TriMedia MPEG-1 Audio Encoder API

Overview

Introduction

The MPEG audio encoder is a TSSA compliant module that accepts a stream of PCM format audio at its input stream and generates an MPEG 1 encoded output stream. For information about the general interface philosophy, you are directed to the TSSA software architecture documentation.

The encoder supports both layer 2 and layer 3 encoding. Its maturity is characterized as “pre-alpha.” This means that there are still things to be cleaned up, and there is a slight possibility that minor interface changes may still be necessary.

The layer 2 encoder is well optimized and the quality of the encode is high.

The layer 3 encoder is fairly well optimized, but the quality of the encode is not what some may wish for a product.

This version of the encoder is intended for use in an evaluation. Subsequent releases will be of final product quality. The layer 2 encoder is closer to final quality than the layer 3 encoder.

Use of either of these encoders may require a patent license, as the MPEG audio encoding standards are covered by patents held by various compaies.

Inputs and Outputs

The encoder has one input and one output. The input is a PCM format stream, as described by a TSA packet. Stereo 16 bit is the only supported input format. The sample rate can be 32k, 44.1k, or 48k, as described by the MPEG spec. The output will be encoded MPEG data.

Errors

The errors reported by the MPEG encoder are all defined in tmalAencMpeg1.h. The base value of these errors is 0x140B0000, as defined in tmLibappErr.h.

The user can install a TSA standard error callback function, and the encoder will call this if it encounters errors while decoding the bitstream. In that case, the errorCode will be one of the values defined in tmalAencMpeg1.h. These errors are invariably fatal in todays code, causint the start function to be exited.

Progress

The user can install a TSA standard progress callback function. The encoder will use this in two cases.

1) To report a change in format, per standard TSSA behavior. The defaults handle this.

2) Every frame. This behavior can be enabled or disabled with an appropriate setting of the progress flags at instance setup time.

Configuration

The encoder currently supports only one configuration command, this being a command to flush the output of the encoder. All other changes in setup must be done using the instance setup structure. The encoder is not designed to change its bitrate “on the fly.”

API Data Structure Descriptions

This section describes the TriMedia MPEG-1 Layer II and Layer III audio encoder data structures.

.

tmalAencMpeg1ConfigTypes_t

```
typedef enum {
       AENC_MPEG1_CONFIG_FLUSH_OUTPUT  = tsaCmdUserBase + 0x00,
} tmalAencMpeg1ConfigTypes_t;
```

Description

Used by the InstanceConfig function to affect the operation of the encoder after it has been started. AENC_MPEG1_CONFIG_FLUSH_OUTPUT can be used to flush the current output packet no matter how much it is filled.

tmalAencMpeg1Layer_t

```
typedef enum {
       AENC_MPEG1_LAYER1            = 0x01,
       AENC_MPEG1_LAYER2            = 0x02,
       AENC_MPEG1_LAYER3            = 0x03
} tmalAencMpeg1Layer_t;
```

Description

Used as a parameter to the InstanceSetup function. Selects the MPEG “layer” to be used for encoding. Layer 1 not supported.

tmalAencMpeg1Copyright_t

```
typedef enum {
       AENC_MPEG1_COPYRIGHT_ON     = 0x01,
       AENC_MPEG1_COPYRIGHT_OFF    = 0x02
} tmalAencMpeg1Copyright_t;
```

Description

Used as a parameter to the InstanceSetup function. Sets the copyright bit in the encoded stream.

tmalAencMpeg1Protection_t

```
typedef enum {
       AENC_MPEG1_CRC_ON           = 0x01,
       AENC_MPEG1_CRC_OFF          = 0x00
} tmalAencMpeg1Protection_t;
```

Description

Used as a parameter to the InstanceSetup function. Enables or disables the usage of CRC checksums which can be used to protect the stream from errors in transmission. Default is off.

