MPFM Programming Reference

Version 2.2

www.radisys.com

World Headquarters
5445 NE Dawson Creek Drive • Hillsboro, OR
97124 USA
Phone: 503-615-1100 • Fax: 503-615-1121
Toll-Free: 800-950-0044

International Headquarters
Gebouw Flevopoort • Televisieweg 1A
NL-1322 AC • Almere, The Netherlands
Phone: 31 36 5365595 • Fax: 31 36 5365620

RadiSys Microware Communications Software Division, Inc.
1500 N.W. 118th Street
Des Moines, Iowa 50325
515-223-8000

Revision D
November 1999

Table of Contents

Chapter 1: MPFM Functions

This chapter introduces the MPFM functions. Functions are introduced according to general use and are arranged alphabetically.

MPFM Functions

Header Files

All MPFM functions require that the following header file be included in applications using any MPFM function:

```
#include <DAVID/mpfm.h>
```

Basic Play Functions

Function	Description
`_os_ss_ma_create()`	Creates MAM
`_os_ss_mv_create()`	Reserves MPEG Video Descriptor
`_os_ss_ma_play()`	Creates MAM
`_os_ss_mv_play()`	Starts MPEG Video Play
`_os_ss_ma_abort()`	Aborts Current MPEG Audio Play
`_os_ss_mv_abort()`	Aborts Current MPEG Video Play
`_os_ss_ma_trigger()`	Defines MPEG Audio Events to Signal
`_os_ss_mv_trigger()`	Defines MPEG Video Events to Signal
`_os_ss_ma_close()`	Frees MAM Descriptor
`_os_ss_mv_close()`	Frees MVM Descriptor

Display Control Functions

Function	Description
`_os_ss_mv_bcolor()`	Sets Display Window Border Color
`_os_ss_mv_hide()`	Disables Display Window Output
`_os_ss_mv_show()`	Enables Window Display

Status and Information Functions

Function	Description
`_os_gs_ma_info()`	Gets Pointer to MAM Descriptor
`_os_gs_ma_status()`	Gets Status of Current MPEG Audio Play
`_os_gs_mv_status()`	Gets Status of Active MPEG Video Play
`_os_gs_mv_info()`	Gets Pointer to MVM Descriptor

Special Video Functions

Function	Description
`_os_ss_mv_at_config()`	Starts MPEG Video Anti-Taping Configuration
`_os_ss_mv_at_off()`	Turns MPEG Video Anti-Taping Off
`_os_ss_mv_at_on()`	Turns MPEG Video Anti-Taping On
`_os_ss_mv_cc_off()`	Turns MPEG Video Closed-Caption Off
`_os_ss_mv_cc_on()`	Turns MPEG Video Closed-Caption On

_os_gs_ma_info()

Gets Pointer to MAM Descriptor

Syntax

```
#include <mpfm.h>
error_code _os_gs_ma_info(
     path_id      path,
     u_int16      mapid,
     mpad         **cmpad);
```

Libraries

`mpfm.l`

Description

`_os_gs_ma_info()` gets a pointer to the Motion Audio Map (MAM) descriptor corresponding to the given audio map ID. The fields in the descriptor are for information purposes only and should only be changed by calling the appropriate functions.

Parameters

`path`	A path to the MPEG audio device.
`mapid`	The map ID as returned by `_os_ss_ma_create()` or `_os_gs_ma_status()`.
`cmpad`	Contains the address of a pointer that points to the MAM descriptor corresponding to the given audio map ID.

Non-Fatal Errors

```
EOS_BMODE
EOS_UNID
EOS_PERMIT
```

_os_gs_ma_status()

Gets Status of Current MPEG Audio Play

Syntax

```
#include <mpfm.h>
error_code _os_gs_ma_status(
     path_id      path,
     masb         *masb_ptr,
     u_int16      *mapid);
```

Libraries

`mpfm.l`

Description

`os_gs_ma_status()` gets the currently active audio map ID and its status. If a map is not active, the `EOS_NOPLAY` error is returned.

Parameters

`path`	A path to the MPEG audio device
`masb_ptr`	Points to the MPEG audio status block to fill. If `masb_ptr` is a null pointer, the status block is not filled and only the currently active map ID is returned.
`mapid`	Points to a location where the currently active map ID is returned You can use the returned audio map ID value to retrieve more information by issuing the `_os_gs_ma_info()` call and reading the MAM descriptor fields.

Non-Fatal Errors

```
EOS_BMODE
EOS_NOPLAY
EOS_PERMIT
```

See Also

`_os_gs_ma_info()`

_os_gs_mv_info()

Gets Pointer to MVM Descriptor

Syntax

```
#include <mpfm.h>
error_code _os_gs_mv_info(
     path_id      path,
     u_int16      mapid,
     mpvd         **cmpvd);
```

Libraries

`mpfm.l`

Description

`_os_gs_mv_info()` gets a pointer to the Motion Video Map (MVM) descriptor in `cmpvd`. You may not alter the contents of the MVM descriptor.

This is a privileged call. Only processes with a user ID of the super user or the user ID of the process that created the MVM may use this call.

