ALP

Application Programming Interface

Table of contents

General remarks

The ALP hardware comes with a software DLL providing all functions required fur the use of ALP hardware components. The following notes describe general software organization rules applicable for the whole library.

- Any ALP is identified by it's serial number, a number of ALP devices can be simultaneously controlled by a single PC.
- The API software provides an ALP device identifier (type ALP_ID) for each unit.
- The patterns to be displayed are organized in 1...16 bit XGA picture sequences. Any sequence is addressed via a sequence handle (type ALP_ID).
- The sequences are loaded into ALP RAM using an API function and this RAM is not under direct access for the user.
- In general, serial operation is required. The specified ALP has to be free (ALP_READY) for
- any subroutine call to an API function, i.e. must not be allocated by another function.
- As an exception, *AlpProjStart* and *AlpSeqPut* can run simultaneously.
- The inquiry functions *AlpxxxInquire* as well as stopping the ALP operation (*AlpDevHalt*) are always allowed.
- All functions provide a return value (long). The parameter list may point to other output data.

Please, consider Release Notes for current implementation comments.

AlpDevAlloc

Format:

long AlpDevAlloc(long *DeviceNum*, long *InitFlag*, ALP_ID* *DeviceIdPtr*)

Description:

This function allocates an ALP hardware system (board set) and returns an ALP handle so that it can be used by subsequent API functions.

An error is reported if the requested device is not available or not ready.

When you no longer need a particular ALP system, free it using *AlpDevFree.*

Parameters:

DeviceNum - specifies the device to be used. Set this parameter to one of the following values:

ALP_DEFAULT	the next available system is allocated
ALP serial number	the system with the specified serial number is allocated

InitFlag - specifies the type of initialization to perform on the selected system. This parameter can be set to one of the following:

ALP_DEFAULT	default initialization

DeviceIdPtr - specifies the address of the variable in which to write the ALP device identifier.

Return values:

ALP_OK	no errors
ALP_ADDR_INVALID	user data access not valid
ALP_NOT_ONLINE	specified ALP not found
ALP_NOT_READY	specified ALP already allocated
ALP_ERROR_INIT	initialization error

AlpDevControl

Format:

long AlpDevControl (ALP_ID *DeviceId,* long *ControlType*, long *ControlValue*)

Description:

This function is to change the display properties of the ALP. The default values are assigned during device allocation by *AlpDevAlloc.*

Parameters:

DeviceId - ALP device identifier

ControlType - control parameter that is to be modified

ControlValue - value of the parameter

The following settings are possible:

ControlParm	*ControlValue*	*Description*
ALP_TRIGGER_POLARITY	ALP_LEVEL_HIGH or ALP_DEFAULT	active high trigger output signal polarity
	ALP_LEVEL_LOW	active low trigger output signal polarity
ALP_VD_EDGE	ALP_EDGE_FALLING or ALP_DEFAULT	high to low trigger input signal variation
	ALP_EDGE_RISING	low to high trigger input signal variation

Return values:

ALP_OK	no errors
ALP_NOT_AVAILABLE	the specified ALP identifier is not valid
ALP_NOT_READY	the specified ALP is in use by another function
ALP_NOT_IDLE	the specified ALP is not in idle state
ALP_PARM_INVALID	one of the parameters is invalid

AlpDevInquire

Format:

long AlpDevInquire(ALP_ID *DeviceId*, long *InquireType*, long **UserVarPtr*)

Description:

This function inquires about a specified ALP device parameter setting.

Parameter:

DeviceId - ALP device identifier for which the information is requested.

InquireType - specifies the ALP device parameter setting about which to inquire. This parameter can be set to one of the following values:

InquireType	Description
ALP_DEVICE_NUMBER	Serial number of the ALP device
ALP_VERSION	Version number of the ALP device
ALP_DEV_STATE	current ALP status, values possible: ALP_DEV_BUSY: ALP is allocated ALP_DEV_READY: ALP is free for further requests ALP_DEV_IDLE: ALP is in wait state
ALP_AVAIL_MEMORY	ALP on-board sequence memory available for further sequence allocation (AlpSeqAlloc) – number of binary pictures
ALP_TRIGGER_POLARITY	trigger output signal polarity: ALP_LEVEL_HIGH or ALP_LEVEL_LOW
ALP_VD_EDGE	trigger input signal slope: ALP_EDGE_FALLING or ALP_EDGE_RISING

UserVarPtr - specifies the address of the variable in which the requested information is to be written. The variable must be of type long.

