MC-ISAR_TC3xx_DemoApp.docx

About this document

Scope and purpose

This Application Note provides details for building and running the Demo Application (DemoApp).

The DemoApp enables users to perform preliminary test on the delivered MCAL driver.

Intended audience

This document is intended for anyone, who needs to perform a quick test of the product delivery.

Table of contents

1 Getting started

The following steps must be performed before building/running the DemoApp:

1. Install the MCAL package and ensure the tool versions required are also installed as per the Release Notes.
2. If the EB tresos Studio GUI is operational, close the GUI before proceeding to the next step.
3. Copy all the plugins provided into the EB tresos plugin folder. The installed package plugin folder can be located at the following path: `<Installed-Package>\McIsar\PluginsTresos\eclipse\plugins`.
4. Delete the tresos link file if it is present in `tresos\tresos\links` path. Do take a backup of the file before deleting it from the said path otherwise the DemoApp build will halt if the file is present at this location.

   The configured XDMs are applicable to:
   1. TC389 which is in the TC38A folder.
   2. TC399 which is in the TC39B folder.
   3. TC377_ED which is in the TC37A_ED folder
   4.TC377 which is in the TC37A folder
   5.TC357_ADAS which is in the TC35A folder
   6. TC367 which is in the TC36A folder
   7. TC337 which is in the TC33A folder
   8. TC337_ED_ADAS which is in the TC33A_ED folder
   9. TC3E7 which is in the TC3EA folder.
   10. TC32A which is in the TC32A folder

Note: The XDMs must be configured according to the configuration parameters provided in the `TC3xx_SW_MCAL_DemoConfigs.xls` document, which is common for theTC3Ex, TC33x, TC36x, TC38x, TC37x, TC35x and TC39x devices. In addition, this document also provides additional information regarding the resource requirements, for example, interrupt priority, Port pin usage, DEM usage and GTM usage for the DemoApp.

2 Building the DemoApp

The DemoApp build is launched by calling the `DemoAppBuild.bat` present in the `<Installed Package>\DemoWorkspace\McalDemo` folder. The application (DemoApp files in this case), source and configuration files are compiled and linked together. The output files, `*.elf/*.hex`, are generated after completion of the build process in the following folder location:
`<InstalledPackage>\DemoWorkspace\McalDemo\<TC33A/TC32A/TC33A_ED/TC36A/TC38A/TC3EA /TC39B/TC37A_ED/TC37A/TC35A>\2_Out\<Tricore_Tasking/Tricore_Gnuc/Tricore_Dcc/Tricore_ Ghs >`

The output `DemoApp_Node0.elf/DemoApp_Node0.hex` and `DemoApp_Node1.elf/DemoApp_Node1.hex` file are generated if the FR/HSSL module is selected for installation.

- DemoApp execution of all modules except for FR /HSSL use `DemoApp_Node0.elf/DemoApp_Node0.hex`
- DemoApp execution of FR/HSSL module use `DemoApp_Node1.elf/DemoApp_Node1.hex` along with `DemoApp_Node0.elf/DemoApp_Node0.hex`

The `DemoApp.elf/DemoApp.hex` output file is generated if the FR and HSSL module are not selected for installation.
The output files are generated in the following folder location:
`<Installed-Package>\DemoWorkspace\McalDemo\< TC33A/TC32A/TC33A_ED/TC36A/TC38A/TC3EA /TC39B/TC37A_ED/TC37A/TC35A>\2_Out\<Tricore_Tasking/Tricore_Gnuc/Tricore_Dcc/Tricore_ Ghs>`

Note: All packages (BASIC, COM-E, CD and DEMO) with all the provided drivers should be installed in the same directory in order to build the DemoApp provided for the MCAL drivers. However, it is possible to install individual packages in a separate directory for use, but the DemoApp cannot be built individually/selectively.

Note: In case of selective installation (for example, only ETH needs to be is installed from COM-E package), remove the modules and their respective `.xdm` files from the EB tresos workspace. Also remove the driver source code, demo code, Irq and integration files.

The steps to build the DemoApp executable are as follows:

1. Open the command prompt in the `<Installed Package>\DemoWorkspace\McalDemo\< TC33A/ TC32A/TC33A_ED/TC36A/TC38A/TC3EA/TC39B/TC37A_ED/TC37A/TC35A>` folder.

2. Call DemoAppBuild.bat file with the below mentioned input arguments:

   First input argument (Compiler Selection):
   GNU: Build with GnuC compiler
   DCC: Build the Windriver compiler
   TASKING: Build with Tasking compiler
   GHS: Build with GreenHills compiler
   If NO value is provided among the above four values DemoApp build with Tasking as Default compiler

   Second input argument (Enable/Disable Tresos Config Generation Selection):
   WITHOUT_TRESOS: Build without Tresos Configuration generation

Building the DemoApp

WITHOUT_TRESOS_WITHOUT_FR: Build without Tresos Configuration generation and without the second build needed by FR and HSSL Module demoapp

WITH_TRESOS: Build with Tresos Configuration generation enabled

If NO value is provided among the above three values DemoApp build with Tresos Configuration generation as Default option

Third input argument (Tresos Tool Path Selection):

User Provided Tresos Tool Path: Build with User provided Tresos Tool Path

If NO value is provided then DemoApp build uses %TRESOS_26_2_0_HOME%/tresos/bin as the Tresos Tool path. **Enviroment variable TRESOS_26_2_0_HOME** value is used to get the tool installation directory.

Note: Use Unix slash while providing tool path

Fourth input argument (Compiler Path Selection):

User Provided Compiler Path: Build with User provided Compiler Path

If NO value is provided then DemoApp build uses compiler binary file path from "C:/sofit/aurix2g_sw_mcal/<Compiler>/…/…" (Compiler binary file directory).

Note 1: Use Unix slash while providing tool path

Note 2: Even if only fourth parameter to be modified, the user has to pass first three parameters compulsorily.