Parameters

`path`	A path to the MPEG video device
`mapid`	The currently active map ID, as returned by the `_os_ss_mv_create()` or `os_ss_mv_status()` call.
`cmpvd`	Points to a location where the requested MVM descriptor's pointer is returned

Non-Fatal Errors

```
EOS_BPNUM
EOS_UNID
EOS_PERMIT
```

_os_gs_mv_status()

Gets Status of Active MPEG Video Play

Syntax

```
#include <mpfm.h>
error_code _os_gs_mv_status(
     path_id      path,
     mvsb         *mvsb_ptr,
     u_int16      *mapid);
```

Libraries

`mpfm.l`

Description

`os_gs_mv_status()` gets the currently active map ID and its status. This function passes a buffer which is filled by the decoder. If a map is not active, an `EOS_NOPLAY` error is returned.

Parameters

`path`	A path to the MPEG video device
`mvsb_ptr`	Points to the MPEG video status block to fill. If `mvsb_ptr` is a null pointer, the status block is not filled and only the currently active map ID is returned.
`mapid`	Points to a location where the currently active map ID is returned You can use the returned map ID value to retrieve more information by issuing the `_os_gs_mv_info()` call and reading the descriptor fields.

Non-Fatal Errors

```
EOS_BPNUM
EOS_NOPLAY
EOS_PERMIT
```

See Also

`_os_gs_mv_info()`

_os_ss_ma_abort()

Aborts Current MPEG Audio Play

Syntax

```
#include <mpfm.h>
error_code _os_ss_ma_abort(path_id path);
```

Libraries

```
mpfm.l
```

Description

Aborts the play that is currently being executed. The play is no longer active. If a play is not active when this call is made, an `EOS_ABORT` error is returned.

Parameters

`path`	A path to the MPEG audio device
	A successful call causes the output to be muted and resets all the fields in the MPEG audio descriptor.

Non-Fatal Errors

```
EOS_BMODE
EOS_ABORT
EOS_PERMIT
```

Note

If this play was running in synchronized mode with a video play, `_os_ss_ma_abort()` ends the synchronized mode. The video playback continues in non-synchronized mode.

_os_ss_ma_close()

Frees MAM Descriptor

Syntax

```
#include <mpfm.h>
error_code _os_ss_ma_close(
     path_id     path,
     u_int32     mapid);
```

Libraries

```
mpfm.l
```

Description

`_os_ss_ma_close()` frees the given MAM descriptor to the MPFM. If the given descriptor was playing (or paused) at the time of this call, the play is aborted before the MAM descriptor is freed.

Parameters

`path`	A path to the MPEG audio device
`mapid`	The currently active map ID

Non-Fatal Errors

```
EOS_BMODE
EOS_UNID
EOS_PERMIT
```

Syntax

```
#include <mpfm.h>
error_code _os_ss_ma_create(
     path_id      path,
     u_int16      type,
     u_int16      *mapid);
```

Libraries

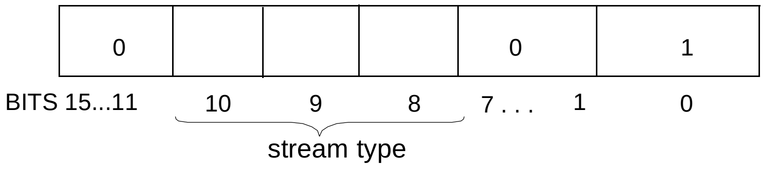
`mpfm.l`

Description

`_os_ss_ma_create()` reserves and initializes a Motion Audio Map (MAM) descriptor.

Parameters

`path`	A path to the MPEG audio device
`type`	The type of MAM to reserve. It has the following format:



Bit 0	This bit is always 1
Bit 1-7	Reserved for future use; must be 0
Bit 8-10	Input stream type

Value	Stream Type
000	ISO/IEC 11172-1 audio system stream.
001	ISO/IEC 13818-3 or 11172-3 audio elementary stream.

Value	Stream Type (continued)
010	ISO/IEC 13818-1 audio transport stream.
011	ISO/IEC 13818-1 audio program stream.
100	ISO/IEC 13818-1 audio PES stream.

Bits 11-15	Reserved for future use; must be 0
`mapid`	Points to a location where the reserved map ID is returned

Non-Fatal Errors

```
EOS_BMODE
EOS_ILLPRM
EOS_NORAM
EOS_MEMFULL
```

_os_ss_ma_play()

Starts MPEG Audio Play

Syntax

```
#include <mpfm.h>
error_code _os_ss_ma_play(
     path_id     path,
     u_int16     mapid,
     u_int32     playoffs,
     u_int32     mapsize,
     u_int32     vpath,
     u_int32     syncoff,
     scl         *sclptr,
     stat_blk    *asyblkptr);
```

Libraries

`mpfm.l`

Description

`_os_ss_ma_play()` starts to play the data belonging to the given Motion Audio Map (MAM) ID. The data comes from the network through a demultiplexing chip. This is an asynchronous call so the application continues executing while the play is executing.