Return values:

ALP_OK	no errors
ALP_PARM_INVALID	one of the parameters is invalid
ALP_ADDR_INVALID	user data access not valid

AlpDevHalt

Format:

long AlpDevHalt (ALP_ID *DeviceId)*

Description:

This function is to put the ALP in an idle wait state (ALP_DEV_IDLE). Current sequence display is canceled (ALP_PROJ_IDLE) and the loading of sequences is finished, respectively.

Parameter:

DeviceId - ALP identifier

Return values:

ALP_OK	no errors
ALP_NOT_AVAILABLE	the specified ALP identifier is not valid

AlpDevFree

Format:

long AlpDevFree(ALP_ID DeviceId)

Description:

This function deallocates a previously allocated ALP device. The memory reserved by calling *AlpSeqAlloc* is also released.

The ALP has to be in idle wait state (ALP_DEV_IDLE). Use *AlpDevHalt* function if necessary to put ALP in wait.

Parameter:

DeviceId - ALP identifier for device to be freed.

Return values:

ALP_OK	no errors
ALP_NOT_AVAILABLE	the specified ALP identifier is not valid
ALP_NOT_READY	the specified ALP is in use by another function
ALP_NOT_IDLE	the ALP is not in idle state

AlpSeqAlloc

Format:

long AlpSeqAlloc(ALP_ID *DeviceId,* long *BitPlanes*, long *PicNum*, ALP_ID **SequenceIdPtr*)

Description:

The function provides ALP memory for a sequence of pictures. Any picture has XGA format (1024x768 pixel) and all pictures of a sequence have the same bit depth. The function allocates memory from the ALP board RAM, the user has no direct read/write access. ALP functions provide data transfer using the sequence memory identifier (*SequenceId*) of type ALP_ID.

Pictures can be loaded into the ALP RAM using the *AlpSeqPut* function.

The availability of ALP memory can be tested using the *AlpDevInquire* function.

When a sequence is no longer required, release it, using *AlpSeqFree*.

Parameters:

DeviceId - ALP device identifier

BitPlanes - bit depth of the patterns to be displayed, following values are supported: *1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16*

PicNum - number of XGA pictures belonging to the sequence, the possible values depend upon the available memory (*ALP_AVAIL_MEMORY)* and the bit depth (*BitPlanes)*

SequenceIdPtr - specifies the address of the variable in which the ALP sequence identifier is to be written.

Return values:

ALP_OK	no errors
ALP_NOT_AVAILABLE	the specified ALP identifier is not valid
ALP_NOT_READY	the specified ALP is in use by another function
ALP_PARM_INVALID	one of the parameters is invalid
ALP_ADDR_INVALID	user data access invalid
ALP_MEMORY_FULL	the memory requested is not available

AlpSeqControl

Format:

long AlpSeqControl (ALP_ID *DeviceId,* ALP_ID *SequenceId,* long *ControlType*,
long *ControlValue*)

Description:

This function is to change the display properties of a sequence. The default values are assigned during sequence allocation by *AlpSeqAlloc.*

Parameters:

DeviceId - ALP device identifier

SequenceId - ALP sequence identifier

ControlType - control parameter that is to be modified

ControlValue - value of the parameter

The following settings are possible:

ControlParm	*ControlValue*	*Description*
ALP_SEQ_REPETE	in the non-continuous mode, a started sequence (*AlpProjStart*) can be automatically repeated	
	ALP_DEFAULT	single display of the sequence
	<number> 1 .... N	the sequence is repeated N times
ALP_FIRSTFRAME ALP_LASTFRAME	<picture number> 0 ... *PicNum* – 1	a sequence can be displayed partial, the number of the first picture must be equal or lower than the number of the last picture
ALP_BITNUM	<bit number> 1 ... *BitPlanes*	a sequence can be displayed with reduced bit depth for faster speed

Return values:

ALP_OK	no errors
ALP_NOT_AVAILABLE	the specified ALP identifier is not valid
ALP_NOT_READY	the specified ALP is in use by another function
ALP_PARM_INVALID	one of the parameters is invalid
ALP_SEQ_IN_USE	the sequence specified is currently in use

AlpSeqTiming

Format:

long AlpSeqTiming (ALP_ID *DeviceId,* ALP_ID *SequenceId,* long *IlluminateTime*,
long *PictureTime*, long *TriggerDelay*, long *TriggerPulseWidth*, long *VdDelay*)

Description:

This function controls the timing properties for the sequence display. Default values are assigned during sequence allocation (*AlpSeqAlloc).*

All timing parameters can be inquired using the *AlpSeqInquire* function.

