Table of Contents

What's New In xDM

Introduction

xDM 0.9 was introduced in the xDAIS 5.00 release. xDAIS 5.10 introduced the xDM 1.00 Beta interfaces for review. xDAIS 5.21 delivers the final version of xDM 1.00, integrating feedback collected from customers during the beta period. This document provides an overview of those changes.

Note! The xDM 0.9 is deprecated, and will be eventually removed - likely in early 2008 once the xDM 1.00 interfaces have gained adoption.

General xDM 1.x Changes

- The xDM 1.x interfaces are provided *in addition to* the previously released xDM 0.9 interfaces. Both 0.9 and 1.x interfaces are provided in the `ti.xdais.dm` package.
    - To enable both xDM 0.9 and 1.x compliant algorithms/frameworks/apps to reside in a system, unique names were created as new interfaces were developed. For example, in 0.9, video decoders used the `IVIDENC_` prefix; in xDM 1.x, they use the `IVIDENC1_` prefix. In this way, there is no naming collision, and the two classes can coexist in a single application.
    - In fact, if an algorithm provider so wished, the algorithm could implement *both* interfaces. It could provide and document two IALG function tables, one complying with the 0.9 interface, and the other complying with the 1.x interface(s). The system integrator could then chose which one to instantiate, perhaps based on which interface the application calling the algorithm was written to.
    - Note that when xDM 1.x changes are backward compatible with the 0.9 spec (e.g. addition of enum values), the prefix generally remained. For example, this was the case with ivideo.h - there is no ivideo1.h as all changes were backward compatible.
- To improve channel density, the sizes of many of the speech structure fields have decreased. Only the speech interfaces were compressed as those interfaces are often uses in very dense systems.
    - Note that the structures were padded when necessary to preserve 32-bit alignment. This is necessary because many of these structures support extended arguments immediately following the base structures; those fields must be 32-bit aligned.
- For improved performance of codec classes which don't require multiple buffers (i.e. `XDM_BufDesc`), `XDM_SingleBufDesc` was introduced.
    - Also, to better enable applications and frameworks integrating xDM algorithms, `XDM1_BufDesc` and `XDM1_SingleBufDesc` were introduced. These buffer descriptors include a per-buffer "access mask" indicating *how* the algorithm accessed that buffer. A typical use case for this flag is that applications can now identify when an algorithm filled a buffer using DMA rather than CPU access, and avoid unnecessary cache writeback-related maintenance of that buffer. Previously, the application had to "know" the behavior of the algorithm, or behave pessimistically, and assume there may be dirty cache write lines.
- Introduced `XDM_EUNSUPPORTED` error value (as a peer to `XDM_EOK` and `XDM_EFAIL`). This is reflected in the 1.x classes as well (e.g. `ISPHDEC1_EUNSUPPORTED`), but not back-ported to 0.9 interfaces.
    - `XDM_ERUNTIME` was removed. As the 0.9 interface included this definition, it is available for backward compatibility in xdm.h, but the including source must `#define XDM_INCLUDE_DOT9_SUPPORT` to have `XDM_ERUNTIME` defined.
- Introduced `XDM_CmdId.XDM_GETVERSION`. Applications can issue this command to algorithms which support this to obtain a string describing the version of the codec.
- Introduced a `data` field of type `XDM1_SingleBufDesc` to **all** `*1_Status` structures. This will be used to support the `XDM_GETVERSION` command, as well as enable the ability to pass arbitrary buffers of data between the codec and application in `control()` calls.

    - This is an IN/OUT buffer. That is, it can be used to both provide information to the algorithm, as well as receive information from it. Consult each algorithm's documentation for further details if they utilize this buffer.
- Video interfaces were updated to be "transcode friendly". In general, this resulted in the addition of macro-block data support.

