TriMedia Motion JPEG Encoder (VencMjpeg) API

Note

This component library is not included with the TriMedia SDE but is available under a separate licensing agreement. Please contact your TriMedia sales representative for more information. ◆

Motion JPEG Encoder API Overview

Motion JPEG (MJPEG) is an implementation of JPEG for a sequence of video frames. As of now no known official standard exists. The MJPEG Encoder Library provided with this release implements standard JFIF (JPEG File Interchange Format File), motion JPEG format A (MJPEG-A) and motion JPEG format B (MJPEG-B) baseline sequential encoding for frames. The encoder encodes bitstreams using

- Lossy, DCT based transform, followed by Huffman run length encoding.
- Input precision of 8 bits
- YCbCr 4:2:2 or YUV 4:1:1 image format.

See Pennebaker & Mitchell, “JPEG Still Image Data Compression Standard,” Van Nostrand-Reinhold NY, 1993 for more details on JPEG, and “Motion JPEG Format”, Draft 2, April 15, 1996, courtesy Apple Computer Inc., for details on MJPEG Formats A and B (See http://www.QuickTimeFAQ/developer/).

MJPEG-A is in full compliance with the ISO JPEG specification. Each frame contains two fields, with the first one being the top field. Each field is a standard JPEG stream. More than one frame in a file makes an MJPEG Sequence file. In addition to standard JFIF markers (JPEG file interchange format), MJPEG-A adds a new application marker called APP1 (id = ”ffe1”). MJPEG-B is nothing but motion JPEG A stripped of all markers. The various fields of the APP1 marker are given below (All fields are 4 bytes long and are in Big Endian order):

1. Unused: typically 0000
2. Tag: It should contain “mjpeg”
3. Field size: size of encoded image data for each field.
4. Padded field size
5. Offset to next field
6. DCT Quantization table offset
7. Huffman Table Offset
8. Start of Image Offset
9. Start of Scan Offset
10. Start of Data Offset

All fields are 4 bytes and in Big Endian order.

This library provides a standard set of seven APIs, like other tm components, which conform to the TriMedia Software Streaming Architecture (TSSA). All interfaces and data

structures are fully compliant with this architecture. The component takes a stream of tmAvPackets (YUV data) as input and produces a stream of tmAvPackets at the output.

The Application Library component (tmalVencMjpeg) provides the basic functionality of OS independent JPEG encoding, while the operating system application library (OL) component (tmolVencMjpeg) takes care of all inputs and outputs.

Performance

The typical performance obtained is around 4Mbits/sec of encoded bit stream, as measured on a cycle accurate simulator with known software tuning on a 100 MHz tml processor

Demonstration Programs

The VencMjpeg component is normally used through the OL layer (tmolVencMjpeg). An example program exolVencMjpeg.c is provided to illustrate the use of this component. It takes a tmAvPacket_t (YUV data) as an input stream, encodes it and puts it into a file. It uses the VdigVI component to capture the images and stream the data as packets to the VencMjpeg component. After processing, the VencMjpeg component streams packets to the Fwrite component which would in turn put it into the file.

Overview of the tmolVencMjpeg / tmalVencMjpeg Component

The tmolVencMjpeg component layer takes care of properly passing OS dependent parameters like empty and full Queue IDs to the tmalVencMjpeg component. Default API's are provided and the API implementation is also largely the same as the default implementations provided with tsaDefaults.c.

The tmalVencMjpeg component library provides an interface consistent with TSSA and provides in all, six C callable functions. A typical Usage Sequence will be:

1. tmalVencMjpegGetCapabilities to get the encoder capabilities data structure.
2. tmalVencMjpegOpen. This opens an instance of the encoder. The encoder does not put any restriction on the number of instances.
3. tmalVencMjpegInstanceSetup. This registers the setup parameters provided by the user into internal instance variables.
4. tmalVencMjpegStart. This encodes YUV data frames sequentially until tmalVencMjpegStop is called.
5. tmalVencMjpegStop. This changes the `state` variable to STOP.
6. tmalVencMjpegClose. This invalidates the instance and frees all memory created by the component.

Input Description

The MJPEG Encoder requests data using the datain callback function registered at the time of setup. Input packets are of type tmAvPacket_t (YUV data). Packet requests are made from within the tmalVencMjpegStart.

The sizes of the image that has to be encoded are present in the tmalVencMjpegImageDescription_t structure registered during setup of the instance.

Output Description

Output packets are of type tmAvPacket_t. The user creates the buffers and puts them into the empty queue. The VencMjpeg uses these buffers to fill encoded data and puts them out through the dataout function.

The output stream will be in any of the following formats:

- JFIF
- Motion JPEG A
- Motion JPEG B

Stopping the VencMjpeg Component

The `VencMjpeg` component may be stopped by calling tmalVencMjpegStop or tmolVencMjpegStop. The former will stop the component after the current frame is processed. The latter is implemented by a call to the tsaDefaultStop. The example exolVencMjpeg.c illustrates one way of stopping the processing chain. When the VencMjpeg stops, it calls its completion function which may be used to synchronize with the other components.

