jam Cmd rejected because address 0x%08X is not in a valid range'

##### Global [MD\_INVALID\_JAM\_TABLE\_ERR\_EID](#AAAAAAAAFD)

'Jam Cmd rejected due to invalid Tbl Id arg = d (Expect 1.. d)'

##### Global [MD\_INVALID\_LEN\_ARG\_ERR\_EID](#AAAAAAAAFF)

'Jam Cmd rejected due to invalid Field Length arg = d (Expect 0,1,2,or 4)'

##### Global [MD\_INVALID\_SIGNATURE\_LENGTH\_ERR\_EID](#AAAAAAAAFT)

'Set Signature cmd rejected due to invalid Signature length'

##### Global [MD\_INVALID\_SIGNATURE\_TABLE\_ERR\_EID](#AAAAAAAAFS)

'Set Signature cmd rejected due to invalid Tbl Id arg = d (Expect 1.. d)'

##### Global [MD\_JAM\_ADDR\_NOT\_16BIT\_ERR\_EID](#AAAAAAAAFJ)

'Jam Cmd rejected because address 0x%08X is not 16-bit aligned'

##### Global [MD\_JAM\_ADDR\_NOT\_32BIT\_ERR\_EID](#AAAAAAAAFI)

'Jam Cmd rejected because address 0x%08X is not 32-bit aligned'

##### Global [MD\_JAM\_DWELL\_ERR\_EID](#AAAAAAAAGD)

'Failed Jam to Dwell Tbl#d Entry #d'

##### Global [MD\_JAM\_DWELL\_INF\_EID](#AAAAAAAAFB)

'Successful Jam to Dwell Tbl#d Entry #d'

##### Global [MD\_JAM\_NULL\_DWELL\_ERR\_EID](#AAAAAAAAGE)

'Failed Jam of a Null Dwell Entry to Dwell Tbl#d Entry #d'

##### Global [MD\_JAM\_NULL\_DWELL\_INF\_EID](#AAAAAAAAFC)

'Successful Jam of a Null Dwell Entry to Dwell Tbl#d Entry #d'

##### Global [MD\_MID\_ERR\_EID](#AAAAAAAAEV)

'Msg with Invalid message ID Rcvd – ID = 0x%08X'

##### Global [MD\_MSG\_LEN\_ERR\_EID](#AAAAAAAAFA)

'Msg with Bad length Rcvd: ID = 0x%08X, Exp Len = d, Len = d'

##### Global [MD\_NO\_TBL\_COPY\_ERR\_EID](#AAAAAAAAFK)

'Didn't update MD tbl #d due to unexpected CFE\_TBL\_GetAddress return: 0x%08X'

##### Global [MD\_NOOP\_INF\_EID](#AAAAAAAAEQ)

'No-op command, Version d.d.d.d'

##### Global [MD\_PIPE\_ERR\_EID](#AAAAAAAAEK)

'SB Pipe Read Error, App will exit. Pipe Return Status = 0x%08X'

##### Global [MD\_RANGE\_ERR\_EID](#AAAAAAAAFN)

'Dwell Table rejected because address (sym='s'/offset=0x%08X) in entry #d was out of range'

##### Global [MD\_RECOVERED\_TBL\_NOT\_VALID\_ERR\_EID](#AAAAAAAAEM)

'MD App will reinitialize Dwell Table #d because recovered table is not valid'

##### Global [MD\_RECOVERED\_TBL\_VALID\_INF\_EID](#AAAAAAAAEL)

'Recovered Dwell Table #d is valid and has been copied to the MD App'

##### Global [MD\_RESET\_CNTRS\_DBG\_EID](#AAAAAAAAER)

'Reset Counters Cmd Received'

##### Global [MD\_RESOLVE\_ERR\_EID](#AAAAAAAAFM)

'Dwell Table rejected because address (sym='s'/offset=0x%08X) in entry #d couldn't be resolved'

##### Global [MD\_SET\_SIGNATURE\_ERR\_EID](#AAAAAAAAGC)

'Failed to set signature for Dwell Tbl#%d. Update returned 0x%08X'

##### Global [MD\_SET\_SIGNATURE\_INF\_EID](#AAAAAAAAFR)

'Successfully set signature for Dwell Tbl#d to 's''

##### Global [MD\_START\_DWELL\_ERR\_EID](#AAAAAAAAGA)

'Start Dwell Table for mask 0x%04X failed for %d of %d tables'

##### Global [MD\_START\_DWELL\_INF\_EID](#AAAAAAAAES)

'Start Dwell Table command processed successfully for table mask 0x%04X'

##### Global [MD\_STOP\_DWELL\_ERR\_EID](#AAAAAAAAGB)

'Stop Dwell Table for mask 0x%04X failed for %d of %d tables'

##### Global [MD\_STOP\_DWELL\_INF\_EID](#AAAAAAAAET)

'Stop Dwell Table command processed successfully for table mask 0x%04X'

##### Global [MD\_SUB\_CMD\_ERR\_EID](#AAAAAAAAGL)

'Failed to subscribe to commands. RC = d'

##### Global [MD\_SUB\_HK\_ERR\_EID](#AAAAAAAAGK)

'Failed to subscribe to HK requests. RC = d'

##### Global [MD\_SUB\_WAKEUP\_ERR\_EID](#AAAAAAAAGM)

'Failed to subscribe to wakeup messages. RC = d'

##### Global [MD\_TBL\_ALIGN\_ERR\_EID](#AAAAAAAAFQ)

'Dwell Table rejected because address (sym='s'/offset=0x08X) in entry #d is not properly aligned for a d-byte dwell'

##### Global [MD\_TBL\_ENA\_FLAG\_EID](#AAAAAAAAFP)

'Dwell Table rejected because value of enable flag (d) is invalid'

##### Global [MD\_TBL\_HAS\_LEN\_ERR\_EID](#AAAAAAAAFO)

'Dwell Table rejected because length (d) in entry #d was invalid'

##### Global [MD\_TBL\_INIT\_INF\_EID](#AAAAAAAAEP)

'Dwell Tables Recovered: d, Dwell Tables Initialized: d'

##### Global [MD\_TBL\_REGISTER\_CRIT\_EID](#AAAAAAAAEO)

'CFE\_TBL\_Register error 0x%08X received for tbl#d'

##### Global [MD\_TBL\_SIG\_LEN\_ERR\_EID](#AAAAAAAAFU)

'Dwell Table rejected because Signature length was invalid'

##### Global [MD\_TBL\_STATUS\_ERR\_EID](#AAAAAAAAEY)

'Received unexpected error 0x%08X from CFE\_TBL\_GetStatus for tbl #d'

##### Global [MD\_TBL\_VAL\_NULL\_PTR\_ERR\_EID](#AAAAAAAAGH)

'Dwell Table rejected because of null table pointer'

##### Global [MD\_UPDATE\_TBL\_DWELL\_ERR\_EID](#AAAAAAAAFY)

'MD\_UpdateTableDwellEntry, TableIndex d: CFE\_TBL\_GetAddress Returned 0x%08x'

##### Global [MD\_UPDATE\_TBL\_EN\_ERR\_EID](#AAAAAAAAFX)

'MD\_UpdateTableEnabledField, TableIndex %d: CFE\_TBL\_GetAddress Returned 0x%08x'

##### Global [MD\_UPDATE\_TBL\_SIG\_ERR\_EID](#AAAAAAAAFZ)

'MD\_UpdateTableSignature, TableIndex d: CFE\_TBL\_GetAddress Returned 0x%08x'

##### Global [MD\_ZERO\_RATE\_CMD\_INF\_EID](#AAAAAAAAFV)

'Dwell Table d is enabled with a delay of zero so no processing will occur'

##### Global [MD\_ZERO\_RATE\_TBL\_INF\_EID](#AAAAAAAAFL)

'Dwell Table is enabled but no processing will occur for table being loaded (rate is zero)'

# Data Structure Documentation

## MD\_AppData\_t Struct Reference

MD global data structure.

#include <md\_app.h>

### Data Fields

* uint8 [CmdCounter](#AAAAAAAAMF)

*MD Application Command Counter.*

* uint8 [ErrCounter](#AAAAAAAAMG)

*MD Application Error Counter.*

* [MD\_HkTlm\_t](#AAAAAAAAGO) [HkPkt](#AAAAAAAAMH)

*Housekeeping telemetry packet.*

* CFE\_SB\_PipeId\_t [CmdPipe](#AAAAAAAAMI)

*Command pipe ID.*

* [MD\_DwellPacketControl\_t](#AAAAAAAABW) [MD\_DwellTables](#AAAAAAAAMJ) [[MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL)]

*Array of packet control structures.*

* [MD\_DwellPkt\_t](#AAAAAAAAGP) [MD\_DwellPkt](#AAAAAAAAMK) [[MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL)]

*Array of dwell packet structures.*

* uint32 [RunStatus](#AAAAAAAAML)

*Application run status.*

* char [MD\_TableName](#AAAAAAAAMM) [[MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL)][CFE\_MISSION\_TBL\_MAX\_NAME\_LENGTH+1]

*Array of table names used for TBL Services.*

* CFE\_TBL\_Handle\_t [MD\_TableHandle](#AAAAAAAAMN) [[MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL)]

*Array of handle ids provided by TBL Services.*

### Detailed Description

MD global data structure.

### Field Documentation

#### uint8 MD\_AppData\_t::CmdCounter

MD Application Command Counter.

#### CFE\_SB\_PipeId\_t MD\_AppData\_t::CmdPipe

Command pipe ID.

#### uint8 MD\_AppData\_t::ErrCounter

MD Application Error Counter.

#### [MD\_HkTlm\_t](#AAAAAAAAGO) MD\_AppData\_t::HkPkt

Housekeeping telemetry packet.

#### [MD\_DwellPkt\_t](#AAAAAAAAGP) MD\_AppData\_t::MD\_DwellPkt[[MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL)]

Array of dwell packet structures.

#### [MD\_DwellPacketControl\_t](#AAAAAAAABW) MD\_AppData\_t::MD\_DwellTables[[MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL)]

Array of packet control structures.

#### CFE\_TBL\_Handle\_t MD\_AppData\_t::MD\_TableHandle[[MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL)]

Array of handle ids provided by TBL Services.

#### char MD\_AppData\_t::MD\_TableName[[MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL)][CFE\_MISSION\_TBL\_MAX\_NAME\_LENGTH+1]

Array of table names used for TBL Services.

#### uint32 MD\_AppData\_t::RunStatus

Application run status.

#### The documentation for this struct was generated from the following file:

* fsw/src/[md\_app.h](#AAAAAAAABU)

## MD\_CmdHandlerTblRec\_t Struct Reference

#include <md\_app.h>

### Data Fields

* uint32 [MsgId](#AAAAAAAAMO)

*Acceptable Message ID.*

* uint32 [CmdCode](#AAAAAAAAMP)

*Acceptable Command Code (if necessary)*

* uint32 [ExpectedLength](#AAAAAAAAMQ)

*Expected Message Length (in bytes) including message header.*

* [MD\_MsgType\_t](#AAAAAAAACJ) [MsgTypes](#AAAAAAAAMR)

*Message Type (i.e. - with/without Cmd Code)*

### Field Documentation

#### uint32 MD\_CmdHandlerTblRec\_t::CmdCode

Acceptable Command Code (if necessary)

#### uint32 MD\_CmdHandlerTblRec\_t::ExpectedLength

Expected Message Length (in bytes) including message header.

#### uint32 MD\_CmdHandlerTblRec\_t::MsgId

Acceptable Message ID.

#### [MD\_MsgType\_t](#AAAAAAAACJ) MD\_CmdHandlerTblRec\_t::MsgTypes

Message Type (i.e. - with/without Cmd Code)

#### The documentation for this struct was generated from the following file:

* fsw/src/[md\_app.h](#AAAAAAAABU)

## MD\_CmdJam\_t Struct Reference

#include <md\_msg.h>

### Data Fields

* CFE\_MSG\_CommandHeader\_t [Header](#AAAAAAAAMS)

*cFE Software Bus Command Message Header*

* uint16 [TableId](#AAAAAAAAMT)

*Table Id: 1..[MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL).*

* uint16 [EntryId](#AAAAAAAAMU)

*Address index: 1..[MD\_DWELL\_TABLE\_SIZE](#AAAAAAAAAM).*

* uint16 [FieldLength](#AAAAAAAAMV)

*Length of Dwell Field : 0, 1, 2, or 4.*

* uint16 [DwellDelay](#AAAAAAAAMW)

*Dwell Delay (number of task wakeup calls before following dwell)*

* CFS\_SymAddr\_t [DwellAddress](#AAAAAAAAMX)

*Dwell Address in #CFS\_SymAddr\_t format.*

### Detailed Description

Jam Dwell Command.

For command details, see [MD\_JAM\_DWELL\_CC](#AAAAAAAABR)

### Field Documentation

#### CFS\_SymAddr\_t MD\_CmdJam\_t::DwellAddress

Dwell Address in #CFS\_SymAddr\_t format.

#### uint16 MD\_CmdJam\_t::DwellDelay

Dwell Delay (number of task wakeup calls before following dwell)

#### uint16 MD\_CmdJam\_t::EntryId

Address index: 1..[MD\_DWELL\_TABLE\_SIZE](#AAAAAAAAAM).

#### uint16 MD\_CmdJam\_t::FieldLength

Length of Dwell Field : 0, 1, 2, or 4.

#### CFE\_MSG\_CommandHeader\_t MD\_CmdJam\_t::Header

cFE Software Bus Command Message Header

#### uint16 MD\_CmdJam\_t::TableId

Table Id: 1..[MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL).

#### The documentation for this struct was generated from the following file:

* fsw/src/[md\_msg.h](#AAAAAAAAGN)

## MD\_CmdStartStop\_t Struct Reference

#include <md\_msg.h>

### Data Fields

* CFE\_MSG\_CommandHeader\_t [Header](#AAAAAAAAMY)

*cFE Software Bus Command Message Header*

* uint16 [TableMask](#AAAAAAAAMZ)

*0x0001=TBL1 bit, 0x0002=TBL2 bit,0x0004=TBL3 bit,0x0008=TBL4 enable bit, etc.*

* uint16 [Padding](#AAAAAAAANA)

*structure padding*

### Detailed Description

Start and Stop Dwell Commands.

For command details, see [MD\_START\_DWELL\_CC](#AAAAAAAABO) and [MD\_STOP\_DWELL\_CC](#AAAAAAAABQ)

### Field Documentation

#### CFE\_MSG\_CommandHeader\_t MD\_CmdStartStop\_t::Header

cFE Software Bus Command Message Header

#### uint16 MD\_CmdStartStop\_t::Padding

structure padding

#### uint16 MD\_CmdStartStop\_t::TableMask

0x0001=TBL1 bit, 0x0002=TBL2 bit,0x0004=TBL3 bit,0x0008=TBL4 enable bit, etc.

#### The documentation for this struct was generated from the following file:

* fsw/src/[md\_msg.h](#AAAAAAAAGN)

## MD\_DwellControlEntry\_t Struct Reference

#include <md\_app.h>

### Data Fields

* uint16 [Length](#AAAAAAAANB)

*Length of dwell field in bytes. 0 indicates null entry.*

* uint16 [Delay](#AAAAAAAANC)

*Delay before following dwell sample in terms of number of task wakeup calls.*

* cpuaddr [ResolvedAddress](#AAAAAAAAND)

*Dwell address in numerical form.*

### Detailed Description

MD structure for specifying individual memory dwell.

### Field Documentation

#### uint16 MD\_DwellControlEntry\_t::Delay

Delay before following dwell sample in terms of number of task wakeup calls.

#### uint16 MD\_DwellControlEntry\_t::Length

Length of dwell field in bytes. 0 indicates null entry.

#### cpuaddr MD\_DwellControlEntry\_t::ResolvedAddress

Dwell address in numerical form.

#### The documentation for this struct was generated from the following file:

* fsw/src/[md\_app.h](#AAAAAAAABU)

## MD\_DwellPacketControl\_t Struct Reference

#include <md\_app.h>

### Data Fields

* uint16 [Enabled](#AAAAAAAANE)

*Is table is enabled for dwell? Valid values are* [*MD\_DWELL\_STREAM\_DISABLED*](#AAAAAAAACH)*,* [*MD\_DWELL\_STREAM\_ENABLED*](#AAAAAAAACI)*.*

* uint16 [AddrCount](#AAAAAAAANF)

*Number of dwell addresses to telemeter.*

* uint32 [Rate](#AAAAAAAANG)

*Packet issuance interval in terms of number of task wakeup calls.*

* uint32 [Countdown](#AAAAAAAANH)

*Counts down from Rate to 0, then read next address.*

* uint16 [PktOffset](#AAAAAAAANI)

*Tracks where to write next data in dwell pkt.*

* uint16 [CurrentEntry](#AAAAAAAANJ)

*Current entry in dwell table.*

* uint16 [DataSize](#AAAAAAAANK)

*Total number of data bytes specified in dwell table.*

* uint16 [Filler](#AAAAAAAANL)

*Preserves alignment.*

* [MD\_DwellControlEntry\_t](#AAAAAAAABV) [Entry](#AAAAAAAANM) [[MD\_DWELL\_TABLE\_SIZE](#AAAAAAAAAM)]

*Array of individual memory dwell specifications.*

### Detailed Description

MD structure for controlling dwell operations.

### Field Documentation

#### uint16 MD\_DwellPacketControl\_t::AddrCount

Number of dwell addresses to telemeter.

#### uint32 MD\_DwellPacketControl\_t::Countdown

Counts down from Rate to 0, then read next address.

#### uint16 MD\_DwellPacketControl\_t::CurrentEntry

Current entry in dwell table.

#### uint16 MD\_DwellPacketControl\_t::DataSize

Total number of data bytes specified in dwell table.

#### uint16 MD\_DwellPacketControl\_t::Enabled

Is table is enabled for dwell? Valid values are [MD\_DWELL\_STREAM\_DISABLED](#AAAAAAAACH), [MD\_DWELL\_STREAM\_ENABLED](#AAAAAAAACI).

