xDAIS-DM (Digital Media) User Guide

Literature Number: SPRUEC8A
October 2006

Texas Instruments

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
Low Power Wireless	www.ti.com/lpw	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

Preface

About This Manual

This document describes the xDAIS-DM (Digital Media) standard that is built over TI's well proven eXpress DSP Algorithm Interoperability Standard (also known as xDAIS) specification.

This document refers to xDAIS-DM as xDM. The terms xDAIS-DM and xDM are used interchangeably in this document.

xDM defines a uniform set of APIs across various multimedia algorithms to ease integration and ensure interoperability.

Intended Audience

This document assumes that you are fluent in the C language, have a good working knowledge of digital signal processing and the requirements of DSP applications, and have had some exposure to the principles and practices of object-oriented programming. This document describes the interfaces between algorithms and the applications that utilize these algorithms. System integrators will see how to incorporate multiple algorithms from separate developers into a complete system. Algorithm writers learn how to ensure that an algorithm can coexist with other algorithms in a single system and how to package an algorithm for deployment into a wide variety of systems.

How to Use This Manual

This document contains the following chapters:

- **Chapter 1 – Overview**, provides an overview of the xDM standard and the generic interfaces defined by it.
- **Chapter 2 – Algorithm Interfaces**, contains the algorithm interfaces that are defined by xDM.

Additional Documents and Resources

The following online system provides reference information about the API used to create xDM-compatible algorithms:

- *xDM API Reference.* XDAIS_INSTALL_DIR/docs/html/index.html

You can use the following books to supplement this user's guide:

- *TMS320 DSP Algorithm Standard Rules and Guidelines* (SPRU352)
- *TMS320 DSP Algorithm Standard API Reference* (SPRU360)
- *TMS320 DSP Algorithm Standard Developer's Guide* (SPRU424)
- *TMS320 DSP Algorithm Standard Demonstration Application* (SPRU361)

Text Conventions

The following conventions are used in this specification:

- Text inside back-quotes (‘‘) represents pseudo-code.
- Program source code, function and macro names, parameters, and command line commands are shown in a `mono-spaced font`.

Contents

Figures

Chapter 1

Overview

This chapter provides an overview of the xDM standard.

Topic	Page

The xDM standard defines a uniform set of APIs across various multimedia codecs to ease integration and ensure interoperability. xDM is built over TI's well proven eXpress DSP Algorithm Interoperability Standard (also known as xDAIS) specification.

1.1 Scope of the Standard

xDM addresses the following:

- Uniform lightweight APIs across various classes of multimedia algorithms, such as audio, video, speech, and image
- Flexibility of extension for various requirements such as metadata parsing, file format, custom processing, and so forth
- Interoperability across various algorithms and vendors

xDM does not address the following:

- Metadata parsing from multimedia streams
- File format or multiplex support
- Digital Rights Managements (DRM) interaction with codecs
- Call back from algorithms and applications to enable data movement and processing
- APIs other than codecs, for example, pre- and post-processing APIs like resizing, echo cancellations, and so forth

These features will be addressed in later versions of xDM (version 1.x, 2.x). The features are not standardized in xDM v1.0, the basic definition can be extended to support above mentioned features.

1.2 Goals of the Standard

The goals of this standard include:

- Enable plug and play architecture for multimedia codecs across various classes of algorithms and vendors.
- Enable faster time to market for multimedia products such as, digital cameras, cell phones, set-top boxes, and portable multimedia players.
- Provide a standard interface based on given class of multimedia codecs (for example, audio, video, image, and speech).
- Define common status and parameters based on given class of multimedia codecs.
- Flexibility of extension of custom functionality.
- Low overhead of interface.
- Reduce integration time for system developers.

1.3 Differences Between xDM 0.9 and 1.0 Beta

This manual describes the general concepts of the xDM interface. The current version of xDM supported is 0.9, which was introduced in xDAIS 5.00. Since that release, TI has worked with customers and third parties to improve the interface, and are including the xDM 1.0 beta interfaces in this xDAIS 5.10 release. This xDM 1.0 interface will continue to be refined, and will be released in its final version in a future release.