Motion JPEG Encoder API Data Structure Descriptions

This section describes all the data structures concerned with the VencMjpeg component

tmalVencMjpegStates_t

```
typedef enum {
       MJPGEn_RUN = 0x1,
       MJPGEn_STOP = 0x2,
       MJPGEn_PAUSE = 0x4,
       MJPGEn_SKIP = 0x8,
}tmalVencMjpegStates_t,*ptmalVencMjpegStates_t;
```

Fields

MJPGEn_RUN	Value of the instance variable's `state` field when the encoder is encoding the stream. `tmalVencMjpegStart` puts the component into this state.
MJPGEn_STOP	Value of the `state` field when the encoder is stopped or is required to be stopped at the end of the current frame.
MJPGEn_PAUSE	Reserved for future use.
MJPGEn_SKIP	Value of the `state` field when the user wants to skip the current frame. It is the user's responsibility to release SKIP and put back RUN or STOP. This can be done by using the `progress` function.

tmalVencMjpegStream_t

```
typedef enum {
       MJPGEn_A                       = 0x1,
       MJPGEn_B                       = 0x2,
       MJPGEn_JFIF                    = 0x4,
       MJPGEn_UNSUPPORTED_STREAM_TYPE = 0x8,
}tmalVencMjpegStream_t,*ptmalVencMjpegStream_t;
```

Fields

MJPGEn_A	Encoded stream type is Motion JPEG-A.
MJPGEn_B	Encoded stream type is Motion JPEG-B.
MJPGEn_JFIF	Encoded stream type is JFIF.
MJPGEn_UNSUPPORTED_STREAM_TYPE	Unknown input stream.

Description

These are the stream types generated by the encoder. This is passed by the user through the image description field of the setup variable.

tmalVencMjpegProgressFlags_t

```
typedef enum {
       MJPGEn_REPORT_START_ENCODING = 0x1,
       MJPGEn_REPORT_STOP_ENCODING  = 0x2,
}tmalVencMjpegProgressFlags_t,*ptmalVencMjpegProgressFlags_t;
```

Fields

`MJPGEn_REPORT_START_ENCODING`	Report before starting the encoding.
`MJPGEn_REPORT_STOP_ENCODING`	Report after encoding has stopped.

tmalVencMjpegBufferType_t

```
typedef enum {
       FULL_BUFFER                    = 0x1,
       PADDED_FF                      = 0x2,
} tmalVencMjpegBufferType_t,*ptmalVencMjpegBufferType_t;
```

Fields

FULL_BUFFER	Output buffer allocated equals the maximum image size.
PADDED_FF	Output buffer allocated is of fixed size.

tmalVencMjpegCapabilities_t/ tmolVencMjpegCapabilities_t

```
typedef struct {
       ptsaDefaultCapabilities_t  defaultCapabilities;
}tmalVencMjpegCapabilities_t,*ptmalVencMjpegCapabilities_t;
```

Fields

defaultCapabilities	Pointer to `tsaDefaultCapabilities_t`

Description

See tsa.h for details. Replace “al” by “ol” in information above to get description of `tmolVencMjpegCapabilities_t`.

tmalVencMjpegImageDescription_t/ tmolVencMjpegImageDescription_t

```
typedef struct {
      UInt32                        height;
      UInt32                        width;
      UInt32                        imageStride;
      UInt32                        is_field;
      tmVideoRGBYUVFormat_t         format;
      tmalVencMjpegStream_t         stream;
}tmalVencMjpegImageDescription_t,*ptmalVencMjpegImageDescription_t;
```

Fields

height	This must contain the height of the input image.
width	This must contain the width of the input image.
imageStride	This must contain the stride of the input image.
is_field	This must be zero for frame based encoding and 1 for field based encoding.
format	This must contain the input image format which is YUV 4:2:2 or YUV 4:1:1
stream	This must contain the output stream type which can be JFIF, MJPEG A or MJPEG B

Description

This contains parameters that describe the image. It also contains a field to indicate the type of the output stream to be generated.

Note

The user is expected to create the image buffers and pass them to the encoder. The expected size of the buffer is the product of the height and imageStride.

tmalVencMjpegInstanceSetup_t/ tmalVencMjpegInstanceSetup_t

```
typedef struct {
       ptsaDefaultInstanceSetup_t   defaultSetup;
       ptmalVencMjpegImageDescription_t ImageDescription;
       tmalVencMjpegBufferType_t    BufferType;
}tmalVencMjpegInstanceSetup_t,*ptmalVencMjpegInstanceSetup_t;
```

Fields

defaultSetup	Stores the default values of this application library.
ImageDescription	This is a structure describing the image parameters.
BufferType	This indicates whether the output data will have padded data or not. This depends on the memory available with the application.

Description

This structure contains all the required parameters to initialize the MJPEG video encoder. Replace “al” by “ol” in information above to get description of `tmolVencMjpegInstanceSetup_t`.

