Project Name:	**TDA9950_SW**
Document Type:	**Manual**
Document Title:	**User's Manual**
Document Number:	**NA**
Author:	Nicolas Mutz
Curent version:	1.3
Status:	Proposed
Date:	12-June-2009
Pages:	
Summary:	This document describes how to use the tmdlHdmiCEC software.
Keywords :	HDMI, CEC
Distribution :	TDA9950 , TDA9989 customers

Document History			
Version	Date	Author	Observations
1.0	07-September-2007	N. Mutz	Approved
1.1	02-October-2008	Cyril Bes	Explain TDA9989 CEC usage with FRO.
1.2	12-June-2009	F.Grandjacques	Add new API to send generic CEC messages
1.3	28-October-2009	Cyril Bes	Add a note about TDA9989 CEC data registers reading/writing

Table of contents

1. Introduction

1.1 Purpose

The purpose of this document is to describe the context of CEC protocol, CEC driver’s software architecture, functions available for customer and a demo software environment.

1.2 Glossary

API	Application Programme Interface
PMP	Project Management Plan (Overall)
SCP	Software Creation Process
SPMP	Software Project Management Plan
SRS	Software Requirements Specification
HW	Hardware
BSL	Board Support Library
Devlib	Device Library
HDMI	High Definition Multimedia Interface
CEC	Consumer Electoronics Control
STB	Set Top Box
CSP	CEC Stak Processor (CEC Interface)

1.3 References

- HDMI Specification 1.3a : Supplement 1 CEC
- CEC White Paper Reference
- HDMI CEC Stack Processor User manual

1.4 Open Points

- Complete CEC Messages

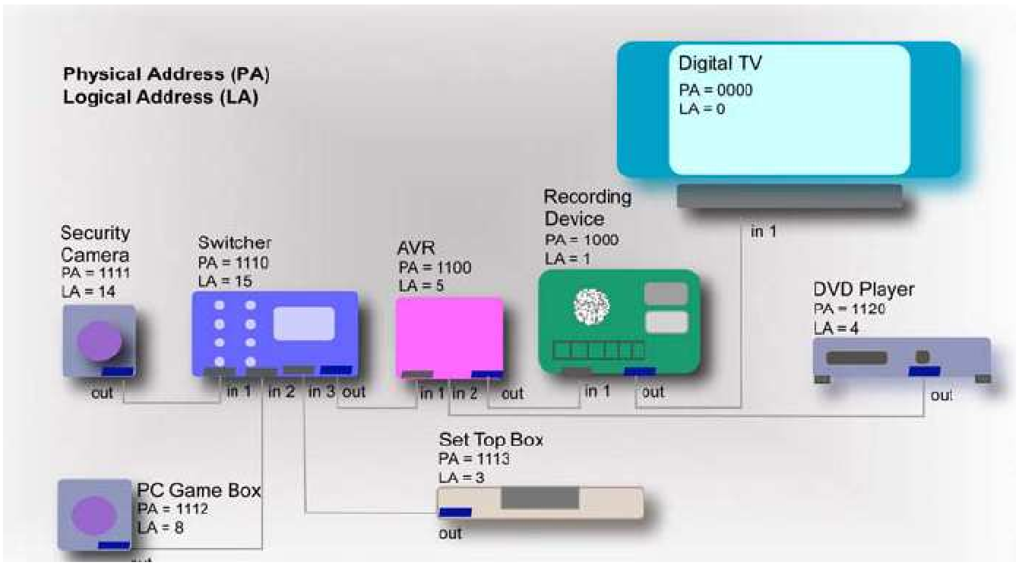
•

2. Overview

2.1 Context of CEC Communication

Consumer Electronics Control (CEC) is a high level protocol that enables control of devices.

CEC bus is a part of High Definition Multimedia Interface (HDMI). This bus supports CEC protocol.



This figure shows an exemple of HDMI CEC System. All devices are connected to each other via the HDMI connection. CEC bus, that is a part of HDMI reaches also all devices. Devices on sytem may be digital TV, DVD Player, Recording Device, Set Top Box, etc.

All those devices can communicate tho each other thanks the CEC bus.

Messages:

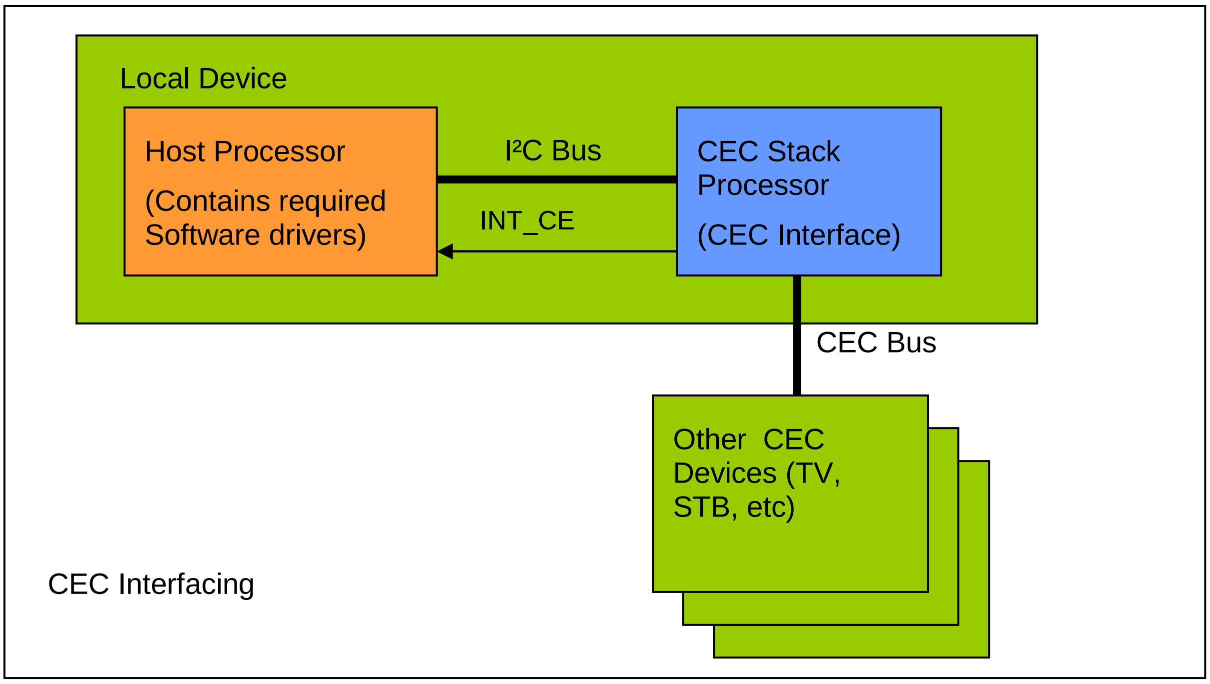
Messages exchanged on the CEC bus are used to perform control of devices. As an example, the message “Image View On” will order TV to display image on Screen. Message is the basic element of CEC protocol.

