Media SDK Guide

Display Encode Mode (DEM) API Reference

1 Introduction

This document describes the AMD Media Framework (AMF) Display Encode Mode (DEM) interface. The AMF DEM interface allows an Independent Software Vendor (ISV) to encode the contents of the display (audio and video) for streaming them to a remote device.

The following section contains an overview of the key data types and interfaces of the AMD Media Framework.

2 AMD Media Framework (AMF)

2.1 Elementary data types

All elementary data types are defined to make the code potentially portable to other operating systems. Examples are as follows:

```
typedef __int64 amf_int64;
typedef __int32 amf_int32;
typedef size_t amf_size;
typedef void* amf_handle;
```

2.2 C++ Interfaces

The interfaces will be implemented in the form of abstract C++ classes (interfaces). Each interface will be derived from the `AMFInterface` basic interface, which exposes two reference counting methods and a query interface method:

```
class AMFInterface
{
public:
AMF_DECLARE_IID("AMFUnknown") // interface definition
virtual amf_long AMF_STD_CALL Acquire()=0;
virtual amf_long AMF_STD_CALL Release()=0;
virtual AMF_ERROR AMF_STD_CALL QueryInterface(AMFInterfaceIDType
interfaceID,void**ppInterface)=0;
};
```

The AMF SDK will provide a default implementation for `AMFInterface` with self-destroying behavior. The SDK will also provide a smart pointer template class for easy interface manipulations.

2.3 Property storage

Each object in AMF implements AMFPropertyStorage or AMFPropertyStorageEx. AMFPropertyStorage implements a property map with a string as an ID and the AMFVARIANT structure as data. A set of helper classes and template functions help in dealing with AMFVARAINT in C++ in a safe manner.

```
class AMFPropertyStorage : virtual public AMFInterface
{
public:
AMF_DECLARE_IID("AMFPropertyStorage")
virtual AMF_ERROR AMF_STD_CALL SetProperty(const wchar_t *pName, AMFVARIANT
value) = 0;
virtual AMF_ERROR AMF_STD_CALL GetProperty(const wchar_t *pName, AMFVARIANT*
pValue) = 0;
template<typename _T> AMF_ERROR AMF_STD_CALL SetProperty(const wchar_t *pName,
const _T& value);
template<typename _T> AMF_ERROR AMF_STD_CALL GetProperty(const wchar_t *pName,
_T* pValue);
virtual bool AMF_STD_CALL HasProperty(const wchar_t *pName) = 0;
virtual AMF_ERROR AMF_STD_CALL Clear() = 0;
virtual AMF_ERROR AMF_STD_CALL AddTo(AMFPropertyStorage *dest, bool bOverwrite,
bool bDeep) = 0;
virtual AMF_ERROR AMF_STD_CALL CopyTo(AMFPropertyStorage *dest, bool bDeep) = 0;
};
AMFPropertyStorageEx adds property description and validation features to
AMFPropertyStorage:
class AMFPropertyStorageEx : virtual public AMFPropertyStorage
{
public:
AMF_DECLARE_IID("AMFPropertyStorageEx")
virtual amf_size AMF_STD_CALL GetPropertiesInfoCount() = 0;
virtual AMF_ERROR AMF_STD_CALL GetPropertyInfo(amf_size szInd,const
AMFPropertyInfo** pInfo) = 0;
virtual AMF_ERROR AMF_STD_CALL GetPropertyInfo(const wchar_t *pName,const
AMFPropertyInfo** pInfo)=0;
virtual AMF_ERROR AMF_STD_CALL ValidateProperty(const wchar_t *pName, AMFVARIANT
val, AMFVARIANT* outValidated) = 0;
};
```

2.4 AMF DEM Buffer Interface

The encoded video elementary bitstream, audio data, or MPEG2 transport stream data are returned as `AMFDemBuffer`. The class looks like:

```
class AMFDemBuffer: public virtual AMFInterface
{
public:
AMF_DECLARE_IID("AMFDemBuffer")
virtual AMF_ERROR AMF_STD_CALL GetMemory(void **ppMem)=0;
virtual DemOutputType AMF_STD_CALL GetDataType()=0;
virtual amf_size AMF_STD_CALL GetMemorySize()=0;
virtual amf_int64 AMF_STD_CALL GetTimeStamp()=0;
};
```

The AMFDemBuffer provides some description of the data returned, in addition to the encoded data.

