Application Report
SPRACB7–June 2018

Integrating AUTOSAR on TI SoC: Fundamentals

Sujith Shivalingappa, Yashwant Dutt and Sunil Kumar MS

ABSTRACT

TI's TDAxx and DRA7xx family of SoC's provide multi-core, heterogeneous, hardware accelerated functionalities, which could be employed to realize Advanced Driver Assist Systems (ADAS) and Digital Cockpit.

AUTOSAR is industry standard software architecture for ECU's. This application report lists some of the key considerations required while deploying AUTOSAR and NON-AUTOSAR software architecture on TI devices, to achieve optimal functional and performance goals of an ADAS system.

The document also details the resource partitioning between AUTOSAR, NON-AUTOSAR software and onus of power-up of peripherals (between the secondary bootloader and the AUTOSAR software).

Contents

List of Figures

Trademarks

1 Introduction

TI device architecture inherently allows co-existence of diverse software on different cores with shared resources. The integrator decides or allocates the required resources to different cores or software hosted on different cores. The basic communication between the cores is established through the mailbox, a dedicated hardware.

In this document, VisionSDK/Processor SDK Vision is used as NON-AUTOSAR software reference and expected that you are familiar with it.

2 Architecture

The TDA2x, DRA7x, DRA7x and TDA3x architectures shown in Figure 1 and Figure 2 depict co-existence of NON-AUTOSAR with AUTOSAR stack.