Features:

CEC protocol enables advanced control called “Feature”. Feature consists in perform sequences of message exchange in purpose to set a devices it the requirent context.

2.2 CEC Interfacing on a Device.

Devices that shall communicate via CEC Bus have to integrate a CEC Interface. In this document, CEC Interface is called CEC Stack Processor (CSP).



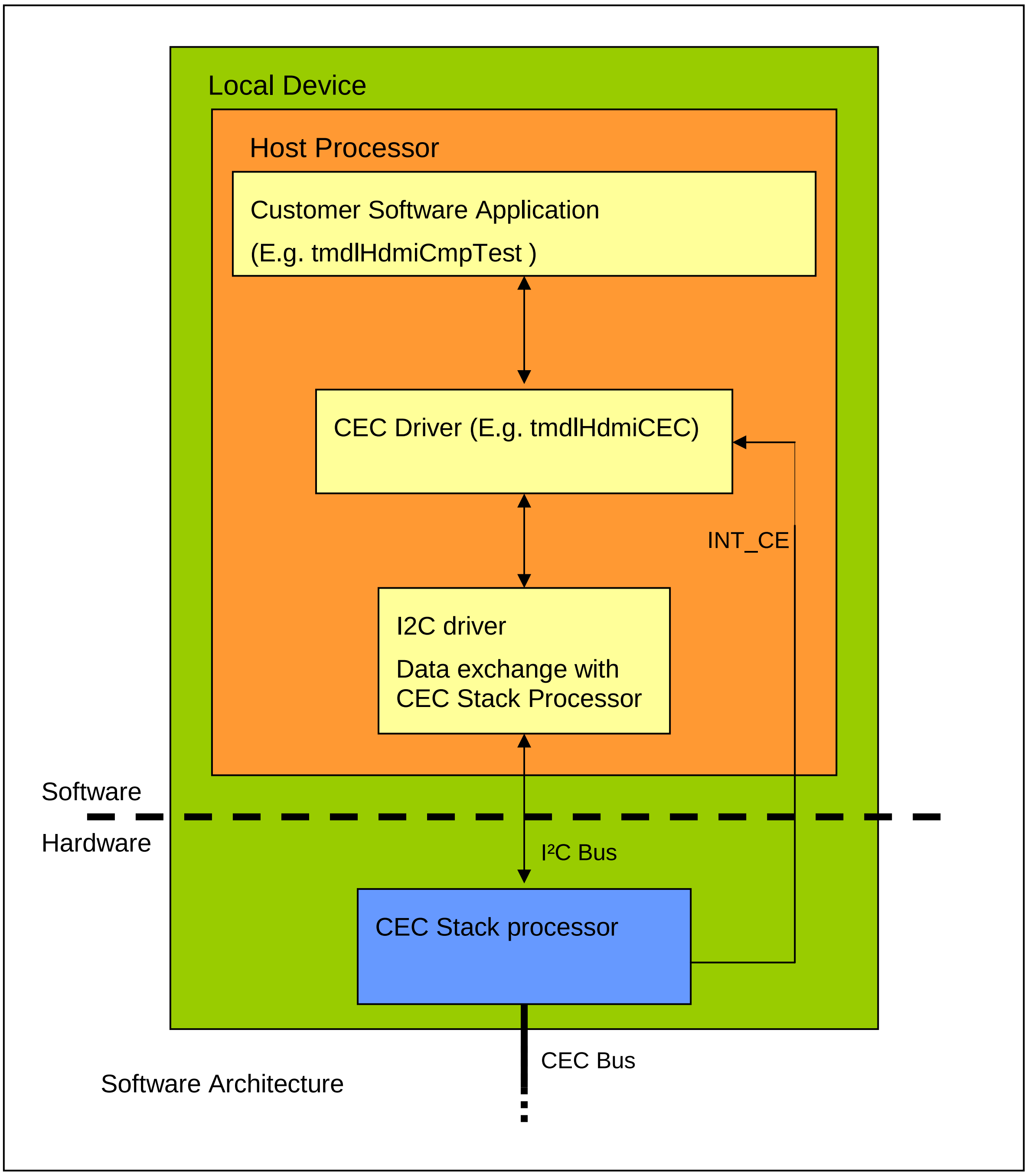
This figure shows interfacing of the local device to the CEC bus.

Local device consists in a CEC Stack Processor and a Host processor. CEC Stack processor is in charge of send CEC messages ordered by Host Processor and receive CEC messages from Other CEC devices. Host processor is the CEC interface user it is in charge of manage the CEC Protocol.

Data transfer between Host processor and CSP is performed via I2C Bus. CSP is always considered as a slave on I2C bus. To advert Host processor any event happends CSP emits a Interrupt signal on INT_CE.

2.3 CEC Software Architecture

This figure shows the general view of Software Architecture:



Software Architecture

Customer Software Application

Customer Software Application is the application that use the CEC Driver. In the current design, this application is located in the tmdlHdmiCECCompTest project .

This application can use all functions CEC Driver provides. Those functions enable opening, setup and closing of an Instance, sending CEC messages and receiving CEC message.

CEC Driver

It is in charge of CEC driver management. In the current design, this driver is located in tmdlHdmiCEC project.

This driver has to:

- Open, setup, close instance
- Build and Decode CEC messages

I2C driver

I2C bus is used to perform data transfer between TDA9950 and Host Processor. It is dependant of the customer application.

CEC Stack Processor

CEC Stack processor is in charge of manage communication on CEC bus. It gets or sends data via the I2C bus and it can send an Interrupt signal to Host Processor. An Interrupt signal is sent in the case of a CEC message reception or on an error occurrence.

3. CEC Driver Software Structure

This section explains CEC Driver Software Struture.

CEC Driver Software is composed in the following files in the table.

File Name	**Description**
TmdlHdmiCec.c	Contains all CEC Function the Customer Application is allowed to use. Those functions are described in the section 4. Basic Command Functions
TmdlHdmiCec_local.c	Contains Function that the Driver needs to acces CSP registers via I²C Bus
TmdlHdmiCec.h	TmdlHdmiCec.c header file
TmdlHdmiCec_local.h	TmdlHdmiCec_local.c header file
TmdlHdmiCec_Function.h	Describes prototypes of the CEC functions
TmdlHdmiCec_Type.h	Describes all type used by CEC Driver

4. Basic Command Functions

This section explains functions of Driver that can be used by the Customer Application.

