User Manual

for S32K1_S32M24X MCL Driver

Document Number: UM2MCLASRR21-11 Rev0000R2.0.0 Rev. 1.0

Chapter 1

Revision History

Revision	Date	Author	Description
1.0	04.08.2023	NXP RTD Team	S32K1_S32M24X Real-Time Drivers AUTOSAR 4.4 & R21-11 Version 2.0.0

Chapter 2

Introduction

This User Manual describes NXP Semiconductor MCL for *platform*. MCL driver configuration parameters and deviations from the specification are described in Driver chapter of this document. MCL driver requirements and APIs are described in the MCL driver software specification document.

2.1 Supported Derivatives

The software described in this document is intended to be used with the following microcontroller devices of NXP Semiconductors:

- s32k116_qfn32
- s32k116_lqfp48
- s32k118_lqfp48
- s32k118_lqfp64
- s32k142_lqfp48
- s32k142_lqfp64
- s32k142_lqfp100
- s32k142w_lqfp48
- s32k142w_lqfp64
- s32k144_lqfp48

- s32k144_lqfp64 / MWCT1014S_lqfp64
- s32k144_lqfp100 / MWCT1014S_lqfp100
- s32k144_mapbga100
- s32k144w_lqfp48
- s32k144w_lqfp64
- s32k146_lqfp64
- s32k146_lqfp100 / MWCT1015S_lqfp100
- s32k146_mapbga100 / MWCT1015S_mapbga100
- s32k146_lqfp144
- s32k148_lqfp100
- s32k148_mapbga100 / MWCT1016S_mapbga100
- s32k148_lqfp144
- s32k148_lqfp176
- s32m241_lqfp64
- s32m242_lqfp64
- s32m243_lqfp64
- s32m244_lqfp64

All of the above microcontroller devices are collectively named as S32K1_S32M24X. Note: MWCT part numbers contain NXP confidential IP for Qi Wireless Power

2.2 Overview

AUTOSAR (AUTomotive Open System ARchitecture) is an industry partnership working to establish standards for software interfaces and software modules for automobile electronic control systems.

AUTOSAR:

- paves the way for innovative electronic systems that further improve performance, safety and environmental friendliness.
- is a strong global partnership that creates one common standard: "Cooperate on standards, compete on implementation".
- is a key enabling technology to manage the growing electrics/electronics complexity. It aims to be prepared for the upcoming technologies and to improve cost-efficiency without making any compromise with respect to quality.
- facilitates the exchange and update of software and hardware over the service life of the vehicle.

2.3 About This Manual

This Technical Reference employs the following typographical conventions:

- **Boldface** style: Used for important terms, notes and warnings.
- *Italic* style: Used for code snippets in the text. Note that C language modifiers such "const" or "volatile" are sometimes omitted to improve readability of the presented code.

Notes and warnings are shown as below:

Note

This is a note.

Warning

This is a warning

2.4 Acronyms and Definitions

Term	Definition
API	Application Programming Interface
ASM	Assembler
BSMI	Basic Software Make file Interface
CAN	Controller Area Network
C/CPP	C and C++ Source Code
CS	Chip Select
CTU	Cross Trigger Unit
DEM	Diagnostic Event Manager
DET	Development Error Tracer
DMA	Direct Memory Access
ECU	Electronic Control Unit
FIFO	First In First Out
LSB	Least Signifigant Bit
MCU	Micro Controller Unit
MIDE	Multi Integrated Development Environment
MSB	Most Significant Bit
N/A	Not Applicable
RAM	Random Access Memory
SIU	Systems Integration Unit
SWS	Software Specification
VLE	Variable Length Encoding
XML	Extensible Markup Language

2.5 Reference List

#	Title	Version
1	S32K1xx Series Reference Manual	Rev. 14, 09/2021
2	Errata S32K116_0N96V	Rev. 22/OCT/2021
3	Errata S32K118_0N97V	Rev. 22/OCT/2021
4	Errata S32K142_0N33V	Rev. 22/OCT/2021
5	Errata S32K144_0N57U	Rev. 22/OCT/2021
6	Errata S32K144W_0P64A	Rev. 22/OCT/2021
7	Errata S32K146_0N73V	Rev. 22/OCT/2021
8	Errata S32K148_0N20V	Rev. 22/OCT/2021
9	S32K1xx Data Sheet	Rev. 14, 08/2021
10	S32M24x Reference Manual	Rev. 2 Draft A, 05/2023
11	Errata S32M244_P64A+P73G	Rev. 0
12	Errata S32M242_N33V+P73G	Rev. 0, 06/2023
13	S32M2xx Data Sheet	Rev. 3 DraftA, 05/2023

Chapter 3

Driver

3.1 Requirements

Requirements for this driver are detailed in the Driver Software Specification document (See Table Reference List).

For CDD: MCL Driver is a Complex Device Driver (CDD), so there are no requirements regarding this module.

It has vendor-specific requirements and implementation.

3.2 Driver Design Summary

The Mcl Driver controls the DMA(Direct Memory Access) and CACHE modules of the S32K1 device. It provides the following features:

- Configuration and initialization of the DMA.
- Configuration and initialization of the CACHE.
- Handling of the DMA interrupt requests.
- DMA Normal Transfer Mode and Scatter/Gather Mode.

3.3 Hardware Resources

The Mcl Driver consists of:

1. A DMA Peripheral which has 1 Hardware Instance with 4 Hardware Channels for S32K11X and 16 Hardware Channels for S32K14X.
2. A CACHE Memory is Local Memory Controller (LMEM).

3.4 Deviations from Requirements

The driver deviates from the Mcl Driver Software Specification in some places.

The table Status Column Description identifies the requirements that are not fully implemented, implemented differently, or out of scope for the Mcl Driver.

The table Mcl Requirements Deviations provides the "Status" column description.

Term	Definition
N/S	Not In Scope
N/F	Not Fully Implemented
N/I	Not Implemented

3.4.0.0.1 Status Column Description

Requirement	Status	Description	Notes
None	None	None	None

3.4.0.0.2 Mcl Requirements Deviations Files **Mcl_<VariantName>_PBcfg.c** and **Mcl_<VariantName>_PBcfg.h** will contain the definitions for all parameters that are variant aware, independent of the configuration class that will be selected (PC, LT, PB).

Files **Mcl_Cfg.c** and **Mcl_Cfg.h** will contain the definitions for all parameters that are not variant aware.

3.5 Driver Limitations

The Mcl Driver has the following limitations:

- When DMA is used with CACHE enabled, the user shall Invalidate/Clean the CACHE in order to synchronize the transfered data.

- When using CACHE Invalidate and Clean functionalities, the user shall take into consideration that all variables that reside in the cache, are affected.
- The Cache Invalidate function need to be called before Cache Enable function is called (make sure that Cache wasn't enabled before).
- The Cache Clean function need to be called before Cache Disable function is called.
- The DMA Driver shall have the Source Address and Destination Address configured at runtime. These two paramaters are not available in the configurator.
- When the DMA transfer has errors, user must call Mcl_SetDmaChannelCommand/Dma_Ip_SetLogic↩ ChannelCommand function to clear error state and error status.
- When using Virtual Address Mapping feature, the user shall handle the linker file to avoid the memory address is invalid.
- For DMA hardware version 2, the ActiveId is not available in CR register.

3.6 Driver usage and configuration tips

3.6.1 MCAL MCL DMA migration guide to RTD MCL

3.6.1.1 Introduction

The RTD MCL DMA Driver brings a Generic Interface, to help the User in application development across multiple SoCs. The Generic Interface consists of software functions that are fully configurable using User defined configurations. The Generic Interface is structured into four function groups:

1. Set Command Functions
2. Get Status Functions
3. Set List Functions
4. Get Information (Parameter) Functions

The Set Command Functions shall trigger actions specific to the invoked Logic Entity. For example, the Logic Dma Instance shall be commanded to: “Stop execution”, “Stop execution with error signaling”, “Pause execution” or “Resume execution”.

The Get Status Functions shall read the status specific to the invoked Logic Entity. For example, the Logic Dma Instance shall return: “Hardware Errors specific to the IP”, “The Active Channel Id” and “Active Status”.

The Set List Functions shall configure a user defined list of settings for the invoked Logic Entity. For example, the Logic Dma Channel shall be configurable with the Transfer List of Parameters like: “Source Address”, “↩ Destination Address”, “Source Signed Offset”, “Destination Signed Offset”, “Major Loop Count”, etc... For the specific ScatterGather mode, the additional Logic Element parameter specifies the Software TCD that shall be loaded with the ScatterGather List of Parameters.

The Get Parameter Functions shall return specific parameter value. For example, the Logic Dma Channel shall return the “Destination Address” value, which contains the last accessed destination memory location.

3.6.1.2 TRESOS Configuration

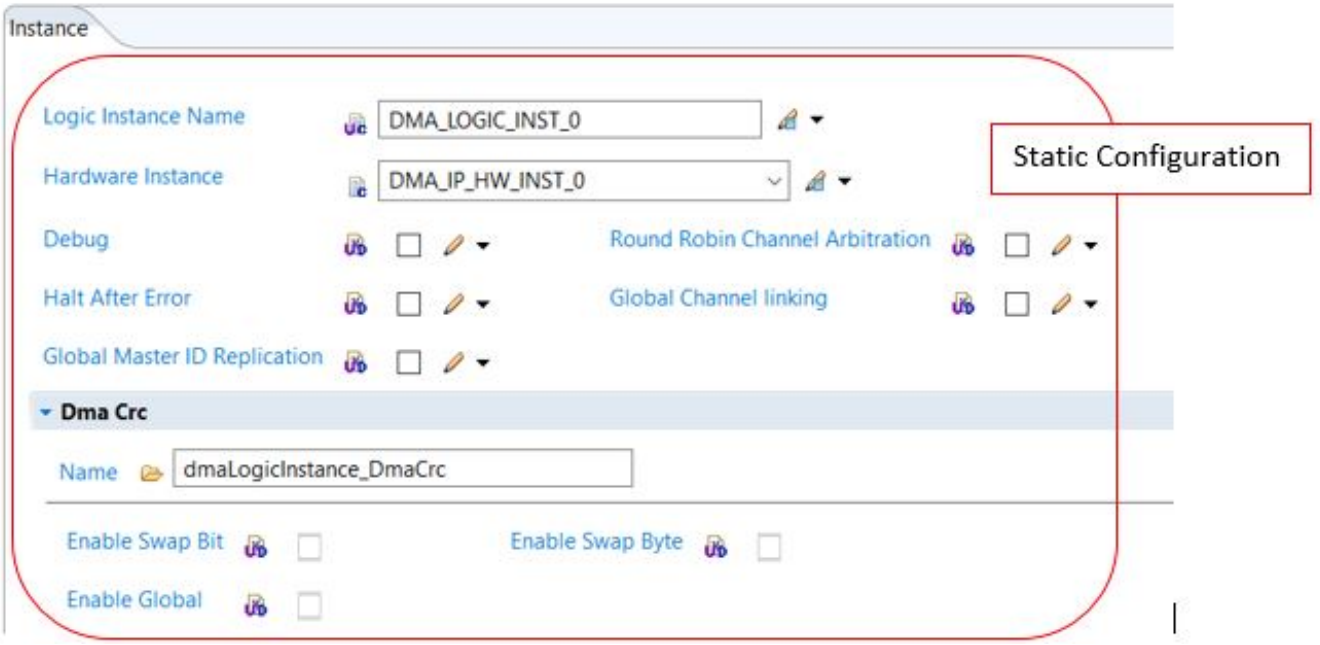

Figure 3.1 DMA Logic Instance

The DMA Logic Instance configuration presented in Figure 3.1 is static, thus it can't be changed dynamically during runtime. The "Logic Instance Name" represents the Handler(Tag) used with the DMA API. The Handler(Tag) is set by the User for the specific Application; for example the "DMA_LOGIC_INSTANCE_0" shall be changed to "DMA_LOGIC_INST_COMMUNICATION".

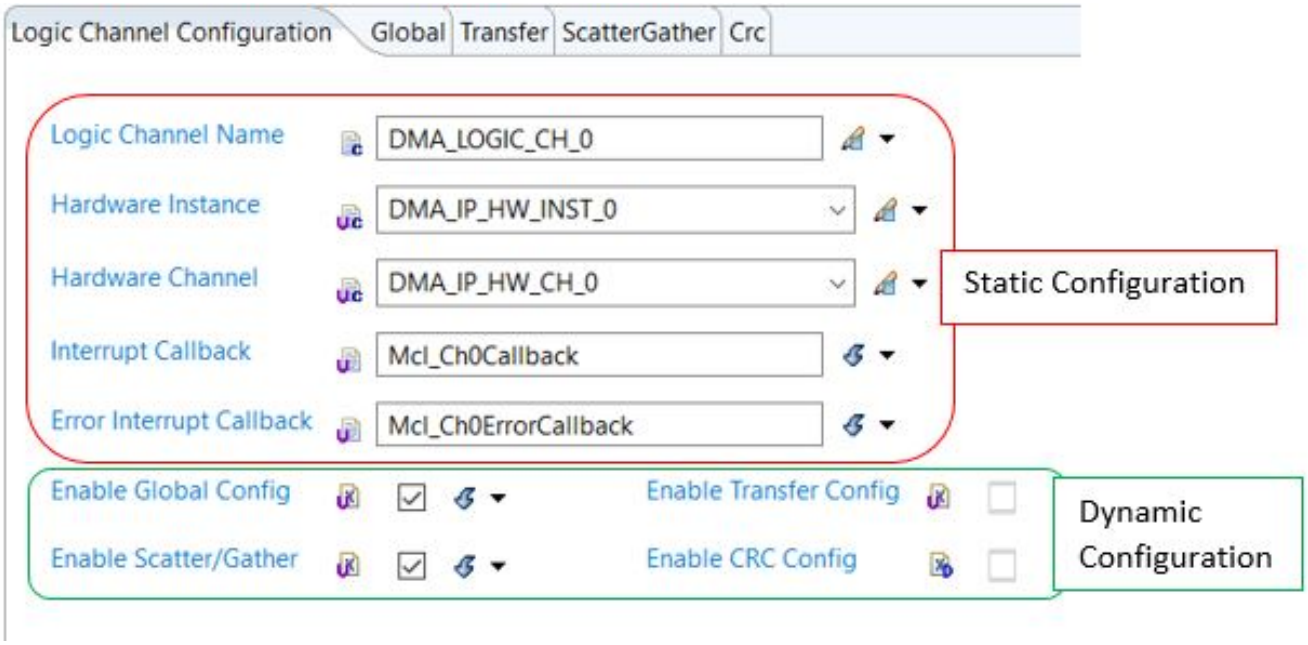

Figure 3.2 DMA Logic Channel

The DMA Logic Channel configuration in Figure 3.2 is composed of static and dynamic settings.

1. The static configuration is needed in order for the DMA Logic Channel to be created and used. The static configuration allocates resources for the DMA Logic Channel basic configuration. The static configuration can't be changed during runtime. The "Logic Channel Name" represents the Handler(Tag) used with the DMA API. The Handler(Tag) is set by the User based on the Application; For example the "DMA_LOGIC_CH_0" shall be changed to "DMA_CH_CAN0_RX" or "DMA_LOGIC_CH_SPI2_TX".
2. The dynamic configuration can be set in TRESOS and shall be automatically loaded in the DMA Logic Channel at initialization time. By enabling a checkbox, the respective configuration option is enabled and system memory is allocated during generation. The dynamic configuration can be set during runtime by using the specific API. By disabling a checkbox, the respective configuration option is disabled and system memory is not allocated during generation.

2.1. The "Enable Global Config" contains settings outside the Transfer Control Descriptor (TCD).

2.2. The "Enable Transfer Config" contains settings of the Transfer Control Descriptor (TCD).

2.3. The "Enable Scatter/Gather" contains settings that extend the "Transfer Config" by creating Software Transfer Control Descriptors (STCDs).

2.4. The "Enable CRC Config" contains settings for the DMA Logic Channel CRC computation functionality.

Note: The DMA Logic Channel can be configured in "Transfer Mode" or "Scatter/Gather Mode", thus only one of the 2 configurations can be set at any time. During runtime, the DMA Logic Channel can be configured between the 2 modes by calling the "SetTransfer" or "SetScatterGather" API.

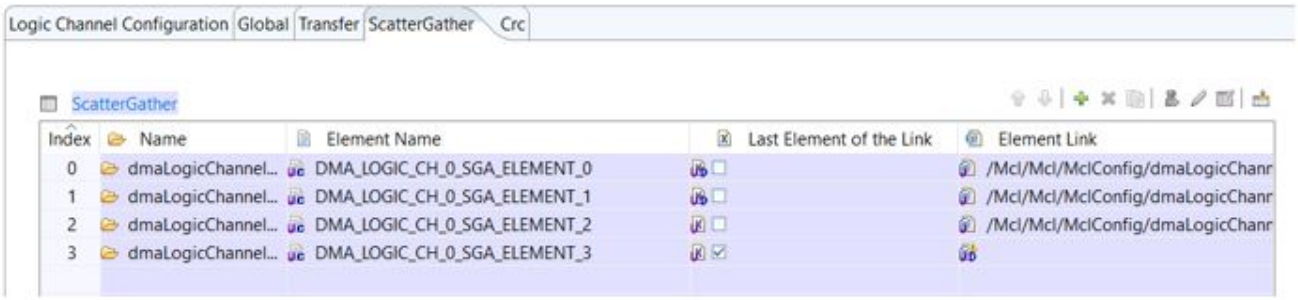

Figure 3.3 DMA Logic Channel – ScatterGather Element List

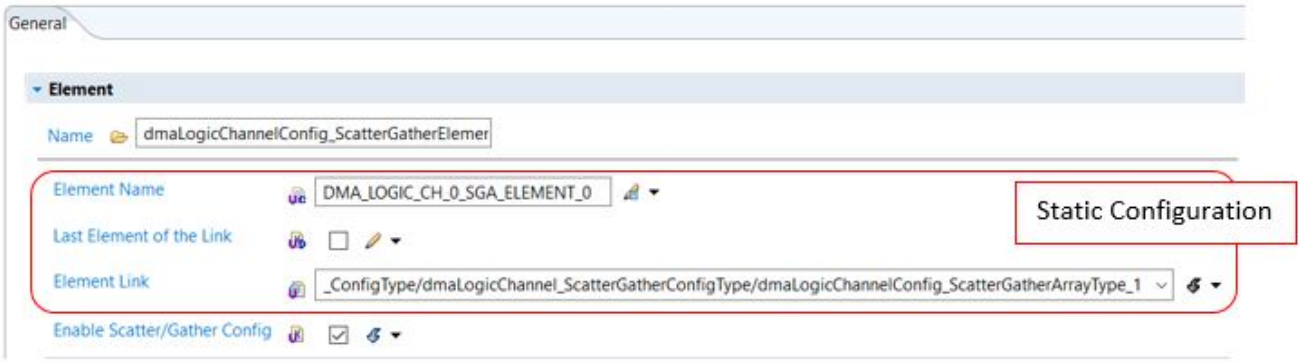

Figure 3.4 DMA Logic Channel – ScatterGather Element Configuration

The DMA Logic Channel ScatterGather Element list (Figure 3.3) shall be loaded with the required number of elements. Each element represents a STCD and is part of a linked list of elements.

The DMA Logic Channel ScatterGather Element Configuration (Figure 3.4) contains the static configuration of the element.

1. The “Element Name” represents the Handler (Tag) (static configuration).
2. The “Last Element of the Link” sets the element as the last link element (static configuration).
3. The “Element Link” points to the next element of the link (static configuration).
4. The “Enable Scatter/Gather Config” enables the element configuration.

Note1: Element can’t be added during runtime. The elements are allocated resources during generation (System memory for STCDs and configuration if set). The element linkage can’t be changed during runtime.

Note2: The DMA Logic Channel ScatterGather Element List can be configured to contain multiple independent chained lists. During runtime, the Logic Channel can be assigned a different Chained List.

Note3: Each Element is allocated 32 bytes of memory space aligned to 32 bytes, representing the STCD. Additionaly, if the configuration is enabled from TRESOS, additional 60 bytes are allocated.

3.6.2 Channel State Machine

3.6.2.1 MCL Driver

The MCL DMA Driver runs based on the presented State Machine Diagram.

The MCL DMA State Machine applies to the MCL DMA Channels and it contains 5 States.

The State Machine supports the following 21 Transitions:

```
T0: Hardware Reset

T1: Mcl_Init() with DMA Channel no Transfer or Scatter/Gather generated configurations

T2: Mcl_DeInit()

T3: Mcl_SetDmaChannelTransferList()

T4: Mcl_Init() with DMA Channel Transfer generated configuration

T5: Mcl_SetDmaChannelScatterGatherList() + Mcl_SetDmaChannelScatterGatherConfig()

T6: Mcl_SetDmaChannelTransferList()

T7: Mcl_SetDmaChannelScatterGatherList() + Mcl_SetDmaChannelScatterGatherConfig()

T8: Mcl_Init() with DMA Channel Scatter/Gather generated configuration

T9: Detection of channel error

T10: Mcl_SetDmaInstanceCommand(CHANNEL, MCL_DMA_CH_ACK_ERROR)

T11: Detection of channel error

T12: Mcl_DeInit()

T13: Mcl_DeInit()

T14: Detection of channel error

T15: Detection of channel error

T16: Mcl_DeInit()

T17: Mcl_DeInit()

T18: Mcl_SetDmaChannelCommand(CHANNEL, MCL_DMA_CH_ACK_ERROR)

T19: Mcl_SetDmaChannelTransferList()

T20: Mcl_SetDmaChannelScatterGatherList() + Mcl_SetDmaChannelScatterGatherConfig()

T21: Detection of channel error
```

When the DMA Driver detects a software or hardware error for the Logic Channel, it shall enter the "MCL_DM↩ A_CH_ERROR_STATE".

To exit from the "MCL_DMA_CH_ERROR_STATE", the application shall use the specified transitions.

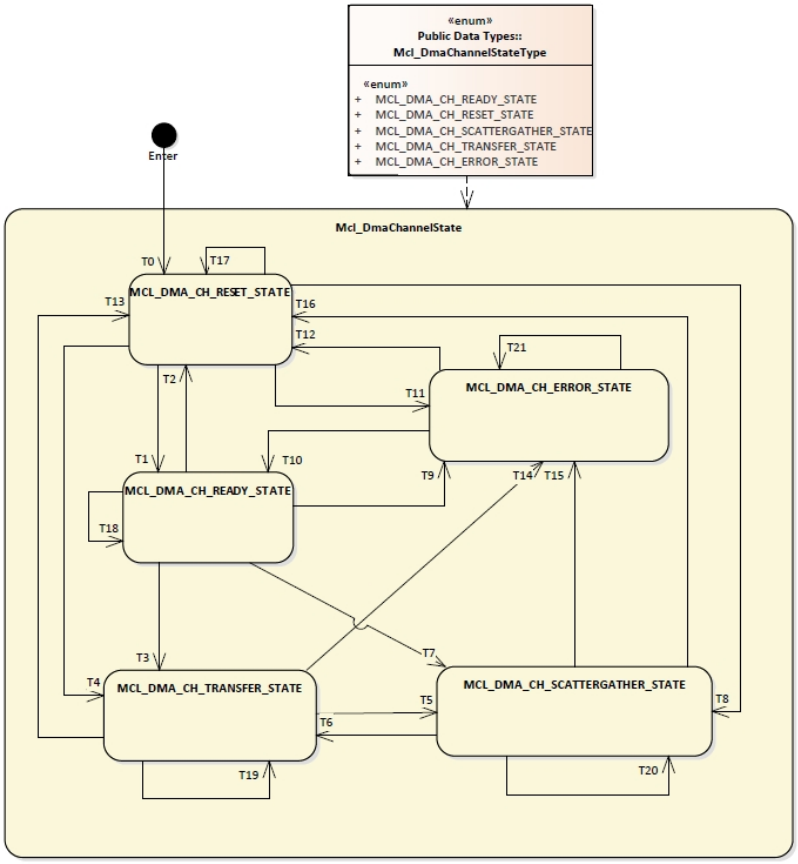

Figure 3.5 MCL DMA Channel State Machine

3.6.2.2 DMA Driver

The DMA IP Driver runs based on the presented State Machine Diagram.

The DMA State Machine applies to the DMA Logic Channels and it contains 5 States.

The State Machine supports the following 21 Transitions:

```
T0: Hardware Reset

T1: Dma_Ip_LogicChannelInit() with no Transfer or Scatter/Gather generated configurations

T2: Dma_Ip_LogicChannelDeinit()

T3: Dma_Ip_SetLogicChannelTransferList()

T4: Dma_Ip_LogicChannelInit() with Transfer generated configuration

T5: Dma_Ip_SetLogicChannelScatterGatherList() + Dma_Ip_SetLogicChannelScatterGatherConfig()

T6: Dma_Ip_SetLogicChannelTransferList()

T7: Dma_Ip_SetLogicChannelScatterGatherList() + Dma_Ip_SetLogicChannelScatterGatherConfig()
```

```
T8: Dma_Ip_LogicChannelInit() with Scatter/Gather generated configuration

T9: Detection of channel error

T10: Dma_Ip_SetLogicChannelCommand(CHANNEL, DMA_IP_CH_CLEAR_ERROR)

T11: Detection of channel error

T12: Dma_Ip_LogicChannelDeinit()

T13: Dma_Ip_LogicChannelDeinit()

T14: Detection of channel error

T15: Detection of channel error

T16: Dma_Ip_LogicChannelDeinit()

T17: Dma_Ip_LogicChannelDeinit()

T18: Dma_Ip_SetLogicChannelCommand(CHANNEL, DMA_IP_CH_CLEAR_ERROR)

T19: Dma_Ip_SetLogicChannelTransferList()

T20: Dma_Ip_SetLogicChannelScatterGatherList() + Dma_Ip_SetLogicChannelScatterGatherConfig()

T21: Detection of channel error
```

When the DMA Driver detects a software or hardware error for the Logic Channel, it shall enter the "DMA_IP_↩ CH_ERROR_STATE".

