MCAL CDDIPC Module Software Design Document

Document Version : 34
Document Owner : Texas Instruments
Document Status : Published
Last Approval Date : Jun 30, 2022

1 Revision History

Version	Date	Author	Document Status	Comments
0.1	12 Apr 2022	Jayapriya J	DONE	Initial version
0.2	18 May 2022	Jayapriya J	DONE	Updated Sec 5.5 and all other review comments .
0.3	25 May 2022	Jayapriya J	DONE	Updated Safety Daignostic API's and other review comments .
v34	09 Jun 2022	Jayapriya J	DONE	Comola Workflow Added, Review Comments Addressed.

2 Terms and Abbreviations

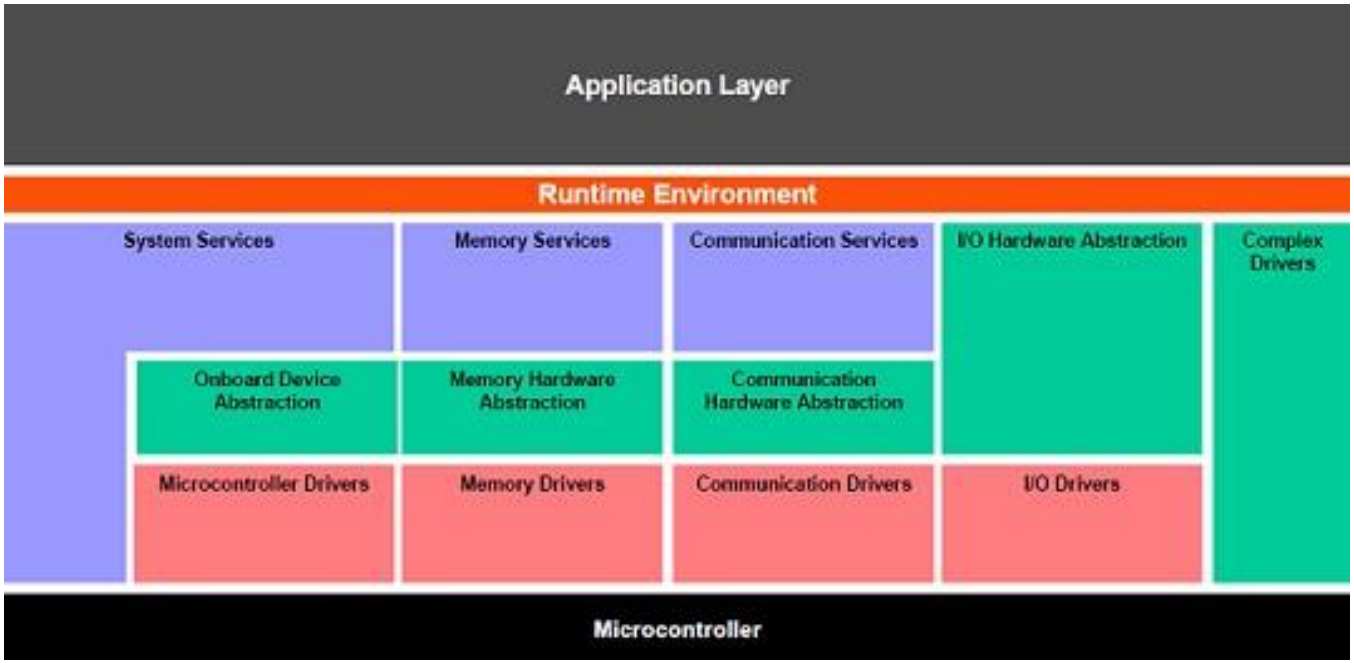
Abbreviation / Term	Meaning / Explanation
CDD IPC	Complex Device Driver Inter Processor Communication
AUTOSAR	AUTomotive Open System ARchitecture
RTE	Runtime Environment
BSW	Basic Software
MCAL	MicroController Abstraction Layer
SBL	Serial Bootloader
API	Application Programming Interface
DET	Default Error Tracer

Abbreviation / Term	Meaning / Explanation
DEM	Diagnostic Event Manager – module to handle diagnostic relevant events.
ECU	Electronic Control Unit
MCU	Micro Controller Unit
OS	Operating System
SoC	System on a Chip
DAR	Decision Analysis and Resolution

3 Introduction

3.1 Overview

The figure below depicts the AUTOSAR layered architecture as 3 distinct layers, Application, Runtime Environment (RTE) and Basic Software (BSW). The BSW is further divided into 4 layers, Services, Electronic Control Unit Abstraction, MicroController Abstraction (MCAL) and Complex Drivers.



AUTOSAR Architecture

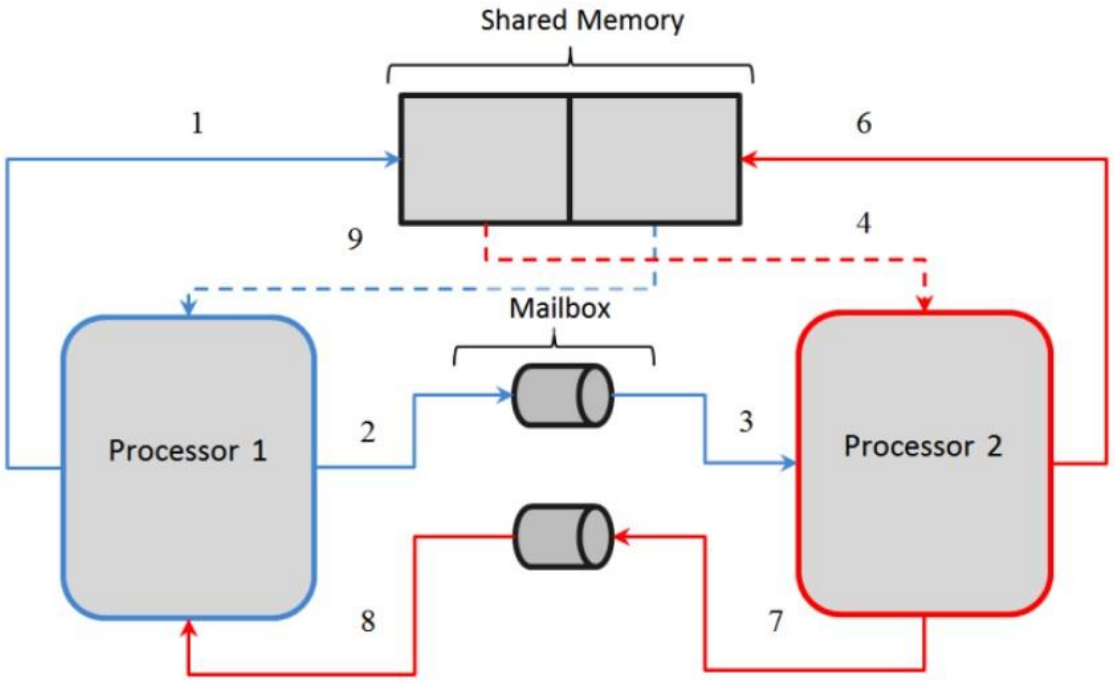
3.2 Purpose and Scope

This document specifies the implementation of MCAL driver for the module CDD IPC, using hardware IP "ksipc_mailbox_rel.1.0.x". All requirements for implementing the driver are mapped in the document itself. Also, document holds the detailed information regarding the MACROs, structures and APIs for the implementation.

3.3 Module Overview

CDD IPC is primarily used for communication with other cores on the SoC. This implementation relies on mailbox and shared memory to establish communication channel. Shared memory holds the messages that requires to be transported and mailbox is used to notify the remote core on availability of a message.

Basic Working Principle



CDD IPC MCAL - Basic Working Principle

Transmission & Reception

Transmission of message from 1 processor to another is performed as depicted below,

1. Processor 1 copies the message to shared area, designated write area to this processor
2. Writes a pre-determined pattern to mailbox
3. Processor 2 receives an interrupt indicating presence of a message in Processor 1, designated write area
4. Processor 2 read the data from shared area
5. Processor 2 processes the received message and has to reply back with different message
6. Processor 2 copies the message to shared area, designated write area to this processor
7. Writes a pre-determined pattern to mailbox
8. Processor 1 receives an interrupt indicating presence of a message in Processor 2, designated write area
9. Processor 1 read the data from shared area
10. Processor 1 processes the received message

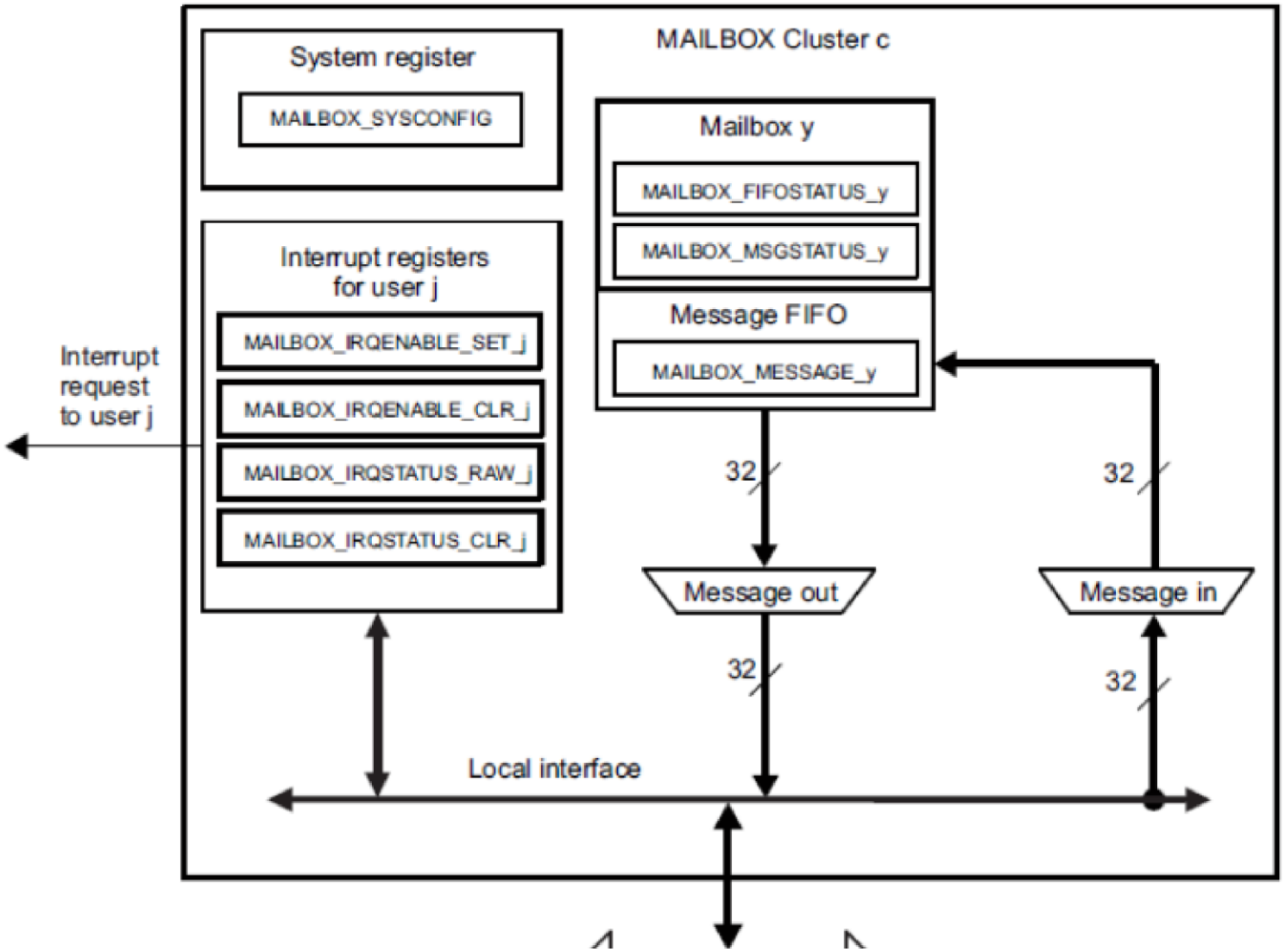
Note- Processor 1 has to send a message of 128 bytes to Processor 2

