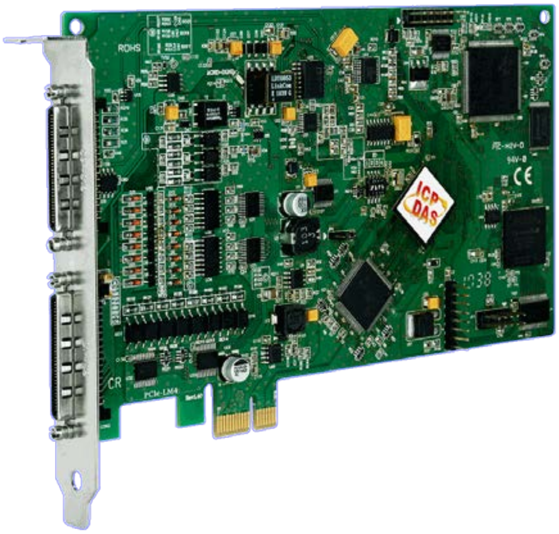
PCIe-LM4 Motion Board
DLL User Manual(Motion)

(Version 1.0)



WARRANTY

All products manufactured by ICP DAS are warranted against defective materials for a period of one year from the date of delivery to the original purchaser.

WARNING

ICP DAS assumes no liability for damages consequent to the use of this product. ICP DAS reserves the right to change this manual at any time without notice. The information furnished by ICP DAS is believed to be accurate and reliable. However, no responsibility is assumed by ICP DAS for its use, nor for any infringements of patents or other rights of third parties resulting from its use.

COPYRIGHT

TRADEMARK

CONTACT US

If you have any questions, please feel free to contact us via email at:
service@icpdas.com
service.icpdas@gmail.com

Revision

Revision	Date	Description	Author
1.00	2020-05-21	Initial version	M. K.

Contents

1 Preface

1.1 Function Overview

The PCIELM4 PCI express board is equipped with four strain gauge input channels, four general analog input channels, a 2-axis motion controller, two analog output channels, sixteen isolated digital input channels and sixteen isolated digital output channels.

The two axes motion controller supports single axis, linear and circular interpolation and continuous interpolation.

The EzGo utility allows the user to configure the PCIELM4 board and execute some basic motion commands.

Sample programs with source code are provided demonstrate the initialization procedure and motion control execution procedure.

The following table gives a short overview of the DLL functions supported by PCIELM4:

Function	Description
Initialization	
pcielm4_open_driver	This function opens and initializes the driver and establishes a connection between the host PC and the PCIELM4 board. This function has to be called first before any other APIs are allowed to be called.
pcielm4_registration	Initializes the motion controller for motion control tasks.
pcielm4_check_card_in_slot	Checks whether the PCIELM4 board with the specified number exists.
pcielm4_reset	Reset and reinitialize the PCIELM4 board.
Axis Digital I/O Function	
pcielm4_set_servo_on	Set the SRV_ON channel output signal to enable/disable the servo drive for pulse command input.
pcielm4_set_erc	The ERC signal sets the servo drive deflection counter to zero.
pcielm4_set_alarm_reset	Reset the servo drive alarm signal (ALM_RST).
pcielm4_set_alarm	Enable/Disable the alarm function and set the active level of the servo alarm signal (ALARM). If the servo drive encounters an abnormality while driving, it sends an signal to the ALARM channel of the PCIELM4. If the servo alarm function of the PCIELM4 has been enabled and the servo alarm signal is active then no pulses will be outputted.
pcielm4_set_inp	Enable/Disable the in-position function and set the active level of the servo in-position signal (INP). In general, when servo drive is set to position mode (P mode), the servo issues a (INP) pulse signal to controller when movement get into position. If the servo in-position function of the PCIELM4 has been enabled, then the controller waits until the in-position signal of the servo has been triggered before

	continuing to execute the next motion command.
pcielm4_set_ready	Enables/Disables the PCIELM4 to check for the servo drive “ready” state and sets the active level of the servo RDY signal.
pcielm4_set_limit	Sets the logic levels of the LMT+ and LMT- channels and the stop mode. The hardware limit signals (LMT+, LMT-) are used for stopping the pulse output when the limit switches are triggered. The hardware limit switches are being used for mechanical protection of the system. If the positive switch (LMT+) is being triggered while the movement is in positive direction the motion stops according to the set stop mode. On the other hand the motion will stop when moving in negative direction and the negative limit switch (LMT-) is active.
pcielm4_get_mdi_status	Reads the current input state of all axis DI channels.
pcielm4_get_servo_on_status	Reads the output signal of the SRV_ON channel. The SRV_ON channel determines whether the servo drive has been enabled to control the motor. The SRV_ON signal is set by “pcielm4_set_servo_on()”.
pcielm4_get_servo_erc_status	Reads the ERC output signal. The ERC signal clears the deviation counter of the servo drive and is set by calling “pcielm4_set_erc()”.
pcielm4_get_servo_almrst_status	Reads the status of the ALM_RST output channel. The ALM_RST signal resets the alarm state of the servo drive. “pcielm4_set_alarm_reset()” sets the output signal.
Motion Control Pulse Setting Function	
pcielm4_set_pls_cfg	Set the pulse output mode for each axis.
pcielm4_set_enc_cfg	Set the parameters of the encoder pulse input.
pcielm4_set_cmdcounter	Set the command counter (position command) value.
pcielm4_get_cmdcounter	Get the current command counter (position command) value.
pcielm4_set_enccounter	Set the encoder counter value.
pcielm4_get_enccounter	Get the current encoder counter value.
pcielm4_set_vring_counter	Set the maximum ring counter position for both the encoder and commanded position counter.
pcielm4_get_vring_counter	Read the maximum ring counter setting.
pcielm4_disable_vring_counter	Disable the ring counter setting.
Automatic Home Search Configuration	
pcielm4_set_home_cfg	Set automatic home search parameters.
Automatic Home Execution	
pcielm4_home_start	Start searching for the home position.
Read Motion Status	
pcielm4_get_motion_done	Read the current motion status of the axis.
pcielm4_get_speed	Get the current axis speed.
pcielm4_get_acc	Get the axis current axis acceleration.
Single Axis Motion Commands	
pcielm4_t_move	Execute a single axis, relative position motion command with a trapezoidal velocity profile (T-curve). The pcielm4_t_move instruction moves the axis the specified travel distance from the current position.
pcielm4_abs_t_move	Execute a single axis, absolute position motion command with a trapezoidal velocity profile (T-curve). The pcielm4_abs_t_move instruction moves the axis to a specified absolute target position. You can execute this instruction even if home is not defined.

pcielm4_s_move	Execute a single axis motion command with an S-curve velocity profile. This command initiates a relative motion. When received, the selected axis will move, with the predefined acceleration and velocity, to a relative position from the current position.
pcielm4_abs_s_move	Execute a single axis, absolute position motion command with an S-curve velocity profile.
pcielm4_velocity_move	Starts a single axis continues pulse driving. Once the axis has reached the driving speed it will indefinitely output pulses at a constant rate until a stop command has been encountered.
Two Axes Linear Interpolation Commands	
pcielm4_t_line2_move	Executes a two axes relative distance linear interpolation motion command with a T-curve velocity profile. The pcielm4_t_line2_move instruction performs linear interpolation for two axes. The target position is specified as a relative position.
pcielm4_abs_t_line2_move	Executes a two axes absolute position interpolation motion command with a T-Curve velocity profile.
pcielm4_s_line2_move	Executes a two axes relative distance interpolation motion command with an S-Curve velocity profile.
pcielm4_abs_s_line2_move	Executes a two axis absolute position interpolation motion command with an S-Curve velocity profile.
Multi-Dimensional Linear Interpolation Commands	
pcielm4_lines_move	Executes a two-dimensional relative position motion command. Positioning is performed on up to two axes with linear interpolation at the specified interpolation speed. The number of interpolation axes can be selected.
pcielm4_abs_lines_move	Executes a two-dimensional absolute position motion command. Positioning is performed on up to two axes with linear interpolation at the specified interpolation speed. The number of interpolation axes can be selected.
Two Dimensional Circular Interpolation Functions	
pcielm4_t_arc2_move	Performs circular interpolation for two axes with a T-curve velocity profile. The center and end position are specified relative to the current position.
pcielm4_abs_t_arc2_move	Executes a two axes circular interpolation motion command with a T-curve velocity profile. The center and target position are specified in absolute position.
Continuous Interpolation Functions	
pcielm4_set_conti_interp_cfg	Assigns the two axis to an interpolation group and sets the group to continuous interpolation mode. Once the group has switch to continuous mode, all the arriving commands are being treated as continuous interpolation commands. In continuous interpolation mode more than one command can be sent at a time. If a new command is being sent while the previous commands is still executing, then the arriving command will first be written to the internal FIFO buffer and starts to executed once the running command has finished. Up to 5000 commands can be stored in the FIFO buffer.
Motion Stop Functions	
pcielm4_stop_move	Stops the current executing motion command for the specified axis. Stops motion before reaching the destination.
pcielm4_set_softlimit	Sets the software limits for the positive and negative direction. Once a software limit position is specified, the PCIELM4 will not accept position commands beyond the

	limit and motion will stop once the limit is hit.
pcielm4_set_softlimit_disable	Disables the axis limits settings.
Multi-Axis Hold/Release Functions	
pcielm4_drv_hold	This command sets the specified axes in holding mode after the current running command has reached its target position. Therefore this instruction takes effect for the next command. The execution of the next command will be put on hold until the “pcielm4_drv_start()” releases the hold operation.
pcielm4_drv_start	Terminates the hold operation. Axes which have been put on hold by “pcielm4_drv_hold()” will continue to execute the next motion command stored in the command FIFO buffer.
Compare Function	
pcielm4_set_compare_trig_cfg	Configures and enables the compare trigger function. The compare function outputs a signal when the compare condition has been met. Two compare modes are being supported: 1. One time compare mode (Single compare mode) 2. Auto increment compare mode.
Latch Function	
pcielm4_set_latch_cfg	Configures and enables position Latch. The latch function captures the encoder counter value at an instant when the latch signal activates. The LTC channel is used to receive the latch pulse. The latch function is hardware implemented and executes at very high speed.
pcielm4_get_latch	Reads the present latched position of the specified axis. Returns the captured position triggered by the latch LTC signal.
Hardware Version	
pcielm4_get_card_version	Gets the PCB and PLD version.
pcielm4_get_fpga_version	Gets the FPGA version.
pcielm4_get_dsp_firmware_version	Gets the current DSP firmware version.
pcielm4_get_dll_version	Gets the DLL version.

Table 1: PCIELM4 DLL functions

1.2 Command Flow Chart

This section illustrates the basic function call sequence required for the initialization of the PCIELM4 and motion command execution.

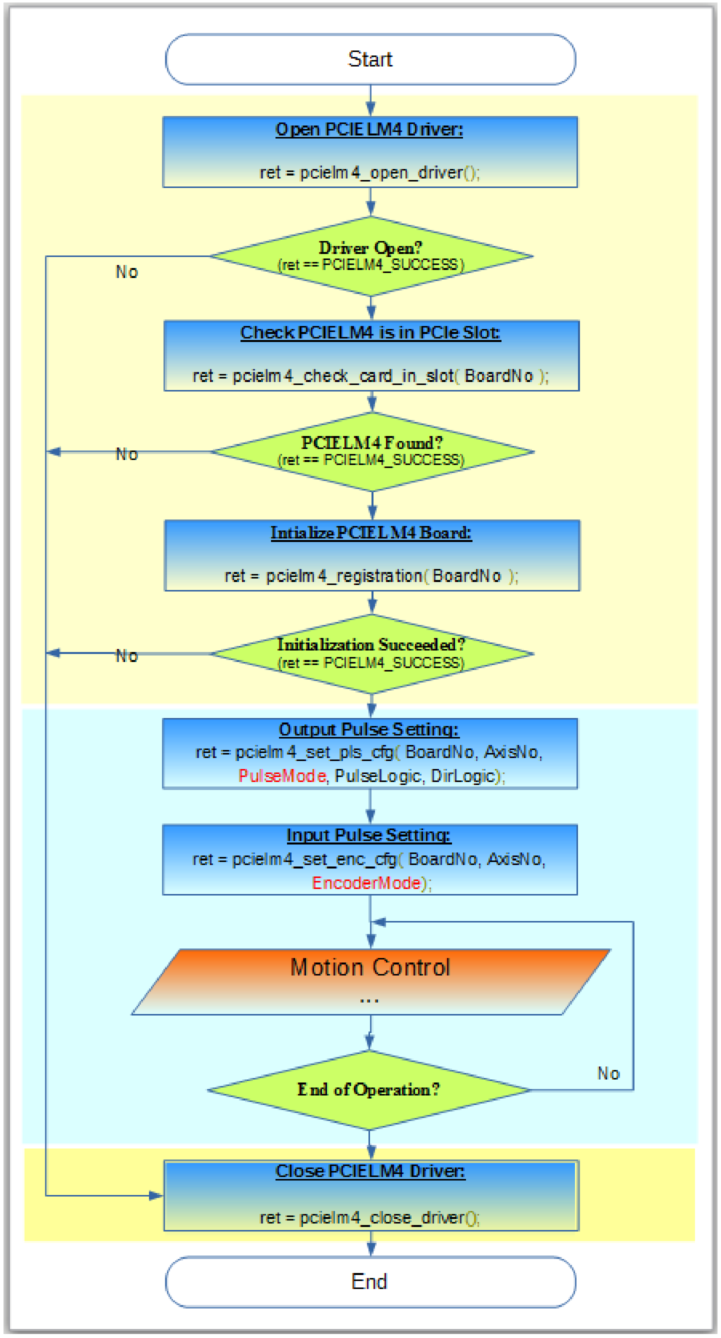

Start
Open PCIELM4 Driver:
ret = pcielm4_open_driver();
Driver Open?
(ret == PCIELM4_SUCCESS)
No
Check PCIELM4 is in PCIe Slot:
ret = pcielm4_check_card_in_slot(BoardNo);
PCIELM4 Found?
(ret == PCIELM4_SUCCESS)
No
Intialize PCIELM4 Board:
ret = pcielm4_registration(BoardNo);
Initialization Succeeded?
(ret == PCIELM4_SUCCESS)
No
Output Pulse Setting:
ret = pcielm4_set_pls_cfg(BoardNo, AxisNo, PulseMode, PulseLogic, DirLogic);
Input Pulse Setting:
ret = pcielm4_set_enc_cfg(BoardNo, AxisNo, EncoderMode);
Motion Control
...
End of Operation?
No
Close PCIELM4 Driver:
ret = pcielm4_close_driver();
End

