**Module Design Document**

**For**

**FlsMem**

**Aug 25 , 2016**

**Prepared For:**

**Software Engineering**

**Nexteer Automotive,**

**Saginaw, MI, USA**

**Prepared By:**

**Software Group,**

**Nexteer Automotive,**

**Saginaw, MI, USAChange History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Description** | **Author** | **Version** | **Date** |
| Initial Version | Lucas Wendling | 1.0 | 10/06/15 |
| Updated with changes for DTS configuration for Flash CRC check | Avinash James | 2.0 | 03/18/16 |
| Updates for DTS Transfer Clear | Avinash James | 3.0 | 3/29/16 |
| Updated for removing the flash ECC single bit error handling and disabling the DTS channels after calculation | Avinash James | 4.0 | 3/31/16 |
| Trusted function call for the DTS clean up updates | Avinash James | 5.0 | 04/18/16 |
| Function name changes and added CodFlsSngBitEcc handler for single bit code flash ecc | Avinash James | 6.0 | 08/25/16 |

Table of Contents[1 Introduction 5](#_Toc448756655)

[1.1 Purpose 5](#_Toc448756656)

[1.2 Scope 5](#_Toc448756657)

[2 FlsMem & High-Level Description 6](#_Toc448756658)

[3 Design details of software module 7](#_Toc448756659)

[3.1 Graphical representation of FlsMem 7](#_Toc448756660)

[3.2 Data Flow Diagram 7](#_Toc448756661)

[3.2.1 Component level DFD 7](#_Toc448756662)

[3.2.2 Function level DFD 7](#_Toc448756663)

[4 Constant Data Dictionary 8](#_Toc448756664)

[4.1 Program (fixed) Constants 8](#_Toc448756665)

[4.1.1 Embedded Constants 8](#_Toc448756666)

[5 Variable Data Dictionary 9](#_Toc448756667)

[5.1 User defined typedef definition/declaration 9](#_Toc448756668)

[5.2 Variable definition for enumerated types 9](#_Toc448756669)

[6 Software Component Implementation 10](#_Toc448756670)

[6.1 Sub-Module Functions 10](#_Toc448756671)

[6.1.1 Init: FlsMemInit1 10](#_Toc448756672)

[6.1.1.1 Design Rationale 10](#_Toc448756673)

[6.1.1.2 Module Outputs 10](#_Toc448756674)

[6.1.2 Init: FlsMemInit2 10](#_Toc448756675)

[6.1.2.1 Design Rationale 10](#_Toc448756676)

[6.1.2.2 Module Outputs 10](#_Toc448756677)

[6.1.3 Per: FlsMemPer2 10](#_Toc448756678)

[6.1.3.1 Design Rationale 10](#_Toc448756679)

[6.1.3.2 Store Module Inputs to Local copies 10](#_Toc448756680)

[6.1.3.3 (Processing of function)……… 10](#_Toc448756681)

[6.1.3.4 Store Local copy of outputs into Module Outputs 10](#_Toc448756682)

[6.2 Server Runnables 11](#_Toc448756683)

[6.3 Interrupt Functions 11](#_Toc448756684)

[6.4 Module Internal (Local) Functions 11](#_Toc448756685)

[6.4.1 Local Function #1 11](#_Toc448756686)

[6.4.1.1 Design Rationale 11](#_Toc448756687)

[6.4.1.2 Processing 11](#_Toc448756688)

[6.5 GLOBAL Function/Macro Definitions 11](#_Toc448756689)

[6.5.1 DTSInit 11](#_Toc448756690)

[6.5.1.1 Design Rationale 11](#_Toc448756691)

[6.5.1.2 Processing 14](#_Toc448756692)

[6.5.2 DTSClnUp 14](#_Toc448756693)

[6.5.2.1 Design Rationale 14](#_Toc448756694)

[6.5.2.2 Processing 14](#_Toc448756695)

[7 Known Limitations with Design 15](#_Toc448756696)

[8 UNIT TEST CONSIDERATION 16](#_Toc448756697)

[Appendix A Abbreviations and Acronyms 17](#_Toc448756698)

[Appendix B Glossary 18](#_Toc448756699)

[Appendix C References 19](#_Toc448756700)

# Introduction

## Purpose

## Scope

The following definitions are used throughout this document:

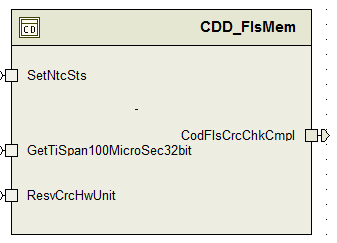
* **Shall**: indicates a mandatory requirement without exception in compliance.
* **Should**: indicates a mandatory requirement; exceptions allowed only with documented justification.
* **May**: indicates an optional action.