The returned data can be one of the following:

```
DEM_AV_TS: Audio and video transport stream
DEM_AV_ES: Audio and video elementary stream
DEM_V_TS: Video transport stream
DEM_V_ES: Video elementary stream
DEM_A_ES: Audio elementary stream
```

The GetTimeStamp() method can be used to determine the timestamps for audio and video synchronization. Timestamps are returned as the number of clock ticks from a 90KHz timer.

2.5 AMF VCEDEM Encoder Interface

The AMFEncoderVCEDEM is the main interface for interacting with the Encoder. It derives from the AMFPropertyStorageEx interface, which can be used to configure specific encoding parameters.

```
class AMFEncoderVCEDEM: public virtual AMFPropertyStorageEx
{
public:
AMF_DECLARE_IID("AMFEncoderVCEDEM")
// Initialization and termination
virtual AMF_ERROR AMF_STD_CALL AcquireRemoteDisplay()=0;
virtual AMF_ERROR AMF_STD_CALL CreateEncoder()=0;
virtual AMF_ERROR AMF_STD_CALL StartEncoding()=0;
virtual AMF_ERROR AMF_STD_CALL StopEncoding()=0;
virtual AMF_ERROR AMF_STD_CALL DestroyEncoder()=0;
virtual AMF_ERROR AMF_STD_CALL ReleaseRemoteDisplay()=0;

// Buffer retrieval
virtual AMF_ERROR AMF_STD_CALL GetNextFrame(AMFDemBuffer** buff)=0;

// This method must be called from another thread
// to terminate waiting inside GetNextThread
virtual AMF_ERROR AMF_STD_CALL CancelGetNextFrame()=0;

// This method flushes configuration to encoder
virtual AMF_ERROR AMF_STD_CALL FlushConfiguration()=0;
};
```

Figure 1 illustrates the high-level sequence for using DEM.

