A DSP/BIOS AIC23 Codec Device Driver for the TMS320DM642 EVM

Software Development Systems

ABSTRACT

This document describes the usage and design of a device driver for the AIC23 audio codec on the TMS320DM642 EVM. This device driver is written in conformance to the DSP/BIOS™ IOM device driver model and uses the generic TMS320C6x1x EDMA McASP driver to transfer samples to and from the serial port. For details on this generic driver, see the application note *A DSP/BIOS EDMA McASP Device Driver for TSM320C6x1x DSPs* (SPRA870).

Contents

List of Figures

List of Tables

1 Usage

The device driver described here is part of an IOM mini-driver. That is, it is implemented as the lower layer of a 2-layer device driver model. The upper layer is called the class driver and can be either the DSP/BIOS GIO, SIO/DIO, or PIP/PIO modules. The class driver provides an independent and generic set of APIs and services for a wide variety of mini-drivers and allows the application to use a common interface for I/O requests. Figure 1 shows the overall DSP/BIOS device driver architecture. For more information about the IOM device driver model as well as the GIO, SIO/DIO, and PIP/PIO modules, see the *DSP/BIOS Device Driver Developer's Guide* (SPRU616).

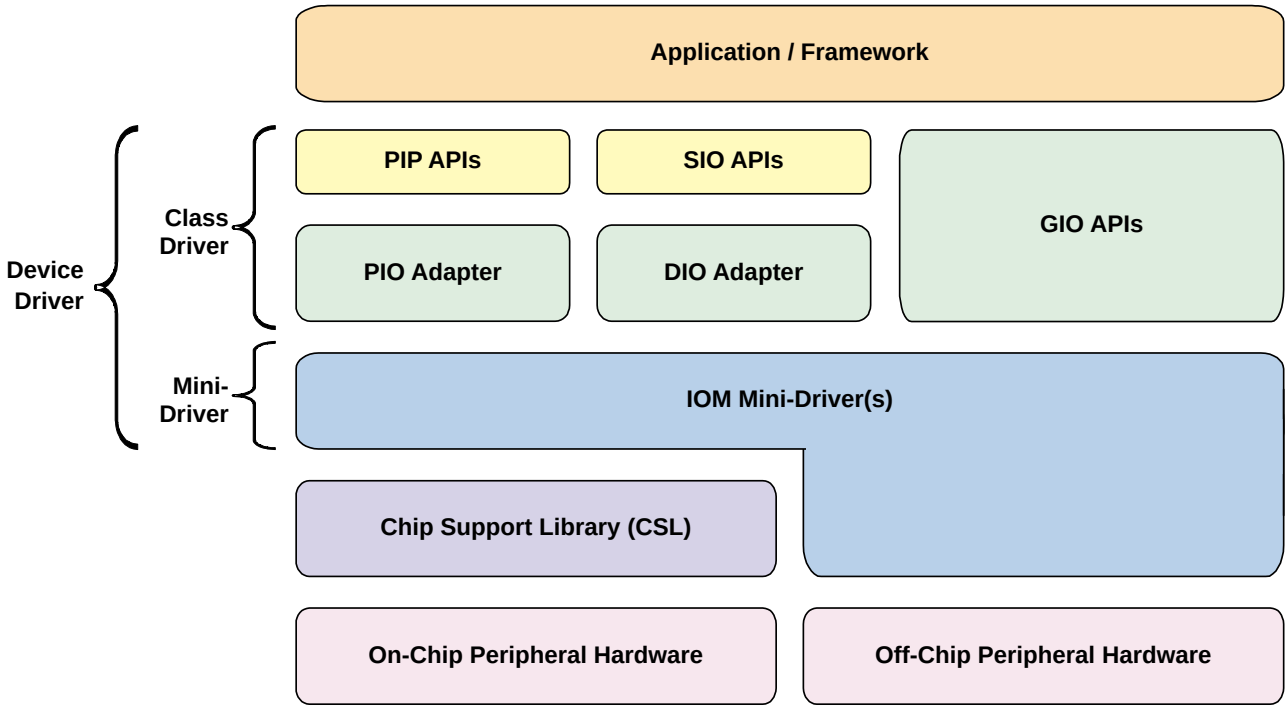