Parameters

`path`	A path to the MPEG audio device
`mapid`	The MAM map ID on which the play is started
`playoffs`	Set to 0
`mapsize`	Set to 0
`vpath`	Depending on its value, `vpath` has one of the following meanings:

	1) If it is set to -1, the play is set to asynchronous mode. The audio starts to play immediately if no video play has been set to wait state. If there is a video play waiting to be synchronized to, `EOS_DEVBSY` is returned for this audio play.
	2) If `vpath` is set to -2, the play enters into a waiting state. It stays in this state until a video play synchronizes to it.
	3) If `vpath` is a valid path to a video play that is in either waiting mode (started with -2) or asynchronous play mode (started with -1), this audio starts to play synchronously with the intended video. If the video is in waiting mode, it starts to play synchronously with the audio.
`syncoff`	The difference between audio and video timing parameters The synchronized offset parameter indicates the constant difference between the timing parameters in the audio and video sequence. This parameter is defined in units of 90 kHz as the most significant 32 bits of the difference between the decoder system clocks in the MPEG video decoder and the MPEG audio decoder. In a formula: dsc (video) - dsc (audio)
`sclptr`	Is set to `NULL`
`asyblkptr`	Points to an asynchronous status block. If status is not needed, this parameter may be NULL.

Non-Fatal Errors

`EOS_BMODE`

```
EOS_UNID
EOS_BMODE
EOS_DEVBSY
EOS_PERMIT
```

_os_ss_ma_trigger()

Defines MPEG Audio Events to Signal

Syntax

```
#include <mpfm.h>
error_code _os_ss_ma_trigger(
     path_id     path,
     u_int16     sigmask);
```

Libraries

`mpfm.l`

Description

`_os_ss_ma_trigger()` activates signalling of MPEG audio events. The driver sets up the corresponding interrupts and sends the appropriate signal when an event (for which a signal has been requested) occurs.

Parameters

`path`	A path to the MPEG audio device
`sigmask`	Indicates which signals to send to the application. By setting or clearing the corresponding bits in the events mask parameter, you can define for which occurrence of the MPEG Audio events you want to receive a signal from the decoder. When one or more of the indicated events happens, a signal is sent to the application. The value of this signal consists of two parts:

signal base	0	0	0	0	0	DER	DEC	UNF	UPD	CSU	EOI
Bit 15. 11	10	9	8	7	6	5	4	3	2	1	0

Signal/Event Mask

The upper five bits of the 16-bit signal value are set by the application when it issued the `_os_ss_ma_trigger()` call to set the base value of the signal.

The remaining bits reflect the events for which an application requires signals from the MPFM. The decoder only signals on those bits that were enabled in the event mask when the `_os_ss_ma_trigger()` was made.

The setting of the signal/event mask remains valid for this path until the path is closed or a new `_os_ss_ma_trigger()` call is issued for this path.

Bits	Name	Description
0	EOI	Program end code detected.
1	CSU	Decoder changed to a new audio stream.
2	UPD	Decoder updated the frame header.
3	UNF	Decoder does not have data to decode (underflow).
4	DEC	Decoder started decoding.
5	DER	Data Error during play.
6-10		Reserved — should be zero.
11-15		Signal base: upper 5 bits of the 16-bit signal to send. Value must be between 00001 and 11111 binary.

Non-Fatal Errors

`EOS_BMODE`

_os_ss_mv_abort()

Aborts Current MPEG Video Play

Syntax

```
#include <mpfm.h>
error_code _os_ss_mv_abort(path_id path);
```

Libraries

`mpfm.l`

Description

`_os_ss_mv_abort()` aborts the active play. If a play is not active, an EOS_ABORT error is returned. The last-displayed picture continues to display.

Parameters

`path`	A path to the MPEG video device

Non-Fatal Errors

```
EOS_BMODE
EOS_ABORT
EOS_PERMIT
```

_os_ss_mv_at_config()

Starts MPEG Video Anti-Taping Configuration

Syntax

```
#include <mpfm.h>
error_code _os_ss_mv_at_config(
     path_id      path,
     u_char       *key,
     u_int32      keylen,
     u_char       *confstr,
     u_int32      strlen);
```

Libraries

`mpfm.l`

Description

`_os_ss_mv_at_config()` sets up the configuration of the MPEG video anti-taping function.

Parameters

`path`	A path to the MPEG video device
`key`	A variable length string that authenticates the right to turn on the anti-copy function
`keylen`	The length of the key in bytes
`confstr`	The configuration string. This is a bit stream of some special format depending on the anti-taping technique used. It carries the information to set up some registers before anti-taping can be used. Contact your anti-taping license provider for more information if it is required by your hardware.
`strlen`	The length of the configuration string in bytes

Non-Fatal Errors

```
EOS_BMODE
EOS_PERMIT
EOS_ILLPRM
```

Note

Before turning the anti-taping function on or off, this system call is necessary to configure the anti-taping hardware.

_os_ss_mv_at_off()

Turns MPEG Video Anti-Taping Off

Syntax

```
#include <mpfm.h>
error_code _os_ss_mv_at_off(
     path_id      path,
     u_char       *key,
     u_int32      keylen);
```

Libraries

`mpfm.l`

Description

`_os_ss_mv_at_off()` turns off the MPEG video anti-taping function.

If the anti-taping function is not configured using `_os_ss_mv_at_config()`, an `EOS_BMODE` error is returned.

Parameters

`path`	A path to the MPEG video device
`key`	A variable length string that authenticates the right to turn off the anti-taping function
`keylen`	The length of the key in bytes

Non-Fatal Errors

```
EOS_BMODE
EOS_PERMIT
EOS_ILLPRM
EOS_BMODE
```

_os_ss_mv_at_on()

Turns MPEG Video Anti-Taping On

Syntax

```
#include <mpfm.h>
error_code _os_ss_mv_at_on(
     path_id     path,
     u_char      *key,
     u_int32     keylen,
     u_int32     mode);
```

Libraries

`mpfm.l`

Description

`_os_ss_mv_at_on()` turns on the MPEG video anti-taping function. If the anti-taping function is not configured using `_os_ss_mv_at_config()`, an `EOS_BMODE` error is returned.