For details about differences between xDM versions 0.9 and 1.0 beta, see XDAIS_INSTALL_DIR/packages/ti/xdais/dm/doc/xdm1_differences.pdf. The following list provides a summary of the changes that are likely to be needed to your xDM-compliant algorithms:

- Modify the prefixes by adding a 1 to use the new encoder and decoder interfaces. For example, change all occurrences of ISPHDEC_ to ISPHDEC1_. This is true for all eight generic interfaces.
- For speech encoders and decoders, include ispeech1.h instead of ispeech.h. In addition, include the appropriate codec family-specific header file. For example, ispeech_pcm.h or ispeech_amr.h.
- For speech encoders and decoders, modify the sizes of the structure fields to match the new sizes. These sizes have been decreased to improve channel density.
- Modify the fields of various structures used by the encode and decode interfaces. For example, most of the *_Params structures have changed. A detailed list of these changes for each encoder and decoder is provided in the xdm1_differences.pdf file.
- Take advantage of new features described in the xdm1_differences.pdf file. For example, new error values, new enumerated types, support for allowing applications to query the codec version, special handling for single buffers.

1.4 Relationship Between xDM and xDAIS

xDM is an extension to the IALG interface standard. It defines and standardizes interfaces for multimedia codecs. The relationship between xDM and xDAIS is depicted in Figure 1-1.

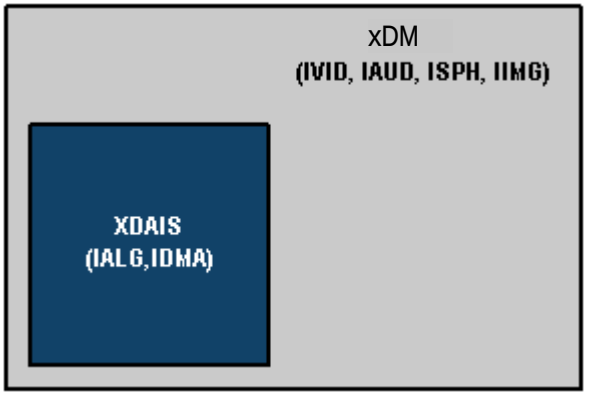

Figure 1-1. Relationship between xDM and xDAIS.

As shown in the figure, xDM is a superset of the xDAIS standard. xDM defines eight generic interfaces for the following categories. The "x" suffix represents a version of the interface. In xDM 0.9, the suffix was omitted; in xDM 1.0, it will be "1".

- IVIDENCx - Generic interface for video encoders
- IVIDDECx - Generic interface for video decoders
- IAUDENCx - Generic interface for audio encoders
- IAUDDECx - Generic interface for audio decoders
- ISPHENCx - Generic interface for speech encoders
- ISPHDECx - Generic interface for speech decoders
- IIMGENCx - Generic interface for image encoders
- IIMGDECx - Generic interface for image decoders

1.4.1 xDAIS Interfaces

The current xDAIS interface from algorithm to application is depicted in Figure 1-2.

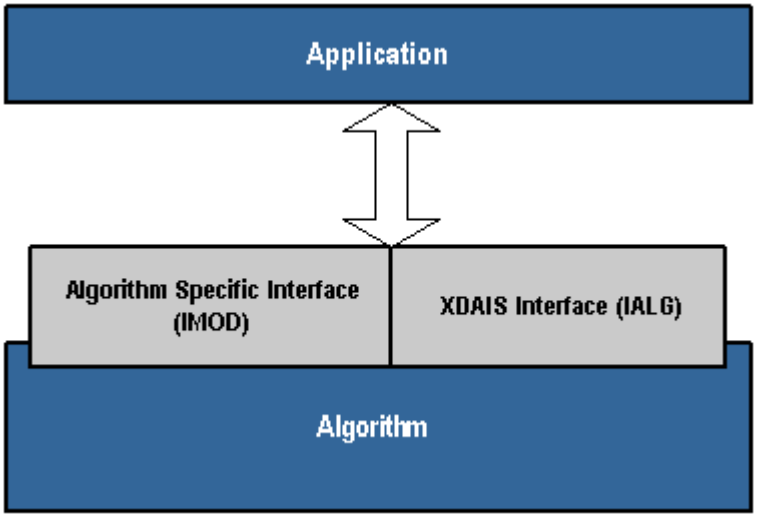

Figure 1-2. Current xDAIS Interface to Codec Library

As per xDAIS, the given algorithm extends the standard IALG interface to the IMOD interface (algorithm specific). The IMOD interface provides basic functionality of the algorithm, while the IALG interface takes care of the memory management. xDAIS does not define the IMOD interface. The algorithm implementer must define the IMOD interface based on his requirements. For example, in case of MP3, the algorithm will implement the IMP3 interface. This interface is kept totally open.