Mailbox hardware

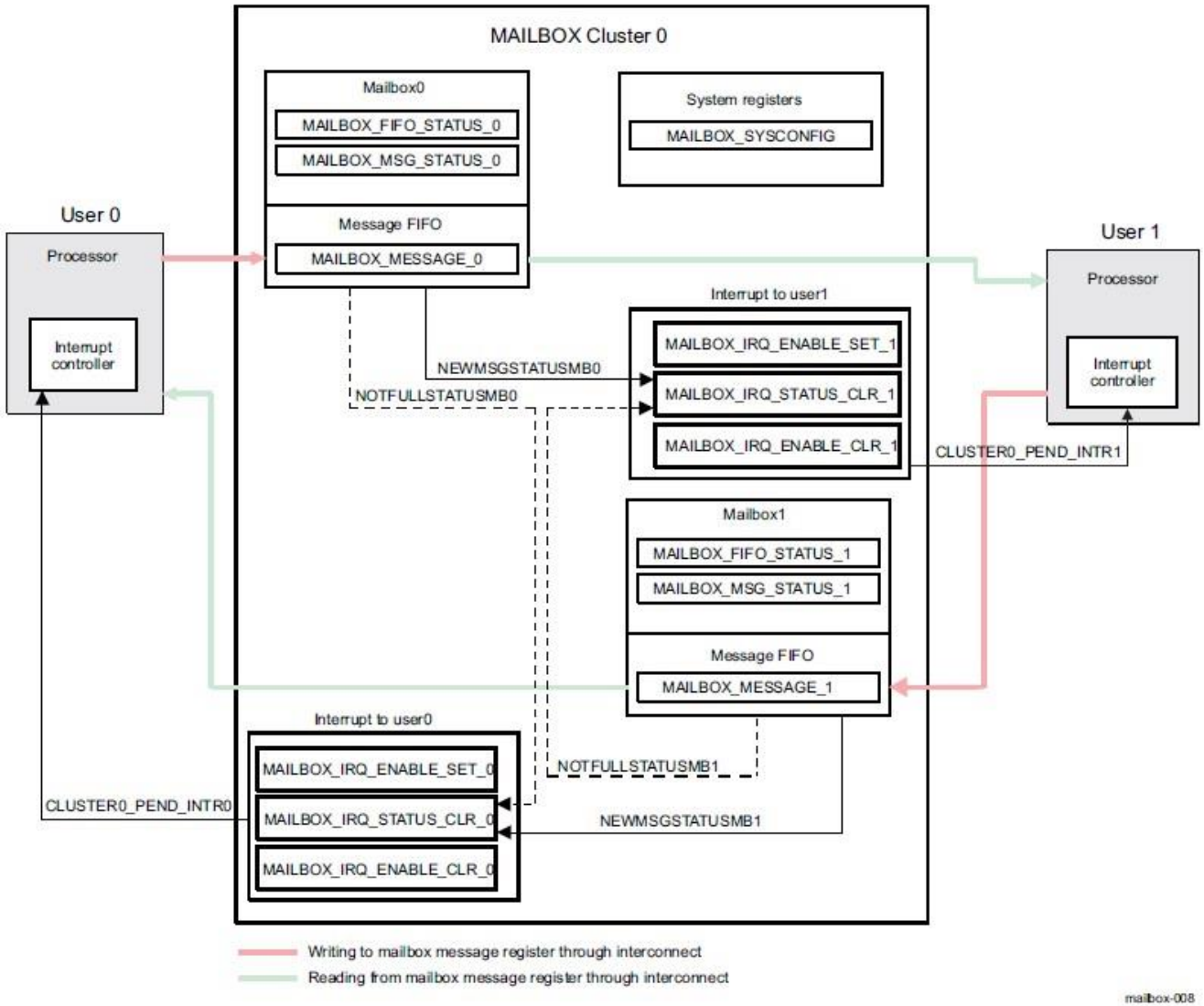
Communication between the on-chip processors uses a queued (FIFO) mailbox-interrupt mechanism. The queued mailbox-interrupt mechanism allows the software to establish a communication channel between two processors through a set of registers and associated interrupt signals by sending and receiving messages. Mailbox could be envisioned as shared FIFO between cores and can generate an interrupt either on reception of a 32 bit word or on FIFO not being empty.

Below shows the block diagram of the Mailbox IP, FIFO (referred as FIFO ID, throughout this document) could be used to write and read messages. The depth of the FIFO depends on the SOC used and an interrupt could be generated either on reception of a 32 bit word or on FIFO not being empty. These interrupts could be routed to any of the cores (refer device specific TRM for restrictions, not all mailboxes interrupts could be routed to all cores).

Mailbox Block Diagram



Mailbox Block Diagram



Example of Communication between 2 cores

It's important to note that the Mailbox hardware shall not be reset, as there could be pending messages in the FIFO. Other entities (such as boot-loader, start up sequence would have reset the mailbox).

	Description
MCAL-7236 - Cdd_IpcInit : Initialization : reset PUBLISHED	Cdd_IpcInit : Initialization : reset

Design Identifier

3.4 Requirements

The CDD IPC driver shall implement as per requirements detailed in Reference 1 - AUTOSAR 4.3.1.

3.4.1 Features Supported

Below listed are some of the key features that are expected to be supported

- Ability to transport fixed messages across cores
- Notify on reception of message from remote core
- Received messages are retained within the driver until consumed by applications
- i.e. Messages will not be available in the new message notification ISR. Service API call required to receive the message
- Number of messages that can be queued is configurable

- Configurable maximum message size
- Ability to announce capability of core to all other cores
- PRE COMPILE Variant is supported

Design Identifier	Description
MCAL-7168 - Cdd_IpcSendMsg : Send Msg PUBLISHED	Cdd_IpcSendMsg : Send Msg
MCAL-7195 - Cdd_IpcNewMsg PUBLISHED	Cdd_IpcNewMsg
MCAL-7151 - Cdd_IpcReceiveMsg : Receive Msg PUBLISHED	Cdd_IpcReceiveMsg : Receive Msg
MCAL-7231 - Cdd_IpcChannelType : numBufs PUBLISHED	Cdd_IpcChannelType : numBufs
MCAL-7213 - Cdd_IpcChannelType : multiplicity PUBLISHED	Cdd_IpcChannelType : multiplicity
MCAL-7241 - Cdd_IpcAnnounce : Announce PUBLISHED	Cdd_IpcAnnounce : Announce
MCAL-7153 - Cdd_IpcInit : Initialization : Variant Support PUBLISHED	Cdd_IpcInit : Initialization : Variant Support

3.4.2 Features Not Supported / NON Compliance

- Always uses shared memory irrespective of the message length.
- Configurability to use different mailbox, user id, FIFO ID & cluster instance is built into driver, that guarantees inter-interoperability with MCAL Hardware drivers.
- VARIANT-POST-BUILD and VARIANT-LINK-TIME Variants are not supported.

Design Identifier	Description
MCAL-7153 - Cdd_IpcInit : Initialization : Variant Support PUBLISHED	Cdd_IpcInit : Initialization : Variant Support
MCAL-7178 - Cdd_IpcInit : Initialization : Arguments PUBLISHED	Cdd_IpcInit : Initialization : Arguments
MCAL-7232 - Cdd_IpcInit : Initialization : Variant PUBLISHED	Cdd_IpcInit : Initialization : Variant

3.5 Assumptions

Below listed are assumed to be valid for this design/implementation, exceptions and other deviations are listed for each explicitly. Care should be taken to ensure these assumptions are addressed.

1. This design assumes that MCAL Hardware driver are used in the remote cores
2. The shared buffer shall be allocated in non-cached region and accessible to all cores participating in IPC
3. The functional clock to the Mailbox module is expected to be on before calling any CDD IPC service APIs
4. The CDD IPC driver as such doesn't perform any PRCM programming to get the functional clock
5. Configurator : This design do not depend on the configurator used. Use of EB Configurator is recommended as other MCAL modules use the same.

3.6 Constraints

Some of the critical constraints of this design are listed below

- The communication channels are created statically, via the configurator and the Remote end points of the remote core should be defined before compilation of generated CDD IPC configurations.
- Reserved end point, CDD IPC will reserve one of the end point which shall be be used to communicate control messages.
- Control endpoint, CDD IPC shall create a control end point which is primarily used for communication of control messages.

Design Identifier	Description
MCAL-7218 - Cdd_IpcChannelType PUBLISHED	Cdd_IpcChannelType
MCAL-7234 - Core IDs PUBLISHED	Core IDs
MCAL-7192 - Cdd_IpcChannelType : localEp PUBLISHED	Cdd_IpcChannelType : localEp

Design Identifier	Description
MCAL-7245 - Cdd_IpcChannelType : remoteEp PUBLISHED	Cdd_IpcChannelType : remoteEp
MCAL-7182 - Cdd_IpcChannelType : remoteProcId PUBLISHED	Cdd_IpcChannelType : remoteProcId
MCAL-7231 - Cdd_IpcChannelType : numBufs PUBLISHED	Cdd_IpcChannelType : numBufs
MCAL-7213 - Cdd_IpcChannelType : multiplicity PUBLISHED	Cdd_IpcChannelType : multiplicity
MCAL-7212 - Cdd_IpcChannelType : Id_symbolic name PUBLISHED	Cdd_IpcChannelType : Id_symbolic name
MCAL-7223 - Cdd_IpcMpType : numProcs PUBLISHED	Cdd_IpcMpType : numProcs

3.7 Hardware and SW platforms

Hardware Platforms

- Refer to specified SoC User Manual to check if IPC module is supported. **Software Platforms**
- Bare-Metal

3.8 Dependencies

CDD IPC driver shall depend on these modules to realize the required functionality.