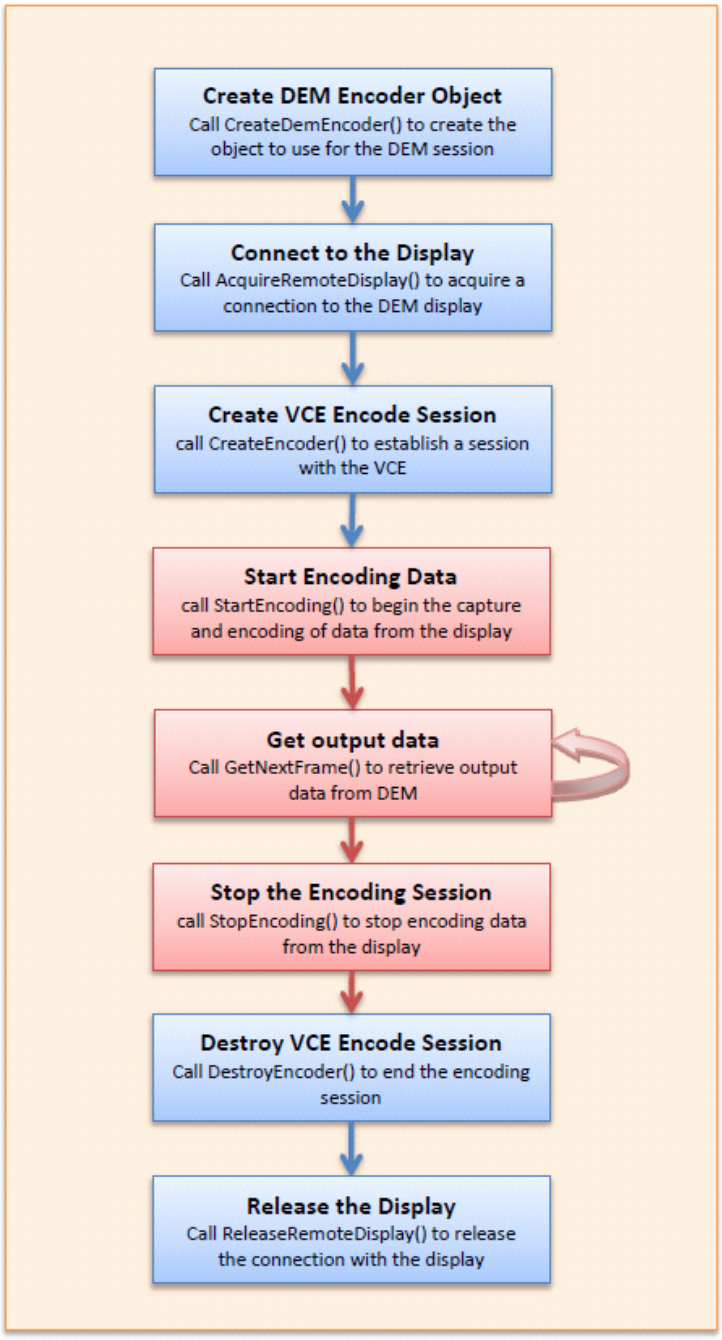

Figure 1 High-level sequence for using DEM

3 Configuring the Encoder

3.1 Static Properties

Static properties must be specified before the `CreateEncoder()` method is called. The following table describes the static properties.

3.1.1 DEM_USAGE

The `DEM_USAGE` property can be used to configure the Display Encode Mode session based on an intended use case. The different modes provide different tradeoffs between latency, video quality, and bitrate targets.

Value	Description
DEM_USAGE_WIRELESS_DISPLAY	This is the default usage for Display Encode Mode. Encoding parameters are configured for a typical Wireless Display use-case. This attempts to minimize latency, while preserving video quality.
DEM_USAGE_LOWLATENCY	This usage is intended for high interactivity use cases, such as remote gaming. The rate control parameters are configured to favour latency over encoding quality, to ensure that interactivity requirements are met.
DEM_USAGE_GENERIC	By default, the generic usage parameters are configured for usage scenarios such as remote video playback. This usage also exposes additional parameters, allowing the advanced DEM user to tailor the application to their use cases.

3.1.2 DEM_OUTPUT_TYPE

The `DEM_OUTPUT_TYPE` property is used to configure the output modes for Audio and Video data.

Value	Description
DEM_AV_TS	Output data is returned as an MPEG-2 Transport Stream containing both Audio and Video streams. Note: This is a WiFi Display compliant MPEG-2 Transport stream. Some applications will not be able to detect audio in this stream.
DEM_AV_ES	Output data is returned as separate Audio and Video elementary streams. It is up to the application to handle any audio/video multiplexing.
DEM_V_TS	Output data is returned as an MPEG-2 Transport stream with video data. Audio data will not be captured.
DEM_V_ES	Output data is returned as an AVC encoded Video Elementary Stream
DEM_A_ES	The Display Encode Mode doesn't currently support an Audio-only elementary stream. This parameter can be returned from AMFDemBuffer::GetDataType()

3.1.3 DEM_PROFILE

The `DEM_PROFILE` property is used for configuring the AVC (H.264) profile.