To exit from the "DMA_IP_CH_ERROR_STATE", the application shall use the specified transitions.

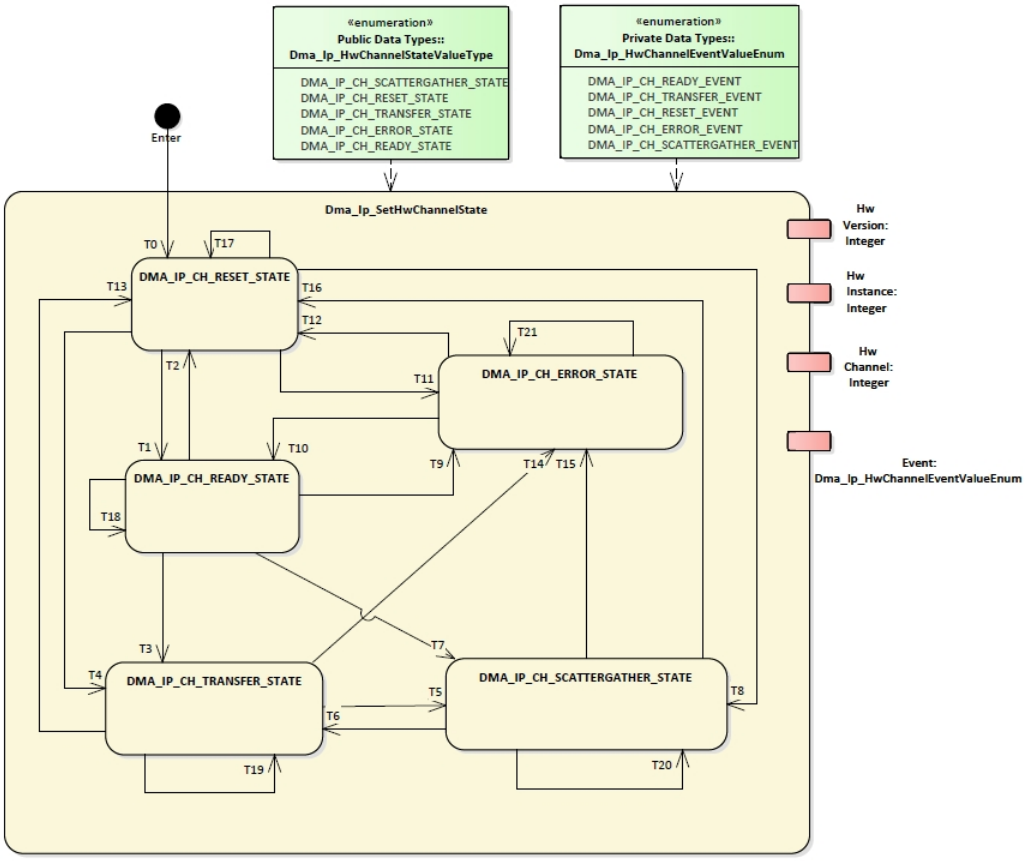

Figure 3.6 DMA IP Channel State Machine

3.7 Runtime errors

The driver generates the following DET errors at runtime.

Function	Error Code	Condition triggering the error
Mcl_Init()	MCL_E_UNINIT	API is called with a NULL pointer as parameter.
Mcl_DeInit()	MCL_E_PARAM_CONFIG	API is called with invalid configuration parameter.
Mcl_DeInit()	MCL_E_UNINIT	API is called with a NULL pointer as parameter.
Mcl_SetDmaInstanceCommand()	MCL_DET_DMA_INSTANCE↩ _COMMAND	API is called with invalid instance command.
Mcl_GetDmaInstanceStatus()	MCL_E_INVALID_INSTANCE	API is called with invalid instance.
Mcl_SetDmaChannelCommand()	MCL_E_INVALID_CHANNEL	API is called with invalid channel.
Mcl_SetDmaChannelCommand()	MCL_E_INVALID_COMMAND	API is called with invalid command.
Mcl_GetDmaChannelStatus()	MCL_E_INVALID_CHANNEL	API is called with invalid channel.
Mcl_SetDmaChannelGlobalList()	MCL_E_INVALID_CHANNEL	API is called with invalid channel.

Function	Error Code	Condition triggering the error
Mcl_SetDmaChannelGlobalList()	MCL_E_INVALID_PARAMETER	API is called with invalid parameter.
Mcl_SetDmaChannelTransferList()	MCL_E_INVALID_CHANNEL	API is called with invalid channel.
Mcl_SetDmaChannelTransferList()	MCL_E_INVALID_PARAMETER	API is called with invalid parameter.
Mcl_SetDmaChannelScatterGatherList()	MCL_E_INVALID_CHANNEL	API is called with invalid channel.
Mcl_SetDmaChannelScatterGatherList()	MCL_E_INVALID_PARAMETER	API is called with invalid parameter.
Mcl_GetDmaChannelParam()	MCL_E_INVALID_CHANNEL	API is called with invalid channel.
Mcl_GetDmaChannelParam()	MCL_E_INVALID_PARAMETER	API is called with invalid parameter.
Mcl_SetDmaChannelScatterGatherConfig()	MCL_E_INVALID_CHANNEL	API is called with invalid channel.
Mcl_SetDmaChannelCrcList()	MCL_E_INVALID_CHANNEL	API is called with invalid channel.
Mcl_CacheEnable()	MCL_E_INVALID_INSTANCE	API is called with invalid instance.
Mcl_CacheDisable()	MCL_E_INVALID_INSTANCE	API is called with invalid instance.
Mcl_CacheInvalidate()	MCL_E_INVALID_INSTANCE	API is called with invalid instance.
Mcl_CacheClean()	MCL_E_INVALID_INSTANCE	API is called with invalid instance.
Mcl_CacheInvalidateByAddr()	MCL_E_INVALID_INSTANCE	API is called with invalid instance.
Mcl_CacheCleanByAddr()	MCL_E_INVALID_INSTANCE	API is called with invalid instance.
Mcl_CacheCleanByAddr()	MCL_E_INVALID_INSTANCE	API is called with invalid instance.

3.8 Symbolic Names Disclaimer

All containers having symbolicNameValue set to TRUE in the AUTOSAR schema will generate defines like:

#define <Mip>Conf_<Container_ShortName>_<Container_ID>

For this reason it is forbidden to duplicate the names of such containers across the RTD configurations or to use names that may trigger other compile issues (e.g. match existing `#ifdefs` arguments).

Chapter 4

Tresos Configuration Plug-in

This chapter describes the Tresos configuration plug-in for the driver. All the parameters are described below.

- Module Mcl
  - Container MclGeneral
    - Parameter MclEnableDemReportErrorStatus
    - Parameter MclEnableDevErrorDetect
    - Parameter Mcl_VersionInfoApi
    - Parameter MclEnableUserModeSupport
    - Parameter MclEnableVirtualAddressMappingSupport
    - Container MclDma
      - Parameter MclEnableDma
    - Container MclCache
      - Parameter MclEnableCache
      - Parameter MclCacheTimeoutValue
      - Parameter MclCacheTimeoutMethod
    - Container MclTrgMux
      - Parameter MclEnableTrgMux
    - Container MclFlexioCommon
      - Parameter MclEnableFlexioCommon
    - Container MclFtmCommon
      - Parameter Mcl_FtmCommonTimebase
  - Container MclConfig
    - Container MclVirtualMemorySection
      - Parameter MclVirtualAddressStart
      - Parameter MclVirtualAddressEnd
      - Parameter MclPhysicalAddressStart
      - Parameter MclPhysicalAddressEnd
    - Container MclDemEventParameterRefs
      - Reference MCL_E_TIMEOUT_FAILURE
    - Container dmaLogicInstance_ConfigType
      - Parameter dmaLogicInstance_IdName
      - Parameter dmaLogicInstance_hwId

- Parameter dmaLogicInstance_enDebug
- Parameter dmaLogicInstance_enRoundRobin
- Parameter dmaLogicInstance_enHaltAfterError
- Parameter dmaLogicInstance_enChLinking

* Container dmaLogicChannel_Type
  - Parameter dmaLogicChannel_LogicName
  - Parameter dmaLogicChannel_HwInstId
  - Parameter dmaLogicChannel_HwChId
  - Parameter dmaLogicChannel_InterruptCallback
  - Parameter dmaLogicChannel_ErrorInterruptCallback
  - Parameter dmaLogicChannel_EnableGlobalConfig
  - Parameter dmaLogicChannel_EnableTransferConfig
  - Parameter dmaLogicChannel_EnableScatterGather
  - Container dmaLogicChannel_ConfigType
  - Container dmaLogicChannel_GlobalConfigType
  - Container dmaLogicChannelConfig_GlobalRequestType
  - Parameter dmaGlobalRequest_enDmaRequest
  - Container dmaLogicChannelConfig_GlobalInterruptType
  - Parameter dmaGlobalInterrupt_enDmaErrorInterrupt
  - Container dmaLogicChannelConfig_GlobalPriorityType
  - Parameter dmaGlobalPriority_LevelPriority
  - Parameter dmaGlobalPriority_enPreemption
  - Parameter dmaGlobalPriority_disPreempt
  - Container dmaLogicChannel_TransferConfigType
  - Container dmaLogicChannelConfig_TransferControlType
  - Parameter dmaLogicChannelConfig_enDmaMajorInterrupt
  - Parameter dmaLogicChannelConfig_enDmaHalfMajorInterrupt
  - Parameter dmaLogicChannelConfig_disDmaAutoHwReq
  - Parameter dmaLogicChannelConfig_bandwidthControl
  - Parameter dmaLogicChannelConfig_ScatterGatherAddressType
  - Parameter dmaLogicChannelConfig_DestinationStoreAddressType
  - Container dmaLogicChannelConfig_TransferSourceType
  - Parameter dmaLogicChannelConfig_SourceAddressType
  - Parameter dmaLogicChannelConfig_SourceSignedOffsetType
  - Parameter dmaLogicChannelConfig_SourceLastAddressAdjustmentType
  - Parameter dmaTransferConfig_TransferSizeType
  - Parameter dmaLogicChannelConfig_SourceModuloType
  - Container dmaLogicChannelConfig_TransferDestinationType
  - Parameter dmaLogicChannelConfig_DestinationAddressType
  - Parameter dmaLogicChannelConfig_DestinationSignedOffsetType
  - Parameter dmaLogicChannelConfig_DestinationLastAddressAdjustmentType
  - Parameter dmaTransferConfig_TransferSizeType
  - Parameter dmaLogicChannelConfig_DestinationModuloType
  - Container dmaLogicChannelConfig_TransferMinorLoopType
  - Parameter dmaLogicChannelConfig_enSourceOffset
  - Parameter dmaLogicChannelConfig_enDestinationOffset
  - Parameter dmaLogicChannelConfig_OffsetValueType

- Parameter dmaLogicChannelConfig_enMinorLoopLinkCh
- Parameter dmaLogicChannelConfig_MinorLoopSizeType
- Reference dynamic_dmaLogicChannelConfig_MinorLoopLinkChValueType
- Container dmaLogicChannelConfig_TransferMajorLoopType
- Parameter dmaLogicChannelConfig_enMajorLoopLinkCh
- Parameter dmaLogicChannelConfig_MajorLoopCountType
- Reference dynamic_dmaLogicChannelConfig_MajorLoopLinkChValueType
- Container dmaLogicChannel_ScatterGatherConfigType
- Container dmaLogicChannelConfig_ScatterGatherArrayType
- Container dmaLogicChannelConfig_ScatterGatherElementConfigType
- Parameter dmaLogicChannelConfig_ScatterGatherElementNameType
- Parameter dmaLogicChannelConfig_LastElementLink_ScatterGatherType
- Parameter dmaLogicChannelConfig_enScatterGatherConfig
- Reference dynamic_dmaLogicChannelConfig_BasicElementLink_ScatterGatherType
- Container dmaLogicChannelConfig_TransferControlType
- Parameter dmaLogicChannelConfig_enStart
- Parameter dmaLogicChannelConfig_enDmaMajorInterrupt
- Parameter dmaLogicChannelConfig_enDmaHalfMajorInterrupt
- Parameter dmaLogicChannelConfig_disDmaAutoHwReq
- Parameter dmaLogicChannelConfig_bandwidthControl
- Parameter dmaLogicChannelConfig_ScatterGatherAddressType
- Parameter dmaLogicChannelConfig_DestinationStoreAddressType
- Container dmaLogicChannelConfig_TransferSourceType
- Parameter dmaLogicChannelConfig_SourceAddressType
- Parameter dmaLogicChannelConfig_SourceSignedOffsetType
- Parameter dmaLogicChannelConfig_SourceLastAddressAdjustmentType
- Parameter dmaTransferConfig_TransferSizeType
- Parameter dmaLogicChannelConfig_SourceModuloType
- Container dmaLogicChannelConfig_TransferDestinationType
- Parameter dmaLogicChannelConfig_DestinationAddressType
- Parameter dmaLogicChannelConfig_DestinationSignedOffsetType
- Parameter dmaLogicChannelConfig_DestinationLastAddressAdjustmentType
- Parameter dmaTransferConfig_TransferSizeType
- Parameter dmaLogicChannelConfig_DestinationModuloType
- Container dmaLogicChannelConfig_TransferMinorLoopType
- Parameter dmaLogicChannelConfig_enSourceOffset
- Parameter dmaLogicChannelConfig_enDestinationOffset
- Parameter dmaLogicChannelConfig_OffsetValueType
- Parameter dmaLogicChannelConfig_enMinorLoopLinkCh
- Parameter dmaLogicChannelConfig_MinorLoopSizeType
- Reference dynamic_dmaLogicChannelConfig_MinorLoopLinkChValueType
- Container dmaLogicChannelConfig_TransferMajorLoopType
- Parameter dmaLogicChannelConfig_enMajorLoopLinkCh
- Parameter dmaLogicChannelConfig_MajorLoopCountType
- Reference dynamic_dmaLogicChannelConfig_MajorLoopLinkChValueType

* Container trgmuxInstaceList
  - Parameter trgmuxHardwareInstance

    - Container trgmuxLogicGroup
        - Parameter trgmuxLogicGroupHardwareInstance
        - Parameter trgmuxLogicGroup_Name
        - Parameter trgmuxLogicGroup_Lock
        - Container trgmuxLogicTrigger
        - Parameter trgmuxLogicTrigger_Name
        - Parameter trgmuxLogicTrigger_Output
        - Parameter trgmuxLogicTrigger_Input
    - Container FlexioCommon
        - Parameter FlexioMclInstances
        - Parameter FlexioDebugEnable
        - Container FlexioMclLogicChannels
        - Parameter FlexioMclChannelId
        - Parameter FlexioMclPinId
        - Parameter FlexioMclAddPinEnable
        - Parameter FlexioMclAddPinId
        - Parameter FlexioMclAddChannelEnable
        - Parameter FlexioMclAddChannelId
- Container CommonPublishedInformation
    - Parameter ArReleaseMajorVersion
    - Parameter ArReleaseMinorVersion
    - Parameter ArReleaseRevisionVersion
    - Parameter ModuleId
    - Parameter SwMajorVersion
    - Parameter SwMinorVersion
    - Parameter SwPatchVersion
    - Parameter VendorId

4.1 Module Mcl

Vendor specific: Configuration of the Mcl (MicroController Library) module.

Included containers:

- MclGeneral
- MclConfig
- CommonPublishedInformation

Property	Value
type	ECUC-MODULE-DEF
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantSupport	true
supportedConfigVariants	VARIANT-POST-BUILD, VARIANT-PRE-COMPILE

4.2 Container MclGeneral

Vendor specific: Configuration of general Mcl parameters.

Included subcontainers:

- MclDma
- MclCache
- MclTrgMux
- MclFlexioCommon
- MclFtmCommon

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

4.3 Parameter MclEnableDemReportErrorStatus

Enable/Disable the Production Error Reporting (DEM).

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	false

4.4 Parameter MclEnableDevErrorDetect

Vendor specific:

Enable/Disable the Development Error Detection (DET).

true: Enabled.

false: Disabled.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	false

4.5 Parameter Mcl_VersionInfoApi

Vendor specific: Enables/Disables the get version info API function

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	false

4.6 Parameter MclEnableUserModeSupport

When this parameter is enabled, the MCL module will adapt to run from User Mode, with the following measures:

b) using 'call trusted function' stubs for all internal function calls that access registers requiring supervisor mode.

for more information, please see chapter 5.7 User Mode Support in IM

Note: Implementation Specific Parameter.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	false

4.7 Parameter MclEnableVirtualAddressMappingSupport

Enable/Disable Virtual Address Mapping support

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	false

4.8 Container MclDma

Vendor specific:

Container for the Dma related configuration parameters.

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

4.9 Parameter MclEnableDma

Vendor specific: Enable/Disable DMA support.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	false

4.10 Container MclCache

Vendor specific:

Container for the CACHE related configuration parameters.

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

4.11 Parameter MclEnableCache

Vendor specific:

Enable/Disable all CACHE support.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	false

4.12 Parameter MclCacheTimeoutValue

Set Cache timeout value

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	4294967295
min	0

4.13 Parameter MclCacheTimeoutMethod

MclCacheTimeoutMethod

Configures the timeout method.

Based on this selection a certain timeout method from OsIf will be used in the driver.

Note: If OSIF_COUNTER_SYSTEM or OSIF_COUNTER_CUSTOM are selected make sure the corresponding timer is enabled in OsIf General configuration.

Note: Implementation Specific Parameter.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	OSIF_COUNTER_DUMMY
literals	['OSIF_COUNTER_DUMMY', 'OSIF_COUNTER_SYSTEM', 'OSIF_COU↩ NTER_CUSTOM']

4.14 Container MclTrgMux

Vendor specific:

Container for the TRGMUX related configuration parameters.

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

4.15 Parameter MclEnableTrgMux

Vendor specific: Enable/Disable TRGMUX support.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	false

4.16 Container MclFlexioCommon

Vendor specific:

Container for the Flexio Common related configuration parameters.

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

4.17 Parameter MclEnableFlexioCommon

Vendor specific: Enable/Disable Flexio common support.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF

Property	Value
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	false

4.18 Container MclFtmCommon

Vendor specific:

Container for the FTM Common related configuration parameters.

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

4.19 Parameter Mcl_FtmCommonTimebase

Enables/Disables the option to set the a common timebase for multiple FTM modules.

Note:

NoteThis is an Implementation Specific Parameter. Enabling this feature will allow the use of the Mcl_SelectCommonTimebase API

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP

Property	Value
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	false

4.20 Container MclConfig

Vendor specific: This container is the base for a multiple configuration set

Included subcontainers:

- MclVirtualMemorySection
- MclDemEventParameterRefs
- dmaLogicInstance_ConfigType
- dmaLogicChannel_Type
- trgmuxInstaceList
- trgmuxLogicGroup
- FlexioCommon

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

4.21 Container MclVirtualMemorySection

Vendor specific:

Data to configure Virtual address and Physical address.

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	0
upperMultiplicity	Infinite
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE

4.22 Parameter MclVirtualAddressStart

This parameter represents the Virtual Address Start.

The address must be aligned to 4 bytes.

Note: Implementation Specific Parameter.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	4294967295
min	0

4.23 Parameter MclVirtualAddressEnd

This parameter represents the Virtual Address End.

The address must be aligned to 4 bytes.

Note: Implementation Specific Parameter.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	255
max	4294967295
min	0

4.24 Parameter MclPhysicalAddressStart

This parameter represents the Physical Address Start.

The address must be aligned to 4 bytes.

Note: Implementation Specific Parameter.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	4294967295
min	0

4.25 Parameter MclPhysicalAddressEnd

This parameter represents the Physical Address End.

The address must be aligned to 4 bytes.

Note: Implementation Specific Parameter.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	255
max	4294967295
min	0

4.26 Container MclDemEventParameterRefs

Container for the references to DemEventParameter elements which shall be invoked using the API Dem_ReportErrorStatus API in case the corresponding error occurs.

The EventId is taken from the referenced DemEventParameter's DemEventId value.

The standardized errors are provided in the container and can be extended by vendor specific error references.

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	0
upperMultiplicity	1
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE

4.27 Reference MCL_E_TIMEOUT_FAILURE

Reference to configured DEM event to report Timeout failure.

Property	Value
type	ECUC-REFERENCE-DEF
origin	NXP
lowerMultiplicity	0
upperMultiplicity	1
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
postBuildVariantValue	false
valueConfigClasses	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
requiresSymbolicNameValue	true
destination	/AUTOSAR/EcucDefs/Dem/DemConfigSet/DemEventParameter

4.28 Container dmaLogicInstance_ConfigType

Vendor specific: Configuration of a DMA Instance.

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	0
upperMultiplicity	2
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE

4.29 Parameter dmaLogicInstance_IdName

Vendor specific:

Logic Instance Name.

Property	Value
type	ECUC-STRING-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	DMA_LOGIC_INST_0

4.30 Parameter dmaLogicInstance_hwId

Vendor specific:

Select the Hardware DMA Instance.

NOTE: This is an Implementation Specific Parameter.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	DMA_IP_HW_INST_0
literals	['DMA_IP_HW_INST_0']

4.31 Parameter dmaLogicInstance_enDebug

Vendor specific:

DMA_CR[EDBG].

Enable Debug.

0 - The assertion of the system debug control input is ignored.

1 - The assertion of the system debug control input causes the eDMA to stall the start of a new channel.

Executing channels are allowed to complete.

Channel execution will resume when either the system debug control input is negated or the EDBG bit is cleared.

Note: Implementation Specific Parameter.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	false

4.32 Parameter dmaLogicInstance_enRoundRobin

Vendor specific:

DMA_CR[ERCA].

Enable Round Robin Channel Arbitration.

0 - Fixed-priority arbitration is used for channel selection within each group.

1 - Round-Robin arbitration is used for channel selection within each group.

Note: Implementation Specific Parameter.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A

Property	Value
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	false

4.33 Parameter dmaLogicInstance_enHaltAfterError

Vendor specific:

DMA_CR[HOE] for eDMA2 instances or DMA_CSR[HAE] for eDMA3 instances.

Halt On/After Error.

0 - Normal operation.

1 - Any error will cause the HALT bit to be set.

Subsequently, all service requests will be ignored until the HALT bit is cleared.

Note: Implementation Specific Parameter.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	false

4.34 Parameter dmaLogicInstance_enChLinking

Vendor specific:

DMA_CR[GCLC].

Global Channel Linking Control.

0 - Channel linking is disabled for all channels.

1 - Channel linking is available and controlled by each channel's link settings.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	false

4.35 Container dmaLogicChannel_Type

Vendor specific:

Logic Channel Configuration.

Included subcontainers:

- dmaLogicChannel_ConfigType

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	0
upperMultiplicity	16
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE

4.36 Parameter dmaLogicChannel_LogicName

Vendor specific:

Logic Channel Name.

Property	Value
type	ECUC-STRING-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	DMA_LOGIC_CH_0

4.37 Parameter dmaLogicChannel_HwInstId

Vendor specific:

Select the Hardware DMA Instance.

NOTE: This is an Implementation Specific Parameter.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	DMA_IP_HW_INST_0
literals	['DMA_IP_HW_INST_0']

4.38 Parameter dmaLogicChannel_HwChId

Vendor specific:

Select the physical eDMA Channel.

NOTE: This is an Implementation Specific Parameter.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	DMA_IP_HW_CH_0
literals	['DMA_IP_HW_CH_0', 'DMA_IP_HW_CH_1', 'DMA_IP_HW_CH_↩ 2', 'DMA_IP_HW_CH_3', 'DMA_IP_HW_CH_4', 'DMA_IP_HW_CH_↩ 5', 'DMA_IP_HW_CH_6', 'DMA_IP_HW_CH_7', 'DMA_IP_HW_CH_↩ 8', 'DMA_IP_HW_CH_9', 'DMA_IP_HW_CH_10', 'DMA_IP_HW_CH_↩ 11', 'DMA_IP_HW_CH_12', 'DMA_IP_HW_CH_13', 'DMA_IP_HW_C↩ H_14', 'DMA_IP_HW_CH_15']

4.39 Parameter dmaLogicChannel_InterruptCallback

Vendor specific:

User callback function to report that the transfer is half or complete depending on configuration.

NOTE: Use NULL_PTR w/o quotes. If the used string is different from NULL_PTR it will be used as the configured function name.

Property	Value
type	ECUC-FUNCTION-NAME-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	NULL_PTR

4.40 Parameter dmaLogicChannel_ErrorInterruptCallback

Vendor specific:

User callback function

NOTE: Use NULL_PTR w/o quotes. If the used string is different from NULL_PTR it will be used as the configured function name.

