- **Multi-instantiable and re-entrant safe driver**
- **Modeled after TI Device Driver Architecture for Stream Class Devices that allows for easy porting and customization**

Description

This EDMA3 Driver has been developed for the following h/w and s/w environment

Hardware:

- Target Boards: DA830 EVM, TCI6608 (Little and Big Endian) Simulator, TCI6616 (Little and Big Endian) Simulator, C6748 EVM, OMAPL138 EVM, C6472 (Little and Big Endian) EVM, TCI6486 (Little and Big Endian) EVM, TI816X EVM, TI814X EVM, TI811X EVM. C6670 EVM, C6678 EVM, tci6614 EVM, tci6614 Simulator, Vayu EVM (DRA7xx), AM335x EVM, AM437x EVM.
- Emulation Setup: TI Emulator.
- Cabling: Standard Serial Connector

Software:

- CCS: 5.5.0
- Operating System: SYS BIOS 6.40.03.39
- C6x Code generation tools: 7.4.4
- TMS470 Code Generation Tools: 5.1.5
- Arm GCC A15 Toolchain: gcc-arm-none-eabi-4_7-2013q3
- ARP32 Code Generation Tools: 1.0.2
- XDC Tool chain: 3.30.04.52
- TCI6608/TCI6616 Simulator: 1.0.0

Other S/W Components Used:

- None

Capabilities

The DSP BIOS EDMA3 Driver adopts a scalable architecture that eases customization/extension

- Isolates H/W and OS Accesses, Easy to maintain & re-target to new platforms
- Can stack custom-functions along control/data-path to realize “driver filters”
- Uses the EDMA3 Resource Manager services
- Supports Multiple Instances

For easy and quick reconfiguration of driver for different SoCs, all its global configurable parameters can be passed at run time to the API EDMA3_DRV_create (), to create the SoC specific EDMA3 Driver Object. In case this configuration is not passed at run time, it can be taken from the EDMA3 configuration file edma3_<PLATFORM_NAME>_cfg.c, for the specific SoC, if it has been provided there. The configuration files can be found in “edma3_lld_<VERSION_NUMBER>\packages\ti\sdo\edma3\rm\src\configs” folder.

Similarly, the shadow region specific information can also be passed at run time to the API EDMA3_DRV_open (), to create region specific EDMA3 Driver Instance. In case this configuration is not passed at run time, it can be taken from the EDMA3 configuration file edma3_<PLATFORM_NAME>_cfg.c, for the specific SoC, if it has been provided there. The configuration files can be found in “edma3_lld_<VERSION_NUMBER>\packages\ti\sdo\edma3\rm\src\configs” folder.

The following table gives a quick overview of the supported API services. For help on interfaces, refer to the EDMA3 Driver Help File.

EDMA3 Driver APIs:

EDMA3_DRV_getVersion ()	The function is used to get the version information of the EDMA LLD
EDMA3_DRV_getVersionStr ()	The function is used to get the version string for the EDMA LLD
EDMA3_DRV_create ()	Create (and initialize) a given EDMA3 Driver (object)
EDMA3_DRV_delete ()	Delete a given EDMA3 Driver (object)
EDMA3_DRV_open ()	Open Instance of the EDMA3 driver
EDMA3_DRV_close ()	Close Instance of the EDMA3 driver
EDMA3_DRV_requestChannel ()	Request for a logical channel for data transfer
EDMA3_DRV_freeChannel ()	Free the earlier requested logical channel
EDMA3_DRV_clearErrorBits ()	Clean up the earlier used logical channel (For removable devices)
EDMA3_DRV_linkChannel ()	Link two logical channels
EDMA3_DRV_unlinkChannel ()	Unlink two earlier linked channels