Value	Description
DEM_PROFILE_CONSTRAINED_BASELINE	Constrained Baseline is the default profile for Display Encode Mode, and is the required profile for the DEM_USAGE_WIRELESS_DISPLAY.
DEM_PROFILE_MAIN	Main profile can be selected for DEM_USAGE_LOWLATENCY and DEM_USAGE_GENERIC. CABAC will be automatically be selected if Main profile is selected.
DEM_PROFILE_HIGH	High profile can be selected for DEM_USAGE_LOWLATENCY and DEM_USAGE_GENERIC. CABAC will be automatically be selected if High profile is selected.

3.1.4 DEM_FEEDBACKS

The `DEM_FEEDBACKS` property is used for configuring the number of feedback slots allocated in the driver handshake. Reprogramming the property in common uses is not recommended.

3.1.5 DEM_INSERT_AUD

The AUD (Access Unit Delimiter) NAL unit is present in the bitstream by default. The `DEM_INSERT_AUD` is used for keeping or removing AUD in the AVC (H.264) elementary stream.

3.2 Dynamic properties

All dynamic properties have default values. These values can be adjusted dynamically during the encoding session to help dynamically meet quality of service requirements.

Note: `FlushConfiguration()` must be called after setting one or more dynamic properties, in order for them to be submitted to the encoder hardware. This is intended to allow a set of properties to be sent to the hardware as a single encoding configuration change.

Since the audio and video capturing and the configuration mechanism are asynchronous, there is no way to apply a configuration update to a specific frame. The configuration change will take effect as soon as the command reaches the hardware.

3.2.1 DEM_IDR_PERIOD

The `DEM_IDR_PERIOD` property is used for configuring the interval between Intra Display Refresh pictures. This option can be configured for all uses.

```
DEM_IDR_PERIOD_MIN = 0;
DEM_IDR_PERIOD_MAX = 6000;
```

DEM_IDR_PERIOD_DEFAULT = 300;

Note: If DEM_INTRA_REFRESH_MB_PER_SLOT is non-zero, IDR pictures are spread over a series of intra refreshed frames. For details, see DEM_INTRA_REFRESH_MB_PER_SLOT.

3.2.2 DEM_SKIPPED_PIC_PERIOD

The DEM_SKIPPED_PIC_PERIOD property is used for configuring the interval between skipped pictures. Using this property, the frame rate of the encoded stream can be reduced during times when the network bandwidth cannot keep up with the amount of encoded data.

Value	Description
0	This is the default value, which means don't skip frames.
1	Insert 1 skipped frame after every captured frame. This skips every other frame.
2	Insert 2 skipped frames after every captured frame. This skips every 2 out of three frames.
DEM_SKIPPED_PIC_PERIOD_MAX	The maximum value is 7. This inserts 7 skipped frames after every captured frames. In effect, this would skip 7 out of every eight frames.

3.2.3 DEM_RATE_CONTROL_METHOD

The DEM_RATE_CONTROL_METHOD property controls the rate control method that is used for DEM.

Value	Description
DEM_PEAK_CONSTRAINED_VBR	Peak Constrained Variable Bitrate is the default rate control method. The Peak Constrained VBR rate control mode provides the best possible quality, while meeting the target bitrate of DEM_BITRATE. In addition, it conforms to a leaky bucket model specified by a peak bitrate of DEM_PEAK_BITRATE, a buffer size of DEM_VBV_BUFFER_SIZE and an initial buffer fullness of DEM_INITIAL_VBV_BUFFER_FULLNESS.
DEM_LATENCY_CONSTRAINED_VBR	Latency constrained variable bitrate favours latency over quality. This rate control method will enforce a hard cap on the size of each frame to ensure that low latency is preserved.
DEM_CBR	The Constant Bitrate rate control mode operates in the same way as the peak constrained VBR mode, with the extra constraint that the peak bitrate is equal to the target bitrate (i.e., DEM_PEAK_BITRATE == DEM_BITRATE).
DEM_NO_RC	Constant QP (DEM_QP_I /DEM_QP_P) is used.