Example: Code listing for running the DemoApp

`DemoAppBuild.bat`
`DemoAppBuild.bat` **`GNU`**
`DemoAppBuild.bat DCC` **`WITHOUT_TRESOS`**
`DemoAppBuild.bat GHS WITH_TRESOS` **`D:/xyz/Tools/tresos/bin`**
`DemoAppBuild.bat TASKING WITH_TRESOS` **`D:/xyz/Tools/tresos/bin E:/xyz/Compilers_Installation/hightec/4.9.4.0/hightec/toolchains/tricore/v4.9.4.0/bin`**

3. The batch file follows two steps: generation of Configuration files and compilation/linking of files and finally generation of the executable file along the given path `<Installed-Package>\DemoWorkspace\McalDemo\< TC33A/ TC32A/TC33A_ED/TC36A/TC38A/TC3EA /TC39B/TC37A_ED/TC37A/TC35A>\2_Out\< Tricore_Tasking/Gnuc/Dcc/Ghs>`.

   Note:It is recommended to rebuild the DemoApp by calling DemoAppBuild.bat program in case if any pop up occurs during build like "sh.exe stopped working"

2.1 Command to merge multiple configured xdm to single epc file

An example command to merge multiple configured `XDM` files to generate an `EPC` file is as follows:

Example: Merging the xdm to generate EPC

```
C:\<tresos-path>\bin\tresos_cmd.bat  -DMapOptionalAsList=false
-Dtarget=AURIX2G -Dderivate=TC389/87 legacy convert Port.xdm Pwm.xdm
Spi.xdm Adc.xdm Dem.xdm Dio.xdm EcuM.xdm Gpt.xdm Icu.xdm Irq.xdm Mcu.xdm
ResourceM.xdm Combined.epc@asc:4.4.0
```

Note: `-Derivate` should be changed as per the desired derivate to run the DemoApp.

2.2 Command to generate code from epc file

An example command to generate code from the `EPC` file is as follows:

Example: Command to generate code from EPC file

```
C:\<tresos-path>\tresos\bin\tresos_cmd.bat -Dtarget=AURIX2G
-Dderivate=TC389/87 legacy generate Combined.epc@asc:4.4.0 -o outputdir -g
Irq_Aurix2GAS440 -g Mcu_Aurix2GAS440 -g Port_Aurix2GAS440 -g
Dem_Aurix2GAS440 -g EcuM_Aurix2GAS440 -g Adc_Aurix2GAS440 -g
Dio_Aurix2GAS440 -g Gpt_Aurix2GAS440 -g Icu_17_TimerIp_Aurix2G440 -g
Pwm_17_GtmCcu6_Aurix2G440 -g Spi_Aurix2GAS440 -g ResourceM_Aurix2GAS440
```

2.3 Utilities used for building the DemoApp

The following utilities are used for the build process:

Table 1 Utilities used for the build process

Description	Path/Where to find it
Make file used for building the application, the file contains rules for generating the executable	<Installed-Package>\DemoWorkspace\McalDemo\< TC33A/ TC32A/TC33A_ED/TC36A/TC38A/ TC3EA/TC39B/TC37A_ED/TC37A/TC35A>\1_ToolEnv\0_Build

Note: Any errors in compilation and linking will be displayed in the DOS prompt.

2.4 DemoApp resource usage

For more information, refer to the `TC3xx_SW_MCAL_DemoConfigs.xls` file.

2.5 Integrating DEM module

Non-productive DEM is delivered along with the MCAL package. User is expected to replace the non-productive DEM with the productive DEM module. If the non-productive DEM module is used, any new event has to be added manually in the generated `Dem_cfg.h` file available in following path of the installed files:

`DemoWorkspace\McalDemo\<TC33A/TC33A_ED/TC36A/TC38A/TC3EA/TC39B/TC37A_ED/TC37A/TC35A>\0_Src\BaseSw\Infra\Autosar_Srv\Dem_cfg.h`

The following code listing provides an example for adding a new event:

Example: Adding DEM Event in Dem_cfg.h file

```
#define DemConf_DemEventParameter_FLS_E_ERASE_FAILED            (1U)
```

3 Running the DemoApp

Before running the DemoApp, flash the DemoApp executable to the board. Start the terminal application (for example, HyperTerminal) in PC with the following COM properties.

Table 2 Terminal properties

Property	Value
Baud rate	115200
Number of data bits	8
Parity	None
Number of stop bits	1
Flow control	None

Run the program on target and the following menu displays in the terminal window.

DemoApp main menu

```
< >: ------------- MAIN Menu -------------
<0>: Go to MCAL DemoApp List
<1>: Goto MCAL Multicore-DemoApp List
```

NOTE:For 337PD only option 0 is available since it is single core device

On pressing 0 in main menu, the following demo menu displays. If a particular driver demo is not available, the option for that driver will not be visible.

DemoApp sub-menu

```
/*************************************************************************/
 < >: ............ AUTOSAR DRIVERS..............
 <a>:   Adc: Demo
 <b>:   Can_17_McmCan: Demo
 <c>:   Dio: Demo
 <d>:   Dma: Demo
 <e>    Eth_17_GEthmac: Demo
 <f>:   Fee: Demo
 <g>:   Fls_17_Dmu: Demo
 <h>    Fr_17_Eray: Demo
 <i>:   Gpt: Demo
 <j>:   Icu_17_TimerIp: Demo
 <k>:   Mcu: Demo
 <l>:   Pwm_17_GtmCcu6Ccu6: Demo
 <m>:   Spi: Demo
 <n>:   Wdg_17_Scu: Demo
 <o>:   Crc: Demo
```

Running the DemoApp

```
 <p>:   FlsLoader: Demo
 <q>:   Smu: Demo
 <r>:   I2c: Demo
 <s>:   Lin_17_AscLin: Demo
 <t>:   Uart: Demo
 <u>:   Bfx: Demo
 <v>:   Stm: Demo
 <w>:   Ocu: Demo
 <x>:   Dsadc: Demo
 <y>:   CanTrcv_17_V9251: Demo
 <z>:   CanTrcv_17_W9255: Demo
 <1>:   Sent: Demo
 <2>:   Hssl: Demo
 <3>:   Iom: Demo
 <.>:   Go to Main Menu
/*****************************************************************/
```