Figure 1: Initialization procedure

1.2.1 Axis Digital I/O Setting

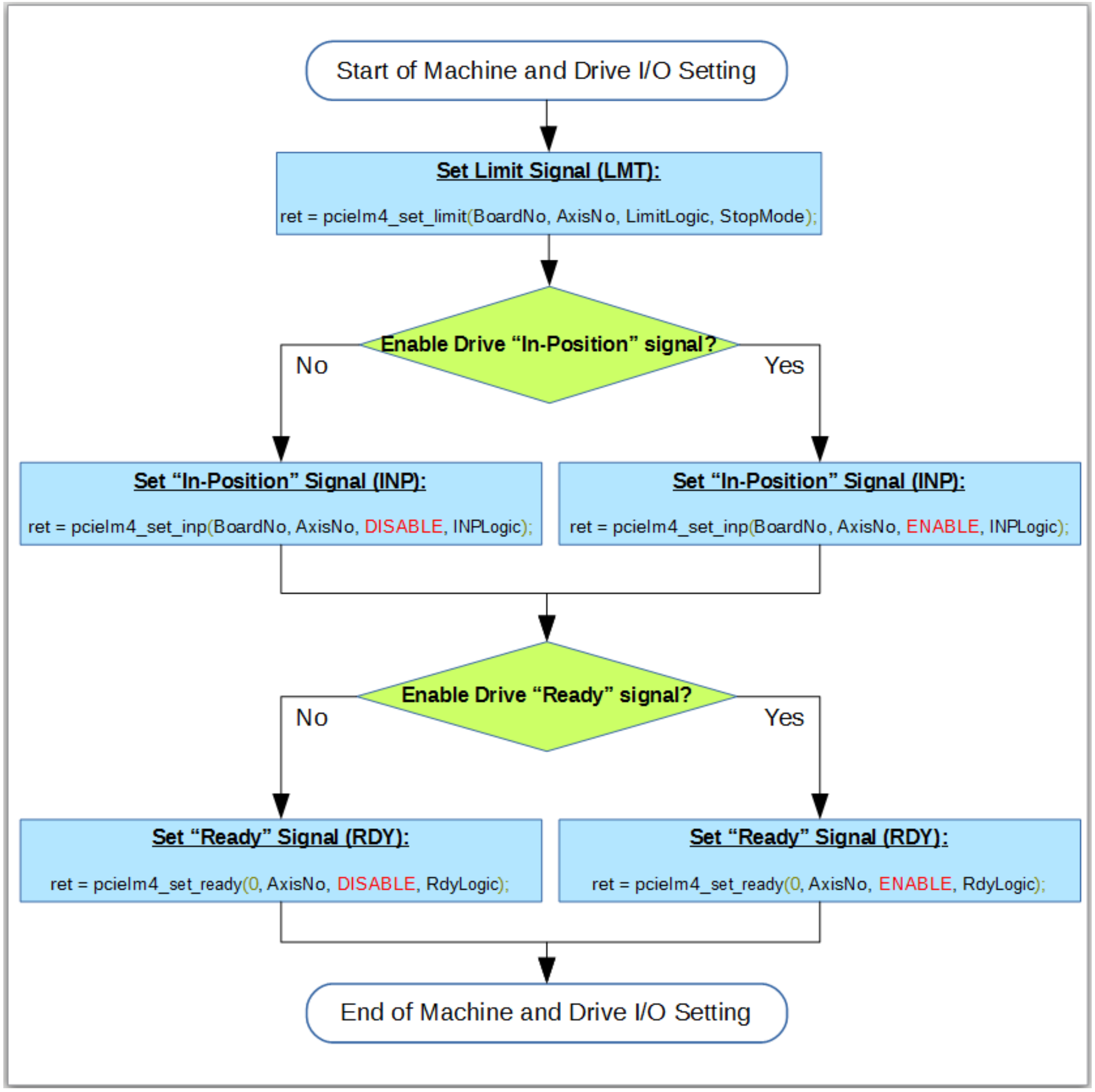

Start of Machine and Drive I/O Setting
Set Limit Signal (LMT):
ret = pcielm4_set_limit(BoardNo, AxisNo, LimitLogic, StopMode);
Enable Drive “In-Position” signal?
No
Yes
Set “In-Position” Signal (INP):
ret = pcielm4_set_inp(BoardNo, AxisNo, DISABLE, INPLogic);
Set “In-Position” Signal (INP):
ret = pcielm4_set_inp(BoardNo, AxisNo, ENABLE, INPLogic);
Enable Drive “Ready” signal?
No
Yes
Set “Ready” Signal (RDY):
ret = pcielm4_set_ready(0, AxisNo, DISABLE, RdyLogic);
Set “Ready” Signal (RDY):
ret = pcielm4_set_ready(0, AxisNo, ENABLE, RdyLogic);
End of Machine and Drive I/O Setting

Figure 2: Axis digital I/O setting

1.2.2 Motion Control Initiation

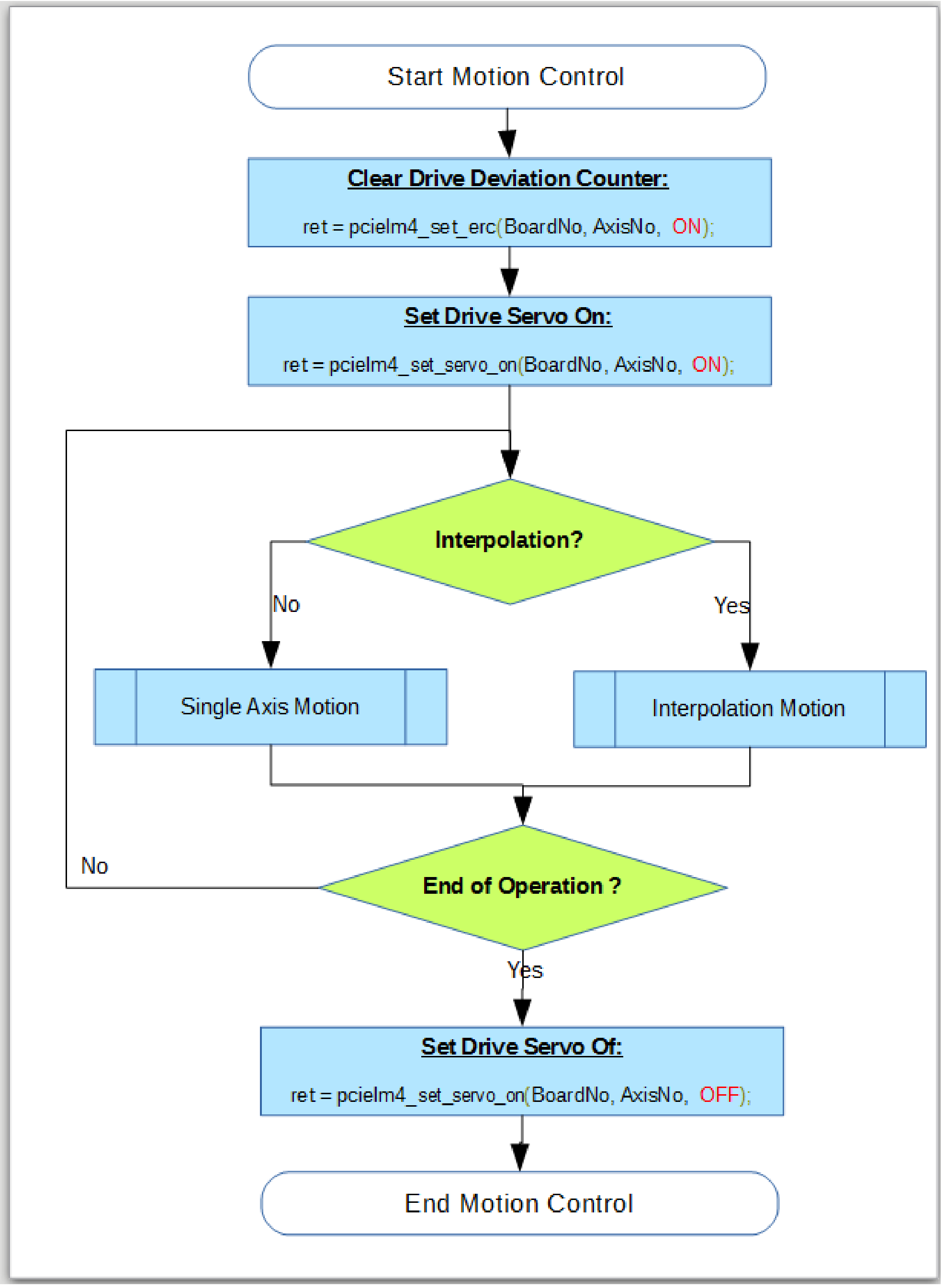

Start Motion Control
Clear Drive Deviation Counter:
ret = pcielm4_set_erc(BoardNo, AxisNo, ON);
Set Drive Servo On:
ret = pcielm4_set_servo_on(BoardNo, AxisNo, ON);
Interpolation?
No
Yes
Single Axis Motion
Interpolation Motion
No
End of Operation ?
Yes
Set Drive Servo Of:
ret = pcielm4_set_servo_on(BoardNo, AxisNo, OFF);
End Motion Control

Figure 3: Initiate motion control

1.2.3 Single Axis Motion Control

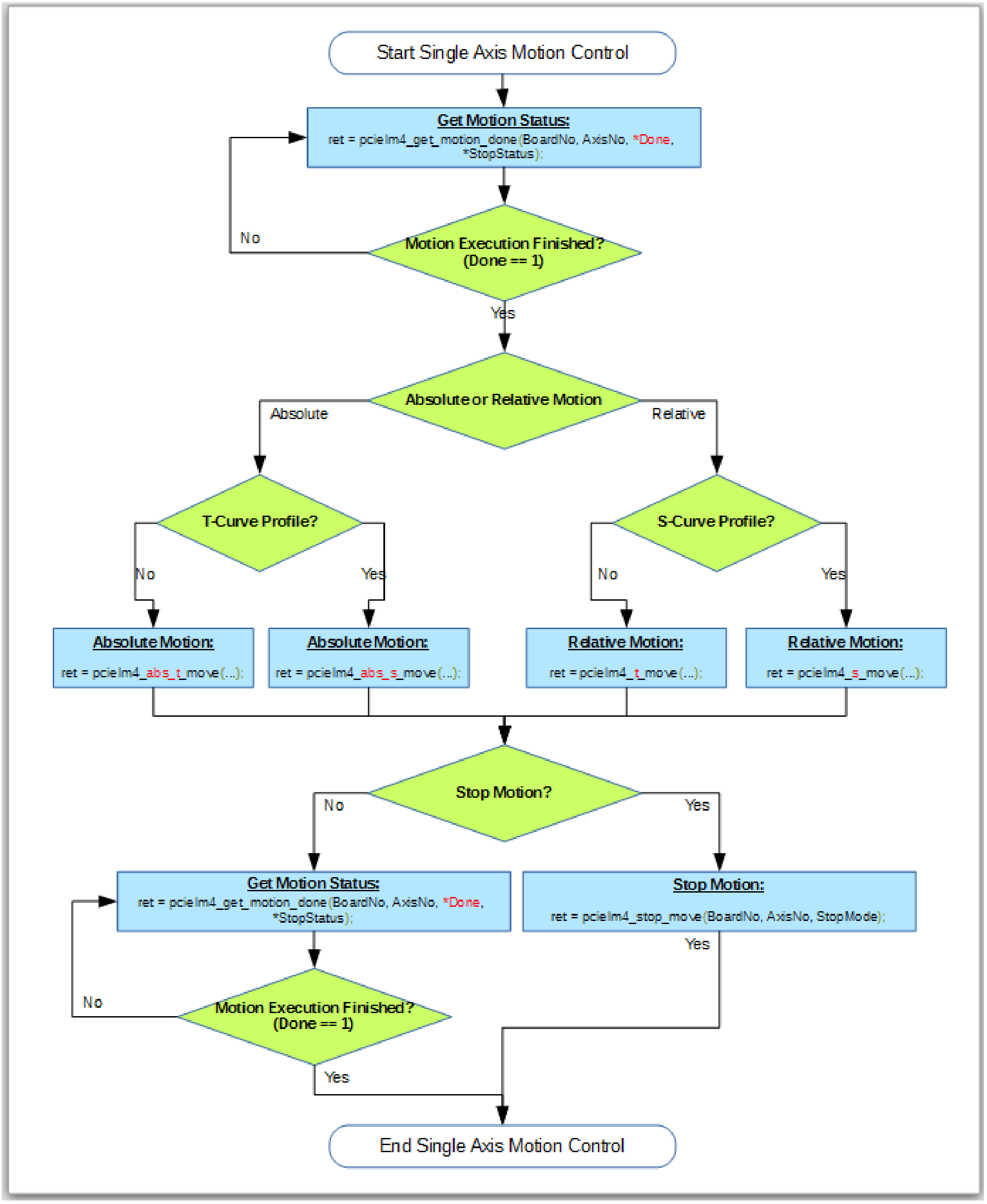

Start Single Axis Motion Control
Get Motion Status:
ret = pcielm4_get_motion_done(BoardNo, AxisNo, *Done, *StopStatus);
No
Motion Execution Finished? (Done == 1)
Yes
Absolute or Relative Motion
Absolute
Relative
T-Curve Profile?
No
Yes
S-Curve Profile?
No
Yes
Absolute Motion:
ret = pcielm4_abs_t_move(...);
Absolute Motion:
ret = pcielm4_abs_s_move(...);
Relative Motion:
ret = pcielm4_t_move(...);
Relative Motion:
ret = pcielm4_s_move(...);
Stop Motion?
No
Yes
Get Motion Status:
ret = pcielm4_get_motion_done(BoardNo, AxisNo, *Done, *StopStatus);
Stop Motion:
ret = pcielm4_stop_move(BoardNo, AxisNo, StopMode);
Yes
No
Motion Execution Finished? (Done == 1)
Yes
End Single Axis Motion Control