Speech

General

- Many `enum` 's were added, providing identifiers to use in the various structure fields.
    - Note, the xDM structure fields themselves are not defined using these `enum` data types in order to minimize structure sizes. Users of these `enum` values are encouraged **not** to use these `enum` data types for storage, but rather only for constants - consistent with xDM 0.9 usage.
- 0.9's `ISPEECH_CompoundLaw` data type is supplied in 1.x's `ISPEECH1_CompandingLaw`
- Fields were broken out into either generic (all speech codecs) or specific (some speech codecs), and the bulk of the common header file in 0.9 (i.e. `ispeech.h`) was separated into a generic (i.e. `ispeech1.h`) and a suite of codec family-specific headers (e.g. `ispeech_pcm.h`, `ispeech_amr.h`, etc). A key benefit of this approach is that future codec families can be introduced without modifying existing headers. This table shows the different fields and classifies them as generic, or which families support them.

Field	Generic (ispeech1.h)	AMR	EVRC	G726	PCM	G723	SMV	WBAMR
CompandingLaw				X	X			
VADSelect		X					X	
VADFlag	X							
PostFilter	X							
NoisePreProc			X			X	X	
TTYMode			X				X	
DTMFMode							X	
DataMode							X	
NullTrafficMode	X							
PackingType		X		X				X
CodecSelect		X						
BitRate		X		X		X		X
Mode			X				X	

Decode

- Introduced new module prefix - `ISPHDEC1`
- Added `ISPHDEC1_FrameType`
- `ISPHDEC1_Params`
    - Removed `dataEnable`
    - Added `codecSelection`, `bitRate` and `tablesPtr`
- `ISPHDEC1_InArgs`
    - Removed `inBufferSize` and `bfiFlag`
    - Added `data` XDM1_SingleBufDesc for SMV codec support - which allows "out of band" data to fill this buffer
- `ISPHDEC1_Status`
    - Removed `dataMode`
    - Added `data`, `compandingLaw`, `packingType`, `codecSelection` and `bitRate`

- ISPHDEC1_OutArgs
  - Removed mode
  - Added extendedError and dataSize
- process()
  - XDM_BufDesc inBufs replaced by XDM1_SingleBufDesc inBuf
  - XDM_BufDesc outBufs replaced by XDM1_SingleBufDesc outBuf

Encode

- Introduced new module prefix - ISPHENC1
- Introduced ISPHENC1_ENOOUTPUT
- Added ISPHENC1_FrameType
- ISPHENC1_Params
  - Added codecSelection, bitRate and tablesPtr
- ISPHENC1_InArgs
  - Added data XDM1_SingleBufDesc for SMV codec support - which allows "out of band" data to be passed in
- ISPHENC1_OutArgs
  - Removed rate and outbufferSize
  - Added extendedError
- ISPHENC1_DynamicParams
  - Removed homingMode
- ISPHENC1_Status
  - Removed dtmfMode and homingMode
  - Added data, compandingLaw, packingType, vadSelection and codecSelection
- process()
  - XDM_BufDesc inBufs replaced by XDM1_SingleBufDesc inBuf
  - XDM_BufDesc outBufs replaced by XDM1_SingleBufDesc outBuf

Video

General

- IVIDDEC1 was introduced with many enhancements, but unfortunately after its release was found to have a critical gap - the max number of video buffers returned by a single process() call was limited to 16, and some codec classes require 17. This was addressed by making a small change to IVIDDEC1, and introducing IVIDDEC2. To that end, users are *strongly* encouraged to use IVIDDEC2 rather than IVIDDEC1.
- Addition of several values to IVIDEO_FrameSkip enum to better enable trick play.
- Addition of several values to IVIDEO_FrameType enum to support more frame types.
- Added IVIDEO_ContentType.IVIDEO_CONTENTTYPE_NA enum value for when the content type is not available.
- Added IVIDEO_OutputFrameStatus enum to better indicate buffer ownership.
- Added IVIDEO1_BufDesc and IVIDEO1_BufDescIn to enable more detailed data about video buffers to be conveyed in various APIs.