On pressing 1 in main menu, the following demo menu displays

```
/*****************************************************************/
 < >: ............ AUTOSAR DRIVERS..............
 <a>:   Gpt: Demo Multicore
 <b>:   Wdg_17_Scu: Demo Multicore
<.>:   Go to Main Menu
/*****************************************************************/
```

3.1 Option 0:MainMenu

3.1.1 Option a: Adc Demo

The Adc demo converts Adc channels AN4 (Rest of the Devices)/AN0 (for TC35A) and AN5 (Rest of the Devices)/AN1(for TC35A)

With 12-bit resolution. G0CH4 and G0CH5 channels are used for all device except TC35A, G0CH1 and G0CH0 channels are used for all device TC35A

Table 3 External connections for Adc demo

Signal name	Connections to be made
AN4(Rest of the Devices)/AN0(for TC35A)	Connect the potentiometer/variable voltage source
AN5(Rest of the Devices)/AN1(for TC35A)	Connect the potentiometer/variable voltage source

Table 4 Adc DemoApp menu

Option	Description of the demo
<1>	For ADC SW Group Demo The conversion results should be the Voltage supplied to the Pin's AN4/AN0 and AN5/AN1
<2>	For ADC HW Group Demo The conversion results should be the Voltage supplied to the Pin's AN4/AN0 and AN5/AN1
<x>	Back to main menu

3.1.2 Option b: Can_17_McmCan Demo

The DemoApp uses the CAN controllers 0 and 1 of MCMCAN. These two CAN nodes communicate through (CAN00(X301)) and (CAN10/ (TC337_ED) CAN03) (X302) of the TriBoard. The following connections have to be made externally to run the demo successfully.

Note:
TC337_ED device: Remove Zero Ohm Resistor R311 and R312 from Triboard. Solder two connecting wires to Tx and Rx pin of CanTrcv 9251 and connect their other end to Pin 13 and Pin 15 of X2 column of T2LA connector placed on X802(Peripheral connector) respectively.
TC337_PD device: Remove Zero Ohm Resistor R311 and R312 from Triboard and place it at R313 and R314 resistor location on the Triboard.

Table 5 External connections for Can_17_McmCan Demo

Signal name	Connections to be made
CANL	(pin 3 of X301) <-> (pin 3 of X302)
CANH	(pin 4 of X301) <-> (pin 4 of X302)

Multiple CAN demo options are supported. The following table provides details of the CAN driver DemoApp menu.

Table 6 Can_17_McmCan DemoApp menu

Option	Description of the demo
<1>	Transfer of Standard Frame between CAN controllers: **Operational Specific:** Sets the mode of the CAN controller 0 and 1 to STARTED, transmits two STANDARD ID frames each from controller 0 to 1 and 1 to 0, sets the mode of CAN controller 0 and 1 to STOPPED.
<2>	Transfer of Extended Frame between CAN controllers: **Operational Specific:** Sets the mode of the CAN controller 0 and 1 to STARTED, transmits one EXTENDED ID frame each from controller 0 to 1 and 1 to 0, sets the mode of CAN controller 0 and 1 to STOPPED.
<3>	Mixed Mode Support: **Operational Specific:** Sets the mode of the CAN controller 0 and 1 to STARTED, transmits one STANDARD ID frame and 1 EXTENDED ID frame each from controller 0 to 1 and 1 to 0 (both Txn and Rxn are using MIXED Mode Hardware Object), sets the mode of CAN controller 0 and 1 to STOPPED.
<4>	Test enabling and disabling of Tx and Rx interrupts: **Operational Specific:** Demonstrates the behavior of the Can_17_McmCan_DisableControllerInterrupts and Can_17_McmCan_EnableControllerInterrupts APIs.

Option	Description of the demo
<5>	Changing Baudrate using Can_SetBaudRate: **Operational Specific:** Demonstrates the usage of Can_17_McmCan_SetBaudrate API
<6>	Activation and Deactivation of Pretended networking: **Operational Specific:** Demonstrates the usage of Can_17_McmCan_SetIcomConfiguration API for Activation and Deactivation of pretended networking.
<7>	Trigger transmit functionality: **Operational Specific:** Demonstrates the usage of Can_17_McmCan_Write when Trigger transmits functionality is enabled.
<8>	FD frames transmission and reception: **Operational Specific:** Sets the mode of CAN controller 0 and 1 to STARTED, transmits two FD frames each from controller 0 to 1 and 1 to 0, sets the mode of CAN controller 0 and 1 to STOPPED.
<x>	Back to main menu

3.1.3 Option c: Dio Demo

The following table provides details of the Dio driver DemoApp menu.

Table 7 Dio DemoApp menu

Option	Description of the demo
<1>	Set one LED port pin ON -Glows the second LED on Triboard
<2>	Set one LED port pin OFF -Switches Off the second LED on Triboard
<3>	Set the LED Series group ON -Glows the second and third LEDs on Triboard
<4>	Set the LED Series group OFF -Switches Off the second and third LEDs on Triboard
<x>	Back to main menu

3.1.4 Option d: Dma Demo

The Dma demo transfers data form the source address to the destination address. On selecting the option d, software transaction from the source memory to the destination memory for a given number of transfer count is triggered. The nature of transaction is dependent on various configuration parameters (available in the EB tresos). DMA channel 16 is used for data transfer.

The Source value is value present at the source address and the Original Destination value is the value present at the destination address before the transaction. After successful execution of demo, the Final Destination value will display the value present at the destination address as “DMA DEMO PASS”.

3.1.5 Option e: Eth_17_GEthMac Demo

The Eth_17_GEthMac demo requires the following to be done before the demo is run:

- Connect the Triboard to a PC/Laptop using the Ethernet cable
- Assign a static address to the PC Ethernet interface where the Triboard is connected

Running the DemoApp

Multiple Ethernet demo options are supported. The following table provides details of the Ethernet driver DemoApp menu.