Property	Value
type	ECUC-FUNCTION-NAME-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	NULL_PTR

4.41 Parameter dmaLogicChannel_EnableGlobalConfig

Vendor specific: Enable and allocate memory for Global Configuration. Global Configuration can be configured during generation time or during runtime.

Note: If the check box is enabled, then memory space is allocated for the configuration.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	false

4.42 Parameter dmaLogicChannel_EnableTransferConfig

Vendor specific: Enable and allocate memory for Transfer Configuration. Transfer Configuration can be configured during generation time or during runtime.

Note: If the check box is enabled, then memory space is allocated for the configuration.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	false

4.43 Parameter dmaLogicChannel_EnableScatterGather

Vendor specific: Enable and allocate memory for ScatterGather Transfer Mode.

The ScatterGather Transfer Mode shall allocate memory for each Element, comprised of: Element Linkage and Element Software TCD.

The Element allocation can be done only in the configurator.

The Element Configuration can be further enabled for each individual element.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	false

4.44 Container dmaLogicChannel_ConfigType

Vendor specific: Logic Channel Configuration.

Included subcontainers:

- dmaLogicChannel_GlobalConfigType
- dmaLogicChannel_TransferConfigType
- dmaLogicChannel_ScatterGatherConfigType

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

4.45 Container dmaLogicChannel_GlobalConfigType

Vendor specific:

Logic Channel Global Configuration.

Included subcontainers:

- dmaLogicChannelConfig_GlobalRequestType
- dmaLogicChannelConfig_GlobalInterruptType
- dmaLogicChannelConfig_GlobalPriorityType

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

4.46 Container dmaLogicChannelConfig_GlobalRequestType

Vendor specific:

TCD Request Control.

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

4.47 Parameter dmaGlobalRequest_enDmaRequest

Vendor specific: Enable the Dma Channel Hardware Request.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	false

4.48 Container dmaLogicChannelConfig_GlobalInterruptType

Vendor specific:

TCD Global Interrupt.

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

4.49 Parameter dmaGlobalInterrupt_enDmaErrorInterrupt

Vendor specific: Enable the Dma Channel Error Interrupt.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	false

4.50 Container dmaLogicChannelConfig_GlobalPriorityType

Vendor specific:

TCD Global Interrupt.

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

4.51 Parameter dmaGlobalPriority_LevelPriority

Vendor specific: Set the Dma Channel Level Priority.

Note:

- If you want to change this value, you must Enable Global Config on "Logic Channel Configuration" tag.

- When have larger than one configured channel, the selected priority shall be from the pool of the configured channels, maintaining priority uniqueness. Ex: the user configures 3 channels: 0, 5, 15, by default the channel priority: 0-0, 5-5, 15-15. But the user wants to changes their channel priorities, the value only is selected one of 3 values(0, 5, 15) and they must be unique.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	DMA_IP_LEVEL_PRIO0
literals	['DMA_IP_LEVEL_PRIO0', 'DMA_IP_LEVEL_PRIO1', 'DMA_IP_LEV↩ EL_PRIO2', 'DMA_IP_LEVEL_PRIO3', 'DMA_IP_LEVEL_PRIO4', 'DM↩ A_IP_LEVEL_PRIO5', 'DMA_IP_LEVEL_PRIO6', 'DMA_IP_LEVEL_P↩ RIO7', 'DMA_IP_LEVEL_PRIO8', 'DMA_IP_LEVEL_PRIO9', 'DMA_IP↩ _LEVEL_PRIO10', 'DMA_IP_LEVEL_PRIO11', 'DMA_IP_LEVEL_PRI↩ O12', 'DMA_IP_LEVEL_PRIO13', 'DMA_IP_LEVEL_PRIO14', 'DMA_I↩ P_LEVEL_PRIO15']

4.52 Parameter dmaGlobalPriority_enPreemption

Vendor specific: Enable the Dma Channel Preemption.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

Property	Value
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	false

4.53 Parameter dmaGlobalPriority_disPreempt

Vendor specific: Disable the Dma Channel Preempt.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	false

4.54 Container dmaLogicChannel_TransferConfigType

Vendor specific:

Logic Channel Transfer Configuration.

Included subcontainers:

- dmaLogicChannelConfig_TransferControlType
- dmaLogicChannelConfig_TransferSourceType
- dmaLogicChannelConfig_TransferDestinationType
- dmaLogicChannelConfig_TransferMinorLoopType
- dmaLogicChannelConfig_TransferMajorLoopType

Property	Value

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

4.55 Container dmaLogicChannelConfig_TransferControlType

Vendor specific:

TCD Global Interrupt.

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

4.56 Parameter dmaLogicChannelConfig_enDmaMajorInterrupt

Vendor specific: Enable the Dma Channel major interrupt.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true

Property	Value
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	false

4.57 Parameter dmaLogicChannelConfig_enDmaHalfMajorInterrupt

Vendor specific: Enable the Dma Channel half major interrupt.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	false

4.58 Parameter dmaLogicChannelConfig_disDmaAutoHwReq

Vendor specific: Disable the Dma Channel automatic request.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	false

4.59 Parameter dmaLogicChannelConfig_bandwidthControl

Vendor specific: Set the Dma Channel bandwidth control.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	DMA_IP_BWC_ENGINE_NO_STALL
literals	['DMA_IP_BWC_ENGINE_NO_STALL', 'DMA_IP_BWC_ENGINE_4C↩ YCLE_STALL', 'DMA_IP_BWC_ENGINE_8CYCLE_STALL']

4.60 Parameter dmaLogicChannelConfig_ScatterGatherAddressType

Vendor specific: ScatterGather Address

Property	Value
type	ECUC-STRING-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0U

4.61 Parameter dmaLogicChannelConfig_DestinationStoreAddressType

Vendor specific: Destination Store Address

Property	Value
type	ECUC-STRING-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0U

4.62 Container dmaLogicChannelConfig_TransferSourceType

Vendor specific: Transfer Source

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

4.63 Parameter dmaLogicChannelConfig_SourceAddressType

Vendor specific: Source Address

Property	Value
type	ECUC-STRING-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A

Property	Value
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0

4.64 Parameter dmaLogicChannelConfig_SourceSignedOffsetType

Vendor specific: Set the Dma Channel source signed offset value.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	32767
min	-32767

4.65 Parameter dmaLogicChannelConfig_SourceLastAddressAdjustmentType

Vendor specific: Set the Dma Channel source signed last address adjustment.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true

Property	Value
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	2147483647
min	-2147483647

4.66 Parameter dmaTransferConfig_TransferSizeType

Vendor specific: Set the Dma Channel source transfer size.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	DMA_IP_TRANSFER_SIZE_1_BYTE
literals	['DMA_IP_TRANSFER_SIZE_1_BYTE', 'DMA_IP_TRANSFER_SIZE_↩ 2_BYTE', 'DMA_IP_TRANSFER_SIZE_4_BYTE', 'DMA_IP_TRANSFE↩ R_SIZE_16_BYTE', 'DMA_IP_TRANSFER_SIZE_32_BYTE']

4.67 Parameter dmaLogicChannelConfig_SourceModuloType

Vendor specific: Set the Dma Channel source modulo.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE

Property	Value
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	31
min	0

4.68 Container dmaLogicChannelConfig_TransferDestinationType

Vendor specific: Transfer Destination

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

4.69 Parameter dmaLogicChannelConfig_DestinationAddressType

Vendor specific: Destination Address

Property	Value
type	ECUC-STRING-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0

4.70 Parameter dmaLogicChannelConfig_DestinationSignedOffsetType

Vendor specific: Set the Dma Channel destination signed offset value.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	32767
min	-32767

4.71 Parameter dmaLogicChannelConfig_DestinationLastAddressAdjustmentType

Vendor specific: Set the Dma Channel destination signed last address adjustment.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	2147483647
min	-2147483647

4.72 Parameter dmaTransferConfig_TransferSizeType

Vendor specific: Set the Dma Channel destination transfer size.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	DMA_IP_TRANSFER_SIZE_1_BYTE
literals	['DMA_IP_TRANSFER_SIZE_1_BYTE', 'DMA_IP_TRANSFER_SIZE_↩ 2_BYTE', 'DMA_IP_TRANSFER_SIZE_4_BYTE', 'DMA_IP_TRANSFE↩ R_SIZE_16_BYTE', 'DMA_IP_TRANSFER_SIZE_32_BYTE']

4.73 Parameter dmaLogicChannelConfig_DestinationModuloType

Vendor specific: Set the Dma Channel destination modulo.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	31
min	0

4.74 Container dmaLogicChannelConfig_TransferMinorLoopType

Vendor specific: Transfer Minor Loop

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

4.75 Parameter dmaLogicChannelConfig_enSourceOffset

Vendor specific: Enable the Dma Channel minor loop source offset.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	false

4.76 Parameter dmaLogicChannelConfig_enDestinationOffset

Vendor specific: Enable the Dma Channel minor loop destination offset.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP

Property	Value
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	false

4.77 Parameter dmaLogicChannelConfig_OffsetValueType

Vendor specific: Set the Dma Channel minor loop signed offset.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	1048575
min	-1048575

4.78 Parameter dmaLogicChannelConfig_enMinorLoopLinkCh

Vendor specific: Enable the Dma Channel minor loop logic channel linking.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1

Property	Value
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	false

4.79 Parameter dmaLogicChannelConfig_MinorLoopSizeType

Vendor specific: Set the Dma Channel minor loop transfer size.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	1073741823
min	0

4.80 Reference dynamic_dmaLogicChannelConfig_MinorLoopLink↩ ChValueType

Vendor specific: Set the Dma Channel minor loop logic channel link.

Property	Value
type	ECUC-REFERENCE-DEF
origin	NXP
lowerMultiplicity	0
upperMultiplicity	1
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE

Property	Value
postBuildVariantValue	false
valueConfigClasses	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
requiresSymbolicNameValue	true
destination	/AUTOSAR/EcucDefs/Mcl/MclConfig/dmaLogicChannel_Type

4.81 Container dmaLogicChannelConfig_TransferMajorLoopType

Vendor specific: Transfer Major Loop

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

4.82 Parameter dmaLogicChannelConfig_enMajorLoopLinkCh

Vendor specific: Enable the Dma Channel major loop logic channel linking.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	false

4.83 Parameter dmaLogicChannelConfig_MajorLoopCountType

Vendor specific: Sets the Dma Channel major loop count.

If minor loop channel linking is enabled, the major loop count limit is between 0 and 511.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	1
max	32767
min	0

4.84 Reference dynamic_dmaLogicChannelConfig_MajorLoopLink↩ ChValueType

Vendor specific: Set the Dma Channel major loop logic channel link.

Property	Value
type	ECUC-REFERENCE-DEF
origin	NXP
lowerMultiplicity	0
upperMultiplicity	1
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
postBuildVariantValue	false
valueConfigClasses	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
requiresSymbolicNameValue	true
destination	/AUTOSAR/EcucDefs/Mcl/MclConfig/dmaLogicChannel_Type

4.85 Container dmaLogicChannel_ScatterGatherConfigType

Vendor specific:

Logic Channel ScatterGather Configuration.

Included subcontainers:

- dmaLogicChannelConfig_ScatterGatherArrayType

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

4.86 Container dmaLogicChannelConfig_ScatterGatherArrayType

Vendor specific: Logic Channel ScatterGather Configuration.

Included subcontainers:

- dmaLogicChannelConfig_ScatterGatherElementConfigType

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	0
upperMultiplicity	256
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD

4.87 Container dmaLogicChannelConfig_ScatterGatherElementConfigType

Vendor specific: Element

Included subcontainers:

- dmaLogicChannelConfig_TransferControlType
- dmaLogicChannelConfig_TransferSourceType
- dmaLogicChannelConfig_TransferDestinationType
- dmaLogicChannelConfig_TransferMinorLoopType
- dmaLogicChannelConfig_TransferMajorLoopType

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

4.88 Parameter dmaLogicChannelConfig_ScatterGatherElementNameType

Vendor specific: Element Name

Property	Value
type	ECUC-STRING-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0_SGA_ELEMENT_0

4.89 Parameter dmaLogicChannelConfig_LastElementLink_ScatterGatherType

Vendor specific: For non-circular lists, the last element shall have this checkbox set.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF

Property	Value
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	false

4.90 Parameter dmaLogicChannelConfig_enScatterGatherConfig

Vendor specific: Enable Scatter/Gather Configuration

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	false

4.91 Reference dynamic_dmaLogicChannelConfig_BasicElementLink_ScatterGatherType

Vendor specific: Element Link. Elements shall be part of the same Logic Channel.

Property	Value
type	ECUC-REFERENCE-DEF
origin	NXP
lowerMultiplicity	0
upperMultiplicity	1
postBuildVariantMultiplicity	false

Property	Value
multiplicityConfigClasses	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
postBuildVariantValue	false
valueConfigClasses	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
requiresSymbolicNameValue	true
destination	/AUTOSAR/EcucDefs/Mcl/MclConfig/dmaLogicChannel_Type/dmaLogic↩ Channel_ConfigType/dmaLogicChannel_ScatterGatherConfigType/dma↩ LogicChannelConfig_ScatterGatherArrayType

4.92 Container dmaLogicChannelConfig_TransferControlType

Vendor specific:

TCD Global Interrupt.

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

4.93 Parameter dmaLogicChannelConfig_enStart

Vendor specific: Enable the Dma Channel start service request (software request).

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

Property	Value
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	false

4.94 Parameter dmaLogicChannelConfig_enDmaMajorInterrupt

Vendor specific: Enable the Dma Channel major interrupt.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	false

4.95 Parameter dmaLogicChannelConfig_enDmaHalfMajorInterrupt

Vendor specific: Enable the Dma Channel half major interrupt.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	false

4.96 Parameter dmaLogicChannelConfig_disDmaAutoHwReq

Vendor specific: Disable the Dma Channel automatic request.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	false

4.97 Parameter dmaLogicChannelConfig_bandwidthControl

Vendor specific: Set the Dma Channel bandwidth control.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	DMA_IP_BWC_ENGINE_NO_STALL
literals	['DMA_IP_BWC_ENGINE_NO_STALL', 'DMA_IP_BWC_ENGINE_4C↩YCLE_STALL', 'DMA_IP_BWC_ENGINE_8CYCLE_STALL']

4.98 Parameter dmaLogicChannelConfig_ScatterGatherAddressType

Vendor specific: ScatterGather Address

Property	Value
type	ECUC-STRING-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0U

4.99 Parameter dmaLogicChannelConfig_DestinationStoreAddressType

Vendor specific: Destination Store Address

Property	Value
type	ECUC-STRING-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0U

4.100 Container dmaLogicChannelConfig_TransferSourceType

Vendor specific: Transfer Source

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

4.101 Parameter dmaLogicChannelConfig_SourceAddressType

Vendor specific: Source Address

Property	Value
type	ECUC-STRING-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0

4.102 Parameter dmaLogicChannelConfig_SourceSignedOffsetType

Vendor specific: Set the Dma Channel source signed offset value.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0

Property	Value
max	32767
min	-32767

4.103 Parameter dmaLogicChannelConfig_SourceLastAddressAdjustmentType

Vendor specific: Set the Dma Channel source signed last address adjustment.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	2147483647
min	-2147483647

4.104 Parameter dmaTransferConfig_TransferSizeType

Vendor specific: Set the Dma Channel source transfer size.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	DMA_IP_TRANSFER_SIZE_1_BYTE

Property	Value
literals	['DMA_IP_TRANSFER_SIZE_1_BYTE', 'DMA_IP_TRANSFER_SIZE_↩ 2_BYTE', 'DMA_IP_TRANSFER_SIZE_4_BYTE', 'DMA_IP_TRANSFE↩ R_SIZE_16_BYTE', 'DMA_IP_TRANSFER_SIZE_32_BYTE']

4.105 Parameter dmaLogicChannelConfig_SourceModuloType

Vendor specific: Set the Dma Channel source modulo.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	31
min	0

4.106 Container dmaLogicChannelConfig_TransferDestinationType

Vendor specific: Transfer Destination

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

4.107 Parameter dmaLogicChannelConfig_DestinationAddressType

Vendor specific: Destination Address

Property	Value
type	ECUC-STRING-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0

4.108 Parameter dmaLogicChannelConfig_DestinationSignedOffsetType

Vendor specific: Set the Dma Channel destination signed offset value.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	32767
min	-32767

4.109 Parameter dmaLogicChannelConfig_DestinationLastAddressAdjustmentType

Vendor specific: Set the Dma Channel destination signed last address adjustment.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	2147483647
min	-2147483647

4.110 Parameter dmaTransferConfig_TransferSizeType

Vendor specific: Set the Dma Channel destination transfer size.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	DMA_IP_TRANSFER_SIZE_1_BYTE
literals	['DMA_IP_TRANSFER_SIZE_1_BYTE', 'DMA_IP_TRANSFER_SIZE_↩ 2_BYTE', 'DMA_IP_TRANSFER_SIZE_4_BYTE', 'DMA_IP_TRANSFE↩ R_SIZE_16_BYTE', 'DMA_IP_TRANSFER_SIZE_32_BYTE']

4.111 Parameter dmaLogicChannelConfig_DestinationModuloType

Vendor specific: Set the Dma Channel destination modulo.

Property	Value
type	ECUC-INTEGER-PARAM-DEF

Property	Value
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	31
min	0

4.112 Container dmaLogicChannelConfig_TransferMinorLoopType

Vendor specific: Transfer Minor Loop

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

4.113 Parameter dmaLogicChannelConfig_enSourceOffset

Vendor specific: Enable the Dma Channel minor loop source offset.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A

Property	Value
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	false

4.114 Parameter dmaLogicChannelConfig_enDestinationOffset

Vendor specific: Enable the Dma Channel minor loop destination offset.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	false

4.115 Parameter dmaLogicChannelConfig_OffsetValueType

Vendor specific: Set the Dma Channel minor loop signed offset.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	1048575
min	-1048575

4.116 Parameter dmaLogicChannelConfig_enMinorLoopLinkCh

Vendor specific: Enable the Dma Channel minor loop logic channel linking.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	false

4.117 Parameter dmaLogicChannelConfig_MinorLoopSizeType

Vendor specific: Set the Dma Channel minor loop transfer size.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	0
max	1073741823
min	0

4.118 Reference dynamic_dmaLogicChannelConfig_MinorLoopLink↩ ChValueType

Vendor specific: Set the Dma Channel minor loop logic channel link.

Property	Value
type	ECUC-REFERENCE-DEF
origin	NXP
lowerMultiplicity	0
upperMultiplicity	1
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
postBuildVariantValue	false
valueConfigClasses	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
requiresSymbolicNameValue	true
destination	/AUTOSAR/EcucDefs/Mcl/MclConfig/dmaLogicChannel_Type

4.119 Container dmaLogicChannelConfig_TransferMajorLoopType

Vendor specific: Transfer Major Loop

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

4.120 Parameter dmaLogicChannelConfig_enMajorLoopLinkCh

Vendor specific: Enable the Dma Channel major loop logic channel linking.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A

Property	Value
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	false

4.121 Parameter dmaLogicChannelConfig_MajorLoopCountType

Vendor specific: Set the Dma Channel major loop count.

If minor loop channel linking is enabled, the major loop count limit is between 0 and 511.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD
defaultValue	1
max	32767
min	0

4.122 Reference dynamic_dmaLogicChannelConfig_MajorLoopLink↩ ChValueType

Vendor specific: Set the Dma Channel major loop logic channel link.

Property	Value
type	ECUC-REFERENCE-DEF
origin	NXP
lowerMultiplicity	0
upperMultiplicity	1
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE

Property	Value
postBuildVariantValue	false
valueConfigClasses	VARIANT-POST-BUILD: POST-BUILD
	VARIANT-PRE-COMPILE: PRE-COMPILE
requiresSymbolicNameValue	true
destination	/AUTOSAR/EcucDefs/Mcl/MclConfig/dmaLogicChannel_Type

4.123 Container trgmuxInstaceList

List of Instance.

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	0
upperMultiplicity	1
postBuildVariantMultiplicity	true
multiplicityConfigClasses	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE

4.124 Parameter trgmuxHardwareInstance

Trigger Mux Hardware Instance.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	TRGMUX_IP_HW_INST_0
literals	['TRGMUX_IP_HW_INST_0']

4.125 Container trgmuxLogicGroup

List of Logic Trigger Groups.

Included subcontainers:

- trgmuxLogicTrigger

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	0
upperMultiplicity	20
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE

4.126 Parameter trgmuxLogicGroupHardwareInstance

Trigger Mux Hardware Instance.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	TRGMUX_IP_HW_INST_0
literals	['TRGMUX_IP_HW_INST_0']

4.127 Parameter trgmuxLogicGroup_Name

Logic Trigger Group.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF

Property	Value
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	TRGMUX_IP_DMA
literals	['TRGMUX_IP_DMA', 'TRGMUX_IP_EXTOUT0', 'TRGMUX_IP_EXT↩ OUT1', 'TRGMUX_IP_ADC0', 'TRGMUX_IP_ADC1', 'TRGMUX_IP_C↩ MP0', 'TRGMUX_IP_FTM0', 'TRGMUX_IP_FTM1', 'TRGMUX_IP_FT↩ M2', 'TRGMUX_IP_FTM3', 'TRGMUX_IP_PDB0', 'TRGMUX_IP_PDB1', 'TRGMUX_IP_FLEXIO', 'TRGMUX_IP_LPIT0', 'TRGMUX_IP_LPUAR↩ T0', 'TRGMUX_IP_LPUART1', 'TRGMUX_IP_LPI2C0', 'TRGMUX_IP_↩ LPSPI0', 'TRGMUX_IP_LPSPI1', 'TRGMUX_IP_LPTMR0']

4.128 Parameter trgmuxLogicGroup_Lock

Logic Trigger Lock.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	false

4.129 Container trgmuxLogicTrigger

List of Logic Triggers.

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	53
postBuildVariantMultiplicity	true
multiplicityConfigClasses	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE

4.130 Parameter trgmuxLogicTrigger_Name

Logic Trigger Name.