- Standard BSW / AUTOSAR modules
- Det : To report development errors. Should be able to turn OFF (especially for production build)
- Dem : To report run time error (e.g. report critical error / warning, when Det is turned off: STD_OFF)
- SchM : For exclusive access (in interrupt context and thread/task context)

3.9 Stakeholders

- Developers
- Test Engineers
- Customer Integrator

3.10 References

	Specification	Comment/Link
1	AUTOSAR 4.3.1	AUTOSAR Specification for CDDIPC Driver
2	BSW General Requirements / Coding guidelines	Autosar and Coding guidelines for the Mcal drivers.
3	Software Product Specification (SPS)	Product Functional requirements.
4	Software Architecture	Mcal Software Architecture.

4 Design Description

4.1 Fundamental Operation

As detailed in Overview, IPC relies on shared memory & mailbox to transmit and receive messages. Section below highlight some of the key concepts.

4.1.1 Shared Memory

As discussed in Basic Principle, shared memory is required for IPC. This shared memory region shall be referred as **VirtIo.**

4.1.2 Processor Identifiers

In order to communicate with multiple cores, each cores requires to be identified uniquely by **procId** in the rest of this document. The

configurator shall allow integrators to select set of cores, with which communication is desired.

4.1.3 End Point

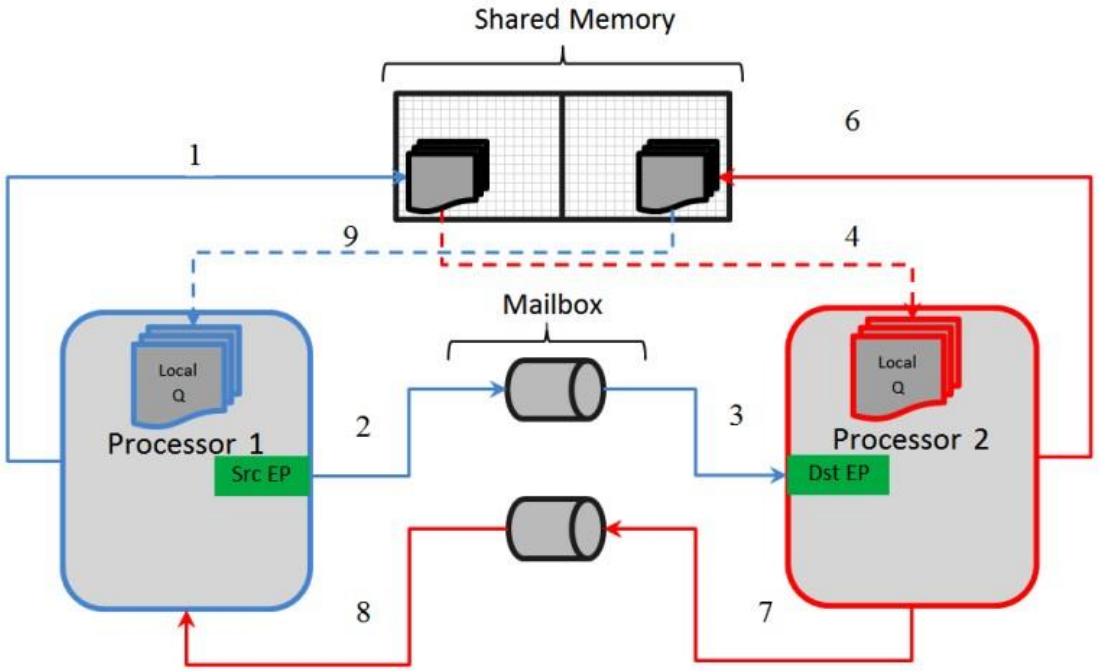
To allow multiple logical channels for communications an end-point shall be used. The end-point shall be an unsigned integer configurable through the configurator, with the exception of reserved end point.

4.1.4 Message Buffer

Referring the diagram below, Processor 1 reads from Processor 2 designated write area and vice versa. To prevent loss of messages (in cases where the receiving processor was slower/loaded with other high priority tasks) the message is copied into local queue. When service API to receive is invoked, the received message from the queue is copied into user provided buffer. Please note that, these copies are CPU based copies.

4.1.5 Queue in shared memory

It could be possible that one processor (producer) might generate faster IPC messages than another processor (consumer). To avoid messages being over-written/lost an shared queue shall be implemented in the shared buffer, as depicted in the diagram below



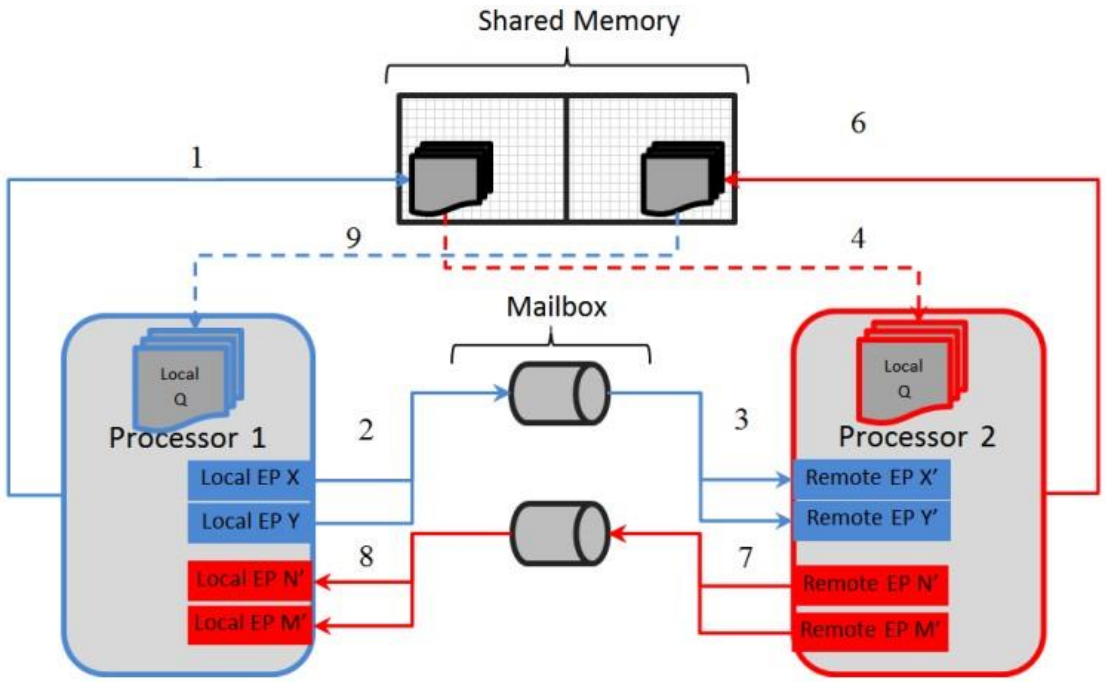
Shared Queue

Some of the key points to note, considering the above example

- The queue implemented in Processor 1 designated shared area
    a. The queue shall be circular queue
    b. The writer (Processor 1, in this e.g.) shall write and advance write-pointer only
    c. The reader (Processor 2, in this e.g.) shall read and advance read-pointer only
    d. The actual message shall be stored in the designated shared area and queue element shall contain a pointer to the message.

4.1.6 Multiple End Point / Communication Channels

CDD IPC shall provide ability to create multiple end-point pairs. As depicted in the diagram below, applications could define multiple end-point pairs to realize multiple communication channels.



Multiple End Points

Design Identifier	Description
MCAL-7245 - Cdd_IpcChannelType : remoteEp PUBLISHED	Cdd_IpcChannelType : remoteEp
MCAL-7194 - Cdd_IpcVertIoType PUBLISHED	Cdd_IpcVertIoType
MCAL-7180 - Cdd_IpcVertIoType : vertIoObjAddr PUBLISHED	Cdd_IpcVertIoType : vertIoObjAddr
MCAL-7152 - Cdd_IpcVertIoType : vertIoObjSize PUBLISHED	Cdd_IpcVertIoType : vertIoObjSize
MCAL-7197 - Cdd_IpcVertIoType : vertIoRingAddr PUBLISHED	Cdd_IpcVertIoType : vertIoRingAddr
MCAL-7199 - Cdd_IpcVertIoType : vertIoRingSize PUBLISHED	Cdd_IpcVertIoType : vertIoRingSize
MCAL-7204 - Cdd_IpcMpType : ownProcID PUBLISHED	Cdd_IpcMpType : ownProcID
MCAL-7223 - Cdd_IpcMpType : numProcs PUBLISHED	Cdd_IpcMpType : numProcs

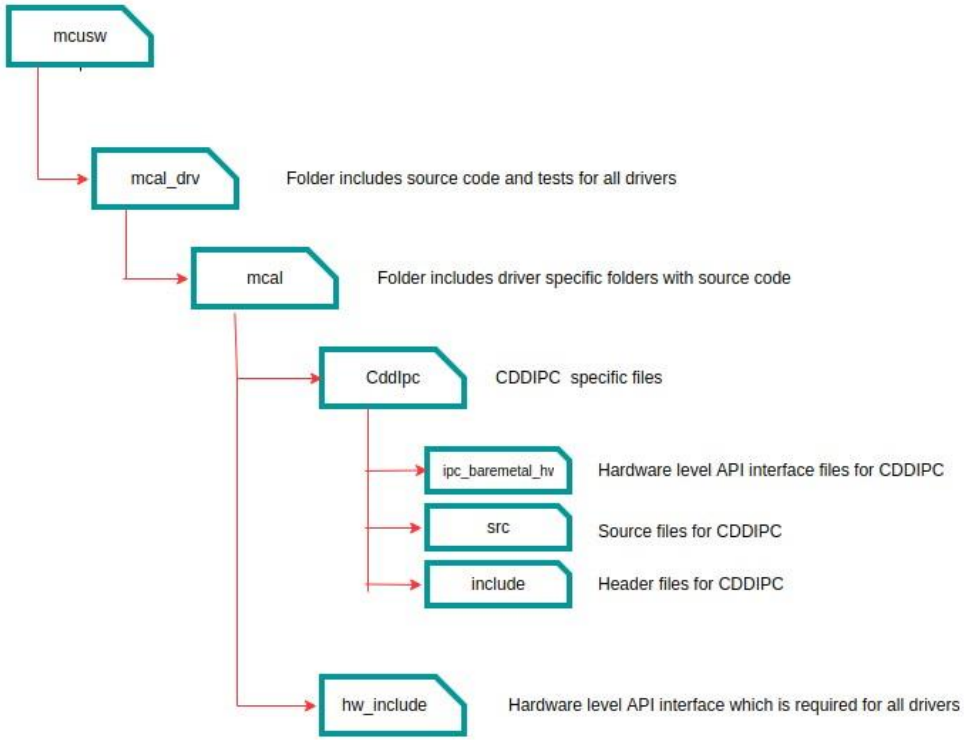
Design Identifier	Description
MCAL-7161 - Cdd_IpcMpType : remoteProcID PUBLISHED	Cdd_IpcMpType : remoteProcID
MCAL-7192 - Cdd_IpcChannelType : localEp PUBLISHED	Cdd_IpcChannelType : localEp