There are several ways anti-taping can be implemented. An anti-taping license provider may allow you to use one or all of these methods to accomplish anti-taping by setting up the `mode` parameter.

Parameters

`path`	A path to the MPEG video device
`key`	A variable length string that authenticates the right to turn on the anti-taping function
`keylen`	The length of the key in bytes
`mode`	The anti-taping mode setup

Non-Fatal Errors

```
EOS_BMODE
EOS_PERMIT
EOS_ILLPRM
```

_os_ss_mv_bcolor()

Sets Display Window Border Color

Syntax

```
#include <mpfm.h>
error_code _os_ss_mv_bcolor(
     path_id      path,
     u_int16      mapid,
     u_int32      colorval);
```

Libraries

`mpfm.l`

Description

`_os_ss_mv_bcolor()` sets the border color. If the specified Motion Video Map (MVM) is currently active, the parameters are copied into the MVM descriptor and activated immediately. If the MVM is not active, (no play is going on) the parameters are copied into the MVM descriptor only.

Parameters

`path`	A path to the MPEG video device
`mapid`	The currently active video map ID
`colorval`	Specifies the value of the color. The following is its format:

0	Red (7 - 0)	Green (7 - 0)	Blue (7 - 0)
Bit 31 24	23 16	15 8	7 0

For each component, black level is at 16 and nominal peak (white) level is at 235 (CCIR 601 restrictions).

Non-Fatal Errors

```
EOS_BMODE
EOS_UNID
EOS_PERMIT
EOS_ILLPRM
```

_os_ss_mv_cc_off()

Turns MPEG Video Closed-Caption Off

Syntax

```
#include <mpfm.h>
error_code _os_ss_mv_cc_off(path_id path);
```

Libraries

`mpfm.l`

Description

`_os_ss_mv_cc_off()` disables the output of the closed-caption of the video stream. The video decoding continues.

Parameters

`path`	A path to the MPEG video device

Non-Fatal Errors

```
EOS_BMODE
EOS_PERMIT
```

_os_ss_mv_cc_on()

Turns MPEG Video Closed-Caption On

Syntax

```
#include <mpfm.h>
error_code _os_ss_mv_cc_on(path_id path);
```

Libraries

```
mpfm.l
```

Description

`_os_ss_mv_cc_on()` enables the output of the closed-caption of the video stream.

Parameters

`path`	A path to the MPEG video device

Non-Fatal Errors

```
EOS_BMODE
EOS_PERMIT
```

_os_ss_mv_close()

Frees MVM Descriptor

Syntax

```
#include <mpfm.h>
error_code _os_ss_mv_close(
     path_id      path,
     u_int16      mapid);
```

Libraries

`mpfm.l`

Description

`_os_ss_mv_close()` aborts any ongoing actions on the specified MVM and frees the used MVM descriptor. The last-displayed picture remains visible. This call is the counterpart to `_os_ss_mv_create()`.

Parameters

`path`	A path to the MPEG video device
`mapid`	The currently active map ID

Non-Fatal Errors

```
EOS_BMODE
EOS_UNID
EOS_PERMIT
```

_os_ss_mv_create()

Reserves MPEG Video Descriptor

Syntax

```
#include <mpfm.h>
error_code _os_ss_mv_create(
     path_id      path,
     u_int16      type,
     u_int16      *mapid);
```

Libraries

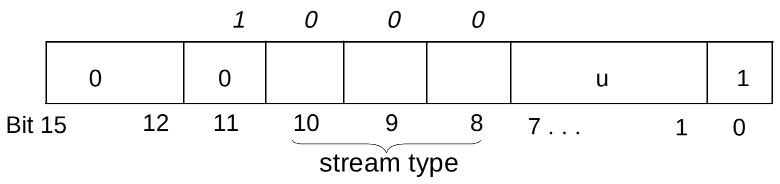
`mpfm.l`

Description

`_os_ss_mv_create()` reserves and initializes a Motion Video Map (MVM) descriptor. See ***Using MPFM***.

Parameters

`path`	A path to the MPEG video device
`type`	The type of descriptor to reserve and has the following format:



Bit 0	This bit is set to 1. The decoder assumes that the data is coming directly from an external device such as a real-time network.
Bit 1-7	Reserved for future use; must be zero.
Bit 8-10	The stream types are defined as follows:

Value	Stream Type
000	ISO/IEC 11172-1 video system stream.
001	ISO/IEC 13818-2 or 11172-2 video elementary stream.
010	ISO/IEC 13818-1 video transport stream.

	Value	**Stream Type (continued)**
	011	ISO/IEC 13818-1 video program stream.
	100	ISO/IEC 13818-1 video PES stream.
Bit 11-15	Reserved. Set to 0.	

`mapid`	Points to a location where the newly created MVM descriptor ID is returned

Non-Fatal Errors

`EOS_BMODE`
`EOS_ILLPRM`
`EOS_MEMFUL`
`EOS_NORAM`

See Also

`_os_ss_mv_play()`

_os_ss_mv_hide()

Disables Display Window Output

Syntax

```
#include <mpfm.h>
error_code _os_ss_mv_hide(path_id path);
```

Libraries

```
mpfm.l
```

Description

`_os_ss_mv_hide()` disables the output of the display window on the next vertical retrace. The display window becomes black, but decoding continues.