#### [MD\_DwellControlEntry\_t](#AAAAAAAABV) MD\_DwellPacketControl\_t::Entry[[MD\_DWELL\_TABLE\_SIZE](#AAAAAAAAAM)]

Array of individual memory dwell specifications.

#### uint16 MD\_DwellPacketControl\_t::Filler

Preserves alignment.

#### uint16 MD\_DwellPacketControl\_t::PktOffset

Tracks where to write next data in dwell pkt.

#### uint32 MD\_DwellPacketControl\_t::Rate

Packet issuance interval in terms of number of task wakeup calls.

#### The documentation for this struct was generated from the following file:

* fsw/src/[md\_app.h](#AAAAAAAABU)

## MD\_DwellPkt\_t Struct Reference

#include <md\_msg.h>

### Data Fields

* CFE\_MSG\_TelemetryHeader\_t [TlmHeader](#AAAAAAAANN)

*cFE SB Tlm Msg Hdr*

* uint8 [TableId](#AAAAAAAAJL)

*TableId from 1 to* [*MD\_NUM\_DWELL\_TABLES*](#AAAAAAAAAL)*.*

* uint8 [AddrCount](#AAAAAAAAJD)

*Number of addresses being sent - 1..[MD\_DWELL\_TABLE\_SIZE](#AAAAAAAAAM) valid.*

* uint16 [ByteCount](#AAAAAAAAJF)

*Number of bytes of dwell data contained in packet.*

* uint32 [Rate](#AAAAAAAAJJ)

*Number of counts between packet sends.*

* uint8 [Data](#AAAAAAAAJH) [[MD\_DWELL\_TABLE\_SIZE](#AAAAAAAAAM) \*4]

*Dwell data ( number of bytes varies up to MD\_DWELL\_TABLE\_SIZE \*4)*

### Detailed Description

##### [Purpose](#cfsmdtlm__cfsmdtlm000002):

Memory Dwell Telemetry Packet format

### Field Documentation

#### uint8 MD\_DwellPkt\_t::AddrCount

Number of addresses being sent - 1..[MD\_DWELL\_TABLE\_SIZE](#AAAAAAAAAM) valid.

##### [Telemetry Mnemonic(s)](#mdtlmmnems__mdtlmmnems000011):

#### uint16 MD\_DwellPkt\_t::ByteCount

Number of bytes of dwell data contained in packet.

##### [Telemetry Mnemonic(s)](#mdtlmmnems__mdtlmmnems000012):

#### uint8 MD\_DwellPkt\_t::Data[[MD\_DWELL\_TABLE\_SIZE](#AAAAAAAAAM) \*4]

Dwell data ( number of bytes varies up to MD\_DWELL\_TABLE\_SIZE \*4)

##### [Telemetry Mnemonic(s)](#mdtlmmnems__mdtlmmnems000014):

#### uint32 MD\_DwellPkt\_t::Rate

Number of counts between packet sends.

##### [Telemetry Mnemonic(s)](#mdtlmmnems__mdtlmmnems000013):

#### uint8 MD\_DwellPkt\_t::TableId

TableId from 1 to [MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL).

##### [Telemetry Mnemonic(s)](#mdtlmmnems__mdtlmmnems000010):

#### CFE\_MSG\_TelemetryHeader\_t MD\_DwellPkt\_t::TlmHeader

cFE SB Tlm Msg Hdr

#### The documentation for this struct was generated from the following file:

* fsw/src/[md\_msg.h](#AAAAAAAAGN)

## MD\_DwellTableLoad\_t Struct Reference

#include <md\_tbldefs.h>

### Data Fields

* uint32 [Enabled](#AAAAAAAANO)

*Is table is enabled for dwell? Valid values are MD\_DWELL\_STREAM\_DISABLED, MD\_DWELL\_STREAM\_ENABLED.*

* [MD\_TableLoadEntry\_t](#AAAAAAAADV) [Entry](#AAAAAAAANP) [[MD\_DWELL\_TABLE\_SIZE](#AAAAAAAAAM)]

*Array of individual memory dwell specifications.*

### Detailed Description

Memory Dwell Table Load structure.

##### Description

To be valid, each of the Entry structures must be valid. See [MD\_TableLoadEntry\_t](#AAAAAAAADV) for details. Tables will be processed beginning with the first entry if it is non-null and continuing until the first null entry is reached. Note that non-null entries may follow a terminator entry; however they will not be processed.

In order to be processed, all of the following must be true:

* There are one or more non-null entries beginning with the first entry of the table.
* The sum of individual entry delays, beginning with the first entry and up until the terminator entry or the end of the table, must be non-zero.
* The table's Enabled field must be set to TRUE. This is initially set in the load, and is controlled with the [MD\_START\_DWELL\_CC](#AAAAAAAABO) and [MD\_STOP\_DWELL\_CC](#AAAAAAAABQ) commands.

### Field Documentation

#### uint32 MD\_DwellTableLoad\_t::Enabled

Is table is enabled for dwell? Valid values are MD\_DWELL\_STREAM\_DISABLED, MD\_DWELL\_STREAM\_ENABLED.

#### [MD\_TableLoadEntry\_t](#AAAAAAAADV) MD\_DwellTableLoad\_t::Entry[[MD\_DWELL\_TABLE\_SIZE](#AAAAAAAAAM)]

Array of individual memory dwell specifications.

#### The documentation for this struct was generated from the following file:

* fsw/src/[md\_tbldefs.h](#AAAAAAAAGT)

## MD\_HkTlm\_t Struct Reference

#include <md\_msg.h>

### Data Fields

* CFE\_MSG\_TelemetryHeader\_t [TlmHeader](#AAAAAAAANT)

*cFE SB Tlm Msg Hdr*

* uint8 [InvalidCmdCntr](#AAAAAAAAIF)

*Count of invalid commands received.*

* uint8 [ValidCmdCntr](#AAAAAAAAIE)

*Count of valid commands received.*

* uint16 [DwellEnabledMask](#AAAAAAAAJR)

*Each bit in bit mask enables a table 0x0001=TBL1 enable bit,0x0002=TBL2 enable bit 0x0004=TBL3 enable bit,0x0008=TBL4 enable bit, etc.*

* uint16 [DwellTblAddrCount](#AAAAAAAAJV) [[MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL)]

*Number of dwell addresses in table.*

* uint16 [NumWaitsPerPkt](#AAAAAAAAKA) [[MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL)]

*Number of delay counts in table.*

* uint16 [ByteCount](#AAAAAAAAJN) [[MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL)]

*Number of bytes of data specified by table.*

* uint16 [DwellPktOffset](#AAAAAAAAJT) [[MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL)]

*Current write offset within dwell pkt data region.*

* uint16 [DwellTblEntry](#AAAAAAAAJX) [[MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL)]

*Next dwell table entry to be processed.*

* uint16 [Countdown](#AAAAAAAAJP) [[MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL)]

*Current value of countdown timer.*

### Detailed Description

##### [Purpose](#cfsmdtlm__cfsmdtlm000001):

Memory Dwell HK Telemetry format

### Field Documentation

#### uint16 MD\_HkTlm\_t::ByteCount[[MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL)]

Number of bytes of data specified by table.

##### [Telemetry Mnemonic(s)](#mdtlmmnems__mdtlmmnems000006):

#### uint16 MD\_HkTlm\_t::Countdown[[MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL)]

Current value of countdown timer.

##### [Telemetry Mnemonic(s)](#mdtlmmnems__mdtlmmnems000009):

#### uint16 MD\_HkTlm\_t::DwellEnabledMask

Each bit in bit mask enables a table 0x0001=TBL1 enable bit,0x0002=TBL2 enable bit 0x0004=TBL3 enable bit,0x0008=TBL4 enable bit, etc.

##### [Telemetry Mnemonic(s)](#mdtlmmnems__mdtlmmnems000003):

#### uint16 MD\_HkTlm\_t::DwellPktOffset[[MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL)]

Current write offset within dwell pkt data region.

##### [Telemetry Mnemonic(s)](#mdtlmmnems__mdtlmmnems000007):

#### uint16 MD\_HkTlm\_t::DwellTblAddrCount[[MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL)]

Number of dwell addresses in table.

##### [Telemetry Mnemonic(s)](#mdtlmmnems__mdtlmmnems000004):

#### uint16 MD\_HkTlm\_t::DwellTblEntry[[MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL)]

Next dwell table entry to be processed.

##### [Telemetry Mnemonic(s)](#mdtlmmnems__mdtlmmnems000008):

#### uint8 MD\_HkTlm\_t::InvalidCmdCntr

Count of invalid commands received.

##### [Telemetry Mnemonic(s)](#mdtlmmnems__mdtlmmnems000001):

#### uint16 MD\_HkTlm\_t::NumWaitsPerPkt[[MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL)]

Number of delay counts in table.

##### [Telemetry Mnemonic(s)](#mdtlmmnems__mdtlmmnems000005):

#### CFE\_MSG\_TelemetryHeader\_t MD\_HkTlm\_t::TlmHeader

cFE SB Tlm Msg Hdr

#### uint8 MD\_HkTlm\_t::ValidCmdCntr

Count of valid commands received.

##### [Telemetry Mnemonic(s)](#mdtlmmnems__mdtlmmnems000002):

#### The documentation for this struct was generated from the following file:

* fsw/src/[md\_msg.h](#AAAAAAAAGN)

## MD\_NoArgsCmd\_t Struct Reference

#include <md\_msg.h>

### Data Fields

* CFE\_MSG\_CommandHeader\_t [CmdHeader](#AAAAAAAANU)

*cFE Software Bus Command Message Header*

### Detailed Description

Generic "no arguments" command.

This command structure is used for commands that do not have any parameters. This includes:

1. The Housekeeping Request Message

2. The Wakeup Message

3. The No-Op Command (For details, see [MD\_NOOP\_CC](#AAAAAAAABK))

4. The Reset Counters Command (For details, see [MD\_RESET\_CNTRS\_CC](#AAAAAAAABN))

### Field Documentation

#### CFE\_MSG\_CommandHeader\_t MD\_NoArgsCmd\_t::CmdHeader

cFE Software Bus Command Message Header

#### The documentation for this struct was generated from the following file:

* fsw/src/[md\_msg.h](#AAAAAAAAGN)

## MD\_TableLoadEntry\_t Struct Reference

#include <md\_tbldefs.h>

### Data Fields

* uint16 [Length](#AAAAAAAANV)

*Length of dwell field in bytes. 0 indicates null entry.*

* uint16 [Delay](#AAAAAAAANW)

*Delay before following dwell sample in terms of number of task wakeup calls.*

* CFS\_SymAddr\_t [DwellAddress](#AAAAAAAANX)

*Dwell Address in #CFS\_SymAddr\_t format.*

### Field Documentation

#### uint16 MD\_TableLoadEntry\_t::Delay

Delay before following dwell sample in terms of number of task wakeup calls.

#### CFS\_SymAddr\_t MD\_TableLoadEntry\_t::DwellAddress

Dwell Address in #CFS\_SymAddr\_t format.

#### uint16 MD\_TableLoadEntry\_t::Length

Length of dwell field in bytes. 0 indicates null entry.

#### The documentation for this struct was generated from the following file:

* fsw/src/[md\_tbldefs.h](#AAAAAAAAGT)

# File Documentation

## fsw/mission\_inc/md\_perfids.h File Reference

### Macros

* #define [MD\_APPMAIN\_PERF\_ID](#AAAAAAAAAC)  26

*CFS Memory Dwell Performance ID.*

### Macro Definition Documentation

#### #define MD\_APPMAIN\_PERF\_ID  26

CFS Memory Dwell Performance ID.

## fsw/platform\_inc/md\_msgids.h File Reference

### Macros

* MD Telemetry Message Ids#define [MD\_HK\_TLM\_MID](#AAAAAAAAAE)  0x0890

*Message Id for Memory Dwell's housekeeping pkt.*

* #define [MD\_DWELL\_PKT\_MID\_BASE](#AAAAAAAAAF)  0x0891

*Base Message Id for Memory Dwell's dwell packets. MIDs will be base, base + 1, etc.*

* MD Command Message Ids#define [MD\_CMD\_MID](#AAAAAAAAAG)  0x1890

*Message Id for Memory Dwell's ground command.*

* #define [MD\_SEND\_HK\_MID](#AAAAAAAAAH)  0x1891

*Message Id for Memory Dwell's 'Send Housekeeping' message.*

* #define [MD\_WAKEUP\_MID](#AAAAAAAAAI)  0x1892

*Message Id for Memory Dwell's wakeup message.*

### Macro Definition Documentation

#### #define MD\_CMD\_MID  0x1890

Message Id for Memory Dwell's ground command.

#### #define MD\_DWELL\_PKT\_MID\_BASE  0x0891

Base Message Id for Memory Dwell's dwell packets. MIDs will be base, base + 1, etc.

#### #define MD\_HK\_TLM\_MID  0x0890

Message Id for Memory Dwell's housekeeping pkt.

#### #define MD\_SEND\_HK\_MID  0x1891

Message Id for Memory Dwell's 'Send Housekeeping' message.

#### #define MD\_WAKEUP\_MID  0x1892

Message Id for Memory Dwell's wakeup message.

## fsw/platform\_inc/md\_platform\_cfg.h File Reference

### Macros

* #define [MD\_TBL\_FILENAME\_FORMAT](#AAAAAAAAAK)  "/cf/md\_dw%02d.tbl"
* #define [MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL)  4
* #define [MD\_DWELL\_TABLE\_SIZE](#AAAAAAAAAM)  25
* #define [MD\_ENFORCE\_DWORD\_ALIGN](#AAAAAAAAAN)  1
* #define [MD\_SIGNATURE\_OPTION](#AAAAAAAAAO)  1
* #define [MD\_SIGNATURE\_FIELD\_LENGTH](#AAAAAAAAAP)  32
* #define [MD\_MISSION\_REV](#AAAAAAAAAQ)  0
* #define [MD\_PIPE\_NAME](#AAAAAAAAAR)  "MD\_CMD\_PIPE"
* #define [MD\_PIPE\_DEPTH](#AAAAAAAAAS)  50

### Macro Definition Documentation

#### #define MD\_DWELL\_TABLE\_SIZE  25

##### [Purpose](#cfsmdcfg__cfsmdcfg000003):

Maximum number of dwell specifications (address/delay/length) in a Dwell Table.

##### Limits

The maximum value for this parameter is limited by its effect on the [MD\_DwellTableLoad\_t](#AAAAAAAAAT) and associated data points, limiting it to 65535 points.

#### #define MD\_ENFORCE\_DWORD\_ALIGN  1

##### [Purpose](#cfsmdcfg__cfsmdcfg000004):

Option of whether 32 bit integers must be aligned to 32 bit boundaries. 1 indicates 'yes' (32-bit boundary alignment enforced), 0 indicates 'no' (16-bit boundary alignment enforced).

##### Limits

Value must be 0 or 1.

#### #define MD\_MISSION\_REV  0

##### [Purpose](#cfsmdcfg__cfsmdcfg000007):

Mission specific version number for MD application

##### Description:

An application version number consists of four parts: major version number, minor version number, revision number and mission specific revision number. The mission specific revision number is defined here and the other parts are defined in "md\_version.h".

##### Limits:

Must be defined as a numeric value that is greater than or equal to zero.

#### #define MD\_NUM\_DWELL\_TABLES  4

##### [Purpose](#cfsmdcfg__cfsmdcfg000002):

Number of memory dwell tables.

##### Limits

Acceptable values for this parameter are 1 to 16.

#### #define MD\_PIPE\_DEPTH  50

#### #define MD\_PIPE\_NAME  "MD\_CMD\_PIPE"

#### #define MD\_SIGNATURE\_FIELD\_LENGTH  32

##### [Purpose](#cfsmdcfg__cfsmdcfg000006):

Number of characters used in the Signature Field

##### Limits

Signature field length needs to be a multiple of 4 so that dwell packet is a multiple of 4 bytes and no compiler padding will occur. Note that the final character of the signature string must be a null character, so the effective length of user definable characters is one less than the defined length. The length specified must therefore be at least 4.

#### #define MD\_SIGNATURE\_OPTION  1

##### [Purpose](#cfsmdcfg__cfsmdcfg000005):

Option of whether a signature field will be reserved in dwell packets. 1 indicates 'yes', 0 indicates 'no'.

##### Limits

Value must be 0 or 1.

#### #define MD\_TBL\_FILENAME\_FORMAT  "/cf/md\_dw%02d.tbl"

##### [Purpose](#cfsmdcfg__cfsmdcfg000001):

Memory Dwell Base Filename

##### Description:

Default base name and location for Memory Dwell filenames AppInit will append 01,02,03, up to number of tables.

##### Limits:

This string shouldn't be longer than #OS\_MAX\_PATH\_LEN for the target platform in question

## fsw/src/md\_app.c File Reference

#include "cfe.h"

#include "md\_app.h"

#include "md\_events.h"

#include "md\_tbldefs.h"

#include "md\_verify.h"

#include "md\_cmds.h"

#include <string.h>

#include "md\_dwell\_tbl.h"

#include "md\_dwell\_pkt.h"

#include "md\_utils.h"

#include "md\_perfids.h"

#include "md\_version.h"

### Functions

* void [MD\_AppMain](#AAAAAAAAAV) (void)

*Entry Point and main loop for the Memory Dwell task.*

* int32 [MD\_AppInit](#AAAAAAAAAW) (void)

*Memory Dwell Application Initialization.*

* void [MD\_InitControlStructures](#AAAAAAAAAX) (void)

*Initialize local control structures.*

* int32 [MD\_InitSoftwareBusServices](#AAAAAAAAAY) (void)

*Initialize Software Bus Services for the Memory Dwell Task.*

* int32 [MD\_InitTableServices](#AAAAAAAAAZ) (void)

*Initialize Table Services for the Memory Dwell Task.*

* int32 [MD\_ManageDwellTable](#AAAAAAAABA) (uint8 TblIndex)

*Check Table Status and take appropriate actions.*

* void [MD\_ExecRequest](#AAAAAAAABB) (const CFE\_SB\_Buffer\_t \*BufPtr)

*Execute requested Memory Dwell commands.*

* void [MD\_HkStatus](#AAAAAAAABC) ()

*Send Housekeeping Status to Health & Safety task.*

* int16 [MD\_SearchCmdHndlrTbl](#AAAAAAAABD) (CFE\_SB\_MsgId\_t MessageID, CFE\_MSG\_FcnCode\_t CommandCode)

*Compares message with MD\_CmdHandlerTbl to identify the message.*

### Variables

* const [MD\_CmdHandlerTblRec\_t](#AAAAAAAABE) [MD\_CmdHandlerTbl](#AAAAAAAABF) []
* [MD\_AppData\_t](#AAAAAAAABG) [MD\_AppData](#AAAAAAAABH)

### Function Documentation

#### int32 MD\_AppInit (void )

Memory Dwell Application Initialization.