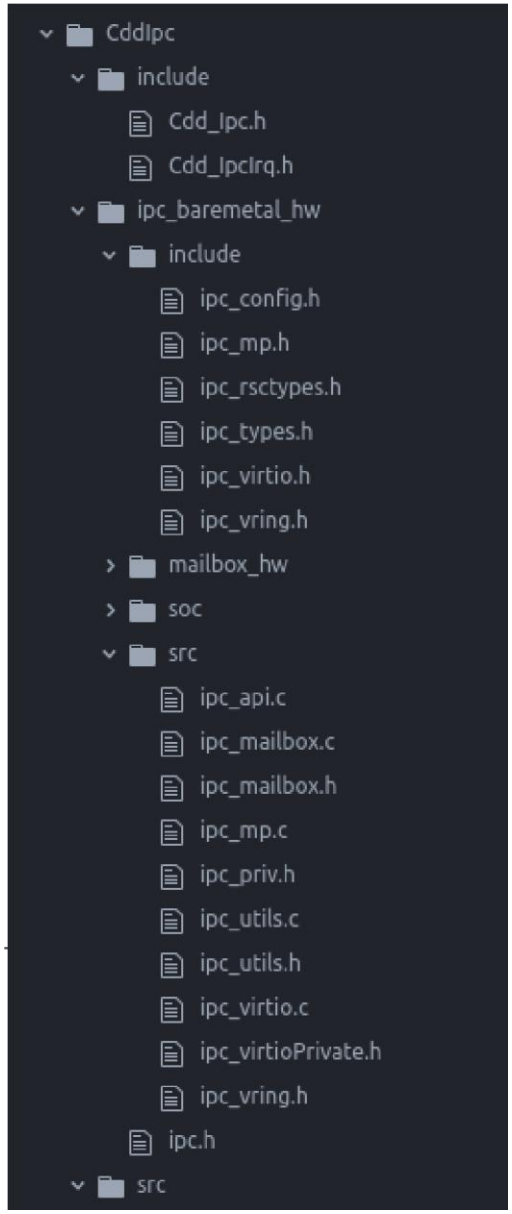
4.2 Directory Structure

The directory structure is as depicted in figures below, the source files can be categorized under “Driver Implementation” and “Configuration” **Driver**

Implemented by

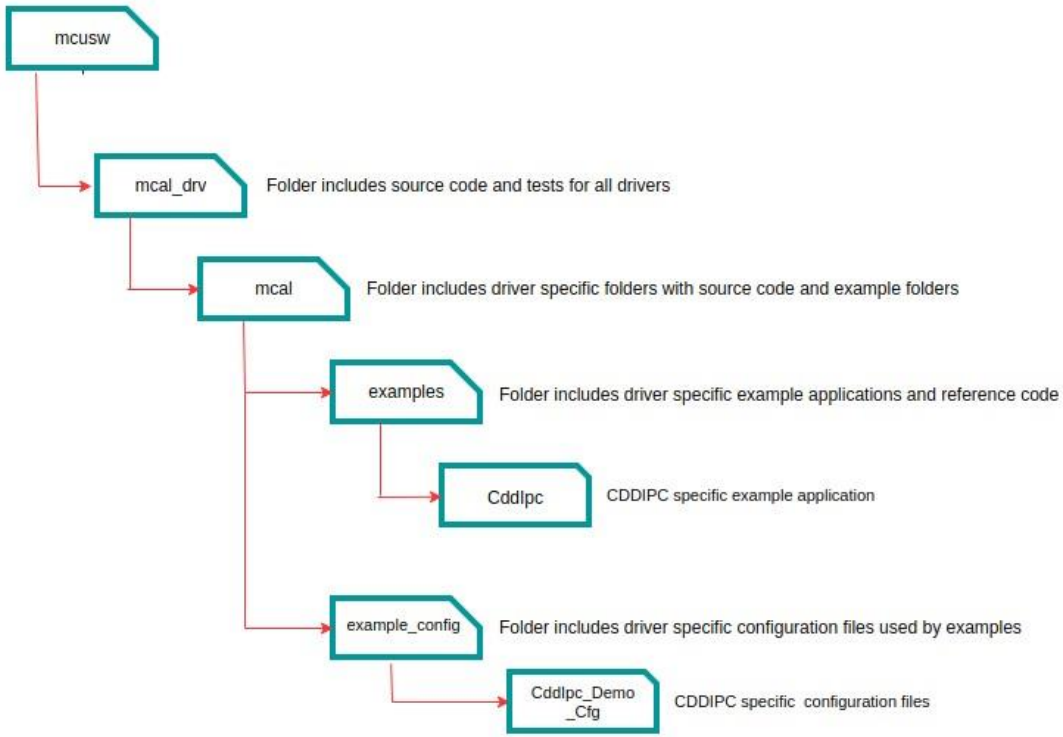
- Cdd_Ipc.h and Cdd_IpcIrq.h : Shall implement the interface provided by the driver
- Cdd_Ipc.c, Cdd_IpcIrq.c : Shall implement the driver functionality
- Cdd_IpcCbk.h : Shall define function prototype that shall be implemented by the applications and invoked by the driver on reception of new message.
- hw_include : Shall be used by example application.


mcusw
mcal_drv
Folder includes source code and tests for all drivers
mcal
Folder includes driver specific folders with source code
CddIpc
CDDIPC specific files
ipc_baremetal_hv
Hardware level API interface files for CDDIPC
src
Source files for CDDIPC
include
Header files for CDDIPC
hw_include
Hardware level API interface which is required for all drivers


CddIpc
include
Cdd_Ipc.h
Cdd_IpcIrq.h
ipc_baremetal_hw
include
ipc_config.h
ipc_mp.h
ipc_rsctypes.h
ipc_types.h
ipc_virtio.h
ipc_vring.h
mailbox_hw
soc
src
ipc_api.c
ipc_mailbox.c
ipc_mailbox.h
ipc_mp.c
ipc_priv.h
ipc_utils.c
ipc_utils.h
ipc_virtio.c
ipc_virtioPrivate.h
ipc_vring.h
ipc.h
src

Example Application

- Cdd_IpcCfg.h and Cdd_IpcCfg.c: Shall implement the generated configuration for pre-compile variant.
- CddIpcApp.c and CddIpcApp.h: Shall implement the example application that demonstrates the use of the driver.


mcusw
mcal_drv
Folder includes source code and tests for all drivers
mcal
Folder includes driver specific folders with source code and example folders
examples
Folder includes driver specific example applications and reference code
CddIpc
CDDIPC specific example application
example_config
Folder includes driver specific configuration files used by examples
CddIpc_Demo
_Cfg
CDDIPC specific configuration files

4.3 Configurator

The AUTOSAR CDD IPC Driver Specification details mandatory parameters that shall be configurable via the configurator. Please refer section 6 of Reference 1 - AUTOSAR 4.3.1.

Following lists this design's specific configurable parameters

4.3.1 Standard configurable parameters

4.3.1.1 CDD IPC General Configuration parameters

Parameter	Usage comment
CddIpcOSCounterId	This shall allow integrators to specify the OS counter instance to be used in OS API GetCounterValue () The driver shall implement timed-wait for all waits (e.g. waiting for reset to complete). This timed wait shall use OS API GetCounterValue ()

Parameter	Usage comment
CddIpcDeviceVariant	This shall allow integrators to select the device variant for which integration is being performed. This parameter shall be used by driver to impose device specific constraints. The user guide shall detail the device specific constraints
CddDevErrorDetect	This parameter turns on ERROR detection and shall be used during development, disabled for production builds
NewMsgNtfyFunc	Specify the integrator defined function that would invoked on reception of new message
CddVersionInfoApi	Enable / Disable Get Version Info service API
CddDeinitApi	Enable / Disable De Initialization of IPC CDD service API
CddIpcAnnounceApi	Enable / Disable Announcement (broadcast) of processors capabilities to other cores. This service API would be mandatory when the remote core hosts Linux
CddRegisterReadBackApi	Enable / Disable service API to Read back of critical registers

Parameter	Usage comment
CddIrqType	Specify category of ISR, Only CAT 2 is supported

Design Identifier	Description
MCAL-7209 - CddIpcOsCounterId PUBLISHED	CddIpcOsCounterId
MCAL-7189 - CddIpcDeviceVariant PUBLISHED	CddIpcDeviceVariant
MCAL-7195 - Cdd_IpcNewMsg PUBLISHED	Cdd_IpcNewMsg
MCAL-7207 - Cdd_IpcGetVersionInfo : Optional API PUBLISHED	Cdd_IpcGetVersionInfo : Optional API
MCAL-7196 - Cdd_IpcDeInit : Optional API PUBLISHED	Cdd_IpcDeInit : Optional API

Design Identifier	Description
MCAL-7240 - Cdd_IpcAnnounce : Optional API PUBLISHED	Cdd_IpcAnnounce : Optional API
MCAL-7220 - CddIpcCategoryOfIsr PUBLISHED	CddIpcCategoryOfIsr
MCAL-7244 - Reception of control message API : Return Values PUBLISHED	Reception of control message API : Return Values
MCAL-7185 - CddIpcDevError PUBLISHED	CddIpcDevError

4.3.1.2 IPC Shared Memory Configuration parameters

Parameter	Usage comment
VertIoRingAddr	Specify the physical address of the shared memory. [Constraints] Please refer (Constraint) & (Cdd IPC Overview)