Figure 1. DSP/BIOS IOM Device Driver Model

Many mini-driver implementations split the code into a codec-specific portion and a generic portion that will work across many different codecs. Figure 2 shows the data flow between the components in a system in which the mini-driver is split into a generic part and a codec-specific part. This device driver uses the generic TMS320C6x1x EDMA McASP device driver to transfer samples to and from the serial port. This means that to use this device driver, an application must not only link with this device driver library (*evmdm642_edma_aic23.l64*), but also with the generic device driver library (*c6x1x_edma_mcasp.l64*). Other than this, the use of the generic device driver is hidden from the user. These device driver libraries are compiled for TMS320C641x, but can also be used with TMS320C671x. For example, if you are using TMS320C6713 and additional floating-point optimizations are needed, recompile the libraries for TMS320C6713. Note that this device driver uses McASP port 0.

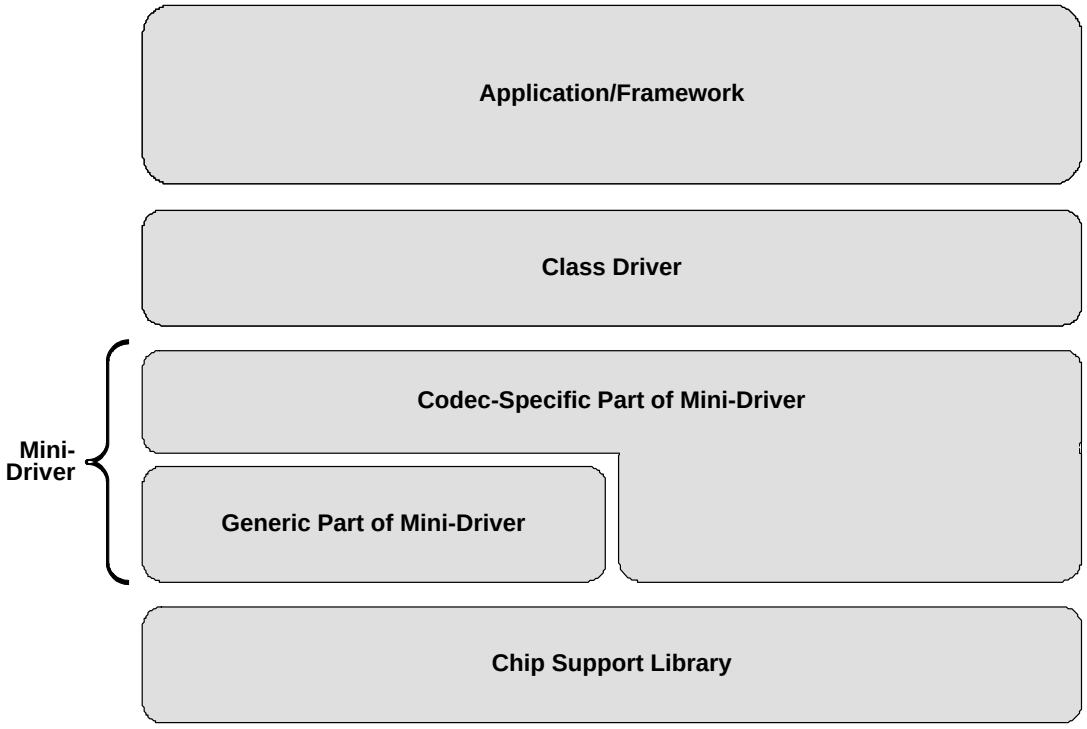

Figure 2. Codec Device Driver Partitioning

1.1 Configuration

To use this driver, a device entry has to be added and configured in the configuration tool. This device driver will set up the generic TMS320C6x1x EDMA McASP driver to meet its needs.