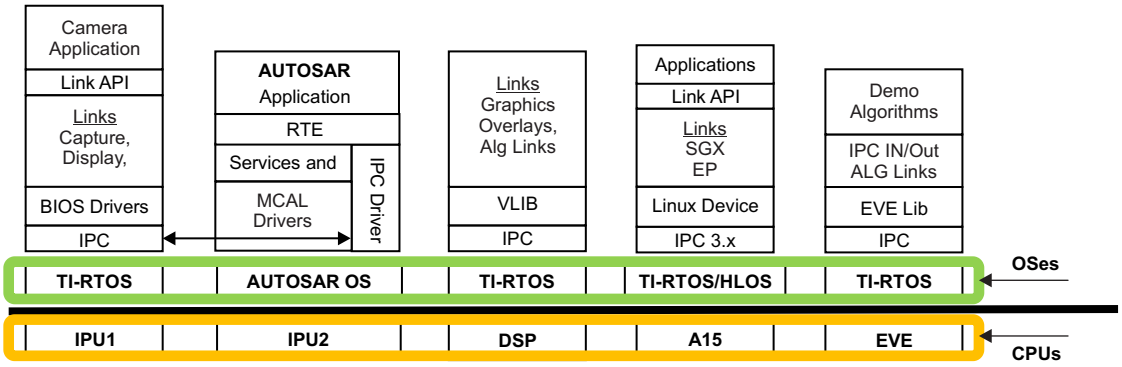

Figure 1. TDA2x/DRA7x Architecture

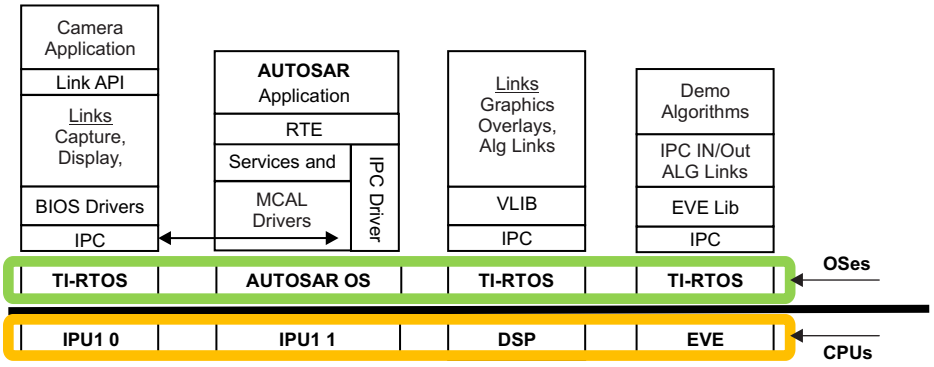

Figure 2. TDA3x Architecture

- IPC : Refers to IPC Lib, which is provided by TI as part of Processor SDK TDAx [1], [2]
- IPC 3.x : Refer to IPC on Linux
- IPC between IPU1_0 to other cores is not show, it is assumed to be available.
- TI provides MCAL layer of the AUTOSAR stack and following modules are provided
- Common: Can, Dio, Eth, Gpt, Mcu, Ipc (CDD) Port, Spin Wdg and Pwm
- TDA3x Specific: Adc and Can FD
- TDA2Px Specific: Can FD

Ipc (CDD) provides the communication link between IPU2_0 to IPU1_0 on TDA2xx and DRA7xx and between IPU1_0 to IPU1_1 on TDA3x.

2.1 Resource Partitioning

2.1.1 Peripherals

The peripheral should be statically allocated to required core. Ensure to allocate peripheral to one core only. There could be multiple instances of same peripheral (for example, serial peripheral interface (SPI), controller area network (CAN), general-purpose input/output (GPIO)), each instance could be allocated to different cores.

2.1.2 Memory

The DDR or memory is a shared resource between cores. Sections or segments of memory (contiguous or non-contiguous) could be assigned to a given core and the access rights to this sections can be controlled by memory management units (MMU), TI devices have multiple MMU.

SDK provided by TI defines memory segment for all the cores ([1], [2])

2.1.3 Shared Memory

Typically, each core performs a discrete set of actions; the output of these actions would be required to be shared with other cores (sometime along with the input) for subsequent actions and processing.

This could be achieved by defining a shared segment and section in memory, which could be accessed by all cores. Section 2.1.3.1 and Section 2.1.3.2 lists two usecases for shared memory segment.

Ensure that MMU allows read and write access to this shared section from each core.

2.1.3.1 Memory for IPC

The hardware provided mailbox allows transportation of a 32-bit value across cores, applications might require to transport much larger data (for example, video frame, CAN command, Algorithm configuration data, and so forth). The basic idea is to allocate space in shared memory to hold the data (video frame) and transport the pointer to this frame across cores.

To establish a logical communication channel, there may be need to send meta-data along with actual data. (for example, a structure could contain one or more pointers that points to actual data and other members could be used to indicate ID, sequence number, originator, and so forth, referred as message containers).

The shared memory is used to maintain message containers. Pointers to this message container are exchanged between cores.

2.1.3.2 Memory for Data

In cases where core hosting AUTOSAR requires memory that has to be shared with NON-AUTOSAR applications hosted on other cores (for example, AUTOSAR core, receives video frame from Eth and that requires to be processed in NON-AUTOSAR core), the memory has to be allocated from the shared area (see [1], [2]).

3 System Initializations

3.1 Power Up/SBL

Figure 3 shows the boot up sequence of the RTOS/sysBios based system.