Parameters

`path`	A path to the MPEG video device

Non-Fatal Errors

```
EOS_BMODE
EOS_PERMIT
```

_os_ss_mv_play()

Starts MPEG Video Play

Syntax

```
#include <mpfm.h>
error_code _os_ss_mv_play(
     path_id      path,
     u_int16      mapid,
     u_int32      playoffs,
     u_int32      mapsize,
     u_int32      speedval,
     u_int32      apath,
     u_int32      syncoff,
     scl          *sclptr,
     stat_blk     *asyblkptr);
```

Libraries

`mpfm.l`

Description

`_os_ss_mv_play()` starts to play the data belonging to the given Motion Video Map (MVM) ID. The data is coming directly from the network

This call is asynchronous so the application continues executing while the play is executing.

Parameters

`path`	A path to the MPEG video device
`mapid`	The ID of the MVM to play
`playoffs`	Is set to 0
`mapsize`	Is set to 0
`speedval`	Is set to 0
`apath`	The audio path to synchronize play to, or contains the following special values:

	1) If the `apath` parameter is set to -1, the play is set to asynchronous mode. The video starts to play immediately if no audio play has been set to wait state. If there is an audio play waiting to be synchronized to, `EOS_DEVBSY` is returned for the video play. 2) If the `apath` parameter is set to -2, the play enters a wait state. 3) If `apath` is a valid path to a video play which is in either waiting mode (started with -2) or asynchronous play mode (started with -1), the video starts to play synchronously with the intended audio. If the audio is in wait mode, it starts to play synchronously with the video.
`syncoff`	The difference between audio and video timing parameters. The synchronized offset parameter indicates the constant difference between the timing parameters in the audio and video sequence. This parameter is defined in units of 90 kHz as the most significant 32 bits of the difference between the decoder system clocks in the MPEG video decoder and the MPEG audio decoder. In a formula: dsc (video) - dsc (audio)
`sclptr`	Should be `NULL`
`asyblkptr`	Points to the video asynchronous status block structure. If no status is needed the value of this pointer may be NULL.

Non-Fatal Errors

`EOS_BMODE`
`EOS_UNID`
`EOS_DEVBSY`

```
EOS_ILLPRM
EOS_MEMFUL
EOS_NORAM
EOS_BPADDR
EOS_PERMIT
```

See Also

`_os_ss_mv_abort()`

_os_ss_mv_show()

Enables Window Display

Syntax

```
#include <mpfm.h>
error_code _os_ss_mv_show(
     path_id      path,
     u_char       page);
```

Libraries

```
mpfm.l
```

Description

`_os_ss_mv_show()` enables the window to be displayed in the full motion video plane.

If an MPEG video play is currently active, the window of the active map will be enabled on the next picture change. Otherwise, it is enabled on the next vertical retrace.

Parameters

`path`	A path to the MPEG video device
`page`	Must be set to 0

Non-Fatal Errors

```
EOS_BMODE
EOS_ILLPRM
EOS_PERMIT
```

_os_ss_mv_trigger()

Defines MPEG Video Events to Signal

Syntax

```
#include <mpfm.h>
error_code _os_ss_mv_trigger(
     path_id      path,
     u_int16      sigmask);
```

Libraries

`mpfm.l`

Description

`_os_ss_mv_trigger()` activates the signalling of MPEG video events. The driver sets up the corresponding interrupts and sends the appropriate signal when an event (for which a signal has been requested) occurs.

Parameters

`path`	A path to the MPEG video device
`sigmask`	Indicates which signals should be sent to the application. By setting or clearing the corresponding bits in the events mask parameter, you can define for which occurrence of the MPEG video events you want to receive a signal from the decoder. When one or more of the indicated events happens, the application receives a signal. The value of this signal consists of two parts:

signal base		0	NIS	BUF	EOS	EOI	CNP	LPD	SOS	GOP	PIC	DER
Bit 15.	11	10	9	8	7	6	5	4	3	2	1	0

Signal/Event Mask Format

The upper five bits of the 16-bit signal value are set by the application when it issues the `_os_ss_mv_trigger()` call to set up the base of the signal.

The remaining bits reflect the events for which an application requires signals from the MPFM. The decoder only signals on those bits that were enabled in the event mask when the `_os_ss_mv_trigger()` call was made.

The setting of the signal/event mask remains valid for this path until either the path is closed or a new `_os_ss_mv_trigger()` call is issued for this path.

Bits	Name	Description
0	DER	Sets signal when data error detected.
1	PIC	Sets signal when picture displayed.
2	GOP	Sets signal on group of pictures.
3	SOS	Sets signal on start of sequence.
4	LPD	Sets signal when last picture displayed.
5	CNP	Not in use.
6	EOI	Sets signal when end of program is detected at input.
7	EOS	Sets signal when end of sequence is detected at input.
8	BUF	Sets signal when buffer underflow is detected.
9	NIS	Sets signal when new sequence parameters are found.
Bits	**Name**	**Description (continued)**
10		Reserved and must be zero.
11-15		Sets signal base: upper 5 bits of 16-bit signal to send (value must be between 00001 and 11111 binary).