Figure 4: Single axis motion control

1.2.4 Interpolation Motion Control

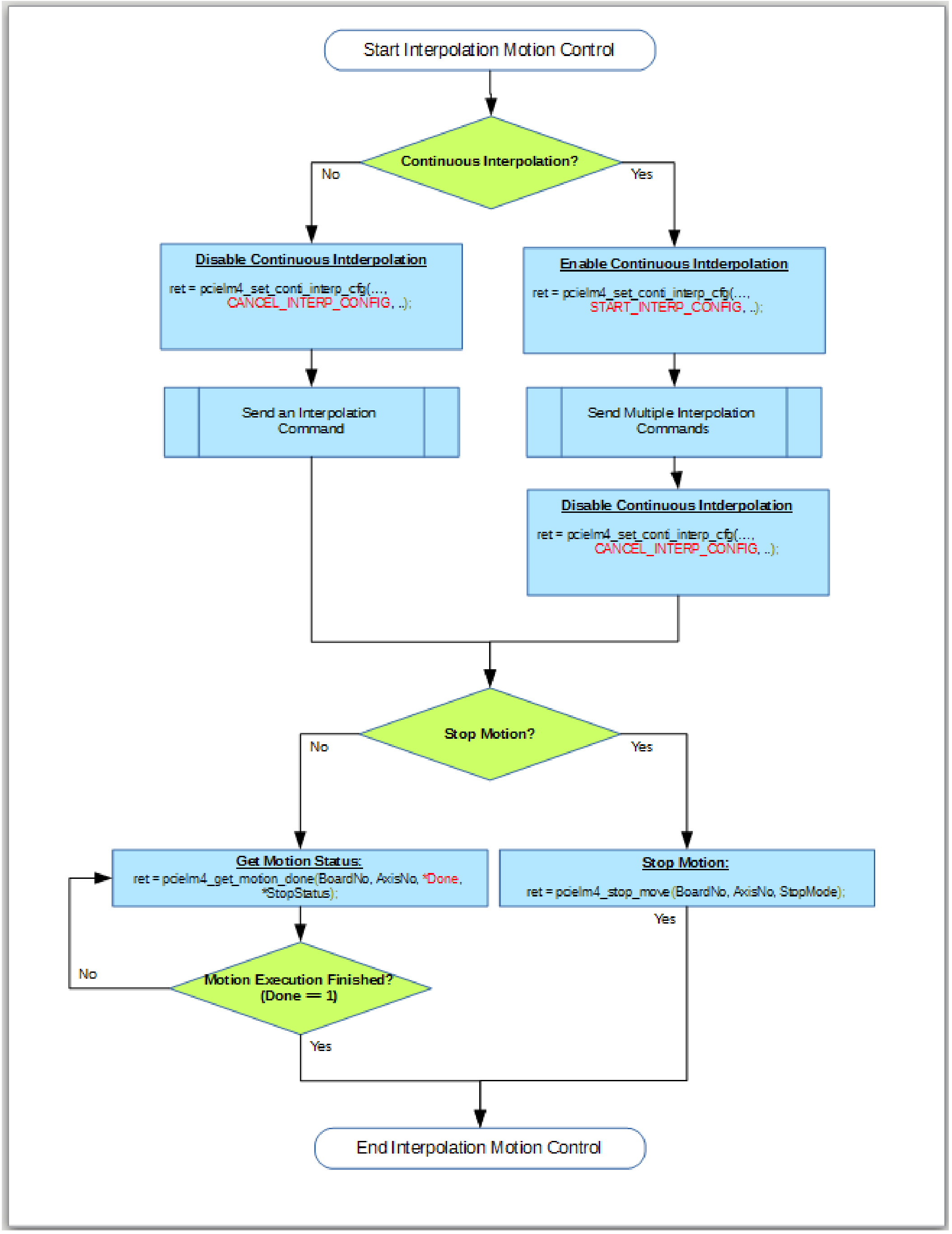

Start Interpolation Motion Control
Continuous Interpolation?
No
Yes
Disable Continuous Intderpolation
ret = pcielm4_set_conti_interp_cfg(...,
CANCEL_INTERP_CONFIG, ..);
Enable Continuous Intderpolation
ret = pcielm4_set_conti_interp_cfg(...,
START_INTERP_CONFIG, ..);
Send an Interpolation Command
Send Multiple Interpolation Commands
Disable Continuous Intderpolation
ret = pcielm4_set_conti_interp_cfg(...,
CANCEL_INTERP_CONFIG, ..);
Stop Motion?
No
Yes
Get Motion Status:
ret = pcielm4_get_motion_done(BoardNo, AxisNo, *Done,
*StopStatus);
Stop Motion:
ret = pcielm4_stop_move(BoardNo, AxisNo, StopMode);
Yes
No
Motion Execution Finished?
(Done == 1)
Yes
End Interpolation Motion Control

Figure 5: Interpolation motion control settings

1.2.5 Initiate Simple Interpolation Motion

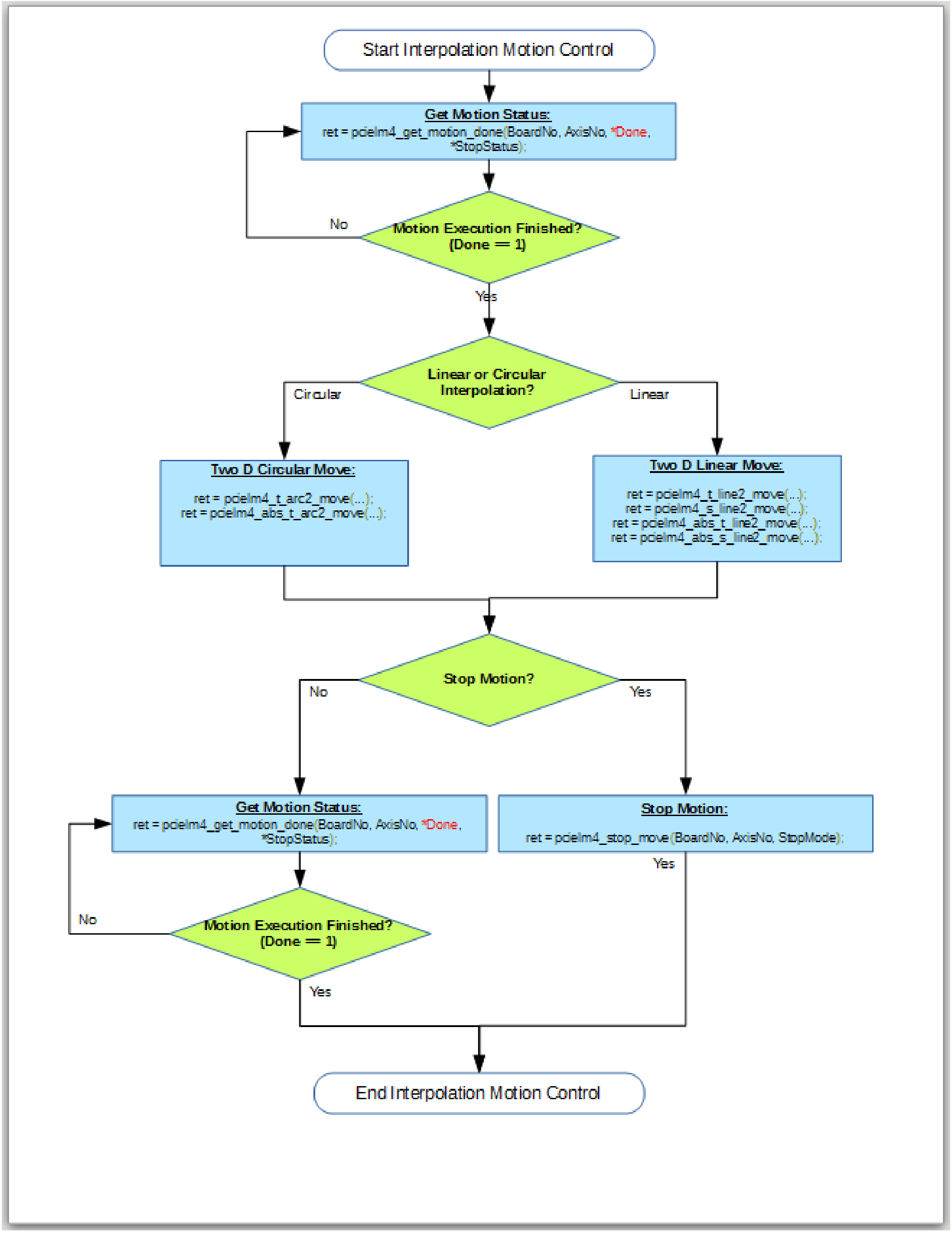
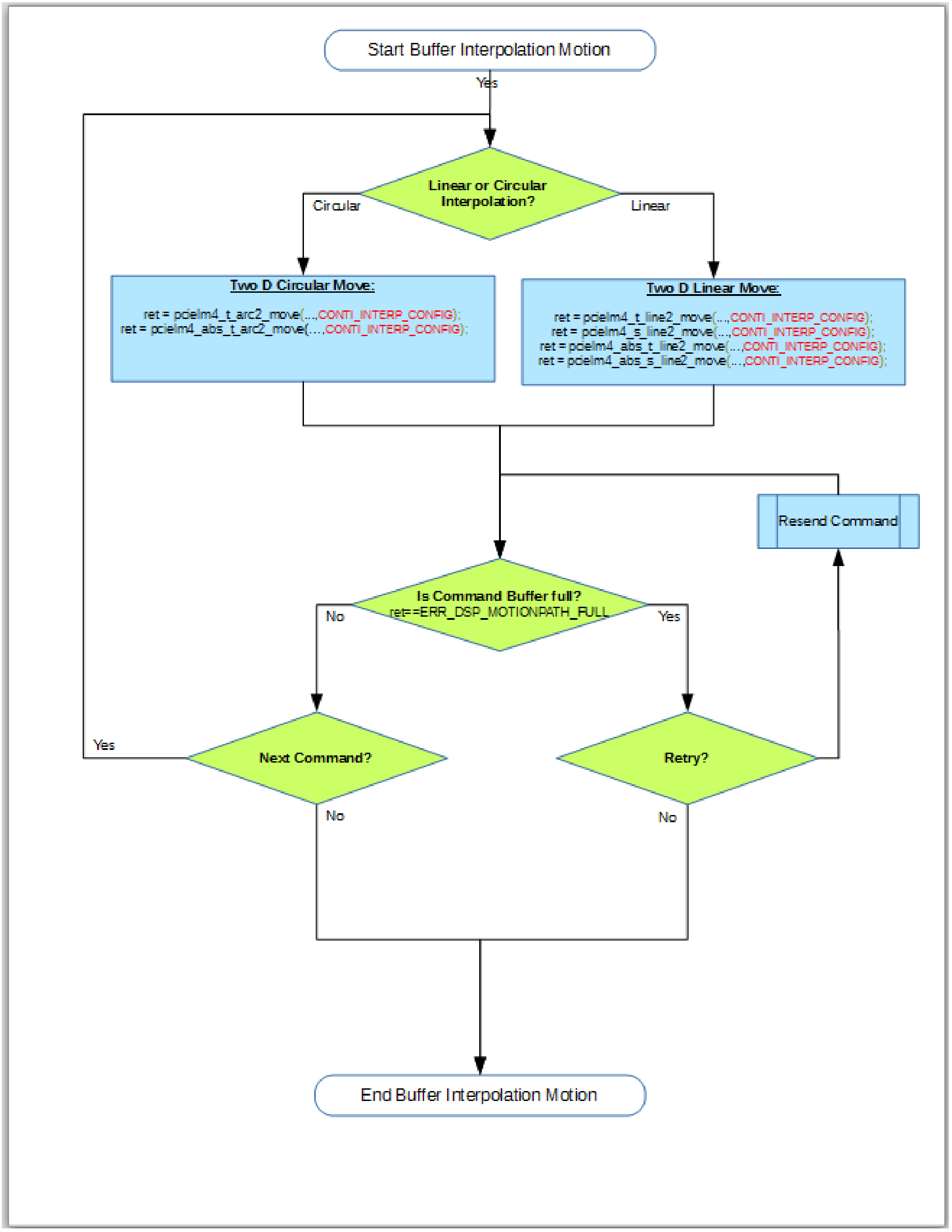

Start Interpolation Motion Control
Get Motion Status:
ret = pcielm4_get_motion_done(BoardNo, AxisNo, *Done, *StopStatus);
Motion Execution Finished? (Done == 1)
No
Yes
Linear or Circular Interpolation?
Circular
Linear
Two D Circular Move:
ret = pcielm4_t_arc2_move(...);
ret = pcielm4_abs_t_arc2_move(...);
Two D Linear Move:
ret = pcielm4_t_line2_move(...);
ret = pcielm4_s_line2_move(...);
ret = pcielm4_abs_t_line2_move(...);
ret = pcielm4_abs_s_line2_move(...);
Stop Motion?
No
Yes
Get Motion Status:
ret = pcielm4_get_motion_done(BoardNo, AxisNo, *Done, *StopStatus);
Stop Motion:
ret = pcielm4_stop_move(BoardNo, AxisNo, StopMode);
Yes
Motion Execution Finished? (Done == 1)
No
Yes
End Interpolation Motion Control

Figure 6: Initiate simple interpolation motion

1.2.6 Initiate Continuous Interpolation Motion


Start Buffer Interpolation Motion
Yes
Linear or Circular Interpolation?
Circular
Linear
Two D Circular Move:
Two D Linear Move:
Resend Command
Is Command Buffer full?
ret==ERR_DSP_MOTIONPATH_FULL
No
Yes
Yes
Next Command?
Retry?
No
No
End Buffer Interpolation Motion

```
ret = pcielm4_t_arc2_move(...,CONTI_INTERP_CONFIG);
ret = pcielm4_abs_t_arc2_move(...,CONTI_INTERP_CONFIG);
```

```
ret = pcielm4_t_line2_move(...,CONTI_INTERP_CONFIG);
ret = pcielm4_s_line2_move(...,CONTI_INTERP_CONFIG);
ret = pcielm4_abs_t_line2_move(...,CONTI_INTERP_CONFIG);
ret = pcielm4_abs_s_line2_move(...,CONTI_INTERP_CONFIG);
```

Figure 7: Initiate continuous interpolation motion

2 System Setting APIs

2.1 Initialization

2.1.1 pcielm4_open_driver

This function opens and initializes the driver and establishes a connection between the host PC and the PCIELM4 board. This function has to be called first before any other APIs are allowed to be called.

Syntax:

```
I32 pcielm4_open_driver (     U16 wBoardNo );
```

Parameters:

Name	Description
wBoardNo:	PCIe board number of PCIELM4 (set via dip switch)

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

- see flow chart Figure 1

2.1.2 pcielm4_registration

The main purpose of this API is to initialize the state machine for motion control. This function has to be executed successfully before calling any other motion control related APIs.

Syntax:

```
I32 pcielm4_registration(      U16 wBoardNo );
```

Parameters:

Name	Description
wBoardNo:	PCIe board number of PCIELM4 (set via dip switch)

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

- This function ensures that the PCIELM4 module is ready for executing motion commands. It is important to call this function before using the any other motion related APIs (Figure 1).

2.1.3 pcielm4_check_card_in_slot

Checks whether the PCIELM4 card is plugged into PCIe slot.

Syntax:

```
I32 pcielm4_check_card_in_slot (     U16 wBoardNo );
```

Parameters:

Name	Description
wBoardNo:	PCIe board number of PCIELM4 (set via dip switch)

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

- This function can be used to scan the PCIe slots for PCIELM4 modules.

2.1.4 pcielm4_reset

Reset and reinitialize the PCIELM4 module.

Syntax:

```
I32 pcielm4_reset (     U16 wBoardNo );
```

Parameters:

Name	Description
wBoardNo:	PCIe board number of PCIELM4 (set via dip switch)

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

- This function internally sets the power off/on the device for a short period of time. During the reset time the PCIELM4 device has no control of the servo drive. Therefore do not call this function when the PCIELM4 needs to be in control of the servo drive or stepping motor.

3 Motion Commands

3.1 Axis Digital I/O Function

The digital IO discussed in this chapter influences the axis operation

3.1.1 pcielm4_set_servo_on

Set the SRV_ON channel output signal to enable/disable the servo drive for pulse command input.

Syntax:

```
I32 pcielm4_set_servo_on (     U16 wBoardNo,
                               U8 bSingleAxis,
                               U8 bServoOnOff );
```

Parameters:

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td><i>wBoardNo:</i></td><td>PCIe board number of PCIELM4 (set via dip switch)</td></tr>
<tr><td><i>bSingleAxis:</i></td><td>Axis definition:
<table>
<tr><th>Axis</th><th>Value</th></tr>
<tr><td>Axis0</td><td>AXIS_0 (0x01)</td></tr>
<tr><td>Axis1</td><td>AXIS_1 (0x02)</td></tr>
</table></td></tr>
<tr><td><i>bServoOnOff:</i></td><td>
<table>
<tr><th>Status</th><th>Value</th></tr>
<tr><td>OFF</td><td>OFF (0x00)</td></tr>
<tr><td>ON</td><td>ON (0x01)</td></tr>
</table></td></tr>
</table>

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

3.1.2 pcielm4_set_erc

The ERC signal sets the servo drive deflection counter to zero.