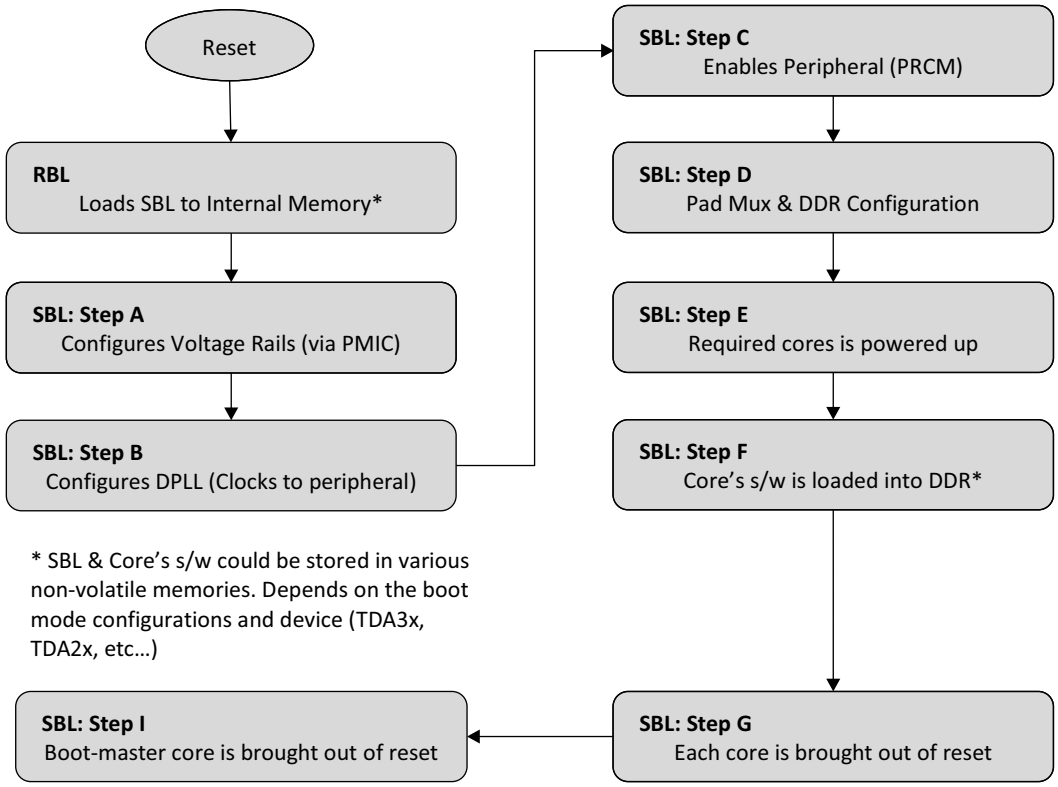

Figure 3. SBL Flow chart

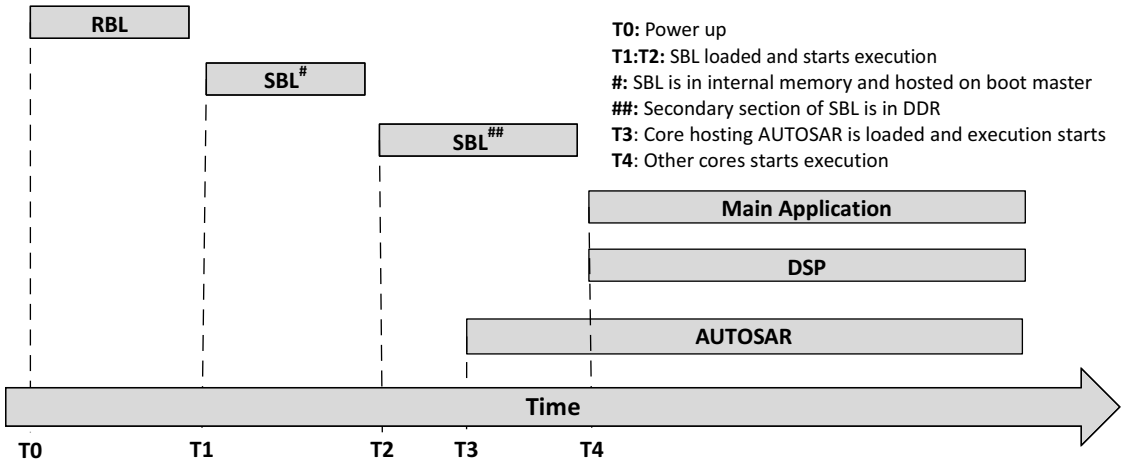

Figure 4. SBL Timeline/Sequence

3.1.1 MMU Configuration

3.1.1.1 TDA2x and DRA7x

TDA2x and DRA7x : IPU2 has a dedicated MMU and should be configured.

3.1.1.1.1 Recommended

IPU2_0 could setup the MMU required for IPU2 (software application hosted on IPU2 could setup the MMU as part of the start-up sequence). There are no restrictions if IPU2_1 is used to configure the MMU. The MCU module will not perform MMU configurations. It is expected that MMU is configured before MCAL modules are initialized.