3.2.4 DEM_BITRATE

The DEM_BITRATE property is used to set the target bitrate for the encoded bitstream.

3.2.5 DEM_PEAK_BITRATE

The DEM_PEAK_BITRATE property specifies the peak bitrate of the leaky bucket model used with the CBR and Peak Constrained VBR rate control modes. DEM_PEAK_BITRATE represents the maximum sustainable bitrate the transmission channel can operate at.

Note: DEM_PEAK_BITRATE does not guarantee an upper bound on the instantaneous bitrate of the encoded bitstream.

3.2.6 DEM_INITIAL_VBV_BUFFER_FULLNESS

The DEM_INITIAL_VBV_BUFFER_FULLNESS property specifies the initial buffer fullness of the leaky bucket model used with the CBR and Peak Constrained VBR rate control modes. DEM_INITIAL_VBV_BUFFER_FULLNESS controls the initial buffering period the decoder needs to wait before starting the decoding process.

Note: This parameter is not exposed for DEM_USAGE_WIRELESSDISPLAY.

3.2.7 DEM_VBV_BUFFER_SIZE

The DEM_VBV_BUFFER_SIZE property specifies the buffer size of the leaky bucket model used with the CBR and Peak Constrained VBR rate control modes. DEM_VBV_BUFFER_SIZE defines a upper bound on the number of bits per picture and therefore controls the video transmission system latency and the maximum instantaneous bitrate.

Notes:

- In order to be able to achieve the target DEM_BITRATE, DEM_INITIAL_VBV_BUFFER_SIZE must be higher than (DEM_BITRATE/frame_rate).
- A tight DEM_VBV_BUFFER_SIZE constrains the instantaneous bitrate fluctuation, which usually results in poor quality for I/IDR-frames, scene changes and high complexity motion/texture frames.
- This parameter is not exposed for DEM_USAGE_WIRELESSDISPLAY.

3.2.8 DEM_INTRA_REFRESH_MB_PER_SLOT

The intra refresh slot mechanism allows an application to insert intra macroblocks without having to encode an entire IDR picture. Basically, the picture is divided into vertical slots (or columns). The size of a slot is determined by the number of macroblocks per slot.

In the Low Latency usage, IDR pictures are automatically spread over a number of frames using the intra refresh mechanism. For example, if 8 slots fill a frame at every DEM_IDR_PERIOD the intra refresh would be spread over a series of 8 frames, refreshing 1 slot per frame.

3.2.9 DEM_MIN_QP/DEM_MAX_QP

The application can specify the minimum and maximum QP values to use for encoding. Values can be set between 0 and 51.

```
DEM_MIN_QP_DEFAULT = 22;
DEM_MAX_QP_DEFAULT = 51;
```

Note: This parameter is not applicable for `DEM_USAGE_WIRELESSDISPLAY`.

3.2.10 DEM_INLOOP_DEBLOCKING

The `DEM_INLOOP_DEBLOCKING` property can be used to enable/disable in-loop deblocking.

Note: This parameter is only exposed for `DEM_USAGE_GENERIC`.

3.2.11 DEM_SLICES_PER_FRAME

The `DEM_SLICES_PER_FRAME` property can be used to divide the frame into AVC (h.264) slices. This parameter determines the number of slices per frame. The default value is 1.

3.2.12 DEM_QP_I / DEM_QP_P

`DEM_QP_I` and `DEM_QP_P` are used for setting the QP value for I and P slices respectively when `DEM_RATE_CONTROL_METHOD` is `DEM_NO_RC`.

3.2.13 DEM_FILLER_DATA

`DEM_FILLER_DATA` is used for enabling/disabling filler data in `DEM_CBR`.

3.2.14 DEM_MOTION_EST_HALF_PIXEL

`DEM_MOTION_EST_HALF_PIXEL` is used for enabling/disabling half-pixel motion estimation.