Syntax:

```
I32 pcielm4_set_erc (    U16 wBoardNo,
                         U8 bSingleAxis,
                         U8 bErcOnOff);
```

Parameters:

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td><i>wBoardNo:</i></td><td>PCIe board number of PCIELM4 (set via dip switch)</td></tr>
<tr><td><i>bSingleAxis:</i></td><td>Axis definition:
<table>
<tr><th>Axis</th><th>Value</th></tr>
<tr><td>Axis0</td><td>AXIS_0 (0x01)</td></tr>
<tr><td>Axis1</td><td>AXIS_1 (0x02)</td></tr>
</table></td></tr>
<tr><td><i>bErcOnOff:</i></td><td>
<table>
<tr><th>Status</th><th>Value</th></tr>
<tr><td>OFF</td><td>OFF (0x00)</td></tr>
<tr><td>ON</td><td>ON (0x01)</td></tr>
</table></td></tr>
</table>

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

- Each servo drive has a deviation counter, which determines the difference between input pulse and feedback pulse. The ERC signal will set the deviation counter of the servo drive to zero and stops the motion if no new pulse command is being issued.

3.1.3 pcielm4_set_alarm_reset

Reset the servo drive alarm signal (ALM_RST).

Syntax:

```
I32 pcielm4_set_alarm_reset (     U16 wBoardNo,
                                  U8 bSingleAxis,
                                  U8 bAlmRstOnOff);
```

Parameters:

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td><i>wBoardNo:</i></td><td>PCIe board number of PCIELM4 (set via dip switch)</td></tr>
<tr><td><i>bSingleAxis:</i></td><td>Axis definition:
<table>
<tr><th>Axis</th><th>Value</th></tr>
<tr><td>Axis0</td><td>AXIS_0 (0x01)</td></tr>
<tr><td>Axis1</td><td>AXIS_1 (0x02)</td></tr>
</table></td></tr>
<tr><td><i>bAlmRstOnOff:</i></td><td>
<table>
<tr><th>Status</th><th>Value</th></tr>
<tr><td>OFF</td><td>OFF (0x00)</td></tr>
<tr><td>ON</td><td>ON (0x01)</td></tr>
</table></td></tr>
</table>

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

- Set the alarm reset DO signal ON to clear the servo drive alarm fault.

3.1.4 pcielm4_set_alarm

Enable/Disable the alarm function and set the active level of the servo alarm signal (ALARM). If the servo drive encounters an abnormality while driving, it sends a signal to the ALARM channel of the PCIELM4. If the servo alarm function has been enabled and the servo alarm signal is active then no pulses will be outputted.

Syntax:

```
I32 pcielm4_set_alarm (    U16 wBoardNo,
                           U8 bSingleAxis,
                           U8 bEnableDisable,
                           U8 bTrigLevel);
```

Parameters:

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td><i>wBoardNo:</i></td><td>PCIe board number of PCIELM4 (set via dip switch)</td></tr>
<tr><td><i>bSingleAxis:</i></td><td>Axis definition:
<table>
<tr><th>Axis</th><th>Value</th></tr>
<tr><td>Axis0</td><td>AXIS_0 (0x01)</td></tr>
<tr><td>Axis1</td><td>AXIS_1 (0x02)</td></tr>
</table></td></tr>
<tr><td><i>bEnableDisable:</i></td><td>Enable the use of the servo alarm signal
<table>
<tr><th>State</th><th>Value</th></tr>
<tr><td>Disable</td><td>DISABLE (0x00)</td></tr>
<tr><td>Enable</td><td>ENABLE (0x01)</td></tr>
</table></td></tr>
<tr><td><i>bTrigLevel:</i></td><td>
<table>
<tr><th>Trigger level</th><th>Value</th></tr>
<tr><td>Active low</td><td>LOGIC_ACTIVE_LOW (0x00)</td></tr>
<tr><td>Active high</td><td>LOGIC_ACTIVE_HIGH (0x01)</td></tr>
</table></td></tr>
</table>

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

3.1.5 pcielm4_set_inp

Enable/Disable the in-position function and set the active level of the servo in-position signal (INP). In general, when the servo drive is set to position mode (P mode), then the servo drive issues a (INP) pulse signal to controller once the motor reaches its target position. If the in-position function has been enabled, then the PCIELM4 waits until the in-position signal of the servo has been triggered before continuing to execute the next motion command.

Syntax:

```
I32 pcielm4_set_inp (     U16 wBoardNo,
                          U8 bSingleAxis,
                          U8 bEnableDisable,
                          U8 bINPLogic);
```

Parameters:

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td><i>wBoardNo:</i></td><td>PCIe board number of PCIELM4 (set via dip switch)</td></tr>
<tr><td><i>bSingleAxis:</i></td><td>Axis definition:
<table>
<tr><th>Axis</th><th>Value</th></tr>
<tr><td>Axis0</td><td>AXIS_0 (0x01)</td></tr>
<tr><td>Axis1</td><td>AXIS_1 (0x02)</td></tr>
</table></td></tr>
<tr><td><i>bEnableDisable:</i></td><td>Enable the use of the servo in-position signal
<table>
<tr><th>State</th><th>Value</th></tr>
<tr><td>Disable</td><td>DISABLE (0x00)</td></tr>
<tr><td>Enable</td><td>ENABLE (0x01)</td></tr>
</table></td></tr>
<tr><td><i>bINPLogic:</i></td><td>
<table>
<tr><th>Trigger level</th><th>Value</th></tr>
<tr><td>Active low</td><td>LOGIC_ACTIVE_LOW (0x00)</td></tr>
<tr><td>Active high</td><td>LOGIC_ACTIVE_HIGH (0x01)</td></tr>
</table></td></tr>
</table>

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

- Call “pcielm4_get_mdi_status()” to get the INP status and “pcielm4_get_motion_done()” to check whether the motion command has finished executing.

3.1.6 pcielm4_set_ready

Enables/Disables the PCIELM4 to check for the servo drive “ready” state and sets the active level of the servo RDY signal.

Syntax:

```
I32 pcielm4_set_ready (    U16 wBoardNo,
                           U8 bSingleAxis,
                           U8 bEnableDisable,
                           U8 bRdyLogic);
```

Parameters:

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td><em>wBoardNo:</em></td><td>PCIe board number of PCIELM4 (set via dip switch)</td></tr>
<tr><td><em>bSingleAxis:</em></td><td>Axis definition:
<table>
<tr><th>Axis</th><th>Value</th></tr>
<tr><td>Axis0</td><td>AXIS_0 (0x01)</td></tr>
<tr><td>Axis1</td><td>AXIS_1 (0x02)</td></tr>
</table></td></tr>
<tr><td><em>bEnableDisable:</em></td><td>Enable the use of the servo ready signal
<table>
<tr><th>State</th><th>Value</th></tr>
<tr><td>Disable</td><td>DISABLE (0x00)</td></tr>
<tr><td>Enable</td><td>ENABLE (0x01)</td></tr>
</table></td></tr>
<tr><td><em>bRdyLogic:</em></td><td>
<table>
<tr><th>Trigger level</th><th>Value</th></tr>
<tr><td>Active low</td><td>LOGIC_ACTIVE_LOW (0x00)</td></tr>
<tr><td>Active high</td><td>LOGIC_ACTIVE_HIGH (0x01)</td></tr>
</table></td></tr>
</table>

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

- If the servo ready function has been enabled and the RDY signal is OFF then the PCIELM4 will not execute a motion command until the RDY signal turns ON.
- Call “pcielm4_get_mdi_status()” to get the RDY status.

3.1.7 pcielm4_set_limit

Sets the logic levels of the LMT+ and LMT- channels and the stop mode. The hardware limit signals (LMT+, LMT-) are used for stopping the pulse output when the limit switches are triggered. The hardware limit switches are being used for mechanical protection of the system. If the positive switch (LMT+) is being triggered while the movement is in positive direction the motion stops according to the set stop mode. On the other hand the motion will stop when moving in negative direction and the negative limit switch (LMT-) is active.

Syntax:

```
I32 pcielm4_set_limit (    U16 wBoardNo,
                           U8 bSingleAxis,
                           U8 bLimitLogic,
                           U8 bStopMode);
```

Parameters:

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td><i>wBoardNo:</i></td><td>PCIe board number of PCIELM4 (set via dip switch)</td></tr>
<tr><td><i>bSingleAxis:</i></td><td>Axis definition:
<table>
<tr><th>Axis</th><th>Value</th></tr>
<tr><td>Axis0</td><td>AXIS_0 (0x01)</td></tr>
<tr><td>Axis1</td><td>AXIS_1 (0x02)</td></tr>
</table></td></tr>
<tr><td><i>bLimitLogic:</i></td><td>
<table>
<tr><th>Trigger level</th><th>Value</th></tr>
<tr><td>Active low</td><td>LOGIC_ACTIVE_LOW (0x00)</td></tr>
<tr><td>Active high</td><td>LOGIC_ACTIVE_HIGH (0x01)</td></tr>
</table></td></tr>
<tr><td><i>bStopMode:</i></td><td>
<table>
<tr><th>Stop Mode</th><th>Value</th></tr>
<tr><td>Disable</td><td>STOP_NONE (0)</td></tr>
<tr><td>Deceleration stop</td><td>STOP_SLOWDOWN (1)</td></tr>
<tr><td>Immediate stop</td><td>STOP_SUDDEN (2)</td></tr>
</table></td></tr>
</table>

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

- If the axis moves in positive direction and triggers the positive limit switch (LMT+)

then the motion will stop but it will not stop when activating the negative limit switch (LMT-) and moving in positive direction. The axis will stop when moving in negative direction and the negative limit switch (LMT-) is active.

- Call “pcielm4_get_mdi_status()” to get the “LMT+” and “LMT-“ status (LMTP, LMTM).

3.1.8 pcielm4_get_mdi_status

Reads the current input state of all digital input channels of the axis.

Syntax:

```
I32 pcielm4_get_mdi_status (      U16 wBoardNo,
                                  U8 bSingleAxis,
                                  U16* pwDIStatus);
```

Parameters:

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td><i>wBoardNo:</i></td><td>PCIe board number of PCIELM4 (set via dip switch)</td></tr>
<tr><td><i>bSingleAxis:</i></td><td>Axis definition:
<table>
<tr><th>Axis</th><th>Value</th></tr>
<tr><td>Axis0</td><td>AXIS_0 (0x01)</td></tr>
<tr><td>Axis1</td><td>AXIS_1 (0x02)</td></tr>
</table></td></tr>
<tr><td><i>pwDIStatus:</i></td><td>
<table>
<tr><th>Bit Position</th><th>Corresponding Signal</th><th>Description</th></tr>
<tr><td>Bit 0</td><td>reserved</td><td></td></tr>
<tr><td>Bit 1</td><td>LMT+</td><td>Positive limit switch</td></tr>
<tr><td>Bit 2</td><td>LMT-</td><td>Negative limit switch</td></tr>
<tr><td>Bit 3</td><td>EMG</td><td>Emergency stop switch</td></tr>
<tr><td>Bit 4</td><td>ALARM</td><td>Servo drive alarm signal</td></tr>
<tr><td>Bit 5</td><td>HOME (ORG)</td><td>Home switch</td></tr>
<tr><td>Bit 6</td><td>SLD (NHOME)</td><td>Slow down switch</td></tr>
<tr><td>Bit 7</td><td>INP</td><td>Servo drive in-position signal</td></tr>
<tr><td>Bit 8</td><td>EZ</td><td>Servo drive Z phase (Index signal)</td></tr>
<tr><td>Bit 9</td><td>RDY</td><td>Servo drive ready signal</td></tr>
<tr><td>Bit 10</td><td>LTC</td><td>Latch input</td></tr>
<tr><td>Bit 11 ~ 15</td><td>reserved</td><td></td></tr>
</table>
If bit is zero: the corresponding signal is OFF
If bit is one: the corresponding signal is ON</td></tr>
</table>

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

Example:

```
I32 ret = 0;
U16 pwDIStatus = 0;

ret = pcielm4_get_mdi_status( 1, AXIS_0, &pwDIStatus );
if (ret == PCIELM4_SUCCESS)
{
      if (pwDIStatus & DI_STATUS_ACTIVE_EMG)
      {MessageBox("Emergency Stop!!");}
      else if (pwDIStatus & DI_STATUS_ACTIVE_LMTP)
      {MessageBox("Reached Positive Limit!!");}
      else if (pwDIStatus & DI_STATUS_ACTIVE_LMTM)
      {MessageBox("Reached Negative Limit!!");}
}
else
{MessageBox("Get IO Status Error !!!");}
```

3.1.9 pcielm4_get_servo_on_status

Reads the output signal of the SRV_ON channel. The SRV_ON channel determines whether the servo drive has been enabled to control the motor. The SRV_ON signal is set by “pcielm4_set_servo_on()”.

Syntax:

```
I32 pcielm4_get_servo_on_status (      U16 wBoardNo,
                                       U8 bSingleAxis,
                                       U8* pbServoOn);
```

Parameters:

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td><i>wBoardNo:</i></td><td>PCIe board number of PCIELM4 (set via dip switch)</td></tr>
<tr><td><i>bSingleAxis:</i></td><td>Axis definition:
<table>
<tr><th>Axis</th><th>Value</th></tr>
<tr><td>Axis0</td><td>AXIS_0 (0x01)</td></tr>
<tr><td>Axis1</td><td>AXIS_1 (0x02)</td></tr>
</table></td></tr>
<tr><td><i>pbServoOn:</i></td><td>
<table>
<tr><th>Status</th><th>Value</th></tr>
<tr><td>OFF</td><td>OFF (0x00)</td></tr>
<tr><td>ON</td><td>ON (0x01)</td></tr>
</table></td></tr>
</table>

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

3.1.10 pcielm4_get_servo_erc_status

Reads the ERC output signal. The ERC signal clears the deviation counter of the servo drive and is set by calling “pcielm4_set_erc()”.

Syntax:

```
I32 pcielm4_get_servo_erc_status (      U16 wBoardNo,
                                        U8 bSingleAxis,
                                        U8* pbServoErc);
```

Parameters:

Name	Description
wBoardNo:	PCIe board number of PCIELM4 (set via dip switch)
bSingleAxis:	Axis definition: Axis | Value Axis0 | AXIS_0 (0x01) Axis1 | AXIS_1 (0x02)
pbServoErc:	Status | Value OFF | OFF (0x00) ON | ON (0x01)

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

3.1.11 pcielm4_get_servo_almrst_status

Reads the status of the ALM_RST output channel. The ALM_RST signal resets the alarm state of the servo drive. “pcielm4_set_alarm_reset()” sets the output signal.

Syntax:

```
I32 pcielm4_get_servo_almrst_status (    U16 wBoardNo,
                                         U8 bSingleAxis,
                                         U8* pbServoAlarm);
```

Parameters:

Name	Description
wBoardNo:	PCIe board number of PCIELM4 (set via dip switch)
bSingleAxis:	Axis definition: Axis | Value Axis0 | AXIS_0 (0x01) Axis1 | AXIS_1 (0x02)
pbServoAlarm:	Status | Value OFF | OFF (0x00) ON | ON (0x01)

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

3.2 Motion Control Pulse Setting Function

3.2.1 pcielm4_set_pls_cfg

Set the pulse output mode for each axis.