##### Description

Initialize all data elements. If this is a PROCESSOR\_RESET, and saving dwell tables to CDS is indicated, then restore image from CDS.

##### Assumptions, External Events, and Notes:

None

##### Returns:

#CFE\_SUCCESS

Any of the return values from #CFE\_ES\_RegisterApp

Any of the return values from #CFE\_EVS\_Register

Any of the return values from #CFE\_SB\_CreatePipe

Any of the return values from #CFE\_SB\_Subscribe

Any of the return values from #CFE\_TBL\_Register

Any of the return values from #CFE\_TBL\_Load

#### void MD\_AppMain (void )

Entry Point and main loop for the Memory Dwell task.

##### Description

Call MD\_AppInit to initialize the task. LOOP: Copy any newly loaded tables Pend on the Software Bus waiting to receive next message. If MD\_WAKEUP\_MID Message is received, call MD\_DwellLoop to send whatever memory values are being 'dwelled on'. If MD\_CMD\_MID Message is received, call MD\_ExecRequest for processing. If MD\_SEND\_HK\_MID Message is received, call MD\_HkStatus for processing.

##### Assumptions, External Events, and Notes:

Associated with each dwell address is a 'Delay' which is the number of wake-up calls to wait before recording the next value for this Dwell Table. The 'Rate' value associated with Dwell Table is the sum of all individual delays. For a table to be dwelled on, its rate must be >=1.

##### Return values:

|  |  |
| --- | --- |
| *None* |  |

#### void MD\_ExecRequest (const CFE\_SB\_Buffer\_t \* *BufPtr*)

Execute requested Memory Dwell commands.

##### Description

Processes messages obtained from the command pipe.

##### Assumptions, External Events, and Notes:

None

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *BufPtr* | a pointer to the message received from the command pipe |

##### Return values:

|  |  |
| --- | --- |
| *None* |  |

#### void MD\_HkStatus (void )

Send Housekeeping Status to Health & Safety task.

##### Description

For each dwell table the housekeeping data includes: number of dwell addresses, number of counts for packet formation, data size in bytes, current entry in data processing, current offset in packet data field, countdown to next data collection.

##### Assumptions, External Events, and Notes:

None

##### Return values:

|  |  |
| --- | --- |
| *None* |  |

#### void MD\_InitControlStructures (void )

Initialize local control structures.

##### Description

Initialize control structures for each of the [MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL) dwell streams.

##### Assumptions, External Events, and Notes:

None

##### Returns:

None

#### int32 MD\_InitSoftwareBusServices (void )

Initialize Software Bus Services for the Memory Dwell Task.

##### Description

Create message pipe. Subscribe to all input and output messages.

##### Assumptions, External Events, and Notes:

None

##### Returns:

#CFE\_SUCCESS

Any of the return values from #CFE\_SB\_CreatePipe

Any of the return values from #CFE\_SB\_Subscribe

#### int32 MD\_InitTableServices (void )

Initialize Table Services for the Memory Dwell Task.

##### Description

Register Tables with Table Services. Check for Recovered Tables. IF Recovered Tables Found THEN Retrieve them. Validate them. IF Recovered Tables are Invalid Initialize Tables ENDIF ELSE Initialize Tables.

##### Assumptions, External Events, and Notes:

None

##### Returns:

#CFE\_SUCCESS

Any of the return values from #CFE\_TBL\_Register

Any of the return values from #CFE\_TBL\_Load

#### int32 MD\_ManageDwellTable (uint8 *TblIndex*)

Check Table Status and take appropriate actions.

##### Description

Checks status of Tables, and takes action if validation requests or update requests are pending.

##### Assumptions, External Events, and Notes:

Assumes that an update is pending for the specified table.

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *TblIndex* | Dwell table identifier. Internal values [0..MD\_NUM\_DWELL\_TABLES-1] are used. |

##### Returns:

#CFE\_SUCCESS May return any value from CFE\_TBL\_GetStatus, CFE\_TBL\_Validate, or CFE\_TBL\_Update.

#### int16 MD\_SearchCmdHndlrTbl (CFE\_SB\_MsgId\_t *MessageID*, CFE\_MSG\_FcnCode\_t *CommandCode*)

Compares message with MD\_CmdHandlerTbl to identify the message.

##### Description

Searches the Command Handler Table for an entry matching the message ID and, if necessary, the Command Code. If an entry is not located, an error code is returned.

##### Assumptions, External Events, and Notes:

None

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *MessageID* | message ID of command message received on command pipe |
| in | *CommandCode* | command code from command message received on command pipe |

##### Returns:

On success, a non-negative Table Index is returned. [MD\_BAD\_CMD\_CODE](#AAAAAAAABI) [MD\_BAD\_MSG\_ID](#AAAAAAAABJ)

### Variable Documentation

#### [MD\_AppData\_t](#AAAAAAAABG) MD\_AppData

#### const [MD\_CmdHandlerTblRec\_t](#AAAAAAAABE) MD\_CmdHandlerTbl[]

**Initial value:**= {

{ [MD\_CMD\_MID](#AAAAAAAAAG), [MD\_NOOP\_CC](#AAAAAAAABK), sizeof([MD\_NoArgsCmd\_t](#AAAAAAAABL)), [MD\_CMD\_MSGTYPE](#AAAAAAAABM)},

{ [MD\_CMD\_MID](#AAAAAAAAAG), [MD\_RESET\_CNTRS\_CC](#AAAAAAAABN), sizeof([MD\_NoArgsCmd\_t](#AAAAAAAABL)), [MD\_CMD\_MSGTYPE](#AAAAAAAABM)},

{ [MD\_CMD\_MID](#AAAAAAAAAG), [MD\_START\_DWELL\_CC](#AAAAAAAABO), sizeof([MD\_CmdStartStop\_t](#AAAAAAAABP)),[MD\_CMD\_MSGTYPE](#AAAAAAAABM)},

{ [MD\_CMD\_MID](#AAAAAAAAAG), [MD\_STOP\_DWELL\_CC](#AAAAAAAABQ), sizeof([MD\_CmdStartStop\_t](#AAAAAAAABP)),[MD\_CMD\_MSGTYPE](#AAAAAAAABM)},

{ [MD\_CMD\_MID](#AAAAAAAAAG), [MD\_JAM\_DWELL\_CC](#AAAAAAAABR), sizeof([MD\_CmdJam\_t](#AAAAAAAABS)), [MD\_CMD\_MSGTYPE](#AAAAAAAABM)},

{ 0, 0, 0, [MD\_TERM\_MSGTYPE](#AAAAAAAABT)}

}

## fsw/src/md\_app.h File Reference

#include "common\_types.h"

#include "md\_platform\_cfg.h"

#include "cfe\_mission\_cfg.h"

#include "md\_msgids.h"

#include "cfs\_utils.h"

#include "md\_msg.h"

### Data Structures

* struct [MD\_DwellControlEntry\_t](#AAAAAAAABV)
* *MD structure for specifying individual memory dwell.* struct [MD\_DwellPacketControl\_t](#AAAAAAAABW)
* *MD structure for controlling dwell operations.* struct [MD\_AppData\_t](#AAAAAAAABG)
* *MD global data structure.* struct [MD\_CmdHandlerTblRec\_t](#AAAAAAAABE)

### Macros

* #define [MD\_DWELL\_TABLE\_BASENAME](#AAAAAAAABX)  "DWELL\_TABLE"
* #define [MD\_BAD\_CMD\_CODE](#AAAAAAAABI)  (-1)
* #define [MD\_BAD\_MSG\_ID](#AAAAAAAABJ)  (-2)
* Function Return Codes for Table Validation function and related routines#define [MD\_ERROR](#AAAAAAAABY)  (-1)

*Generic error value.*

* #define [MD\_TBL\_ENA\_FLAG\_ERROR](#AAAAAAAABZ)  (0xc0000001)

*Enable flag in table load is invalid (valid values are 0 and 1)*

* #define [MD\_ZERO\_RATE\_TBL\_ERROR](#AAAAAAAACA)  (0xc0000002)

*Table has zero value for total delay, and at least one dwell specified.*

* #define [MD\_RESOLVE\_ERROR](#AAAAAAAACB)  (0xc0000003)

*Symbolic address couldn't be resolved.*

* #define [MD\_INVALID\_ADDR\_ERROR](#AAAAAAAACC)  (0xc0000004)

*Invalid address found.*

* #define [MD\_INVALID\_LEN\_ERROR](#AAAAAAAACD)  (0xc0000005)

*Invalid dwell length found.*

* #define [MD\_NOT\_ALIGNED\_ERROR](#AAAAAAAACE)  (0xc0000006)

*Dwell address improperly aligned for specified dwell length.*

* #define [MD\_SIG\_LEN\_TBL\_ERROR](#AAAAAAAACF)  (0xc0000007)

*Signature not null terminated in table.*

### Enumerations

* enum [MD\_Dwell\_States](#AAAAAAAACG) { [MD\_DWELL\_STREAM\_DISABLED](#AAAAAAAACH), [MD\_DWELL\_STREAM\_ENABLED](#AAAAAAAACI) }*MD enum used for representing values for enable state.*
* enum [MD\_MsgType\_t](#AAAAAAAACJ) { [MD\_TERM\_MSGTYPE](#AAAAAAAABT) =0, [MD\_MSG\_MSGTYPE](#AAAAAAAACK), [MD\_CMD\_MSGTYPE](#AAAAAAAABM) }

### Functions

* void [MD\_AppMain](#AAAAAAAACL) (void)

*Entry Point and main loop for the Memory Dwell task.*

* int32 [MD\_AppInit](#AAAAAAAACM) (void)

*Memory Dwell Application Initialization.*

* void [MD\_InitControlStructures](#AAAAAAAACN) (void)

*Initialize local control structures.*

* int32 [MD\_InitSoftwareBusServices](#AAAAAAAACO) (void)

*Initialize Software Bus Services for the Memory Dwell Task.*

* int32 [MD\_InitTableServices](#AAAAAAAACP) (void)

*Initialize Table Services for the Memory Dwell Task.*

* int32 [MD\_ManageDwellTable](#AAAAAAAACQ) (uint8 TblIndex)

*Check Table Status and take appropriate actions.*

* void [MD\_ExecRequest](#AAAAAAAACR) (const CFE\_SB\_Buffer\_t \*BufPtr)

*Execute requested Memory Dwell commands.*

* void [MD\_HkStatus](#AAAAAAAACS) (void)

*Send Housekeeping Status to Health & Safety task.*

* int16 [MD\_SearchCmdHndlrTbl](#AAAAAAAACT) (CFE\_SB\_MsgId\_t MessageID, CFE\_MSG\_FcnCode\_t CommandCode)

*Compares message with MD\_CmdHandlerTbl to identify the message.*

### Variables

* [MD\_AppData\_t](#AAAAAAAABG) [MD\_AppData](#AAAAAAAACU)

### Macro Definition Documentation

#### #define MD\_BAD\_CMD\_CODE  (-1)

#### #define MD\_BAD\_MSG\_ID  (-2)

#### #define MD\_DWELL\_TABLE\_BASENAME  "DWELL\_TABLE"

#### #define MD\_ERROR  (-1)

Generic error value.

#### #define MD\_INVALID\_ADDR\_ERROR  (0xc0000004)

Invalid address found.

#### #define MD\_INVALID\_LEN\_ERROR  (0xc0000005)

Invalid dwell length found.

#### #define MD\_NOT\_ALIGNED\_ERROR  (0xc0000006)

Dwell address improperly aligned for specified dwell length.

#### #define MD\_RESOLVE\_ERROR  (0xc0000003)

Symbolic address couldn't be resolved.

#### #define MD\_SIG\_LEN\_TBL\_ERROR  (0xc0000007)

Signature not null terminated in table.

#### #define MD\_TBL\_ENA\_FLAG\_ERROR  (0xc0000001)

Enable flag in table load is invalid (valid values are 0 and 1)

#### #define MD\_ZERO\_RATE\_TBL\_ERROR  (0xc0000002)

Table has zero value for total delay, and at least one dwell specified.

### Enumeration Type Documentation

#### enum [MD\_Dwell\_States](#AAAAAAAACG)

MD enum used for representing values for enable state.

##### Enumerator:

|  |  |
| --- | --- |
| MD\_DWELL\_STREAM\_DISABLED |  |
| MD\_DWELL\_STREAM\_ENABLED |  |

#### enum [MD\_MsgType\_t](#AAAAAAAACJ)

##### Enumerator:

|  |  |
| --- | --- |
| MD\_TERM\_MSGTYPE | Command Handler Table Terminator Type. |
| MD\_MSG\_MSGTYPE | Message Type (requires Message ID match) |
| MD\_CMD\_MSGTYPE | Command Type (requires Message ID and Command Code match) |

### Function Documentation

#### int32 MD\_AppInit (void )

Memory Dwell Application Initialization.

##### Description

Initialize all data elements. If this is a PROCESSOR\_RESET, and saving dwell tables to CDS is indicated, then restore image from CDS.

##### Assumptions, External Events, and Notes:

None

##### Returns:

#CFE\_SUCCESS

Any of the return values from #CFE\_ES\_RegisterApp

Any of the return values from #CFE\_EVS\_Register

Any of the return values from #CFE\_SB\_CreatePipe

Any of the return values from #CFE\_SB\_Subscribe

Any of the return values from #CFE\_TBL\_Register

Any of the return values from #CFE\_TBL\_Load

#### void MD\_AppMain (void )

Entry Point and main loop for the Memory Dwell task.

##### Description

Call MD\_AppInit to initialize the task. LOOP: Copy any newly loaded tables Pend on the Software Bus waiting to receive next message. If MD\_WAKEUP\_MID Message is received, call MD\_DwellLoop to send whatever memory values are being 'dwelled on'. If MD\_CMD\_MID Message is received, call MD\_ExecRequest for processing. If MD\_SEND\_HK\_MID Message is received, call MD\_HkStatus for processing.

##### Assumptions, External Events, and Notes:

Associated with each dwell address is a 'Delay' which is the number of wake-up calls to wait before recording the next value for this Dwell Table. The 'Rate' value associated with Dwell Table is the sum of all individual delays. For a table to be dwelled on, its rate must be >=1.

##### Return values:

|  |  |
| --- | --- |
| *None* |  |

#### void MD\_ExecRequest (const CFE\_SB\_Buffer\_t \* *BufPtr*)

Execute requested Memory Dwell commands.

##### Description

Processes messages obtained from the command pipe.

##### Assumptions, External Events, and Notes:

None

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *BufPtr* | a pointer to the message received from the command pipe |

##### Return values:

|  |  |
| --- | --- |
| *None* |  |

#### void MD\_HkStatus (void )

Send Housekeeping Status to Health & Safety task.

##### Description

For each dwell table the housekeeping data includes: number of dwell addresses, number of counts for packet formation, data size in bytes, current entry in data processing, current offset in packet data field, countdown to next data collection.

##### Assumptions, External Events, and Notes:

None

##### Return values:

|  |  |
| --- | --- |
| *None* |  |

#### void MD\_InitControlStructures (void )

Initialize local control structures.

##### Description

Initialize control structures for each of the [MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL) dwell streams.

##### Assumptions, External Events, and Notes:

None

##### Returns:

None

#### int32 MD\_InitSoftwareBusServices (void )

Initialize Software Bus Services for the Memory Dwell Task.

##### Description

Create message pipe. Subscribe to all input and output messages.

##### Assumptions, External Events, and Notes:

None

##### Returns:

#CFE\_SUCCESS

Any of the return values from #CFE\_SB\_CreatePipe

Any of the return values from #CFE\_SB\_Subscribe

#### int32 MD\_InitTableServices (void )

Initialize Table Services for the Memory Dwell Task.

##### Description

Register Tables with Table Services. Check for Recovered Tables. IF Recovered Tables Found THEN Retrieve them. Validate them. IF Recovered Tables are Invalid Initialize Tables ENDIF ELSE Initialize Tables.

##### Assumptions, External Events, and Notes:

None

##### Returns:

#CFE\_SUCCESS

Any of the return values from #CFE\_TBL\_Register

Any of the return values from #CFE\_TBL\_Load

#### int32 MD\_ManageDwellTable (uint8 *TblIndex*)

Check Table Status and take appropriate actions.

##### Description

Checks status of Tables, and takes action if validation requests or update requests are pending.

##### Assumptions, External Events, and Notes:

Assumes that an update is pending for the specified table.

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *TblIndex* | Dwell table identifier. Internal values [0..MD\_NUM\_DWELL\_TABLES-1] are used. |

##### Returns:

#CFE\_SUCCESS May return any value from CFE\_TBL\_GetStatus, CFE\_TBL\_Validate, or CFE\_TBL\_Update.

#### int16 MD\_SearchCmdHndlrTbl (CFE\_SB\_MsgId\_t *MessageID*, CFE\_MSG\_FcnCode\_t *CommandCode*)

Compares message with MD\_CmdHandlerTbl to identify the message.

##### Description

Searches the Command Handler Table for an entry matching the message ID and, if necessary, the Command Code. If an entry is not located, an error code is returned.