tmalAencMpeg1Private_t

```
typedef enum {
       AENC_MPEG1_PRIVATE_ON       = 0x01,
       AENC_MPEG1_PRIVATE_OFF      = 0x02
} tmalAencMpeg1Private_t;
```

Description

Used as a parameter to the InstanceSetup function. Sets the state of the “private” bit found in MPEG bitstreams.

tmalAencMpeg1Original_t

```
typedef enum {
       AENC_MPEG1_ORIGINAL         = 0x01,
       AENC_MPEG1_COPY             = 0x02
} tmalAencMpeg1Original_t;
```

Description

Used as a parameter to the InstanceSetup function. Sets the state of the “original” bit found in MPEG bitstreams.

tmalAencMpeg1Emphasis_t

```
typedef enum {
       AENC_MPEG1_NO_EMPHASIS      = 0x01,
       AENC_MPEG1_50_15_EMPHASIS   = 0x02,
       AENC_MPEG1_CCITT_EMPHASIS   = 0x03,
} tmalAencMpeg1Emphasis_t;
```

Description

Used as a parameter to the InstanceSetup function. Sets the state of the “emphasis” bit found in MPEG bitstreams. Defaults to `AENC_MPEG1_NO_EMPHASIS`.

tmAencMpeg1ProgressFlags_t

```
typedef enum {
       AENC_MPEG1_PROG_REPORT_EVERY_FRAME       = 0x01
} tmAencMpeg1ProgressFlags_t;
```

Description

Controls the operation of the progress function. If the progress report flag (found in default instance setup) is set to this value, then the user-installed progress function will be called on each frame as it is decoded.

tmalAencMpeg1Capabilities_t

```
typedef struct {
       ptsaDefaultCapabilities_t    defaultCapabilities;
} tmalAencMpeg1Capabilities_t, *ptmalAencMpeg1Capabilities_t;
```

Fields

`defaultCapabilities`	Pointer to the default capabilities struct. Refer to tsa.h.

Description

Standard TSSA capabilities structure. Used by applications to find out about the inputs and outputs of the component.

tmalAencMpeg1InstanceSetup_t

```
typedef struct {
    ptsaDefaultInstanceSetup_t  defaultSetup;
    tmalAencMpeg1Layer_t        layer;
    UInt32                      bitRate;
    UInt32                      quality;
    tmalAencMpeg1Copyright_t    copyright;
    tmalAencMpeg1Protection_t   protection;
    tmalAencMpeg1Private_t      private;
    tmalAencMpeg1Original_t     original;
    tmalAencMpeg1Emphasis_t     emphasis;
} tmalAencMpeg1InstanceSetup_t, *ptmalAencMpeg1InstanceSetup_t;
```

Fields

defaultSetup	Pointer to the default instance setup struct, refer to tsa.h.
layer	This value determines the encoding profile. Note that the MPEG layer can also be determined by the format of the output descriptor. If the format is installed it overrides this value. Layer I is currently not supported.
AENC_MPEG1_LAYER1	Not supported by this version of the encoder.
AENC_MPEG1_LAYER2	Encoder generates MPEG-1 Layer II bit stream.
AENC_MPEG1_LAYER3	Encoder generates MPEG-1 Layer III bit stream.
bitrate	Determines the bit rate of the encoder output in kbits per second. Legal values are 0, 32, 48, 56, 64, 80, 96, 112, 128, 160, 192, 224, 256, 320, 384 for layer II and 0, 32, 40, 48, 56, 64, 80, 96, 112, 128, 160, 192, 224, 256, 320 for layer III. For layer II, however, correct behavior cannot be guaranteed for values less than 112 kbits per second.
quality	Used only in layer III mode. It determines the break up condition for the bit allocation iteration loop. A lower number stands for more iterations and better audio quality. Supported values are integers between 0 and 30.
copyright	This field determines whether the MPEG stream will be marked as copyright protected.
AENC_MPEG1_COPYRIGHT_ON	Copyright protection bit is set.