The `NIS` event indicates that either one or both of the values stored in the `md_imgsize` or `md_picrt` fields of the

MPEG MVM were changed by the full-motion system because the MPEG decoder found new values in the MPEG video stream. Current values (if any) are still available in the MPEG video status block fields for `mvs_imgsize` and `mvs_picrt`.

An `EOI` event is generated when an ISO-1172 `MPEG_program_end_code` is detected by the MPEG video decoder, or an EOS event is generated when a `sequence_end_code` is detected by the MPEG video decoder. When the input stream is an ISO/IEC 13818-1 transport stream, the `EOI` event can be generated after the video decoder is out of data for a period of time. The length of this time is under the decoder driver's discretion.

The LPD event indicates that the last picture of an MPEG video sequence is about to appear on the output of the MPEG video decoder. Generally, this is the last picture in display order, before a `sequence_end_code` is sent by the MPEG video stream.

Non-Fatal Errors

```
EOS_BMODE
EOS_ILLPRM
```

Chapter 2: System Data Structures

The following system data structures are discussed in this chapter:

Motion Video Map (MVM) Descriptor

This data structure is defined in `mpfm.h` with the name `mpvd`. It is created by `_os_ss_mv_create()`.

Table 2-1 MVM Descriptor

Offset	Length	Name	Description
0	2	`md_id`	`md_id` is used in many calls to indicate the map on which the operation should be performed. It is also referred to as the **map ID**.
2	2	`md_type`	This field contains the type of MVM. See `_os_ss_mv_create()`.
4	2	`md_stream`	This field contains the number (0-31) of the selected MPEG video stream.
6	2	`md_pid`	This field contains the MPEG-2 transport video stream Packet ID (PID) (0...0x1fff). For details about the PID, refer to the MPEG-2 transport packet structure specified in the ISO/IEC 13818 DIS recommendation.
8	4	`md_bcol`	The contents of this field specify the format of the border color value. Use `_os_ss_mv_bcolor()` to set this field.

Table 2-1 MVM Descriptor (continued)

Offset	Length	Name	Description
12	4	`md_timecd`	This field contains the time-code taken from the MPEG video stream when the picture displays. The format is `H, M, S, P` where `H` represents hours, `M` minutes, `S` seconds, and `P` picture. `P` may have a value from 0 to the picture rate minus one. Remember, this field is using standard numeric formatting, each byte can contain a value from 0 to 256.
16	2	`md_tmpref`	This field contains the temporal reference taken from the MPEG video stream when a picture displays. Within a group of pictures, it counts from 0 to 1023 and then goes to 0 again. The first picture after a group of pictures header has the temporal reference reset to 0. Remember, this field is using standard numeric formatting, it can contain a value from 0 to 65535.

Table 2-1 MVM Descriptor (continued)

Offset	Length	Name	Description
18	1	`md_picrt`	This field contains the current picture rate which is taken from the MPEG video stream when found. It may have the values shown below: **Value** — **Picture Rate** 23 — 23.976 24 — 24 25 — 25 29 — 29.97 30 — 30
19	1		This field contains the alignment character.
20	8		Reserved

Notes

a. W = Width, H = Height (2 bytes each)
b. H = Horizontal, V = Vertical (2 bytes each)

Motion Audio Map (MAM) Descriptor

This data structure is defined in the `mpfm.h` file with the name of `mpad`. The map is created by `_os_ss_ma_create()`.

Table 2-2 MAM Descriptor

Offset	Length	Name	Description
0	2	`md_id`	This field contains the MAM ID. The `md_id` field is used in many calls to indicate the map on which the operation should be done. It is also referred to as `map ID`.
2	2	`md_type`	This field contains the MAM type. For the specification of this field, see the `_os_ss_ma_create()` function.
4	2	`md_stream`	This field contains the stream number (0...31) of the selected MPEG audio stream.
6	2	`md_pid`	This field contains the MPEG-2 transport audio stream packet ID (0...0x1fff). For details about PID, refer to the MPEG-2 transport packet structure specified in the ISO/IEC 13818 DIS recommendation.

Table 2-2 MAM Descriptor (continued)

Offset	Length	Name	Description
8	1	`md_at_ll`	This field contains the attenuation value for the left-to-left audio path. The value in this field becomes active when this MAM becomes active. If the sound is muted, the attenuation value may be determined by an `_os_gs_ma_status()` call. During `_os_ss_ma_create()`, this field is initialized to 0x80 (no attenuation and muted).
9	1	`md_at_lr`	This field contains the attenuation value for the left-to-right audio path. The value in this field becomes active when this MAM becomes active. If the sound is muted, the attenuation value may be determined by an `_os_gs_ma_status()` call. During `_os_ss_ma_create()`, this field is initialized to 0xff (maximum attenuation and muted).
10	1	`md_at_rr`	This field contains the attenuation value for the right-to-right audio path. The value in this field becomes active when this MAM becomes active. If the sound is muted, the attenuation value may be determined via an `_os_gs_ma_status()` call. During `_os_ss_ma_create()`, this field is initialized to 0x80 (no attenuation and muted).