##### Assumptions, External Events, and Notes:

None

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *MessageID* | message ID of command message received on command pipe |
| in | *CommandCode* | command code from command message received on command pipe |

##### Returns:

On success, a non-negative Table Index is returned. [MD\_BAD\_CMD\_CODE](#AAAAAAAABI) [MD\_BAD\_MSG\_ID](#AAAAAAAABJ)

### Variable Documentation

#### [MD\_AppData\_t](#AAAAAAAABG) MD\_AppData

#### 

## fsw/src/md\_cmds.c File Reference

#include "md\_cmds.h"

#include "md\_utils.h"

#include "md\_msg.h"

#include "md\_platform\_cfg.h"

#include <string.h>

#include "md\_app.h"

#include "md\_events.h"

#include "cfs\_utils.h"

#include "md\_dwell\_tbl.h"

### Functions

* void [MD\_ProcessStartCmd](#AAAAAAAACW) (const CFE\_SB\_Buffer\_t \*BufPtr)

*Process Memory Dwell Start Command.*

* void [MD\_ProcessStopCmd](#AAAAAAAACX) (const CFE\_SB\_Buffer\_t \*BufPtr)

*Process Memory Dwell Stop Command.*

* void [MD\_ProcessJamCmd](#AAAAAAAACY) (const CFE\_SB\_Buffer\_t \*BufPtr)

*Process Memory Dwell Jam Command.*

### Variables

* [MD\_AppData\_t](#AAAAAAAABG) [MD\_AppData](#AAAAAAAACZ)

### Function Documentation

#### void MD\_ProcessJamCmd (const CFE\_SB\_Buffer\_t \* *BufPtr*)

Process Memory Dwell Jam Command.

##### Description

Extract command arguments, take appropriate actions, issue event, and increment the command counter or error counter as appropriate.

##### Assumptions, External Events, and Notes:

Correct message length has been verified.

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *BufPtr* | a pointer to the message received from the command pipe |

##### Return values:

|  |  |
| --- | --- |
| *None* |  |

#### void MD\_ProcessStartCmd (const CFE\_SB\_Buffer\_t \* *BufPtr*)

Process Memory Dwell Start Command.

##### Description

Extract command arguments, take appropriate actions, issue event, and increment the command counter or error counter as appropriate.

##### Assumptions, External Events, and Notes:

Correct message length has been verified.

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *BufPtr* | a pointer to the message received from the command pipe |

##### Return values:

|  |  |
| --- | --- |
| *None* |  |

#### void MD\_ProcessStopCmd (const CFE\_SB\_Buffer\_t \* *BufPtr*)

Process Memory Dwell Stop Command.

##### Description

Extract command arguments, take appropriate actions, issue event, and increment the command counter or error counter as appropriate.

##### Assumptions, External Events, and Notes:

Correct message length has been verified.

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *BufPtr* | a pointer to the message received from the command pipe |

##### Return values:

|  |  |
| --- | --- |
| *None* |  |

### Variable Documentation

#### [MD\_AppData\_t](#AAAAAAAABG) MD\_AppData

#### 

## fsw/src/md\_cmds.h File Reference

#include "md\_msg.h"

### Functions

* void [MD\_ProcessStartCmd](#AAAAAAAADB) (const CFE\_SB\_Buffer\_t \*BufPtr)

*Process Memory Dwell Start Command.*

* void [MD\_ProcessStopCmd](#AAAAAAAADC) (const CFE\_SB\_Buffer\_t \*BufPtr)

*Process Memory Dwell Stop Command.*

* void [MD\_ProcessJamCmd](#AAAAAAAADD) (const CFE\_SB\_Buffer\_t \*BufPtr)

*Process Memory Dwell Jam Command.*

* void [MD\_ProcessSignatureCmd](#AAAAAAAADE) (const CFE\_SB\_Buffer\_t \*BufPtr)

*Process Set Signature Command.*

### Function Documentation

#### void MD\_ProcessJamCmd (const CFE\_SB\_Buffer\_t \* *BufPtr*)

Process Memory Dwell Jam Command.

##### Description

Extract command arguments, take appropriate actions, issue event, and increment the command counter or error counter as appropriate.

##### Assumptions, External Events, and Notes:

Correct message length has been verified.

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *BufPtr* | a pointer to the message received from the command pipe |

##### Return values:

|  |  |
| --- | --- |
| *None* |  |

#### void MD\_ProcessSignatureCmd (const CFE\_SB\_Buffer\_t \* *BufPtr*)

Process Set Signature Command.

##### Description

Extract command arguments, take appropriate actions, issue event, and increment the command counter or error counter as appropriate.

##### Assumptions, External Events, and Notes:

Correct message length has been verified.

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *BufPtr* | a pointer to the message received from the command pipe |

##### Return values:

|  |  |
| --- | --- |
| *None* |  |

#### void MD\_ProcessStartCmd (const CFE\_SB\_Buffer\_t \* *BufPtr*)

Process Memory Dwell Start Command.

##### Description

Extract command arguments, take appropriate actions, issue event, and increment the command counter or error counter as appropriate.

##### Assumptions, External Events, and Notes:

Correct message length has been verified.

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *BufPtr* | a pointer to the message received from the command pipe |

##### Return values:

|  |  |
| --- | --- |
| *None* |  |

#### void MD\_ProcessStopCmd (const CFE\_SB\_Buffer\_t \* *BufPtr*)

Process Memory Dwell Stop Command.

##### Description

Extract command arguments, take appropriate actions, issue event, and increment the command counter or error counter as appropriate.

##### Assumptions, External Events, and Notes:

Correct message length has been verified.

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *BufPtr* | a pointer to the message received from the command pipe |

##### Return values:

|  |  |
| --- | --- |
| *None* |  |

## fsw/src/md\_dwell\_pkt.c File Reference

#include "md\_dwell\_pkt.h"

#include "md\_utils.h"

#include "md\_app.h"

#include "md\_events.h"

#include "cfs\_utils.h"

#include <string.h>

### Functions

* void [MD\_DwellLoop](#AAAAAAAADG) (void)

*Process Dwell Packets.*

* int32 [MD\_GetDwellData](#AAAAAAAADH) (uint16 TblIndex, uint16 EntryIndex)

*Read a Single Dwell Address.*

* void [MD\_SendDwellPkt](#AAAAAAAADI) (uint16 TableIndex)

*Send Memory Dwell Packet.*

* void [MD\_StartDwellStream](#AAAAAAAADJ) (uint16 TableIndex)

*Start Dwell Stream.*

### Variables

* [MD\_AppData\_t](#AAAAAAAABG) [MD\_AppData](#AAAAAAAADK)

### Function Documentation

#### void MD\_DwellLoop (void )

Process Dwell Packets.

##### Description

Look at each table. If the table is enabled and its countdown counter times out, add dwell data to dwell packet until a packet delay is reached or the dwell packet is completed. If dwell packet is completed, send the packet and reset the counter.

##### Assumptions, External Events, and Notes:

None

##### Return values:

|  |  |
| --- | --- |
| *None* |  |

#### int32 MD\_GetDwellData (uint16 *TblIndex*, uint16 *EntryIndex*)

Read a Single Dwell Address.

##### Description

Copy memory value from a single dwell address to a dwell packet.

##### Assumptions, External Events, and Notes:

None

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *TblIndex* | identifies source dwell table (0..) |
| in | *EntryIndex* | identifies dwell entry within specified table (0..) |

##### Return values:

|  |  |
| --- | --- |
| *CFE\_SUCCESS* | if copy was performed successfully; non-zero otherwise. |

#### void MD\_SendDwellPkt (uint16 *TableIndex*)

Send Memory Dwell Packet.

##### Description

Send contents of memory pointed to by a table as a telemetry packet.

##### Assumptions, External Events, and Notes:

None

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *TableIndex* | Identifies source dwell table for the dwell packet. |

##### Return values:

|  |  |
| --- | --- |
| *None* |  |

#### void MD\_StartDwellStream (uint16 *TableIndex*)

Start Dwell Stream.

##### Description

Initialize parameters used to control generation of dwell packets.

##### Assumptions, External Events, and Notes:

None

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *TableIndex* | Identifies source dwell table for the dwell packet. |

##### Return values:

|  |  |
| --- | --- |
| *None* |  |

### Variable Documentation

#### [MD\_AppData\_t](#AAAAAAAABG) MD\_AppData

#### 

## fsw/src/md\_dwell\_pkt.h File Reference

#include "common\_types.h"

### Functions

* void [MD\_DwellLoop](#AAAAAAAADM) (void)

*Process Dwell Packets.*

* int32 [MD\_GetDwellData](#AAAAAAAADN) (uint16 TblIndex, uint16 EntryIndex)

*Read a Single Dwell Address.*

* void [MD\_SendDwellPkt](#AAAAAAAADO) (uint16 TableIndex)

*Send Memory Dwell Packet.*

* void [MD\_StartDwellStream](#AAAAAAAADP) (uint16 TableIndex)

*Start Dwell Stream.*

### Function Documentation

#### void MD\_DwellLoop (void )

Process Dwell Packets.

##### Description

Look at each table. If the table is enabled and its countdown counter times out, add dwell data to dwell packet until a packet delay is reached or the dwell packet is completed. If dwell packet is completed, send the packet and reset the counter.

##### Assumptions, External Events, and Notes:

None

##### Return values:

|  |  |
| --- | --- |
| *None* |  |

#### int32 MD\_GetDwellData (uint16 *TblIndex*, uint16 *EntryIndex*)

Read a Single Dwell Address.

##### Description

Copy memory value from a single dwell address to a dwell packet.

##### Assumptions, External Events, and Notes:

None

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *TblIndex* | identifies source dwell table (0..) |
| in | *EntryIndex* | identifies dwell entry within specified table (0..) |

##### Return values:

|  |  |
| --- | --- |
| *CFE\_SUCCESS* | if copy was performed successfully; non-zero otherwise. |

#### void MD\_SendDwellPkt (uint16 *TableIndex*)

Send Memory Dwell Packet.

##### Description

Send contents of memory pointed to by a table as a telemetry packet.

##### Assumptions, External Events, and Notes:

None

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *TableIndex* | Identifies source dwell table for the dwell packet. |

##### Return values:

|  |  |
| --- | --- |
| *None* |  |

#### void MD\_StartDwellStream (uint16 *TableIndex*)

Start Dwell Stream.

##### Description

Initialize parameters used to control generation of dwell packets.

##### Assumptions, External Events, and Notes:

None

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *TableIndex* | Identifies source dwell table for the dwell packet. |

##### Return values:

|  |  |
| --- | --- |
| *None* |  |

## fsw/src/md\_dwell\_tbl.c File Reference

#include "md\_dwell\_tbl.h"

#include "md\_utils.h"

#include "md\_app.h"

#include "cfs\_utils.h"

#include "md\_events.h"

#include <string.h>

#include "md\_tbldefs.h"

### Functions

* int32 [MD\_TableValidationFunc](#AAAAAAAADR) (void \*TblPtr)

*Dwell Table Validation Function.*

* int32 [MD\_ReadDwellTable](#AAAAAAAADS) (const [MD\_DwellTableLoad\_t](#AAAAAAAAAT) \*TblPtr, uint16 \*ActiveAddrCountPtr, uint16 \*SizePtr, uint32 \*RatePtr)

*Read Dwell Table to extract address count, byte size, and rate.*

* int32 [MD\_CheckTableEntries](#AAAAAAAADT) ([MD\_DwellTableLoad\_t](#AAAAAAAAAT) \*TblPtr, uint16 \*ErrorEntryArg)

*Validate dwell entries in specified Dwell Table.*

* int32 [MD\_ValidTableEntry](#AAAAAAAADU) ([MD\_TableLoadEntry\_t](#AAAAAAAADV) \*TblEntryPtr)

*Validate dwell table entry.*

* void [MD\_CopyUpdatedTbl](#AAAAAAAADW) ([MD\_DwellTableLoad\_t](#AAAAAAAAAT) \*MD\_LoadTablePtr, uint8 TblIndex)

*Generate internal data structures based on Dwell Table Load.*

* int32 [MD\_UpdateTableEnabledField](#AAAAAAAADX) (uint16 TableIndex, uint16 FieldValue)

*Update Dwell Table's Enabled Field.*

* int32 [MD\_UpdateTableDwellEntry](#AAAAAAAADY) (uint16 TableIndex, uint16 EntryIndex, uint16 NewLength, uint16 NewDelay, CFS\_SymAddr\_t NewDwellAddress)

*Update Values for a Dwell Table Entry.*

### Variables

* [MD\_AppData\_t](#AAAAAAAABG) [MD\_AppData](#AAAAAAAADZ)

### Function Documentation

#### int32 MD\_CheckTableEntries ([MD\_DwellTableLoad\_t](#AAAAAAAAAT) \* *TblPtr*, uint16 \* *ErrorEntryArg*)

Validate dwell entries in specified Dwell Table.

##### Description

Validate memory ranges for dwell address and field length, and validate field length size for entries in specified Dwell Table.

##### Assumptions, External Events, and Notes:

For table to be valid, each entry must be a null entry (specified by a zero field length) or the entry's address and length field must pass various checks.

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *TblPtr* | Table pointer |
| out | *\*ErrorEntryArg* | Entry number (0..) of first detected error, if any. |

##### Returns:

#CFE\_SUCCESS [MD\_RESOLVE\_ERROR](#AAAAAAAACB) Symbolic address couldn't be resolved. [MD\_INVALID\_ADDR\_ERROR](#AAAAAAAACC) Invalid address found. [MD\_INVALID\_LEN\_ERROR](#AAAAAAAACD) Invalid dwell length found. [MD\_NOT\_ALIGNED\_ERROR](#AAAAAAAACE) Dwell address improperly aligned for specified dwell length.

#### void MD\_CopyUpdatedTbl ([MD\_DwellTableLoad\_t](#AAAAAAAAAT) \* *MD\_LoadTablePtr*, uint8 *TblIndex*)

Generate internal data structures based on Dwell Table Load.

##### Description

Copies Enabled field. Copies Signature field. For each dwell table entry, copies field length, and delay value. Evaluates and saves resolved dwell address for each dwell entry. Evaluates and saves additional summary data based on entry contents.

##### Assumptions, External Events, and Notes:

Dwell table contents have been validated before reaching this point. Run when a table is loaded by command, or when a table is recovered on start up.

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *MD\_LoadTablePtr* | Pointer to Table Services buffer. |
| in | *TblIndex* | An identifier specifying which dwell table is to be copied. Internal values [0..MD\_NUM\_DWELL\_TABLES-1] are used. |

##### Return values:

|  |  |
| --- | --- |
| *None* |  |

#### int32 MD\_ReadDwellTable (const [MD\_DwellTableLoad\_t](#AAAAAAAAAT) \* *TblPtr*, uint16 \* *ActiveAddrCountPtr*, uint16 \* *SizePtr*, uint32 \* *RatePtr*)

Read Dwell Table to extract address count, byte size, and rate.

##### Description

Read active entries and count number of dwell addresses, number of bytes to be dwelled on, and number of wakeup calls between sending of dwell packets.

##### Assumptions, External Events, and Notes:

None

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *TblPtr* | Table pointer |
| out | *\*ActiveAddrCountPtr* | Number of addresses to be sampled. |
| out | *\*SizePtr* | Size, in bytes, of data to be sampled. |
| out | *\*RatePtr* | Rate, in number of wakeup calls, between sending of dwell packets. |

##### Return values:

|  |  |
| --- | --- |
| *CFE\_SUCCESS* |  |

#### int32 MD\_TableValidationFunc (void \* *TblPtr*)

Dwell Table Validation Function.

##### Description

This function indicates whether the Dwell Table corresponding to the input table pointer is valid.

##### Assumptions, External Events, and Notes:

This function gets registered with Table Services as a callback function for validating Dwell Tables that are loaded from the ground, and so must be in accordance with the call signature specified by Table Services. In addition, the function is used by Memory Dwell to validate Dwell Tables that have been recovered from a Critical Data Store.

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *TblPtr* | Table pointer |

##### Returns:

#CFE\_SUCCESS [MD\_TBL\_ENA\_FLAG\_ERROR](#AAAAAAAABZ) Enable flag in table load is invalid (valid values are 0 and 1) [MD\_ZERO\_RATE\_TBL\_ERROR](#AAAAAAAACA) Table has zero value for total delay, and at least one dwell specified. [MD\_RESOLVE\_ERROR](#AAAAAAAACB) Symbolic address couldn't be resolved. [MD\_INVALID\_ADDR\_ERROR](#AAAAAAAACC) Invalid address found. [MD\_INVALID\_LEN\_ERROR](#AAAAAAAACD) Invalid dwell length found. [MD\_NOT\_ALIGNED\_ERROR](#AAAAAAAACE) Dwell address improperly aligned for specified dwell length.

#### int32 MD\_UpdateTableDwellEntry (uint16 *TableIndex*, uint16 *EntryIndex*, uint16 *NewLength*, uint16 *NewDelay*, CFS\_SymAddr\_t *NewDwellAddress*)

Update Values for a Dwell Table Entry.

##### Description

Update Values for a Dwell Table Entry.

##### Assumptions, External Events, and Notes:

TableIndex is in [0..MD\_NUM\_DWELL\_TABLES-1] range. EntryIndex is in [0..MD\_DWELL\_TABLE\_SIZE-1] range. NewLength is 0, 1, 2, or 4. NewDwellAddress is a valid dwell address.

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *TableIndex* | An identifier specifying which dwell table is to be modified. Internal values [0..MD\_NUM\_DWELL\_TABLES-1] are used. |
| in | *EntryIndex* | An identifier specifying which entry is to be modified. Internal values [0..MD\_DWELL\_TABLE\_SIZE-1] are used. |
| in | *NewLength* | Number of bytes to be read. |
| in | *NewDelay* | Number of counts before next dwell. |
| in | *NewDwellAddress* | Memory address to be dwelled on. |

##### Returns:

#CFE\_SUCCESS #CFE\_TBL\_ERR\_BAD\_APP\_ID #CFE\_TBL\_ERR\_NO\_ACCESS #CFE\_TBL\_ERR\_INVALID\_HANDLE #CFE\_ES\_ERR\_APPNAME #CFE\_ES\_ERR\_BUFFER #CFE\_TBL\_ERR\_NEVER\_LOADED

#### int32 MD\_UpdateTableEnabledField (uint16 *TableIndex*, uint16 *FieldValue*)

Update Dwell Table's Enabled Field.