Note: For AS440, Eth_17_GEthmac Demo is available as Eth_17_GEthmacV2 Demo

Table 8 Eth_17_EthMac DemoApp menu

Option	Description of the demo
<1>	Promiscuous Mode: Receive All Frames - Receives All Frames Transmitted on Ethernet Bus irrespective of the Destination Address in ETH Frame
<2>	PING Application - Asks User to Enter the IP Address to Ping and then transmits an ARP Request to the IP Address Entered and Waits for an ARP Reply from the PC and prints MAC Address of the PC having that IP Address
<3>	Transmit ETH Frame of User Defined Payload Length - Asks User to Enter the PayLoad Length(42...1500) of ETH Frame and then transmits an Ethernet Frame of Frame Type(0xABCD) of User defined Payload length(Total Frame Length = Payload Length +18)
<4>	Read PHY Registers - Read Ethernet Transceiver(PHY) registers Values(CTRL,STAT,PHYCTL1,PHYCTL2,MIICTRL)
<x>	Back to main menu

3.1.6 Option f: Fee Demo

For the Fee demo, two NVM blocks and one QS block (Blk 3) have been configured. The Fee demo will erase the entire DFlash (that is, DF_EEPROM) area and then the `Fee_Init` API will be called. This is done only for the first time when FEE demo is called after a power-on reset. Each time the FEE demo is called, the following steps will be run sequentially and automatically.

Sequence 1: Block 1 is written
Sequence 2: Block 2 is written
Sequence 3: Block 1 is written again with different data
Sequence 4: Block 2 is written again with different data
Sequence 5: Block 1 is read; Latest data written to Block 1 (that is, the data written in Step 3) is read and compared
Sequence 6: Block 2 is read; Latest data written to Block 2 (that is, the data written in Step 4) is read and compared
Sequence 7: Block 3 is written
Sequence 8: Block 3 is read; data written to Block 3 is read and compared.
Sequence 9: Block 3 is erased and block state is checked if it is set to ‘erased’.

After successful execution of above sequences, Fee demo results are displayed

3.1.7 Option g: Fls_17_Dmu Demo

The Fls_17_Dmu demo writes the data into the DFlash and then reads back from the Dflash. The read data is compared with the data written for verification.

Sequence 1: All sectors of data Flash are erased and verified for erase success.
Sequence 2: Data is written into the DFlash.

Sequence 3: Written data is compared with 'margin 0' directly from DFlash.
Sequence 4: Written data is read into local buffer and compared.
Sequence 5: Cancel operation is verified by cancelling ongoing Read, Write and Erase operations.

After successful execution of above sequences, Fls_17_Dmu demo results are displayed.

3.1.8 Option h: Fr_17_Eray Demo

Table 9 External connections for Fr_17_Eray Demo

Signal name	Connections to be made
ERAY_A	Pin 3 of ERAY_A of Board 1 - Pin 3 of ERAY_A of Board 2
ERAY_A	Pin 4 of ERAY_A of Board 1 - Pin 4 of ERAY_A of Board 2
ERAY_A	Pin 5 of ERAY_A of Board 1 - Pin 5 of ERAY_A of Board 2
ERAY_B	Pin 3 of ERAY_B of Board 1 - Pin 3 of ERAY_B of Board 2
ERAY_B	Pin 4 of ERAY_B of Board 1 - Pin 4 of ERAY_B of Board 2
ERAY_B	Pin 5 of ERAY_B of Board 1 - Pin 5 of ERAY_B of Board 2

Fr_17_Eray demo establishes FR Cluster consisting of two nodes Node0 and Node1.

Step 1: Download the `DemoApp_Node0.hex` to Node0 Triboard and `DemoApp_Node1.hex` to Node1 TriBoard.
Step 2: Select the Option 1 for choosing FR Node 1 Slave node in Slave TriBoard.
Step 3: Both nodes synchronize to cluster after Module initialization.
Step 4: All the cluster related information are read and verified in Node0.
Step 5: Node0 transmits the data to Node1 on to network.
Step 6: Node1 transmits the received data to Node0 on to network.
Step 7: Transmitted and received data on Node0 is verified.

3.1.9 Option i: Gpt Demo

The following table provides the Gpt demo options supported.

Table 10 Gpt DemoApp menu

Option	Description of the demo
<1>	Start continuous timer: LED must start blinking; The demo starts GPT Channel in continuous mode. The notification function will be called after timeout; inside notification function, a LED is toggled.
<2>	Stop timer: LED must stop blinking; the demo stops the GPT Channel.
<.>	Back to main menu

3.1.10 Option j: Icu_17_TimerIp Demo

Icu_17_TimerIp demo will measure the High Time, Duty Cycle and Period of the PWM generated by PWM demo. Hence, Output of PWM needs to be connected to the input of ICU. First, generate duty cycle using PWM demo then run Icu_17_TimerIp demo.

Table 11 External connections for Icu_17_TimerIp Demo

Signal name	Connections to be made
PWM <-> ICU	Pin 33.7 <->Pin 02.0 (Plug Connector X703: T2LA Board X10 Connector Pin 2 and Plug Connector X704 X1 Connector Pin 3 needs to interconnected using a connecting wire) – This connection is for TC39x,TC38x,TC37x,TC36x and TC337 Pin 00.0 <->Pin 02.0 (Plug Connector X703: T2LA Board X1 Connector Pin 2 and Plug Connector X704 X1 Connector Pin 3 needs to interconnected using a connecting wire) – This connection is for TC35x,TC337_ED

The output will be displayed in the following format:

- High Time = xx
- Period = xx
- Duty Cycle = xx

Note: High Time and Period are displayed in hexadecimal and Duty Cycle is displayed in decimal.

Note: In case the connections are made (PWM->ICU), and the Duty Cycle measured is displayed as 0, then additional hint message is printed indicating that a capture operation is in progress.

3.1.11 Option k: Mcu Demo

Table 12 Mcu DemoApp menu

Option	Description of the demo
<1>	Perform Software Reset : The Microcontroller will get a Soft Reset
<2>	Get Reset Reason
<x>	Back to main menu

Note: In order to display the menu on the HyperTerminal after the Option <1>, press the ENTER key on the keyboard.