EDMA3_DRV_chainChannel ()	Chain two logical channels
EDMA3_DRV_unchainChannel ()	Unchain two earlier chained channels
EDMA3_DRV_setOptField ()	Set a particular subfield within the OPT field of a PaRAM Set
EDMA3_DRV_getOptField ()	Get a particular subfield within the OPT field of a PaRAM Set
EDMA3_DRV_setSrcParams ()	Set the Source specific parameters of the data transfer
EDMA3_DRV_setDestParams ()	Set the Destination specific parameters of the data transfer
EDMA3_DRV_setSrcIndex ()	Set the Source specific indices for the data transfer
EDMA3_DRV_setDestIndex ()	Set the Destination specific indices for the data transfer
EDMA3_DRV_setTransferParams ()	Set the various counts for the data transfer
EDMA3_DRV_enableTransfer ()	Enable the transfer on a particular logical channel
EDMA3_DRV_disableTransfer ()	Disable the transfer on a particular logical channel
EDMA3_DRV_bindPaRAMTcc ()	Bind the resources PaRAM Set and TCC
EDMA3_DRV_setQdmaTrigWord ()	Set the Trigger word for a QDMA channel
EDMA3_DRV_getPaRAM ()	Get the PaRAM Set associated with a logical channel
EDMA3_DRV_setPaRAM ()	Set the PaRAM Set associated with a logical channel
EDMA3_DRV_setPaRAMEntry ()	Set a particular PaRAM set entry of the PaRAM set associated with the specified logical channel
EDMA3_DRV_getPaRAMEntry ()	Get a particular PaRAM set entry of the PaRAM set associated with the specified logical channel
EDMA3_DRV_setPaRAMField ()	Set a particular PaRAM set field of the PaRAM set associated with the specified logical channel
EDMA3_DRV_getPaRAMField ()	Get a particular PaRAM set field of the PaRAM set associated with the specified logical channel
EDMA3_DRV_setEvtQPriority ()	Sets EDMA TC priority. User can program the priority of the Event Queues at a system-wide level. This means that the user can set the priority of an IO initiated by either of the TCs (Transfer Controllers) relative to IO initiated by the other bus masters on the device (ARM, DSP, USB, etc)
EDMA3_DRV_mapChToEvtQ ()	Associate Channel to Event Queue
EDMA3_DRV_getMapChToEvtQ ()	Get the Event Queue mapped to the specified DMA channel
EDMA3_DRV_setCCRegister ()	Set the Channel Controller (CC) Register value, by specifying the register offset and the new value.
EDMA3_DRV_getCCRegister ()	Get the Channel Controller (CC) Register value, by specifying the register offset.
EDMA3_DRV_waitAndClearTcc ()	Wait for a transfer completion interrupt to occur on the specific TCC.
EDMA3_DRV_checkAndClearTcc ()	Returns the status of a previously initiated transfer.
EDMA3_DRV_getPaRAMPhyAddr ()	Get the PaRAM Set Physical Address associated with a logical

	channel.
EDMA3_DRV_Ioctl ()	This function provides IOCTL functionality for EDMA3 Driver.
EDMA3_DRV_getInstHandle	Return the previously opened EDMA3 Driver Instance handle
EDMA3_DRV_disableLogicalChannel	Disable the event driven DMA channel or QDMA channel
EDMA3_DRV_registerTccCb	Registers a transfer completion handler for a specific DMA/QDMA channel
EDMA3_DRV_unregisterTccCb	Un-register the previously registered callback function against a DMA/QDMA channel
EDMA3_DRV_setTcErrorInt	Enable/disable specific EDMA3 Transfer Controller Interrupts
EDMA3_DRV_getChannelStatus	Get the current status of the DMA/QDMA channel
EDMA3_DRV_mapTccLinkCh	Associates a link channel and a TCC
EDMA3_DRVinitXbarEventMap ()	This function provides the functionality to map cross bar event to EDMA channels.

EDMA3 Driver Performance Characteristics

S No	Target	Code	Data (In bytes)		Total
		(In bytes)	Initialized	Uninitialized	(In bytes)
1	C64X+	21120	584	57146	78850
2	C67x	21280	576	57144	79000

Memory Usage Statistics for tda2xx M4 and C66 Core

	CODE (In bytes)	Data (In bytes)		Total (In bytes)
Library for M4		**Initialized**	**Uninitialized**	
edma3_lld_drv_sample.aem4	1468	13084	52	14604
edma3_lld_drv.aem4	9392	55	57140	66587
edma3_lld_rm.aem4	11704	11462	48364	71530
edma3_lld_rm_sample.aem4	1328	3236	52	4616
Library for DSP				
edma3_lld_drv_sample.ae66	2688	13120	12	15820
edma3_lld_drv.ae66	21280	576	57114	79000
edma3_lld_rm.ae66	27264	12523	48388	88175
edma3_lld_rm_sample.ae66	1376	40	8	1424
Library for EVE				
edma3_lld_drv_sample.aearp32f	2020	13131	52	15203
edma3_lld_drv.aearp32f	13208	593	57140	70941
edma3_lld_rm.aearp32f	15020	12363	48364	75747
edma3_lld_rm_sample.aearp32f	1920	3283	52	5255

References

[1] EDMA3 Module Hardware Specifications

[2] DSP BIOS Documentation

[3] EDMA3 Driver Documentation

Glossary

PaRAM Set Parameter RAM Set in an EDMA3 Controller

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information 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

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

LICENSE