Syntax:

```
I32 pcielm4_set_pls_cfg (     U16 wBoardNo,
                              U8 bSingleAxis,
                              U16 wPulseMode,
                              U8 bPulseLogic,
                              U8 bDirectionLogic);
```

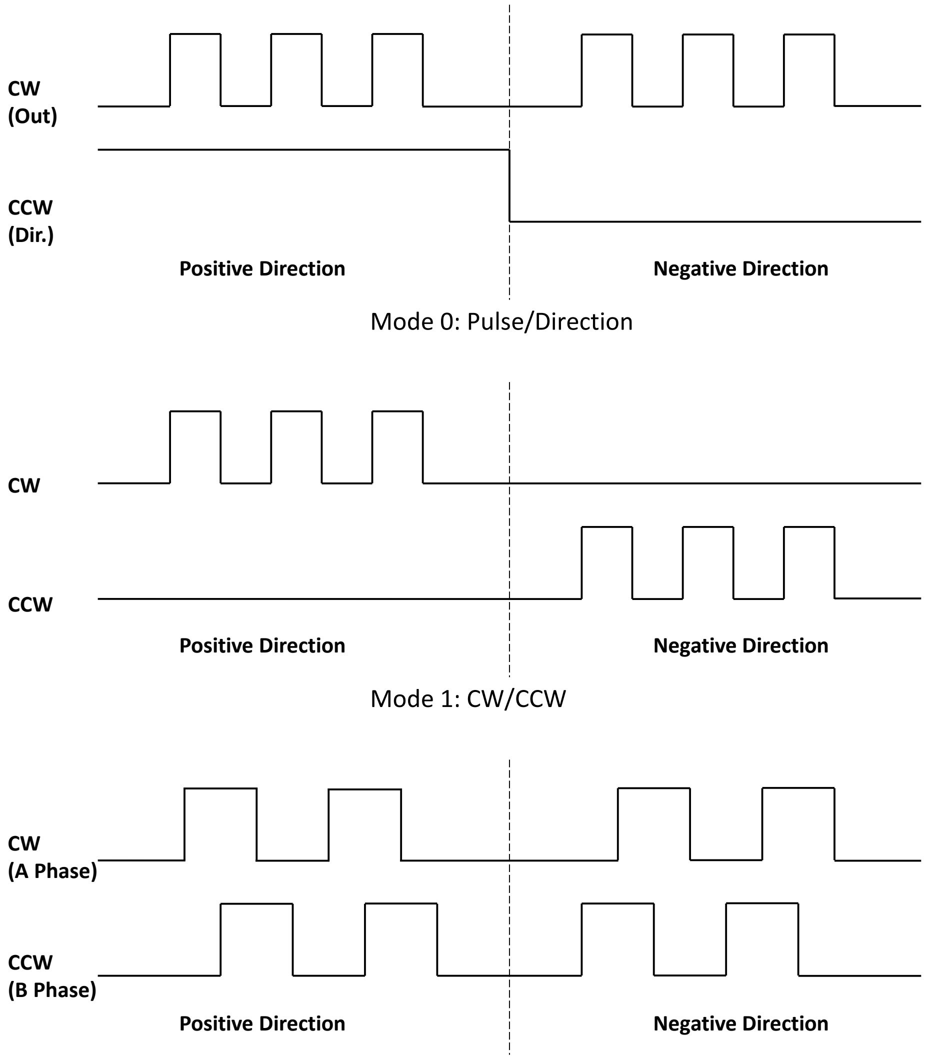
Parameters:

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td><i>wBoardNo:</i></td><td>PCIe board number of PCIELM4 (set via dip switch)</td></tr>
<tr><td><i>bSingleAxis:</i></td><td>Axis definition:
<table>
<tr><th>Axis</th><th>Value</th></tr>
<tr><td>Axis0</td><td>AXIS_0 (0x01)</td></tr>
<tr><td>Axis1</td><td>AXIS_1 (0x02)</td></tr>
</table></td></tr>
<tr><td><i>wPulseMode:</i></td><td>
<table>
<tr><th>Mode</th><th>Value</th></tr>
<tr><td>Pulse/Direction</td><td>PULSE_MODE_PULSE_DIRECTION (0)</td></tr>
<tr><td>CW/CCW</td><td>PULSE_MODE_CW_CCW (1)</td></tr>
<tr><td>A/B Phase</td><td>PULSE_MODE_AB_DIVID_4 (2)</td></tr>
</table></td></tr>
<tr><td><i>bPulseLogic:</i></td><td>
<table>
<tr><th>Logic Level</th><th>Value</th></tr>
<tr><td>Low</td><td>PULSE_LOGIC_ACTIVE_LOW (0x1)</td></tr>
<tr><td>High</td><td>PULSE_LOGIC_ACTIVE_HIGH (0x0)</td></tr>
</table></td></tr>
<tr><td><i>bDirectionLogic:</i></td><td>Direction logic-enable signal level:
<table>
<tr><th>Logic Level</th><th>Value</th></tr>
<tr><td>Low</td><td>PULSE_FORWARD_ACTIVE_LOW (0x1)</td></tr>
<tr><td>High</td><td>PULSE_FORWARD_ACTIVE_HIGH (0x0)</td></tr>
</table>
Note that in "CW / CCW" and "A / B Phase" mode this parameter is invalid</td></tr>
</table>

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:



Mode 0: Pulse/Direction

Mode 1: CW/CCW

Mode 2: AB Phase

Figure 8: Pulse output modes

3.2.2 pcielm4_set_enc_cfg

Set the parameters of the encoder pulse input.

Syntax:

```
I32 pcielm4_set_enc_cfg (     U16 wBoardNo,
                              U8 bSingleAxis,
                              U16 wEncoderMode);
```

Parameters:

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td><em>wBoardNo:</em></td><td>PCIe board number of PCIELM4 (set via dip switch)</td></tr>
<tr><td><em>bSingleAxis:</em></td><td>Axis definition:
<table>
<tr><th>Axis</th><th>Value</th></tr>
<tr><td>Axis0</td><td>AXIS_0 (0x01)</td></tr>
<tr><td>Axis1</td><td>AXIS_1 (0x02)</td></tr>
</table></td></tr>
<tr><td><em>wEncoderMode:</em></td><td>
<table>
<tr><th>Mode</th><th>Value</th></tr>
<tr><td>CW/CCW</td><td>ENCODER_MODE_CW_CCW (1)</td></tr>
<tr><td>A/B Phase</td><td>ENCODER_MODE_AB_DIVID_4 (2)</td></tr>
<tr><td>A/B Phase divide 2</td><td>ENCODER_MODE_AB_DIVID_2 (3)</td></tr>
<tr><td>A/B Phase divide 4</td><td>ENCODER_MODE_AB (4)</td></tr>
</table></td></tr>
</table>

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

3.2.3 pcielm4_set_cmdcounter

Set the command counter (position command) value.

Syntax:

```
I32 pcielm4_set_cmdcounter (        U16 wBoardNo,
                                    U8 bSingleAxis,
                                    I32 lLogicPos );
```

Parameters:

Name	Description
wBoardNo:	PCIe board number of PCIELM4 (set via dip switch)
bSingleAxis:	Axis definition: <table><tr><th>Axis</th><th>Value</th></tr><tr><td>Axis0</td><td>AXIS_0 (0x01)</td></tr><tr><td>Axis1</td><td>AXIS_1 (0x02)</td></tr></table>
lLogicPos:	Command counter value

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

- This function can only be called if no pulses are being outputted.

3.2.4 pcielm4_get_cmdcounter

Get the current command counter (position command) value.

Syntax:

```
I32 pcielm4_get_cmdcounter (      U16 wBoardNo,
                                  U8 bSingleAxis,
                                  I32* plLogicPosCount );
```

Parameters:

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td><em>wBoardNo:</em></td><td>PCIe board number of PCIELM4 (set via dip switch)</td></tr>
<tr><td><em>bSingleAxis:</em></td><td>Axis definition:
<table>
<tr><th>Axis</th><th>Value</th></tr>
<tr><td>Axis0</td><td>AXIS_0 (0x01)</td></tr>
<tr><td>Axis1</td><td>AXIS_1 (0x02)</td></tr>
</table></td></tr>
<tr><td><em>plLogicPosCount:</em></td><td>Pointer to current command counter (position command) value</td></tr>
</table>

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

3.2.5 pcielm4_set_enccounter

Set the encoder counter value.

Syntax:

```
I32 pcielm4_set_enccounter (      U16 wBoardNo,
                                  U8 bSingleAxis,
                                  I32 lEncPos);
```

Parameters:

Name	Description
wBoardNo:	PCIe board number of PCIELM4 (set via dip switch)
bSingleAxis:	Axis definition: Axis / Value Axis0 / AXIS_0 (0x01) Axis1 / AXIS_1 (0x02)
lEncPos:	Encoder counter value

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

3.2.6 pcielm4_get_enccounter

Get the current encoder counter value.

Syntax:

```
I32 pcielm4_get_enccounter (     U16 wBoardNo,
                                 U8 bSingleAxis,
                                 I32* plEncoderPosCount);
```

Parameters:

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td><i>wBoardNo:</i></td><td>PCIe board number of PCIELM4 (set via dip switch)</td></tr>
<tr><td><i>bSingleAxis:</i></td><td>Axis definition:
<table>
<tr><th>Axis</th><th>Value</th></tr>
<tr><td>Axis0</td><td>AXIS_0 (0x01)</td></tr>
<tr><td>Axis1</td><td>AXIS_1 (0x02)</td></tr>
</table></td></tr>
<tr><td><i>plEncoderPosCount:</i></td><td>Pointer to current encoder counter</td></tr>
</table>

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

3.2.7 pcielm4_set_vring_counter

Set the maximum ring counter position for both the encoder and commanded position counter.

Syntax:

```
I32 pcielm4_set_vring_counter (      U16 wBoardNo,
                                     U8 bSingleAxis,
                                     U32 dwRingValue);
```

Parameters:

Name	Description
wBoardNo:	PCIe board number of PCIELM4 (set via dip switch)
bSingleAxis:	Axis definition: Axis / Value Axis0 / AXIS_0 (0x01) Axis1 / AXIS_1 (0x02)
dwRingValue:	The upper limit of the encoder counter value (range: 2~2147483647)

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

- The counter setting will disable the software limit setting
- Ring counter function is not supported when axis is in compare trigger mode
- Use the function "pcielm4_disable_vring_counter()" to turn off the ring counter setting
- The ring position counter operation:

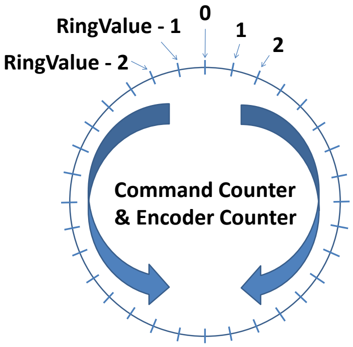


Figure 9: Ring position counter

3.2.8 pcielm4_get_vring_counter

Read the maximum ring counter setting.

Syntax:

```
I32 pcielm4_get_vring_counter (      U16 wBoardNo,
                                     U8 bSingleAxis,
                                     U32* pdwRingValue);
```

Parameters:

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td><i>wBoardNo:</i></td><td>PCIe board number of PCIELM4 (set via dip switch)</td></tr>
<tr><td><i>bSingleAxis:</i></td><td>Axis definition:<table><tr><th>Axis</th><th>Value</th></tr><tr><td>Axis0</td><td>AXIS_0 (0x01)</td></tr><tr><td>Axis1</td><td>AXIS_1 (0x02)</td></tr></table></td></tr>
<tr><td><i>pdwRingValue:</i></td><td><ul><li>Pointer to current ring counter value setting (range: 2~2147483647)</li><li>Returns the value set by “pcielm4_set_vring_counter()”</li></ul></td></tr>
</table>

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

3.2.9 pcielm4_disable_vring_counter

Disable the ring counter setting.

Syntax:

```
I32 pcielm4_disable_vring_counter (      U16 wBoardNo,
                                         U8 bSingleAxis);
```

Parameters:

Name	Description
wBoardNo:	PCIe board number of PCIELM4 (set via dip switch)
bSingleAxis:	Axis definition: Axis | Value Axis0 | AXIS_0 (0x01)

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

- The counter value range will be set back to -2,147,483,648 to 2,147,483,647.

4 Automatic Home Search

4.1 Automatic Home Search Configuration

4.1.1 pcielm4_set_home_cfg

Set automatic home search parameters.

Syntax:

```
I32 pcielm4_set_home_cfg (      U16 wBoardNo,
                                U8 bSingleAxis,
                                U8 bHomeLogic,
                                U8 bNHomeLogic,
                                U8 bIndexLogic,
                                U8 bHomeSteps,
                                I32 lStep4Offset);
```

Parameters:

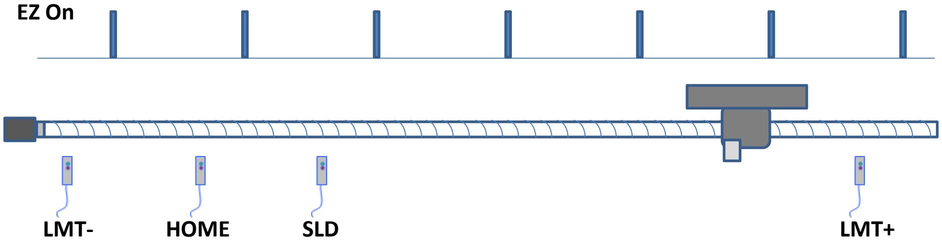
<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td><i>wBoardNo:</i></td><td>PCIe board number of PCIELM4 (set via dip switch)</td></tr>
<tr><td><i>bSingleAxis:</i></td><td>Axis definition:
<table>
<tr><th>Axis</th><th>Value</th></tr>
<tr><td>Axis0</td><td>AXIS_0 (0x01)</td></tr>
<tr><td>Axis1</td><td>AXIS_1 (0x02)</td></tr>
</table></td></tr>
<tr><td><i>bHomeLogic</i>:</td><td>Logic level of origin (HOME) input signal:
<table>
<tr><th>Trigger level</th><th>Value</th></tr>
<tr><td>Active low</td><td>LOGIC_ACTIVE_LOW (0x00)</td></tr>
<tr><td>Active high</td><td>LOGIC_ACTIVE_HIGH (0x01)</td></tr>
</table></td></tr>
<tr><td><i>bNHomeLogic:</i></td><td>Logic level of the slow down (SLD) input signal:
<table>
<tr><th>Trigger level</th><th>Value</th></tr>
<tr><td>Active low</td><td>LOGIC_ACTIVE_LOW (0x00)</td></tr>
<tr><td>Active high</td><td>LOGIC_ACTIVE_HIGH (0x01)</td></tr>
</table></td></tr>
<tr><td><i>bIndexLogic:</i></td><td>Logic level of servo drive Z phase (Index signal)
<table>
<tr><th>Trigger level</th><th>Value</th></tr>
<tr><td>Active low</td><td>LOGIC_ACTIVE_LOW (0x00)</td></tr>
</table></td></tr>
</table>