Note: The Mcu demo should be run before the Wdg_17_Scu demo otherwise the MCU software reset demo might not work.

3.1.12 Option l: Pwm_17_GtmCcu6 Demo

Table 13 Pwm_17_GtmCcu6 DemoApp menu

Option	Description of the demo
<1>	Start PWM, default DutyCycle :50%
<2>	Enter New DutyCycle: For Example: 0 75, where 0 is Id (Pwm Channel Id) and 75 is Data (duty in %)
<3>	Stop PWM Demo
<4>	Back to main menu

3.1.13 Option m: Spi Demo

Spi DemoApp uses Sequence 0(Sync) and Sequence 1(Async) to transfer data using QSPI0 and QSPI2 kernels respectively. It transfers data synchronously using an IB (Internal Buffer) and asynchronously using EB (External Buffer) mechanism. In the first option, the SPI data transmitted from transmit port pin is being read back on the receive port pin on the triboard whereas the second option uses the internal loop back mode feature of QSPI kernel to transfer the data.

Table 14 External connections for Spi Demo

Signal name	Connections to be made
MTSR(QSPI0) <-> MRTS(QSPI0)	P20.14 <-> P20.12 (Plug Connector X702: T2LA Board X3 Connector Pin 6 and Pin 5 needs to interconnected using a connecting wire)

Note: For QSPI2 internal loop back mode feature is being used which is enabled through Spi_ControlLoopBack () API. Hence, no hardware connections are required for it.

Table 15 Spi DemoApp menu

Option	Description of the demo
<1>	Transfer IB data over QSPI0 using Synch Sequence
<2>	Transfer EB data over QSPI2 using Asynch Sequence
<x>	Back to main menu

Option 1: Transmit and receive 20 bytes (10 bytes each on 2 different SPI Channels) of SPI data {Value: 0-19} transmitted synchronously over QSPI0 kernel using the IB (Internal Buffer) mechanism. This SPI Data is transmitted using a Synch Sequence0 having one Job0 and two SPI Channel 0 and 1 linked to it. SPI Sequence result is polled to know the Sequence transmission completion status before reading the received data for evaluation of successful data transfer.

Option 2: Transmit and receive 20 bytes of SPI data {Value: 50-69} transmitted asynchronously in Interrupt Mode by DMA over QSPI2 kernel using the EB (External Buffer) mechanism. The SPI Data is transmitted using an Asynch Sequence1 having one Job1 and one SPI Channel2 linked to it. This SPI data transfer happens using the internal loopback mode feature without using any physical pin connection. SPI driver and HW status is being checked to know when it is free/busy. SPI Sequence result is being polled to know the Sequence transmission completion status before using the received data for evaluation of successful data transfer.

3.1.14 Option n: Wdg_17_Scu Demo

The following table provides the Wdg_17_Scu demo options supported:

Table 16 Wdg_17_Scu DemoApp menu

Option	Description of the demo
<s>	Set WDG0 in slow mode. Each press of key‘s’ will set trigger timeout period of 7 Seconds. 2nd LED on the Tri-Board toggles until watchdog timeout occurs (at the rate of WdgSlowRefreshTime).
<f>	Set WDG0 in Fast mode. Each press of key ‘f’ will set trigger timeout period of 4 Seconds. 2nd LED on the Tri-Board toggles until watchdog timeout occurs (at the rate of WdgFastRefreshTime).
<t>	Trigger WDG0 timeout for 4 seconds in current mode. 2nd LED on the Tri-Board toggles until watchdog timeout occurs.

Option	Description of the demo
<x>	Stop WDG0 Timer.
<.>	Stop All WDG Timers and exit WDG demo.

Note: The Wdg_17_Scu demo to be started with option's' or option 'f' and user should not check the Wdg demo with the option't'.

Note: The Wdg_17_Scu demo uses the STM timer internally. If this demo is run after 42.94 s (the timeout value for STM), DET is observed in the Wdg_17_Scu demo.

Note: Disconnect the TriBoard from Debugger and reset once before running the WDG demo.

3.1.15 Option o: Crc Demo

Table 17 Crc DemoApp menu

Option	Description of the demo
<1>	CRC Driver Demo 8bitCRC with 0x1D polynomial and "MODE" based method calculation
<2>	CRC Driver Demo 8bitCRC with 0x2F polynomial and "MODE" based method calculation
<3>	CRC Driver Demo 16bitCRC with 0x1021 polynomial and "MODE" based method calculation
<4>	CRC Driver Demo 32bitCRC with 0x04C11DB7 polynomial and "MODE" based method calculation
<5>	CRC Driver Demo 32bitCRC with 0xF4ACFB13 polynomial and "MODE" based method calculation
<6>	CRC Version Information
<x>	Back to main menu

Note: MODE is the implementation mode of that Crc and is selected from the configuration. It is Runtime, Table or Hardware.

3.1.16 Option p: FlsLoader Demo

In this demo FlsLoader, basic APIs usage is demonstrated. First `FlsLoader_Init` API's execution status is printed and then following menu options will be given.

Table 18 FlsLoader DemoApp Menu

Option	Description of FlsLoader demo
<1>	DFlash Programming: DFlash erase and programming demo. Need to provide the valid start address of a Dflash sector, which needs to be erased and programmed. Example: <Parameter ID> <Space><Dflash sector start address in Hex> 0 AF000000<Enter Key> <Enter Key> (total 2 times) Erases single Dflash sector as per the input address, then programs with 512 bytes of dummy data. Also prints results of each operation.

Option	Description of FlsLoader demo
<2>	DFlash Read: DFlash read demo. Reads first 8 words from the previously programmed location and prints on the screen.
<3>	PFlash Programming: PFlash erase and programming demo. Need to provide the valid start address of a Pflash sector, which needs to be erased and programmed. Example: <Parameter ID> <Space><Pflash sector start address in Hex> 0 A0100000<Enter Key> <Enter Key> (total 2 times) Erases single Pflash sector as per the input address, then programs with 512 bytes of dummy data. Also prints results of each operation.
<4>	PFlash Read: PFlash read demo. Reads first 8 words from the previously programmed location and prints on the screen.
<x>	Go back to main menu: Calls `FlsLoader_DeInit` API prints the returned value and exits the FlsLoader demo.