Motion JPEG Encoder API Function Descriptions

This section describes the various API functions for the MJPEG Encoder component.

tmalVencMjpegGetCapabilities / tmolVencMjpegGetCapabilities

```
tmLibappErr_t tmalVencMjpegGetCapabilities (
       ptmalVencMjpegCapabilities_t *cap
)
tmLibappErr_t tmolVencMjpegGetCapabilities (
       ptmalVencMjpegCapabilities_t *cap
)
```

Parameters

cap	Pointer to the `tmalVencMjpegCapabilities_t` data structure.

Return Codes

`TMLIBAPP_OK`	Returned on successful completion.
`MJ_ERR_NULL_POINTER`	Returned if `cap` is equal to Null.

Description

This function initializes the `cap` struct with the MJPEG Encoder component's values.

tmalVencMjpegOpen / tmolVencMjpegOpen

```
tmLibappErr_t tmalVencMjpegOpen(
      Int                          *instance
);
tmLibappErr_t tmolVencMjpegOpen(
      Int                          *instance
);
```

Parameters

Pointer to the instance.

Return Codes

`TMLIBAPP_OK`	Returned on successful completion
`TMLIBAPP_ERR_MEMALLOC_FAILED`	Returned if unable to allocate memory for compressor.

Description

This function will create an instance of the `VencMjpeg` component.

tmalVencMjpegClose / tmolVencMjpegClose

```
tmLibappErr_t tmalVencMjpegClose(
      Int                          instance
);
tmLibappErr_t tmolVencMjpegClose(
      Int                          instance
);
```

Parameters

instance	The instance value assigned at the call of the MJPEG Open function

Return Codes

TMLIBAPP_OK	Returned on successful completion.
TMLIBAPP_ERR_INVALID_INSTANCE	Returned if the passed parameter is not a valid instance.

Description

This function invalidates the instance and frees all memory allocated for instance variables and Decompression Instance.

tmolVencMjpegGetInstanceSetup

```
tmLibappErr_t tmolVencMjpegGetInstanceSetup(
      Int                          instance,
      ptmolVencMjpegInstanceSetup_t setup
);
```

Parameters

instance	Instance value assigned at the call of the MJPEG Open function.
setup	Pointer to the Instance setup structure of VencMjpeg.

Return Codes

TMLIBAPP_OK	Returned on successful completion.

Description

This function assigns the pointer to `tmolVencMjpegInstanceSetup_t` structure allocated by the MJPEG Open function to `setup`.

tmalVencMjpegInstanceSetup / tmolVencMjpegInstanceSetup

```
tmLibappErr_t tmalVencMjpegInstanceSetup (
      Int                          instance,
      ptmalVencMjpegInstanceSetup_t instanceSetup
);
tmLibappErr_t tmolVencMjpegInstanceSetup(
      Int                          instance,
      ptmalVencMjpegInstanceSetup_t instanceSetup
);
```

Parameters

instance	Instance value assigned at the call of the MJPEG Open function.
instanceSetup	Pointer to the instanceSetup data structure.

Return Codes

TMLIBAPP_OK	Returned on successful completion.
TMLIBAPP_ERR_INVALID_INSTANCE	Returned if the instance value is not valid.
TMLIBAPP_ERR_INVALID_SETUP	Returned if any of the fields in the `setup` data struct are invalid.

Description

This function registers the setup parameters provided by the user into the internal instance variables.

tmalVencMjpegStart / tmolVencMjpegStart

```
tmLibappErr_t tmalVencMjpegStart(
       Int                          instance
)
tmLibappErr_t tmolVencMjpegStart(
       Int                          instance
)
```

Parameters

instance	The instance value of the MJPEG video encoder.

Return Codes

TMLIBAPP_OK	Returned on successful completion.
TMLIBAPP_ERR_DATAIN_FAILED	Returned if `datain` callback function fails.
TMLIBAPP_ERR_DATAOUT_FAILED	Returned if `dataout` callback function fails.

Description

This function does the encoding of the data in a streaming mode.

tmalVencMjpegEncodeFrame

```
tmLibappErr_t tmalVencMjpegEncodeFrame(
       Int                          instance
)
```

Parameters

instance	The instance value assigned at the call of the MJPEG Open function.

Return Codes

TMLIBAPP_OK	Returned on successful completion.
TMLIBAPP_ERR_DATAIN_FAILED	Returned if `datain` callback function fails.
TMLIBAPP_ERR_DATAOUT_FAILED	Returned if `dataout` callback function fails.

Description

This function does the encoding of the data in a push mode, one packet at a time.

tmalVencMjpegStop / tmolVencMjpegStop

```
tmLibappErr_t tmalVencMjpegStop(
       Int                          instance
)
tmLibappErr_t tmolVencMjpegStop(
       Int                          instance
)
```

Parameters

instance	The instance value of the MJPEG video encoder.

Return Codes

TMLIBAPP_OK	Returned on successful completion.
TMLIBAPP_ERR_INVALID_INSTANCE	Returned if the instance has not been opened.
TMLIBAPP_ERR_NOT_SETUP	Returned if the instance has not been setup.

Description

`tmalVencMjpegStop` merely changes the component's `state` variable to STOP. `tmolVencMjpegStop` calls `tsaDefaultStop`.