- **Init function:** Type _EVMDM642_EDMA_AIC23_init.
- **Function table ptr:** Type _EVMDM642_EDMA_AIC23_Fxns.
- **Function table type:** Select IOM_Fxns.
- **Device id:** This property is ignored by this device driver, since there is only one AIC23 codec on the TMS320DM642 EVM.
- **Device params ptr:** A pointer to your instance of the device parameter structure. Set this property to 0x0 to use the default parameters. The parameter structure and its defaults are described below.
- **Device global data ptr:** This property must be set to 0x0.

1.2 Device Parameters

```
typedef struct EVMDM642_EDMA_AIC23_DevParams {
    Int versionId;      /* Set to the version number used by the application */
    Bool cacheCalls;    /* Set to TRUE if buffers are in are in external memory */
    Uns enableClkg;     /* Set VALUE for internal clock generator */
    Uns enableHclkg;    /* Set VALUE for internal high frequency clock generator */
    Uns enableFsyncg;   /* Set VALUE for internal framesync generator */
    Int irqId;          /* IRQ number to use for EDMA interrupt */
    Int inEvtIrqId;     /* IRQ number used for McASP Event interrupt */
    Int outEvtIrqId;    /* IRQ number used for McASP Event interrupt */
    C6X1X_EDMA_MCASP_EvtCallback *evtCallback; /* Register events callback */
    Uns inEvtIntrMask;  /* Interrupt mask, set while executing input ISR */
    Uns outEvtIntrMask; /* Interrupt mask, set while executing output ISR */
    Uns edmaIntrMask;   /* Interrupt mask, set while executing edma ISR */
    AIC23_Params aic23Config; /* codec register setup */
} EVMDM642_EDMA_AIC23_DevParams;
```

- **versionId:** This parameter identifies the correct the version of the driver.
- **cacheCalls:** If this parameter is set to TRUE, the device driver will treat buffers issued to any IOM channel associated with the device as if they are in cacheable memory and the L2 data cache is enabled. The default value of this parameter is TRUE.
- **enableClkg, enableHclkg, enableFsyncg:** Those parameters select clock sources. Please see generic C6x1X_EDMA_MCASP driver manual (SPRA870) for parameter details.
- **irqId, inEvtIrqId, outEvtIrqId:** Those parameters select which IRQ number to use for the interrupts. Please see generic C6x1X_EDMA_MCASP driver manual (SPRA870) for parameter details.
- **evtCallback:** This parameter is a pointer to a event callback structure. If NULL, no event registered.
- **edmaIntrMask, inEvtIntrMask, outEvtIntrMask:** Those are interrupt masks used to specify which interrupts are disabled before calling ISR.
- **aic23Config:** The codec registers setup. If the device parameters pointer is NULL, the default parameters are used. Here are the default setups for the registers.
  - **Register 0:** Left input channel volume control. Default value is 0x0017.
  - **Register 1:** Right input channel volume control. Default value is 0x0017.
  - **Register 2:** Left channel headphone volume control. Default value is 0x01F9.
  - **Register 3:** Right channel headphone volume control. Default value is 0x01F9.
  - **Register 4:** Analog audio path control. Default value is 0x0011.
  - **Register 5:** Digital audio path control. Default value is 0x0000.
  - **Register 6:** Power down control. Default value is 0x0000.
  - **Register 7:** Digital audio interface format control. Default value is 0x0043.

- **Register 8:** Sample rate control. Default sample rate is 48Khz. Default value is 0x0002 for DM642 EVM production board (codec clock is 18.432MHz). If we use DM642 EVM beta board, the value should be changed to 0x0000 (codec clock frequency is 12.288MHz). To change the value, we can either modify evmdm642_edma_aic23.h or let the application pass the codec parameters to the driver.
- **Register 9:** Digital interface activation. Default value is 0x0001.

1.3 Channel Parameters

This driver does not have any channel parameters. Any values that are passed as channel parameters will be ignored (NULL is suggested).