# FlsMem & High-Level Description

*See FDD*

# Design details of software module

## Graphical representation of FlsMem

**

## Data Flow Diagram

### Component level DFD

*See FDD*

### Function level DFD

*See FDD*

# Constant Data Dictionary

## Program (fixed) Constants

### Embedded Constants

#### Local Constants

|  |  |  |  |
| --- | --- | --- | --- |
| Constant Name | Resolution | Units | Value |
| **CPU1PEID\_CNT\_U32** | **1** | **uint32** | **0x01U** |
| **CODFLSTOCRCSPID\_CNT\_U32** | **1** | **uint32** | **0x02U** |
| **CRCTOLCLRAMSPID\_CNT\_U32** | **1** | **uint32** | **0x00U** |
| **USRMODDIS\_CNT\_U32** | **1** | **uint32** | **0x00U** |
| **FLSBLKLEN\_CNT\_U32** | **1** | **uint32** | **0x0003FFFCU** |
| **DTSDATALEN\_CNT\_U32** | **1** | **uint32** | **4U** |
| **CRCCHKMAXALLWDTI\_CNT\_U32** | **1** | **uint32** | **2000** |
| **MAXNROFDTSCH\_CNT\_U32** | **1** | **uint32** | **32** |
| **DUMMYREADADDR1\_CNT\_U32** | **1** | **uint32** | **(0xFFFFFE1FU)** |
| **DUMMYREADADDR2\_CNT\_U32** | **1** | **uint32** | **(0xFFFFFE2FU)** |
| **DUMMYREADADDR3\_CNT\_U32** | **1** | **uint32** | **(0xFFFFFE4FU)** |
| **DUMMYREADADDR4\_CNT\_U32** | **1** | **uint32** | **(0xFFFFFE8FU)** |

# Variable Data Dictionary

## User defined typedef definition/declaration

*<This section documents any user types uniquely used for the module.>*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Typedef Name | Element Name | User Defined Type | Legal Range  (min) | Legal Range  (max) |
| FlsCrcCfgBlkRec | CrcFlsBlkStrtAdr | uint32 | 0 | 0xFFFFFFFFH |
|  | CrcFlsBlkLen | uint32 | 0 | 0xFFFFFFFFH |
|  | PreCalcnCrcFlsAdr | uint32\* | 0 | 0xFFFFFFFFH |

## Variable definition for enumerated types

|  |  |  |
| --- | --- | --- |
| Enum Name | Element Name | Value |
| <(Name given for the user defined typdef of type struct/union)  (Variable name qualified in refer[2])> | <(Variable name qualified Refer[2])> | <Define the value > |

# Software Component Implementation

## Sub-Module Functions

## Init: FlsMemInit1

## Design Rationale

*Empty function for purposes of memory mapping*

## Module Outputs

*None*

## Init: FlsMemInit2

## Design Rationale

*The FlsMemInit2 function is a non RTE function which shall be called to set up the DTS configuration for the Flash CRC check. The DTS channel configuration has to be applied only when the system is waking up from a Power On Reset or after a flash programming reset. In such a scenario a Hardware CRC unit is allocated by function call to the CRC module and once a hardware assignment is successful, the DTS channels are configured for chaining for the entire definition of the flash blocks (Boot, App, Cal1, Cal2 etc.). Record the time when the DTS transfer is initiated so that a check on a timeout can be made in the periodic function where a maximum timeout of 200 ms is checked for*

*This function shall be called in the startup sequence. Hence it is a non RTE function*

*See FDD for more.*

## Module Outputs

*None*

*None*

## Per: FlsMemPer2

## Design Rationale

*See FDD*

## Store Module Inputs to Local copies

*Refer to FDD*

## (Processing of function)………

*Refer to FDD*

## Store Local copy of outputs into Module Outputs

*Refer to FDD*

## Server Runnables - CodFlsSngBitEcc

## Design Rationale

*See FDD*

## Store Module Inputs to Local copies

*Refer to FDD*

## (Processing of function)………

*Refer to FDD*

## Store Local copy of outputs into Module Outputs

*Refer to FDD*

## Interrupt Functions

*None*

## Module Internal (Local) Functions

## Local Function #1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Function Name** | (Exact name used) | Type | Min | Max |
| **Arguments Passed** | None | <Refer MDD guidelines[1]> | <Refer MDD guidelines[1]> | <Refer MDD guidelines[1]> |
|  |  |  |  |  |
| **Return Value** |  |  |  |  |