<table>
<tr><td></td><td><table><tr><td>Active high</td><td>LOGIC_ACTIVE_HIGH (0x01)</td></tr></table></td></tr>
<tr><td><i>bHomeSteps:</i></td><td><table>
<tr><th>Bit Position</th><th>Corresponding Home Step</th></tr>
<tr><td>Bit 1</td><td>Step1: High speed near home search(Search SLD)</td></tr>
<tr><td>Bit 2</td><td>Reserved</td></tr>
<tr><td>Bit 3</td><td>Step2: Low speed home search (Search HOME)</td></tr>
<tr><td>Bit 4</td><td>Reserved</td></tr>
<tr><td>Bit 5</td><td>Step3: Low speed servo drive Z phase (Index signal) search (Search EZ)</td></tr>
<tr><td>Bit 6</td><td>Reserved</td></tr>
<tr><td>Bit 7</td><td>Step4: High speed offset drive (Offset)</td></tr>
<tr><td>Bit 8</td><td>Reserved</td></tr>
</table></td></tr>
<tr><td><i>lStep4Offset:</i></td><td>Offset position</td></tr>
</table>

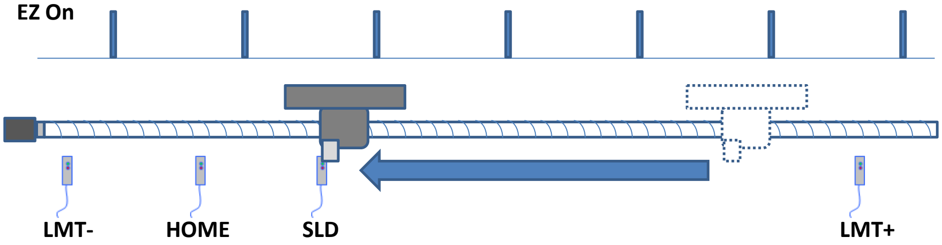
Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

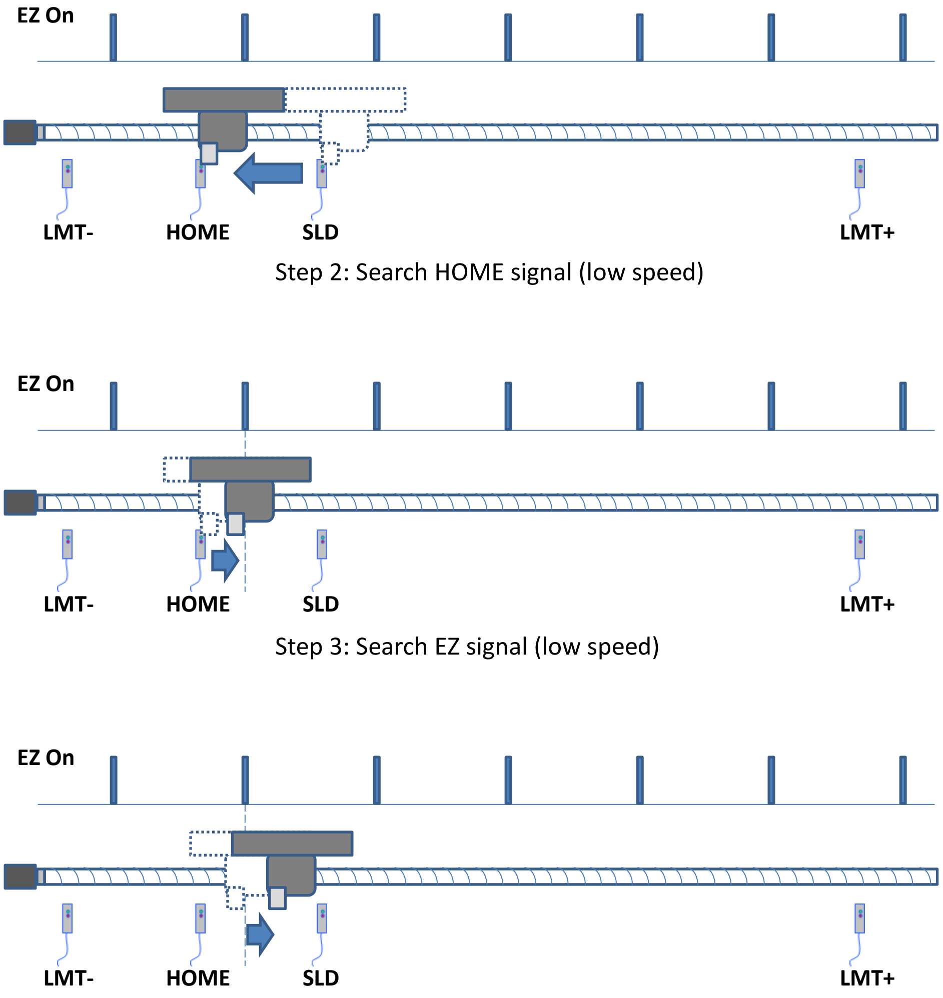
Remarks:



Step 0: Wait auto homing command



Step 1: Search SLD signal (high speed).



Step 2: Search HOME signal (low speed)

Step 3: Search EZ signal (low speed)

Step 4: Offset (high speed).

Figure 10: Automatic home search

4.2 Automatic Home Execution

4.2.1 pcielm4_home_start

Start searching for the home position.

Syntax:

```
I32 pcielm4_home_start (     U16 wBoardNo,
                             U8 bSingleAxis,
                             U32 dwStartSpeed,
                             U32 dwAcceleration,
                             U32 dwDeceleration,
                             U32 dwNHomeSearchSpeed,
                             U32 dwHomeSearchSpeed,
                             U8 bHomingDirection);
```

Parameters:

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td><i>wBoardNo:</i></td><td>PCIe board number of PCIELM4 (set via dip switch)</td></tr>
<tr><td><i>bSingleAxis:</i></td><td>Axis definition:
<table>
<tr><th>Axis</th><th>Value</th></tr>
<tr><td>Axis0</td><td>AXIS_0 (0x01)</td></tr>
<tr><td>Axis1</td><td>AXIS_1 (0x02)</td></tr>
</table></td></tr>
<tr><td><i>dwStartSpeed:</i></td><td>Start speed (PPS)</td></tr>
<tr><td><i>dwAcceleration:</i></td><td>Acceleration (PPS/Sec)</td></tr>
<tr><td><i>dwDeceleration:</i></td><td>Deceleration (PPS/Sec)</td></tr>
<tr><td><i>dwNHomeSearchSpeed:</i></td><td>Near home search (step 1) and offset drive (step 4) speed (PPS) (<i>dwNHomeSearchSpeed</i> > <i>dwHomeSearchSpeed</i>)</td></tr>
<tr><td><i>dwHomeSearchSpeed:</i></td><td>Home search (step 2) and servo drive servo drive Z phase (Index) search (step 3) speed</td></tr>
<tr><td><i>bHomingDirection:</i></td><td>Home search direction
<table>
<tr><th>Direction</th><th>Value</th></tr>
<tr><td>Negative</td><td>AUTO_HOME_REVERSE (0)</td></tr>
<tr><td>Positive</td><td>AUTO_HOME_FORWARD (1)</td></tr>
</table></td></tr>
</table>

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

5 Motion Control Instructions

5.1 Read Motion Status

5.1.1 pcielm4_get_motion_done

Read the current motion status of the axis.

Syntax:

```
I32 pcielm4_get_motion_done (   U16 wBoardNo,
                                U8 bSingleAxis,
                                U8* pbDone,
                                U16* pwStopStatus);
```

Parameters:

Name	Description
wBoardNo:	PCIe board number of PCIELM4 (set via dip switch)
bSingleAxis:	Axis definition: Axis: Value Axis0: AXIS_0 (0x01) Axis1: AXIS_1 (0x02)
pbDone:	0: a motion command is being executed (axis is outputting pulse) 1: motion has finished
pwStopStatus:	Indicates the cause of a motion stop: If *wStopStatus =0* then motion command is still running

Bit	Corresponding stop cause
Bit 0	Command has reached the target position. Motion has finished without any error DRIVE_FINISH_OUTPUT_FIXED_PULSE
Bit 1	Automatic home search has finished (see “pcielm4_home_start()”) DRIVE_FINISH_WITH_AUTO_HOME
Bit 2	Motion command has been interrupted by a stop command (“pcielm4_stop_move()”) DRIVE_FINISH_WITH_STOP_COMMAND
Bit 3	The axis finished outputting pulse commands and waits for the in-position signal of the servo drive (see “pcielm4_set_inp()”) DRIVE_FINISH_WAIT_FOR_INPOS
Bit 4	Motion has been aborted because the maximum positive position has been exceeded (see “pcielm4_set_softlimit()”)

	DRIVE_FINISH_WITH_SW_LIMIT_POSITIVE
Bit 5	Maximum negative position has been exceeded (see “pcielm4_set_softlimit()”) DRIVE_FINISH_WITH_SW_LIMIT_NEGATIVE
Bit 6	The positive limit switch has been activated (LMT+) DRIVE_FINISH_WITH_LIMIT_POSITIVE
Bit 7	The negative limit switch has been activated (LMT-) DRIVE_FINISH_WITH_LIMIT_NEGATIVE
Bit 8	The servo drive alarm signal has been activated (ALM) DRIVE_FINISH_WITH_ALARM
Bit 9	The alarm has been activated (EMG) DRIVE_FINISH_WITH_EMG
Bit 10 ~ Bit 15	reserved

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

5.1.2 pcielm4_get_speed

Get the current axis speed.

Syntax:

```
I32 pcielm4_get_speed (      U16 wBoardNo,
                             U8 bSingleAxis,
                             I32* plSpeed);
```

Parameters:

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td><i>wBoardNo:</i></td><td>PCIe board number of PCIELM4 (set via dip switch)</td></tr>
<tr><td><i>bSingleAxis:</i></td><td>Axis definition:
<table>
<tr><th>Axis</th><th>Value</th></tr>
<tr><td>Axis0</td><td>AXIS_0   (0x01)</td></tr>
<tr><td>Axis1</td><td>AXIS_1   (0x02)</td></tr>
</table></td></tr>
<tr><td><i>plSpeed:</i></td><td>Pointer to current axis speed (PPS)</td></tr>
</table>

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

5.1.3 pcielm4_get_acc

Get the axis current axis acceleration.

Syntax:

```
I32 pcielm4_get_acc (        U16 wBoardNo,
                             U8 bSingleAxis,
                             I32* plAcc);
```

Parameters:

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td><i>wBoardNo:</i></td><td>PCIe board number of PCIELM4 (set via dip switch)</td></tr>
<tr><td><i>bSingleAxis:</i></td><td>Axis definition:
<table>
<tr><th>Axis</th><th>Value</th></tr>
<tr><td>Axis0</td><td>AXIS_0   (0x01)</td></tr>
<tr><td>Axis1</td><td>AXIS_1   (0x02)</td></tr>
</table></td></tr>
<tr><td><i>plAcc:</i></td><td>Pointer to current axis acceleration (PPS/Sec)</td></tr>
</table>

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

5.2 Single Axis Motion Commands

5.2.1 Introduction

This chapter describes the independent axis positioning motion commands. The motion between the specified axes is independent, and each axis follows its own profile. The user specifies the desired absolute position or relative position, acceleration ramp, and deceleration ramp, for each axis. Two speed profiles are being supported: trapezoidal (T) and sinusoidal (S) curve:

T-Curve:

- The drive speed accelerates from the initial speed in a linear form with the specified acceleration slope to the constant driving speed. When the remaining number of output pulses becomes less than the deceleration pulses, deceleration starts. Deceleration continues until the initial speed has been reached and driving stops.

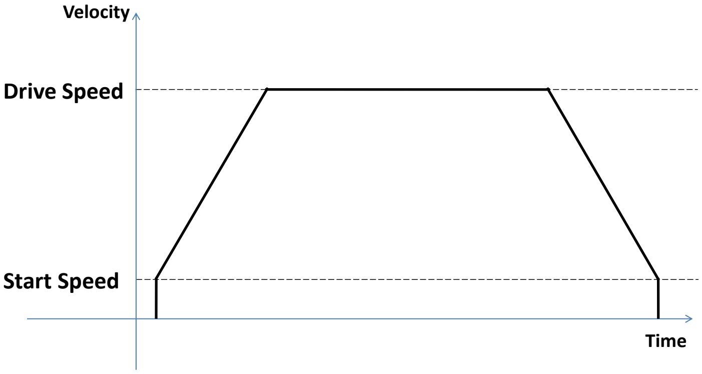


Figure 11: T-Curve velocity profile

S-Curve:

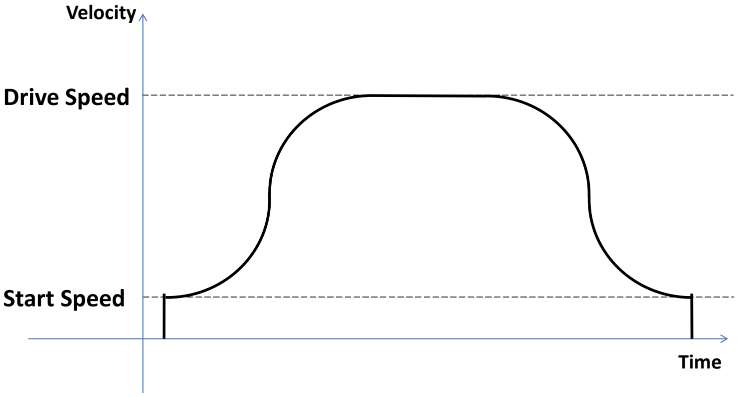


Figure 12: S-Curve velocity profile

5.2.2 pcielm4_t_move

Execute a single axis, relative position motion command with a trapezoidal velocity profile (T-curve). The *pcielm4_t_move* instruction moves the axis the specified travel distance from the current position.

Syntax:

```
I32 pcielm4_t_move (     U16 wBoardNo,
                         U8 bSingleAxis,
                         U32 dwStartSpeed,
                         U32 dwDriveSpeed,
                         U32 dwEndSpeed,
                         U32 dwAcceleration,
                         U32 dwDeceleration,
                         I32  lFixedPulse );
```

Parameters:

Name	Description
wBoardNo:	PCIe board number of PCIELM4 (set via dip switch)
bSingleAxis:	Axis definition: <table><tr><th>Axis</th><th>Value</th></tr><tr><td>Axis0</td><td>AXIS_0 (0x01)</td></tr><tr><td>Axis1</td><td>AXIS_1 (0x02)</td></tr></table>
dwStartSpeed:	Start speed (PPS)
dwDriveSpeed:	Drive speed (PPS)
dwEndSpeed:	End speed (PPS)
dwAcceleration:	Acceleration (PPS/Sec)
dwDeceleration:	Deceleration (PPS/Sec)
lFixedPulse:	Relative moving distance (Pulse) > 0: driving in positive direction < 0: driving in negative direction

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

5.2.3 pcielm4_abs_t_move

Execute a single axis, absolute position motion command with a trapezoidal velocity profile (T-curve). The *pcielm4_abs_t_move* instruction moves the axis to a specified absolute target position. You can execute this instruction even if home is not defined.