3.1.1.1.2 Optional

SBL could be used to setup the MMU for IPU2, below listed are some of the key points to consider, in this case.

- Feature Additions: When ECU software has to be updated to support new features and these features requires change in memory map, the SBL would require an update. Update of SBL is generally regarded as complex process as compared to update of application software.

SBL could setup MMU on or before Step D, see Figure 3.

3.1.1.2 TDA3x

The SBL sets up MMU on or before Step D (see Figure 3).

Optionally, a simple pre-application start code could be used to program the MMU. However, it should be ensured that the pre-application start code should run first.

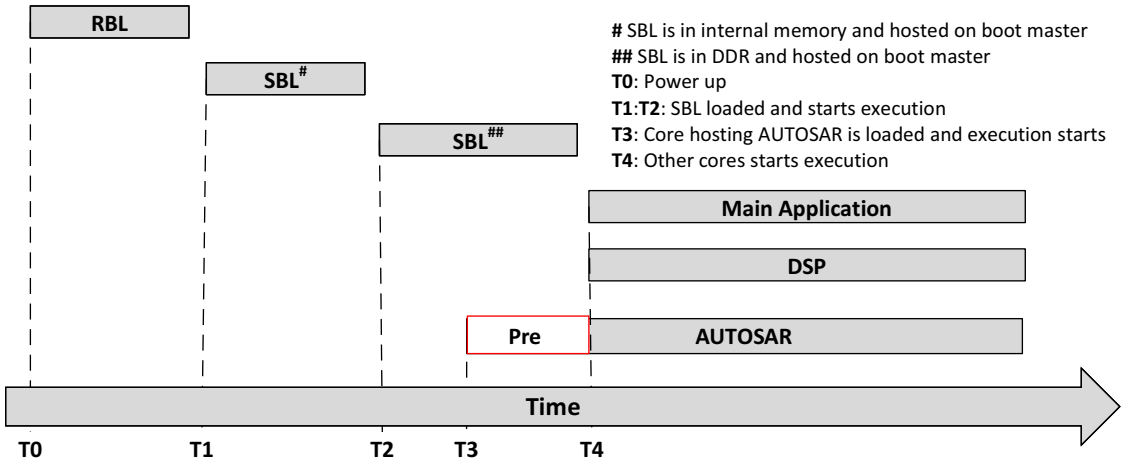

Figure 5. SBL Timeline/Sequence With AUTOSAR Start-Up

3.1.2 Interrupts

On the TDAxx and DRA7x class of processors, an interrupt from the peripheral could be routed to a core (potentially to any core), this operation is achieved thorough a dedicated hardware block called “crossbar”.

The interrupt crossbar (interrupt routing) selection register is 32-bit register. Each register control routing of more than 1 interrupt source.

Since core hosting AUTOSAR and NON-AUTOSAR software are asynchronous with regard to each other and could potentially access same crossbar, which could lead to a potential failure. The following recommendation could be employed to avoid this potential failure.

3.1.2.1 Interrupt Crossbar Options

3.1.2.1.1 Recommendation

The distribution of peripheral across core ensures that shared crossbar is not used. The crossbar used by one core will not be used by any other core in SoC.

3.1.2.1.2 Optional

SBL performs the required crossbar configuration for all peripherals at Step D for peripherals that are used by AUTOSAR (see Figure 3).

3.1.3 Peripheral Power Up (PRCM) and Functional Clock

The SBL should power up and setup the functional clock required for the peripheral. From the AUTOSAR perspective, no explicit power up or functional clock setup would be required.

4 MCAL Modules

4.1 MCU

Not all the MCU module APIs are needed or supported and as show in Figure 3; clocks are configured as part of SBL.

4.1.1 Feature Not Supported

The following service API's are not supported and should not be used:

- `Mcu_SetMode ():` All the SoC power control is performed by SBL
- `Mcu_GetPllStatus ():` Always return's MCU_PLL_LOCKED, as PLL is configured by SBL
- `Mcu_DistributePllClock ():` Always returns E_OK, as PLL is configured by SBL
- `Mcu_InitClock ():` Should not be used as clock is initialized in SBL

4.1.2 Recommendation

Use Port Build Variant to configure Mcu module.