Note: The lock and unlock features demo is implemented in the DemoApp, but it is disabled by a compile switch to avoid unexpected memory lock situations.

3.1.17 Option q: Smu Demo

The following table provides the Smu demo options supported. This demo will trigger the SW alarm 0 to generate and verify the SMU Interrupt Request 0.

Table 19 Smu DemoApp menu

Option	Description of the demo
<1>	Trigger SWAlarm10[0] to send the SMU Interrupt Req0
<x>	Back to main menu

3.1.18 Option r: I2c Demo

I2c DemoApp writes and reads back data from the EEPROM. A stream of data will be written in the EEPROM.

I2c is connected to a serial EEPROM via P15.4 and P15.5 of the Triborad with a size of 2KBit (2 x 128 x 8). The slave address of this EEPROM device is 0x50.

Result of successful execution: If the stream data matches with the data present at EEPROM, I2c DemoApp will print the message “Sent and Recieved data matched Read successful”.

Result of failed execution: If the stream data mismatches with the data present at EEPROM, I2c DemoApp will print the message “Sent and Recieved data mismatched Read unsuccessful”.

3.1.19 Option s: Lin_17_AscLin Demo

The DemoApp uses ASCLIN1 Kernel unit.

The following table provides the details of the Lin_17_AscLin driver DemoApp menu

Table 20 Lin_17_AscLin DemoApp menu

Option	Description of the demo
<1>	Send Sleep Command to bus - Lin_GoToSleep Sends Sleep command (0x00,0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF) on Lin bus and sets channel state to LIN_CH_SLEEP
<2>	Send Wakeup Signal to Bus (250us to 5ms) - Lin_WakeUp Sends wake up signal on Lin bus and sets channel state to LIN_OPERATIONAL
<3>	Internal Sleep - Lin_GoToSleepInternal Sets channel state to LIN_CH_SLEEP
<4>	Internal Wakeup - Lin_WakeupInternal Sets channel state to LIN_OPERATIONAL
<.>	Back to main menu

3.1.20 Option t: Uart Demo

The DemoApp uses ASCLIN2 Kernel unit.

Connect the UART ASCLIN2_ATX pin to the ASCLIN2_ARXA pin (external loop back).

Table 21 External connections for Uart Demo

Signal name	Connections to be made
ATX <-> ARXA	P10.5<-> P14.3 (Plug Connector X702: T2LA Board X9 Connector Pin 3 and Plug Connector X703: T2LA Board X10 Connector Pin 4 needs to be interconnected using a connecting wire)

The following table provides the details of the Uart driver DemoApp menu:

Table 22 Uart DemoApp menu

Option	Description of the demo
<1>	Get UART Channel Status -Will display the status of UART channel.
<2>	Transmit and Receive Data on the UART Channel -Will perform read and write operation on UART channel and displays the status.
<3>	Abort After Data Transmission and Reception -Will display number of bytes transmitted before Aborting the Write Operation and number of bytes received before Aborting the Read Operation.
<4>	Abort On-Going Data Transmission and Reception -Will display number of bytes transmitted before Aborting the ongoing Write Operation and number of bytes received before Aborting the ongoing Read Operation.
<.>	Back to main menu

3.1.21 Option u: Bfx Demo

The following table provides the details of the Bfx driver DemoApp menu:

Table 23 Bfx DemoApp menu

Running the DemoApp

Option	Description of the demo
<1>	SetBit,GetBit and ClrBit functionality check -SetBit changes status of given bit position of data to 1, GetBit returns TRUE if status of given bit position is 1 and ClrBit changes status of given bit position of data to 0.
<2>	SetBits and GetBits functionality check -SetBits sets the logical status of the bits of the Data parameter for given position bits to 1 -GetBits returns status of bits of data parameter for given position of bits.
<3>	SetBitMask,ClrBitMask and TstBitMask functionality -SetBitMask sets logical status of the bits of the Data parameter to 1 according to mask value -TstBitMask returns TRUE when the logical status of all the bits defined in the Mask parameter are also set at the same bit position in the Data input parameter. - ClrBitMask clears the logical status of the bits of the Data parameter to 0, for all the bit positions for which the logical status of bit in the Mask parameter is set to 1.
<4>	Even Parity functionality check - Function returns TRUE when the number of bits whose logical status is set to 1 in the Data input parameter is even, otherwise the function returns FALSE.
<5>	Left shift and right shift functionality check -Left Shift function shifts data bits to left by given shift count. The least significant bit (right-most bit) is replaced by a 0 bit and the most significant bit (left-most bit) is discarded for every single bit shift cycle. -Right Shift function shifts the bits of the Data parameter to the right by ShiftCnt count. The most significant bit (left-most bit) is replaced by a 0 bit and the least significant bit (right-most bit) is discarded for every single bit shift cycle.
<6>	Left rotate and right rotate functionality check -Left Rotate rotates the bits of the Data parameter to the left by ShiftCnt count. The most significant bit (left-most bit) is rotated to the least significant bit (right-most bit) location for every single bit shift cycle. -Right Rotate rotates the bits of the Data parameter to the right by ShiftCnt count. The least significant bit (right-most bit) is rotated to the most significant bit (left-most bit) location for every single bit shift cycle.
<7>	Toggle bit functionality check -Toggle bit toggles all the bits of the Data parameter (1's complement of the Data parameter).
<x>	Back to main menu

3.1.22 Option v: Stm Demo

The following table provides the details of the Stm driver DemoApp menu:

Table 24 Stm DemoApp menu

Option	Description of the demo
<1>	Precondition :- Stm Module is enabled by calling the Api Enable Module() Perform Enable Alarm Service(): In Continuous Mode Enter Number of Ticks: 1)100000000(1 Sec), 2)50000000(0.5 Sec), 3)200000000(2 Secs).