Parameter	Usage comment
VertIoRingSize	Please retain the recommended configurations. When changing, ensure the size is same across all cores that uses IPC.
VertIoObjSize	Non shared memory, used for book-keeping of VRING. Note that memory shall be allocated in the generate configuration and alignment to 128 byte boundary shall be ensured.
reserved	Reserved for future use

Design Identifier	Description
MCAL-7152 - Cdd_IpcVertIoType : vertIoObjSize PUBLISHED	Cdd_IpcVertIoType : vertIoObjSize
MCAL-7197 - Cdd_IpcVertIoType : vertIoRingAddr PUBLISHED	Cdd_IpcVertIoType : vertIoRingAddr
MCAL-7199 - Cdd_IpcVertIoType : vertIoRingSize PUBLISHED	Cdd_IpcVertIoType : vertIoRingSize

4.3.1.3 CDD IPC Processor Identifier

Parameter	Usage comment
OwnProcID	Select the current processor on which the MCAL/AUTOSAR is hosted

Design Identifier	Description
MCAL-7204 - Cdd_IpcMpType : ownProcID PUBLISHED	Cdd_IpcMpType : ownProcID
MCAL-7210 - Cdd_IpcMpType PUBLISHED	Cdd_IpcMpType

4.3.1.4 CDD IPC Processor Identifier Remote

Parameter	Usage comment
ProcID	Select the list of remote processor ID, which with IPC is desired. Please note that all selected remote processors shall host TI IPC

Design Identifier	Description
MCAL-7161 - Cdd_IpcMpType : remoteProcID PUBLISHED	Cdd_IpcMpType : remoteProcID

CDD IPC Communication Channels

Parameter	Usage comment
CommId	Specify an unique integer that uniquely identifies the communication channel. This ID shall be used in service API's to transmit/receive/notify arrival on new message. Configurator shall support symbolic names for the communication ID's
LocalEp	Local End Point, specify an unique integer that uniquely identifies the communication channel end-point on a given processor. The reserved end-point shall not be accepted by the configurator
RemoteEp	Remote End Point, specify an unique integer that uniquely identifies the communication channel end-point on a associated remote processor. The reserved end-point shall not be accepted by the configurator
RemoteProcID	Remote processor, shall be one of the processor listed in (CDD IPC Processor Identifier Remote)

Parameter	Usage comment
MaxNumMsgQueue	Specify the maximum number of messages that can queued (before received by call to receive service API) (Message Buffer)
MaxMsgSize	Specify the maximum size of all possible messages that could be received. (before received by call to receive service API) (Message Buffer)

- The driver shall reserves space to implement a queue of elements and the size shall be MaxNumMsgQueue * MaxMsgSize bytes.

Design Identifier	Description
MCAL-7192 - Cdd_IpcChannelType : localEp PUBLISHED	Cdd_IpcChannelType : localEp
MCAL-7234 - Core IDs PUBLISHED	Core IDs
MCAL-7218 - Cdd_IpcChannelType PUBLISHED	Cdd_IpcChannelType
MCAL-7245 - Cdd_IpcChannelType : remoteEp PUBLISHED	Cdd_IpcChannelType : remoteEp

Design Identifier	Description
MCAL-7182 - Cdd_IpcChannelType : remoteProcId PUBLISHED	Cdd_IpcChannelType : remoteProcId
MCAL-7231 - Cdd_IpcChannelType : numBufs PUBLISHED	Cdd_IpcChannelType : numBufs
MCAL-7213 - Cdd_IpcChannelType : multiplicity PUBLISHED	Cdd_IpcChannelType : multiplicity
MCAL-7216 - Cdd_IpcChannelType : maxMsgSize PUBLISHED	Cdd_IpcChannelType : maxMsgSize
MCAL-7212 - Cdd_IpcChannelType : Id_symbolic name PUBLISHED	Cdd_IpcChannelType : Id_symbolic name

4.3.2 Implementation specific parameters (computed)

The configurator shall determine the maximum number of channels that are configured and generate a macro to define the same. This shall be used to perform range checks on channel configurations and channel ID provided at driver initialization time. Refer section (MACROS, Data Types & Structures)

4.3.3 Variant Support

The driver shall support VARIANT-PRE-COMPILE only.

4.4 Error Classification

Errors are classified in two categories, development error and runtime / production error.

4.4.1 Development Errors

Type of Error	Related Error code	Value (Hex)
API error return code: Init function failed	CDD_IPC_E_INIT_FAILED	0x01
Service API is called without module initialization	CDD_IPC_E_UNINIT	0x02
API parameter checking: invalid value	CDD_IPC_E_PARAM_POINTER	0x03
API service for initialization is called when already initialized	CDD_IPC_E_ALREADY_INITIALIZED	0x04

Error code indicating wrong configuration	CDD_IPC_E_INVALID_CONFIG	0x05
Error code indicating sending of an message failed	CDD_IPC_E_SEND	0x06
Error code indicating sending of an message failed	CDD_IPC_E_RECEIVE_RETRY	0x07
Error code indicating feature is not supported	CDD_IPC_E_NOT_SUPPORTED	0x08

Design Identifier	Description
MCAL-7226 - Cdd_IpcSendMsg : Arg Check : ID PUBLISHED	Cdd_IpcSendMsg : Arg Check : ID
MCAL-7225 - Cdd_IpcSendMsg : Arg Check : NULL PTR PUBLISHED	Cdd_IpcSendMsg : Arg Check : NULL PTR
MCAL-7205 - Cdd_IpcSendMsg : Arg Check : Size PUBLISHED	Cdd_IpcSendMsg : Arg Check : Size
MCAL-7177 - Cdd_IpcSendMsg : full queue PUBLISHED	Cdd_IpcSendMsg : full queue

Design Identifier	Description
MCAL-7193 - Cdd_IpcReceiveMsg : No pending messages PUBLISHED	Cdd_IpcReceiveMsg : No pending messages
MCAL-7176 - Cdd_IpcReceiveMsg : Arg Check : NULL PTR PUBLISHED	Cdd_IpcReceiveMsg : Arg Check : NULL PTR
MCAL-7155 - Cdd_IpcAnnounce : Arg Check : NULL PTR PUBLISHED	Cdd_IpcAnnounce : Arg Check : NULL PTR
MCAL-7179 - Cdd_IpcGetVersionInfo : Arg Check : NULL PTR PUBLISHED	Cdd_IpcGetVersionInfo : Arg Check : NULL PTR
MCAL-7160 - Service APIs : State Checks PUBLISHED	Service APIs : State Checks

4.4.2 Error Detection

The detection of development errors is configurable (ON / OFF) at pre-compile time. The switch CddDevErrorDetect shall activate or deactivate the detection of all development errors.

4.4.3 Error notification (DET)

All detected development errors are reported to Det_ReportError service of the Development Error Tracer (DET).

4.4.4 Runtime Errors

The following runtime/production errors shall be detectable by CDD IPC driver

Type of Error	Related Error code	Value (Hex)
This error shall be reported when Mailbox is not functional	CDD_IPC_E_HARDWARE_ERROR	Defined By Integrator

4.4.5 Error notification (DEM)

All detected run time errors shall be reported to Dem_ReportErrorStatus () service of the Diagnostic Event Manager (DEM).

Design Identifier	Description
MCAL-7230 - Dem PUBLISHED	Dem
MCAL-7185 - CddIpcDevError PUBLISHED	CddIpcDevError

5 Implementation Details

5.1 Data structures and resources

MACROS, Data Types & Structures

The sections below lists some of key data structures that shall be implemented and used in driver implementation

Maximum number of channels

Identifier	Comments
CDD_IPC_CORE_ID_MAX	Defines the maximum number of remote cores supported by this implementation. This macro shall be used to allocate memory (statically) in the driver implementation.
Cdd_IpcChannelBufType	Channel RP Msg buffer type .Used to hold the received buffer, before apps can pick it up
Cdd_IpcVertIoType	Defines Shared Memories for VRING and VRING OBJECT ,VRING is the shared memory between cores and VRING OBJECT is used to implement queue. Please refer design/user guide for details.
Cdd_IpcChannelType	Defines logical communication channel between cores

Identifier	Comments
Cdd_IpcConfigType	CDD IPC Configuration type
Cdd_IpcRegRbValues	Defines the configuration registers of the MCAL modules

Cdd_IpcMpType

Used to specify the core identifiers, these values shall be generated by the configurator and not explicitly by the user of this module.

Type	Identifier	Comments
uint32	ownProcID	Defines processor ID on which MCAL/AUTOSAR is being hosted
uint32	numProcs	Number of remote processor which with IPC is desired
uint32	remoteProcID	Array of uin32, that specifies the remote processor identifier
uint32	reserved	Reserved for future use

Design Identifier	Description
MCAL-7210 - Cdd_IpcMpType PUBLISHED	Cdd_IpcMpType
MCAL-7204 - Cdd_IpcMpType : ownProcID PUBLISHED	Cdd_IpcMpType : ownProcID
MCAL-7223 - Cdd_IpcMpType : numProcs PUBLISHED	Cdd_IpcMpType : numProcs
MCAL-7161 - Cdd_IpcMpType : remoteProcID PUBLISHED	Cdd_IpcMpType : remoteProcID

Cdd_IpcVertIoType

Defines Shared Memories for VRING and VRING OBJECT, these values shall be generated by the configurator and not explicitly by the user of this module.

Type	Identifier	Comments
void *	vertIoRingAddr	Defines address that shall be shared between cores, also refer (Assumptions)

Type	Identifier	Comments
uint32	vertIoRingSize	Size in number of bytes
uint32	reserved	Reserved for future use

Design Identifier	Description
MCAL-7194 - Cdd_IpcVertIoType PUBLISHED	Cdd_IpcVertIoType
MCAL-7180 - Cdd_IpcVertIoType : vertIoObjAddr PUBLISHED	Cdd_IpcVertIoType : vertIoObjAddr

Cdd_IpcChannelType

Defines logical communication channel between cores, these values shall be generated by the configurator and not explicitly by the user of this module.

Type	Identifier	Comments
uint32	id	Unique identifiers for a channel

Type	Identifier	Comments
uint32	localEp	Local End Point identifier, on which MCAL/AUTOSAR is hosted
uint32	remoteProcId	Remote Processor Identifier
uint32	numMsgQueued	Maximum depth of the queue, that holds received messages
uint32	maxMsgSize	Maximum size of the message that could be received
uint32	reserved	Reserved for future use

Design Identifier	Description
MCAL-7218 - Cdd_IpcChannelType PUBLISHED	Cdd_IpcChannelType
MCAL-7243 - Cdd_IpcChannelType : Id PUBLISHED	Cdd_IpcChannelType : Id

Design Identifier	Description
MCAL-7192 - Cdd_IpcChannelType : localEp PUBLISHED	Cdd_IpcChannelType : localEp
MCAL-7245 - Cdd_IpcChannelType : remoteEp PUBLISHED	Cdd_IpcChannelType : remoteEp
MCAL-7182 - Cdd_IpcChannelType : remoteProcId PUBLISHED	Cdd_IpcChannelType : remoteProcId
MCAL-7231 - Cdd_IpcChannelType : numBufs PUBLISHED	Cdd_IpcChannelType : numBufs
MCAL-7216 - Cdd_IpcChannelType : maxMsgSize PUBLISHED	Cdd_IpcChannelType : maxMsgSize

Cdd_IpcConfigType

CDD IPC Configuration type, these values shall be generated by the configurator and not explicitly by the user of this module.