It exists three types of Functions:

- Configuration Function (open, setup, and close instance and get other information information about device or software)
- Decode Function (Receive messages, decode message and provide it to Customer Application)
- Send Message Function (All functions that enable transmission of CEC message)

4.1 Configuration Function

Function Name	**Description**
tmdlHdmiCecGetSWVersion	Get the Software Version of the driver.
tmdlHdmiCecGetNumberOfUnits	Get the amount of available CEC devices in the system (connected to Host processor)
tmdlHdmiCecOpen	Open the first Instance
tmdlHdmiCecOpenM	Open all instance instead the first
tmdlHdmiCecClose	Close an instance
tmdlHdmiCecInstanceConfig	Configures an instance (Currently make nothing)
tmdlHdmiCecInstanceSetup	Setup the parameters of one instance

4.2 Decode Message Function

Function Name	**Description**
tmdlHdmiCecHandleInterrupt	This function is used to read data from CSP registers and decode received messages.
tmdlHdmiCecRegisterCallbacks	Indicates the Callback Function address

4.3 Send Message Function

Function Name	**Description**
tmdlHdmiCecAbortMessage	Send”Abort” Message. This message is reserved for testing purposes. Receiver has always to answer with a “FeatureAbort” messages.
tmdlHdmiCecActiveSource	Send”Active Source” to every devices on CEC bus. It indicates that is the current active source of data and provide its Physical Address
tmdlHdmiCecVersion	Send”CecVersion” message This message indicates Cec Version used by this device.

tmdlHdmiCecDeviceVendorID	Send"Device Vendor ID" message. This message indicates the Vendor ID of device.
tmdlHdmiCecFeatureAbort	Send "Feature Abort". Indicate that the received message of Feature generates problem and provides is the reason
tmdlHdmiCecGetCecVersion	Send"GetCecVersion" message. This message is a request to ask to a specific device its "Cec Version".
tmdlHdmiCecGetMenuLanguage	Send"GetMenuLanguage" message. This message is a request to ask to a specific device the "Menu Language" used.
tmdlHdmiCecGiveDevicePowerStatus	Send"GiveDevicePowerStatus" message. This message is a request to ask to a specific device its "Power Status"
tmdlHdmiCecGiveDeviceVendorID	Send"GiveDeviceVendorID" message. This message is a request to ask to a specific device its Vendor ID
tmdlHdmiCecGiveOsdName	Send"GiveOsdName" message. This message is a request to ask to a specific device its OSD Name
tmdlHdmiCecGivePhysicalAdress	Send"GivePhysicalAddress" message. This message is a request to qsk to a specific device its Physical Address.
tmdlHdmiCecImageViewOn	Send "ImageViewOn" message. Sent by a source device to Turn TV in the in "Image Display" state
tmdlHdmiCecMenuStatus	Send"MenuStatus" message. This message give to a specific device the menu status (Usualy the TV)
tmdlHdmiCecPollingMessage	Send"PollingMessage". This message enable the polling of a specific device. Receiver device has to acknoleadge at reception
tmdlHdmiCecReportPhysicalAddress	Send"ReportPhysicalAddress" message. This message is a response to "GivePysicalAddress" and reports the Physical Address of the asked Device.
tmdlHdmiCecReportPowerStatus	Send"ReportPowerStatus" This message is the response to "GiveDevicePowerStatus" and reports Power Status of the asked device.
tmdlHdmiCecRequestActiveSource	Send"RequestActiveSource" message. This message is sent to every device on Cec Bus. Its aim is to know which device is currently to source of data. Response expected is "Active Source" from the current active if there is one.
tmdlHdmiCecStandby	Send"Standby" message. This message switch one specific device or all devices in the Standby state.
tmdlHdmiCecTextViewOn	Send "TextViewOn" message. Sent by a source device to display menu and text on TV
tmdlHdmiCecSendMessage	Send any CEC messages on the Bus. The application is responsible of building a table with opcode and parameters.

- Function described is this section are available for customer

5. Basic Operations

This Section explains basic operation to do by Customer Application in purpose to handle CEC Driver.

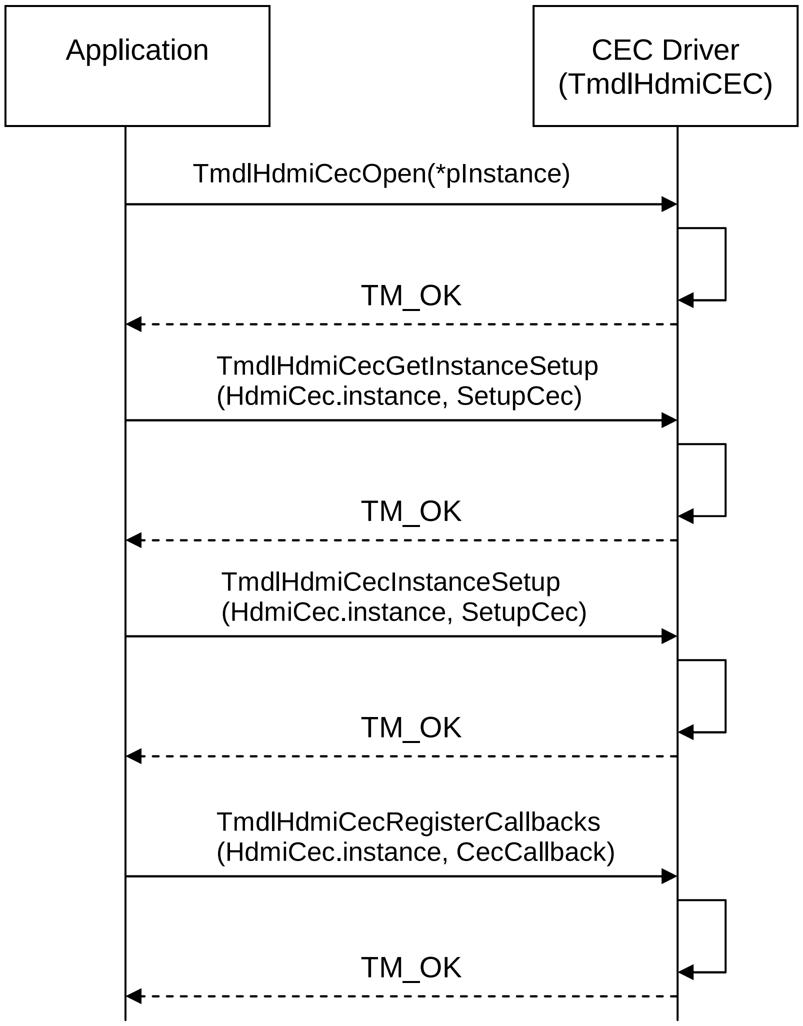
5.1 Instance Initialization

To use the CEC Driver, user has first to create and configure a CEC Driver instance.