AENC_MPEG1_COPYRIGHT_OFF	Copyright protection bit is not set.
protection	This field determines if a CRC word is calculated and inserted into the MPEG bitstream for every audio frame.
AENC_MPEG1_CRC_ON	CRC is calculated and written in MPEG stream.
AENC_MPEG1_CRC_OFF	CRC is not calculated.
private	Determines the value of the private bit to be written into the MPEG frame headers.
AENC_MPEG1_PRIVATE_ON	Bit stream is marked as private.
AENC_MPEG1_PRIVATE_OFF	Bit stream is not marked as private.
original	This value determines whether the MPEG bit stream is marked as original.
AENC_MPEG1_ORIGINAL	Bit stream is marked as an original.
AENC_MPEG1_COPY	Bit stream is marked as a copy.
emphasis	This field characterizes the nature of the pre-emphasis applied to the audio input outside of the encoder.
AENC_MPEG1_NO_EMPHASIS	No emphasis applied.
AENC_MPEG1_50_15_EMPHASIS	50/15 microsecond emphasis applied.
AENC_MPEG1_CCITT_EMPHASIS	Refer to CCITT J.17

Description

This structure, through the fields described above, controls the operational mods of the enocder.

API Function Descriptions

This section describes the TriMedia MPEG-1 Layer II and Layer III audio encoder functions.

tmolAencMpeg1GetCapabilities

```
extern tmLibappErr_t tmolAencMpeg1GetCapabilities  (
       ptmolAencMpeg1Capabilities_t *pcap
);
```

Parameters

pcaps (0)	Pointer to a capabilities structure pointer.

Return Codes

`TMLIBAPP_OK`

Side Effects

Fills in the pointer of a static `tmolAencMpeg1Capabilities_t` structure maintained by the encoder to describe the capabilities and requirements of this library.

tmalAencMpeg1GetCapabilities

```
extern tmLibappErr_t tmolAencMpeg1GetCapabilities  (
       ptmalAencMpeg1Capabilities_t *cap
);
```

Parameters

cap (0)	Pointer to a capabilities structure pointer.

Return Codes

`TMLIBAPP_OK`

Side Effects

Fills in the pointer of a static `tmAencMpeg1Capabilities_t` structure maintained by the encoder to describe the capabilities and requirements of this library.

tmolAencMpeg1Open

```
extern tmLibappErr_t tmolAencMpeg1Open (
       Int                          *instance
);
```

tmalAencMpeg1Open

```
extern tmLibappErr_t tmalAencMpeg1Open (
       Int                          *instance
);
```

Parameters

instance (0)	Pointer to an integer instance variable which will be used to identify the encoder in subsequent transactions.

Return Codes

TMLIBAPP_OK	
TMLIBAPP_ERR_MEMALLOC_FAILED	If a memalloc failed.
TMLIBAPP_ERR_NO_INSTANCE_AVAILABLE	If no further instance is available.

Side Effects

Instantiates an encoder and calloc's an instance structure. Allocates an instance setup structure and fills it with default values.

tmolAencMpeg1Close

```
extern tmLibappErr_t tmolAencMpeg1Close (
      Int                          instance
);
```

tmalAencMpeg1Close

```
extern tmLibappErr_t tmalAencMpeg1Close (
      Int                          instance
);
```

Parameters

instance (I)	As returned by `tmolAencMpeg1Open`

Return Codes

`TMLIBAPP_OK`

`TMLIBAPP_ERR_INVALID_INSTANCE` If the instance is invalid.

Side Effects

Shuts down this instance of the encoder and deletes task. Frees instance variable memory and sets up variable memory.

tmolAencMpeg1GetInstanceSetup

```
extern tmLibappErr_t tmolAencMpeg1GetInstanceSetup (
       Int                          instance,
       ptmolAencMpeg1InstanceSetup_t *rsetup
);
```

tmalAencMpeg1GetInstanceSetup

```
extern tmLibappErr_t tmalAencMpeg1GetInstanceSetup (
       Int                          instance,
       ptmalAencMpeg1InstanceSetup_t *rsetup
);
```

Parameters

instance (I)	As returned by `tmolAencMpeg1Open`.
rsetup (O)	Pointing to a pointer to a setup structure.