The application talks to the codec library via the IMOD interface. Optionally, an application can directly talk to the codec library via the MOD interface, which provides high level functionality.

The xDM standard proposes a new interface as depicted in Figure 1-3.

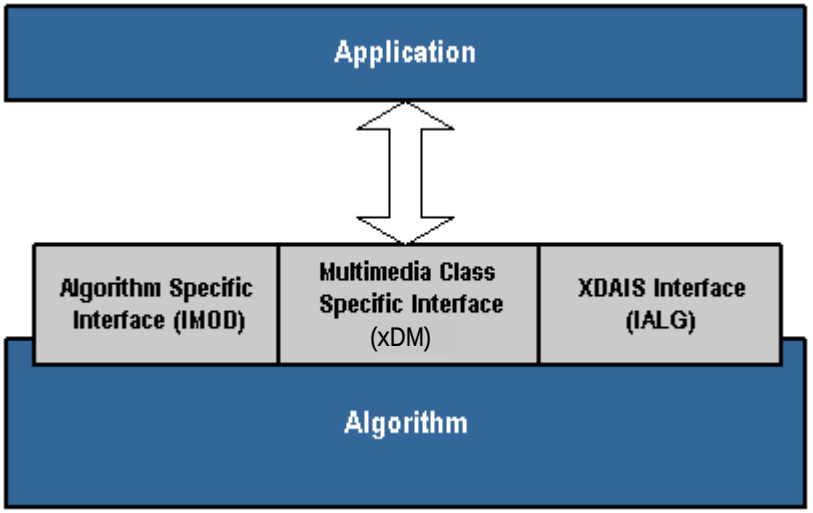

Figure 1-3. Proposed Interface to Codec Library

The algorithm uses one of eight predefined standard interface called xDM. This interface is a superset of the IALG interface. You can tailor a given algorithm or implementation by extending the xDM interface to the IMOD interface. In simple cases, IMOD will be identical to xDM. Optionally, the algorithm can add more functionality to the xDM interface to define the IMOD interface. In this case, the application talks to the codec library via the IMOD interface as done previously.

Chapter 2

Algorithm Interfaces

This chapter describes all the algorithm interfaces that are defined by the xDM standard. Reference information about the interfaces is provided at XDAIS_INSTALL_DIR/docs/html/index.html.

Topic **Page**

2.1 IALG Interface

IALG interface defines a framework independent interface for the creation of algorithm instance objects. For more information on the IALG interface, see *TMS320 DSP Algorithm Standard API Reference*.

2.2 xDM Interface

The xDM interface consists of eight interfaces to tailor to various multimedia algorithms (for example, audio, video, encoders, decoders, and so forth). Reference information about these interfaces is provided at in an online reference system at XDAIS_INSTALL_DIR/docs/html/index.html.

These new interfaces define two new functions:

- `process()`
- `control()`

The following figure summarizes the valid sequences of execution of the functions for a particular algorithm instance.