CEC Driver Instance enables handling of one CEC Interface (CSP). Customer Application has to create as many Instances as the Host Processor has CEC Interfaces to manage. For example, if Host Processor has to manage 3 CSPs, then Customer Application has to create 3 CEC Driver Instances.

Actor interaction in case of receive CEC message

The following diagram shows interaction between Application and CEC Driver while the Instance initialization.


Application
CEC Driver (TmdlHdmiCEC)
TmdlHdmiCecOpen(*pInstance)
TM_OK
TmdlHdmiCecGetInstanceSetup (HdmiCec.instance, SetupCec)
TM_OK
TmdlHdmiCecInstanceSetup (HdmiCec.instance, SetupCec)
TM_OK
TmdlHdmiCecRegisterCallbacks (HdmiCec.instance, CecCallback)
TM_OK

Initialization happens in four steps.

Open CEC Instance

Get default parameters of Instance (Get Instance setup)

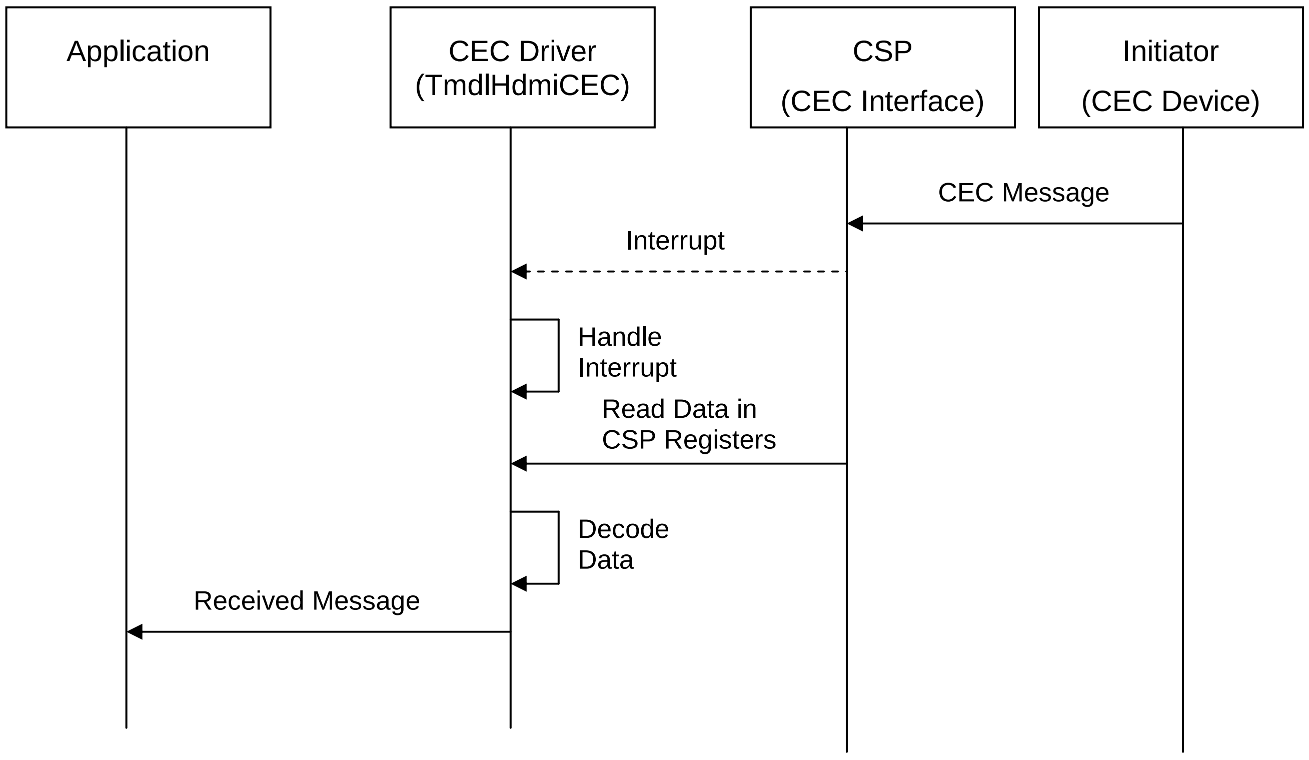
Setup parameters of Instance (Instance setup)

Setup the register callback address.

5.2 Receive CEC Message

Actor interaction in case of receive CEC message

This Diagram shows exchanges that occur between actors while a reception of message.



CSP receives a message from a CEC Device

CSP emits an Interrupt to the CEC Driver in purpose to relate message has arrived

CEC Driver triggers Handle Interrupt function to read data from CSP registers, decodes message and provides the received message to Application.

Application performs treatment according to received message.