Type	Identifier	Comments
Cdd_IpcMpType	coreIds	Used to specify the core identifiers refer (Cdd_IpcMpType)
Cdd_IpcVertIoType	vertIoCfg	VertIO configurations refer (Cdd_IpcVertIoType)
uint32	channelCount	Number of communication channels configured by the integrator
Cdd_IpcChannelType*	pChCfg	Pointer to constant, refer (Cdd_IpcChannelType)
uint32	reserved	Reserved for future use

Design Identifier	Description
MCAL-7233 - Cdd_IpcConfigType PUBLISHED	Cdd_IpcConfigType
MCAL-7229 - Cdd_IpcConfigType : numCh PUBLISHED	Cdd_IpcConfigType : numCh

Cdd_IpcRegRbValues

Name	Type	Range	Comments
numRegisters	uint32	0 to 0xFFFFFFFF	Will specify number of registers values provided
regValues	uint32	0 to 0xFFFFFFFF	Values of critical registers that's read and provided
reserved	uint32	0	Reserved for future use

Global Variables

This design expects that implementation will require to use following global variables.

Variable	Type	Description	Default Value
Cdd_IpcDrvStatus	uint32	Initialization status of the driver is maintained	FALSE
Cdd_IpcDrvObj	Cdd_IpcDriverObjType	IPC driver object, local to the implementation and scope shall be limited to Cdd_Ipc.c	Un defined

Design Identifier	Description
MCAL-7221 - Cdd_IpcInit : Initialization : states PUBLISHED	Cdd_IpcInit : Initialization : states

5.2 Dynamic Behavior - Control Flow Diagram

States

CDD IPC shall maintain two distinct states **Initialized** & **Un Initialized Initialized**

State <====> Un Initialized State

- **Initialized State**
- All service API's shall be honored.
- All configured communication channels created.
- Shall be able to receive messages from configured remote core on configured end-point.
- **Un Initialized State**
- All service API's shall NOT be honored.
- Any service API invoked shall return CDD_IPC_E_INIT_FAILED

Design Identifier	Description
MCAL-7221 - Cdd_IpcInit : Initialization : states PUBLISHED	Cdd_IpcInit : Initialization : states
MCAL-7184 - Cdd_IpcInit : Initialization : states PUBLISHED	Cdd_IpcInit : Initialization : states
MCAL-7160 - Service APIs : State Checks PUBLISHED	Service APIs : State Checks

5.3 Dynamic Behavior - Data Flow Diagram

Not Applicable

5.4 Application Parameters

Cdd_IpcNewMessageNotify

Parameter	Description	Possible Value ranges	Unit of Value	Default Value	Variant
uint32 commId	commId is the value specified by integrator while creating the communication channel id		-	-	NA

Cdd_IpcNewCtrlMessageNotify

Parameter	Description	Possible Value ranges	Unit of Value	Default Value	Variant
uint32 remoteProcId	One of the Remote processor ID Specified During Initialization		-	-	N.A

Cdd_IpcInit

Parameter	Description	Possible Value ranges	Unit of Value	Default Value	Variant
void					

Cdd_IpcDeinit

Parameter	Description	Possible Value ranges	Unit of Value	Default Value	Variant
void					

Cdd_IpcSendMsg

Parameter	Description	Possible Value ranges	Unit of Value	Default Value	Variant
chId	chId Refers to communication ID specified while configuring this module.		-	-	N.A
pBuf	Non NULL_PTR that describes the message that has to sent		-	-	N.A
bufLen	Message length in bytes		-	-	N.A

Cdd_IpcReceiveMsg

Parameter	Description	Possible Value ranges	Unit of Value	Default Value	Variant
chId	chId Refers to communication ID specified while configuring this module.		-	-	N.A

Parameter	Description	Possible Value ranges	Unit of Value	Default Value	Variant
pBuf	Non NULL_PTR that can hold the received message. Call shall ensure sufficient memory is available, shall be greater than or equal to maximum size specified in configuration. Refer maxMsgSize		-	-	N.A
bufLen	Message length in bytes		-	-	N.A

Cdd_IpcAnnounce

Parameter	Description	Possible Value ranges	Unit of Value	Default Value	Variant
pBuf	Non NULL_PTR that describes the message that has to sent		-	-	N.A
chId	A Valid Communication Channel Identifier		-	-	N.A

Cdd_IpcGetVersionInfo

Parameter	Description	Possible Value ranges	Unit of Value	Default Value	Variant
VersionInfoPtr	A pointer of type Std_VersionInfoType, which holds the read back values		-	-	N.A

Cdd_IpcRegisterReadBack

Parameter	Description	Possible Value ranges	Unit of Value	Default Value	Variant
RegRbPtr	A pointer of type Cdd_IpcRegRbValues, which holds the read back values		-	-	N.A

Cdd_IpcReceiveCtrlMsg

Parameter	Description	Possible Value ranges	Unit of Value	Default Value	Variant
pRemoteProcId	Holds the remote processor identifier, one of valid remote processors specified while initializing, refer RemoteProcID		-	-	N.A
pRemoteEndPt	Holds the remote processor end point, that is the originator of this control message		-	-	N.A

Parameter	Description	Possible Value ranges	Unit of Value	Default Value	Variant
pBuf	Non NULL_PTR that can hold the received message. Call shall ensure sufficient memory is available, shall be greater than or equal to maximum size specified in configuration. Refer maxMsgSize		-	-	N.A
bufLen	Received message length in bytes		-	-	N.A

Cdd_IpcIsInitDone

Parameter	Description	Possible Value ranges	Unit of Value	Default Value	Variant
void					

Cdd_IpcGetMaxMsgSize

Parameter	Description	Possible Value ranges	Unit of Value	Default Value	Variant
chId	Communication channel identifier				

5.5 Safety Diagnostic Features

NAVSS.MBOX4 -Indexed mailbox data in software

To insert an index into each mailbox message that increments and can be used for validation of proper message sequence.

The CDDIPC MCAL driver provides the API - **Cdd_Ipccheckmailboxindex** to check mismatch of index and notify error for this diagnostic feature.

NAVSS.MBOX6-Periodic software check of mailbox status

To verify the Mailbox state in the interrupt handler for new message or fifo not-full interrupt before taking action based on the interrupt and to check periodically the status of the mailbox, in order to catch the case of interrupt not fired.

The CDDIPC MCAL driver provides the API - **Cdd_IpcGetmailboxstatus** to check the periodic status of mailbox and verify the state in interrupt handler for new message or fifo not-full interrupt.

NAVSS.MBOX8-Software readback of written configuration

Software Readback of Written Configuration ensures that the configuration register are written with the expected value. Periodic readback of configuration registers can provide a diagnostic for inadvertent writes to these registers.

The CDDIPC MCAL driver provides the API - **Cdd_IpcRegisterReadBack** to readback static and written configuration registers to implement this diagnostic feature.

6 Low Level Definitions

6.1 Driver API's

Sections below defines the expected API's to part of this implementation

6.1.1 Cdd_IpcNewMessageNotify

Is a function implemented by the application, with prototype as void Cdd_IpcNewMessageNotify (uint32 commId), where commId is the value specified by integrator while creating the communication channel id

	Description	*Comments*
Function Name	Cdd_IpcNewMessageNotify	Is a symbolic name, integrators can specify desired name
Syntax	void Cdd_IpcNewMessageNotify(uint32 commId)	Shall be implemented by the MCAL consumer
Called Context	Interrupt	This function would be invoked by driver in interrupt context. Also refer Flow chart for implementation of the ISR(Cdd_IpcNewMessageNotify ISR).

Reentrancy	Non Reentrant	
Parameter in	uint32 commId	commId is the value specified by integrator while creating the communication channel id
Return Value	None	NA

Design Identifier	Description
MCAL-7195 - Cdd_IpcNewMsg PUBLISHED MCAL-7203 - New Control Message PUBLISHED	Cdd_IpcNewMsg
MCAL-7208 - Extern Cdd_IpcNewMsgCb PUBLISHED	Extern Cdd_IpcNewMsgCb

6.1.2 Cdd_IpcNewCtrlMessageNotify

Is a function implemented by the application, with prototype as **void Cdd_IpcNewCtrlMessageNotify (uint32 remoteProcId)** , where remoteProcId is remote processor ID Specified During Initialization

	Description	*Comments*
Function Name	Cdd_IpcNewCtrlMessageNotify	Is a symbolic name, integrators can specify desired name
Syntax	void Cdd_IpcNewCtrlMessageNotify(uint32 remoteProcId)	Shall be implemented by the MCAL consumer
Called Context	Interrupt	This function would be invoked by driver in interrupt context. Also refer Control End Point
Reentrancy	Non Reentrant	
Parameter in	uint32 remoteProcId	One of the Remote processor ID Specified During Initialization
Return Value	None	NA

Design Identifier	Description
MCAL-7186 - extern Cdd_IpcNewCtrlMessageNotify PUBLISHED	extern Cdd_IpcNewCtrlMessageNotify

Design Identifier	Description
MCAL-7203 - New Control Message PUBLISHED	New Control Message

6.1.3 Cdd_IpcInit

	Description	*Comments*
Service Name	Cdd_IpcInit	First API to be invoked to initialize the module
Syntax	Std_ReturnType Cdd_IpcInit(void)	Service for CDD Initialization
Service ID	0x02	
Sync / Async	Sync	
Reentrancy	Non Reentrant	

Parameter in	none	NA
Parameters out	none	NA
Return Value	Standard return type	E_OK or CDD_IPC_E_INIT_FAILED in case of initialization failure id, or CDD_IPC_E_ALREADY_INITIALIZED in case of reinitialization