Table 2-2 MAM Descriptor (continued)

Offset	Length	Name	Description
11	1	`md_at_rl`	This field contains the attenuation value for the right-to-left audio path. The value in this field becomes active when this MAM becomes active. If the sound is muted, the attenuation value may be determined via an `_os_gs_ma_status()` call. During `_os_ss_ma_create()`, this field is initialized to 0xff (maximum attenuation and muted).
12	8		Reserved

Asynchronous Status Block Descriptors

Asynchronous status block data structures are used in `_os_ss_ma_play()` or `_os_ss_mv_play()` to allow MPFM to optionally send a signal and provide status information to the application on the termination of audio or video play. Both audio and video plays use this same type of data structure, named `stat_blk`, but are given different meanings under each context.

Video Asynchronous Status Block Descriptor

Table 2-3 Video Asynchronous Status Block Descriptor

Offset	Length	Name	Description
0	2	asy_stat	This field contains the hardware status bits signal to send on termination. During an MPEG video play, these bits are copied from the hardware status. You must clear the status bits in the asy_stat field before play is started. If the play finishes because of an error (bit 15 ASV_DER in the asy_stat field is set), the asy_sig field is filled with the appropriate error code. **Error Code** — **Description** EOS_WRITE — Overflow EOS_ABORT — An abort EOS_READ, EOS_NOTRDY, or EOS_BUSERR — DMA error
2	2	asy_sig	This field contains the signal number to send to the application when the play operation finishes or a fatal error occurs. If it is 0, no signal is sent when the operation finishes or an error occurs If the play is aborted by an error situation or by the application (via _os_ss_mv_abort()), this field contains the resulting error code.

Figure 2-1 Layout of the asy_stat Field

	ASV_DER	0	ASV_NIS	ASV_BUF	ASV_EOS	ASV_EOI	ASV_CNP	ASV_LPD	ASV_SOS	ASV_GOP	ASV_PIC	ASV_DON
BITS	15	14...10	9	8	7	6	5	4	3	2	1	0

Bit #	Event	Description
`0`	`ASV_DON`	Operation is finished
`1`	`ASV_PIC`	New picture decoded
`2`	`ASV_GOP`	Group of pictures found
`3`	`ASV_SOS`	Start of sequence found
`4`	`ASV_LPD`	Last picture displayed
`5`	`ASV_CNP`	Not used
`6`	`ASV_EOI`	End of program found
`7`	`ASV_EOS`	End of sequence found
`8`	`ASV_BUF`	Buffer underflow found
`9`	`ASV_NIS`	New image or picture rate available in status block
`10-14`	reserved	Reserved bits. Must be 0.
`15`	`ASV_DER`	Fatal error

Note

C level definitions for these bits are located in the `mpfm.h` header file. These bits are not the same as the bit definitions used for audio trigger signals.

The bits `ASV_PIC`, `ASV_GOP`, `ASV_SOS`, and `ASV_LPD` are set on every picture change. These bits reflect the kind of picture currently displayed, where the `ASV_GOP` and `ASV_SOS` bits are set to 1 when the current picture is the first picture in a group of pictures or sequence, respectively.

The ASV_LPD bit is set to one when the last picture of an MPEG video sequence is about to appear on the output of the MPEG video decoder. Generally, this is the last picture, in display order, before the sequence_end_code in an MPEG video stream.

Bits ASV_EOI and ASV_EOS are set to one when the MPEG video data containing an end-of-program or end-of-sequence, respectively, is sent to the video hardware FIFO. The bits are set to 0 on the next data transfer.

Bit ASV_BUF is set to one when the decoder runs out of data (when decoding) and is set to 0 on the next data transfer.

The ASV_NIS bit is set to 1 after the transfer of data that contained sequence parameters different from those currently known in the system. The bit is set to 0 after the next data transfer.

Table 2-4 ASV_DON/ASV_DER Bit Combinations

Bit 0 ASV_DON	Bit 15 ASV_DER	Description
0	0	Not started or playing
0	1	Error detected, play continues
1	0	Play finished normally
1	1	Fatal error occurred, error code in asy_sig

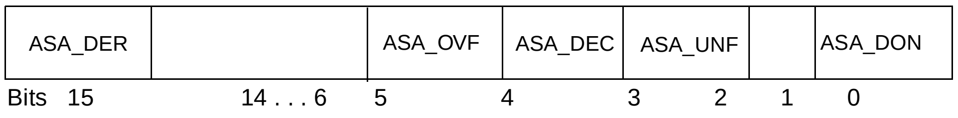
Audio Asynchronous Status Block Offsets

Table 2-5 Audio Asynchronous Status Block Offsets

Offset	Length	Name	Description
0	2	asy_stat	This field contains the current status of the operation. During an MPEG audio play, these bits are copied from the MPEG audio decoder status. You must clear the asy_stat field before play is started. The ASA_UNF bit is set to 1 if, during decoding, the decoder's buffer runs out of data. The bit is set to 0 at the next data transfer. The ASA_DEC bit is set to 1 as long as the decoder is decoding. The ASA_OVF bit is set to 1 if, during a data transfer, the decoders buffer has no room for this data. If such an overflow occurs, the play is aborted.
2	2	asy_sig	This field contains the signal number to send to the application when the audio operation finishes. If asy_sig is zero, no signal is sent when the operation finishes. If the play finishes because of an error (bit 15 ASA_DER in the asy_stat field is set), the asy_sig field is filled with the appropriate error code.