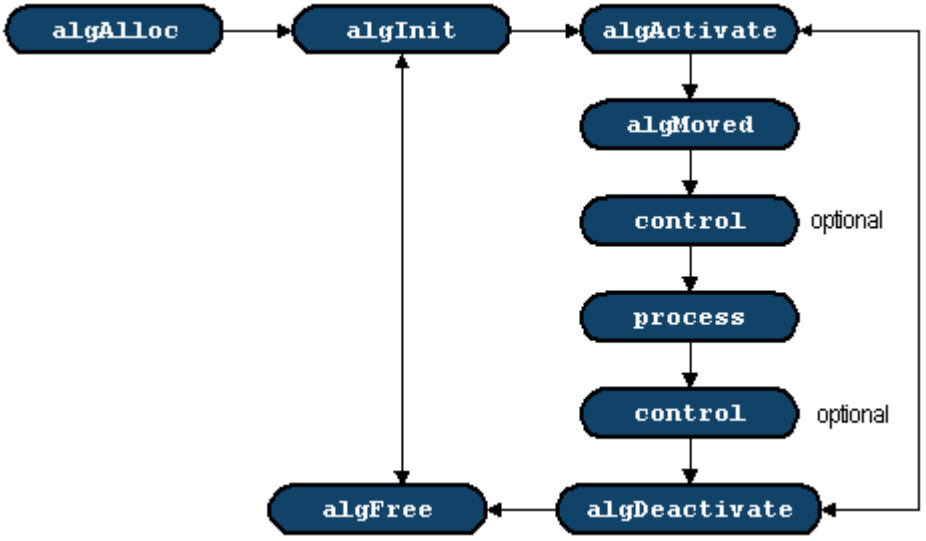

Figure 2-1. xDM Interface Function Call Order.

The xDM interface function call order is similar to the IALG interface function call order, except that xDM removes usage of the `algControl()` method. Hence when xDAIS is used along with xDM, instead of using the `algControl()` method (which is part of the IALG interface), the algorithm developer must use the `control()` method (which is part of the xDM interface).

2.3 XDM and IMOD Interface

You can tailor a given algorithm or implementation by extending the xDM interface to the IMOD interface. In simple cases, IMOD will be identical to xDM. Optionally, the algorithm can add more functionality to the xDM interface to define the IMOD interface as shown in Figure 2-2.

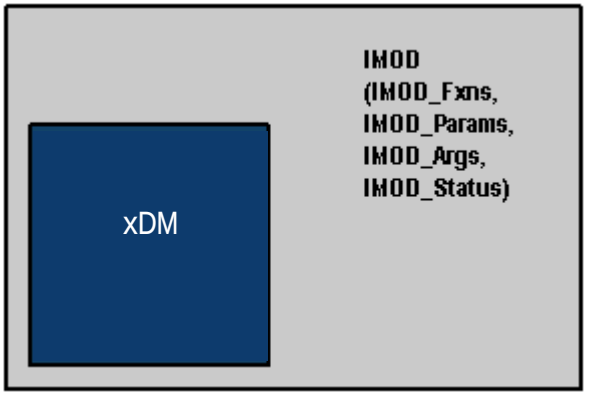

Figure 2-2. XDM and IMOD Interfaces

The relationship between the xDM and IMOD interfaces are as follows:

- IMOD_Fxns = XDM_Fxns + extension functions
- IMOD_Params = XDM_Params (consist of creation and run time) + extension parameters (creation and runtime)
- IMOD_Args = XDM_Args + extension arguments
- IMOD_Status = XDM_Status (consist of common status) + extension status (consist of detail status)

Note that the fields in most of these structures changed from xDM version 0.9 to 1.0 beta. See Section 1.3, Differences Between xDM 0.9 and 1.0 Beta, for more information.

Chapter 3

Glossary

Algorithm: Technically, an algorithm is a sequence of operations, each chosen from a finite set of well-defined operations (for example, computer instructions), that halts in a finite time, and computes a mathematical function.

API: Acronym for application programming interface. A specific set of constants, types, variables, and functions used to programmatically interact with a piece of software.

Endian: Refers to which bytes are most significant in multi-byte data types. In big-endian architectures, the leftmost bytes (those with a lower address) are most significant. In little-endian architectures, the rightmost bytes are most significant. HP, IBM, Motorola 68000, and SPARC systems store multi-byte values in big-endian order, while Intel 80x86, DEC VAX, and DEC Alpha systems store them in little-endian order. Internet standard byte ordering is also big-endian. The TMS320C6000 is bi-endian because it supports both systems.

Framework: Part of an application that is designed to remain invariant while selected software components are added, removed, or modified. Very general frameworks are sometimes described as application-specific operating systems.

Interface: A set of related functions, types, constants, and variables. An interface is often specified with a C header file.

Method: A synonym for a function that is part of an interface.

Module: A module is an implementation of one (or more) interfaces. In addition, all modules follow certain design elements that are common to all XDM compatible software components. Roughly speaking, a module is a C language implementation of a C++ class.