4.1.2.1 MMU Configuration

The MCU module will not configure MMU. The MMU configuration is expected to be performed by other software (Startup Code, with regard to AUTOSAR), similarly for Cache.

4.1.2.2 Crossbar configuration

It is recommended that Section 3.1.2.1.1 is used, in which case normal initialization of the MCU would suffice. In cases where Section 3.1.2.1.1 cannot be used, SBL should perform the required cross bar configuration (see Section 3.1.2.1.2). Ensure to bypass crossbar configuration in Mcu by:

```
Mcu_ConfigType.Mcu_IrqXbarConfig = (const Mcu_IrqXbarConfigType *) NULL_PTR;
Mcu_ConfigType.Mcu_NumberOfIrqSources = 0U;
```

For more details on crossbar, see Section 3.1.2.

4.1.2.3 Clock Configuration

All required peripheral clock configuration should be performed by SBL, hence, MCU should not perform it.

```
Mcu_ConfigType. Mcu_ClockConfig = (const Mcu_ClockConfigType *) NULL_PTR;
Mcu_ConfigType. Mcu_NumberOfClockConfig = 0U;
```

The peripheral clock could be sourced from multiple locations and PLL's. These peripherals require a very accurate functional clock for deterministic operations, which require re-configuring one or more PLL (and or dividers).

A single PLL could potentially drive multiple peripherals, hence, it is recommended that one entity perform all of the required PLL configurations, dividers, and so forth.

It is recommended that all peripheral PRCM/Clock configurations are done in Step B (see Figure 3).

4.2 Port

4.2.1 Recommendation

Use Post Build variant to configure Port.

4.2.1.1 TDA2xx, DRA7x I/O Pad Properties

As described in TDAxx/DRA7x TRM manuals, located at http://www.ti.com/processors/automotive-processors/tdax-adas-socs/technical-documents.html (see "Pad Configuration/Isolation Requirements), while configuring mode selection, delay, mux mode, and so forth, the device I/O's should be isolated. It is recommended that I/O properties are not changed in application software. It should be performed once (in a power cycle) and preferably at start up. This configuration should be part of SBL pre-application start function.

It is recommended to:

- Identify the I/O pad requirements (such as mode, pull type, slew control, and so forth)
- Configure the required I/O pad properties
- For the sequence to program the pad properties, see the *Port_ConfigurePadCore ()* function in file mcal_\Port\src\tdaxxx\Port_PlatformTda2xx.c.

Use "PORT_PAD_IO_PROPERTIES_UPDATE" to turn OFF port module updating the I/O pad properties.

4.2.1.2 GPIO Module Reset

Ensure that the other PIN's of the GPIO instance being used are not used by other modules and cores. If other cores are used, the reset of the GPIO instance should be done by SBL at Step D (see Figure 3).

```
Port_ConfigType.Port_DioRegConfigType[X].Port_DioDoReset = FALSE;
```

Provided that other cores are using same instance of GPIO.

`Port_ConfigType.Port_DioRegConfigType[X].Port_DioRegId` is interpreted as X

It is recommended that all GPIO soft reset is done after Step E (see Figure 3).

4.2.1.3 Pin Mode

Most of the PIN's on TDAxx family of devices are multiplexed with two or more functionalities. It is the system designer's responsibility to ensure the correct mode is selected for the given PIN. It is recommended that an entity outside AUTOSAR (such as SBL) perform all of the pin mode selections including mmr lock and unlock of control registers, to avoid multiple master configuring and accessing shared resources.

```
Port_ConfigType.Port_PinConfigType = (Port_PinConfigType *) NULL_PTR;
Port_ConfigType.NumberOfPortPins = 0U;
```

It is recommended that all GPIO soft reset is done after Step E (see Figure 3). The device manual of TDA2xx and DRA7x also recommends this.