Syntax:

```
I32 pcielm4_abs_t_move (     U16 wBoardNo,
                             U8 bSingleAxis,
                             U32 dwStartSpeed,
                             U32 dwDriveSpeed,
                             U32 dwEndSpeed,
                             U32 dwAcceleration,
                             U32 dwDeceleration,
                             I32  lFixedPulse );
```

Parameters:

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td><i>wBoardNo:</i></td><td>PCIe board number of PCIELM4 (set via dip switch)</td></tr>
<tr><td><i>bSingleAxis:</i></td><td>Axis definition:<table><tr><th>Axis</th><th>Value</th></tr><tr><td>Axis0</td><td>AXIS_0   (0x01)</td></tr><tr><td>Axis1</td><td>AXIS_1   (0x02)</td></tr></table></td></tr>
<tr><td><i>dwStartSpeed:</i></td><td>Start speed (PPS)</td></tr>
<tr><td><i>dwDriveSpeed:</i></td><td>Drive speed (PPS)</td></tr>
<tr><td><i>dwEndSpeed:</i></td><td>End speed (PPS)</td></tr>
<tr><td><i>dwAcceleration:</i></td><td>Acceleration (PPS/Sec)</td></tr>
<tr><td><i>dwDeceleration:</i></td><td>Deceleration (PPS/Sec)</td></tr>
<tr><td><i>lFixedPulse:</i></td><td>Absolute position (Pulse)</td></tr>
</table>

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

- The direction is being determined by the relative position between the start position and absolute target position.

5.2.4 pcielm4_s_move

Execute a single axis motion command with an S-curve velocity profile. This command initiates a relative motion. When received, the selected axis will move with the defined acceleration and velocity setting to a relative position from the current position.

Syntax:

```
I32 pcielm4_s_move (    U16 wBoardNo,
                        U8 bSingleAxis,
                        U32 dwStartSpeed,
                        U32 dwDriveSpeed,
                        U32 dwEndSpeed,
                        U32 dwAcceleration,
                        U32 dwDeceleration,
                        I32  lFixedPulse );
```

Parameters:

Name	Description
wBoardNo:	PCIe board number of PCIELM4 (set via dip switch)
bSingleAxis:	Axis definition: Axis | Value Axis0 | AXIS_0 (0x01) Axis1 | AXIS_1 (0x02)
dwStartSpeed:	Start speed (PPS)
dwDriveSpeed:	Drive speed (PPS)
dwEndSpeed:	End speed (PPS)
dwAcceleration:	Acceleration (PPS/Sec)
dwDeceleration:	Deceleration (PPS/Sec)
lFixedPulse:	Relative moving distance (Pulse) > 0: driving in positive direction < 0: driving in negative direction

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

5.2.5 pcielm4_abs_s_move

Execute a single axis, absolute position motion command with an S-curve velocity profile.

Syntax:

```
I32 pcielm4_abs_s_move (   U16 wBoardNo,
                           U8 bSingleAxis,
                           U32 dwStartSpeed,
                           U32 dwDriveSpeed,
                           U32 dwEndSpeed,
                           U32 dwAcceleration,
                           U32 dwDeceleration,
                           I32  lFixedPulse );
```

Parameters:

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td><i>wBoardNo:</i></td><td>PCIe board number of PCIELM4 (set via dip switch)</td></tr>
<tr><td><i>bSingleAxis:</i></td><td>Axis definition:
<table>
<tr><th>Axis</th><th>Value</th></tr>
<tr><td>Axis0</td><td>AXIS_0 (0x01)</td></tr>
<tr><td>Axis1</td><td>AXIS_1 (0x02)</td></tr>
</table></td></tr>
<tr><td><i>dwStartSpeed:</i></td><td>Start speed (PPS)</td></tr>
<tr><td><i>dwDriveSpeed:</i></td><td>Drive speed (PPS)</td></tr>
<tr><td><i>dwEndSpeed:</i></td><td>End speed (PPS)</td></tr>
<tr><td><i>dwAcceleration:</i></td><td>Acceleration (PPS/Sec)</td></tr>
<tr><td><i>dwDeceleration:</i></td><td>Deceleration (PPS/Sec)</td></tr>
<tr><td><i>lFixedPulse:</i></td><td>Absolute position (Pulse)</td></tr>
</table>

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

5.2.6 pcielm4_velocity_move

Starts a single axis continues pulse output driving. Once the axis has reached the driving speed it will indefinitely output pulses at a constant rate until a stop command has been encountered.

Syntax:

```
I32 pcielm4_velocity_move (     U16 wBoardNo,
                                U8 bSingleAxis,
                                U32 dwStartSpeed,
                                U32 dwDriveSpeed,
                                U32 dwAcceleration,
                                U8  bDirection);
```

Parameters:

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td><i>wBoardNo:</i></td><td>PCIe board number of PCIELM4 (set via dip switch)</td></tr>
<tr><td><i>bSingleAxis:</i></td><td>Axis definition:
<table>
<tr><th>Axis</th><th>Value</th></tr>
<tr><td>Axis0</td><td>AXIS_0 (0x01)</td></tr>
<tr><td>Axis1</td><td>AXIS_1 (0x02)</td></tr>
</table></td></tr>
<tr><td><i>dwStartSpeed:</i></td><td>Start speed (PPS)</td></tr>
<tr><td><i>dwDriveSpeed:</i></td><td>Drive speed (PPS)</td></tr>
<tr><td><i>dwAcceleration:</i></td><td>Acceleration (PPS/Sec)</td></tr>
<tr><td><i>bDirection:</i></td><td>Driving direction:
<table>
<tr><th>Direction</th><th>Value</th></tr>
<tr><td>Negative</td><td>MOVE_DIRECTION_REVERSE (0)</td></tr>
<tr><td>Positive</td><td>MOVE_DIRECTION_FORWARD (1)</td></tr>
</table></td></tr>
</table>

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

5.3 Two Axes Linear Interpolation Commands

In linear interpolation mode, motion between the axes is coordinated to maintain the prescribed vector speed, acceleration, and deceleration along the specified path.

5.3.1 pcielm4_t_line2_move

Executes a two axes relative distance linear interpolation motion command with a T-curve velocity profile. The *pcielm4_t_line2_move* instruction performs linear interpolation for two axes. The target position is specified as a relative position.

Syntax:

```
I32 pcielm4_t_line2_move (     U16 wBoardNo,
                               U8 bMainAxis,
                               U8 bSlaveAxis,
                               U32 dwStartSpeed,
                               U32 dwDriveSpeed,
                               U32 dwEndSpeed,
                               U32 dwAcceleration,
                               U32 dwDeceleration,
                               I32 lMainAxisRelDist,
                               I32 lSlaveAxisRelDist,
                               U16 wInterpMode);
```

Parameters:

Name	Description
wBoardNo:	PCIe board number of PCIELM4 (set via dip switch)
bMainAxis:	Axis definition: Axis | Value Axis0 | AXIS_0 (0x01) Axis1 | AXIS_1 (0x02)
bSlaveAxis:	
dwStartSpeed:	Magnitude of start velocity vector (PPS)
dwDriveSpeed:	Magnitude of drive velocity vector (PPS)
dwEndSpeed:	Magnitude of end velocity vector (PPS)
dwAcceleration:	Magnitude of acceleration vector (PPS/Sec)
dwDeceleration:	Magnitude of deceleration vector (PPS/Sec)
lMainAxisRelDist:	Relative distance of the main axis (Pulse) > 0: relative distance in positive direction < 0: relative distance in negative direction

<table>
<tr><td><i>lSlaveAxisRelDist:</i></td><td>Relative distance of the slave axis (Pulse)
> 0: relative distance in positive direction
< 0: relative distance in negative direction</td></tr>
<tr><td><i>wInterpMode:</i></td><td>Command execution mode:
<table>
<tr><th>Mode</th><th>Value</th></tr>
<tr><td>General</td><td>GENERAL_INTERP_CONFIG (0)
• No command buffering takes place.
• A new command can only be executed if the previous command has finished.</td></tr>
<tr><td>Continuous interpolation</td><td>CONTI_INTERP_CONFIG (1)
• Command buffering; up to 5000 command can be stored
• Use this mode to generate a continuous motion path.</td></tr>
<tr><td>Reserved</td><td>CONTI_FIFO_BUFFER_MODE (2)</td></tr>
</table></td></tr>
</table>

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

5.3.2 pcielm4_abs_t_line2_move

Executes a two axes linear interpolation and absolute position motion command with a T-Curve velocity profile.

Syntax:

```
I32 pcielm4_abs_t_line2_move (     U16 wBoardNo,
                                   U8 bMainAxis,
                                   U8 bSlaveAxis,
                                   U32 dwStartSpeed,
                                   U32 dwDriveSpeed,
                                   U32 dwEndSpeed,
                                   U32 dwAcceleration,
                                   U32 dwDeceleration,
                                   I32 lMainAxisFinishPoint,
                                   I32 lSlaveAxisFinishPoint,
                                   U16 wInterpMode);
```

Parameters:

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td><i>wBoardNo:</i></td><td>PCIe board number of PCIELM4 (set via dip switch)</td></tr>
<tr><td><i>bMainAxis:</i></td><td rowspan="2">Axis definition:
<table>
<tr><th>Axis</th><th>Value</th></tr>
<tr><td>Axis0</td><td>AXIS_0 (0x01)</td></tr>
<tr><td>Axis1</td><td>AXIS_1 (0x02)</td></tr>
</table></td></tr>
<tr><td><i>bSlaveAxis:</i></td></tr>
<tr><td><i>dwStartSpeed:</i></td><td>Magnitude of start velocity vector (PPS)</td></tr>
<tr><td><i>dwDriveSpeed:</i></td><td>Magnitude of drive velocity vector (PPS)</td></tr>
<tr><td><i>dwEndSpeed:</i></td><td>Magnitude of end velocity vector (PPS)</td></tr>
<tr><td><i>dwAcceleration:</i></td><td>Magnitude of acceleration vector (PPS/Sec)</td></tr>
<tr><td><i>dwDeceleration:</i></td><td>Magnitude of deceleration vector (PPS/Sec)</td></tr>
<tr><td><i>lMainAxisFinishPoint:</i></td><td>Absolute end position of the main axis (Pulse)</td></tr>
<tr><td><i>lSlaveAxisFinishPoint:</i></td><td>Absolute end position of the slave axis (Pulse)</td></tr>
<tr><td><i>wInterpMode:</i></td><td>Command execution mode:
<table>
<tr><th>Mode</th><th>Value</th></tr>
<tr><td>General</td><td>GENERAL_INTERP_CONFIG (0)
• No command buffering takes place.
• A new command can only be executed if the previous command has finished.</td></tr>
<tr><td>Continuous interpolation</td><td>CONTI_INTERP_CONFIG (1)
• Command buffering; up to 5000 command can be stored.
• Use this mode to generate a continuous motion path.</td></tr>
</table></td></tr>
</table>

	Reserved	CONTI_FIFO_BUFFER_MODE (2)

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

- Flow chart: see Figure 5

5.3.3 pcielm4_s_line2_move

Executes a two axes linear relative distance interpolation motion command with an S-Curve velocity profile.

Syntax:

```
I32 pcielm4_s_line2_move (     U16 wBoardNo,
                               U8 bMainAxis,
                               U8 bSlaveAxis,
                               U32 dwStartSpeed,
                               U32 dwDriveSpeed,
                               U32 dwEndSpeed,
                               U32 dwAcceleration,
                               U32 dwDeceleration,
                               I32 lMainAxisRelDist,
                               I32 lSlaveAxisRelDist,
                               U16 wInterpMode);
```

Parameters:

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td><i>wBoardNo:</i></td><td>PCIe board number of PCIELM4 (set via dip switch)</td></tr>
<tr><td><i>bMainAxis:</i>
<i>bSlaveAxis:</i></td><td>Axis definition:<table><tr><th>Axis</th><th>Value</th></tr><tr><td>Axis0</td><td>AXIS_0 (0x01)</td></tr><tr><td>Axis1</td><td>AXIS_1 (0x02)</td></tr></table></td></tr>
<tr><td><i>dwStartSpeed:</i></td><td>Magnitude of start velocity vector (PPS)</td></tr>
<tr><td><i>dwDriveSpeed:</i></td><td>Magnitude of drive velocity vector (PPS)</td></tr>
<tr><td><i>dwEndSpeed:</i></td><td>Magnitude of end velocity vector (PPS)</td></tr>
<tr><td><i>dwAcceleration:</i></td><td>Magnitude of acceleration vector (PPS/Sec)</td></tr>
<tr><td><i>dwDeceleration:</i></td><td>Magnitude of deceleration vector (PPS/Sec)</td></tr>
<tr><td><i>lMainAxisRelDist:</i></td><td>Relative distance of the main axis (Pulse)
> 0: relative distance in positive direction
< 0: relative distance in negative direction</td></tr>
<tr><td><i>lSlaveAxisRelDist:</i></td><td>Relative distance of the slave axis (Pulse)
> 0: relative distance in positive direction
< 0: relative distance in negative direction</td></tr>
<tr><td><i>wInterpMode:</i></td><td>Command execution mode:<table><tr><th>Mode</th><th>Value</th></tr><tr><td>General</td><td>GENERAL_INTERP_CONFIG (0)
• No command buffering takes place.
• A new command can only be executed if the previous</td></tr></table></td></tr>
</table>

		command has finished.
	Continuous interpolation	CONTI_INTERP_CONFIG (1) • Command buffering; up to 5000 command can be stored • Use this mode to generate a continuous motion path.
	Reserved	CONTI_FIFO_BUFFER_MODE (2)

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

5.3.4 pcielm4_abs_s_line2_move

Executes a two axis absolute position interpolation motion command with an S-Curve velocity profile.

Syntax:

```
I32 pcielm4_abs_s_line2_move (     U16 wBoardNo,
                                   U8 bMainAxis,
                                   U8 bSlaveAxis,
                                   U32 dwStartSpeed,
                                   U32 dwDriveSpeed,
                                   U32 dwEndSpeed,
                                   U32 dwAcceleration,
                                   U32 dwDeceleration,
                                   I32 lMainAxisFinishPoint,
                                   I32 lSlaveAxisFinishPoint,
                                   U16 wInterpMode);
```

Parameters:

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td><i>wBoardNo:</i></td><td>PCIe board number of PCIELM4 (set via dip switch)</td></tr>
<tr><td><i>bMainAxis:</i></td><td rowspan="2">Axis definition:
<table>
<tr><th>Axis</th><th>Value</th></tr>
<tr><td>Axis0</td><td>AXIS_0 (0x01)</td></tr>
<tr><td>Axis1</td><td>AXIS_1 (0x02)</td></tr>
</table></td></tr>
<tr><td><i>bSlaveAxis:</i></td></tr>
<tr><td><i>dwStartSpeed:</i></td><td>Magnitude of start velocity vector (PPS)</td></tr>
<tr><td><i>dwDriveSpeed:</i></td><td>Magnitude of drive velocity vector (PPS)</td></tr>
<tr><td><i>dwEndSpeed:</i></td><td>Magnitude of end velocity vector (PPS)</td></tr>
<tr><td><i>dwAcceleration:</i></td><td>Magnitude of acceleration vector (PPS/Sec)</td></tr>
<tr><td><i>dwDeceleration:</i></td><td>Magnitude of deceleration vector (PPS/Sec)</td></tr>
<tr><td><i>lMainAxisFinishPoint:</i></td><td>Absolute end position of the main axis (Pulse)</td></tr>
<tr><td><i>lSlaveAxisFinishPoint:</i></td><td>Absolute end position of the slave axis (Pulse)</td></tr>
<tr><td><i>wInterpMode:</i></td><td>Command execution mode:
<table>
<tr><th>Mode</th><th>Value</th></tr>
<tr><td>General</td><td>GENERAL_INTERP_CONFIG (0)
• No command buffering takes place.
• A new command can only be executed if the previous command has finished.</td></tr>
<tr><td>Continuous interpolation</td><td>CONTI_INTERP_CONFIG (1)
• Command buffering; up to 5000 command can be stored.
• Use this mode to generate a continuous motion path.</td></tr>
</table></td></tr>
</table>

	Reserved	CONTI_FIFO_BUFFER_MODE (2)	

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

5.4 Multi-Dimensional Linear Interpolation Commands

5.4.1 pcielm4_lines_move

Executes a two dimensional relative position motion command. Positioning is performed on two axes with linear interpolation at the specified interpolation speed.