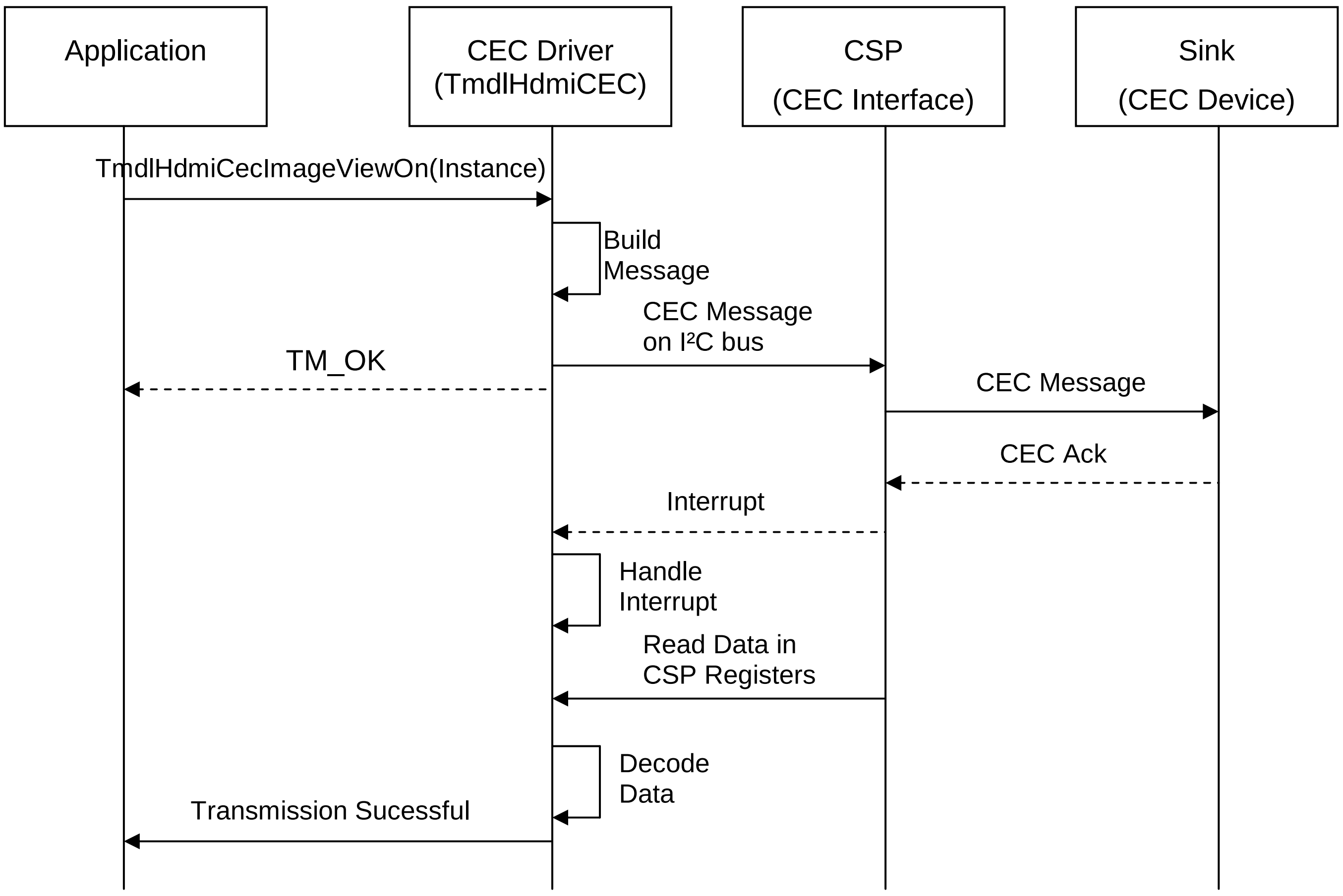
Customer Application operations in case of receive CEC message

When Application get message, its is allowed to treat it. Treatments depend of user requirements.

5.3 Send CEC Message

Actor interaction in case of send CEC message

This Sequence Diagram shows exchanges that occur between actors while a send of message.



Application Orders send of message to the Driver.

Drive builds message and send it to CSP via I²C bus

CSP translates this message and send it on CEC bus. Sink Acknowledge CEC message reception.

CSP emit an Interrupt to CEC Driver in purpose to relate transfer success.

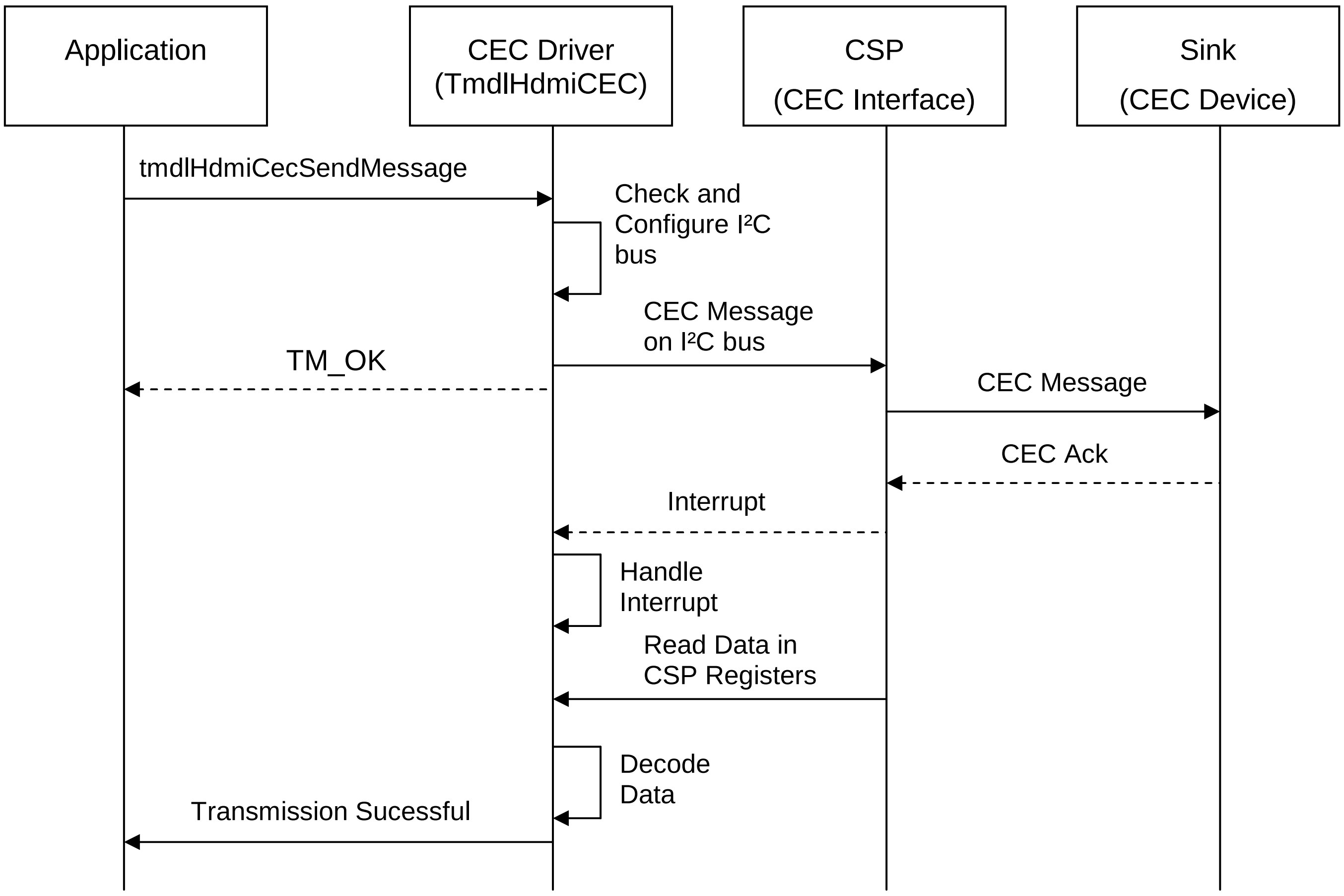
CEC Driver triggers Handle Interrupt Function to read data from CSP registers, decodes message and adverts Application of the transmission Success.

Application can continue

5.4 Send Generic CEC Message

Actor interaction in case of send CEC message

This Sequence Diagram shows exchanges that occur between actors while a send of message.



Application build messages (Opcode and parameters) and transfer the message to the Driver.

The buffer of data build by the application and send to the driver should contain Opcode first and then the parameters.