Property	Value
type	ECUC-STRING-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	TRGMUX_LOGIC_GROUP_0_TRIGGER↩ _0

4.131 Parameter trgmuxLogicTrigger_Output

Logic Trigger Output.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-POST-BUILD: PRE-COMPILE

Property	Value
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	TRGMUX_IP_OUTPUT_DMA_CH0
literals	['TRGMUX_IP_OUTPUT_DMA_CH0', 'TRGMUX_IP_OUTPUT_DMA_CH1', 'TRGMUX_IP_OUTPUT_DMA_CH2', 'TRGMUX_IP_OUTPUT_DMA_CH3', 'TRGMUX_IP_OUTPUT_EXTOUT0_TRGMUX_OUT0', 'TRGMUX_IP_OUTPUT_EXTOUT0_TRGMUX_OUT1', 'TRGMUX_IP_OUTPUT_EXTOUT0_TRGMUX_OUT2', 'TRGMUX_IP_OUTPUT_EXTOUT0_TRGMUX_OUT3', 'TRGMUX_IP_OUTPUT_EXTOUT1_TRGMUX_OUT4', 'TRGMUX_IP_OUTPUT_EXTOUT1_TRGMUX_OUT5', 'TRGMUX_IP_OUTPUT_EXTOUT1_TRGMUX_OUT6', 'TRGMUX_IP_OUTPUT_EXTOUT1_TRGMUX_OUT7', 'TRGMUX_IP_OUTPUT_ADC0_ADHWT_TRIG_0', 'TRGMUX_IP_OUTPUT_ADC0_ADHWT_TRIG_1', 'TRGMUX_IP_OUTPUT_ADC0_ADHWT_TRIG_2', 'TRGMUX_IP_OUTPUT_ADC0_ADHWT_TRIG_3', 'TRGMUX_IP_OUTPUT_ADC1_ADHWT_TRIG_0', 'TRGMUX_IP_OUTPUT_ADC1_ADHWT_TRIG_1', 'TRGMUX_IP_OUTPUT_ADC1_ADHWT_TRIG_2', 'TRGMUX_IP_OUTPUT_ADC1_ADHWT_TRIG_3', 'TRGMUX_IP_OUTPUT_CMP0_SAMPLE_INPUT', 'TRGMUX_IP_OUTPUT_FTM0_HWTRIG0', 'TRGMUX_IP_OUTPUT_FTM0_FAULT0', 'TRGMUX_IP_OUTPUT_FTM0_FAULT1', 'TRGMUX_IP_OUTPUT_FTM0_FAULT2', 'TRGMUX_IP_OUTPUT_FTM1_HWTRIG0', 'TRGMUX_IP_OUTPUT_FTM1_FAULT0', 'TRGMUX_IP_OUTPUT_FTM1_FAULT1', 'TRGMUX_IP_OUTPUT_FTM1_FAULT2', 'TRGMUX_IP_OUTPUT_FTM2_HWTRIG0', 'TRGMUX_IP_OUTPUT_FTM2_FAULT0', 'TRGMUX_IP_OUTPUT_FTM2_FAULT1', 'TRGMUX_IP_OUTPUT_FTM2_FAULT2', 'TRGMUX_IP_OUTPUT_FTM3_HWTRIG0', 'TRGMUX_IP_OUTPUT_FTM3_FAULT0', 'TRGMUX_IP_OUTPUT_FTM3_FAULT1', 'TRGMUX_IP_OUTPUT_FTM3_FAULT2', 'TRGMUX_IP_OUTPUT_PDB0_TRIGGER_IN0', 'TRGMUX_IP_OUTPUT_PDB1_TRIGGER_IN0', 'TRGMUX_IP_OUTPUT_FLEXIO_TRG_TIM0', 'TRGMUX_IP_OUTPUT_FLEXIO_TRG_TIM1', 'TRGMUX_IP_OUTPUT_FLEXIO_TRG_TIM2', 'TRGMUX_IP_OUTPUT_FLEXIO_TRG_TIM3', 'TRGMUX_IP_OUTPUT_LPIT0_TRG_CH0', 'TRGMUX_IP_OUTPUT_LPIT0_TRG_CH1', 'TRGMUX_IP_OUTPUT_LPIT0_TRG_CH2', 'TRGMUX_IP_OUTPUT_LPIT0_TRG_CH3', 'TRGMUX_IP_OUTPUT_LPUART0_TRG', 'TRGMUX_IP_OUTPUT_LPUART1_TRG', 'TRGMUX_IP_OUTPUT_LPI2C0_TRG', 'TRGMUX_IP_OUTPUT_LPSPI0_TRG', 'TRGMUX_IP_OUTPUT_LPSPI1_TRG', 'TRGMUX_IP_OUTPUT_LPTMR0_ALT0']

4.132 Parameter trgmuxLogicTrigger_Input

Logic Trigger Input.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1

Property	Value
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-POST-BUILD: PRE-COMPILE
	VARIANT-PRE-COMPILE: PRE-COMPILE
defaultValue	TRGMUX_IP_INPUT_LOGIC0_VSS
literals	['TRGMUX_IP_INPUT_LOGIC0_VSS', 'TRGMUX_IP_INPUT_LOGIC1_VDD', 'TRGMUX_IP_INPUT_TRGMUX_IN0', 'TRGMUX_IP_INPUT_TRGMUX_IN1', 'TRGMUX_IP_INPUT_TRGMUX_IN2', 'TRGMUX_IP_INPUT_TRGMUX_IN3', 'TRGMUX_IP_INPUT_TRGMUX_IN4', 'TRGMUX_IP_INPUT_TRGMUX_IN5', 'TRGMUX_IP_INPUT_TRGMUX_IN6', 'TRGMUX_IP_INPUT_TRGMUX_IN7', 'TRGMUX_IP_INPUT_TRGMUX_IN8', 'TRGMUX_IP_INPUT_TRGMUX_IN9', 'TRGMUX_IP_INPUT_TRGMUX_IN10', 'TRGMUX_IP_INPUT_TRGMUX_IN11', 'TRGMUX_IP_INPUT_CMP0_OUT', 'TRGMUX_IP_INPUT_LPIT_CH0', 'TRGMUX_IP_INPUT_LPIT_CH1', 'TRGMUX_IP_INPUT_LPIT_CH2', 'TRGMUX_IP_INPUT_LPIT_CH3', 'TRGMUX_IP_INPUT_LPTMR0', 'TRGMUX_IP_INPUT_FTM0_INIT_TRIG', 'TRGMUX_IP_INPUT_FTM0_EXT_TRIG', 'TRGMUX_IP_INPUT_FTM1_INIT_TRIG', 'TRGMUX_IP_INPUT_FTM1_EXT_TRIG', 'TRGMUX_IP_INPUT_FTM2_INIT_TRIG', 'TRGMUX_IP_INPUT_FTM2_EXT_TRIG', 'TRGMUX_IP_INPUT_FTM3_INIT_TRIG', 'TRGMUX_IP_INPUT_FTM3_EXT_TRIG', 'TRGMUX_IP_INPUT_ADC0_COCO_0', 'TRGMUX_IP_INPUT_ADC0_COCO_1', 'TRGMUX_IP_INPUT_ADC1_COCO_0', 'TRGMUX_IP_INPUT_ADC1_COCO_1', 'TRGMUX_IP_INPUT_PDB0_ADCH0_TRIG', 'TRGMUX_IP_INPUT_PDB0_PULSE_OUT', 'TRGMUX_IP_INPUT_PDB1_ADCH0_TRIG', 'TRGMUX_IP_INPUT_PDB1_PULSE_OUT', 'TRGMUX_IP_INPUT_RTC_ALARM', 'TRGMUX_IP_INPUT_RTC_SECOND', 'TRGMUX_IP_INPUT_FLEXIO_TRIG0', 'TRGMUX_IP_INPUT_FLEXIO_TRIG1', 'TRGMUX_IP_INPUT_FLEXIO_TRIG2', 'TRGMUX_IP_INPUT_FLEXIO_TRIG3', 'TRGMUX_IP_INPUT_LPUART0_RX_DATA', 'TRGMUX_IP_INPUT_LPUART0_TX_DATA', 'TRGMUX_IP_INPUT_LPUART0_RX_IDLE', 'TRGMUX_IP_INPUT_LPUART1_RX_DATA', 'TRGMUX_IP_INPUT_LPUART1_TX_DATA', 'TRGMUX_IP_INPUT_LPUART1_RX_IDLE', 'TRGMUX_IP_INPUT_LPI2C0_MASTER_TRIG', 'TRGMUX_IP_INPUT_LPI2C0_SLAVE_TRIG', 'TRGMUX_IP_INPUT_LPSPI0_FRAME', 'TRGMUX_IP_INPUT_LPSPI0_RX_DATA', 'TRGMUX_IP_INPUT_LPSPI1_FRAME', 'TRGMUX_IP_INPUT_LPSPI1_RX_DATA', 'TRGMUX_IP_INPUT_SIM_SW_TRIG']

4.133 Container FlexioCommon

List of Flexio instances available on the platform.

Included subcontainers:

- FlexioMclLogicChannels

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	0
upperMultiplicity	8
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: POST-BUILD

4.134 Parameter FlexioMclInstances

Select one of the Flexio instance available on the platform.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	FLEXIO_0
literals	['FLEXIO_0']

4.135 Parameter FlexioDebugEnable

Enable Debug Mode

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE

Property	Value
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	false

4.136 Container FlexioMclLogicChannels

Flexio Logic Channel Configuration

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	4
postBuildVariantMultiplicity	false
multiplicityConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE

4.137 Parameter FlexioMclChannelId

Select one of the Flexio channels available on the platform for selected instance.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	CHANNEL_0
literals	['CHANNEL_0', 'CHANNEL_1', 'CHANNEL_2', 'CHANNEL_3']

4.138 Parameter FlexioMclPinId

Select one of the Flexio pins available on the platform for selected instance.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	PIN_0
literals	['PIN_0', 'PIN_1', 'PIN_2', 'PIN_3', 'PIN_4', 'PIN_5', 'PIN_6', 'PIN_7']

4.139 Parameter FlexioMclAddPinEnable

Enable feature to select one more Flexio pin.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	false

4.140 Parameter FlexioMclAddPinId

Select one of the Flexio pins available on the platform for selected instance.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	PIN_1
literals	['PIN_0', 'PIN_1', 'PIN_2', 'PIN_3', 'PIN_4', 'PIN_5', 'PIN_6', 'PIN_7']

4.141 Parameter FlexioMclAddChannelEnable

Enable feature to select one more Flexio channel.

Property	Value
type	ECUC-BOOLEAN-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	false

4.142 Parameter FlexioMclAddChannelId

Select one of the Flexio channels available on the platform for selected instance.

Property	Value
type	ECUC-ENUMERATION-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1

Property	Value
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	true
valueConfigClasses	VARIANT-PRE-COMPILE: PRE-COMPILE
	VARIANT-POST-BUILD: PRE-COMPILE
defaultValue	CHANNEL_0
literals	['CHANNEL_0', 'CHANNEL_1', 'CHANNEL_2', 'CHANNEL_3']

4.143 Container CommonPublishedInformation

Vendor specific:

Common container, aggregated by all modules. It contains published information about vendor and versions.

Included subcontainers:

- None

Property	Value
type	ECUC-PARAM-CONF-CONTAINER-DEF
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A

4.144 Parameter ArReleaseMajorVersion

Vendor specific:

Major version number of AUTOSAR specification on which the appropriate implementation is based on.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A

Property	Value
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
	VARIANT-POST-BUILD: PUBLISHED-INFORMATION
defaultValue	4
max	4
min	4

4.145 Parameter ArReleaseMinorVersion

Vendor specific:

Minor version number of AUTOSAR specification on which the appropriate implementation is based on.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
	VARIANT-POST-BUILD: PUBLISHED-INFORMATION
defaultValue	7
max	7
min	7

4.146 Parameter ArReleaseRevisionVersion

Vendor specific:

Revision version number of AUTOSAR specification on which the appropriate implementation is based on.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1

Property	Value
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
	VARIANT-POST-BUILD: PUBLISHED-INFORMATION
defaultValue	0
max	0
min	0

4.147 Parameter ModuleId

Vendor specific:

Module ID of this module from Module List.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
	VARIANT-POST-BUILD: PUBLISHED-INFORMATION
defaultValue	255
max	255
min	255

4.148 Parameter SwMajorVersion

Vendor specific:

Major version number of the vendor specific implementation of the module. The numbering is vendor specific.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP

Property	Value
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
	VARIANT-POST-BUILD: PUBLISHED-INFORMATION
defaultValue	2
max	2
min	2

4.149 Parameter SwMinorVersion

Vendor specific:

Minor version number of the vendor specific implementation of the module. The numbering is vendor specific.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
	VARIANT-POST-BUILD: PUBLISHED-INFORMATION
defaultValue	0
max	0
min	0

4.150 Parameter SwPatchVersion

Vendor specific:

Patch level version number of the vendor specific implementation of the module. The numbering is vendor specific.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
	VARIANT-POST-BUILD: PUBLISHED-INFORMATION
defaultValue	0
max	0
min	0

4.151 Parameter VendorId

Vendor specific:

Vendor ID of the dedicated implementation of this module according to the AUTOSAR vendor list.

Property	Value
type	ECUC-INTEGER-PARAM-DEF
origin	NXP
symbolicNameValue	false
lowerMultiplicity	1
upperMultiplicity	1
postBuildVariantMultiplicity	N/A
multiplicityConfigClasses	N/A
postBuildVariantValue	false
valueConfigClasses	VARIANT-PRE-COMPILE: PUBLISHED-INFORMATION
	VARIANT-POST-BUILD: PUBLISHED-INFORMATION
defaultValue	43
max	43
min	43

This chapter describes the Tresos configuration plug-in for the MCL Driver. The most of the parameters are described below.

Chapter 5

Module Index

5.1 Software Specification

Here is a list of all modules:

Chapter 6

Module Documentation

6.1 CACHE IP Driver

6.1.1 Detailed Description

Enum Reference

- enum Cache_Ip_Type

  This type contains the Cache Ip types.
- enum Cache_Ip_BusType

  This type contains the Cache Ip Bus types.

Function Reference

- Std_ReturnType Cache_Ip_Enable (const Cache_Ip_Type CacheType, const Cache_Ip_BusType BusType)

  This function enables the Cache Ip Driver.
- Std_ReturnType Cache_Ip_Disable (const Cache_Ip_Type CacheType, const Cache_Ip_BusType Bus↩ Type)

  This function disables the Cache Ip Driver.
- Std_ReturnType Cache_Ip_Invalidate (const Cache_Ip_Type CacheType, const Cache_Ip_BusType Bus↩ Type)

  This function Invalidates the Cache Ip Driver.
- Std_ReturnType Cache_Ip_Clean (const Cache_Ip_Type CacheType, const Cache_Ip_BusType BusType, const boolean EnInvalidate)

  This function Clean the Cache Ip Driver.
- Std_ReturnType Cache_Ip_InvalidateByAddr (const Cache_Ip_Type CacheType, const Cache_Ip_BusType BusType, const uint32 Addr, const uint32 Length)

  This function Invalidates By Address the Cache Ip Driver.
- Std_ReturnType Cache_Ip_CleanByAddr (const Cache_Ip_Type CacheType, const Cache_Ip_BusType BusType, const boolean EnInvalidate, const uint32 Addr, const uint32 Length)

  This function Clean By Address the Cache Ip Driver.

6.1.2 Enum Reference

6.1.2.1 Cache_Ip_Type

```
enum Cache_Ip_Type
```

This type contains the Cache Ip types.

The cache types specifies what type of cache shall be used when calling the interface. The CACHE_IP_LMEM select LMEM caches. The CACHE_IP_CORE select CORE caches.

Definition at line 109 of file Cache_Ip.h.

6.1.2.2 Cache_Ip_BusType

```
enum Cache_Ip_BusType
```

This type contains the Cache Ip Bus types.

The cache bus types specifies what type of cache shall be used when calling the interface. The CACHE_IP_PC_BUS selects Processor Code (PC) bus (used with Cache Lmem). The CACHE_IP_PS_BUS selects Processor System (PS) bus (used with Cache Lmem). The CACHE_IP_ALL_BUS selects PC and PS bus (used with Cache Lmem). The CACHE_IP_INSTRUCTION selects instruction cache (used with Cache Core). The CACHE_IP_DATA selects data cache (used with Cache Core).

Definition at line 124 of file Cache_Ip.h.

6.1.3 Function Reference

6.1.3.1 Cache_Ip_Enable()

```
Std_ReturnType Cache_Ip_Enable (
            const Cache_Ip_Type CacheType,
            const Cache_Ip_BusType BusType )
```

This function enables the Cache Ip Driver.

This service is a reentrant function that shall enable the Cache Ip driver.

Parameters

in	*CacheType*	Specifies the cache type.
in	*BusType*	Specifies the bus type.

Returns

E_OK is returned if the enable action finished correctly. E_NOT_OK is returned if the enable action finished incorrectly.

6.1.3.2 Cache_Ip_Disable()

```
Std_ReturnType Cache_Ip_Disable (
            const Cache_Ip_Type CacheType,
            const Cache_Ip_BusType BusType )
```

This function disables the Cache Ip Driver.

This service is a reentrant function that shall disables the Cache Ip driver.

Parameters

in	*CacheType*	Specifies the cache type.
in	*BusType*	Specifies the bus type.

Returns

E_OK is returned if the disable action finished correctly. E_NOT_OK is returned if the disable action finished incorrectly.

6.1.3.3 Cache_Ip_Invalidate()

```
Std_ReturnType Cache_Ip_Invalidate (
            const Cache_Ip_Type CacheType,
            const Cache_Ip_BusType BusType )
```

This function Invalidates the Cache Ip Driver.

This service is a reentrant function that shall Invalidates the Cache Ip driver. The Invalidate operation applies for the entire selected Cache Type.

A cache invalidate operation ensures that updates made visible by observers that access memory at the point to which the invalidate is defined are made visible to an observer that controls the cache. This might result in the loss of updates to the locations affected by the invalidate operation that have been written by observers that access the cache. If the address of an entry on which the invalidate operates does not have a Normal Cacheable attribute, or if the cache is disabled, then an invalidate operation also ensures that this address is not present in the cache.

Parameters

in	*CacheType*	Specifies the cache type.
in	*BusType*	Specifies the bus type.

Returns

E_OK is returned if the invalidation finished correctly. E_NOT_OK is returned if the invalidation finished incorrectly.

6.1.3.4 Cache_Ip_Clean()

```
Std_ReturnType Cache_Ip_Clean (
            const Cache_Ip_Type CacheType,
            const Cache_Ip_BusType BusType,
            const boolean EnInvalidate )
```

This function Clean the Cache Ip Driver.

This service is a reentrant function that shall Clean the Cache Ip driver. The Clean operation applies for the entire selected Cache Type.

A cache clean operation ensures that updates made by an observer that controls the cache are made visible to other observers that can access memory at the point to which the operation is performed. Once the Clean has completed, the new memory values are guaranteed to be visible to the point to which the operation is performed, for example to the point of unification. The cleaning of a cache entry from a cache can overwrite memory that has been written by another observer only if the entry contains a location that has been written to by an observer in the shareability domain of that memory location.

A cache clean and invalidate operation behaves as the execution of a clean operation followed immediately by an invalidate operation. Both operations are performed to the same location.

Parameters

in	*CacheType*	Specifies the cache type.
in	*BusType*	Specifies the bus type.
in	*enInvalidate*	Specifies to execute operation Clean&Invalidate.

Returns

E_OK is returned if the cleaning finished correctly. E_NOT_OK is returned if the cleaning finished incorrectly.

6.1.3.5 Cache_Ip_InvalidateByAddr()

```
Std_ReturnType Cache_Ip_InvalidateByAddr (
            const Cache_Ip_Type CacheType,
            const Cache_Ip_BusType BusType,
            const uint32 Addr,
            const uint32 Length )
```

This function Invalidates By Address the Cache Ip Driver.

This service is a reentrant function that shall Invalidates By Address the Cache Ip driver. The Invalidate By Address operation applies for the memory segment specified by the start Address and Length. The operation Invalidates only multiple of Cache Line Size, thus the specified memory segment shall be aligned and multiple of the Cache Line Size.

A cache invalidate operation ensures that updates made visible by observers that access memory at the point to which the invalidate is defined are made visible to an observer that controls the cache. This might result in the loss of updates to the locations affected by the invalidate operation that have been written by observers that access the cache. If the address of an entry on which the invalidate operates does not have a Normal Cacheable attribute, or if the cache is disabled, then an invalidate operation also ensures that this address is not present in the cache.

Parameters

in	*CacheType*	Specifies the cache type.
in	*BusType*	Specifies the bus type.
in	*Addr*	Specifies the memory segment start address.
in	*Length*	Specifies the memory segment length.

Returns

E_OK is returned if the invalidation finished correctly. E_NOT_OK is returned if the invalidation finished incorrectly.

6.1.3.6 Cache_Ip_CleanByAddr()

```
Std_ReturnType Cache_Ip_CleanByAddr (
            const Cache_Ip_Type CacheType,
            const Cache_Ip_BusType BusType,
            const boolean EnInvalidate,
            const uint32 Addr,
            const uint32 Length )
```

This function Clean By Address the Cache Ip Driver.

This service is a reentrant function that shall Clean By Address the Cache Ip driver. The Clean By Address operation applies for the memory segment specified by the start Address and Length. The operation Cleans only multiple of Cache Line Size, thus the specified memory segment shall be aligned and multiple of the Cache Line Size.

A cache clean operation ensures that updates made by an observer that controls the cache are made visible to other observers that can access memory at the point to which the operation is performed. Once the Clean has completed, the new memory values are guaranteed to be visible to the point to which the operation is performed, for example to the point of unification. The cleaning of a cache entry from a cache can overwrite memory that has been written by another observer only if the entry contains a location that has been written to by an observer in the shareability domain of that memory location.

A cache clean and invalidate operation behaves as the execution of a clean operation followed immediately by an invalidate operation. Both operations are performed to the same location.

Parameters

in	*CacheType*	Specifies the cache type.
in	*BusType*	Specifies the bus type.
in	*enInvalidate*	Specifies to execute operation Clean&Invalidate.
in	*Addr*	Specifies the memory segment start address.
in	*Length*	Specifies the memory segment length.

Returns

E_OK is returned if the cleaning finished correctly. E_NOT_OK is returned if the cleaning finished incorrectly.

6.2 MCL Driver

6.2.1 Detailed Description

Data Structures

- struct Mcl_DmaChannelGlobalListType

  This type contains the Mcl Dma Channel Global List. More...
- struct Mcl_DmaChannelTransferListType

  This type contains the Mcl Dma Channel Transfer List. More...
- struct Mcl_DmaChannelScatterGatherListType

  This type contains the Mcl Dma Channel Scatter/Gather List. More...
- struct Mcl_DmaInstanceStatusType

  This type contains the Mcl Dma Instance Status. More...
- struct Mcl_DmaChannelStatusType

  This type contains the Mcl Dma Channel Status. More...
- struct Mcl_ConfigType

  This type contains the Mcl Configuration. More...

Macros

- #define MCL_DET_INIT

  API service ID for Mcl_Init function.
- #define MCL_DET_DMA_INSTANCE_COMMAND

  API service ID for Mcl_SetDmaInstanceCommand function.
- #define MCL_DET_DMA_INSTANCE_STATUS

  API service ID for Mcl_GetDmaInstanceStatus function.
- #define MCL_DET_DMA_CHANNEL_COMMAND

  API service ID for Mcl_SetDmaChannelCommand function.
- #define MCL_DET_DMA_CHANNEL_STATUS

  API service ID for Mcl_GetDmaChannelStatus function.
- #define MCL_DET_DMA_GLOBAL

  API service ID for Mcl_SetDmaChannelGlobalList function.
- #define MCL_DET_DMA_TRANSFER

  API service ID for Mcl_SetDmaChannelTransferList function.
- #define MCL_DET_DMA_SCATTER_GATHER_LIST

  API service ID for Mcl_SetDmaChannelScatterGatherList function.
- #define MCL_DET_DMA_INFORMATION

  API service ID for Mcl_GetDmaChannelParam function.
- #define MCL_DET_DMA_SCATTER_GATHER_CONFIG

  API service ID for Mcl_SetDmaChannelScatterGatherConfig function.
- #define MCL_DET_DEINIT

  API service ID for Mcl_DeInit function.
- #define MCL_DET_CACHE_ENABLE

API service ID for Mcl_CacheEnable function.

- #define MCL_DET_CACHE_DISABLE

  API service ID for Mcl_CacheDisable function.
- #define MCL_DET_CACHE_INVALIDATE

  API service ID for Mcl_CacheInvalidate function.
- #define MCL_DET_CACHE_CLEAN

  API service ID for Mcl_CacheClean function.
- #define MCL_DET_CACHE_INVALIDATE_BY_ADDRESS

  API service ID for Mcl_CacheInvalidateByAddr function.
- #define MCL_DET_CACHE_CLEAN_BY_ADDRESS

  API service ID for Mcl_CacheCleanByAddr function.
- #define MCL_DET_TRGMUX_INPUT

  API service ID for Mcl_SetTrgMuxInput function.
- #define MCL_DET_TRGMUX_LOCK

  API service ID for Mcl_SetTrgMuxLock function.
- #define MCL_DET_TRGMUX_SET_INPUT

  API service ID for Mcl_SetTrgMuxInput function.
- #define MCL_DET_TRGMUX_SET_LOC

  API service ID for Mcl_SetTrgMuxLock function.
- #define MCL_E_UNINIT

  All API's having pointers as parameters shall return this error if called with with a NULL value.
- #define MCL_E_PARAM_POINTER

  All API's having pointers as parameters shall return this error if called with with a NULL value.
- #define MCL_E_INVALID_INSTANCE

  All API's called with wrong instance shall return this error.
- #define MCL_E_INVALID_CHANNEL

  All API's called with wrong channel shall return this error.
- #define MCL_E_INVALID_COMMAND

  All API's called with wrong instance shall return this error.
- #define MCL_E_INVALID_PARAMETER

  All API's called with wrong read parameter shall return this error.
- #define MCL_E_INVALID_STATE

  All API's called in wrong sequence shall return this error.
- #define MCL_E_INCONSISTENCY

  All API's called while hardware has error status shall return this error.
- #define MCL_E_TIMEOUT

  All API's called with a timeout value shall return this error if execution is not finished in the allocated timeframe.
- #define MCL_E_PROTECTED

  If DET error reporting is enabled, the MCL will check if registers are protected.
- #define MCL_E_INIT_FAILED

  If VariantPreCompile is used, the configuration pointer shall have a NULL_PTR value. If VariantPostBuild is used, the configuration pointer shall be different from NULL_PTR. And in case of violate will return MCL_E_INIT_↩ FAILED.

Enum Reference

- enum Mcl_DmaInstanceCmdType

  This type contains the Mcl Dma Instance Commands.
- enum Mcl_DmaChannelCmdType

  This type contains the Mcl Dma Channel Commands.
- enum Mcl_DmaChannelGlobalParamType

  This type contains the Mcl Dma Channel Global Parameters.
- enum Mcl_DmaChannelTransferParamType

  This type contains the Mcl Dma Channel Transfer Parameters.
- enum Mcl_DmaChannelStateType

  This type contains the Mcl Dma Channel State values.
- enum Mcl_DmaChannelInfoParamType

  This type contains the Mcl Dma Channel Information Parameters.
- enum Mcl_CacheType

  This type contains the Mcl Cache Type selection.
- enum Mcl_CacheBusType

  This type contains the Mcl Cache Bus Type selection.
- enum Mcl_ReturnType

  This type contains the Mcl Return Type.

Function Reference

- void Mcl_Init (const Mcl_ConfigType *const ConfigPtr)

  This function initializes the Mcl Driver.
- void Mcl_DeInit (void)

  This function deinitializes the Mcl Driver.
- void Mcl_SetDmaInstanceCommand (const uint32 Instance, const Mcl_DmaInstanceCmdType Command)

  This function sets Dma Instance Command.
- void Mcl_GetDmaInstanceStatus (const uint32 Instance, Mcl_DmaInstanceStatusType *const Status)

  This function gets Dma Instance Status.
- void Mcl_SetDmaChannelCommand (const uint32 Channel, const Mcl_DmaChannelCmdType Command)

  This function sets Dma Channel Command.
- void Mcl_GetDmaChannelStatus (const uint32 Channel, Mcl_DmaChannelStatusType *const Status)

  This function gets Dma Channel Status.
- void Mcl_SetDmaChannelGlobalList (const uint32 Channel, const Mcl_DmaChannelGlobalListType List[], const uint32 ListDimension)

  This function sets Dma Channel Global List settings.
- void Mcl_SetDmaChannelTransferList (const uint32 Channel, const Mcl_DmaChannelTransferListType List[], const uint32 ListDimension)

  This function sets Dma Channel Transfer List settings.
- void Mcl_SetDmaChannelScatterGatherList (const uint32 Channel, const uint32 Element, const Mcl_DmaChannelScatterG List[], const uint32 ListDimension)

  This function sets Dma Channel Scatter/Gather List settings.
- void Mcl_GetDmaChannelParam (const uint32 Channel, const Mcl_DmaChannelInfoParamType Param, uint32 *const Value)

This function gets the Dma Channel Parameter value.