Parameters:

DeviceId	ALP device identifier	
SequenceId	ALP sequence identifier	
	Value	Description
IlluminateTime	duration of the display of one picture in the sequence	
	ALP_DEFAULT	the sequence is displayed with the highest possible rate according to the bit depth
	< microseconds >	time during that a single picture of the sequence is displayed, if the value is too small ALP_DEFAULT is effective, the value can be inquired by *AlpSeqInquire*
PictureTime	time between the start of two consecutive pictures (i.e. the parameter defines the image display rate) with the following relation: <ALP_PICTURE_TIME> = < ALP_ILLUMINATE_TIME> + <ALP_DARK_TIME>	
	ALP_DEFAULT	= ALP_ILLUMINATE_TIME (ALP_DARK_TIME = 0)
	<microseconds>	The time between the start of two pictures in a sequence is increased to the specified value. If the value is too small ALP_DEFAULT is effective. The current value can be inquired by *AlpSeqInquire* .
TriggerDelay	ALP_DEFAULT	0
	<microseconds> 0 ... 100.000	delay of the display start with respect to the trigger output (master mode)
TriggerPulseWidth	ALP_DEFAULT	= ½ * <ALP_ILLUMINATE_TIME>
	<microseconds> 0 ... max	length of the trigger signal, the maximum value is <ALP_PICTURE_TIME>
VdDelay	ALP_DEFAULT	0
	<microseconds> 0 ... 100.000	delay of the start of the display with respect to the VD input signal (slave mode)

Return values:

ALP_OK	no error
ALP_NOT_AVAILABLE	the specified ALP identifier is not valid
ALP_NOT_READY	the specified ALP is in use by another function
ALP_PARM_INVALID	one of the parameters in invalid
ALP_SEQ_IN_USE	the specified sequence is currently in use

AlpSeqInquire

Format:

long AlpSeqInquire (ALP_ID *DeviceId,* ALP_ID *SequenceId,* long *InquireType*,
long **UserVarPtr*)

Description:

This function provides information about the settings of the specified picture sequence. The settings are controlled either during allocation (*AlpSeqAlloc*) or using the *AlpSeqControl* and *AlpSeqTiming* functions, respectively.

Parameters:

DeviceId - ALP device identifier
SequenceId - ALP sequence identifier
InquireType - specifies the sequence parameter setting about which to inquire
UserVarPtr - specifies the address of the variable in which the requested information is to be written

The *InquireType* parameter can be set to one of the following values:

InquireType	*Description*
ALP_BITPLANES	bit depth of the pictures in the sequence
ALP_BITNUM	bit depth for display
ALP_PICNUM	number of pictures in the sequence
ALP_FIRSTFRAME	number of the first picture in the sequence selected for display
ALP_LASTFRAME	number of the last picture in the sequence selected for display
ALP_SEQ_REPETE	number of automatically repeated displays of the sequence
ALP_PICTURE_TIME	time between the start of consecutive pictures in the sequence in µs, the corresponding picture rate [fps] = 1.000.000 / *ALP_PICTURE_TIME* [µs]
ALP_MIN_PICTURE_TIME	minimum time between the start of consecutive pictures
ALP_ILLUMINATE_TIME	duration of the display of one picture in µs
ALP_MIN_ILLUMINATE_TIME	minimum duration of the display of one picture in µs
ALP_TRIGGER_DELAY	delay of the start of picture display with respect to the trigger output (master mode) in µs
ALP_MAX_TRIGGER_DELAY	maximal duration of trigger delay in µs
ALP_TRIGGER_PULSEWIDTH	duration of the output trigger in µs
ALP_VD_DELAY	delay of the start of picture display with respect to the VD trigger input in µs
ALP_MAX_VD_DELAY	maximal duration of VD trigger delay in µs

Return values:

ALP_OK	no errors
ALP_NOT_AVAILABLE	the specified ALP identifier is not valid
ALP_NOT_READY	the specified ALP is in use by another function
ALP_PARM_INVALID	one of the parameters in invalid

AlpSeqPut

Format:

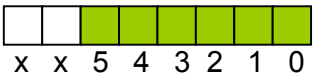
long AlpSeqPut (ALP_ID *DeviceId,* ALP_ID *SequenceId,* long *PicOffset*, long *PicLoad,*
void **UserArrayPtr*)

Description:

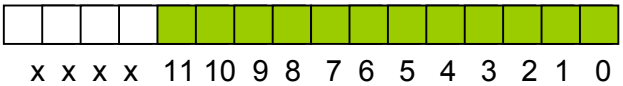
This function allows to load user supplied data via the USB connection into the ALP memory of a previously allocated sequence (*AlpSeqAlloc)* or a part of such a sequence. The loading operation can be running in parallel to the display of *other* sequences. Data can not be loaded into sequences that are currently started for display.
In general, the user provides the data to be loaded in an array of the size *1024 x 768 x PicLoad* bytes. The data format depends upon the bit depth specified for the sequence:

BitPlanes = 1...8: 1 Byte (char unsigned)
BitPlanes = 9...16: 2 Byte (short)
The examples illustrate the use of the *lowest* bit positions (green) :

Example1: *BitPlanes* = 6



Example2: *BitPlanes* = 12



The function is loading *PicNum* pictures into the ALP memory reserved for the specified sequence starting at picture *PicOffset.* The calling program is suspended until the loading operation is completed.

Parameters:

DeviceId - ALP device identifier

SequenceId - ALP sequence identifier

PicOffset - Picture number in the sequence (starting at 0), where the data upload is started, the following values are allowed:

ALP_DEFAULT	0
<picture number>	0 ... *PicNum* – 1

PicLoad - number of pictures that are to be loaded into the sequence memory, the following values are allowed:

ALP_DEFAULT	loading a complete sequence
<number of pictures>	1 ... *PicNum – PicOffset*

UserArrayPtr - pointer to the user data to be loaded
type: *char* or *short*, size: *PicLoad* x 1024 x 768

Return values:

ALP_OK	no errors
ALP_NOT_READY	the specified ALP is invalid or in use by another function
ALP_PARM_INVALID	one of the parameters in invalid
ALP_ERROR_COMM	the ALP has been disconnected during the loading operation. loading is incomplete
ALP_SEQ_IN_USE	a display is started for the sequence to be loaded
ALP_ADDR_INVALID	user data access invalid

AlpSeqFree

Format:

long AlpSeqFree(ALP_ID *DeviceId,* ALP_ID *SequenceId*)

Description:

This function deallocates a previously allocated sequence. The ALP memory reserved for the specified sequence in the device *DeviceId* is released.

Parameters:

DeviceId - ALP identifier

SequenceId - ALP sequence identifier

Return values:

ALP_OK	no error
ALP_NOT_READY	the specified ALP is invalid or in use by another function
ALP_NOT_IDLE	the ALP is not in the idle wait state
ALP_SEQ_IN_USE	the sequence specified is currently in use
ALP_PARM_INVALID	one of the parameters is invalid

AlpProjControl

Format:

long AlpProjControl(ALP_ID *DeviceId,* long *ControlType*, long *ControlValue*)

Description:

This function controls the system parameters that are in effect for all sequences. These parameters are maintained until they are modified again or until the ALP is freed. Default vales are in effect after ALP allocation. All parameters can be read out using the *AlpProjInquire* function.

This function is only allowed if the ALP is in idle wait state (ALP_PROJ_IDLE), that can be enforced by the *AlpProjHalt* function.

Parameters:

DeviceId - ALP identifier

ControlType - name of the control parameter

ControlValue - value of the control parameter

The following settings are possible:

ControlType	*ControlValue*	*Description*
ALP_PROJ_MODE	ALP_MASTER (default)	the ALP operation is controlled by internal timing, a trigger signal is sent out for any picture displayed
	ALP_SLAVE_VD	the ALP operation is controlled by external trigger, the next picture in a sequence is displayed after the detection of an external input trigger (VD or composite-video) signal
ALP_PROJ_SYNC	ALP_SYNCHRONOUS (default)	the calling program gets control back after completion of the sequence display
	ALP_ASYNCHRONOUS	the calling program gets control back immediately

Return values:

ALP_OK	no error
ALP_NOT_READY	the specified ALP is invalid or in use by another function
ALP_PARM_INVALID	one of the parameters in invalid
ALP_NOT_IDLE	the ALP is not in the idle wait state

AlpProjInquire

Format:

long AlpProjInquire (ALP_ID *DeviceId*, long *InquireType*, long **UserVarPtr*)

Description:

This function provides information about the general ALP settings for the sequence display.