1.4 Control Commands

This device driver has no run-time control commands.

2 Architecture

The codec specific portion of the mini-driver inherits the features of the generic TMS320C6x1x EDMA McASP driver. It uses two codec specific functions, mbBindDev() and mdCreateChan(), to do the DM642 EVM and AIC23 specific setup. These functions then call mbBindDev() and mdCreateChan() in the generic driver to complete generic portions of the driver initialization. The only thing the codec-specific part does is to set up the codec and leaves the transfers of samples to the generic device driver. The fact that this device driver uses the generic device driver is hidden from the user in all aspects except that the generic device driver library has to be linked into the application.

The function mdBindDev() is responsible for configuring the codec through the control channel based on the EVMDM642_EDMA_AIC23_DevParams structure that is passed in. It does some basic setup then calls a function called AIC23_setParams() function in aic23.c to do most of the real configuration work. The function mdCreateChan() generates the EDMA configuration used for the data transfers.

3 Constraints

- Inherits the constraints of the generic TMS320C6x1x EDMA McASP driver.
- AIC23 codec setting is done through I2C. The driver assumes the application will call BSL function “EVMDM642_init” in GBL user init function to open I2C port and export a handle. The driver doesn’t open the I2C port nor close the port and the I2C handle is shared with other drivers.

4 References

All these documents are available on the TI Developer’s Village.

1. *A DSP/BIOS EDMA McASP Device Driver for TMS320C6x1x DSPs* (SPRA870)
2. *TLV320AIC23 Stereo Audio Codec, 8 to 96 KHz, With Integrated Headphone Amplifier Data Manual*, SLWS106D
3. *DSP/BIOS Driver Developer’s Guide* (SPRU616)
4. *TMS320C6000 Chip Support Library API Reference Guide* (SPRU401)
5. *TMS320C6000 Peripherals Reference Guide* (SPRU190)
6. *TMS320C6000 DSP/BIOS Application Programming Interface (API) Reference Guide* (SPRU403)

Appendix A Device Driver Data Sheet

A.1 Device Driver Library Name

evmdm642_edma_aic23.l64

When building an application the generic c6x1x_edma_mcasp.l64 library is required.

A.2 DSP/BIOS Modules Used

Same as for the generic TMS320C6x1x EDMA McASP device driver.

A.3 DSP/BIOS Objects Used

Same as for the generic TMS320C6x1x EDMA McASP device driver.

A.4 CSL Modules Used

Same as for the generic TMS320C6x1x EDMA McASP device driver.

A.5 CPU Interrupts Used

Same as for the generic TMS320C6x1x EDMA McASP device driver.

A.6 Peripherals Used

Same as for the generic TMS320C6x1x EDMA McASP device driver.

A.7 Maximum Interrupt Latency

Same as for the generic TMS320C6x1x EDMA McASP device driver.

A.8 Memory Usage

Includes the memory usage of the generic TMS320C6x1x EDMA McASP device driver.

Table A–1. Device Driver Memory Usage

	Uninitialized memory	Initialized memory
CODE	—	376 words
DATA	38 words	92 words

NOTE: This data was gathered using the sectti command utility.
Uninitialized data: .bss
Initialized data: .cinit + .const
Initialized code: .text + .text:init

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. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

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.

Reproduction of information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		**Applications**	
Amplifiers	amplifier.ti.com	Audio	www.ti.com/audio
Data Converters	dataconverter.ti.com	Automotive	www.ti.com/automotive
DSP	dsp.ti.com	Broadband	www.ti.com/broadband
Interface	interface.ti.com	Digital Control	www.ti.com/digitalcontrol
Logic	logic.ti.com	Military	www.ti.com/military
Power Mgmt	power.ti.com	Optical Networking	www.ti.com/opticalnetwork
Microcontrollers	microcontroller.ti.com	Security	www.ti.com/security
		Telephony	www.ti.com/telephony
		Video & Imaging	www.ti.com/video
		Wireless	www.ti.com/wireless

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