Return Codes

`TMLIBAPP_OK`

`TMLIBAPP_ERR_INVALID_INSTANCE` If the instance is invalid.

Description

Fills in the pointer to the setup structure allocated by `tmolAdecAc3Open` or the current setup structure after the setup function has been called.

tmolAencMpeg1InstanceSetup

```
extern tmLibappErr_t tmolAencMpeg1InstanceSetup    (
       Int                          instance,
       tmolAencMpeg1InstanceSetup_t *setup
);
```

tmalAencMpeg1InstanceSetup

```
extern tmLibappErr_t tmalAencMpeg1InstanceSetup    (
       Int                          instance,
       tmalAencMpeg1InstanceSetup_t *setup
```

Parameters

instance (I)	As returned by `tmolAencMpeg1Open`.
setup (I)	Pointing to the setup structure.

Return Codes

`TMLIBAPP_OK`	
`TMLIBAPP_ERR_INVALID_INSTANCE`	If the instance is invalid.
`TMLIBAPP_ERR_NO_QUEUE`	If either the queues for the input or for the first output pin are not assigned.
`AENC_MPEG1_ERR_LAYER_NOT_SUPPORTED`	If selected MPEG layer is not supported.

`TMLIBAPP_ERR_UNSUPPORTED_DATACLASS`
`TMLIBAPP_ERR_UNSUPPORTED_DATATYPE`
`TMLIBAPP_ERR_UNSUPPORTED_DATASUBTYPE`
	If the format in either the input or output descriptor is incorrect.

`AENC_MPEG1_ERR_ILL_SAMPLERATE`	If the sample rate specified in the input format is not supported.
`AENC_MPEG1_ERR_ILL_BITRATE`	If the bit rate specified in the setup struct is not supported.
`AENC_MPEG1_ERR_ILL_QUALITY`	If the quality value specified in the setup struct is not supported.
`AENC_MPEG1_ERR_ILL_EMPHASIS`	If the emphasis value of the setup struct is not supported.

Description

Initializes the instance of the encoder and configures it.

tmolAencMpeg1Start

```
extern tmLibappErr_t tmolAencMpeg1Start (
       Int                          instance
);
```

Parameters

instance (I)	Instance value from `tmolAencMpeg1Open()`.

Return Codes

`TMLIBAPP_OK`	
`TMLIBAPP_ERR_INVALID_INSTANCE`	If the encoder has not been opened by this instance.
`TMLIBAPP_ERR_NOT_SETUP`	If the encoder has not been initialized by `tmalAdecAc3InstanceSetup`.

Description

Starts the `AencMpeg1` encoder's `tmalAencMpeg1Start` function as task.

tmalAencMpeg1Start

```
extern tmLibappErr_t tmalAencMpeg1Start (
       Int                          instance
);
```

Parameters

instance (I)	Instance value from `tmalAencMpeg1Open()`.

Return Codes

`TMLIBAPP_OK`	
`TMLIBAPP_ERR_INVALID_INSTANCE`	If the encoder has not been opened by this instance.
`TMLIBAPP_ERR_NOT_SETUP`	If the encoder has not been initialized.

Description

Starts the data processing loop of the MPEG-1 encoder.

tmolAencMpeg1InstanceConfig

```
extern tmLibappErr_t tmolAencMpeg1InstanceConfig   (
       Int                          instance,
       Int32                        flags,
       tsaControlArgs_t             args
);
```

Parameters

instance (I)	Instance value from tmolAencMpeg1Open()
flags (I)	Flags used for the pSOS command queue which is used to send the command to the AL layer config function.
args (I/O)	Argument struct containing the command, a parameter pointer to a return value field for the return value of the function `tmalAencMpeg1InstanceConfig` and a timeout value.