Design Identifier	Description
MCAL-7153 - Cdd_IpcInit : Initialization : Variant Support PUBLISHED	Cdd_IpcInit : Initialization : Variant Support
MCAL-7154 - Cdd_IpcInit : Creation of comm channels PUBLISHED	Cdd_IpcInit : Creation of comm channels
MCAL-7166 - Cdd_IpcInit : Initialization : Interrupt PUBLISHED	Cdd_IpcInit : Initialization : Interrupt
MCAL-7170 - Cdd_IpcInit : Initialization : Checks : vertIoRingSize PUBLISHED	Cdd_IpcInit : Initialization : Checks : vertIoRingSize
MCAL-7178 - Cdd_IpcInit : Initialization : Arguments PUBLISHED	Cdd_IpcInit : Initialization : Arguments

Design Identifier	Description
MCAL-7184 - Cdd_IpcInit : Initialization : states PUBLISHED	Cdd_IpcInit : Initialization : states
MCAL-7200 - Cdd_IpcInit : Initialization : Checks : Cdd_IpcChannelType : bufSize PUBLISHED	Cdd_IpcInit : Initialization : Checks : Cdd_IpcChannelType : bufSize
MCAL-7221 - Cdd_IpcInit : Initialization : states PUBLISHED	Cdd_IpcInit : Initialization : states
MCAL-7232 - Cdd_IpcInit : Initialization : Variant PUBLISHED	Cdd_IpcInit : Initialization : Variant
MCAL-7236 - Cdd_IpcInit : Initialization : reset PUBLISHED	Cdd_IpcInit : Initialization : reset

6.1.4 Cdd_IpcDeinit

	Description	***Comments***

Service Name	Cdd_IpcDeinit	Last API to be invoked to de initialize the module, can be turned OFF CddDeinitApi
Syntax	Std_ReturnType Cdd_IpcDeinit (void)	Service for CDD Initialization
Service ID	0x08	
Sync / Async	Sync	
Reentrancy	Non Reentrant	
Parameter in	none	NA
Parameters out	none	NA
Return Value	Standard return type	E_OK

Design Identifier	Description
MCAL-7196 - Cdd_IpcDeInit : Optional API PUBLISHED	Cdd_IpcDeInit : Optional API

6.1.5 Cdd_IpcSendMsg

	Description	***Comments***
Service Name	Cdd_IpcSendMsg	Service for sending an message to remote cores
Syntax	Std_ReturnType Cdd_IpcSendMsg(uint32 chId, void *pBuf, uint32 bufLen)	
Service ID	0x03	
Sync / Async	Sync	
Reentrancy	Non Reentrant	

Parameter in	chId	chId Refers to communication ID specified while configuring this module.
Parameter in	pBuf	Non NULL_PTR that describes the message that has to sent
Parameter in	bufLen	Message length in bytes
Return Value	Standard return type	E_OK on successful transmission, CDD_IPC_E_SEND on error and CDD_IPC_E_UNINIT when initialized

Design Identifier	Description
MCAL-7168 - Cdd_IpcSendMsg : Send Msg PUBLISHED	Cdd_IpcSendMsg : Send Msg
MCAL-7242 - Cdd_IpcSendMsg : Non Blocking PUBLISHED	Cdd_IpcSendMsg : Non Blocking
MCAL-7169 - Cdd_IpcSendMsg : Non Re Entrant PUBLISHED	Cdd_IpcSendMsg : Non Re Entrant

Design Identifier	Description
MCAL-7226 - Cdd_IpcSendMsg : Arg Check : ID PUBLISHED	Cdd_IpcSendMsg : Arg Check : ID
MCAL-7225 - Cdd_IpcSendMsg : Arg Check : NULL PTR PUBLISHED	Cdd_IpcSendMsg : Arg Check : NULL PTR
MCAL-7205 - Cdd_IpcSendMsg : Arg Check : Size PUBLISHED	Cdd_IpcSendMsg : Arg Check : Size
MCAL-7177 - Cdd_IpcSendMsg : full queue PUBLISHED	Cdd_IpcSendMsg : full queue

6.1.6 Cdd_IpcReceiveMsg

	Description	***Comments***
Service Name	Cdd_IpcReceiveMsg	Service for reception of N bytes of data from remote cores
Syntax	Std_ReturnType Cdd_IpcReceiveMsg(uint32 chId, void *pBuf, uint32 bufLen)	

Service ID	0x04	
Sync / Async	Sync	
Reentrancy	Non Reentrant	
Parameter in	chId	chId Refers to communication ID specified while configuring this module
Parameter in out	pBuf	Non NULL_PTR that can hold the received message. Call shall ensure sufficient memory is available, shall be greater than or equal to maximum size specified in configuration. Refer maxMsgSize
Parameter in	bufLen	Message length in bytes
Return Value	Standard return type	E_OK on successful reception, CDD_IPC_E_RECEIVE_RETRY on no messages, CDD_IPC_E_UNINIT, when uninitialized.

Design Identifier	Description
MCAL-7151 - Cdd_IpcReceiveMsg : Receive Msg PUBLISHED	Cdd_IpcReceiveMsg : Receive Msg
MCAL-7162 - Cdd_IpcReceiveMsg : Non Reentrant PUBLISHED	Cdd_IpcReceiveMsg : Non Reentrant
MCAL-7174 - Cdd_IpcReceiveMsg : Non Blocking PUBLISHED	Cdd_IpcReceiveMsg : Non Blocking
MCAL-7176 - Cdd_IpcReceiveMsg : Arg Check : NULL PTR PUBLISHED	Cdd_IpcReceiveMsg : Arg Check : NULL PTR
MCAL-7193 - Cdd_IpcReceiveMsg : No pending messages PUBLISHED	Cdd_IpcReceiveMsg : No pending messages

Design Identifier	Description
MCAL-7235 - Cdd_IpcReceiveMsg : Update numbre of bytes PUBLISHED	Cdd_IpcReceiveMsg : Update numbre of bytes

6.1.7 Cdd_IpcAnnounce

	Description	***Comments***
Service Name	Cdd_IpcAnnounce	Used to broadcast capabilities of this core to all other cores, can be turned OFF CddIpcAnnounceApi
Syntax	Std_ReturnType Cdd_IpcAnnounce(void *pBuf, uint32 chId)	Service broadcast of message to all cores for a given commnication channel
Service ID	0x05	
Sync / Async	Sync	

Reentrancy	Non Reentrant	
Parameter in	pBuf	Non NULL_PTR that describes the message that has to sent
Parameter in	chId	A Valid Communication Channel Identifier
Return Value	Standard return type	E_OK on successful transmission and CDD_IPC_E_SEND on error and CDD_IPC_E_UNINIT when initialized. Also check Constraint

Design Identifier	Description
MCAL-7246 - Cdd_IpcAnnounce : Non Blocking PUBLISHED	Cdd_IpcAnnounce : Non Blocking
MCAL-7241 - Cdd_IpcAnnounce : Announce PUBLISHED	Cdd_IpcAnnounce : Announce
MCAL-7211 - Cdd_IpcAnnounce : Optional PUBLISHED	Cdd_IpcAnnounce : Optional

Design Identifier	Description
MCAL-7217 - Cdd_IpcAnnounce : Message Length PUBLISHED	Cdd_IpcAnnounce : Message Length
MCAL-7219 - Cdd_IpcAnnounce : Non Reentrant PUBLISHED	Cdd_IpcAnnounce : Non Reentrant
MCAL-7155 - Cdd_IpcAnnounce : Arg Check : NULL PTR PUBLISHED	Cdd_IpcAnnounce : Arg Check : NULL PTR

6.1.8 Cdd_IpcGetVersionInfo

	Description	***Comments***
Service Name	Cdd_IpcGetVersionInfo	Can potentially be turned OFF, via configuration parameter CddVersionInfoApi
Syntax	void Cdd_IpcGetVersionInfo(Std_VersionInfoType VersionInfoPtr)	

Service ID	0x01	
Sync / Async	Sync	
Reentrancy	Reentrant	
Parameters out	VersionInfoPtr	A pointer of type Std_VersionInfoType, which holds the read back values
Return Value	None	

Design Identifier	Description
MCAL-7206 - Cdd_IpcGetVersionInfo : Functionality PUBLISHED	Cdd_IpcGetVersionInfo : Functionality
MCAL-7179 - Cdd_IpcGetVersionInfo : Arg Check : NULL PTR PUBLISHED	Cdd_IpcGetVersionInfo : Arg Check : NULL PTR
MCAL-7157 - Cdd_IpcGetVersionInfo : Optional PUBLISHED	Cdd_IpcGetVersionInfo : Optional

6.1.9 Cdd_IpcReceiveCtrlMsg

	Description	***Comments***
Service Name	Cdd_IpcReceiveCtrlMsg	Service for reception of N bytes of control data from remote cores
Syntax	Std_ReturnType Cdd_IpcReceiveCtrlMsg(uint32 *pRemoteProcId, uint32 *pRemoteEndPt, void *pBuf, uint32 bufLen)	
Service ID	0x09	
Sync / Async	Sync	
Reentrancy	Non Reentrant	
Parameter in out	pRemoteProcId	Holds the remote processor identifier, one of valid remote processors specified while initializing, referRemoteProcID

Parameter in out	pRemoteEndPt	Holds the remote processor end point, that is the originator of this control message
Parameter in out	pBuf	Non NULL_PTR that can hold the received message. Call shall ensure sufficient memory is available, shall be greater than or equal to maximum size specified in configuration. Refer maxMsgSize
Parameter in	bufLen	Received message length in bytes
Return Value	Standard return type	E_OK on successful reception, E_NOT_OK on no messages, CDD_IPC_E_UNINIT, when uninitialized and CDD_IPC_E_PARAM_POINTER when any one of the pointer is NULL

Design Identifier	Description
MCAL-7167 - Reception of control message API **PUBLISHED**	Reception of control message API
MCAL-7228 - Reception of control message API : Non Blocking **PUBLISHED**	Reception of control message API : Non Blocking

Design Identifier	Description
MCAL-7244 - Reception of control message API : Return Values PUBLISHED	Reception of control message API : Return Values
MCAL-7191 - Reception of control message API : Arg Check : NULL PTR PUBLISHED	Reception of control message API : Arg Check : NULL PTR
MCAL-7201 - Reception of control message API : Return values PUBLISHED	Reception of control message API : Return values

6.1.10 Cdd_IpcIsInitDone

	Description	***Comments***
Service Name	Cdd_IpcIsInitDone	Returns TRUE if Cdd_ipcInit is completed else FALSE
Syntax	boolean Cdd_IpcIsInitDone(void)	
Service ID	0x0A	

Sync / Async	Sync	
Reentrancy	Reentrant	
Return Value	Boolean	Returns TRUE if Cdd_IpcInit was completed else FALSE

6.1.11 Cdd_IpcGetMaxMsgSize

	Description	***Comments***
Service Name	Cdd_IpcGetMaxMsgSize	Returns max msg size for the specified channel
Syntax	uint32 Cdd_IpcGetMaxMsgSize(uint32 chId)	
Service ID	0x0B	
Sync / Async	Sync	

Reentrancy	Reentrant	
Return Value	uint32	Returns max msg size for the specified channel

Design Identifier	Description
MCAL-7216 - Cdd_IpcChannelType : maxMsgSize PUBLISHED	Cdd_IpcChannelType : maxMsgSize

6.1.12 Cdd_IpcRegisterReadBack

As noted from previous implementation, the mailbox configuration registers could potentially be corrupted by other entities (s/w or h/w). One of the recommended detection methods would be to periodically read-back the configuration and confirm configuration is consistent. The service API defined below shall be implemented to enable this detection.