- void Mcl_SetDmaChannelScatterGatherConfig (const uint32 Channel, const uint32 Element)

  This function configures the Dma Channel Scatter/Gather.
- void Mcl_SetTrgMuxInput (const uint32 Trigger, const uint32 Input)

  This function sets the Trgmux Trigger Input selection.
- void Mcl_SetTrgMuxLock (const uint32 Trigger)

  This function sets the Trgmux Trigger Lock.
- void Mcl_CacheEnable (Mcl_CacheType CacheType, Mcl_CacheBusType BusType)

  This function enables the Cache.
- void Mcl_CacheDisable (Mcl_CacheType CacheType, Mcl_CacheBusType BusType)

  This function disables the Cache.
- void Mcl_CacheInvalidate (Mcl_CacheType CacheType, Mcl_CacheBusType BusType)

  This function Invalidates the Cache.
- void Mcl_CacheClean (Mcl_CacheType CacheType, Mcl_CacheBusType BusType, boolean EnInvalidate)

  This function Cleans the Cache.
- void Mcl_CacheInvalidateByAddr (Mcl_CacheType CacheType, Mcl_CacheBusType BusType, uint32 Addr, uint32 Length)

  This function Invalidates the Cache by address.
- void Mcl_CacheCleanByAddr (Mcl_CacheType CacheType, Mcl_CacheBusType BusType, boolean En↩ Invalidate, uint32 Addr, uint32 Length)

  This function Cleans the Cache by address.
- void Mcl_SelectCommonTimebase (uint8 instance, uint16 elementSyncList)

  Implementation specific function to updates the Global Timebase bits of configured modules.
- void Mcl_GetVersionInfo (Std_VersionInfoType *const VersionInfo)

  Returns the version information of this module.

6.2.1.1 MISRA-C:2012 violations

6.2.2 Data Structure Documentation

6.2.2.1 struct Mcl_DmaChannelGlobalListType

This type contains the Mcl Dma Channel Global List.

The Mcl Dma Channel Global List contains a pair composed from Dma Channel Global Parameter Type and the Value of the parameter.

Definition at line 515 of file CDD_Mcl.h.

Data Fields

- Mcl_DmaChannelGlobalParamType Param

  The Mcl Dma Channel Global Parameter Type selects a parameter form the Global Parameter enum type.
- uint32 Value

  The Value stores the parameter's value.

6.2.2.1.1 Field Documentation

6.2.2.1.1.1 Param `Mcl_DmaChannelGlobalParamType Param`

The Mcl Dma Channel Global Parameter Type selects a parameter form the Global Parameter enum type.

Definition at line 516 of file CDD_Mcl.h.

6.2.2.1.1.2 Value `uint32 Value`

The Value stores the parameter's value.

Definition at line 517 of file CDD_Mcl.h.

6.2.2.2 struct Mcl_DmaChannelTransferListType

This type contains the Mcl Dma Channel Transfer List.

The Mcl Dma Channel Transfer List contains a pair composed from Dma Channel Transfer Parameter Type and the Value of the parameter.

Definition at line 527 of file CDD_Mcl.h.

Data Fields

- Mcl_DmaChannelTransferParamType Param
  The Mcl Dma Channel Transfer Parameter Type selects a parameter form the Transfer Parameter enum type.
- uint32 Value
  The Value stores the parameter's value.

6.2.2.2.1 Field Documentation

6.2.2.2.1.1 Param `Mcl_DmaChannelTransferParamType Param`

The Mcl Dma Channel Transfer Parameter Type selects a parameter form the Transfer Parameter enum type.

Definition at line 528 of file CDD_Mcl.h.

6.2.2.2.1.2 Value `uint32 Value`

The Value stores the parameter's value.

Definition at line 529 of file CDD_Mcl.h.

6.2.2.3 struct Mcl_DmaChannelScatterGatherListType

This type contains the Mcl Dma Channel Scatter/Gather List.

The Mcl Dma Channel Scatter/Gather List contains a pair composed from Dma Channel Scatter/Gather Parameter Type and the Value of the parameter.

Definition at line 539 of file CDD_Mcl.h.

Data Fields

- Mcl_DmaChannelTransferParamType Param
  The Mcl Dma Channel Transfer Parameter Type selects a parameter form the Transfer Parameter enum type.
- uint32 Value
  The Value stores the parameter's value.

6.2.2.3.1 Field Documentation

6.2.2.3.1.1 Param `Mcl_DmaChannelTransferParamType Param`

The Mcl Dma Channel Transfer Parameter Type selects a parameter form the Transfer Parameter enum type.

Definition at line 540 of file CDD_Mcl.h.

6.2.2.3.1.2 Value `uint32 Value`

The Value stores the parameter's value.

Definition at line 541 of file CDD_Mcl.h.

6.2.2.4 struct Mcl_DmaInstanceStatusType

This type contains the Mcl Dma Instance Status.

The Mcl Dma Instance Status contains the Hardware Errors, Active Id and Active indication for the running Dma Channel.

Definition at line 552 of file CDD_Mcl.h.

Data Fields

- uint32 Errors

  [VALUE] The Errors value is read from the DMA Instance Error Register (ES) as it is.
- uint8 ActiveId

  [VALUE] The ActiveId value is read from the DMA Instance Control Register (CR) field ACTIVE_ID.
- boolean Active

  [BOOLEAN] The Active value is read from the DMA Instance Control Register (CR) field ACTIVE.
- uint8 Version

  [VALUE] The Version value is read from the DMA Instance Control Register (CR) field VER.

6.2.2.4.1 Field Documentation

6.2.2.4.1.1 Errors `uint32 Errors`

[VALUE] The Errors value is read from the DMA Instance Error Register (ES) as it is.

Definition at line 553 of file CDD_Mcl.h.

6.2.2.4.1.2 ActiveId `uint8 ActiveId`

[VALUE] The ActiveId value is read from the DMA Instance Control Register (CR) field ACTIVE_ID.

Definition at line 554 of file CDD_Mcl.h.

6.2.2.4.1.3 Active `boolean Active`

[BOOLEAN] The Active value is read from the DMA Instance Control Register (CR) field ACTIVE.

Definition at line 555 of file CDD_Mcl.h.

6.2.2.4.1.4 Version `uint8 Version`

[VALUE] The Version value is read from the DMA Instance Control Register (CR) field VER.

Definition at line 556 of file CDD_Mcl.h.

6.2.2.5 struct Mcl_DmaChannelStatusType

This type contains the Mcl Dma Channel Status.

The Mcl Dma Channel Status contains the Hardware Errors, Active status and Done indication for the running Dma Channel.

Definition at line 566 of file CDD_Mcl.h.

Data Fields

- Mcl_DmaChannelStateType ChannelState

  [VALUE] The ChStateValue value is read from the internal DMA Driver Channel State Machine. Check UM for additional information.
- uint32 Errors

  [VALUE] The Errors value is read from the DMA Channel Error Register (CHx_ES) as it is.
- boolean Active

  [BOOLEAN] The Active value is read from the DMA Channel Control and Status Register (CHx_CSR) field ACTIVE.
- boolean Done

  [BOOLEAN] The Active value is read from the DMA Channel Control and Status Register (CHx_CSR) field DONE.

6.2.2.5.1 Field Documentation

6.2.2.5.1.1 ChannelState `Mcl_DmaChannelStateType ChannelState`

[VALUE] The ChStateValue value is read from the internal DMA Driver Channel State Machine. Check UM for additional information.

Definition at line 567 of file CDD_Mcl.h.

6.2.2.5.1.2 Errors `uint32 Errors`

[VALUE] The Errors value is read from the DMA Channel Error Register (CHx_ES) as it is.

Definition at line 568 of file CDD_Mcl.h.

6.2.2.5.1.3 Active `boolean Active`

[BOOLEAN] The Active value is read from the DMA Channel Control and Status Register (CHx_CSR) field ACTIVE.

Definition at line 569 of file CDD_Mcl.h.

6.2.2.5.1.4 Done `boolean Done`

[BOOLEAN] The Active value is read from the DMA Channel Control and Status Register (CHx_CSR) field DONE.

Definition at line 570 of file CDD_Mcl.h.

6.2.2.6 struct Mcl_ConfigType

This type contains the Mcl Configuration.

The Mcl Configuration structure contains pointers to the Ip's configuration structure. Based on the available support, specific configurations shall be stored.

Definition at line 338 of file Mcl_Types.h.

6.2.3 Macro Definition Documentation

6.2.3.1 MCL_DET_INIT

```
#define MCL_DET_INIT
```

API service ID for Mcl_Init function.

Parameters used when raising an error/exception

Definition at line 127 of file CDD_Mcl.h.

6.2.3.2 MCL_DET_DMA_INSTANCE_COMMAND

```
#define MCL_DET_DMA_INSTANCE_COMMAND
```

API service ID for Mcl_SetDmaInstanceCommand function.

Parameters used when raising an error/exception

Definition at line 135 of file CDD_Mcl.h.

6.2.3.3 MCL_DET_DMA_INSTANCE_STATUS

```
#define MCL_DET_DMA_INSTANCE_STATUS
```

API service ID for Mcl_GetDmaInstanceStatus function.

Parameters used when raising an error/exception

Definition at line 141 of file CDD_Mcl.h.

6.2.3.4 MCL_DET_DMA_CHANNEL_COMMAND

```
#define MCL_DET_DMA_CHANNEL_COMMAND
```

API service ID for Mcl_SetDmaChannelCommand function.

Parameters used when raising an error/exception

Definition at line 147 of file CDD_Mcl.h.

6.2.3.5 MCL_DET_DMA_CHANNEL_STATUS

```
#define MCL_DET_DMA_CHANNEL_STATUS
```

API service ID for Mcl_GetDmaChannelStatus function.

Parameters used when raising an error/exception

Definition at line 153 of file CDD_Mcl.h.

6.2.3.6 MCL_DET_DMA_GLOBAL

```
#define MCL_DET_DMA_GLOBAL
```

API service ID for Mcl_SetDmaChannelGlobalList function.

Parameters used when raising an error/exception

Definition at line 159 of file CDD_Mcl.h.

6.2.3.7 MCL_DET_DMA_TRANSFER

```
#define MCL_DET_DMA_TRANSFER
```

API service ID for Mcl_SetDmaChannelTransferList function.

Parameters used when raising an error/exception

Definition at line 165 of file CDD_Mcl.h.

6.2.3.8 MCL_DET_DMA_SCATTER_GATHER_LIST

```
#define MCL_DET_DMA_SCATTER_GATHER_LIST
```

API service ID for Mcl_SetDmaChannelScatterGatherList function.

Parameters used when raising an error/exception

Definition at line 171 of file CDD_Mcl.h.

6.2.3.9 MCL_DET_DMA_INFORMATION

```
#define MCL_DET_DMA_INFORMATION
```

API service ID for Mcl_GetDmaChannelParam function.

Parameters used when raising an error/exception

Definition at line 177 of file CDD_Mcl.h.

6.2.3.10 MCL_DET_DMA_SCATTER_GATHER_CONFIG

```
#define MCL_DET_DMA_SCATTER_GATHER_CONFIG
```

API service ID for Mcl_SetDmaChannelScatterGatherConfig function.

Parameters used when raising an error/exception

Definition at line 183 of file CDD_Mcl.h.

6.2.3.11 MCL_DET_DEINIT

```
#define MCL_DET_DEINIT
```

API service ID for Mcl_DeInit function.

Parameters used when raising an error/exception

Definition at line 193 of file CDD_Mcl.h.

6.2.3.12 MCL_DET_CACHE_ENABLE

```
#define MCL_DET_CACHE_ENABLE
```

API service ID for Mcl_CacheEnable function.

Parameters used when raising an error/exception

Definition at line 200 of file CDD_Mcl.h.

6.2.3.13 MCL_DET_CACHE_DISABLE

```
#define MCL_DET_CACHE_DISABLE
```

API service ID for Mcl_CacheDisable function.

Parameters used when raising an error/exception

Definition at line 206 of file CDD_Mcl.h.

6.2.3.14 MCL_DET_CACHE_INVALIDATE

```
#define MCL_DET_CACHE_INVALIDATE
```

API service ID for Mcl_CacheInvalidate function.

Parameters used when raising an error/exception

Definition at line 212 of file CDD_Mcl.h.

6.2.3.15 MCL_DET_CACHE_CLEAN

```
#define MCL_DET_CACHE_CLEAN
```

API service ID for Mcl_CacheClean function.

Parameters used when raising an error/exception

Definition at line 218 of file CDD_Mcl.h.

6.2.3.16 MCL_DET_CACHE_INVALIDATE_BY_ADDRESS

```
#define MCL_DET_CACHE_INVALIDATE_BY_ADDRESS
```

API service ID for Mcl_CacheInvalidateByAddr function.

Parameters used when raising an error/exception

Definition at line 224 of file CDD_Mcl.h.

6.2.3.17 MCL_DET_CACHE_CLEAN_BY_ADDRESS

```
#define MCL_DET_CACHE_CLEAN_BY_ADDRESS
```

API service ID for Mcl_CacheCleanByAddr function.

Parameters used when raising an error/exception

Definition at line 230 of file CDD_Mcl.h.

6.2.3.18 MCL_DET_TRGMUX_INPUT

```
#define MCL_DET_TRGMUX_INPUT
```

API service ID for Mcl_SetTrgMuxInput function.

Parameters used when raising an error/exception

Definition at line 240 of file CDD_Mcl.h.

6.2.3.19 MCL_DET_TRGMUX_LOCK

```
#define MCL_DET_TRGMUX_LOCK
```

API service ID for Mcl_SetTrgMuxLock function.

Parameters used when raising an error/exception

Definition at line 246 of file CDD_Mcl.h.

6.2.3.20 MCL_DET_TRGMUX_SET_INPUT

```
#define MCL_DET_TRGMUX_SET_INPUT
```

API service ID for Mcl_SetTrgMuxInput function.

Parameters used when raising an error/exception

Definition at line 259 of file CDD_Mcl.h.

6.2.3.21 MCL_DET_TRGMUX_SET_LOC

```
#define MCL_DET_TRGMUX_SET_LOC
```

API service ID for Mcl_SetTrgMuxLock function.

Parameters used when raising an error/exception

Definition at line 265 of file CDD_Mcl.h.

6.2.3.22 MCL_E_UNINIT

```
#define MCL_E_UNINIT
```

All API's having pointers as parameters shall return this error if called with with a NULL value.

Definition at line 277 of file CDD_Mcl.h.

6.2.3.23 MCL_E_PARAM_POINTER

```
#define MCL_E_PARAM_POINTER
```

All API's having pointers as parameters shall return this error if called with with a NULL value.

Definition at line 284 of file CDD_Mcl.h.

6.2.3.24 MCL_E_INVALID_INSTANCE

```
#define MCL_E_INVALID_INSTANCE
```

All API's called with wrong instance shall return this error.

Definition at line 290 of file CDD_Mcl.h.

6.2.3.25 MCL_E_INVALID_CHANNEL

```
#define MCL_E_INVALID_CHANNEL
```

All API's called with wrong channel shall return this error.

Definition at line 296 of file CDD_Mcl.h.

6.2.3.26 MCL_E_INVALID_COMMAND

```
#define MCL_E_INVALID_COMMAND
```

All API's called with wrong instance shall return this error.

Definition at line 302 of file CDD_Mcl.h.

6.2.3.27 MCL_E_INVALID_PARAMETER

```
#define MCL_E_INVALID_PARAMETER
```

All API's called with wrong read parameter shall return this error.

Definition at line 308 of file CDD_Mcl.h.

6.2.3.28 MCL_E_INVALID_STATE

```
#define MCL_E_INVALID_STATE
```

All API's called in wrong sequence shall return this error.

Definition at line 314 of file CDD_Mcl.h.

6.2.3.29 MCL_E_INCONSISTENCY

```
#define MCL_E_INCONSISTENCY
```

All API's called while hardware has error status shall return this error.

Definition at line 320 of file CDD_Mcl.h.

6.2.3.30 MCL_E_TIMEOUT

```
#define MCL_E_TIMEOUT
```

All API's called with a timeout value shall return this error if execution is not finished in the allocated timeframe.

Definition at line 327 of file CDD_Mcl.h.

6.2.3.31 MCL_E_PROTECTED

```
#define MCL_E_PROTECTED
```

If DET error reporting is enabled, the MCL will check if registers are protected.

Definition at line 333 of file CDD_Mcl.h.

6.2.3.32 MCL_E_INIT_FAILED

```
#define MCL_E_INIT_FAILED
```

If VariantPreCompile is used, the configuration pointer shall have a NULL_PTR value. If VariantPostBuild is used, the configuration pointer shall be different from NULL_PTR. And in case of violate will return MCL_E_INIT_↩ FAILED.

Definition at line 340 of file CDD_Mcl.h.

6.2.4 Enum Reference

6.2.4.1 Mcl_DmaInstanceCmdType

```
enum Mcl_DmaInstanceCmdType
```

This type contains the Mcl Dma Instance Commands.

The Commands trigger specific actions in the Dma Instance.

Enumerator

MCL_DMA_INST_STOP	The Stop Command stops the executing channel and forces the Minor Loop to finish.
MCL_DMA_INST_STOP_ERROR	The StopError Command stops the executing channel, forces the Minor Loop to finish and generates an error interrupt.
MCL_DMA_INST_PAUSE	The Pause Command allows the ongoing transfer to finish and pauses any new transfer.
MCL_DMA_INST_RESUME	The Resume Command allows the transfer to continue.

Definition at line 352 of file CDD_Mcl.h.

6.2.4.2 Mcl_DmaChannelCmdType

enum Mcl_DmaChannelCmdType

This type contains the Mcl Dma Channel Commands.

The Commands trigger specific actions in the Dma Channel.

Enumerator

MCL_DMA_CH_START_REQUEST	The Start Request Command enables the Dma Channel to be triggered by hardware requests.
MCL_DMA_CH_STOP_REQUEST	The Stop Request Command disables the Dma Channel to be triggered by hardware requests.
MCL_DMA_CH_START_SERVICE	The Start Service Command sends a start request to the Dma Channel.
MCL_DMA_CH_ACK_DONE	The Ack Done Command resets the Dma Channel Done status.
MCL_DMA_CH_ACK_ERROR	The Ack Error Command resets the Dma Channel Error status.

Definition at line 365 of file CDD_Mcl.h.

6.2.4.3 Mcl_DmaChannelGlobalParamType

enum Mcl_DmaChannelGlobalParamType

This type contains the Mcl Dma Channel Global Parameters.

The Parameters set specific functionalities.

Enumerator

MCL_DMA_CH_SET_EN_HARDWARE_REQ	[BOOLEAN] The EnRequest Parameter enables the Dma Channel Request.
MCL_DMA_CH_SET_EN_ERROR_INTERRUPT	[BOOLEAN] The EnError Parameter enables the Dma Channel Error Interrupt.
MCL_DMA_CH_SET_GROUP_PRIORITY	[VALUE] The Group Parameter sets the Dma Channel Group Priority.
MCL_DMA_CH_SET_LEVEL_PRIORITY	[VALUE] The Level Parameter sets the Dma Channel Level Priority.
MCL_DMA_CH_SET_EN_PREEMPTION_PRIORITY	[BOOLEAN] The EnPreemption Parameter enables the Dma Channel Preemption.
MCL_DMA_CH_SET_DIS_PREEMPT_PRIORITY	[BOOLEAN] The DisPreempt Parameter disables the Dma Channel Preempt.

Definition at line 379 of file CDD_Mcl.h.

6.2.4.4 Mcl_DmaChannelTransferParamType

enum Mcl_DmaChannelTransferParamType

This type contains the Mcl Dma Channel Transfer Parameters.

The Parameters set specific functionalities.

Enumerator

MCL_DMA_CH_SET_SOURCE_ADDRESS	[VALUE] The Source Address Parameter sets the Dma Channel source address value.
MCL_DMA_CH_SET_SOURCE_SIGNED_OFFSET	[VALUE] The Source Signed Offset Parameter sets the Dma Channel source signed offset value.
MCL_DMA_CH_SET_SOURCE_SIGNED_LAST_ADDR_ADJ	[VALUE] The Source Signed Last Address Adjustment Parameter sets the Dma Channel source signed last address adjustment.
MCL_DMA_CH_SET_SOURCE_TRANSFER_SIZE	[VALUE] The Source Transfer Size Parameter sets the Dma Channel source transfer size.
MCL_DMA_CH_SET_SOURCE_MODULO	[VALUE] The Source Modulo Parameter sets the Dma Channel source modulo.
MCL_DMA_CH_SET_DESTINATION_ADDRESS	[VALUE] The Destination Address Parameter sets the Dma Channel destination address value.
MCL_DMA_CH_SET_DESTINATION_SIGNED_OFFSET	[VALUE] The Destination Signed Offset Parameter sets the Dma Channel destination signed offset value.
MCL_DMA_CH_SET_DESTINATION_SIGNED_LAST_ADDR_ADJ	[VALUE] The Destination Signed Last Address Adjustment Parameter sets the Dma Channel destination signed last address adjustment.

Enumerator

MCL_DMA_CH_SET_DESTINATION_TRANSFER_SIZE	[VALUE] The Destination Transfer Size Parameter sets the Dma Channel destination transfer size.
MCL_DMA_CH_SET_DESTINATION_MODULO	[VALUE] The Destination Modulo Parameter sets the Dma Channel destination modulo.
MCL_DMA_CH_SET_MINORLOOP_EN_SRC_OFFSET	[BOOLEAN] The Minor Loop Enable Source Offset Parameter enables the Dma Channel minor loop source offset.
MCL_DMA_CH_SET_MINORLOOP_EN_DST_OFFSET	[BOOLEAN] The Minor Loop Enable Destination Offset Parameter enables the Dma Channel minor loop destination offset.
MCL_DMA_CH_SET_MINORLOOP_SIGNED_OFFSET	[VALUE] The Minor Loop Signed Offset Parameter sets the Dma Channel minor loop signed offset.
MCL_DMA_CH_SET_MINORLOOP_EN_LINK	[BOOLEAN] The Minor Loop Enable Link Parameter enables the Dma Channel minor loop logic channel linking.
MCL_DMA_CH_SET_MINORLOOP_LOGIC_LINK_CH	[VALUE] The Minor Loop Logic Channel Link Parameter sets the Dma Channel minor loop logic channel link.
MCL_DMA_CH_SET_MINORLOOP_SIZE	[VALUE] The Minor Loop Size Parameter sets the Dma Channel minor loop transfer size.
MCL_DMA_CH_SET_MAJORLOOP_EN_LINK	[BOOLEAN] The Major Loop Enable Link Parameter enables the Dma Channel major loop logic channel linking.
MCL_DMA_CH_SET_MAJORLOOP_LOGIC_LINK_CH	[VALUE] The Major Loop Logic Channel Link Parameter sets the Dma Channel major loop logic channel link.
MCL_DMA_CH_SET_MAJORLOOP_COUNT	[VALUE] The Major Loop Count Parameter sets the Dma Channel major loop count.
MCL_DMA_CH_SET_CONTROL_SOFTWARE_REQUEST	[BOOLEAN] The Enable Start Parameter enables the Dma Channel start service request.
MCL_DMA_CH_SET_CONTROL_EN_MAJOR_INTERRUPT	[BOOLEAN] The Enable Major Interrupt Parameter enables the Dma Channel major interrupt.
MCL_DMA_CH_SET_CONTROL_EN_HALF_MAJOR_INTERRUPT	[BOOLEAN] The Enable Half Interrupt Parameter enables the Dma Channel half major interrupt.
MCL_DMA_CH_SET_CONTROL_DIS_AUTO_REQUEST	[BOOLEAN] The Disable Automatic Request Parameter disables the Dma Channel automatic request.
MCL_DMA_CH_SET_CONTROL_BANDWIDTH	[VALUE] The Bandwidth Control Parameter sets the Dma Channel bandwidth control.

Definition at line 404 of file CDD_Mcl.h.

6.2.4.5 Mcl_DmaChannelStateType

```
enum Mcl_DmaChannelStateType
```

This type contains the Mcl Dma Channel State values.

The states represent the Channel status during runtime.

Definition at line 445 of file CDD_Mcl.h.

6.2.4.6 Mcl_DmaChannelInfoParamType

enum Mcl_DmaChannelInfoParamType

This type contains the Mcl Dma Channel Information Parameters.

The Parameters get specific information.