Option	Description of the demo
	An interrupt is raised after the delay of provided tick's value and a defined call back function is called results in LED BLINKING continuously on the Triboard.
<2>	Perform Disable Alarm() This results in disabling all the interrupts services Enabled while Enable Alarm service. Set one LED port pin ON - LED must stop blinking and remains ON.
<x>	Enter this option for returning to the main MENU.

3.1.23 Option w: Ocu Demo

The Ocu demo provides an overview of the compare match event functionality as Notification call, Pin Action Compare match event programmed and also Pin State change by calling the OCU APIs. GTM ATOM05 channel is used with Port 02 and Pin 5

The following table provides the details of the Ocu driver DemoApp menu:

Table 25 Ocu DemoApp menu

Option	Description of the demo
<1>	Enables and starts compare event for default threshold 0x123456-value set for 24-bit Timer. Notification prints the current ticks value and channel is stopped
<2>	Enables and Starts the compare match event on the channel with user passed Relative threshold value. Notification prints the current ticks value and channel is stopped
<3>	Enables and Starts the compare match event on the channel with user passed reference value and Absolute threshold value. Notification prints the current ticks value and channel is stopped
<4>	Sets the Pin State Low or High based on Users input and Prints the changed state of Pin associated with Ocu Channel.
<5>	Sets the Pin Action as user Passed value and programs 3 compare match events to print the Pin action on channel then the channel is stopped.
<X>	Go to main menu.

3.1.24 Option x: Dsadc Demo

The Dsadc demo converts Dsadc channels AN2 and AN3 with 14-bit resolution for channel 0. G0CH2 and G0CH3 channels are used for the TC38x for applying differential input.

Table 26 External connections for Dsadc Demo

Signal name	Connections to be made
AN2	Connect the potentiometer/variable voltage source
AN3	Connect the potentiometer/variable voltage source

Table 27 Adc DemoApp menu

Option	Description of the demo
<1>	Start DSADC conversion result The conversion results should be the Voltage supplied to the Pins AN2 and AN3
<2>	Stop Conversion should stop conversion for the Voltage supplied to the Pin's AN2 and AN3

Option	Description of the demo
<3>	Get Channel should return Current Channel Status for the Voltage supplied to the Pin's AN2 and AN3
<x>	Back to main menu

3.1.25 Option y: CanTrcv_17_V9251 Demo

The DemoApp uses the CAN transceiver channels 0 and 1, which are interfaced through (CAN00(X301)) and (CAN10/ (TC337_ED) CAN03) (X302) of the TriBoard. The following connections have to be made externally to run the demo successfully.

Note:
TC337_ED device: Remove Zero Ohm Resistor R311 and R312 from Triboard. Solder two connecting wires to Tx and Rx pin of CanTrcv 9251 and connect their other end to Pin 13 and Pin 15 of X2 column of T2LA connector placed on X802(Peripheral connector) respectively.
TC337_PD device: Remove Zero Ohm Resistor R311 and R312 from Triboard and place it at R313 and R314 resistor location on the Triboard

Table 28 External connections for CanTrcv_17_V9251 Demo

Signal name	Connections to be made
CANL	(pin 3 of X301) <-> (pin 3 of X302)
CANH	(pin 4 of X301) <-> (pin 4 of X302)

The following table provides the details of the CanTrcv_17_V9251 driver DemoApp menu:

Table 29 CanTrcv_17_W9251 DemoApp menu

Option	Description of the demo
<1>	Set the Opmodes of CAN Transceiver channels 0 and 1 to NORMAL mode. If set opmode to NORMAL mode is successful then only transfer of Standard Frame between CAN controllers will take place. **Operational Specific:** Sets the Opmodes of CAN Transceiver channels 0 and 1 to NORMAL mode. Sets the mode of CAN controller 0 and 1 to STARTED, Transmits two STANDARD ID frames each from controller 0 to 1 and 1 to 0, Sets the mode of CAN controller 0 and 1 to STOPPED.
<2>	Get the Opmodes of CAN Transceiver channels 0 and 1.
<x>	Enter this option for returning to the main MENU.

3.1.26 Option z: CanTrcv_17_W9255 Demo

The demo application is not supported for this module since it requires transceiver from an external hardware.

3.1.27 Option 1: Sent Demo

The Sent DemoApp provides an overview of Sent channel enable, disable and read data from sensors in Fast and SPC modes.

External connections for Sent Demo

Connect I/O pin of Sent protocol specific sensor to port-00 and pin01 (P0.1).

Table 30 Sent DemoApp menu

Option	Description of the demo
<A>	Start Sent DemoApp. This operation prompt user to enter ChannelId. Note: ChannelId to provide in this demo is 0. Channel 0 configured in Sent.xdm.
<X>	Enter this option for returning to the main MENU.
	Menu For Selected Channel Id
<1>	Set Sent Channel to Enable state. If Sent Channel enabled, Get Channel Status should capture and return Channel state
<2>	Set Sent Channel to Disable state. If Sent Channel disabled, Get Channel Status should capture and return Channel state
<3>	Reads Sent Channel data form the sensor connected to configured channel. Note: Sensor connected in this operation should support Sent Fast channel mode (Ex: TLE4998S4).
<4>	Captures status of the configured channel.
<5>	Starts SPC Sync Master Pulse and Reads Sent Channel data form the sensor connected to configured channel. Note: Sensor connected in this operation should support Sent SPC mode (Ex: TLE4998C4).
<X>	Enter this option for returning to the main MENU.

3.1.28 Option 2: HSSL Demo

Hssl Demo transfers data between a Master and Slave device and hence requires two Triboards to run. An IEEE 1394 connector (FireWire 400 alpha 6 conductor connector) needs to be soldered on the twoTriboard at X201 (HSCT 0) location. An IEEE 1394 Fire wire cable should be connected between two boards before the start of the demo application.