## Design Rationale

## Processing

## GLOBAL Function/Macro Definitions

## DtsInin

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Function Name** | DTSInit | Type | Min | Max |
| **Arguments Passed** | CrcHwIdxInReg | uint32 | 0 | 0xFFFFFFFF |
|  | CrcHwIdxOutReg | uint32 | 0 | 0xFFFFFFFF |
| **Return Value** | None |  |  |  |

## Design Rationale

Trusted function that performs all register initialization from the CM102A\_FlsMem\_DTSPeripheralCfg.xlsx spreadsheet in the FDD. The DTSMstrCfg channel master registers can be written only in supervisor mode. After the Channel master register for a given channel has been written, the selected Processor Element can write to that channel’s registers. However, for simplicity, all DTS register initialization and chaining is being done in one trusted function.

The chaining is done in the following manner

1. Consider the first flash region to have the CRC calculated
2. Calculate the number of DTS chains required for the length of the CRC region. Each DTS channel can address up to a maximum of 0x3FFFC bytes of data (0xFFFF maximum transfer count multiplied by 4 bytes of data in each transfer).

Hence number of channel is equal to Region length/0x3FFFC + {1} if (Region length % 0x3FFFC is non zero)

1. Clear the DTS Transfer flag to make sure no pending requests are present for all the used channels
2. Configure the DTS channels starting from 0 using the configuration defined as per CM102A\_FlsMem\_DTSPeripheralCfg.xlsx for the above calculated number of chains
3. Configure the next DTS channel to transfer the CRC result from CRC HW output register to Per Instance Memory
4. Configure the next DTS channel to transfer zero value to the CRC HW output register to clear the output register to continue with next flash region operation
5. Repeat Step 1 thru 5 for all the flash regions(Boot, App, Cal1, Cal2 etc) The definition of the flash region is in the generated file CDD\_FlsMem\_Cfg.c which takes inputs defined in the Vector configurator Tool
6. Disable chaining on the last channel
7. Enable the Interrupt on the second last channel
8. Clear the interrupt status register which shall be monitored in the periodic
9. Start the DTS transfer



## Processing

## DtsClnUp

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Function Name** | DTSClnUp | Type | Min | Max |
| **Arguments Passed** | None |  |  |  |
|  |  |  |  |  |
| **Return Value** | None |  |  |  |

## Design Rationale

***None***

## Processing

*None*

# Known Limitations with Design

We have made use of a static constant global variable (**static const uint32 CrcClrData\_M = 0U**) for the purpose of clearing the CRC hardware

Also the result array (**HwCrcCalcdRes\_C[8]**) has been also declared as a global array for the purpose of DTS write access in the MotCtrlMgr\_MemMap memory map section

In the CodFlsSngBitEcc function in FDD the reads are done to the same variable. In the implementation we have used 4 different variables for the purpose that compiler wont optimize those reads. Any optimization settings change in the compiler would need a reevaluation of the use of volatile temporary variables

# UNIT TEST CONSIDERATION

None

Abbreviations and Acronyms

| **Abbreviation or Acronym** | **Description** |
| --- | --- |
|  |  |
|  |  |

Glossary

**Note**: Terms and definitions from the source “Nexteer Automotive” take precedence over all other definitions of the same term. Terms and definitions from the source “Nexteer Automotive” are formulated from multiple sources, including the following:

* ISO 9000
* ISO/IEC 12207
* ISO/IEC 15504
* Automotive SPICE® Process Reference Model (PRM)
* Automotive SPICE® Process Assessment Model (PAM)
* ISO/IEC 15288
* ISO 26262
* IEEE Standards
* SWEBOK
* PMBOK
* Existing Nexteer Automotive documentation

| **Term** | **Definition** | **Source** |
| --- | --- | --- |
| MDD | Module Design Document |  |
| DFD | Data Flow Diagram |  |

References

| **Ref. #** | **Title** | **Version** |
| --- | --- | --- |
| 1 | AUTOSAR Specification of Memory Mapping (Link:[AUTOSAR\_SWS\_MemoryMapping.pdf](http://www.autosar.org/download/R4.0/AUTOSAR_SWS_MemoryMapping.pdf)) | v1.3.0 R4.0 Rev 2 |
| 2 | MDD Guideline | EA4 01.00.00 |
| 3 | [Software Naming Conventions.doc](http://misagweb01.nexteer.com/eRoomReq/Files/erooms8/NextGeneration/0_fc55f/Software%20Naming%20Conventions%2003x(In%20Work).doc) | 1.0 |
| 4 | [Software Design and Coding Standards.doc](http://eroom1.nexteer.com/eRoomReq/Files/erooms8/NextGeneration/0_1a67a9/Software%20Design%20and%20Coding%20Standards.doc) | 2.0 |