Error Code	Description
EOS_WRITE	Overflow
EOS_ABORT	Abort
EOS_READ, EOS_NOTRDY, or EOS_BUSERR	DMA error

Figure 2-2 Layout of the asy_stat Field



Bit #	Event	Description
`0`	`ASA_DON`	Operation is finished
`1-2`		Not used
`3`	`ASA_UNF`	Buffer underflow
`4`	`ASA_DEC`	Decoding
`5`	`ASA_OVF`	Buffer overflow
`6-14`		Not used
`15`	`ASA_DER`	Fatal error

Note

C level definitions for these bits are located in the `mpfm.h` header file. Note that these bits are not the same as the bit definitions used for video triggers

Video Trigger and Statmask Field Format

The following is the format of the video trigger and statmask field. They are used in the `_os_ss_mv_trigger()` call:

Figure 2-3 Format of the Video Trigger and Statmask Field

NIS	BUF	EOS	EOI	CNP	LPD	SOS	GOP	PIC	DER
9	8	7	6	5	4	3	2	1	0

Bit	Name	Description
`0`	`SV_DER`	Data error detected
`1`	`SV_PIC`	New picture decoded
`2`	`SV_GOP`	Group of pictures found
`3`	`SV_SOS`	Start of sequence found
`4`	`SV_LPD`	Last picture displayed
`5`	`SV_CNP`	Old SCL structure no longer in user
`6`	`SV_EOI`	End of program found in FIFO
`7`	`SV_EOS`	End of sequence found in FIFO
`8`	`SV_BUF`	Buffer underflow found
`9`	`SV_NIS`	New image size or picture rate available in status block
`10`		Reserved — must be 0
`11-15`		Signal base (see `_os_ss_mv_trigger()`)

You may use this template in either of two ways:

- To inform MPFM to send a signal when the corresponding event has occurred by issuing `_os_ss_mv_trigger()`call
- To determine why a signal was received by checking each bit of this template

Audio Trigger and Statmask Field Format

The following is the format of the audio trigger and statmask field. They are used in `_os_ss_ma_trigger()`call:

Figure 2-4 Format of the Audio Trigger and Statmask Field

DER	DEC	BUF	UPD	CSU	EOI
5	4	3	2	1	0

Bit	Name	Description
0	`SA_EOI`	End of program or stream found in FIFO
1	`SA_CSU`	Audio decoder has changed to a new stream
2	`SA_UPD`	Decoder has updated the frame header
3	`SA_BUF`	Buffer underflow found
4	`SA_DEC`	Audio decoder has started decoding
5	`SA_DER`	Data error detected

You may use this template in either of two ways:

- To inform MPFM to send a signal when the corresponding event has occurred
- To determine why a signal was received

MPEG Video Status Block (MVSB)

The MPEG Video Status Block (MVSB) is a data structure containing the information shown in **Table 2-6**. You can use the `_os_gs_mv_status()` call to obtain this information. Bits in all fields are set to 1 as long as the decoder is not started.

Table 2-6 MVSB

Offset	Length	Name	Description
0	2	`mvs_lcntr`	Not used
2	4	`mvs_curadr`	Not used
6	4	`mvs_speed`	Not used
10	4	`mvs_imgsz`	This field contains the image size (width, height).
14	4	`mvs_timecd`	This field contains the picture time-code.
18	2	`mvs_tmpref`	This field contains the picture's temporal reference.
20	2	`mvs_stream`	Not used
22	1	`mvs_picrt`	This field contains the current picture rate.
23	1		Reserved

Table 2-6 MVSB (continued)

Offset	Length	Name	Description
24	4	`mvs_dsc`	This field contains the lower 32 bits of the decoder system clock’s current value in 90 kHz clock resolution.
28	4		Reserved

MPEG Audio Status Block (MASB)

The MPEG Audio Status Block (MASB) is a data structure containing the following information. You can use the `_os_gs_ma_status()` call to obtain this information.

Table 2-7 MASB

Offset	Length	Name	Description
0	2	`mas_stream`	This field contains the MPEG audio stream number that the MPEG audio decoder is currently decoding. If the decoder is not decoding any stream (because the selected stream is not yet available), all bits in this field have a value of 1.
2	4	`mas_att`	This field contains the value of the attenuator. This value does not necessarily match the value set by the user. On some occasions (for example, at `_os_ss_ma_abort()`) the decoder may change the attenuator settings to prevent annoying clicks. The settings as used by the decoder can be found in this field.

Table 2-7 MASB (continued)

Offset	Length	Name	Description
6	4	`mas_head`	This field contains the audio frame header. Each audio frame contains a header describing the nature of the audio stream. Refer to the ISO/IEC 13818-3 committee draft to see the layout and meaning of the fields in the 32-bit header. Before the first header arrives at the decoder, all bits in this field have a value of 1.
10	4	`mas_curadr`	Not used
14	4	`mas_dsc`	This field contains the 32 bits of the audio system decoder clock's current value in 90KHz resolution. Before the decoder receives MPEG audio data, this field is -1.
18	14		Reserved

Index

Product Discrepancy Report

To: Microware Customer Support

FAX: 515-224-1352

From:____

Company:____

Phone:____

Fax:______________________Email:___________________

Product Name:

Description of Problem:

____

____

____

____

____

____

____

____

____

____

Host Platform______________________________________

Target Platform_____________________________________

RadiSys.
MICROWARE SOFTWARE