Parameters:

DeviceId - ALP device identifier for that the information is inquired

InquireType - property for that the parameter provided. The following values are allowed:

InquireType	*Value*	*Description*
ALP_PROJ_SYNC	*ALP_SYNCHRONOUS* or *ALP_ASYNCHRONOUS*	
ALP_PROJ_MODE	*ALP_MASTER* or *ALP_SLAVE_VD*	
ALP_PROJ_STATE	ALP_PROJ_ACTIVE	ALP projection active
	ALP_PROJ_IDLE	no projection active

UserVarPtr - specifies the address of the variable in which the requested information is to be written

Return values:

ALP_OK	no errors
ALP_NOT_READY	the specified ALP is invalid or in use by another function
ALP_PARM_INVALID	one of the parameters in invalid
ALP_ADDR_INVALID	user data access invalid

AlpProjStart

Format:

long AlpProjStart (ALP_ID *DeviceId,* ALP_ID *SequenceId)*

Description:

A call to this function causes the display of the specified sequence that was previousely loaded by the *AlpSeqPut* function. The sequence is displayed once or with the number of repetitions controlled by *ALP_SEQ_REPETE.*

The calling program gets control back immediately if the ALP is running in asynchronous mode (*ALP_PROJ_SYNC = ALP_ASYNCHRONOUS* and the ALP was in idle wait state (*ALP_PROJ_IDLE*). The ALP state (ALP_PROJ_STATE) can be checked with the *AlpProjInquire* function to guarantee the immediate start of the sequence display. If required, a current display can be interrupted using the *AlpProjHalt* function

If running in synchronous mode (*ALP_PROJ_SYNC = ALP_SYNCHRONOUS*) the calling program is suspended until the picture sequence display is completed.

Parameters:

DeviceId - ALP device identifier

SequenceId - ALP sequence identifier for the sequence to be displayed

Return values:

ALP_OK	no error
ALP_NOT_READY	the specified ALP is invalid or in use by another function
ALP_SEQ_IN_USE	the sequence data are currently loaded (*AlpSeqPut*)
ALP_PARM_INVALID	one of the parameters is invalid

AlpProjStartCont

Format:

long AlpProjStartCont (ALP_ID *DeviceId,* ALP_ID *SequenceId)*

Description:

This function displays the specified sequence in an endless loop.

The calling program gets control back immediately, independent upon the setting in *ALP_PROJ_SYNC*.

The sequence display can be stopped using the *AlpProjHalt* or *AlpDevHalt* , respectively.

Parameters:

DeviceId - ALP device identifier

SequenceId - ALP sequence identifier for the sequence to be displayed

Return values:

ALP_OK	no error
ALP_NOT_READY	the specified ALP is invalid or in use by another function
ALP_PARM_INVALID	one of the parameters is invalid

AlpProjHalt

Format:

long AlpProjHalt (ALP_ID *DeviceId)*

Description:

This function can be used to stop a running sequence display and to set the ALP in idle wait state ALP_PROJ_IDLE.

Parameters:

DeviceId - ALP device identifier

Retrun values:

ALP_OK	no error
ALP_NOT_AVAILABLE	the specified ALP is invalid

AlpProjWait

Format:

long AlpProjWait(ALP_ID *DeviceId)*

Description:

This function is used to wait for the completion of the running sequence display in case of asynchronous ALP mode operation (*ALP_PROJ_SYNC =ALP_ASYNCHRONOUS*).

Using this function in synchronous ALP mode or during the display of an endless loop (*AlpProjStartCont*) causes the ALP_PARM_INVALID error return value.

Parameters:

DeviceId - ALP device identifier

Return values:

ALP_OK	no error
ALP_NOT_READY	the specified ALP is invalid or in use by another function
ALP_PARM_INVALID	one of the parameters is invalid or the ALP is running in asynchronous mode