Example : unsigned char Data[0] = ReceiverLogicalAddress;

unsigned char Data[1] = CEC_OPCODE_CEC_VERSION ;

unsigned char Data[2] = CECVersion;

Driver check and configure harware to send it to CSP via I²C bus

CSP translates this message and send it on CEC bus. Sink Acknowledge CEC message reception.

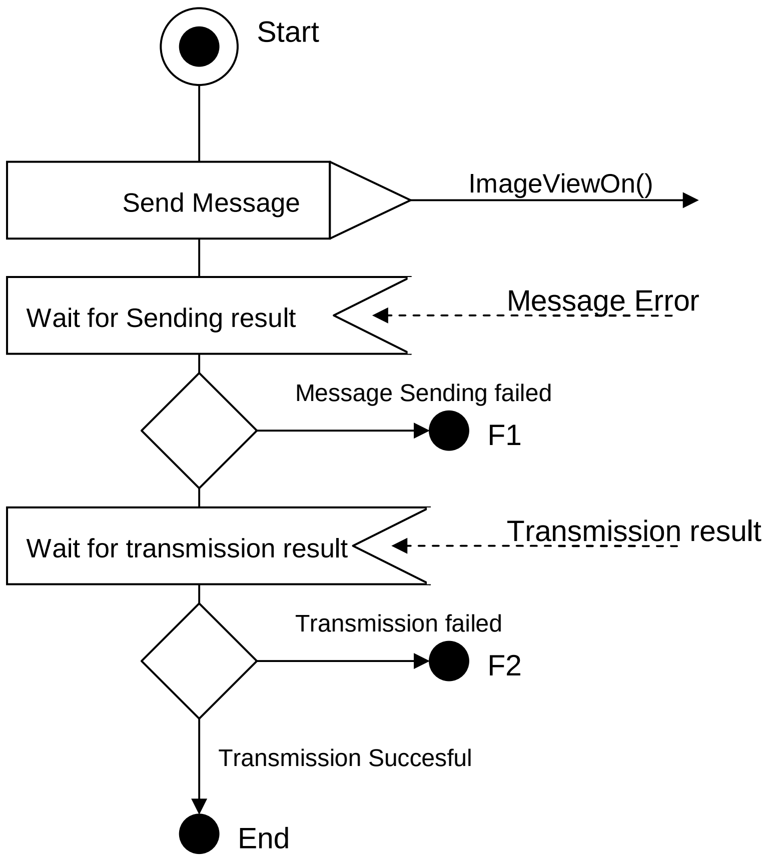
CSP emit an Interrupt to CEC Driver in purpose to relate transfer success.

CEC Driver triggers Handle Interrupt Function to read data from CSP registers, decodes message and adverts Application of the transmission Success.

Application can continue

Customer Application operations in case of send CEC message

This State diagram explains what are operations Customer Application has to do in order to send a message.



Application starts with trigger a send message function of CEC Driver (ImageViewOn)

First it has to wait for the message Sending result. This result corresponds to the function return.

If Message sending failed, then Application has to leave sending of message operation (F1)

Else Application has to wait for the transmission result.

If transmission result is transmission failed, then Application has to leave sending of message operation (F2)

Else, CEC message sending Succeed.

6. Diversity

6.1 TDA9989/TDA19989 CEC Data registers read/write accesses

CDR	CEC Data Registers	07h - 19h	R/W

Messages in the CEC Data Registers must be written and read as contiguous ranges of registers.

6.1.1 Writing the CEC data registers

- The CEC Data Registers should be written starting from the first CEC Data Register, for the number of registers indicated by the contents of that first CEC Data Register, in one contiguous operation.
- The length of the message is given by the byte in the first CEC Data Register. This will be at least 3 for the shortest message. A value less than 3 indicates an invalid message.
- If fewer CEC Data Registers are written than the number indicated by the first CEC Data Register, the partial message will be ignored and no confirmation will be returned.
- If more CEC Data Registers are written than the number indicated by the first CEC Data Register, the message will be processed as soon as the message's last CEC Data Register is written, and the extra bytes written will be ignored.
- If the highest CEC Data Register is written and more message bytes are indicated by the first CEC Data Register, the message will be processed as soon as the highest CEC Data Register is written, and the extra bytes written will be ignored.

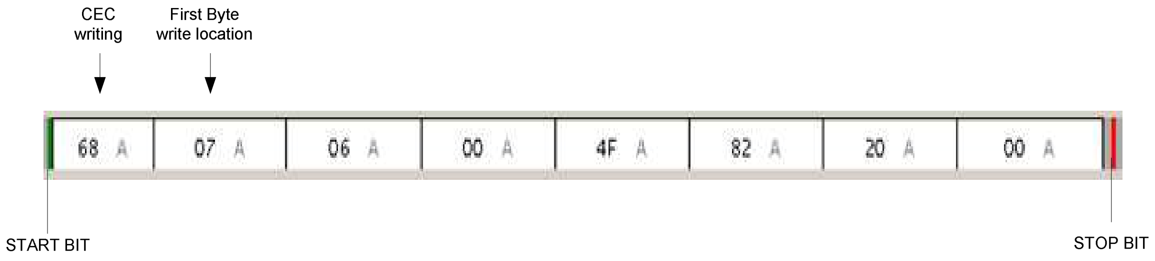


Figure 1 CEC Data register writing example

6.1.2 Reading the CEC data registers

- The CEC Data Registers only contain a valid message when the INT line is set and the INT bit in the Status Register is set.
- If CEC Data Registers are read when the INT line is not set, the first CEC Data Register will contain 0, indicating that there are no bytes to read. Any further reads before a STOP condition will return the value 00h.
- The CEC Data Registers should be read starting from the first CEC Data Register, for the number of registers indicated by that first CEC Data Register, in one contiguous operation.

- If the host writes CEC Data Registers and then begins reading without first resetting the Address Pointer Register, reading will automatically commence from the first CEC Data Register.
- If reading stops before all indicated CEC Data Registers are read, the INT line is reset and the message is discarded, it will not be available for reading again.
- If reading continues for more CEC Data Registers than are indicated by the first CEC Data Register, the value 00h will be read. The INT line is reset when the last valid CEC Data Register for the current message is read.