3.2.15 DEM_MOTION_EST_QUARTER_PIXEL

`DEM_MOTION_EST_QUARTER_PIXEL` is used for enabling/disabling quarter-pixel motion estimation.

3.2.16 DEM_TIMING_INFO_PRESENT

The `DEM_TIMING_INFO_PRESENT` flag is set to `true` and fixed frame rate is present in SPS VUI parameters by default. `DEM_TIMING_INFO_PRESENT` is used for keeping/removing VUI timing information.

3.3 List of Parameters

Create Parameters

Parameter Name	Values	Description
Usage	Generic, Wireless, LowLatency	Select usage Default: Wireless
Output-type	AV_ES, AV_TS, V_ES, V_TS	Select output type Default: AV_TS
Profile	Baseline, Main, High	Select profile Default: Baseline
Feedbacks	3 – 32	Number of feedback slots

Insert-AUD	On, Off	Insert AUD in transport stream

Configuration Parameters

Rate-Control	VBR,VBR_LAT,CBR, NONE	Select Peak Constrained VBR, Latency Constrained VBR, Constant Bitrate or No rate control Default: Peak constrained VBR. **Note:** It is used for Generic usage mode.
Deblocking	On, Off	Turn on/off deblocking filter Default: On **Note:** It is used for Generic usage mode.
Filler-Data	On, Off	Enable FillerData for Constant Bitrate mode.
HPEL	On, Off	Enable half pixel motion estimation Default: On **Note:** It is used for Generic usage mode.
QPEL	On, Off	Enable quarter pel motion estimation Default: On **Note:** It is used for Generic usage mode.
Timing-Info-Present	On, Off	Set VUI->timing info present flag Default: On **Note:** It is used for all usage modes.
Bitrate	bit/s	Target bitrate **Note:** It is used for all usage modes.
Peak-Bitrate	bit/s	Peak bitrate **Note:** It is used for all usage modes.
IDR-period	0 – 6000	IDR period, value 0 turns IDR off Default: 300 **Note:** It is used for all usage modes.
Skipped-PIC-period	0 – 7	Number of skipped frames. 0 means no skip, 1 means skip every second frame, 2 means skip every two of three frames etc. Default: 0 **Note:** It is used for all usage modes.

Slices-per-frame	Number of slices	Number of slices per frame Default: 1 **Note:** It is used for Generic and Low latency usage modes.
IR-MB-Per-Slot	Number of macro-blocks	Number of intra-refresh macro-blocks per slot **Note:** It is used for all usage modes.
VBV-buff-fullness	0 - 100	Initial VBV buffer fullness Default: 100 **Note:** It is used for Generic usage mode.
VBV-buff-size	Bits	VBV buffer size **Note:** It is used for Generic usage mode.
Min-QP	QP	Min QP Default: 22 **Note:** It is used for Generic usage mode.
Max-QP	QP	Max QP Default: 51 **Note:** It is used for Generic usage mode.
QP-I	QP	QP for I frames in NONE Rate Control mode Default: 22
QP-P	QP	QP for P frames in NONE Rate Control mode. Default: 22

4 Additional Information

4.1 Video Coding Engine: DEM plus other encoding jobs

The DEM mode may not have exclusive use of the Video Coding Engine (VCE) hardware block. For example, other applications may use VCE for Video Conferencing, Video Encoding, or Video Transcoding.

In order to balance its workload, the VCE may reduce its capture rate from 60fps to 30fps or from 50fps to 25fps. This reduction allows some processing time for other jobs. Other DEM frame rates will not be reduced when handling other jobs.

Contact

Advanced Micro Devices, Inc.
One AMD Place
P.O. Box 3453
Sunnyvale, CA, 94088-3453
Phone: +1.408.749.4000

For AMD Accelerated Parallel Processing:

URL: developer.amd.com/appsdk
Developing: developer.amd.com/
Forum: developer.amd.com/openclforum