	Description	***Comments***
Service Name	Cdd_IpcRegisterReadBack	Can potentially be turned OFF

Syntax	Std_ReturnType Cdd_IpcRegisterReadBack(Cdd_IpcRegRbValues *RegRbPtr)	Cdd_IpcRegRbValues defines the type that holds critical values. This service can be turned OFF CddRegisterReadBackApi
Service ID	0x07	
Sync / Async	Sync	
Reentrancy	Non Reentrant	
Parameters out	RegRbPtr	A pointer of type Cdd_IpcRegRbValues, which holds the read back values
Return Value	Standard return type	E_OK or E_NOT_OK in case of error

The critical register set shall be determined at implementation.

Design Identifier	Description
MCAL-7224 - Cdd_IpcRegisterReadBack : Optional API PUBLISHED	Cdd_IpcRegisterReadBack : Optional API

Design Identifier	Description
MCAL-7158 - CDDIPC: Safety Diagnostics: NAVSS.MBOX8: Software readback of written configuration PUBLISHED	Safety Diagnostics: NAVSS.MBOX8: Software readback of written configuration

6.1.13 **Cdd_Ipccheckmailboxindex**

	Description	***Comments***
Function Name	Cdd_Ipccheckmailboxindex	Returns index of mailbox for each mailbox instance.
Syntax	Std_ReturnType Cdd_Ipccheckmailboxindex(P2VAR(Cdd_IpcChannelType*localEp), P2VAR(Cdd_IpcChannelType*remoteEp))	
Service ID	0x0C	
Sync / Async	Sync	

Reentrancy	Non Reentrant	
Parameter in	localEp, remoteEp	A pointer of type ,Cdd_IpcChannelType which holds the endpoint identifiers.
Return Value	None	NA

Design Identifier	Description
MCAL-7173 - CDDIPC: Safety Diagnostics: NAVSS.MBOX4: Indexed mailbox data in software PUBLISHED	Safety Diagnostics: NAVSS.MBOX4: Indexed mailbox data in software

6.1.14 Cdd_IpcGetmailboxstatus

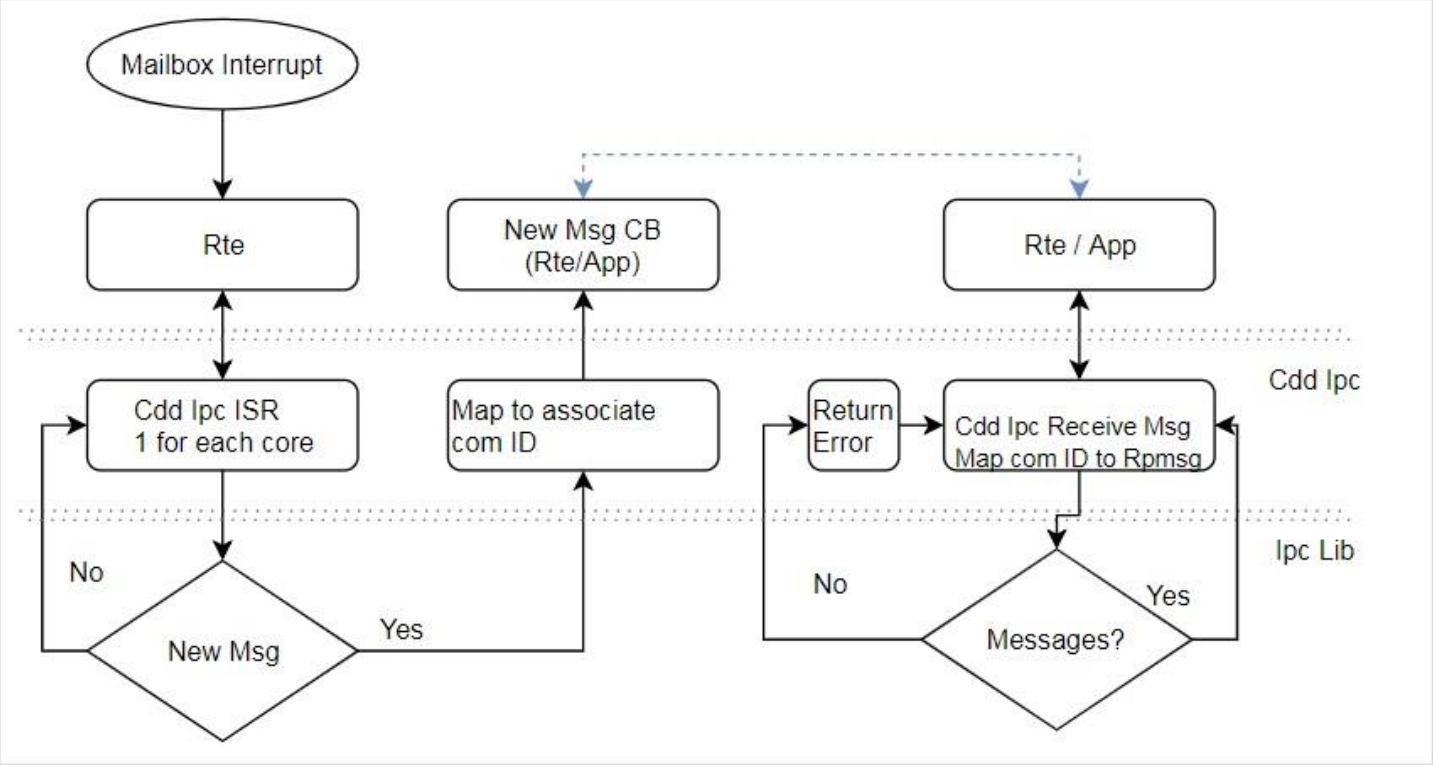
	Description	***Comments***
Function Name	Cdd_IpcGetmailboxstatus	Returns status of mailbox for the specified channel.

Syntax	Std_ReturnType Cdd_Ipccheckmailboxindex(void *ipc_MailboxData,void *Ipc_MailboxData)	
Service ID	0x0D	
Sync / Async	Sync	
Reentrancy	Non Reentrant	
Parameter in	None	NA
Return Value	None	NA

Design Identifier	Description
MCAL-7150 - CDDIPC: Safety Diagnostics: NAVSS.MBOX5: ISR reads Mailbox state PUBLISHED	Safety Diagnostics: NAVSS.MBOX5: ISR reads Mailbox state

Cdd_IpcNewMessageNotify ISR

The flow chart below depict the behaviour of ISR on reception of mailbox non-empty interrupt


Mailbox Interrupt
Rte
New Msg CB
(Rte/App)
Rte / App
Cdd Ipc
Cdd Ipc ISR
1 for each core
Map to associate
com ID
Return
Error
Cdd Ipc Receive Msg
Map com ID to Rpmsg
Ipc Lib
No
New Msg
Yes
No
Yes
Messages?

New Message ISR Flow chart & Reception Service API call

Design Identifier	Description
MCAL-7208 - Extern Cdd_IpcNewMsgCb PUBLISHED	Extern Cdd_IpcNewMsgCb

7 Performance Objectives

7.1 Resource Consumption Objectives

	ROM - Data(KB)	RAM - Program(KB)	RAM - Data(KB)	Stack Size (KB)	EEPROM (KB)	% CPU Utilization
5	NA	NA	1	2	NA	NA

ROM - Program(KB)

7.2 Critical timing and Performance

Not Applicable

8 Decision Analysis & Resolution (DAR)

Sections below list some of the important design decisions and rational behind those decision.

8.1 Allocation of memory

No.	Decision Criteria	Alternatives	Selected alternative	Rationale	Trade-offs

1	Simpler interface and minimize potential errors	• **Allocated by user of this module** The user / integrator allocates the required memory and provides a pointer to the allocated memory while configuring this driver
		  • **Advantages:**
		  • Complete control for the integrators, size and location of the allocated memory
		  • **Disadvantages:**
		  • Additional configuration parameter
		  • Location specified via a global memory
			Use of **Local Allocation**	To minimize the checks and enhanced ease-of-use, **Local Allocation** is chosen	

No.	Decision Criteria	Alternatives	Selected alternative	Rationale	Trade-offs
		• Additional checks in the driver required for alignment and null pointer checks			

No.	Decision Criteria	Alternatives	Selected alternative	Rationale	Trade-offs

		• **Local Allocation**Driver allocates the memory statically, i.e. via an array.
		  • **Advantages:**
		  • Minimal configuration parameters
		  • Alignment can be easily enforced
		  • **Disadvantages:**
		  • The size of memory is computed and integrator will have to analyze the system memory requirement post compilation of the driver			

9 Testing Guidelines

The sections below identify some of the aspects of design that would require emphasis during testing of this design implementation

- **Boundary Checks**
- Since variable length messages could be transmitted, tests on message size range shall be performed.
- Ensure associated error codes are returned on error
- **Latency Measurements**
- Test cases shall ensure, latencies are measured for transmission and reception
- **Concurrency**
- Since a core can communicate with multiple cores on different channels, data integrity checks shall be performed when communicating with multiple cores on multiple channels, concurrently.

10 Template Revision History

Author Name	Description	Version	Date
Krishna	Updated based on ASPICE requirements	0.8	20 Aug 2020
Krishna	Updated based on the feedback from Jon N	0.9	09 Oct 2020
Krishna	Updated the traceability scheme	1.0	17 Dec 2020
Yaniv Machani	Initial version	0.1	03 Oct 2018
Yaniv Machani	Updated to include EP views	0.4	02 Nov 2018
Yaniv Weizman	Restructuring and editing to further meet the A-SPICE and EP requirements	0.5	27 Dec 2018
Yaniv Weizman	Adding link to Architecture review template	0.6	22 Oct 2019

Author Name	Description	Version	Date
Yaniv Weizman	Adding requirement type column for requirements table (Functional/Non-Functional). Adding DAR table	0.65	13 Nov 2019
Yaniv Weizman	Adding tables for Testing guidelines	0.7	18 Nov 2019