##### Description

Update Dwell Table's Enabled Field.

##### Assumptions, External Events, and Notes:

TableIndex is in [0..MD\_NUM\_DWELL\_TABLES-1] range. FieldValue is MD\_DWELL\_STREAM\_ENABLED or MD\_DWELL\_STREAM\_DISABLED.

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *TableIndex* | An identifier specifying which dwell table is to be modified. Internal values [0..MD\_NUM\_DWELL\_TABLES-1] are used. |
| in | *FieldValue* | New value for Enabled field. |

##### Returns:

#CFE\_SUCCESS #CFE\_TBL\_ERR\_BAD\_APP\_ID #CFE\_TBL\_ERR\_NO\_ACCESS #CFE\_TBL\_ERR\_INVALID\_HANDLE #CFE\_ES\_ERR\_APPNAME #CFE\_ES\_ERR\_BUFFER #CFE\_TBL\_ERR\_NEVER\_LOADED

#### int32 MD\_ValidTableEntry ([MD\_TableLoadEntry\_t](#AAAAAAAADV) \* *TblEntryPtr*)

Validate dwell table entry.

##### Description

Validates whether specified dwell table entry is valid.

##### Assumptions, External Events, and Notes:

To be valid, entry must either be a null entry (specified by a zero field length) or all of its address and length fields must pass various checks.

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *TblEntryPtr* | Entry pointer |

##### Returns:

#CFE\_SUCCESS [MD\_RESOLVE\_ERROR](#AAAAAAAACB) Symbolic address couldn't be resolved. [MD\_INVALID\_ADDR\_ERROR](#AAAAAAAACC) Invalid address found. [MD\_INVALID\_LEN\_ERROR](#AAAAAAAACD) Invalid dwell length found. [MD\_NOT\_ALIGNED\_ERROR](#AAAAAAAACE) Dwell address improperly aligned for specified dwell length.

### Variable Documentation

#### [MD\_AppData\_t](#AAAAAAAABG) MD\_AppData

#### 

## fsw/src/md\_dwell\_tbl.h File Reference

#include "cfe.h"

#include "md\_tbldefs.h"

### Functions

* int32 [MD\_TableValidationFunc](#AAAAAAAAEB) (void \*TblPtr)

*Dwell Table Validation Function.*

* void [MD\_CopyUpdatedTbl](#AAAAAAAAEC) ([MD\_DwellTableLoad\_t](#AAAAAAAAAT) \*MD\_LoadTablePtr, uint8 TblIndex)

*Generate internal data structures based on Dwell Table Load.*

* int32 [MD\_UpdateTableEnabledField](#AAAAAAAAED) (uint16 TableIndex, uint16 FieldValue)

*Update Dwell Table's Enabled Field.*

* int32 [MD\_UpdateTableDwellEntry](#AAAAAAAAEE) (uint16 TableIndex, uint16 EntryIndex, uint16 NewLength, uint16 NewDelay, CFS\_SymAddr\_t NewDwellAddress)

*Update Values for a Dwell Table Entry.*

* int32 [MD\_ReadDwellTable](#AAAAAAAAEF) (const [MD\_DwellTableLoad\_t](#AAAAAAAAAT) \*TblPtr, uint16 \*ActiveAddrCountPtr, uint16 \*SizePtr, uint32 \*RatePtr)

*Read Dwell Table to extract address count, byte size, and rate.*

* int32 [MD\_ValidTableEntry](#AAAAAAAAEG) ([MD\_TableLoadEntry\_t](#AAAAAAAADV) \*TblEntryPtr)

*Validate dwell table entry.*

* int32 [MD\_CheckTableEntries](#AAAAAAAAEH) ([MD\_DwellTableLoad\_t](#AAAAAAAAAT) \*TblPtr, uint16 \*ErrorEntryArg)

*Validate dwell entries in specified Dwell Table.*

### Function Documentation

#### int32 MD\_CheckTableEntries ([MD\_DwellTableLoad\_t](#AAAAAAAAAT) \* *TblPtr*, uint16 \* *ErrorEntryArg*)

Validate dwell entries in specified Dwell Table.

##### Description

Validate memory ranges for dwell address and field length, and validate field length size for entries in specified Dwell Table.

##### Assumptions, External Events, and Notes:

For table to be valid, each entry must be a null entry (specified by a zero field length) or the entry's address and length field must pass various checks.

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *TblPtr* | Table pointer |
| out | *\*ErrorEntryArg* | Entry number (0..) of first detected error, if any. |

##### Returns:

#CFE\_SUCCESS [MD\_RESOLVE\_ERROR](#AAAAAAAACB) Symbolic address couldn't be resolved. [MD\_INVALID\_ADDR\_ERROR](#AAAAAAAACC) Invalid address found. [MD\_INVALID\_LEN\_ERROR](#AAAAAAAACD) Invalid dwell length found. [MD\_NOT\_ALIGNED\_ERROR](#AAAAAAAACE) Dwell address improperly aligned for specified dwell length.

#### void MD\_CopyUpdatedTbl ([MD\_DwellTableLoad\_t](#AAAAAAAAAT) \* *MD\_LoadTablePtr*, uint8 *TblIndex*)

Generate internal data structures based on Dwell Table Load.

##### Description

Copies Enabled field. Copies Signature field. For each dwell table entry, copies field length, and delay value. Evaluates and saves resolved dwell address for each dwell entry. Evaluates and saves additional summary data based on entry contents.

##### Assumptions, External Events, and Notes:

Dwell table contents have been validated before reaching this point. Run when a table is loaded by command, or when a table is recovered on start up.

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *MD\_LoadTablePtr* | Pointer to Table Services buffer. |
| in | *TblIndex* | An identifier specifying which dwell table is to be copied. Internal values [0..MD\_NUM\_DWELL\_TABLES-1] are used. |

##### Return values:

|  |  |
| --- | --- |
| *None* |  |

#### int32 MD\_ReadDwellTable (const [MD\_DwellTableLoad\_t](#AAAAAAAAAT) \* *TblPtr*, uint16 \* *ActiveAddrCountPtr*, uint16 \* *SizePtr*, uint32 \* *RatePtr*)

Read Dwell Table to extract address count, byte size, and rate.

##### Description

Read active entries and count number of dwell addresses, number of bytes to be dwelled on, and number of wakeup calls between sending of dwell packets.

##### Assumptions, External Events, and Notes:

None

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *TblPtr* | Table pointer |
| out | *\*ActiveAddrCountPtr* | Number of addresses to be sampled. |
| out | *\*SizePtr* | Size, in bytes, of data to be sampled. |
| out | *\*RatePtr* | Rate, in number of wakeup calls, between sending of dwell packets. |

##### Return values:

|  |  |
| --- | --- |
| *CFE\_SUCCESS* |  |

#### int32 MD\_TableValidationFunc (void \* *TblPtr*)

Dwell Table Validation Function.

##### Description

This function indicates whether the Dwell Table corresponding to the input table pointer is valid.

##### Assumptions, External Events, and Notes:

This function gets registered with Table Services as a callback function for validating Dwell Tables that are loaded from the ground, and so must be in accordance with the call signature specified by Table Services. In addition, the function is used by Memory Dwell to validate Dwell Tables that have been recovered from a Critical Data Store.

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *TblPtr* | Table pointer |

##### Returns:

#CFE\_SUCCESS [MD\_TBL\_ENA\_FLAG\_ERROR](#AAAAAAAABZ) Enable flag in table load is invalid (valid values are 0 and 1) [MD\_ZERO\_RATE\_TBL\_ERROR](#AAAAAAAACA) Table has zero value for total delay, and at least one dwell specified. [MD\_RESOLVE\_ERROR](#AAAAAAAACB) Symbolic address couldn't be resolved. [MD\_INVALID\_ADDR\_ERROR](#AAAAAAAACC) Invalid address found. [MD\_INVALID\_LEN\_ERROR](#AAAAAAAACD) Invalid dwell length found. [MD\_NOT\_ALIGNED\_ERROR](#AAAAAAAACE) Dwell address improperly aligned for specified dwell length.

#### int32 MD\_UpdateTableDwellEntry (uint16 *TableIndex*, uint16 *EntryIndex*, uint16 *NewLength*, uint16 *NewDelay*, CFS\_SymAddr\_t *NewDwellAddress*)

Update Values for a Dwell Table Entry.

##### Description

Update Values for a Dwell Table Entry.

##### Assumptions, External Events, and Notes:

TableIndex is in [0..MD\_NUM\_DWELL\_TABLES-1] range. EntryIndex is in [0..MD\_DWELL\_TABLE\_SIZE-1] range. NewLength is 0, 1, 2, or 4. NewDwellAddress is a valid dwell address.

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *TableIndex* | An identifier specifying which dwell table is to be modified. Internal values [0..MD\_NUM\_DWELL\_TABLES-1] are used. |
| in | *EntryIndex* | An identifier specifying which entry is to be modified. Internal values [0..MD\_DWELL\_TABLE\_SIZE-1] are used. |
| in | *NewLength* | Number of bytes to be read. |
| in | *NewDelay* | Number of counts before next dwell. |
| in | *NewDwellAddress* | Memory address to be dwelled on. |

##### Returns:

#CFE\_SUCCESS #CFE\_TBL\_ERR\_BAD\_APP\_ID #CFE\_TBL\_ERR\_NO\_ACCESS #CFE\_TBL\_ERR\_INVALID\_HANDLE #CFE\_ES\_ERR\_APPNAME #CFE\_ES\_ERR\_BUFFER #CFE\_TBL\_ERR\_NEVER\_LOADED

#### int32 MD\_UpdateTableEnabledField (uint16 *TableIndex*, uint16 *FieldValue*)

Update Dwell Table's Enabled Field.

##### Description

Update Dwell Table's Enabled Field.

##### Assumptions, External Events, and Notes:

TableIndex is in [0..MD\_NUM\_DWELL\_TABLES-1] range. FieldValue is MD\_DWELL\_STREAM\_ENABLED or MD\_DWELL\_STREAM\_DISABLED.

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *TableIndex* | An identifier specifying which dwell table is to be modified. Internal values [0..MD\_NUM\_DWELL\_TABLES-1] are used. |
| in | *FieldValue* | New value for Enabled field. |

##### Returns:

#CFE\_SUCCESS #CFE\_TBL\_ERR\_BAD\_APP\_ID #CFE\_TBL\_ERR\_NO\_ACCESS #CFE\_TBL\_ERR\_INVALID\_HANDLE #CFE\_ES\_ERR\_APPNAME #CFE\_ES\_ERR\_BUFFER #CFE\_TBL\_ERR\_NEVER\_LOADED

#### int32 MD\_ValidTableEntry ([MD\_TableLoadEntry\_t](#AAAAAAAADV) \* *TblEntryPtr*)

Validate dwell table entry.

##### Description

Validates whether specified dwell table entry is valid.

##### Assumptions, External Events, and Notes:

To be valid, entry must either be a null entry (specified by a zero field length) or all of its address and length fields must pass various checks.

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *TblEntryPtr* | Entry pointer |

##### Returns:

#CFE\_SUCCESS [MD\_RESOLVE\_ERROR](#AAAAAAAACB) Symbolic address couldn't be resolved. [MD\_INVALID\_ADDR\_ERROR](#AAAAAAAACC) Invalid address found. [MD\_INVALID\_LEN\_ERROR](#AAAAAAAACD) Invalid dwell length found. [MD\_NOT\_ALIGNED\_ERROR](#AAAAAAAACE) Dwell address improperly aligned for specified dwell length.

## fsw/src/md\_events.h File Reference

### Macros

* #define [MD\_INIT\_INF\_EID](#AAAAAAAAEJ)  1

*'MD Initialized. Version d.d.d.d'*

* #define [MD\_PIPE\_ERR\_EID](#AAAAAAAAEK)  2

*'SB Pipe Read Error, App will exit. Pipe Return Status = 0x%08X'*

* #define [MD\_RECOVERED\_TBL\_VALID\_INF\_EID](#AAAAAAAAEL)  3

*'Recovered Dwell Table #d is valid and has been copied to the MD App'*

* #define [MD\_RECOVERED\_TBL\_NOT\_VALID\_ERR\_EID](#AAAAAAAAEM)  4

*'MD App will reinitialize Dwell Table #d because recovered table is not valid'*

* #define [MD\_DWELL\_TBL\_TOO\_LARGE\_CRIT\_EID](#AAAAAAAAEN)  5

*'Dwell Table(s) are too large to register: d > d bytes, d > d entries'*

* #define [MD\_TBL\_REGISTER\_CRIT\_EID](#AAAAAAAAEO)  6

*'CFE\_TBL\_Register error 0x%08X received for tbl#d'*

* #define [MD\_TBL\_INIT\_INF\_EID](#AAAAAAAAEP)  7

*'Dwell Tables Recovered: d, Dwell Tables Initialized: d'*

* #define [MD\_NOOP\_INF\_EID](#AAAAAAAAEQ)  10

*'No-op command, Version d.d.d.d'*

* #define [MD\_RESET\_CNTRS\_DBG\_EID](#AAAAAAAAER)  11

*'Reset Counters Cmd Received'*

* #define [MD\_START\_DWELL\_INF\_EID](#AAAAAAAAES)  12

*'Start Dwell Table command processed successfully for table mask 0x%04X'*

* #define [MD\_STOP\_DWELL\_INF\_EID](#AAAAAAAAET)  13

*'Stop Dwell Table command processed successfully for table mask 0x%04X'*

* #define [MD\_EMPTY\_TBLMASK\_ERR\_EID](#AAAAAAAAEU)  14

*'s command rejected because no tables were specified in table mask (0x%04X)'*

* #define [MD\_MID\_ERR\_EID](#AAAAAAAAEV)  15

*'Msg with Invalid message ID Rcvd – ID = 0x%08X'*

* #define [MD\_CC\_NOT\_IN\_TBL\_ERR\_EID](#AAAAAAAAEW)  16

*'Command Code d not found in MD\_CmdHandlerTbl structure'*

* #define [MD\_CC\_NOT\_IN\_LOOP\_ERR\_EID](#AAAAAAAAEX)  17

*'Command Code d not found in command processing loop'*

* #define [MD\_TBL\_STATUS\_ERR\_EID](#AAAAAAAAEY)  20

*'Received unexpected error 0x%08X from CFE\_TBL\_GetStatus for tbl #d'*

* #define [MD\_CMD\_LEN\_ERR\_EID](#AAAAAAAAEZ)  21

*'Cmd Msg with Bad length Rcvd: ID = 0x%08X, CC = d, Exp Len = d, Len = d'*

* #define [MD\_MSG\_LEN\_ERR\_EID](#AAAAAAAAFA)  22

*'Msg with Bad length Rcvd: ID = 0x%08X, Exp Len = d, Len = d'*

* #define [MD\_JAM\_DWELL\_INF\_EID](#AAAAAAAAFB)  30

*'Successful Jam to Dwell Tbl#d Entry #d'*

* #define [MD\_JAM\_NULL\_DWELL\_INF\_EID](#AAAAAAAAFC)  31

*'Successful Jam of a Null Dwell Entry to Dwell Tbl#d Entry #d'*

* #define [MD\_INVALID\_JAM\_TABLE\_ERR\_EID](#AAAAAAAAFD)  32

*'Jam Cmd rejected due to invalid Tbl Id arg = d (Expect 1.. d)'*

* #define [MD\_INVALID\_ENTRY\_ARG\_ERR\_EID](#AAAAAAAAFE)  33

*'Jam Cmd rejected due to invalid Entry Id arg = d (Expect 1.. d)'*

* #define [MD\_INVALID\_LEN\_ARG\_ERR\_EID](#AAAAAAAAFF)  34

*'Jam Cmd rejected due to invalid Field Length arg = d (Expect 0,1,2,or 4)'*

* #define [MD\_CANT\_RESOLVE\_JAM\_ADDR\_ERR\_EID](#AAAAAAAAFG)  35

*'Jam Cmd rejected because symbolic address 's' couldn't be resolved'*

* #define [MD\_INVALID\_JAM\_ADDR\_ERR\_EID](#AAAAAAAAFH)  36

*'Jam Cmd rejected because address 0x%08X is not in a valid range'*

* #define [MD\_JAM\_ADDR\_NOT\_32BIT\_ERR\_EID](#AAAAAAAAFI)  37

*'Jam Cmd rejected because address 0x%08X is not 32-bit aligned'*

* #define [MD\_JAM\_ADDR\_NOT\_16BIT\_ERR\_EID](#AAAAAAAAFJ)  38

*'Jam Cmd rejected because address 0x%08X is not 16-bit aligned'*

* #define [MD\_NO\_TBL\_COPY\_ERR\_EID](#AAAAAAAAFK)  39

*'Didn't update MD tbl #d due to unexpected CFE\_TBL\_GetAddress return: 0x%08X'*

* #define [MD\_ZERO\_RATE\_TBL\_INF\_EID](#AAAAAAAAFL)  40

*'Dwell Table is enabled but no processing will occur for table being loaded (rate is zero)'*

* #define [MD\_RESOLVE\_ERR\_EID](#AAAAAAAAFM)  41

*'Dwell Table rejected because address (sym='s'/offset=0x%08X) in entry #d couldn't be resolved'*

* #define [MD\_RANGE\_ERR\_EID](#AAAAAAAAFN)  42

*'Dwell Table rejected because address (sym='s'/offset=0x%08X) in entry #d was out of range'*