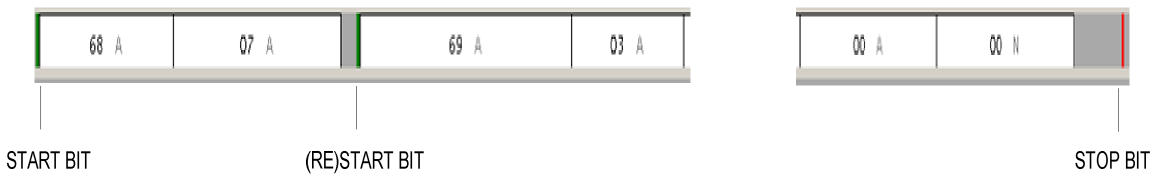


Figure 2 CEC Data register reading example

6.2 TDA9989/TDA19989 diversity

CEC driver TDA9989/TDA19989 diversity is handled by preprocessor flag **TMFL_TDA9989**. This flag shall be defined at compilation time in order to use CEC driver for TDA9989/TDA19989 IC.

6.3 TDA9989/TDA19989 CEC clock modes

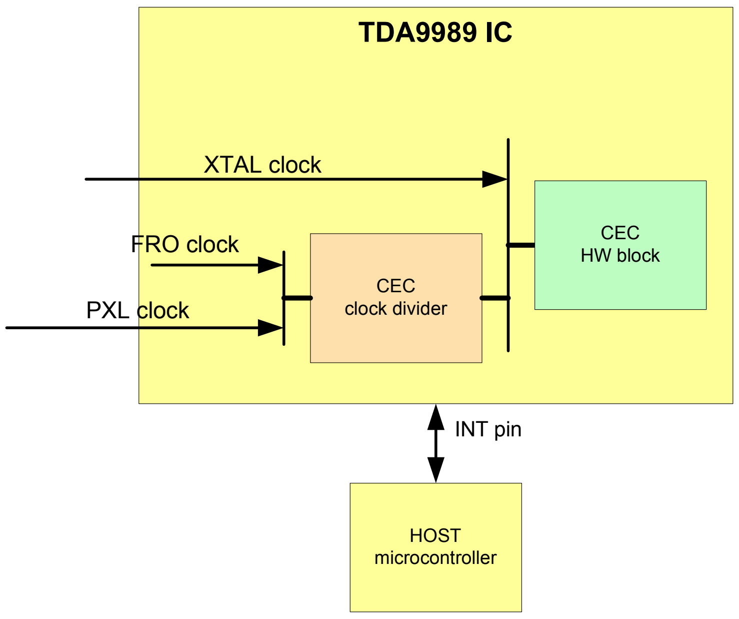


Figure 3 TDA9989 CEC clock modes

TDA9989 CEC hardware block may be clocked by one of the following clock sources:

- an external crystal oscillator
- the external video pixel clock
- an internal free running clock (FRO)

FRO and PXL clocking schemes require a so called "CEC calibration process". The aim of this process is to determine the incoming clock frequency and CEC clock divider circuit settings in order to feed CEC HW block with a 12MHz input clock.

The remaining of this chapter explains what has to be done when application wants to use the internal free running clock (FRO) as CEC clock source.

- The CEC driver initialization sequence shall be done as depicted below:

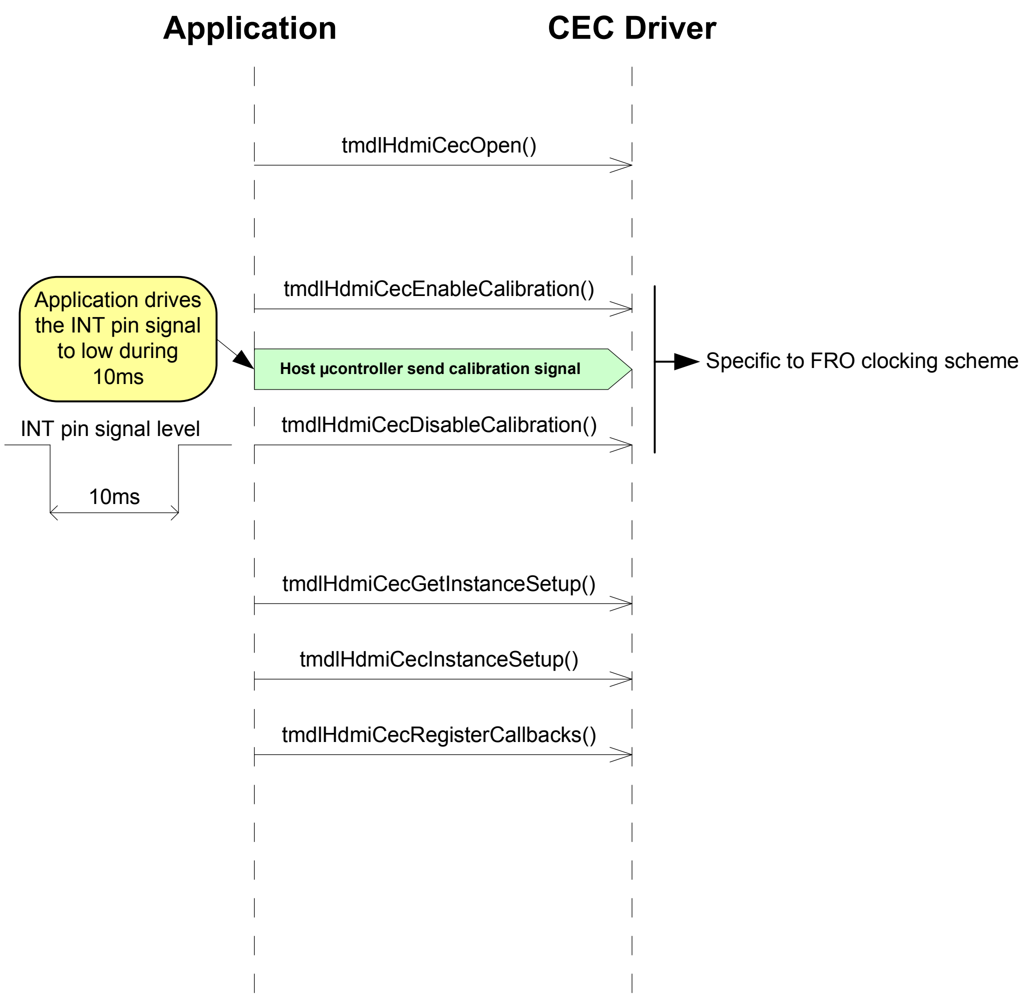

Application
CEC Driver
tmdlHdmiCecOpen()
tmdlHdmiCecEnableCalibration()
Application drives the INT pin signal to low during 10ms
Host µcontroller send calibration signal
Specific to FRO clocking scheme
INT pin signal level
tmdlHdmiCecDisableCalibration()
10ms
tmdlHdmiCecGetInstanceSetup()
tmdlHdmiCecInstanceSetup()
tmdlHdmiCecRegisterCallbacks()

Figure 4 CEC TDA9989 driver init sequence

In order to send the "calibration signal" the microcontroller has to drive low the TDA9989 INT pin for **exactly 10 ms**. Thanks to this time reference, CEC clock divider is able to deduce incoming signal frequency.