Return Codes

`TMLIBAPP_OK`

`TMLIBAPP_ERR_INVALID_INSTANCE` If the encoder has not been opened by this instance.

`TMLIBAPP_ERR_NOT_SETUP` If the encoder has not been initialized.

(or error messages from the command queue handler)

Description

Using the default InstanceConfig function, the AL layer's instance config function is invoked. Causes the encoder to execute a command. This function can be used when the encoder is running.

tmalAencMpeg1InstanceConfig

```
extern tmLibappErr_t tmalAencMpeg1InstanceConfig   (
       Int                          instance,
       ptsaControlArgs_t              cmdArgs
);
```

Parameters

instance (I)	Instance value from `tmalAencMpeg1Open()`
cmdArgs	Argument struct containing the command and a parameter pointer. The timeout and the return parameter field are not used by the AL layer function.

Return Codes

`TMLIBAPP_OK`	
`TMLIBAPP_ERR_INVALID_INSTANCE`	If the encoder has not been opened by this instance.
`TMLIBAPP_ERR_NOT_SETUP`	If the encoder has not been initialized.
`TMLIBAPP_ERR_INVALID_COMMAND`	If the config function cannot interprete the command.

Description

Causes the encoder to execute a command. This function can be used when the encoder is running.

tmolAencMpeg1Stop

```
extern tmLibappErr_t tmolAencMpeg1Stop (
       Int                          instance
);
```

Parameters

instance (I)	Instance value from `tmolAencMpeg1Open()`.

Return Codes

`TMLIBAPP_OK`	
`TMLIBAPP_ERR_INVALID_INSTANCE`	If the encoder has not been opened by this instance.
`TMLIBAPP_ERR_NOT_SETUP`	If the encoder has not been initialized.

Description

Invokes the default stop procedure which stops the encoder task and sends pause packets to the connected components.

tmalAencMpeg1Stop

```
extern tmLibappErr_t tmalAencMpeg1Stop (
       Int                          instance
);
```

Parameters

instance (I)	Instance value from `tmalAencMpeg1Open()`.

Return Codes

`TMLIBAPP_OK`	
`TMLIBAPP_ERR_INVALID_INSTANCE`	If the encoder has not been opened by this instance.
`TMLIBAPP_ERR_NOT_SETUP`	If the encoder has not been initialized.

Description

Forces the encoder to the main processing loop of the start function.

tmalAencMpeg1EncodeFrame

```
extern tmLibappErr_t tmalAencMpeg1EncodeFrame (
       Int                          instance,
       tmAvPacket_t                 *inpacket,
       tmAvPacket_t                 *outPacket
);
```

Parameters

instance (I)	Instance value from `tmalAencMpeg1Open()`.
inpacket (I)	Pointer to a full data packet carrying PCM stereo data.
outPacket (O)	Pointer to an empty data packet into which the encoder writes the encoded audio frame.

Return Codes

`TMLIBAPP_OK`	
`TMLIBAPP_ERR_INVALID_INSTANCE`	If the encoder has not been opened by this instance
`TMLIBAPP_ERR_NOT_SETUP`	If the encoder has not been initialized.
`AENC_MPEG1_ERR_NOT_ENOUGH_INPUT_SAMPLES`	If the input packet contains less samples than a full audio frame (1152 for layer II).
`AENC_MPEG1_ERR_OUTBUF_TOO_SMALL`	If the empty output packet is not large enough to store an encoded audio frame.
`AENC_MPEG1_ERR_LAYER_NOT_SUPPORTED`	If the encoder is not supporting the layer chosen during instance setup in non streaming mode.

Description

Encodes one frame of audio data. The user of this function must ensure that the input packet contains the exact number of samples required for one frame. The encoder does not perform any type of buffering between subsequent calls of this function.