Decode

- Introduced new module prefix - IVIDDEC2 (and IVIDDEC1, though users are strongly encouraged to use IVIDDEC2)
- IVIDDEC2_OutArgs

    - Moved `extendedError` from `IVIDDEC2_OutArgs` into the `IVIDEO1_BufDesc` type. This enables the application to better identify errors in the decoded buffers and errors in the display buffers.
    - Moved `decodedFrameType` into the `IVIDEO1_BufDesc` type to enable per-buffer frame types.
    - Changed `outputID` from a single ID to an array of IDs, to enable multiple decoded buffers to be returned in a single `process()` call.
    - Similarly, changed `displayBufs` from a single struct to an array of structs.
    - Added `decodedBufs`
    - Added `outputMbDataID` and `mbDataBuf` to enable MB data generation for each process() call.
    - Added `freeBufID` array to identify buffers that are "unlocked" by the algorithm, and can finally be recycled by the application.
    - Added `outBufsInUseFlag`
- `IVIDDEC2_DynamicParams`
    - Added `frameOrder` field, and associated `IVIDDEC2_FrameOrder` enum, to enable the application to specify how the codec should deliver decoded frames.
    - Added `newFrameFlag` field to indicate to the algorithm that it should start a new frame.
    - Added `mbDataFlag` field to indicate to the algorithm that it should generate MB data.
- `IVIDDEC2_Status`
    - Added `data`

Encode

- Introduced new module prefix - `IVIDENC1`
- `IVIDENC1_Params`
    - Added `reconChromaFormat` field to enable the application to specify the data format the algorithm should present the reconstruction buffers in.
- `IVIDENC1_InArgs`
    - Added `inputID` and `topFieldFirstFlag` fields
- `IVIDENC1_OutArgs`
    - Added `outputID` and `encodedBuf` fields
    - Modified the data type of `reconBufs` to the more descriptive `IVIDEO1_BufDesc`
- `IVIDENC1_DynamicParams`
    - Removed `forceIFrame` field
    - Added `forceFrame`, `interFrameInterval` and `mbDataFlag` fields
- `IVIDENC1_Status`
    - Added `data` field
- `process()`
    - `XDM_BufDesc inBufs` replaced by `IVIDEO1_BufDescIn inBuf`

Audio

Decode

- Introduced new module prefix - `IAUDDEC1`
- `IAUDDEC1_Params`
    - Removed `maxSampleRate`, `maxBitRate`, `maxNoOfCh` fields.
    - Added `outputPCMWidth` and `pcmFormat` fields.
- `IAUDDEC1_InArgs`
    - Added `desiredChannelMode` and `lfeFlag` fields.
- `IAUDDEC1_OutArgs`
    - Added `numSamples`, `channelMode`, `lfeFlag`, `dualMonoMode` and `sampleRate`

fields.

- IAUDDEC1_DynamicParams
  - Removed outputFormat field.
  - Added downSampleSbrFlag field.
- IAUDDEC1_Status
  - Removed numChannels, numLFEChannels, autoPosition, fastFwdLen, and frameLen fields.
  - Added data, validFlag, lfeFlag, channelMode, numSamples, and dualMonoMode fields.
  - Renamed outputFormat field to pcmFormat.

Encode

- Introduced new module prefix - IAUDENC1
- IAUDENC1_Params
  - Removed encodingPreset, maxSampleRate, and maxNoOfCh fields.
  - Added sampleRate, bitRate, channelMode, encMode, inputFormat, inputBitsPerSample, dualMonoMode, crcFlag, ancFlag, and lfeFlag fields.
  - Renamed maxBitrate to maxBitRate.
- IAUDENC1_InArgs
  - Added numInSamples and ancData fields.
- IAUDENC1_OutArgs
  - Added numZeroesPadded and numInSamples fields.
- IAUDENC1_DynamicParams
  - Removed inputFormat, numChannels, and numLFEChannels fields.
  - Added channelMode, lfeFlag, and dualMonoMode fields.
- IAUDENC1_Status
  - Removed frameLen field.
  - Added data, validFlag, lfeFlag, bitRate, sampleRate, channelMode, and encMode fields.

Image

Decode

- Introduced new module prefix - IIMGDEC1
- IIMGDEC1_Status
  - Renamed outChromaformat to outputChromaFormat
  - Added data field
- IIMGDEC1_OutArgs
  - Renamed bytesconsumed to bytesConsumed

Encode

- Introduced new module prefix - IIMGENC1
- IIMGENC1_Status
  - Added data field

See Also

- TI eXpressDSP wiki:
  - All articles in the xDAIS Category
  - xDM FAQ

- Updating to a new xDM version