The following table provides the details of the Hssl driver DemoApp menu:

Table 31 Hssl DemoApp Menu

Option	Description of the demo
<1>	Hssl Register Write: **Operational Specific:** Performs write operation, Single data value is written to the slave or target device's register
<2>	Hssl Register Read: **Operational Specific:** Performs read operation, Reads back the value written to the slave
<3>	Multiple Hssl Register Write: **Operational Specific:** Performs multi write operation, Multiple data values are written to the slave device
<4>	Start Stream Data: **Operational Specific:** Performs write Stream Operation, 32 bytes of data is written continously to the slave device in stream mode
<5>	Stop Stream Data: **Operational Specific:** Stops the stream operation

Option	Description of the demo
<6>	Read ID: **Operational Specific:** Reads the Slave ID which is present the slave device
<X>	Go to main menu.

Hssl Demo execution steps are as below:

Step 1: Download the `DemoApp_Node0.hex` to Master Node Triboard and `DemoApp_Node1.hex` to Slave Node Triboard.

Step 2: Select the Option 2 for choosing HSSL Slave in Slave Node Triboard.

Step 3: Master Node reads and writes the data to and from the Slave Node.

Step 4: Transmitted and received data is validated on the Master Node.

Note:

1. For testing stream operation both boards needs to be reset before starting the test.

3.1.29 Option 3: Iom Demo

Iom Demo execution steps are as below:

Step 1: Do the external connection between Port 33.0 to Port 33.3 (P3_X1.11 to P3_X3.13)

Step 2: Run the Demoapp by selecting Option3 in main menu.

Step 3: Following result is expected for Demoapp to Pass:

****

Demo to check service to clear the FPC cell edgeond threshold

****

RESULT_START

Result = Pass;

RESULT_END

3.2 Option 1:Main Menu:Multicore

To demonsrate the multicore feature of GPT and WDG module Core 0 and Core 1 is selected

Since TC337PD is single core device multicore option is not provided

3.2.1 Option 1:Gpt: Demo Multicore

1. The Gpt multicore option starts timer in Core0 and Core1.

2. LEDs starts Blinking continuously for both cores (Core 0 2nd LED, Core 1 – 5th LED)

3. CORE0 LEDS stop blinking after 128 times

4. CORE1 LEDS stop blinking after 192 times

3.2.2 Option 2: Wdg_17_Scu: Demo Multicore

1. Core 0 is set to SLOW_MODE with time out trigger period of 10 seconds

2. Core 1 is set to FAST_MODE with time out trigger period of 5 seconds

3. Core 0 LED (2nd LED) starts to blink continously until timeout occurs (10 seconds)

2. Core 1 LED (3rd LED) starts to blink continously until timeout occurs (5 seconds)

Revision history

Document version	Date of release	Description of changes
Release v18.0	2021-04-01	1. SPI Demo Content updated. 2. Ethernet Demo Content updated. 3. ICU demo pin connection info corrected.
Release v17.0	2020-12-11	Review comment updated for option 1 main menu (added multicore) Added Note in Building the Demo App section
Release v16.0	2020-12-07	Updated for , 1. GPT,WDG multicore option 2. 3.1.10 connection information correction
Release v15.0	2020-08-21	Minor correction and PIN connection updated for DIO ADC Demo Minor correction Wdg Demo, Minor correction
Release v14.0	2020-05-19	Added Note for testing HSSL Demo
Release v13.0	2020-05-18	Renamed Document as per Jira 0000053912-11124
Release v12.0	2020-03-23	Minor correction and PIN connection updated for ICU in table 11
Release v11.0	2020-01-29	Updated Iom, Sent and Can module for Demoapp. XDM details are updated in chapter 1.
Release v10.0	2019-10-04	Updated Hssl demo configuration description in section 3 and Lin_17_AscLin, Eth_17_GEthMac DemoApp menu details.
Release v9.0	2019-07-30	Updated Uart,Icu,Pwm demo configuration description in section 3
Release v8.0	2019-04-23	Updated Ocu Demo Port Pin configuration and GPT Demo details. Added information in Sections 2 and 3 for TC37A_ED device support.
Release v7.0	2019-04-12	Updated for Ocu Demo, Dsadc Demo and CanTrcv_17_V9251 Demo details.
Release v6.0	2019-02-07	Added info in Section 2 Getting Started about the EB tresos link file deletion.
Release v5.0	2019-02-04	Updated for Uart DemoApp Menu and hardware connection details.
Release v4.0	2018-10-12	Updated for I2c, Lin_17_AscLin, Uart, Bfx, and Stm DemoApps.
Release v3.0	2018-10-05	Modified for ADC hardware group demo.
Release v2.0	2018-05-30	Modified version for the Beta package.
Release v1.0	2017-12-14	Initial version.

Trademarks
All referenced product or service names and trademarks are the property of their respective owners.

Edition 2021-04-01
Published by
Infineon Technologies AG
81726 Munich, Germany

Do you have a question about this document?
Email: erratum@infineon.com

Document reference
AP32383

IMPORTANT NOTICE
The information contained in this application note is given as a hint for the implementation of the product only and shall in no event be regarded as a description or warranty of a certain functionality, condition or quality of the product. Before implementation of the product, the recipient of this application note must verify any function and other technical information given herein in the real application. Infineon Technologies hereby disclaims any and all warranties and liabilities of any kind (including without limitation warranties of non-infringement of intellectual property rights of any third party) with respect to any and all information given in this application note.

The data contained in this document is exclusively intended for technically trained staff. It is the responsibility of customer's technical departments to evaluate the suitability of the product for the intended application and the completeness of the product information given in this document with respect to such application.

For further information on the product, technology, delivery terms and conditions and prices please contact your nearest Infineon Technologies office (**www.infineon.com**).

WARNINGS
Due to technical requirements products may contain dangerous substances. For information on the types in question please contact your nearest Infineon Technologies office.

Except as otherwise explicitly approved by Infineon Technologies in a written document signed by authorized representatives of Infineon Technologies, Infineon Technologies' products may not be used in any applications where a failure of the product or any consequences of the use thereof can reasonably be expected to result in personal injury.