Enumerator

MCL_DMA_CH_GET_SOURCE_ADDRESS	[VALUE] The Source Address Parameter gets the Dma Channel source address.
MCL_DMA_CH_GET_DESTINATION_ADDR↩ ESS	[VALUE] The Destination Address Parameter gets the Dma Channel destination address.
MCL_DMA_CH_GET_BEGIN_ITER_COUNT	[VALUE] The Begin Iteration Count Parameter gets the Dma Channel begin iteration count.
MCL_DMA_CH_GET_CURRENT_ITER_COU↩ NT	[VALUE] The Current Iteration Count Parameter gets the Dma Channel current iteration count.
MCL_DMA_CH_GET_MAJOR_INTERRUPT	[BOOLEAN] The Major Interrupt Parameter gets the Dma Channel major interrupt.
MCL_DMA_CH_GET_HALF_MAJOR_INTER↩ RUPT	[BOOLEAN] The Half Major Interrupt Parameter gets the Dma Channel half major interrupt.

Definition at line 460 of file CDD_Mcl.h.

6.2.4.7 Mcl_CacheType

enum Mcl_CacheType

This type contains the Mcl Cache Type selection.

The Cache Types select specific cache memory types.

Enumerator

MCL_CACHE_LMEM	The Cache Lmem Parameter selects LMEM cache types.
MCL_CACHE_CORE	The Cache Core Parameter selects CORE cache types.

Definition at line 483 of file CDD_Mcl.h.

6.2.4.8 Mcl_CacheBusType

```
enum Mcl_CacheBusType
```

This type contains the Mcl Cache Bus Type selection.

The Cache Bus Types select Code and System caches and bus.

Enumerator

MCL_CACHE_PC_BUS	The Cache PC Bus selects Processor Code (PC) bus (used with Cache Lmem).
MCL_CACHE_PS_BUS	The Cache PS Bus selects Processor System (PS) bus (used with Cache Lmem).
MCL_CACHE_ALL_BUS	The Cache All Bus selects PC and PS bus (used with Cache Lmem).
MCL_CACHE_INSTRUCTION	The Cache Instruction Parameter selects instruction cache (used with Cache Core).
MCL_CACHE_DATA	The Cache Data Parameter selects data cache (used with Cache Core).

Definition at line 494 of file CDD_Mcl.h.

6.2.4.9 Mcl_ReturnType

```
enum Mcl_ReturnType
```

This type contains the Mcl Return Type.

The Return Type give information for the execution of interfaces.

Definition at line 308 of file Mcl_Types.h.

6.2.5 Function Reference

6.2.5.1 Mcl_Init()

```
void Mcl_Init (
            const Mcl_ConfigType *const ConfigPtr )
```

This function initializes the Mcl Driver.

This service is a non reentrant function that shall initialize the Mcl driver. The initialization is applied for the enabled IPs, configured statically.

Parameters

in	*ConfigPtr*	Pointer to the configuration structure.

Returns

void

6.2.5.2 Mcl_DeInit()

```
void Mcl_DeInit (
            void  )
```

This function deinitializes the Mcl Driver.

This service is a non reentrant function that shall deinitialize the Mcl driver. The deinitialization is applied for the enabled IPs, configured statically.

Returns

void

6.2.5.3 Mcl_SetDmaInstanceCommand()

```
void Mcl_SetDmaInstanceCommand (
            const uint32 Instance,
            const Mcl_DmaInstanceCmdType Command )
```

This function sets Dma Instance Command.

This service is a reentrant function that shall command the Dma Instance. The command shall trigger specific functionalities of the Dma Instance.

Parameters

in	*Instance*	Selection value of the Logic Instance.
in	*Command*	The command for the Logic Instance.

Returns

void

6.2.5.4 Mcl_GetDmaInstanceStatus()

```
void Mcl_GetDmaInstanceStatus (
            const uint32 Instance,
            Mcl_DmaInstanceStatusType *const Status )
```

This function gets Dma Instance Status.

This service is a reentrant function that shall get the Dma Instance status. The command shall read specific functionalities of the Dma Instance.

Parameters

in	*Instance*	Selection value of the Logic Instance.
out	*Status*	Pointer to the Dma Instance status.

Returns

void

6.2.5.5 Mcl_SetDmaChannelCommand()

```
void Mcl_SetDmaChannelCommand (
            const uint32 Channel,
            const Mcl_DmaChannelCmdType Command )
```

This function sets Dma Channel Command.

This service is a reentrant function that shall command the Dma Channel. The command shall trigger specific functionalities of the Dma Channel.

Parameters

in	*Channel*	Specifies the Logic Channel Tag defined by the user.
in	*Command*	The command for the Logic Channel.

Returns

void

6.2.5.6 Mcl_GetDmaChannelStatus()

```
void Mcl_GetDmaChannelStatus (
            const uint32 Channel,
            Mcl_DmaChannelStatusType *const Status )
```

This function gets Dma Channel Status.

This service is a reentrant function that shall get the Dma Channel status. The command shall read specific functionalities of the Dma Channel.

Parameters

in	*Channel*	Specifies the Logic Channel Tag defined by the user.
out	*Status*	Pointer to the Dma Channel status.

Returns

void

6.2.5.7 Mcl_SetDmaChannelGlobalList()

```
void Mcl_SetDmaChannelGlobalList (
            const uint32 Channel,
            const Mcl_DmaChannelGlobalListType List[],
            const uint32 ListDimension )
```

This function sets Dma Channel Global List settings.

This service is a reentrant function that shall set the Dma Channel global parameters list. The list is composed of an array of Dma Channel global parameters settings. The settings list(array) is defined by the user needs: it contains the desired parameters to be configured, in any desired order.

How to use this interface:

1. Use the "Mcl_DmaChannelGlobalListType" to create a list(array) with the desired paramaters to configure (see parameters: "Mcl_DmaChannelGlobalParamType") The list can declared globally or locally: Global example: Mcl_DmaChannelGlobalListType global_Mcl_DmaChannelGlobalList0[NUMBER_OF_PARAMETERS] = {...}; Local example: Mcl_DmaChannelGlobalListType Mcl_DmaChannelGlobalList[NUMBER_OF_PAR↩ AMETERS]; Mcl_DmaChannelGlobalList[PARAMETER0].Param = MCL_DMA_CH_SET_EN_BUFF↩ ERED_WRITES; Mcl_DmaChannelGlobalList[PARAMETER0].Value = TRUE; Mcl_DmaChannelGlobal↩ List[PARAMETER1].Param = ...; Mcl_DmaChannelGlobalList[PARAMETER1].Value = ...;
2. Call the "Mcl_SetDmaChannelGlobalList()" interface: Mcl_SetDmaChannelGlobalList(LOGIC_CHANNE↩ Lx, Mcl_DmaChannelGlobalList, NUMBER_OF_PARAMETERS);

Parameters

in	*Channel*	Specifies the Logic Channel Tag defined by the user.
in	*List*	Pointer to the Global List Array.
in	*ListDimension*	Number of entries in the List.

Returns

void

6.2.5.8 Mcl_SetDmaChannelTransferList()

```
void Mcl_SetDmaChannelTransferList (
            const uint32 Channel,
            const Mcl_DmaChannelTransferListType List[],
            const uint32 ListDimension )
```

This function sets Dma Channel Transfer List settings.

This service is a reentrant function that shall set the Dma Channel transfer parameters list. The list is composed of an array of Dma Channel transfer parameters settings. The settings array is defined by the user needs: it contains entries for each desired parameter to be configured, in any suitable order.

How to use this interface:

1. Use the "Mcl_DmaChannelTransferListType" to create a list(array) with the desired paramaters to configure (see parameters: "Mcl_DmaChannelTransferParamType") The list can declared globally or locally: Global example: Mcl_DmaChannelTransferListType global_Mcl_DmaChannelTransferList0[NUMBER_O↩ F_PARAMETERS] = {...}; Local example: Mcl_DmaChannelTransferListType Mcl_DmaChannelTransfer↩ List[NUMBER_OF_PARAMETERS]; Mcl_DmaChannelTransferList[PARAMETER0].Param = MCL_D↩ MA_CH_SET_SOURCE_ADDRESS; Mcl_DmaChannelTransferList[PARAMETER0].Value = Mcl_Dma↩ ChannelTransferList[PARAMETER1].Param = MCL_DMA_CH_SET_DESTINATION_ADDRESS; Mcl↩ _DmaChannelTransferList[PARAMETER1].Value =
2. Call the "Mcl_SetDmaChannelTransferList()" interface: Mcl_SetDmaChannelTransferList(LOGIC_CHAN↩ NELx, Mcl_DmaChannelTransferList, NUMBER_OF_PARAMETERS);

Parameters

in	*Channel*	Specifies the Logic Channel Tag defined by the user.
in	*List*	Pointer to the Transfer List Array.
in	*ListDimension*	Number of entries in the List.

Returns

void

6.2.5.9 Mcl_SetDmaChannelScatterGatherList()

```
void Mcl_SetDmaChannelScatterGatherList (
            const uint32 Channel,
            const uint32 Element,
            const Mcl_DmaChannelScatterGatherListType List[],
            const uint32 ListDimension )
```

This function sets Dma Channel Scatter/Gather List settings.

This service is a reentrant function that shall set the Dma Channel scatter/gather parameters list. The Scatter/↩ Gather List configures Logic Elements belonging to the same Dma Logic Channel. The settings array is defined by the user needs: it contains entries for each desired parameter to be configured, in any suitable order.

How to use this interface:

1. Use the "Mcl_DmaChannelScatterGatherListType" to create a list(array) with the desired paramaters to configure (see parameters: "Mcl_DmaChannelTransferParamType") The list can declared globally or locally: Global example: Mcl_DmaChannelScatterGatherListType global_Mcl_DmaChannelScatterGatherList0[N↩ UMBER_OF_PARAMETERS] = {...}; Local example: Mcl_DmaChannelScatterGatherListType Mcl↩ _DmaChannelScatterGatherList[NUMBER_OF_PARAMETERS]; Mcl_DmaChannelScatterGatherList[P↩ ARAMETER0].Param = MCL_DMA_CH_SET_SOURCE_ADDRESS; Mcl_DmaChannelScatterGather↩ List[PARAMETER0].Value = Mcl_DmaChannelScatterGatherList[PARAMETER1].Param = MCL_DMA↩ _CH_SET_DESTINATION_ADDRESS; Mcl_DmaChannelScatterGatherList[PARAMETER1].Value =
2. Call the "Mcl_SetDmaChannelScatterGatherList()" interface: Mcl_SetDmaChannelScatterGatherList(LO↩ GIC_CHANNELx, LOGIC_ELEMENTy, Mcl_DmaChannelScatterGatherList, NUMBER_OF_PARAM↩ ETERS);

Parameters

in	*Channel*	Specifies the Logic Channel Tag defined by the user.
in	*Element*	Specifies the Logic Element Id.
in	*List*	Pointer to the Scatter/Gather List Array.
in	*ListDimension*	Number of entries in the List.

Returns

void

6.2.5.10 Mcl_GetDmaChannelParam()

```
void Mcl_GetDmaChannelParam (
            const uint32 Channel,
            const Mcl_DmaChannelInfoParamType Param,
            uint32 *const Value )
```

This function gets the Dma Channel Parameter value.

This service is a reentrant function that shall get the Dma Channel parameters value.

Parameters

in	*Channel*	Specifies the Logic Channel Tag defined by the user.
in	*Param*	Selection parameter.
out	*Value*	Pointer to the parameter value.

Returns

void

6.2.5.11 Mcl_SetDmaChannelScatterGatherConfig()

```
void Mcl_SetDmaChannelScatterGatherConfig (
            const uint32 Channel,
            const uint32 Element )
```

This function configures the Dma Channel Scatter/Gather.

This service is a reentrant function that shall configure the Dma Channel scatter/gather functionality. The Scatter/↩ Gather settings, for the specified Dma Logic Channel, are loaded into the Software TCDs. Each software T↩ CD corresponds to a Logic Element. The specified Logic Element shall be loaded into the Dma Logic Channel's Hardware TCD. The Logic Elements (describing the Software TCDs) form a simple chained list, the "Element" function parameter representing the lists's head.

Parameters

in	*Channel*	Specifies the Logic Channel Tag defined by the user.
in	*Element*	Specifies the Logic Element Id representing the list's head.

Returns

void

6.2.5.12 Mcl_SetTrgMuxInput()

```
void Mcl_SetTrgMuxInput (
            const uint32 Trigger,
            const uint32 Input )
```

This function sets the Trgmux Trigger Input selection.

This service is a reentrant function that shall configure the Trgmux Trigger functionality.

Parameters

in	*Trigger*	Selection value of the Logic Trigger.
in	*Input*	Selection value for the Logic Trigger's Input.

Returns

void

6.2.5.13 Mcl_SetTrgMuxLock()

```
void Mcl_SetTrgMuxLock (
            const uint32 Trigger )
```

This function sets the Trgmux Trigger Lock.

This service is a reentrant function that shall configure the Trgmux Trigger Lock functionality.

Parameters

in	*Trigger*	Selection value of the Logic Trigger.

Returns

void

6.2.5.14 Mcl_CacheEnable()

```
void Mcl_CacheEnable (
            Mcl_CacheType CacheType,
            Mcl_CacheBusType BusType )
```

This function enables the Cache.

This service is a reentrant function that shall enable the Cache functionality.

Parameters

in	*CacheType*	Selection value of the Cache Type.
in	*BusType*	Selection value of the Bus Type.

Returns

void

6.2.5.15 Mcl_CacheDisable()

```
void Mcl_CacheDisable (
            Mcl_CacheType CacheType,
            Mcl_CacheBusType BusType )
```

This function disables the Cache.

This service is a reentrant function that shall disable the Cache functionality.

Parameters

in	*CacheType*	Selection value of the Cache Type.
in	*BusType*	Selection value of the Bus Type.

Returns

void

6.2.5.16 Mcl_CacheInvalidate()

```
void Mcl_CacheInvalidate (
            Mcl_CacheType CacheType,
            Mcl_CacheBusType BusType )
```

This function Invalidates the Cache.

This service is a reentrant function that shall Invalidate the Cache functionality. The Invalidation applies to the entire Cache.

Parameters

in	*CacheType*	Selection value of the Cache Type.
in	*BusType*	Selection value of the Bus Type.

Returns

void

6.2.5.17 Mcl_CacheClean()

```
void Mcl_CacheClean (
            Mcl_CacheType CacheType,
            Mcl_CacheBusType BusType,
            boolean EnInvalidate )
```

This function Cleans the Cache.

This service is a reentrant function that shall Clean the Cache functionality. The Clean applies to the entire Cache. By enabling the Invalidation, the function shall execute specific Cache Clean&Invalidate function.

Parameters

in	*CacheType*	Selection value of the Cache Type.
in	*BusType*	Selection value of the Bus Type.
in	*EnInvalidate*	Enable the Invalidation specific functionality.

Returns

void

6.2.5.18 Mcl_CacheInvalidateByAddr()

```
void Mcl_CacheInvalidateByAddr (
            Mcl_CacheType CacheType,
            Mcl_CacheBusType BusType,
            uint32 Addr,
            uint32 Length )
```

This function Invalidates the Cache by address.

This service is a reentrant function that shall Invalidate the Cache by address. The Invalidation applies to the area in Cache specified by the address and length. The buffer shall be aligned to the Cache Line size.

Parameters

in	*CacheType*	Selection value of the Cache Type.
in	*BusType*	Selection value of the Bus Type.
in	*Addr*	Address value of the buffer.
in	*Length*	Length value of the buffer.

Returns

void

6.2.5.19 Mcl_CacheCleanByAddr()

```
void Mcl_CacheCleanByAddr (
            Mcl_CacheType CacheType,
            Mcl_CacheBusType BusType,
            boolean EnInvalidate,
            uint32 Addr,
            uint32 Length )
```

This function Cleans the Cache by address.

This service is a reentrant function that shall Clean the Cache by address. The Clean applies to the area in Cache specified by the address and length. By enabling the Invalidation, the function shall execute specific Cache Clean&Invalidate function.

Parameters

in	*CacheType*	Selection value of the Cache Type.
in	*BusType*	Selection value of the Bus Type.
in	*EnInvalidate*	Enable the Invalidation specific functionality.
in	*Addr*	Address value of the buffer.
in	*Length*	Length value of the buffer.

Returns

void

6.2.5.20 Mcl_SelectCommonTimebase()

```
void Mcl_SelectCommonTimebase (
            uint8 instance,
            uint16 elementSyncList )
```

Implementation specific function to updates the Global Timebase bits of configured modules.

This function is used to set the global timebase bits for modules that support the global timebase feature. The function selects the module that gives the common timebase and the modules that are use this timebase (as bits in elementSyncList). Then it synchronizes the modules. example: elementSyncList is 0x0003 - modules 0 and 1 use the timebase given by instance elementSyncList is 0x0005 - modules 0 and 2 use the timebase given by instance

Parameters

in	*instance*	FTM module id
in	*elementSyncList*	FTM module mask value

Returns

void

6.2.5.21 Mcl_GetVersionInfo()

```
void Mcl_GetVersionInfo (
            Std_VersionInfoType *const VersionInfo )
```

Returns the version information of this module.

Returns the version information of MCL module.

Parameters

out	*VersionInfo*	A pointer to a structure used to get version information.

Returns

void

6.3 DMA IP Driver

6.3.1 Detailed Description

Data Structures

- struct Dma_Ip_LogicChannelGlobalListType

  This type contains the Dma Ip Logic Channel Global List. *More...*
- struct Dma_Ip_LogicChannelTransferListType

  This type contains the Dma Ip Channel Transfer List. *More...*
- struct Dma_Ip_LogicInstanceStatusType

  This type contains the Dma Ip Instance Status. *More...*
- struct Dma_Ip_LogicChannelStatusType

  This type contains the Dma Ip Channel Status. *More...*
- struct Dma_Ip_LogicChannelIdType

  This type contains the Dma Ip Logic Channel Identification. *More...*
- struct Dma_Ip_GlobalConfigType

  This type contains the Dma Ip Global Configuration. *More...*
- struct Dma_Ip_TransferConfigType

  This type contains the Dma Ip Transfer Configuration. *More...*
- struct Dma_Ip_ScatterGatherConfigType

  This type contains the Dma Ip Scatter/Gather Configuration. *More...*
- struct Dma_Ip_LogicChannelConfigType

  This type contains the Dma Ip Logic Channel Configuration. *More...*
- struct Dma_Ip_LogicInstanceIdType

  This type contains the Dma Ip Logic Instance Identification. *More...*
- struct Dma_Ip_LogicInstanceConfigType

  This type contains the Dma Ip Logic Instance Configuration. *More...*
- struct Dma_Ip_HwChannelStateType

  This type contains the Dma Ip Hardware Channel State. *More...*
- struct Dma_Ip_VirtualSectionConfigType

  This type contains the Dma Ip Virtual memory section configuration Type. *More...*
- struct Dma_Ip_VirtualMemoryConfigType

  This type contains the Dma Ip Virtual memory configuration Type. *More...*
- struct Dma_Ip_InitType

  This type contains the Dma Ip Initialization. *More...*

Types Reference

- typedef void(* Dma_Ip_Callback) (void)

  This type contains the Dma Ip Callback interface.

Enum Reference

- enum Dma_Ip_LogicInstanceCmdType

  This type contains the Dma Ip Logic Instance Commands.
- enum Dma_Ip_LogicChannelCmdType

  This type contains the Dma Ip Logic Channel Commands.
- enum Dma_Ip_LogicChannelGlobalParamType

  This type contains the Dma Ip Logic Channel Global Parameters.
- enum Dma_Ip_LogicChannelTransferParamType

  This type contains the Dma Ip Logic Channel Transfer Parameters.
- enum Dma_Ip_LogicChannelInfoParamType

  This type contains the Dma Ip Logic Channel Information Parameters.
- enum Dma_Ip_ReturnType

  This type contains the Dma Ip Return Type.
- enum Dma_Ip_HwChannelStateValueType

  This type contains the Dma Ip Channel State Value Type.

Function Reference

- Dma_Ip_ReturnType Dma_Ip_Init (const Dma_Ip_InitType *const DmaInit)

  This function initializes the Dma Ip Driver.
- Dma_Ip_ReturnType Dma_Ip_Deinit (void)

  This function deinitializes the Dma Ip Driver.
- Dma_Ip_ReturnType Dma_Ip_SetLogicInstanceCommand (const uint32 LogicInst, const Dma_Ip_LogicInstanceCmdTy Command)

  This function sets Dma Ip Instance Command.
- Dma_Ip_ReturnType Dma_Ip_GetLogicInstanceStatus (const uint32 LogicInst, Dma_Ip_LogicInstanceStatusType *const Status)

  This function gets Dma Ip Instance Status.
- Dma_Ip_ReturnType Dma_Ip_LogicChannelInit (const uint32 LogicCh)

  This function initializes the Dma Ip Logic Channel.
- Dma_Ip_ReturnType Dma_Ip_LogicChannelDeinit (const uint32 LogicCh)

  This function deinitializes the Dma Ip Logic Channel.
- Dma_Ip_ReturnType Dma_Ip_SetLogicChannelCommand (const uint32 LogicCh, const Dma_Ip_LogicChannelCmdTyp Command)

  This function sets Dma Ip Logic Channel Command.
- Dma_Ip_ReturnType Dma_Ip_GetLogicChannelStatus (const uint32 LogicCh, Dma_Ip_LogicChannelStatusType *const ChStatus)

  This function gets Dma Ip Logic Channel Status.
- Dma_Ip_ReturnType Dma_Ip_SetLogicChannelGlobalList (const uint32 LogicCh, const Dma_Ip_LogicChannelGlobalLi List[], const uint32 ListDimension)

  This function sets Dma Ip Logic Channel Global List settings.
- Dma_Ip_ReturnType Dma_Ip_SetLogicChannelTransferList (const uint32 LogicCh, const Dma_Ip_LogicChannelTransfe List[], const uint32 ListDimension)

  This function sets Dma Ip Logic Channel Transfer List settings.
- Dma_Ip_ReturnType Dma_Ip_SetLogicChannelScatterGatherList (const uint32 LogicCh, const uint32 Element, const Dma_Ip_LogicChannelScatterGatherListType List[], const uint32 ListDimension)

This function sets Dma Ip Logic Channel Scatter/Gather List settings.

- Dma_Ip_ReturnType Dma_Ip_GetLogicChannelParam (const uint32 LogicCh, const Dma_Ip_LogicChannelInfoParamT Param, uint32 *const Value)

  This function gets the Dma Ip Logic Channel Parameter value.

- Dma_Ip_ReturnType Dma_Ip_SetLogicChannelScatterGatherConfig (const uint32 LogicCh, const uint32 Element)

  This function configures the Dma Ip Logic Channel Scatter/Gather.

6.3.2 Data Structure Documentation

6.3.2.1 struct Dma_Ip_LogicChannelGlobalListType

This type contains the Dma Ip Logic Channel Global List.

The Dma Ip Channel Global List contains a pair composed from Dma Channel Global Parameter Type and the Value of the parameter. The Dma Ip Channel Global Parameter Type selects a parameter form the Global Parameter enum type. The Value stores the parameter's value.

Definition at line 234 of file Dma_Ip.h.

6.3.2.2 struct Dma_Ip_LogicChannelTransferListType

This type contains the Dma Ip Channel Transfer List.

The Dma Ip Channel Transfer List contains a pair composed from Dma Channel Transfer Parameter Type and the Value of the parameter. The Dma Ip Channel Transfer Parameter Type selects a parameter form the Transfer Parameter enum type. The Value stores the parameter's value.

This type contains the Dma Ip Channel ScatterGather List.

The Dma Ip Channel Transfer List contains a pair composed from Dma Channel ScatterGather Parameter Type and the Value of the parameter. The Dma Ip Channel ScatterGather Parameter Type selects a parameter form the ScatterGather Parameter enum type. The Value stores the parameter's value.

Definition at line 257 of file Dma_Ip.h.

6.3.2.3 struct Dma_Ip_LogicInstanceStatusType

This type contains the Dma Ip Instance Status.

The Dma Ip Instance Status contains the Hardware Errors, Active Id and Active indication for the running Dma Channel. The Errors shall contain the Hardware Errors. The ActiveId shall contain the running Dma Channel Id. The Active shall contain the running Dma Channel Active status.

Definition at line 273 of file Dma_Ip.h.

Data Fields

- uint32 Errors

  [VALUE] The Errors value is read from the DMA Instance Error Register (ES) as it is.
- uint8 ActiveId

  [VALUE] The ActiveId value is read from the DMA Instance Control Register (CR) field ACTIVE_ID.
- boolean Active

  [BOOLEAN] The Active value is read from the DMA Instance Control Register (CR) field ACTIVE.
- uint8 Version

  [VALUE] The Version value is read from the DMA Instance Control Register (CR) field VER.

6.3.2.3.1 Field Documentation

6.3.2.3.1.1 Errors `uint32 Errors`

[VALUE] The Errors value is read from the DMA Instance Error Register (ES) as it is.

Definition at line 274 of file Dma_Ip.h.

6.3.2.3.1.2 ActiveId `uint8 ActiveId`

[VALUE] The ActiveId value is read from the DMA Instance Control Register (CR) field ACTIVE_ID.

Definition at line 275 of file Dma_Ip.h.

6.3.2.3.1.3 Active `boolean Active`

[BOOLEAN] The Active value is read from the DMA Instance Control Register (CR) field ACTIVE.

Definition at line 276 of file Dma_Ip.h.