5 References

1. Processor SDK for TDAx ADAS SoCs - Linux and TI-RTOS Support
2. Processor Software Development Kit for DRA7x Jacinto™ Processors – Linux, Android, and RTOS
3. Jacinto™ automotive processors
4. MCAL Version 01.09.00 (Contact your FAE support for the MCAL releases)

IMPORTANT NOTICE FOR TI DESIGN INFORMATION AND RESOURCES

Texas Instruments Incorporated ('TI") technical, application or other design advice, services or information, including, but not limited to, reference designs and materials relating to evaluation modules, (collectively, "TI Resources") are intended to assist designers who are developing applications that incorporate TI products; by downloading, accessing or using any particular TI Resource in any way, you (individually or, if you are acting on behalf of a company, your company) agree to use it solely for this purpose and subject to the terms of this Notice.

TI's provision of TI Resources does not expand or otherwise alter TI's applicable published warranties or warranty disclaimers for TI products, and no additional obligations or liabilities arise from TI providing such TI Resources. TI reserves the right to make corrections, enhancements, improvements and other changes to its TI Resources.

You understand and agree that you remain responsible for using your independent analysis, evaluation and judgment in designing your applications and that you have full and exclusive responsibility to assure the safety of your applications and compliance of your applications (and of all TI products used in or for your applications) with all applicable regulations, laws and other applicable requirements. You represent that, with respect to your applications, you have all the necessary expertise to create and implement safeguards that (1) anticipate dangerous consequences of failures, (2) monitor failures and their consequences, and (3) lessen the likelihood of failures that might cause harm and take appropriate actions. You agree that prior to using or distributing any applications that include TI products, you will thoroughly test such applications and the functionality of such TI products as used in such applications. TI has not conducted any testing other than that specifically described in the published documentation for a particular TI Resource.

You are authorized to use, copy and modify any individual TI Resource only in connection with the development of applications that include the TI product(s) identified in such TI Resource. NO OTHER LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE TO ANY OTHER TI INTELLECTUAL PROPERTY RIGHT, AND NO LICENSE TO ANY TECHNOLOGY OR INTELLECTUAL PROPERTY RIGHT OF TI OR ANY THIRD PARTY IS GRANTED HEREIN, including but not limited to any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information regarding or referencing third-party products or services does not constitute a license to use such products or services, or a warranty or endorsement thereof. Use of TI Resources may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

TI RESOURCES ARE PROVIDED "AS IS" AND WITH ALL FAULTS. TI DISCLAIMS ALL OTHER WARRANTIES OR REPRESENTATIONS, EXPRESS OR IMPLIED, REGARDING TI RESOURCES OR USE THEREOF, INCLUDING BUT NOT LIMITED TO ACCURACY OR COMPLETENESS, TITLE, ANY EPIDEMIC FAILURE WARRANTY AND ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT OF ANY THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

TI SHALL NOT BE LIABLE FOR AND SHALL NOT DEFEND OR INDEMNIFY YOU AGAINST ANY CLAIM, INCLUDING BUT NOT LIMITED TO ANY INFRINGEMENT CLAIM THAT RELATES TO OR IS BASED ON ANY COMBINATION OF PRODUCTS EVEN IF DESCRIBED IN TI RESOURCES OR OTHERWISE. IN NO EVENT SHALL TI BE LIABLE FOR ANY ACTUAL, DIRECT, SPECIAL, COLLATERAL, INDIRECT, PUNITIVE, INCIDENTAL, CONSEQUENTIAL OR EXEMPLARY DAMAGES IN CONNECTION WITH OR ARISING OUT OF TI RESOURCES OR USE THEREOF, AND REGARDLESS OF WHETHER TI HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

You agree to fully indemnify TI and its representatives against any damages, costs, losses, and/or liabilities arising out of your non-compliance with the terms and provisions of this Notice.

This Notice applies to TI Resources. Additional terms apply to the use and purchase of certain types of materials, TI products and services. These include; without limitation, TI's standard terms for semiconductor products http://www.ti.com/sc/docs/stdterms.htm), evaluation modules, and samples (http://www.ti.com/sc/docs/sampterms.htm).

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265