* #define [MD\_TBL\_HAS\_LEN\_ERR\_EID](#AAAAAAAAFO)  43

*'Dwell Table rejected because length (d) in entry #d was invalid'*

* #define [MD\_TBL\_ENA\_FLAG\_EID](#AAAAAAAAFP)  44

*'Dwell Table rejected because value of enable flag (d) is invalid'*

* #define [MD\_TBL\_ALIGN\_ERR\_EID](#AAAAAAAAFQ)  45

*'Dwell Table rejected because address (sym='s'/offset=0x08X) in entry #d is not properly aligned for a d-byte dwell'*

* #define [MD\_SET\_SIGNATURE\_INF\_EID](#AAAAAAAAFR)  46

*'Successfully set signature for Dwell Tbl#d to 's''*

* #define [MD\_INVALID\_SIGNATURE\_TABLE\_ERR\_EID](#AAAAAAAAFS)  47

*'Set Signature cmd rejected due to invalid Tbl Id arg = d (Expect 1.. d)'*

* #define [MD\_INVALID\_SIGNATURE\_LENGTH\_ERR\_EID](#AAAAAAAAFT)  49

*'Set Signature cmd rejected due to invalid Signature length'*

* #define [MD\_TBL\_SIG\_LEN\_ERR\_EID](#AAAAAAAAFU)  50

*'Dwell Table rejected because Signature length was invalid'*

* #define [MD\_ZERO\_RATE\_CMD\_INF\_EID](#AAAAAAAAFV)  51

*'Dwell Table d is enabled with a delay of zero so no processing will occur'*

* #define [MD\_DWELL\_TBL\_INF\_EID](#AAAAAAAAFW)  52

*'MD Dwell Tbl verify results: good = %d, bad = %d, unused = %d'*

* #define [MD\_UPDATE\_TBL\_EN\_ERR\_EID](#AAAAAAAAFX)  53

*'MD\_UpdateTableEnabledField, TableIndex %d: CFE\_TBL\_GetAddress Returned 0x%08x'*

* #define [MD\_UPDATE\_TBL\_DWELL\_ERR\_EID](#AAAAAAAAFY)  54

*'MD\_UpdateTableDwellEntry, TableIndex d: CFE\_TBL\_GetAddress Returned 0x%08x'*

* #define [MD\_UPDATE\_TBL\_SIG\_ERR\_EID](#AAAAAAAAFZ)  55

*'MD\_UpdateTableSignature, TableIndex d: CFE\_TBL\_GetAddress Returned 0x%08x'*

* #define [MD\_START\_DWELL\_ERR\_EID](#AAAAAAAAGA)  56

*'Start Dwell Table for mask 0x%04X failed for %d of %d tables'*

* #define [MD\_STOP\_DWELL\_ERR\_EID](#AAAAAAAAGB)  57

*'Stop Dwell Table for mask 0x%04X failed for %d of %d tables'*

* #define [MD\_SET\_SIGNATURE\_ERR\_EID](#AAAAAAAAGC)  58

*'Failed to set signature for Dwell Tbl#%d. Update returned 0x%08X'*

* #define [MD\_JAM\_DWELL\_ERR\_EID](#AAAAAAAAGD)  59

*'Failed Jam to Dwell Tbl#d Entry #d'*

* #define [MD\_JAM\_NULL\_DWELL\_ERR\_EID](#AAAAAAAAGE)  60

*'Failed Jam of a Null Dwell Entry to Dwell Tbl#d Entry #d'*

* #define [MD\_INIT\_TBL\_NAME\_ERR\_EID](#AAAAAAAAGF)  61

*'TableName could not be made. Err=0x%08X, Idx=%d"*

* #define [MD\_INIT\_TBL\_FILENAME\_ERR\_EID](#AAAAAAAAGG)  62

*'TblFileName could not be made. Err=0x%08X, Idx=%d"*

* #define [MD\_TBL\_VAL\_NULL\_PTR\_ERR\_EID](#AAAAAAAAGH)  55

*'Dwell Table rejected because of null table pointer'*

* #define [MD\_DWELL\_LOOP\_GET\_DWELL\_DATA\_ERR\_EID](#AAAAAAAAGI)  70

*'Dwell Table failed to read entry d in table d'*

* #define [MD\_CREATE\_PIPE\_ERR\_EID](#AAAAAAAAGJ)  71

*'Failed to create pipe. RC = d'*

* #define [MD\_SUB\_HK\_ERR\_EID](#AAAAAAAAGK)  72

*'Failed to subscribe to HK requests. RC = d'*

* #define [MD\_SUB\_CMD\_ERR\_EID](#AAAAAAAAGL)  73

*'Failed to subscribe to commands. RC = d'*

* #define [MD\_SUB\_WAKEUP\_ERR\_EID](#AAAAAAAAGM)  74

*'Failed to subscribe to wakeup messages. RC = d'*

### Macro Definition Documentation

#### #define MD\_CANT\_RESOLVE\_JAM\_ADDR\_ERR\_EID  35

'Jam Cmd rejected because symbolic address 's' couldn't be resolved'

##### [Event Message](#mdevents__mdevents000024):

'Jam Cmd rejected because symbolic address 's' couldn't be resolved'

##### Type: ERROR

##### Cause:

This error event is issued when symbolic address passed in Jam command couldn't be resolved by use of the on-board Symbol Table.

#### #define MD\_CC\_NOT\_IN\_LOOP\_ERR\_EID  17

'Command Code d not found in command processing loop'

##### [Event Message](#mdevents__mdevents000015):

'Command Code d not found in command processing loop'

##### Type: ERROR

##### Cause:

This event is issued when a command for the Memory Dwell task is received with a function code which is not included in the task's command code processing loop.

#### #define MD\_CC\_NOT\_IN\_TBL\_ERR\_EID  16

'Command Code d not found in MD\_CmdHandlerTbl structure'

##### [Event Message](#mdevents__mdevents000014):

'Command Code d not found in MD\_CmdHandlerTbl structure'

##### Type: ERROR

##### Cause:

This event is issued when a command for the Memory Dwell task is received with a function code which is not listed in the internal MD\_CmdHandlerTbl structure, which is used to associate an expected length for the command.

#### #define MD\_CMD\_LEN\_ERR\_EID  21

'Cmd Msg with Bad length Rcvd: ID = 0x%08X, CC = d, Exp Len = d, Len = d'

##### [Event Message](#mdevents__mdevents000017):

'Cmd Msg with Bad length Rcvd: ID = 0x%08X, CC = d, Exp Len = d, Len = d'

##### Type: ERROR

##### Cause:

This event is issued when the Memory Dwell task receives a command which has a length that is inconsistent with the expected length for its command code.

#### #define MD\_CREATE\_PIPE\_ERR\_EID  71

'Failed to create pipe. RC = d'

##### [Event Message](#mdevents__mdevents000053):

'Failed to create pipe. RC = d'

##### Type: ERROR

##### Cause:

This event message is issued when MD cannot create the software bus pipe. The RC field is the return code from #CFE\_SB\_CreatePipe.

#### #define MD\_DWELL\_LOOP\_GET\_DWELL\_DATA\_ERR\_EID  70

'Dwell Table failed to read entry d in table d'

##### [Event Message](#mdevents__mdevents000052):

'Dwell Table failed to read entry d in table d'

##### Type: ERROR

##### Cause:

This event message is issued when the PSP returns an error when attempting to read the memory address of an entry

#### #define MD\_DWELL\_TBL\_INF\_EID  52

'MD Dwell Tbl verify results: good = %d, bad = %d, unused = %d'

##### [Event Message](#mdevents__mdevents000040):

'MD Dwell Tbl verify results: good = %d, bad = %d, unused = %d'

##### Type: INFORMATION

##### Cause:

This event message is issued when a table validation has been completed for memory dwell table load

The good field is number of entries that passed, the bad field is number of entries that failed, the unused field is the number of entries that weren't checked because they were marked unused.

#### #define MD\_DWELL\_TBL\_TOO\_LARGE\_CRIT\_EID  5

'Dwell Table(s) are too large to register: d > d bytes, d > d entries'

##### [Event Message](#mdevents__mdevents000005):

'Dwell Table(s) are too large to register: d > d bytes, d > d entries'

##### Type: CRITICAL

##### Cause:

Issued when a #CFE\_TBL\_ERR\_INVALID\_SIZE error message is received from #CFE\_TBL\_Register call. Load structure can be reduced by reducing [MD\_DWELL\_TABLE\_SIZE](#AAAAAAAAAM), number of entries per Dwell Table.

#### #define MD\_EMPTY\_TBLMASK\_ERR\_EID  14

's command rejected because no tables were specified in table mask (0x%04X)'

##### [Event Message](#mdevents__mdevents000012):

's command rejected because no tables were specified in table mask (0x%04X)'

##### Type: ERROR

##### Cause:

None of the valid Table Ids (1..[MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL)) are contained in the table mask argument for the Start Dwell or Stop Dwell command.

#### #define MD\_INIT\_INF\_EID  1

'MD Initialized. Version d.d.d.d'

##### [Event Message](#mdevents__mdevents000001):

'MD Initialized. Version d.d.d.d'

##### Type: INFORMATION

##### Cause:

Issued upon successful completion of task initialization.

#### #define MD\_INIT\_TBL\_FILENAME\_ERR\_EID  62

'TblFileName could not be made. Err=0x%08X, Idx=%d"

##### [Event Message](#mdevents__mdevents000050):

'TblFileName could not be made. Err=0x%08X, Idx=%d"

##### Type: ERROR

##### Cause:

This event message is issued if the snprintf call used to create the TblFileName in [MD\_InitTableServices](#AAAAAAAACP) function fails.

The Err field in the event text is the value returned from snprintf. The Idx field in the event text is the TblIndex at which the error occurred.

#### #define MD\_INIT\_TBL\_NAME\_ERR\_EID  61

'TableName could not be made. Err=0x%08X, Idx=%d"

##### [Event Message](#mdevents__mdevents000049):

'TableName could not be made. Err=0x%08X, Idx=%d"

##### Type: ERROR

##### Cause:

This event message is issued if the snprintf call used to create the TableName in [MD\_InitTableServices](#AAAAAAAACP) function fails.

The Err field in the event text is the value returned from snprintf. The Idx field in the event text is the TblIndex at which the error occurred.

#### #define MD\_INVALID\_ENTRY\_ARG\_ERR\_EID  33

'Jam Cmd rejected due to invalid Entry Id arg = d (Expect 1.. d)'

##### [Event Message](#mdevents__mdevents000022):

'Jam Cmd rejected due to invalid Entry Id arg = d (Expect 1.. d)'

##### Type: ERROR

##### Cause:

This error event is issued when a Jam Dwell Command is received with an invalid value for the entry id argument. Values in the range 1..[MD\_DWELL\_TABLE\_SIZE](#AAAAAAAAAM) are expected.

#### #define MD\_INVALID\_JAM\_ADDR\_ERR\_EID  36

'Jam Cmd rejected because address 0x%08X is not in a valid range'

##### [Event Message](#mdevents__mdevents000025):

'Jam Cmd rejected because address 0x%08X is not in a valid range'

##### Type: ERROR

##### Cause:

The resolved address (numerical value for symbol + offset) and field length specified by the Jam command were found to specify a dwell be outside valid ranges.

#### #define MD\_INVALID\_JAM\_TABLE\_ERR\_EID  32

'Jam Cmd rejected due to invalid Tbl Id arg = d (Expect 1.. d)'

##### [Event Message](#mdevents__mdevents000021):

'Jam Cmd rejected due to invalid Tbl Id arg = d (Expect 1.. d)'

##### Type: ERROR

##### Cause:

This error event is issued when a Jam Dwell Command is received with an invalid value for the table id argument. Values in the range 1..[MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL) are expected.

#### #define MD\_INVALID\_LEN\_ARG\_ERR\_EID  34

'Jam Cmd rejected due to invalid Field Length arg = d (Expect 0,1,2,or 4)'

##### [Event Message](#mdevents__mdevents000023):

'Jam Cmd rejected due to invalid Field Length arg = d (Expect 0,1,2,or 4)'

##### Type: ERROR

##### Cause:

This error event is issued when a Jam Dwell Command is received with an invalid value for the field length argument.

#### #define MD\_INVALID\_SIGNATURE\_LENGTH\_ERR\_EID  49

'Set Signature cmd rejected due to invalid Signature length'

##### [Event Message](#mdevents__mdevents000037):

'Set Signature cmd rejected due to invalid Signature length'

##### Type: ERROR

##### Cause:

This error event is issued when a Set Signature Command is received with a string not containing null termination within the allowable length.

#### #define MD\_INVALID\_SIGNATURE\_TABLE\_ERR\_EID  47

'Set Signature cmd rejected due to invalid Tbl Id arg = d (Expect 1.. d)'

##### [Event Message](#mdevents__mdevents000036):

'Set Signature cmd rejected due to invalid Tbl Id arg = d (Expect 1.. d)'

##### Type: ERROR

##### Cause:

This error event is issued when a Set Signature Command is received with an invalid value for the table id argument. Values in the range 1..[MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL) are expected.

#### #define MD\_JAM\_ADDR\_NOT\_16BIT\_ERR\_EID  38

'Jam Cmd rejected because address 0x%08X is not 16-bit aligned'

##### [Event Message](#mdevents__mdevents000027):

'Jam Cmd rejected because address 0x%08X is not 16-bit aligned'

##### Type: ERROR

##### Cause:

The Jam command specified a 2-byte read, and the resolved address (numerical value for symbol + offset) is not on a 2-byte boundary.

#### #define MD\_JAM\_ADDR\_NOT\_32BIT\_ERR\_EID  37

'Jam Cmd rejected because address 0x%08X is not 32-bit aligned'

##### [Event Message](#mdevents__mdevents000026):

'Jam Cmd rejected because address 0x%08X is not 32-bit aligned'

##### Type: ERROR

##### Cause:

The Jam command specified a 4-byte read, and the resolved address (numerical value for symbol + offset) is not on a 4-byte boundary.

#### #define MD\_JAM\_DWELL\_ERR\_EID  59

'Failed Jam to Dwell Tbl#d Entry #d'

##### [Event Message](#mdevents__mdevents000047):

'Failed Jam to Dwell Tbl#d Entry #d'

##### Type: ERROR

##### Cause:

This event is issued for a failed jam operation.

The Tbl field is the index of the table for which the failure occurred. The Entry field is the index of the entry for which the failure occurred.

#### #define MD\_JAM\_DWELL\_INF\_EID  30

'Successful Jam to Dwell Tbl#d Entry #d'

##### [Event Message](#mdevents__mdevents000019):

'Successful Jam to Dwell Tbl#d Entry #d'

##### Type: INFORMATION

##### Cause:

This event is issued for a successful jam operation.

#### #define MD\_JAM\_NULL\_DWELL\_ERR\_EID  60

'Failed Jam of a Null Dwell Entry to Dwell Tbl#d Entry #d'

##### [Event Message](#mdevents__mdevents000048):

'Failed Jam of a Null Dwell Entry to Dwell Tbl#d Entry #d'

##### Type: ERROR

##### Cause:

This event is issued for a failed jam operation in which a null dwell entry is specified. A null entry is specified when the input dwell length is zero. All dwell fields (address, length, and delay) will be set to zero in this case.

The Tbl field is the index of the table for which the failure occurred. The Entry field is the index of the entry for which the failure occurred.

#### #define MD\_JAM\_NULL\_DWELL\_INF\_EID  31

'Successful Jam of a Null Dwell Entry to Dwell Tbl#d Entry #d'

##### [Event Message](#mdevents__mdevents000020):

'Successful Jam of a Null Dwell Entry to Dwell Tbl#d Entry #d'

##### Type: INFORMATION

##### Cause:

This event is issued for a jam operation in which a null dwell entry is specified. A null entry is specified when the input dwell length is zero. All dwell fields (address, length, and delay) will be set to zero in this case.

#### #define MD\_MID\_ERR\_EID  15

'Msg with Invalid message ID Rcvd – ID = 0x%08X'

##### [Event Message](#mdevents__mdevents000013):

'Msg with Invalid message ID Rcvd – ID = 0x%08X'

##### Type: ERROR

##### Cause:

This event is issued if the Memory Dwell task receives a message with an unrecognized Message ID.

#### #define MD\_MSG\_LEN\_ERR\_EID  22

'Msg with Bad length Rcvd: ID = 0x%08X, Exp Len = d, Len = d'

##### [Event Message](#mdevents__mdevents000018):

'Msg with Bad length Rcvd: ID = 0x%08X, Exp Len = d, Len = d'

##### Type: ERROR

##### Cause:

This event is issued when the Memory Dwell task receives a message which has a length that is inconsistent with the expected length for its message id.

#### #define MD\_NO\_TBL\_COPY\_ERR\_EID  39

'Didn't update MD tbl #d due to unexpected CFE\_TBL\_GetAddress return: 0x%08X'

##### [Event Message](#mdevents__mdevents000028):

'Didn't update MD tbl #d due to unexpected CFE\_TBL\_GetAddress return: 0x%08X'

##### Type: ERROR

##### Cause:

This event is issued after the following sequence occurs:

1) #CFE\_TBL\_GetStatus returned #CFE\_TBL\_INFO\_UPDATE\_PENDING

2) #CFE\_TBL\_Update returned #CFE\_SUCCESS, a call is made to

3) #CFE\_TBL\_GetAddress returned something other than CFE\_TBL\_INFO\_UPDATED.

When this happens, the newly loaded table contents are *not* copied to MD task structures.

#### #define MD\_NOOP\_INF\_EID  10

'No-op command, Version d.d.d.d'

##### [Event Message](#mdevents__mdevents000008):

'No-op command, Version d.d.d.d'

##### Type: INFORMATION

##### Cause:

Issued upon receipt of a Memory Dwell no-op command.

#### #define MD\_PIPE\_ERR\_EID  2

'SB Pipe Read Error, App will exit. Pipe Return Status = 0x%08X'

##### [Event Message](#mdevents__mdevents000002):

'SB Pipe Read Error, App will exit. Pipe Return Status = 0x%08X'

##### Type: ERROR

##### Cause:

This event is issued following error return from #CFE\_SB\_RcvMsg call.