6.3.2.3.1.4 Version `uint8 Version`

[VALUE] The Version value is read from the DMA Instance Control Register (CR) field VER.

Definition at line 277 of file Dma_Ip.h.

6.3.2.4 struct Dma_Ip_LogicChannelStatusType

This type contains the Dma Ip Channel Status.

The Dma Ip Channel Status contains the Hardware Errors, Active status and Done indication for the running Dma Channel. The Channel State Value shall contain the internal driver state of the Dma Channel. The Errors shall contain the Hardware Dma Channel Errors. The Active shall contain the running Dma Channel Id. The Done shall contain the running Dma Channel Active status.

Definition at line 291 of file Dma_Ip.h.

Data Fields

- Dma_Ip_HwChannelStateValueType ChStateValue

  [VALUE] The ChStateValue value is read from the internal DMA Driver Channel State Machine. Check UM for additional information.

- uint32 Errors

  [VALUE] The Errors value is read from the DMA Channel Error Register (CHx_ES) as it is.
- boolean Active

  [BOOLEAN] The Active value is read from the DMA Channel Control and Status Register (CHx_CSR) field ACTIVE.
- boolean Done

  [BOOLEAN] The Active value is read from the DMA Channel Control and Status Register (CHx_CSR) field DONE.

6.3.2.4.1 Field Documentation

6.3.2.4.1.1 ChStateValue `Dma_Ip_HwChannelStateValueType ChStateValue`

[VALUE] The ChStateValue value is read from the internal DMA Driver Channel State Machine. Check UM for additional information.

Definition at line 292 of file Dma_Ip.h.

6.3.2.4.1.2 Errors `uint32 Errors`

[VALUE] The Errors value is read from the DMA Channel Error Register (CHx_ES) as it is.

Definition at line 294 of file Dma_Ip.h.

6.3.2.4.1.3 Active `boolean Active`

[BOOLEAN] The Active value is read from the DMA Channel Control and Status Register (CHx_CSR) field ACTIVE.

Definition at line 295 of file Dma_Ip.h.

6.3.2.4.1.4 Done `boolean Done`

[BOOLEAN] The Active value is read from the DMA Channel Control and Status Register (CHx_CSR) field DONE.

Definition at line 296 of file Dma_Ip.h.

6.3.2.5 struct Dma_Ip_LogicChannelIdType

This type contains the Dma Ip Logic Channel Identification.

The Logic Channel is identified by the following structure: The Logic Channel Id contains the ID value. The Hardware Version Id contains the DMA Hardware Ip Block Version. The Hardware Instance Id contains the DMA Hardware Instance identification. The Hardware Channel Id contains the DMA Hardware Channel identification. The Hardware Crc Instance Id contains the DMA Hardware Crc Instance identification if CRC_IP is available. The Hardware Crc Channel Id contains the DMA Hardware Crc Channel identification if CRC_IP is available. The Interrupt Callback stores pointer to the user defined interrupt callback. The Error Interrupt Callback stores pointer to the user defined error interrupt callback.

Definition at line 179 of file Dma_Ip_Types.h.

Data Fields

- uint32 LogicChId

  DMA logic channel number.
- uint8 HwVersId

  DMA hardware version.
- uint8 HwInstId

  DMA hardware instance number.
- uint8 HwChId

  DMA hardware channel number.
- Dma_Ip_Callback IntCallback

  The channel callback is installed in the interrupt and is called automatically from the interrupt every time it is triggered.
- Dma_Ip_Callback ErrIntCallback

  The channel error callback is installed in the error interrupt and is called automatically from the interrupt every time it is triggered.

6.3.2.5.1 Field Documentation

6.3.2.5.1.1 LogicChId `uint32 LogicChId`

DMA logic channel number.

Definition at line 180 of file Dma_Ip_Types.h.

6.3.2.5.1.2 HwVersId `uint8 HwVersId`

DMA hardware version.

Definition at line 181 of file Dma_Ip_Types.h.

6.3.2.5.1.3 HwInstId `uint8 HwInstId`

DMA hardware instance number.

Definition at line 182 of file Dma_Ip_Types.h.

6.3.2.5.1.4 HwChId `uint8 HwChId`

DMA hardware channel number.

Definition at line 183 of file Dma_Ip_Types.h.

6.3.2.5.1.5 IntCallback `Dma_Ip_Callback IntCallback`

The channel callback is installed in the interrupt and is called automatically from the interrupt every time it is triggered.

Definition at line 184 of file Dma_Ip_Types.h.

6.3.2.5.1.6 ErrIntCallback `Dma_Ip_Callback ErrIntCallback`

The channel error callback is installed in the error interrupt and is called automatically from the interrupt every time it is triggered.

Definition at line 186 of file Dma_Ip_Types.h.

6.3.2.6 struct Dma_Ip_GlobalConfigType

This type contains the Dma Ip Global Configuration.

The Global Configuration of an Logic Channel, contains all the information describing a channel, but are not present in the DMA TCD. It contains configuration split in four main areas: Control defines functionality covering the channel's bus control. Request defines functionality covering the channel's request interface Interrupt defines functionality covering special channel interrupts. Priority defines functionality covering the channel's priority mechanism.

Definition at line 202 of file Dma_Ip_Types.h.

6.3.2.7 struct Dma_Ip_TransferConfigType

This type contains the Dma Ip Transfer Configuration.

The Transfer Configuration of an Logic Channel, contains all the information describing a channel transfer functionalities and are present in the DMA TCD. It contains configuration split in five main areas: Control defines functionality covering the channel control functions. It includes additionaly the Scatter/Gather Address and Destination Store Address.

Definition at line 235 of file Dma_Ip_Types.h.

6.3.2.8 struct Dma_Ip_ScatterGatherConfigType

This type contains the Dma Ip Scatter/Gather Configuration.

The Scatter/Gather Configuration of a Logic Channel, contains all the information describing the channel's needed resources for a Scatter/Gather element. It contains a pointer to the transfer configuration, a pointer to a Software TCD, a pointer to the next Scatter/Gather configuration and a status flag for loading the transfer configuration into the Software TCD.

Definition at line 294 of file Dma_Ip_Types.h.

Data Fields

Type	Name	Description
Dma_Ip_TransferConfigType *	TransferConfig	Configuration that shall be loaded into the Software TCD.
Dma_Ip_SwTcdRegType *	Stcd	Software TCD shall be loaded with own TCD configuration.
boolean	Loaded	
struct Dma_Ip_ScatterGatherConfigType *	NextConfig	Next Logic Channel configuration. If the address is not NULL, then the ESG flag is enabled (Scatter/Gather address) and the destination adjustment is disabled. The next configuration address is stored.

6.3.2.9 struct Dma_Ip_LogicChannelConfigType

This type contains the Dma Ip Logic Channel Configuration.

The Logic Channel Configuration consists of the Logic Channel Identifier, pointer to the Global Configuration, pointer to the Transfer Configuration and pointer to the Scatter/Gather configuration. The Logic Channel Configuration contains all data needed to initialize a Logic Channel.

Definition at line 313 of file Dma_Ip_Types.h.

Data Fields

Type	Name	Description
Dma_Ip_LogicChannelIdType	LogicChId	The Logic Channel ID contains configuration information and identification.
const Dma_Ip_GlobalConfigType *	pxGlobalConfig	The Global Configuration pointer shall contain the global data.
const Dma_Ip_TransferConfigType *	pxTransferConfig	The Transfer Configuration pointer shall contain the transfer data.
Dma_Ip_ScatterGatherConfigType **	ppxScatterGatherConfigArray	The Scatter/Gather Configuration pointer shall contain a pointer to an array containing all Scatter/Gather Logic Elements.

6.3.2.10 struct Dma_Ip_LogicInstanceIdType

This type contains the Dma Ip Logic Instance Identification.

The Logic Instance is identified by the following structure: The Logic Instance Id contains the ID value. The Hardware Version Id contains the DMA Hardware Ip Block Version. The Hardware Instance Id contains the DMA Hardware Instance identification.

Definition at line 329 of file Dma_Ip_Types.h.

Data Fields

Type	Name	Description
uint32	LogicInstId	DMA logic instance number.
uint8	HwVersId	DMA hardware version number.
uint8	HwInstId	DMA hardware instance number.

6.3.2.11 struct Dma_Ip_LogicInstanceConfigType

This type contains the Dma Ip Logic Instance Configuration.

The Logic Instance Configuration contains all the information describing an instance functionality.

Definition at line 341 of file Dma_Ip_Types.h.

Data Fields

Type	Name	Description
Dma_Ip_LogicInstanceIdType	LogicInstId	DMA logic instance number.
boolean	EnDebug	When in debug mode, the DMA stalls the start of a new channel. Executing channels are allowed to complete. DMA resumes channel execution when the system exits debug mode or clears the EDBG field to 0.
boolean	EnRoundRobin	Enable Round Robin Channel Arbitration.
boolean	EnHaltAfterError	When this field is set to 1, any error causes the HALT field to be set to 1. Then all service requests are ignored until the HALT field is cleared to 0.
boolean	EnChLinking	Global Channel Linking Control.
boolean	EnGlMasterIdReplication	If master ID replication is disabled, the privileged protection level (Supervisor mode) for DMA transfers is used.

6.3.2.12 struct Dma_Ip_HwChannelStateType

This type contains the Dma Ip Hardware Channel State.

The Hardware Channel State contains the channel's state based on runtime actions. The structure links the hardware channel state with the Logic Channel.

Definition at line 359 of file Dma_Ip_Types.h.

6.3.2.13 struct Dma_Ip_VirtualSectionConfigType

This type contains the Dma Ip Virtual memory section configuration Type.

Definition at line 383 of file Dma_Ip_Types.h.

6.3.2.14 struct Dma_Ip_VirtualMemoryConfigType

This type contains the Dma Ip Virtual memory configuration Type.

Definition at line 395 of file Dma_Ip_Types.h.

6.3.2.15 struct Dma_Ip_InitType

This type contains the Dma Ip Initialization.

The Dma Ip Initialization contains all the information required to initialize the Dma Peripheral. Each pointer shall be loaded with a specific configuration used be the Dma.

Definition at line 409 of file Dma_Ip_Types.h.

6.3.3 Types Reference

6.3.3.1 Dma_Ip_Callback

```
typedef void(* Dma_Ip_Callback) (void)
```

This type contains the Dma Ip Callback interface.

The Callback is defined by the user and installed by the driver in the corresponding IRQ.

Returns

void

Definition at line 163 of file Dma_Ip_Types.h.

6.3.4 Enum Reference

6.3.4.1 Dma_Ip_LogicInstanceCmdType

```
enum Dma_Ip_LogicInstanceCmdType
```

This type contains the Dma Ip Logic Instance Commands.

The Commands trigger specific actions in the Dma Logic Instance.

Enumerator

DMA_IP_INST_CANCEL_TRANSFER	The Cancel Transfer cancels the executing channel and forces the Minor Loop to finish.
DMA_IP_INST_CANCEL_TRANSFER_WITH↩ _ERROR	The Cancel Transfer With Error Command cancels the executing channel, forces the Minor Loop to finish and generates an error interrupt.
DMA_IP_INST_HALT	The Halt Command allows the ongoing transfer to finish and halts any new transfer.
DMA_IP_INST_RESUME	The Resume Command allows the transfer to continue.

Definition at line 105 of file Dma_Ip.h.

6.3.4.2 Dma_Ip_LogicChannelCmdType

enum Dma_Ip_LogicChannelCmdType

This type contains the Dma Ip Logic Channel Commands.

The Commands trigger specific actions in the Dma Ip Logic Channel.

Enumerator

DMA_IP_CH_SET_HARDWARE_REQUEST	The Set Hardware Request Command enables the Dma Channel to be triggered by hardware requests.
DMA_IP_CH_CLEAR_HARDWARE_REQUEST	The Clear Hardware Request Command disables the Dma Channel to be triggered by hardware requests.
DMA_IP_CH_SET_SOFTWARE_REQUEST	The Set Software Request Command sends a soft start request to the Dma Channel.
DMA_IP_CH_CLEAR_DONE	The Clear Done Command resets the Dma Channel Done status.
DMA_IP_CH_CLEAR_ERROR	The Clear Error Command resets the Dma Channel Error status.

Definition at line 117 of file Dma_Ip.h.

6.3.4.3 Dma_Ip_LogicChannelGlobalParamType

enum Dma_Ip_LogicChannelGlobalParamType

This type contains the Dma Ip Logic Channel Global Parameters.

The Parameters set specific functionalities for the Dma Ip Logic Channel.

Enumerator

DMA_IP_CH_SET_EN_HARDWARE_REQ	[BOOLEAN] The EnRequest Parameter enables the Dma Channel Request.
DMA_IP_CH_SET_EN_ERROR_INTERRUPT	[BOOLEAN] The EnError Parameter enables the Dma Channel Error Interrupt.
DMA_IP_CH_SET_GROUP_PRIORITY	[VALUE] The Group Parameter sets the Dma Channel Group Priority.
DMA_IP_CH_SET_LEVEL_PRIORITY	[VALUE] The Level Parameter sets the Dma Channel Level Priority.
DMA_IP_CH_SET_EN_PREEMPTION_PRIO↩ RITY	[BOOLEAN] The EnPreemption Parameter enables the Dma Channel Preemption.
DMA_IP_CH_SET_DIS_PREEMPT_PRIORITY	[BOOLEAN] The DisPreempt Parameter disables the Dma Channel Preempt.

Definition at line 130 of file Dma_Ip.h.

6.3.4.4 Dma_Ip_LogicChannelTransferParamType

enum Dma_Ip_LogicChannelTransferParamType

This type contains the Dma Ip Logic Channel Transfer Parameters.

The Parameters set specific functionalities.

Enumerator

DMA_IP_CH_SET_SOURCE_ADDRESS	[VALUE] The Source Address Parameter sets the Dma Channel source address value.
DMA_IP_CH_SET_SOURCE_SIGNED_OFFSET	[VALUE] The Source Signed Offset Parameter sets the Dma Channel source signed offset value.
DMA_IP_CH_SET_SOURCE_SIGNED_LAST↩_ADDR_ADJ	[VALUE] The Source Signed Last Address Adjustment Parameter sets the Dma Channel source signed last address adjustment.
DMA_IP_CH_SET_SOURCE_TRANSFER_SIZE	[VALUE] The Source Transfer Size Parameter sets the Dma Channel source transfer size.
DMA_IP_CH_SET_SOURCE_MODULO	[VALUE] The Source Modulo Parameter sets the Dma Channel source modulo.
DMA_IP_CH_SET_DESTINATION_ADDRESS	[VALUE] The Destination Address Parameter sets the Dma Channel destination address value.
DMA_IP_CH_SET_DESTINATION_SIGNED_↩OFFSET	[VALUE] The Destination Signed Offset Parameter sets the Dma Channel destination signed offset value.
DMA_IP_CH_SET_DESTINATION_SIGNED_↩LAST_ADDR_ADJ	[VALUE] The Destination Signed Last Address Adjustment Parameter sets the Dma Channel destination signed last address adjustment.
DMA_IP_CH_SET_DESTINATION_TRANSFE↩R_SIZE	[VALUE] The Destination Transfer Size Parameter sets the Dma Channel destination transfer size.
DMA_IP_CH_SET_DESTINATION_MODULO	[VALUE] The Destination Modulo Parameter sets the Dma Channel destination modulo.
DMA_IP_CH_SET_MINORLOOP_EN_SRC_O↩FFSET	[BOOLEAN] The Minor Loop Enable Source Offset Parameter enables the Dma Channel minor loop source offset.
DMA_IP_CH_SET_MINORLOOP_EN_DST_O↩FFSET	[BOOLEAN] The Minor Loop Enable Destination Offset Parameter enables the Dma Channel minor loop destination offset.
DMA_IP_CH_SET_MINORLOOP_SIGNED_O↩FFSET	[VALUE] The Minor Loop Signed Offset Parameter sets the Dma Channel minor loop signed offset.
DMA_IP_CH_SET_MINORLOOP_EN_LINK	[BOOLEAN] The Minor Loop Enable Link Parameter enables the Dma Channel minor loop logic channel linking.
DMA_IP_CH_SET_MINORLOOP_LOGIC_LIN↩K_CH	[VALUE] The Minor Loop Logic Channel Link Parameter sets the Dma Channel minor loop logic channel link.

Enumerator

DMA_IP_CH_SET_MINORLOOP_SIZE	[VALUE] The Minor Loop Size Parameter sets the Dma Channel minor loop transfer size.
DMA_IP_CH_SET_MAJORLOOP_EN_LINK	[BOOLEAN] The Major Loop Enable Link Parameter enables the Dma Channel major loop logic channel linking.
DMA_IP_CH_SET_MAJORLOOP_LOGIC_LI↩ NK_CH	[VALUE] The Major Loop Logic Channel Link Parameter sets the Dma Channel major loop logic channel link.
DMA_IP_CH_SET_MAJORLOOP_COUNT	[VALUE] The Major Loop Count Parameter sets the Dma Channel major loop count.
DMA_IP_CH_SET_CONTROL_SOFTWARE_↩ REQUEST	[BOOLEAN] The Enable Start Parameter enables the Dma Channel start service request. The main usage is for ScatterGather Element configuration.
DMA_IP_CH_SET_CONTROL_EN_MAJOR_I↩ NTERRUPT	[BOOLEAN] The Enable Major Interrupt Parameter enables the Dma Channel major interrupt.
DMA_IP_CH_SET_CONTROL_EN_HALF_M↩ AJOR_INTERRUPT	[BOOLEAN] The Enable Half Interrupt Parameter enables the Dma Channel half major interrupt.
DMA_IP_CH_SET_CONTROL_DIS_AUTO_R↩ EQUEST	[BOOLEAN] The Disable Automatic Request Parameter disables the Dma Channel automatic request.
DMA_IP_CH_SET_CONTROL_BANDWIDTH	[VALUE] The Bandwidth Control Parameter sets the Dma Channel bandwidth control.

Definition at line 157 of file Dma_Ip.h.

6.3.4.5 Dma_Ip_LogicChannelInfoParamType

enum Dma_Ip_LogicChannelInfoParamType

This type contains the Dma Ip Logic Channel Information Parameters.

The Parameters get specific information.

Enumerator

DMA_IP_CH_GET_SOURCE_ADDRESS	[VALUE] The Source Address Parameter gets the Dma Channel source address.
DMA_IP_CH_GET_DESTINATION_ADDRESS	[VALUE] The Destination Address Parameter gets the Dma Channel destination address.
DMA_IP_CH_GET_BEGIN_ITER_COUNT	[VALUE] The Begin Iteration Count Parameter gets the Dma Channel begin iteration count.
DMA_IP_CH_GET_CURRENT_ITER_COUNT	[VALUE] The Current Iteration Count Parameter gets the Dma Channel current iteration count.
DMA_IP_CH_GET_MAJOR_INTERRUPT	[BOOLEAN] The Major Interrupt Parameter gets the Dma Channel major interrupt.
DMA_IP_CH_GET_HALF_MAJOR_INTERR↩ UPT	[BOOLEAN] The Half Major Interrupt Parameter gets the Dma Channel half major interrupt.

Definition at line 199 of file Dma_Ip.h.

6.3.4.6 Dma_Ip_ReturnType

enum Dma_Ip_ReturnType

This type contains the Dma Ip Return Type.

The Return Type give information for the execution of interfaces.

Definition at line 127 of file Dma_Ip_Types.h.

6.3.4.7 Dma_Ip_HwChannelStateValueType

enum Dma_Ip_HwChannelStateValueType

This type contains the Dma Ip Channel State Value Type.

The Channel State type provides information about the channel's general state. The Reset State is present when the Dma Channel is uninitialized. The Ready State is present when the Dma Channel is initialized and without error. The Error State is present when the Dma Channel is initialized and with error.

Definition at line 144 of file Dma_Ip_Types.h.

6.3.5 Function Reference

6.3.5.1 Dma_Ip_Init()

```
Dma_Ip_ReturnType Dma_Ip_Init (
            const Dma_Ip_InitType *const DmaInit )
```

This function initializes the Dma Ip Driver.

This service is a non reentrant function that shall initialize the Dma Ip driver.

Parameters

in	*DmaInit*	Pointer to the configuration structure.

Returns

Dma_Ip_ReturnType DMA_IP_STATUS_SUCCESS is returned if the initialization finished ok. DMA_I↩ P_STATUS_WRONG_STATE is returned if the Dma Ip Channel state is not Dma_Ip_Ch_ResetState.

6.3.5.2 Dma_Ip_Deinit()

```
Dma_Ip_ReturnType Dma_Ip_Deinit (
            void  )
```

This function deinitializes the Dma Ip Driver.

This service is a non reentrant function that shall deinitialize the Dma Ip driver.

Returns

Dma_Ip_ReturnType DMA_IP_STATUS_SUCCESS is returned if the deinitialization finished ok

6.3.5.3 Dma_Ip_SetLogicInstanceCommand()

```
Dma_Ip_ReturnType Dma_Ip_SetLogicInstanceCommand (
            const uint32 LogicInst,
            const Dma_Ip_LogicInstanceCmdType Command )
```

This function sets Dma Ip Instance Command.

This service is a reentrant function that shall command the Dma Instance. The command shall trigger specific functionalities of the Dma Instance.

Parameters

in	*LogicInst*	Selection value of the Logic Instance.
in	*Command*	The command for the Logic Instance.

Returns

Dma_Ip_ReturnType DMA_IP_STATUS_SUCCESS is returned if the command finished ok.

6.3.5.4 Dma_Ip_GetLogicInstanceStatus()

```
Dma_Ip_ReturnType Dma_Ip_GetLogicInstanceStatus (
            const uint32 LogicInst,
            Dma_Ip_LogicInstanceStatusType *const Status )
```

This function gets Dma Ip Instance Status.

This service is a reentrant function that shall get the Dma Instance status. The command shall read specific functionalities of the Dma Instance.

Parameters

in	*LogicInst*	Selection value of the Logic Instance.
out	*Status*	Pointer to the Dma Instance status.

Returns

Dma_Ip_ReturnType DMA_IP_STATUS_SUCCESS is returned if the get status finished ok.

6.3.5.5 Dma_Ip_LogicChannelInit()

```
Dma_Ip_ReturnType Dma_Ip_LogicChannelInit (
            const uint32 LogicCh )
```

This function initializes the Dma Ip Logic Channel.

This service is a non reentrant function that shall initialize the Dma Ip Logic Channel.

Parameters

in	*LogicCh*	Selection value of the Logic Channel.

Returns

Dma_Ip_ReturnType DMA_IP_STATUS_SUCCESS is returned if the initialization finished ok. DMA_I↩P_STATUS_WRONG_STATE is returned if the Dma Ip Channel state is not Dma_Ip_Ch_ResetState.

6.3.5.6 Dma_Ip_LogicChannelDeinit()

```
Dma_Ip_ReturnType Dma_Ip_LogicChannelDeinit (
            const uint32 LogicCh )
```

This function deinitializes the Dma Ip Logic Channel.

This service is a non reentrant function that shall deinitialize the Dma Ip Logic Channel.

Parameters

in	*LogicCh*	Selection value of the Logic Channel.

Returns

Dma_Ip_ReturnType DMA_IP_STATUS_SUCCESS is returned if the deinitialization finished ok.

6.3.5.7 Dma_Ip_SetLogicChannelCommand()

```
Dma_Ip_ReturnType Dma_Ip_SetLogicChannelCommand (
            const uint32 LogicCh,
            const Dma_Ip_LogicChannelCmdType Command )
```

This function sets Dma Ip Logic Channel Command.

This service is a reentrant function that shall command the Dma Channel. The command shall trigger specific functionalities of the Dma Channel.

Parameters

in	*LogicCh*	Selection value of the Logic Channel.
in	*Command*	The command for the Logic Channel.

Returns

Dma_Ip_ReturnType DMA_IP_STATUS_SUCCESS is returned if the initialization finished ok. DMA_I↩P_STATUS_WRONG_STATE is returned if the Dma Ip Channel state is not Dma_Ip_Ch_ReadyState.

6.3.5.8 Dma_Ip_GetLogicChannelStatus()

```
Dma_Ip_ReturnType Dma_Ip_GetLogicChannelStatus (
            const uint32 LogicCh,
            Dma_Ip_LogicChannelStatusType *const ChStatus )
```

This function gets Dma Ip Logic Channel Status.

This service is a reentrant function that shall get the Dma Channel status. The command shall read specific functionalities of the Dma Channel.

Parameters

in	*LogicCh*	Selection value of the Logic Channel.
out	*Status*	Pointer to the Dma Channel status.

Returns

Dma_Ip_ReturnType DMA_IP_STATUS_SUCCESS is returned if the get status finished ok.

6.3.5.9 Dma_Ip_SetLogicChannelGlobalList()

```
Dma_Ip_ReturnType Dma_Ip_SetLogicChannelGlobalList (
            const uint32 LogicCh,
            const Dma_Ip_LogicChannelGlobalListType List[],
            const uint32 ListDimension )
```

This function sets Dma Ip Logic Channel Global List settings.

This service is a reentrant function that shall set the Dma Ip Logic Channel global parameters list. The list is composed of an array of Dma Ip Logic Channel global parameters settings. The settings list(array) is defined by the user needs: it contains the desired parameters to be configured, in any desired order.