- Each time the TDA9989 is set in standby mode the FRO will be cut for power saving purposes. When powering up, Application has to perform a new calibration procedure before using CEC. Here is the corresponding API sequence :

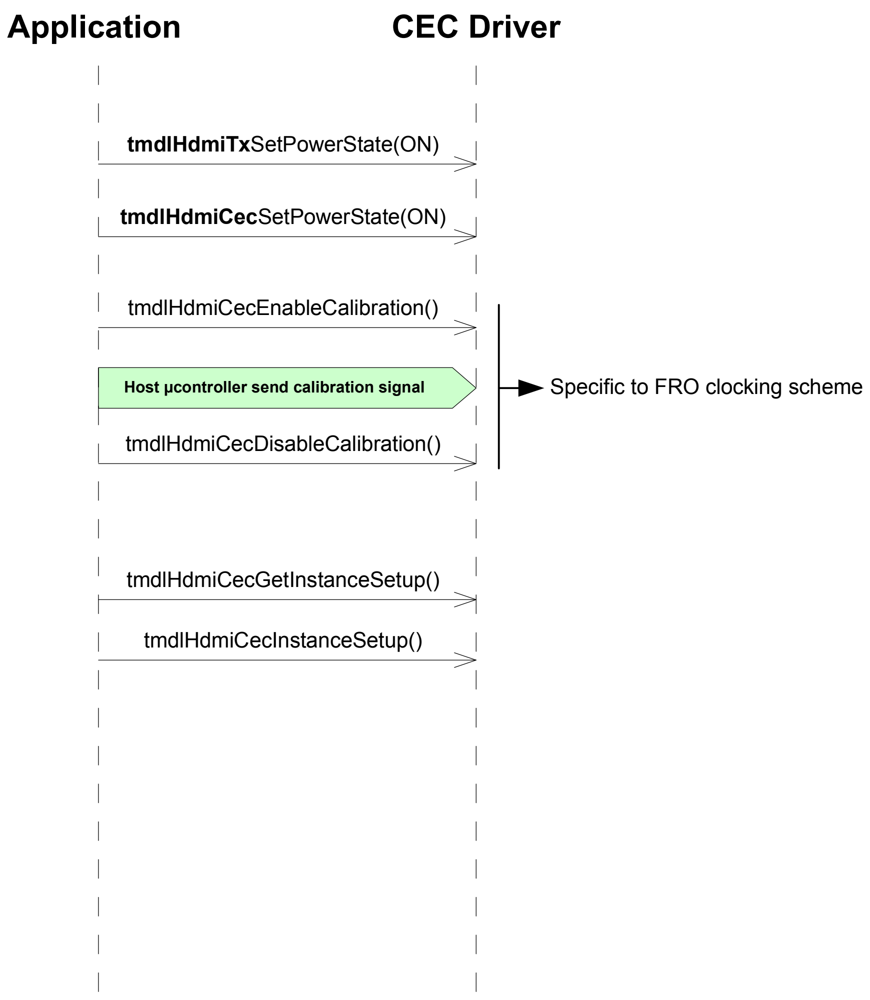

Application
CEC Driver
tmdlHdmiTxSetPowerState(ON)
tmdlHdmiCecSetPowerState(ON)
tmdlHdmiCecEnableCalibration()
Host µcontroller send calibration signal
Specific to FRO clocking scheme
tmdlHdmiCecDisableCalibration()
tmdlHdmiCecGetInstanceSetup()
tmdlHdmiCecInstanceSetup()

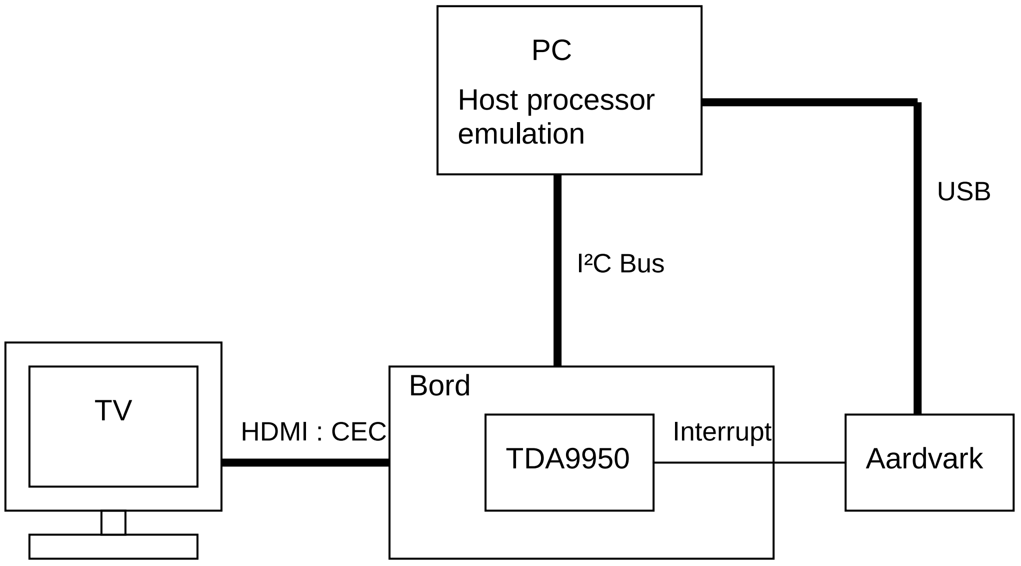
Figure 5 CEC TDA9989 recovering from STANDBY sequence

7. Test

This section explains an example of test to make in purpose to validate CEC Driver.

7.1 Test environment

This figure shows the test enviromenent currently used.



System is composed of:

- A PC: PC emulates the Host processor. Software running on PC is the Customer Application and the CEC Driver.
- A Board: Board performs connection between PC and CEC bus. It contains the TDA9950 (CSP) that is the CEC interface.
- A TV: TV is the CEC device used for test
- Aardvark: This device is used to notice interrupt event

7.2 Test File Sources

In purpose to test CEC Driver, a Test Application is implemented as Customer Application.

This Test Application is composed of the following files.

File Name	**Description**
main.c	This file contains the test application.
menu.c	This file provides handling of menu. Menu interacts with test application to
main.h	main.c header file
menu.h	menu.c header file
Infr_i2c.h	Describes structure and functions of I²C Driver
interfaceExternal.h	Describes function interfacing with Application (CSP and user)
tmdlHdmiCEC.h	Describe all types and functions of CEC Driver
TmdlHdmiCEC_cfg.h	Describes structure and functions for Instance management
TmdlHdmiCEC_local.h	Describes structure and functions of an Instance