#### #define MD\_RANGE\_ERR\_EID  42

'Dwell Table rejected because address (sym='s'/offset=0x%08X) in entry #d was out of range'

##### [Event Message](#mdevents__mdevents000031):

'Dwell Table rejected because address (sym='s'/offset=0x%08X) in entry #d was out of range'

##### Type: ERROR

##### Cause:

The specified address was not in allowable memory ranges.

If this load was initiated via ground command, the Table Id will be known from the command. If this load was recovered after a reset, a subsequent event message will identify the Table Id.

#### #define MD\_RECOVERED\_TBL\_NOT\_VALID\_ERR\_EID  4

'MD App will reinitialize Dwell Table #d because recovered table is not valid'

##### [Event Message](#mdevents__mdevents000004):

'MD App will reinitialize Dwell Table #d because recovered table is not valid'

##### Type: ERROR

##### Cause:

Issued when a Dwell Table is recovered and found to be invalid.

#### #define MD\_RECOVERED\_TBL\_VALID\_INF\_EID  3

'Recovered Dwell Table #d is valid and has been copied to the MD App'

##### [Event Message](#mdevents__mdevents000003):

'Recovered Dwell Table #d is valid and has been copied to the MD App'

##### Type: INFORMATION

##### Cause:

Issued upon successful recovery of a Dwell Table.

#### #define MD\_RESET\_CNTRS\_DBG\_EID  11

'Reset Counters Cmd Received'

##### [Event Message](#mdevents__mdevents000009):

'Reset Counters Cmd Received'

##### Type: DEBUG

##### Cause:

Issued upon receipt of a Memory Dwell Reset Counters command.

#### #define MD\_RESOLVE\_ERR\_EID  41

'Dwell Table rejected because address (sym='s'/offset=0x%08X) in entry #d couldn't be resolved'

##### [Event Message](#mdevents__mdevents000030):

'Dwell Table rejected because address (sym='s'/offset=0x%08X) in entry #d couldn't be resolved'

##### Type: ERROR

##### Cause:

The specified symbol wasn't found in the system symbol table. This could be either because there is no symbol table, or because the symbol isn't present in an existing symbol table.

If this load was initiated via ground command, the Table Id will be known from the command. If this load was recovered after a reset, a subsequent event message will identify the Table Id.

#### #define MD\_SET\_SIGNATURE\_ERR\_EID  58

'Failed to set signature for Dwell Tbl#%d. Update returned 0x%08X'

##### [Event Message](#mdevents__mdevents000046):

'Failed to set signature for Dwell Tbl#%d. Update returned 0x%08X'

##### Type: ERROR

##### Cause:

This event message is issued when the table address cannot be acquired for the table specified in the [MD\_ProcessSignatureCmd](#AAAAAAAADE).

The Tbl field is the index of the dwell table for which the error occurred. The returned field is the value returned from #MD\_UpdateTableSignature.

#### #define MD\_SET\_SIGNATURE\_INF\_EID  46

'Successfully set signature for Dwell Tbl#d to 's''

##### [Event Message](#mdevents__mdevents000035):

'Successfully set signature for Dwell Tbl#d to 's''

##### Type: INFORMATION

##### Cause:

A 'Set Signature' command was received and processed nominally to associate a signature with the specified dwell table. All dwell packets derived from that dwell table will include the specified signature string.

#### #define MD\_START\_DWELL\_ERR\_EID  56

'Start Dwell Table for mask 0x%04X failed for %d of %d tables'

##### [Event Message](#mdevents__mdevents000044):

'Start Dwell Table for mask 0x%04X failed for %d of %d tables'

##### Type: ERROR

##### Cause:

This event message is issued when the table address cannot be acquired for one of the tables being started with the [MD\_ProcessStartCmd](#AAAAAAAADB). In addition to this summary message, a [MD\_UPDATE\_TBL\_EN\_ERR\_EID](#AAAAAAAAFX) event is issed for each failure.

The mask field is the mask of tables to start specified in the command. The integer fields "\%d or \%d" states the total number of tables that could not be started.

#### #define MD\_START\_DWELL\_INF\_EID  12

'Start Dwell Table command processed successfully for table mask 0x%04X'

##### [Event Message](#mdevents__mdevents000010):

'Start Dwell Table command processed successfully for table mask 0x%04X'

##### Type: INFORMATION

##### Cause:

Issued upon receipt of a Memory Dwell Start command. Upon receipt of this command, the specified tables are started for processing.

#### #define MD\_STOP\_DWELL\_ERR\_EID  57

'Stop Dwell Table for mask 0x%04X failed for %d of %d tables'

##### [Event Message](#mdevents__mdevents000045):

'Stop Dwell Table for mask 0x%04X failed for %d of %d tables'

##### Type: ERROR

##### Cause:

This event message is issued when the table address cannot be acquired for one of the tables being started with the [MD\_ProcessStopCmd](#AAAAAAAADC). In addition to this summary message, a [MD\_UPDATE\_TBL\_EN\_ERR\_EID](#AAAAAAAAFX) event is issed for each failure.

The mask field is the mask of tables to start specified in the command. The integer fields "\%d or \%d" states the total number of tables that could not be started.

#### #define MD\_STOP\_DWELL\_INF\_EID  13

'Stop Dwell Table command processed successfully for table mask 0x%04X'

##### [Event Message](#mdevents__mdevents000011):

'Stop Dwell Table command processed successfully for table mask 0x%04X'

##### Type: INFORMATION

##### Cause:

Issued upon receipt of a Memory Dwell Start command. Upon receipt of this command, the specified tables are stopped.

#### #define MD\_SUB\_CMD\_ERR\_EID  73

'Failed to subscribe to commands. RC = d'

##### [Event Message](#mdevents__mdevents000055):

'Failed to subscribe to commands. RC = d'

##### Type: ERROR

##### Cause:

This event message is issued when MD cannot subscribe to commands. The RC field is the return code from #CFE\_SB\_Subscribe

#### #define MD\_SUB\_HK\_ERR\_EID  72

'Failed to subscribe to HK requests. RC = d'

##### [Event Message](#mdevents__mdevents000054):

'Failed to subscribe to HK requests. RC = d'

##### Type: ERROR

##### Cause:

This event message is issued when MD cannot subscribe to housekeeping requests. The RC field is the return code from #CFE\_SB\_Subscribe

#### #define MD\_SUB\_WAKEUP\_ERR\_EID  74

'Failed to subscribe to wakeup messages. RC = d'

##### [Event Message](#mdevents__mdevents000056):

'Failed to subscribe to wakeup messages. RC = d'

##### Type: ERROR

##### Cause:

This event message is issued when MD cannot subscribe to wakeup messages. The RC field is the return code from #CFE\_SB\_Subscribe

#### #define MD\_TBL\_ALIGN\_ERR\_EID  45

'Dwell Table rejected because address (sym='s'/offset=0x08X) in entry #d is not properly aligned for a d-byte dwell'

##### [Event Message](#mdevents__mdevents000034):

'Dwell Table rejected because address (sym='s'/offset=0x08X) in entry #d is not properly aligned for a d-byte dwell'

##### Type: ERROR

##### Cause:

Either a 4-byte dwell was specified and address is not 4-byte aligned, or a 2-byte dwell was specified and address is not 2-byte aligned.

If this load was initiated via ground command, the Table Id will be known from the command. If this load was recovered after a reset, a subsequent event message will identify the Table Id.

#### #define MD\_TBL\_ENA\_FLAG\_EID  44

'Dwell Table rejected because value of enable flag (d) is invalid'

##### [Event Message](#mdevents__mdevents000033):

'Dwell Table rejected because value of enable flag (d) is invalid'

##### Type: ERROR

##### Cause:

The dwell table's enable value was neither zero nor one.

If this load was initiated via ground command, the Table Id will be known from the command. If this load was recovered after a reset, a subsequent event message will identify the Table Id.

#### #define MD\_TBL\_HAS\_LEN\_ERR\_EID  43

'Dwell Table rejected because length (d) in entry #d was invalid'

##### [Event Message](#mdevents__mdevents000032):

'Dwell Table rejected because length (d) in entry #d was invalid'

##### Type: ERROR

##### Cause:

The dwell table contains an invalid value for a dwell length.

If this load was initiated via ground command, the Table Id will be known from the command. If this load was recovered after a reset, a subsequent event message will identify the Table Id.

#### #define MD\_TBL\_INIT\_INF\_EID  7

'Dwell Tables Recovered: d, Dwell Tables Initialized: d'

##### [Event Message](#mdevents__mdevents000007):

'Dwell Tables Recovered: d, Dwell Tables Initialized: d'

##### Type: INFORMATION

##### Cause:

Issued at the end of Table Initialization, specifying how many tables were recovered and how many initialized.

#### #define MD\_TBL\_REGISTER\_CRIT\_EID  6

'CFE\_TBL\_Register error 0x%08X received for tbl#d'

##### [Event Message](#mdevents__mdevents000006):

'CFE\_TBL\_Register error 0x%08X received for tbl#d'

##### Type: CRITICAL

##### Cause:

Issued when an error message, other than #CFE\_TBL\_ERR\_INVALID\_SIZE, is received from #CFE\_TBL\_Register call.

#### #define MD\_TBL\_SIG\_LEN\_ERR\_EID  50

'Dwell Table rejected because Signature length was invalid'

##### [Event Message](#mdevents__mdevents000038):

'Dwell Table rejected because Signature length was invalid'

##### Type: ERROR

##### Cause:

The dwell table contains an invalid Signature (not null terminated).

If this load was initiated via ground command, the Table Id will be known from the command. If this load was recovered after a reset, a subsequent event message will identify the Table Id.

#### #define MD\_TBL\_STATUS\_ERR\_EID  20

'Received unexpected error 0x%08X from CFE\_TBL\_GetStatus for tbl #d'

##### [Event Message](#mdevents__mdevents000016):

'Received unexpected error 0x%08X from CFE\_TBL\_GetStatus for tbl #d'

##### Type: ERROR

##### Cause:

This event is issued on receipt of an unexpected error message from CFE\_TBL\_GetStatus. Normal processing continues; no special action is taken.

#### #define MD\_TBL\_VAL\_NULL\_PTR\_ERR\_EID  55

'Dwell Table rejected because of null table pointer'

##### [Event Message](#mdevents__mdevents000051):

'Dwell Table rejected because of null table pointer'

##### Type: ERROR

##### Cause:

This event message is issued when the table pointer passed to [MD\_TableValidationFunc](#AAAAAAAAEB) is null.

#### #define MD\_UPDATE\_TBL\_DWELL\_ERR\_EID  54

'MD\_UpdateTableDwellEntry, TableIndex d: CFE\_TBL\_GetAddress Returned 0x%08x'

##### [Event Message](#mdevents__mdevents000042):

'MD\_UpdateTableDwellEntry, TableIndex d: CFE\_TBL\_GetAddress Returned 0x%08x'

##### Type: ERROR

##### Cause:

This event message is issued when the table address cannot be acquired in the [MD\_UpdateTableDwellEntry](#AAAAAAAAEE) function. This event is issued when the #CFE\_TBL\_GetAddress function returns a value other than CFE\_SUCCESS or CFE\_TBL\_INFO\_UPDATED.

The TableIndex field is the index of the dwell table for which the failure occurred The Returned field is the value returned from #CFE\_Tbl\_GetAddress

#### #define MD\_UPDATE\_TBL\_EN\_ERR\_EID  53

'MD\_UpdateTableEnabledField, TableIndex %d: CFE\_TBL\_GetAddress Returned 0x%08x'

##### [Event Message](#mdevents__mdevents000041):

'MD\_UpdateTableEnabledField, TableIndex %d: CFE\_TBL\_GetAddress Returned 0x%08x'

##### Type: ERROR

##### Cause:

This event message is issued when the table address cannot be acquired in the [MD\_UpdateTableEnabledField](#AAAAAAAAED) function. This event is issued when the #CFE\_TBL\_GetAddress function returns a value other than CFE\_SUCCESS or CFE\_TBL\_INFO\_UPDATED.

The TableIndex field is the index of the dwell table for which the failure occurred The Returned field is the value returned from #CFE\_Tbl\_GetAddress

#### #define MD\_UPDATE\_TBL\_SIG\_ERR\_EID  55

'MD\_UpdateTableSignature, TableIndex d: CFE\_TBL\_GetAddress Returned 0x%08x'

##### [Event Message](#mdevents__mdevents000043):

'MD\_UpdateTableSignature, TableIndex d: CFE\_TBL\_GetAddress Returned 0x%08x'

##### Type: ERROR

##### Cause:

This event message is issued when the table address cannot be acquired in the #MD\_UpdateTableSignature function. This event is issued when the #CFE\_TBL\_GetAddress function returns a value other than CFE\_SUCCESS or CFE\_TBL\_INFO\_UPDATED.

The TableIndex field is the index of the dwell table for which the failure occurred The Returned field is the value returned from #CFE\_Tbl\_GetAddress

#### #define MD\_ZERO\_RATE\_CMD\_INF\_EID  51

'Dwell Table d is enabled with a delay of zero so no processing will occur'

##### [Event Message](#mdevents__mdevents000039):

'Dwell Table d is enabled with a delay of zero so no processing will occur'

##### Type: INFORMATIONAL

##### Cause:

1) The calculated rate, the total of delays for all active entries, current equals zero

2) The table is currently enabled

If the command either changes the delay values in the table (such that the total delay is 0) while the table is enabled, or if the table is enabled while the total delay value is 0, this event will be sent.

#### #define MD\_ZERO\_RATE\_TBL\_INF\_EID  40

'Dwell Table is enabled but no processing will occur for table being loaded (rate is zero)'

##### [Event Message](#mdevents__mdevents000029):

'Dwell Table is enabled but no processing will occur for table being loaded (rate is zero)'

##### Type: INFORMATIONAL

##### Cause:

1) The calculated rate, the total of delays for all active entries, equals zero, and 2) The table is enabled

If this load was initiated via ground command, the Table Id will be known from the command. If this load was recovered after a reset, a subsequent event message will identify the Table Id.

## fsw/src/md\_msg.h File Reference

include "md\_platform\_cfg.h"

#include "cfe.h"

#include "cfs\_utils.h"

#include "md\_msgdefs.h"

### Data Structures

* struct [MD\_NoArgsCmd\_t](#AAAAAAAABL)
* *Generic "no arguments" command.* struct [MD\_CmdStartStop\_t](#AAAAAAAABP)
* *Start and Stop Dwell Commands.* struct [MD\_CmdJam\_t](#AAAAAAAABS)
* *Jam Dwell Command.* struct [MD\_HkTlm\_t](#AAAAAAAAGO)
* struct [MD\_DwellPkt\_t](#AAAAAAAAGP)

### Macros

* #define [MD\_HK\_TLM\_LNGTH](#AAAAAAAAGQ)  sizeof([MD\_HkTlm\_t](#AAAAAAAAGO))
* #define [MD\_DWELL\_PKT\_LNGTH](#AAAAAAAAGR)  (sizeof([MD\_DwellPkt\_t](#AAAAAAAAGP)))

### Macro Definition Documentation

#### #define MD\_DWELL\_PKT\_LNGTH  (sizeof([MD\_DwellPkt\_t](#AAAAAAAAGP)))

#### #define MD\_HK\_TLM\_LNGTH  sizeof([MD\_HkTlm\_t](#AAAAAAAAGO))

#### 

## fsw/src/md\_msgdefs.h File Reference

### Memory Dwell Command Codes

* #define [MD\_NOOP\_CC](#AAAAAAAABK)  0
* #define [MD\_RESET\_CNTRS\_CC](#AAAAAAAABN)  1
* #define [MD\_START\_DWELL\_CC](#AAAAAAAABO)  2
* #define [MD\_STOP\_DWELL\_CC](#AAAAAAAABQ)  3
* #define [MD\_JAM\_DWELL\_CC](#AAAAAAAABR)  4

### Macro Definition Documentation

#### #define MD\_JAM\_DWELL\_CC  4

##### [Purpose](#cfsmdcmds__cfsmdcmds000005):

Jam Dwell

##### Description

This command inserts the specified dwell parameters (dwell address, dwell field length, and delay count) into the specified table, at the specified index.

Note that it is safe to send a Jam command to an active Dwell Table. ('Active' indicates a Table which is enabled; thus, the Dwell Table is actively being used to generate a dwell packet telemetry stream.) Note that changes made to a Dwell Table using a Jam command will not be saved across process resets in this version of Memory Dwell.

For details on what constitutes a valid Dwell Table see [MD\_DwellTableLoad\_t](#AAAAAAAAAT). In particular, note that a valid entry *may* be inserted past a terminator entry; however it won't be processed as long as it remains following a terminator entry.

##### Command Structure

[MD\_CmdJam\_t](#AAAAAAAABS)

##### Command Verification

Nominal successful execution of this command may be verified with the following telemetry:

* - command execution counter increments.
* The [MD\_JAM\_DWELL\_INF\_EID](#AAAAAAAAFB) or [MD\_JAM\_NULL\_DWELL\_INF\_EID](#AAAAAAAAFC) informational event message is issued.

##### Error Conditions

This command may fail for the following reason(s):

* Unexpected command length (Event message [MD\_CMD\_LEN\_ERR\_EID](#AAAAAAAAEZ) is issued)
* Table Id other than 1..MD\_NUM\_DWELL\_TABLES (Event message [MD\_INVALID\_JAM\_TABLE\_ERR\_EID](#AAAAAAAAFD) is issued)
* Entry Id other than 1..MD\_DWELL\_TABLE\_SIZE (Event message [MD\_INVALID\_ENTRY\_ARG\_ERR\_EID](#AAAAAAAAFE) is issued)
* Unrecognized Dwell Address symbol (Event message [MD\_CANT\_RESOLVE\_JAM\_ADDR\_ERR\_EID](#AAAAAAAAFG) is issued)
* Dwell Field Length other than 0, 1, 2, or 4 (Event message [MD\_INVALID\_LEN\_ARG\_ERR\_EID](#AAAAAAAAFF) is issued)
* Specified Dwell Address is out of range (Event message [MD\_INVALID\_JAM\_ADDR\_ERR\_EID](#AAAAAAAAFH) is issued)
* Specified Dwell Address is not properly aligned for the specified Dwell Length (Event message [MD\_JAM\_ADDR\_NOT\_32BIT\_ERR\_EID](#AAAAAAAAFI) or [MD\_JAM\_ADDR\_NOT\_16BIT\_ERR\_EID](#AAAAAAAAFJ) is issued)

Any time the command fails, the command error counter increments.