How to use this interface:

1. Use the "Dma_Ip_LogicChannelGlobalListType" to create a list(array) with the desired paramaters to configure (see parameters: "Dma_Ip_LogicChannelGlobalParamType") The list can declared globally or locally: A. Global example: Dma_Ip_LogicChannelGlobalListType global_Dma_Ip_ChannelGlobalList0[NUMBER_↩ OF_PARAMETERS] = {...}; B. Local example: Dma_Ip_LogicChannelGlobalListType Dma_Ip_Channel↩ GlobalList[NUMBER_OF_PARAMETERS]; Dma_Ip_ChannelGlobalList[PARAMETER0].Param = DM↩ A_IP_CH_SET_EN_PREEMPTION_PRIORITY; Dma_Ip_ChannelGlobalList[PARAMETER0].Value = TRUE; Dma_Ip_ChannelGlobalList[PARAMETER1].Param = ...; Dma_Ip_ChannelGlobalList[PARAM↩ ETER1].Value = ...;
2. Call the "Dma_Ip_SetLogicChannelGlobalList()" interface: Dma_Ip_SetLogicChannelGlobalList(LOGIC↩ _CHANNELx, Dma_Ip_ChannelGlobalList, NUMBER_OF_PARAMETERS);

Parameters

in	*Channel*	Specifies the Logic Channel Id.
in	*List*	Pointer to the Global List Array.
in	*ListDimension*	Number of entries in the List.

Returns

Dma_Ip_ReturnType DMA_IP_STATUS_SUCCESS is returned if the initialization finished ok. DMA_I↩ P_STATUS_WRONG_STATE is returned if the Dma Ip Channel state is not Dma_Ip_Ch_ReadyState.

6.3.5.10 Dma_Ip_SetLogicChannelTransferList()

```
Dma_Ip_ReturnType Dma_Ip_SetLogicChannelTransferList (
            const uint32 LogicCh,
            const Dma_Ip_LogicChannelTransferListType List[],
            const uint32 ListDimension )
```

This function sets Dma Ip Logic Channel Transfer List settings.

-> This service is a reentrant function that shall set the Dma Ip Logic Channel transfer parameters list. -> The "Transfer List" loads the configuration directly into the Hardware TCD and disables the ScatterGather for the Hardware TCD. -> The list is composed of an array of Dma Ip Logic Channel transfer parameters settings. -> The settings array is defined by the user needs: it contains entries for each desired parameter to be configured, in any suitable order.

-> How to use this interface: <-

1. Use the "Dma_Ip_LogicChannelTransferListType" to create a list(array) with the desired paramaters to configure (see parameters: "Dma_Ip_LogicChannelTransferParamType") The list can declared globally or locally: Global example: #define DMA_IP_TRANSFER_LIST0_DIMENSION ((uint32)2U) Dma_Ip_LogicChannelTransferListType global_Dma_Ip_ChannelTransferList0[DMA_IP_TRANSFE↩ R_LIST0_DIMENSION] = {...}; Local example: #define DMA_IP_TRANSFER_LIST0_DIMENSION ((uint32)2U) Dma_Ip_LogicChannelTransferListType Dma_Ip_ChannelTransferList0[DMA_IP_TRA↩ NSFER_LIST0_DIMENSION]; Dma_Ip_ChannelTransferList0[PARAMETER0].Param = DMA_IP_↩ CH_SET_VAL_SOURCE_ADDRESS; Dma_Ip_ChannelTransferList0[PARAMETER0].Value = Dma↩ _Ip_ChannelTransferList0[PARAMETER1].Param = DMA_IP_CH_SET_DESTINATION_ADDRESS; Dma_Ip_ChannelTransferList0[PARAMETER1].Value =
2. Call the "Dma_Ip_SetLogicChannelTransferList()" interface: Dma_Ip_SetLogicChannelTransferList(LOG↩ IC_CHANNELx, Dma_Ip_ChannelTransferList0, DMA_IP_TRANSFER_LIST0_DIMENSION);

-> Coding Example: <- -> The user shall create the desired configuration list for his specific application. "User↩ DefinedFileName.h" #define DMA_IP_TRANSFER_LIST0_DIMENSION ((uint32)8U) #define DMA_IP_SE↩ T_TRANSFER_TYPE0(CHANNEL, DIMENSION, SADDR, SOFF, SSIZE, DADDR, DOFF, DSIZE, MINOR_↩ SIZE, MAJOR_COUNT) \ Dma_Ip_LogicChannelTransferListType Dma_Ip_ChannelTransferList0[DIMENSI↩ ON]; \ Dma_Ip_ChannelTransferList0[0U].Param = DMA_IP_CH_SET_VAL_SOURCE_ADDRESS; \ Dma↩ _Ip_ChannelTransferList0[0U].Value = SADDR; \ Dma_Ip_ChannelTransferList0[1U].Param = DMA_IP_CH_↩ SET_SOURCE_SIGNED_OFFSET; \ Dma_Ip_ChannelTransferList0[1U].Value = SOFF; \ Dma_Ip_Channel↩ TransferList0[2U].Param = DMA_IP_CH_SET_SOURCE_TRANSFER_SIZE; \ Dma_Ip_ChannelTransfer↩ List0[2U].Value = SSIZE; \ Dma_Ip_ChannelTransferList0[3U].Param = DMA_IP_CH_SET_DESTINATION↩ _ADDRESS; \ Dma_Ip_ChannelTransferList0[3U].Value = DADDR; \ Dma_Ip_ChannelTransferList0[4U].Param = DMA_IP_CH_SET_DESTINATION_SIGNED_OFFSET; \ Dma_Ip_ChannelTransferList0[4U].Value = D↩ OFF; \ Dma_Ip_ChannelTransferList0[5U].Param = DMA_IP_CH_SET_DESTINATION_TRANSFER_SIZE;

\ Dma_Ip_ChannelTransferList0[5U].Value = DSIZE; \ Dma_Ip_ChannelTransferList0[6U].Param = DMA_I↩ P_CH_SET_MINORLOOP_SIZE; \ Dma_Ip_ChannelTransferList0[6U].Value = MINOR_SIZE; \ Dma_Ip_↩ ChannelTransferList0[7U].Param = DMA_IP_CH_SET_MAJORLOOP_COUNT; \ Dma_Ip_ChannelTransfer↩ List0[7U].Value = MAJOR_COUNT; \ Dma_Ip_SetLogicChannelTransferList(CHANNEL, Dma_Ip_Channel↩ TransferList0, DIMENSION);

"ApplicationFileName.c" void ConfigureDmaChannel(ConfigType * pxConfig) { uint32 MinorLoopSize = 2U; uint32 MajorLoopCount; if(pxConfig->MajorLoopCondition == TRUE) { MajorLoopCount = 8U; } else { MajorLoop↩ Count = 24U; } DMA_IP_SET_TRANSFER_TYPE0(pxConfig->LogicChannel, DMA_IP_TRANSFER_LIS↩ T0_DIMENSION, pxConfig->SourceBuffer, 2U, DMA_IP_TRANSFER_SIZE_2_BYTE, &RegisterAddress, 0U, DMA_IP_TRANSFER_SIZE_2_BYTE, MinorLoopSize, MajorLoopCount); }

Parameters

in	*Channel*	Specifies the Logic Channel Id.
in	*List*	Pointer to the Transfer List Array.
in	*ListDimension*	Number of entries in the List.

Returns

Dma_Ip_ReturnType DMA_IP_STATUS_SUCCESS is returned if the initialization finished ok. DMA_I↩ P_STATUS_WRONG_STATE is returned if the Dma Ip Channel state is not Dma_Ip_Ch_ReadyState.

6.3.5.11 Dma_Ip_SetLogicChannelScatterGatherList()

```
Dma_Ip_ReturnType Dma_Ip_SetLogicChannelScatterGatherList (
            const uint32 LogicCh,
            const uint32 Element,
            const Dma_Ip_LogicChannelScatterGatherListType List[],
            const uint32 ListDimension )
```

This function sets Dma Ip Logic Channel Scatter/Gather List settings.

-> This service is a reentrant function that shall set the Dma Ip Logic Channel scatter/gather parameters list. -> The "Scatter/Gather List" configures Logic Elements belonging to the same Dma Logic Channel. -> The "Scatter/↩ Gather List" loads the configuration into the Software TCD. The Software TCD has the Scatter/Gather Enable set (ESG bit) and the Next Software TCD Address already loaded during the configuration generation process. -> The "Scatter/Gather List" shall not be able to modify the Scatter/Gather Element linkage (reorder of elements in the chain). The linkage of the elements is set only during the configuration process. -> The settings array is defined by the user needs: it contains entries for each desired parameter to be configured, in any suitable order. -> This service does not load the Logic Element into the Hardware TCD. This functionality is covered by Dma_Ip_Set↩ LogicChannelScatterGatherConfig.

How to use this interface:

1. Use the "Dma_Ip_LogicChannelScatterGatherListType" to create a list(array) with the desired paramaters to configure (see parameters: "Dma_Ip_LogicChannelTransferParamType") The list can declared globally or locally: Global example: Dma_Ip_LogicChannelScatterGatherListType global_Dma_Ip_Channel↩ ScatterGatherList0[NUMBER_OF_PARAMETERS] = {...}; Local example: Dma_Ip_LogicChannel↩ ScatterGatherListType Dma_Ip_ChannelScatterGatherList[NUMBER_OF_PARAMETERS]; Dma_Ip↩ _ChannelScatterGatherList[PARAMETER0].Param = DMA_IP_CH_SET_VAL_SOURCE_ADDRESS; Dma_Ip_ChannelScatterGatherList[PARAMETER0].Value = Dma_Ip_ChannelScatterGatherList[PARA↩ METER1].Param = DMA_IP_CH_SET_DESTINATION_ADDRESS; Dma_Ip_ChannelScatterGather↩ List[PARAMETER1].Value =
2. Call the "Dma_Ip_SetLogicChannelScatterGatherList()" interface: Dma_Ip_SetLogicChannelScatter↩ GatherList(LOGIC_CHANNELx, LOGIC_ELEMENTy, Dma_Ip_ChannelScatterGatherList, NUMBE↩ R_OF_PARAMETERS);

Returns

Dma_Ip_ReturnType DMA_IP_STATUS_SUCCESS is returned if the initialization finished ok. DMA_I↩ P_STATUS_WRONG_STATE is returned if the Dma Ip Channel state is not Dma_Ip_Ch_ReadyState.

6.3.5.12 Dma_Ip_GetLogicChannelParam()

```
Dma_Ip_ReturnType Dma_Ip_GetLogicChannelParam (
            const uint32 LogicCh,
            const Dma_Ip_LogicChannelInfoParamType Param,
            uint32 *const Value )
```

This function gets the Dma Ip Logic Channel Parameter value.

This service is a reentrant function that shall get the Dma Channel parameters value.

Parameters

in	*LogicCh*	Selection value of the Logic Channel.
in	*Param*	Selection parameter.
out	*Value*	Pointer to the parameter value.

Returns

Dma_Ip_ReturnType DMA_IP_STATUS_SUCCESS is returned if the get information finished ok.

6.3.5.13 Dma_Ip_SetLogicChannelScatterGatherConfig()

```
Dma_Ip_ReturnType Dma_Ip_SetLogicChannelScatterGatherConfig (
            const uint32 LogicCh,
            const uint32 Element )
```

This function configures the Dma Ip Logic Channel Scatter/Gather.

This service is a reentrant function that shall configure the Dma Channel scatter/gather functionality. The specified Logic Element (corresponding to a Software TCD) shall be loaded into the Dma Logic Channel's Hardware TCD. The Logic Elements (describing the Software TCDs) form a simple chained list and the "Element" parameter represents the lists's head.

Parameters

in	*LogicCh*	Selection value of the Logic Channel.
in	*Element*	Selection value of the Logic Element representing the list's head.

Returns

Dma_Ip_ReturnType DMA_IP_STATUS_SUCCESS is returned if the initialization finished ok. DMA_↩ IP_STATUS_WRONG_STATE is returned if the Dma Ip Channel state is not Dma_Ip_Ch_ReadyState. DMA_IP_STATUS_ERROR is returned if the Dma Ip Channel contains an error.

6.4 FLEXIO IP Driver

6.4.1 Detailed Description

Enum Reference

- enum Flexio_Mcl_Ip_TimerPolarityType
- enum Flexio_Mcl_Ip_PinPolarityType
- enum Flexio_Mcl_Ip_PinConfigType
- enum Flexio_Mcl_Ip_TriggerPolarityType
- enum Flexio_Mcl_Ip_TriggerSourceType
- enum Flexio_Mcl_Ip_TimerModeType
- enum Flexio_Mcl_Ip_TimerOutputType
- enum Flexio_Mcl_Ip_TimerDecrementType
- enum Flexio_Mcl_Ip_TimerResetType
- enum Flexio_Mcl_Ip_TimerDisableType
- enum Flexio_Mcl_Ip_TimerEnableType
- enum Flexio_Mcl_Ip_TimerStopType
- enum Flexio_Mcl_Ip_TimerStartType

6.4.2 Enum Reference

6.4.2.1 Flexio_Mcl_Ip_TimerPolarityType

enum Flexio_Mcl_Ip_TimerPolarityType

Enumerator

FLEXIO_TIMER_POLARITY_POSEDGE	Shift on positive edge of Shift clock
FLEXIO_TIMER_POLARITY_NEGEDGE	Shift on negative edge of Shift clock

Definition at line 123 of file Flexio_Mcl_Ip_HwAccess.h.

6.4.2.2 Flexio_Mcl_Ip_PinPolarityType

enum Flexio_Mcl_Ip_PinPolarityType

Enumerator

FLEXIO_PIN_POLARITY_HIGH	Pin is active high
FLEXIO_PIN_POLARITY_LOW	Pin is active low

Definition at line 131 of file Flexio_Mcl_Ip_HwAccess.h.

6.4.2.3 Flexio_Mcl_Ip_PinConfigType

enum Flexio_Mcl_Ip_PinConfigType

Enumerator

FLEXIO_PIN_CONFIG_DISABLED	Shifter pin output disabled
FLEXIO_PIN_CONFIG_OPEN_DRAIN	Shifter pin open drain or bidirectional output enable
FLEXIO_PIN_CONFIG_BIDIR_OUTPUT	Shifter pin bidirectional output data
FLEXIO_PIN_CONFIG_OUTPUT	Shifter pin output

Definition at line 138 of file Flexio_Mcl_Ip_HwAccess.h.

6.4.2.4 Flexio_Mcl_Ip_TriggerPolarityType

enum Flexio_Mcl_Ip_TriggerPolarityType

Enumerator

FLEXIO_TRIGGER_POLARITY_HIGH	Trigger is active high
FLEXIO_TRIGGER_POLARITY_LOW	Trigger is active low

Definition at line 171 of file Flexio_Mcl_Ip_HwAccess.h.

6.4.2.5 Flexio_Mcl_Ip_TriggerSourceType

enum Flexio_Mcl_Ip_TriggerSourceType

Enumerator

FLEXIO_TRIGGER_SOURCE_EXTERNAL	External trigger selected
FLEXIO_TRIGGER_SOURCE_INTERNAL	Internal trigger selected

Definition at line 178 of file Flexio_Mcl_Ip_HwAccess.h.

6.4.2.6 Flexio_Mcl_Ip_TimerModeType

enum Flexio_Mcl_Ip_TimerModeType

Enumerator

FLEXIO_TIMER_MODE_DISABLED	Timer Disabled.
FLEXIO_TIMER_MODE_8BIT_BAUD	Dual 8-bit counters baud/bit mode.
FLEXIO_TIMER_MODE_8BIT_PWM	Dual 8-bit counters PWM mode.
FLEXIO_TIMER_MODE_16BIT	Single 16-bit counter mode.
FLEXIO_TIMER_MODE_16BIT_DIS	Single 16-bit counter disable mode.
FLEXIO_TIMER_MODE_8BIT_DUAL	Dual 8-bit counters word mode.
FLEXIO_TIMER_MODE_8BIT_DUAL_PWM	Dual 8-bit counters PWM low mode.
FLEXIO_TIMER_16BIT_INPUT_CAPTURE_MODE	Single 16-bit input capture mode.

Definition at line 185 of file Flexio_Mcl_Ip_HwAccess.h.

6.4.2.7 Flexio_Mcl_Ip_TimerOutputType

enum Flexio_Mcl_Ip_TimerOutputType

Enumerator

FLEXIO_TIMER_INITOUT_ONE	Timer output is logic one when enabled, unaffected by timer reset.
FLEXIO_TIMER_INITOUT_ZERO	Timer output is logic zero when enabled, unaffected by timer reset.
FLEXIO_TIMER_INITOUT_ONE_RESET	Timer output is logic one when enabled and on timer reset.
FLEXIO_TIMER_INITOUT_ZERO_RESET	Timer output is logic zero when enabled and on timer reset.

Definition at line 198 of file Flexio_Mcl_Ip_HwAccess.h.

6.4.2.8 Flexio_Mcl_Ip_TimerDecrementType

enum Flexio_Mcl_Ip_TimerDecrementType

Enumerator

FLEXIO_TIMER_DECREMENT_CLK_SHIFT↩ _TMR	Decrement counter on FlexIO clock, Shift clock equals Timer output.
FLEXIO_TIMER_DECREMENT_TRG_SHIFT↩ _TMR	Decrement counter on Trigger input (both edges), Shift clock equals Timer output.
FLEXIO_TIMER_DECREMENT_PIN_SHIFT_↩ PIN	Decrement counter on Pin input (both edges), Shift clock equals Pin input.
FLEXIO_TIMER_DECREMENT_TRG_SHIFT↩ _TRG	Decrement counter on Trigger input (both edges), Shift clock equals Trigger input.

Definition at line 207 of file Flexio_Mcl_Ip_HwAccess.h.

6.4.2.9 Flexio_Mcl_Ip_TimerResetType

enum Flexio_Mcl_Ip_TimerResetType

Enumerator

FLEXIO_TIMER_RESET_NEVER	Timer never reset.
FLEXIO_TIMER_RESET_PIN_OUT	Timer reset on Timer Pin equal to Timer Output.
FLEXIO_TIMER_RESET_TRG_OUT	Timer reset on Timer Trigger equal to Timer Output.
FLEXIO_TIMER_RESET_PIN_RISING	Timer reset on Timer Pin rising edge.
FLEXIO_TIMER_RESET_TRG_RISING	Timer reset on Trigger rising edge.
FLEXIO_TIMER_RESET_TRG_BOTH	Timer reset on Trigger rising or falling edge.

Definition at line 216 of file Flexio_Mcl_Ip_HwAccess.h.

6.4.2.10 Flexio_Mcl_Ip_TimerDisableType

enum Flexio_Mcl_Ip_TimerDisableType

Enumerator

FLEXIO_TIMER_DISABLE_NEVER	Timer never disabled.
FLEXIO_TIMER_DISABLE_TIM_DISABLE	Timer disabled on Timer N-1 disable.
FLEXIO_TIMER_DISABLE_TIM_CMP	Timer disabled on Timer compare.
FLEXIO_TIMER_DISABLE_TIM_CMP_TRG_↩ LOW	Timer disabled on Timer compare and Trigger Low.
FLEXIO_TIMER_DISABLE_PIN	Timer disabled on Pin rising or falling edge.
FLEXIO_TIMER_DISABLE_PIN_TRG_HIGH	Timer disabled on Pin rising or falling edge provided Trigger is high.
FLEXIO_TIMER_DISABLE_TRG	Timer disabled on Trigger falling edge.

Definition at line 227 of file Flexio_Mcl_Ip_HwAccess.h.

6.4.2.11 Flexio_Mcl_Ip_TimerEnableType

enum Flexio_Mcl_Ip_TimerEnableType

Enumerator

FLEXIO_TIMER_ENABLE_ALWAYS	Timer always enabled.
FLEXIO_TIMER_ENABLE_TIM_ENABLE	Timer enabled on Timer N-1 enable.
FLEXIO_TIMER_ENABLE_TRG_HIGH	Timer enabled on Trigger high.
FLEXIO_TIMER_ENABLE_TRG_PIN_HIGH	Timer enabled on Trigger high and Pin high.
FLEXIO_TIMER_ENABLE_PIN_POSEDGE	Timer enabled on Pin rising edge.
FLEXIO_TIMER_ENABLE_PIN_POSEDGE_TRG↩ _HIGH	Timer enabled on Pin rising edge and Trigger high.
FLEXIO_TIMER_ENABLE_TRG_POSEDGE	Timer enabled on Trigger rising edge.
FLEXIO_TIMER_ENABLE_TRG_EDGE	Timer enabled on Trigger rising or falling edge.

Definition at line 239 of file Flexio_Mcl_Ip_HwAccess.h.

6.4.2.12 Flexio_Mcl_Ip_TimerStopType

enum Flexio_Mcl_Ip_TimerStopType

Enumerator

FLEXIO_TIMER_STOP_BIT_DISABLED	Stop bit disabled.
FLEXIO_TIMER_STOP_BIT_TIM_CMP	Stop bit is enabled on timer compare.
FLEXIO_TIMER_STOP_BIT_TIM_DIS	Stop bit is enabled on timer disable.
FLEXIO_TIMER_STOP_BIT_TIM_CMP_DIS	Stop bit is enabled on timer compare and disable.

Definition at line 252 of file Flexio_Mcl_Ip_HwAccess.h.

6.4.2.13 Flexio_Mcl_Ip_TimerStartType

enum Flexio_Mcl_Ip_TimerStartType

Enumerator

FLEXIO_TIMER_START_BIT_DISABLED	Start bit disabled.
FLEXIO_TIMER_START_BIT_ENABLED	Start bit enabled.

Definition at line 278 of file Flexio_Mcl_Ip_HwAccess.h.

6.5 FTM IP Driver

6.5.1 Detailed Description

Function Reference

- void Ftm_Mcl_Ip_SelectCommonTimebase (uint8 Instance, uint32 SyncList)

  Implementation specific function to updates the Global Timebase bits of configured modules.

6.5.2 Function Reference

6.5.2.1 Ftm_Mcl_Ip_SelectCommonTimebase()

```
void Ftm_Mcl_Ip_SelectCommonTimebase (
            uint8 Instance,
            uint32 SyncList )
```

Implementation specific function to updates the Global Timebase bits of configured modules.

This function is used to set the global timebase bits for modules that support the global timebase feature. The function selects the module that gives the common timebase and the modules that are use this timebase (as bits in syncList). Then it synchronizes the modules. example: syncList is 0x0003 - modules 0 and 1 use the timebase given by instance syncList is 0x0005 - modules 0 and 2 use the timebase given by instance

Parameters

in	*Instance*	FTM module id
in	*SyncList*	FTM module mask value

Returns

void

6.6 TRGMUX IP Driver

6.6.1 Detailed Description

Function Reference

- Trgmux_Ip_StatusType Trgmux_Ip_Init (const Trgmux_Ip_InitType *const pxTrgmuxInit)
  This function initializes the Trgmux Ip Driver.
- Trgmux_Ip_StatusType Trgmux_Ip_SetInput (const uint32 LogicTrigger, const uint32 Input)
  This function sets the Trgmux Ip Input.
- Trgmux_Ip_StatusType Trgmux_Ip_SetLock (const uint32 LogicTrigger)
  This function locks the Trgmux Ip Trigger.

6.6.2 Function Reference

6.6.2.1 Trgmux_Ip_Init()

```
Trgmux_Ip_StatusType Trgmux_Ip_Init (
            const Trgmux_Ip_InitType *const pxTrgmuxInit )
```

This function initializes the Trgmux Ip Driver.

This service is a non reentrant function that shall initialize the Trgmux Ip driver.

Parameters

in	*pxTrgmuxInit*	Pointer to the initialization structure.

Returns

TRGMUX_IP_STATUS_LOCKED The trigger is locked. TRGMUX_IP_STATUS_SUCCESS The initialization was successful.

6.6.2.2 Trgmux_Ip_SetInput()

```
Trgmux_Ip_StatusType Trgmux_Ip_SetInput (
            const uint32 LogicTrigger,
            const uint32 Input )
```

This function sets the Trgmux Ip Input.

This service is a reentrant function that shall set the Trgmux Ip Input.

Parameters

in	*LogicTrigger*	The logic trigger id.
in	*Input*	The input value.

Returns

TRGMUX_IP_STATUS_LOCKED The trigger is locked. TRGMUX_IP_STATUS_SUCCESS The initialization was successful.

6.6.2.3 Trgmux_Ip_SetLock()

```
Trgmux_Ip_StatusType Trgmux_Ip_SetLock (
             const uint32 LogicTrigger )
```

This function locks the Trgmux Ip Trigger.

This service is a reentrant function that shall lock the Trgmux Ip Trigger.

Parameters

in	*LogicTrigger*	The logic trigger id.

Returns

TRGMUX_IP_STATUS_LOCKED The trigger is locked. TRGMUX_IP_STATUS_SUCCESS The initialization was successful.

How to Reach Us:

Home Page:
nxp.com

Web Support:
nxp.com/support

Information in this document is provided solely to enable system and software implementers to use NXP products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document. NXP reserves the right to make changes without further notice to any products herein.

NXP makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does NXP assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters that may be provided in NXP data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including "typicals," must be validated for each customer application by customer's technical experts. NXP does not convey any license under its patent rights nor the rights of others. NXP sells products pursuant to standard terms and conditions of sale, which can be found at the following address: nxp.com/SalesTermsandConditions.