##### Criticality

None.

##### See also:

#### #define MD\_NOOP\_CC  0

##### [Purpose](#cfsmdcmds__cfsmdcmds000001):

Memory Dwell No-Op Command

##### Description

This command increments the MD application's valid command execution counter.

##### Command Structure

[MD\_NoArgsCmd\_t](#AAAAAAAABL)

##### Command Verification

Successful execution of this command may be verified with the following telemetry:

* - command execution counter will increment
* The [MD\_NOOP\_INF\_EID](#AAAAAAAAEQ) informational event message will be generated

##### Error Conditions

This command may fail for the following reason(s):

* Unexpected command length.

Evidence of an unexpected command length error may be found in the following telemetry:

* - command error counter will increment.
* The [MD\_CMD\_LEN\_ERR\_EID](#AAAAAAAAEZ) error event message will be issued.

##### Criticality

None

##### See also:

#### #define MD\_RESET\_CNTRS\_CC  1

##### [Purpose](#cfsmdcmds__cfsmdcmds000002):

Memory Dwell Reset Counters Command

##### Description

This command resets the following counters within the Memory Dwell housekeeping telemetry:

* Command Execution Counter ()
* Command Error Counter ()

##### Command Structure

[MD\_NoArgsCmd\_t](#AAAAAAAABL)

##### Command Verification

Successful execution of this command may be verified with the following telemetry:

* - command execution counter will be set to zero.
* - command error counter will be set to zero.
* The [MD\_RESET\_CNTRS\_DBG\_EID](#AAAAAAAAER) debug event message will be generated.

##### Error Conditions

This command may fail for the following reason(s):

* Unexpected command length.

Evidence of an unexpected command length error may be found in the following telemetry:

* - command error counter will increment.
* The [MD\_CMD\_LEN\_ERR\_EID](#AAAAAAAAEZ) error event message will be issued.

##### Criticality

This command is not inherently dangerous. However, it is possible for ground systems and on-board safing procedures to be designed such that they react to changes in the counter values that are reset by this command.

##### See also:

#### #define MD\_START\_DWELL\_CC  2

##### [Purpose](#cfsmdcmds__cfsmdcmds000003):

Memory Dwell Start Dwell Command

##### Description

This command sets the Enabled flag(s) associated with the Dwell Table(s) that have been designated by the command's TableMask argument.

When this Enabled flag is set, and the associated Dwell Table has one or more dwell specifications defined beginning with the Table's first entry, and the Table has a non-zero value for total delays (aka as the Rate), dwell processing will occur.

The first dwell occurs on receipt of the first Wakeup Message from the Scheduler following the Start Dwell Command. That dwell means that a value from memory is read, and inserted into the Dwell Packet.

Dwell Packets are issued at a rate specified by the Rate value associated with the Dwell Table. The Rate is a multiple of Wakeup Messages issued from the Scheduler. The Rate value is calculated as the sum of all the individual delays specified by individual dwell entries in a Dwell Table.

Note that the dwell state will not be affected for the Dwell Tables *not* designated by the TableMask argument. Thus, for example, if Dwell Table #1 has already been started, and a Start Dwell Command is issued to start Dwell Tables #2 and #3, Dwell Table #1 will still be in started state following the command.

Note that if this command is issued when the Dwell Table has already been started, the effect will be to restart the table. The current entry will be set to the first entry and any data previously collected will be lost.

Note that the value of the Enabled flag is also updated when a Dwell Table is loaded.

##### Command Structure

[MD\_CmdStartStop\_t](#AAAAAAAABP)

##### Command Verification

Successful execution of this command may be verified with the following telemetry:

* - command execution counter will increment.
* - Dwell Table #x flag, for x=1..[MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL), set to 1 (TRUE).
* The [MD\_START\_DWELL\_INF\_EID](#AAAAAAAAES) informational event message will be issued.

##### Error Conditions

This command may fail for the following reasons:

* Unexpected command length.
* Dwell Table mask argument contains no valid table values ( 1..[MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL)).

Evidence of an unexpected command length error may be found in the following telemetry:

* - command error counter increments.
* The [MD\_CMD\_LEN\_ERR\_EID](#AAAAAAAAEZ) error event message is issued.

Evidence of an invalid value for Dwell Table mask argument may be found in the following telemetry:

* - command error counter increments.
* The [MD\_EMPTY\_TBLMASK\_ERR\_EID](#AAAAAAAAEU) error event message is issued.

##### Criticality

None.

##### See also:

[MD\_STOP\_DWELL\_CC](#AAAAAAAABQ)

#### #define MD\_STOP\_DWELL\_CC  3

##### [Purpose](#cfsmdcmds__cfsmdcmds000004):

Memory Dwell Stop Dwell Command

##### Description

This command clears the Enabled flag(s) associated with the Dwell Table(s) that have been designated by the command's TableMask argument.

When the Enabled flag associated with a Dwell Table is cleared, dwell processing cannot occur for that Dwell Table.

Note that the value of the Enabled flag is also updated when a Dwell Table is loaded.

Note that the dwell state will not be affected for the Dwell Tables *not* designated by the TableMask argument. Thus, for example, if a Stop Dwell Command is issued to stop Dwell Table #2, all *other* Dwell Tables will remain in the same state following the command that they were in before the command was received.

##### Command Structure

[MD\_CmdStartStop\_t](#AAAAAAAABP)

##### Command Verification

Successful execution of this command may be verified with the following telemetry:

* - command execution counter increments.
* - Dwell Table #x flag, for x=1..[MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL), clears (i.e. is set to zero/FALSE).
* The [MD\_STOP\_DWELL\_INF\_EID](#AAAAAAAAET) informational event message is issued.

##### Error Conditions

This command may fail for the following reason(s):

* Unexpected command length.
* Dwell Table mask argument contains no valid table values ( 1..[MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL)).

Evidence of an unexpected command length error may be found in the following telemetry:

* - command error counter increments.
* The [MD\_CMD\_LEN\_ERR\_EID](#AAAAAAAAEZ) error event message is issued.

Evidence of an invalid value for Dwell Table argument may be found in the following telemetry:

* - command error counter increments.
* The [MD\_EMPTY\_TBLMASK\_ERR\_EID](#AAAAAAAAEU) error event message is issued.

##### Criticality

None.

##### See also:

[MD\_START\_DWELL\_CC](#AAAAAAAABO)

## fsw/src/md\_tbldefs.h File Reference

#include "md\_platform\_cfg.h"

#include "cfe.h"

#include "cfs\_utils.h"

### Data Structures

* struct [MD\_TableLoadEntry\_t](#AAAAAAAADV)
* struct [MD\_DwellTableLoad\_t](#AAAAAAAAAT)

### *Memory Dwell Table Load structure.* Macros

* #define [MD\_TBL\_LOAD\_ENTRY\_LNGTH](#AAAAAAAAGU)  sizeof([MD\_TableLoadEntry\_t](#AAAAAAAADV))
* #define [MD\_TBL\_LOAD\_LNGTH](#AAAAAAAAGV)  sizeof([MD\_DwellTableLoad\_t](#AAAAAAAAAT))

### Macro Definition Documentation

#### #define MD\_TBL\_LOAD\_ENTRY\_LNGTH  sizeof([MD\_TableLoadEntry\_t](#AAAAAAAADV))

#### #define MD\_TBL\_LOAD\_LNGTH  sizeof([MD\_DwellTableLoad\_t](#AAAAAAAAAT))

#### 

## fsw/src/md\_utils.c File Reference

#include "cfe.h"

#include "md\_utils.h"

#include "md\_app.h"

#include <string.h>

### Functions

* bool [MD\_TableIsInMask](#AAAAAAAAGX) (int16 TableId, uint16 TableMask)

*Determine if specified TableId is contained in argument mask.*

* void [MD\_UpdateDwellControlInfo](#AAAAAAAAGY) (uint16 TableIndex)

*Update Dwell Table Control Info.*

* bool [MD\_ValidEntryId](#AAAAAAAAGZ) (uint16 EntryId)

*Validate Entry Index.*

* bool [MD\_ValidAddrRange](#AAAAAAAAHA) (cpuaddr Addr, uint32 Size)

*Validate Dwell Address.*

* bool [MD\_ValidTableId](#AAAAAAAAHB) (uint16 TableId)

*Validate Table ID.*

* bool [MD\_ValidFieldLength](#AAAAAAAAHC) (uint16 FieldLength)

*Validate Field Length.*

### Variables

* [MD\_AppData\_t](#AAAAAAAABG) [MD\_AppData](#AAAAAAAAHD)

### Function Documentation

#### bool MD\_TableIsInMask (int16 *TableId*, uint16 *TableMask*)

Determine if specified TableId is contained in argument mask.

##### Description

Determines whether specified Table Id is contained in argument mask.

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *TableId* | identifies dwell table (1..[MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL)) |
| in | *TableMask* | Mask representing current status of all dwell tables. |

##### Returns:

Returns TRUE or FALSE

#### void MD\_UpdateDwellControlInfo (uint16 *TableIndex*)

Update Dwell Table Control Info.

##### Description

Updates the control structure used by the application for dwell packet processing with address count, data size, and rate.

##### Assumptions, External Events, and Notes:

A zero value for length in a dwell table entry represents the end of the active portion of a dwell table.

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *TableIndex* | identifies dwell control structure (0..[MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL)-1) |

##### Return values:

|  |  |
| --- | --- |
| *None* |  |

#### bool MD\_ValidAddrRange (cpuaddr *Addr*, uint32 *Size*)

Validate Dwell Address.

##### Description

This function validates that the memory range as specified by the input address and size is valid for reading.

##### Assumptions, External Events, and Notes:

None

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *Addr* | Dwell address. |
| in | *Size* | Size, in bytes, of field to be read. |

##### Returns:

Returns TRUE or FALSE

#### bool MD\_ValidEntryId (uint16 *EntryId*)

Validate Entry Index.

##### Description

Checks for valid value (1..MD\_DWELL\_TABLE\_SIZE ) for entry id specified in Jam command.

##### Assumptions, External Events, and Notes:

None

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *EntryId* | EntryId (starting at one) for dwell control structure entry. |

##### Returns:

Returns TRUE or FALSE

#### bool MD\_ValidFieldLength (uint16 *FieldLength*)

Validate Field Length.

##### Description

Check valid range for dwell field length.

##### Assumptions, External Events, and Notes:

Valid values for dwell field length are 0, 1, 2, and 4. 0 corresponds to a null entry in Dwell Table.

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *FieldLength* | Length of field, in bytes, to be copied for dwell. |

##### Returns:

Returns TRUE or FALSE

#### bool MD\_ValidTableId (uint16 *TableId*)

Validate Table ID.

##### Description

Check valid range for TableId argument used in several Memory Dwell commands. Valid range is 1..[MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL).

##### Assumptions, External Events, and Notes:

Note that this value will be internally converted to 0..([MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL)-1) for indexing into arrays.

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *TableId* | Table ID. |

##### Returns:

Returns TRUE or FALSE

### Variable Documentation

#### [MD\_AppData\_t](#AAAAAAAABG) MD\_AppData

#### 

## fsw/src/md\_utils.h File Reference

#include "md\_msg.h"

#include "cfe.h"

### Functions

* bool [MD\_TableIsInMask](#AAAAAAAAHF) (int16 TableId, uint16 TableMask)

*Determine if specified TableId is contained in argument mask.*

* void [MD\_UpdateDwellControlInfo](#AAAAAAAAHG) (uint16 TableIndex)

*Update Dwell Table Control Info.*

* bool [MD\_ValidEntryId](#AAAAAAAAHH) (uint16 EntryId)

*Validate Entry Index.*

* bool [MD\_ValidAddrRange](#AAAAAAAAHI) (cpuaddr Addr, uint32 Size)

*Validate Dwell Address.*

* bool [MD\_ValidTableId](#AAAAAAAAHJ) (uint16 TableId)

*Validate Table ID.*

* bool [MD\_ValidFieldLength](#AAAAAAAAHK) (uint16 FieldLength)

*Validate Field Length.*

### Function Documentation

#### bool MD\_TableIsInMask (int16 *TableId*, uint16 *TableMask*)

Determine if specified TableId is contained in argument mask.

##### Description

Determines whether specified Table Id is contained in argument mask.

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *TableId* | identifies dwell table (1..[MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL)) |
| in | *TableMask* | Mask representing current status of all dwell tables. |

##### Returns:

Returns TRUE or FALSE

#### void MD\_UpdateDwellControlInfo (uint16 *TableIndex*)

Update Dwell Table Control Info.

##### Description

Updates the control structure used by the application for dwell packet processing with address count, data size, and rate.

##### Assumptions, External Events, and Notes:

A zero value for length in a dwell table entry represents the end of the active portion of a dwell table.

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *TableIndex* | identifies dwell control structure (0..[MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL)-1) |

##### Return values:

|  |  |
| --- | --- |
| *None* |  |

#### bool MD\_ValidAddrRange (cpuaddr *Addr*, uint32 *Size*)

Validate Dwell Address.

##### Description

This function validates that the memory range as specified by the input address and size is valid for reading.

##### Assumptions, External Events, and Notes:

None

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *Addr* | Dwell address. |
| in | *Size* | Size, in bytes, of field to be read. |

##### Returns:

Returns TRUE or FALSE

#### bool MD\_ValidEntryId (uint16 *EntryId*)

Validate Entry Index.

##### Description

Checks for valid value (1..MD\_DWELL\_TABLE\_SIZE ) for entry id specified in Jam command.

##### Assumptions, External Events, and Notes:

None

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *EntryId* | EntryId (starting at one) for dwell control structure entry. |

##### Returns:

Returns TRUE or FALSE

#### bool MD\_ValidFieldLength (uint16 *FieldLength*)

Validate Field Length.

##### Description

Check valid range for dwell field length.

##### Assumptions, External Events, and Notes:

Valid values for dwell field length are 0, 1, 2, and 4. 0 corresponds to a null entry in Dwell Table.

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *FieldLength* | Length of field, in bytes, to be copied for dwell. |

##### Returns:

Returns TRUE or FALSE

#### bool MD\_ValidTableId (uint16 *TableId*)

Validate Table ID.

##### Description

Check valid range for TableId argument used in several Memory Dwell commands. Valid range is 1..[MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL).

##### Assumptions, External Events, and Notes:

Note that this value will be internally converted to 0..([MD\_NUM\_DWELL\_TABLES](#AAAAAAAAAL)-1) for indexing into arrays.

##### Parameters:

|  |  |  |
| --- | --- | --- |
| in | *TableId* | Table ID. |

##### Returns:

Returns TRUE or FALSE

## fsw/src/md\_verify.h File Reference

#include "cfe\_platform\_cfg.h"

#include "md\_platform\_cfg.h"

## fsw/src/md\_version.h File Reference

### Macros

* #define [MD\_MAJOR\_VERSION](#AAAAAAAAHN)  2
* #define [MD\_MINOR\_VERSION](#AAAAAAAAHO)  4
* #define [MD\_REVISION](#AAAAAAAAHP)  0

### Macro Definition Documentation

#### #define MD\_MAJOR\_VERSION  2

#### #define MD\_MINOR\_VERSION  4

#### #define MD\_REVISION  0

#### 

## fsw/tables/md\_dw01.c File Reference

#include "cfe.h"

#include "md\_tbldefs.h"

#include "md\_app.h"

#include "cfs\_utils.h"

#include "cfe\_tbl\_filedef.h"

#include "md\_platform\_cfg.h"

### Variables

* [MD\_DwellTableLoad\_t](#AAAAAAAAAT) [MD\_Default\_Dwell1\_Tbl](#AAAAAAAAHR)

### Variable Documentation

#### [MD\_DwellTableLoad\_t](#AAAAAAAAAT) MD\_Default\_Dwell1\_Tbl

#### 

## fsw/tables/md\_dw02.c File Reference

#include "cfe.h"

#include "md\_tbldefs.h"

#include "md\_app.h"

#include "cfs\_utils.h"

#include "cfe\_tbl\_filedef.h"

#include "md\_platform\_cfg.h"

### Variables

* [MD\_DwellTableLoad\_t](#AAAAAAAAAT) [MD\_Default\_Dwell2\_Tbl](#AAAAAAAAHT)

### Variable Documentation

#### [MD\_DwellTableLoad\_t](#AAAAAAAAAT) MD\_Default\_Dwell2\_Tbl

#### 

## fsw/tables/md\_dw03.c File Reference

#include "cfe.h"

#include "md\_tbldefs.h"

#include "md\_app.h"

#include "cfs\_utils.h"

#include "cfe\_tbl\_filedef.h"

#include "md\_platform\_cfg.h"

### Variables

* [MD\_DwellTableLoad\_t](#AAAAAAAAAT) [MD\_Default\_Dwell3\_Tbl](#AAAAAAAAHV)

### Variable Documentation

#### [MD\_DwellTableLoad\_t](#AAAAAAAAAT) MD\_Default\_Dwell3\_Tbl

#### 

## fsw/tables/md\_dw04.c File Reference

#include "cfe.h"

#include "md\_tbldefs.h"

#include "md\_app.h"

#include "cfs\_utils.h"

#include "cfe\_tbl\_filedef.h"

#include "md\_platform\_cfg.h"

### Variables

* [MD\_DwellTableLoad\_t](#AAAAAAAAAT) [MD\_Default\_Dwell4\_Tbl](#AAAAAAAAHX)

### Variable Documentation

#### [MD\_DwellTableLoad\_t](#AAAAAAAAAT) MD\_Default\_Dwell4\_Tbl