Syntax:

```
I32 pcielm4_lines_move (      U16 wBoardNo,
                              U16 wAxes,
                              U16 wAccDecMode,
                              U32 dwStartSpeed,
                              U32 dwDriveSpeed,
                              U32 dwEndSpeed,
                              U32 dwAcceleration,
                              U32 dwDeceleration,
                              const I32 lRelativeDistance [MAX_AXIS_NO],
                              U16 wInterpMode);
```

Parameters:

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td><i>wBoardNo:</i></td><td>PCIe board number of PCIELM4 (set via dip switch)</td></tr>
<tr><td><i>wAxes:</i></td><td>For multiple axis select the corresponding axis bit:
<table>
<tr><th>Axis</th><th>Value</th></tr>
<tr><td>Axis0</td><td>AXIS_0   (0x01)</td></tr>
<tr><td>Axis1</td><td>AXIS_1   (0x02)</td></tr>
</table>
<code>wAxes= AXIS_0| AXIS_2;</code></td></tr>
<tr><td><i>wAccDecMode:</i></td><td>
<table>
<tr><th>Velocity Profile</th><th>Value</th></tr>
<tr><td>T-Curve</td><td>ACC_DEC_T_CURVE (0x6565)</td></tr>
<tr><td>S-Curve</td><td>ACC_DEC_S_CURVE   (0x6666)</td></tr>
</table></td></tr>
<tr><td><i>dwStartSpeed:</i></td><td>Magnitude of start velocity vector (PPS)</td></tr>
<tr><td><i>dwDriveSpeed:</i></td><td>Magnitude of drive velocity vector (PPS)</td></tr>
<tr><td><i>dwEndSpeed:</i></td><td>Magnitude of end velocity vector (PPS)</td></tr>
<tr><td><i>dwAcceleration:</i></td><td>Magnitude of acceleration vector (PPS/Sec)</td></tr>
<tr><td><i>dwDeceleration:</i></td><td>Magnitude of deceleration vector (PPS/Sec)</td></tr>
<tr><td><i>lRelativeDistance[]:</i></td><td>Relative distance of selected axes (Pulse)
> 0: relative distance in positive direction
< 0: relative distance in negative direction</td></tr>
</table>

<table>
<tr>
<td></td>
<td>
<table>
<tr><th>Array</th><th>Relative distance (Pulse)</th></tr>
<tr><td>[0]</td><td>AXIS_0 relative moving distance</td></tr>
<tr><td>[1]</td><td>AXIS_1 relative moving distance</td></tr>
</table>
</td>
</tr>
<tr>
<td><i>wInterpMode:</i></td>
<td>
Command execution mode:
<table>
<tr><th>Mode</th><th>Value</th></tr>
<tr><td>General</td><td>GENERAL_INTERP_CONFIG (0)
• No command buffering takes place.
• A new command can only be executed if the previous command has finished.</td></tr>
<tr><td>Continuous interpolation</td><td>CONTI_INTERP_CONFIG (1)
• Command buffering; up to 5000 command can be stored
• Use this mode to generate a continuous motion path.</td></tr>
<tr><td>Reserved</td><td>CONTI_FIFO_BUFFER_MODE (2)</td></tr>
</table>
</td>
</tr>
</table>

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

Example:

```
I32 ret;
U16 BitMultiAxes = (0x01 | 0x02); //Axis 0,1
I32 FixedPulse[2] = {10000, 20000};

ret = pcielm4_lines_move(1, BitMultiAxes, ACC_DEC_T_CURVE, 0, 10000,
0, 20000, 20000, FixedPulse, 0);

if (ret != PCIELM4_SUCCESS)
{MessageBox("Lines Move Error!!");}
```

5.4.2 pcielm4_abs_lines_move

Executes a two-dimensional absolute position motion command. Positioning is performed on two axes with linear interpolation at the specified interpolation speed.

Syntax:

```
I32 pcielm4_abs_lines_move (    U16 wBoardNo,
                                U16 wAxes,
                                U16 wAccDecMode,
                                U32 dwStartSpeed,
                                U32 dwDriveSpeed,
                                U32 dwEndSpeed,
                                U32 dwAcceleration,
                                U32 dwDeceleration,
                                const I32 lAbsolutePosition [MAX_AXIS_NO],
                                U16 wInterpMode);
```

Parameters:

Name	Description
wBoardNo:	PCIe board number of PCIELM4 (set via dip switch)
wAxes:	For multiple axis select the corresponding axis bit: <table><tr><th>Axis</th><th>Value</th></tr><tr><td>Axis0</td><td>AXIS_0 (0x01)</td></tr><tr><td>Axis1</td><td>AXIS_1 (0x02)</td></tr></table> *wAxes*= AXIS_0| AXIS_2;
wAccDecMode:	<table><tr><th>Velocity Profile</th><th>Value</th></tr><tr><td>T-Curve</td><td>ACC_DEC_T_CURVE (0x6565)</td></tr><tr><td>S-Curve</td><td>ACC_DEC_S_CURVE (0x6666)</td></tr></table>
dwStartSpeed:	Magnitude of start velocity vector (PPS)
dwDriveSpeed:	Magnitude of drive velocity vector (PPS)
dwEndSpeed:	Magnitude of end velocity vector (PPS)
dwAcceleration:	Magnitude of acceleration vector (PPS/Sec)
dwDeceleration:	Magnitude of deceleration vector (PPS/Sec)
lAbsolutePosition []:	Absolute position of selected axes (Pulse) <table><tr><th>Array</th><th>Absolute Position (Pulse)</th></tr><tr><td>[0]</td><td>AXIS_0 absolute position</td></tr><tr><td>[1]</td><td>AXIS_1 absolute position</td></tr></table>
wInterpMode:	Command execution mode: <table><tr><th>Mode</th><th>Value</th></tr></table>

	General	GENERAL_INTERP_CONFIG (0) • No command buffering takes place. • A new command can only be executed if the previous command has finished.
	Continuous interpolation	CONTI_INTERP_CONFIG (1) • Command buffering; up to 5000 command can be stored • Use this mode to generate a continuous motion path.
	Reserved	CONTI_FIFO_BUFFER_MODE (2)

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

5.5 Two Dimensional Circular Interpolation Functions

5.5.1 pcielm4_t_arc2_move

Performs a two-dimensional circular interpolation with a T-curve velocity profile. The center and end position are specified relative to the current position.

Syntax:

```
I32 pcielm4_t_arc2_move (     U16 wBoardNo,
                              U8 bMainAxis,
                              U8 bSlaveAxis,
                              U32 dwStartSpeed,
                              U32 dwDriveSpeed,
                              U32 dwEndSpeed,
                              U32 dwAcceleration,
                              U32 dwDeceleration,
                              U8 bArcDirection,
                              I32 lMainAxisCenterPoint,
                              I32 lSlaveAxisCenterPoint,
                              I32 lMainAxisFinishPoint,
                              I32 lSlaveAxisFinishPoint,
                              U16 wInterpMode);
```

Parameters:

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td><em>wBoardNo:</em></td><td>PCIe board number of PCIELM4 (set via dip switch)</td></tr>
<tr><td><em>bMainAxis:</em></td><td rowspan="2">Axis definition:
<table>
<tr><th>Axis</th><th>Value</th></tr>
<tr><td>Axis0</td><td>AXIS_0 (0x01)</td></tr>
<tr><td>Axis1</td><td>AXIS_1 (0x02)</td></tr>
</table></td></tr>
<tr><td><em>bSlaveAxis:</em></td></tr>
<tr><td><em>dwStartSpeed:</em></td><td>Magnitude of start velocity vector (PPS)</td></tr>
<tr><td><em>dwDriveSpeed:</em></td><td>Magnitude of drive velocity vector (PPS)</td></tr>
<tr><td><em>dwEndSpeed:</em></td><td>Magnitude of end velocity vector (PPS)</td></tr>
<tr><td><em>dwAcceleration:</em></td><td>Magnitude of acceleration vector (PPS/Sec)</td></tr>
<tr><td><em>dwDeceleration:</em></td><td>Magnitude of deceleration vector (PPS/Sec)</td></tr>
<tr><td><em>bArcDirection:</em></td><td>Rotation direction
<table>
<tr><th>Direction</th><th>Value</th></tr>
<tr><td>CW</td><td>ARC_DIR_CW (0)</td></tr>
<tr><td>CCW</td><td>ARC_DIR_CCW (1)</td></tr>
</table></td></tr>
</table>

<table>
<tr><td><em>lMainAxisCenterPoint:</em></td><td>Relative center point of the main axis (Pulse)</td></tr>
<tr><td><em>lSlaveAxisCenterPoint:</em></td><td>Relative center point of the slave axis (Pulse)</td></tr>
<tr><td><em>lMainAxisFinishPoint:</em></td><td>Relative end point of the main axis (Pulse)</td></tr>
<tr><td><em>lSlaveAxisFinishPoint:</em></td><td>Relative end point of the slave axis (Pulse)</td></tr>
<tr><td><em>wInterpMode:</em></td><td>Command execution mode:
<table>
<tr><th>Mode</th><th>Value</th></tr>
<tr><td>General</td><td>GENERAL_INTERP_CONFIG (0)
• No command buffering takes place.
• A new command can only be executed if the previous command has finished.</td></tr>
<tr><td>Continuous interpolation</td><td>CONTI_INTERP_CONFIG (1)
• Command buffering; up to 5000 command can be stored
• Use this mode to generate a continuous motion path.</td></tr>
<tr><td>Reserved</td><td>CONTI_FIFO_BUFFER_MODE (2)</td></tr>
</table>
</td></tr>
</table>

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

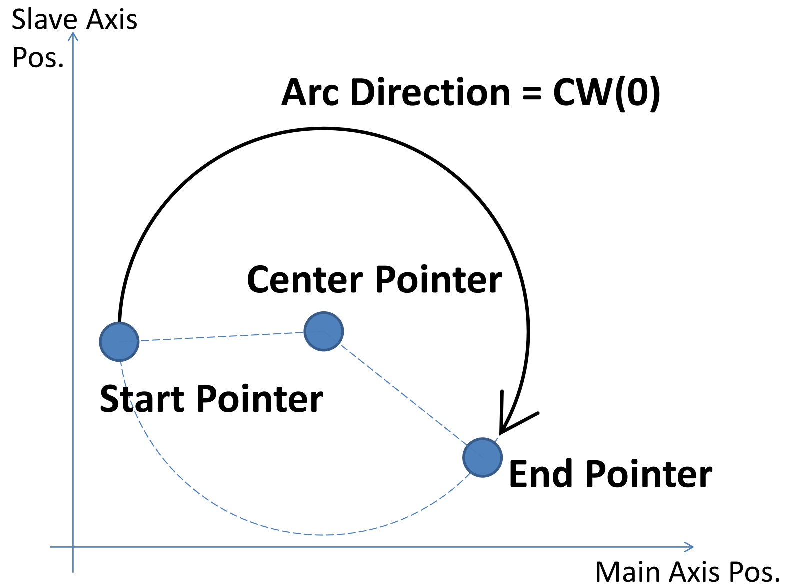


Figure 13: Clockwise circular interpolation

5.5.2 pcielm4_abs_t_arc2_move

Executes a two axes circular interpolation motion command with a T-curve velocity profile. The center and target position are specified in absolute position.

Syntax:

```
I32 pcielm4_abs_t_arc2_move (   U16 wBoardNo,
                                U8 bMainAxis,
                                U8 bSlaveAxis,
                                U32 dwStartSpeed,
                                U32 dwDriveSpeed,
                                U32 dwEndSpeed,
                                U32 dwAcceleration,
                                U32 dwDeceleration,
                                U8 bArcDirection,
                                I32 lMainAxisCenterPoint,
                                I32 lSlaveAxisCenterPoint,
                                I32 lMainAxisFinishPoint,
                                I32 lSlaveAxisFinishPoint,
                                U16 wInterpMode);
```

Parameters:

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td><i>wBoardNo:</i></td><td>PCIe board number of PCIELM4 (set via dip switch)</td></tr>
<tr><td><i>bMainAxis:</i></td><td rowspan="2">Axis definition:<table><tr><th>Axis</th><th>Value</th></tr><tr><td>Axis0</td><td>AXIS_0 (0x01)</td></tr><tr><td>Axis1</td><td>AXIS_1 (0x02)</td></tr></table></td></tr>
<tr><td><i>bSlaveAxis:</i></td></tr>
<tr><td><i>dwStartSpeed:</i></td><td>Magnitude of start velocity vector (PPS)</td></tr>
<tr><td><i>dwDriveSpeed:</i></td><td>Magnitude of drive velocity vector (PPS)</td></tr>
<tr><td><i>dwEndSpeed:</i></td><td>Magnitude of end velocity vector (PPS)</td></tr>
<tr><td><i>dwAcceleration:</i></td><td>Magnitude of acceleration vector (PPS/Sec)</td></tr>
<tr><td><i>dwDeceleration:</i></td><td>Magnitude of deceleration vector (PPS/Sec)</td></tr>
<tr><td><i>bArcDirection:</i></td><td>Rotation direction<table><tr><th>Direction</th><th>Value</th></tr><tr><td>CW</td><td>ARC_DIR_CW (0)</td></tr><tr><td>CCW</td><td>ARC_DIR_CCW (1)</td></tr></table></td></tr>
<tr><td><i>lMainAxisCenterPoint:</i></td><td>Absolute center point of the main axis (Pulse)</td></tr>
<tr><td><i>lSlaveAxisCenterPoint:</i></td><td>Absolute center point of the slave axis (Pulse)</td></tr>
<tr><td><i>lMainAxisFinishPoint:</i></td><td>Absolute end point of the main axis (Pulse)</td></tr>
</table>

<table>
<tr><td><em>lSlaveAxisFinishPoint:</em></td><td>Absolute end point of the slave axis (Pulse)</td></tr>
<tr><td><em>wInterpMode:</em></td><td>Command execution mode:
<table>
<tr><th>Mode</th><th>Value</th></tr>
<tr><td>General</td><td>GENERAL_INTERP_CONFIG (0)
• No command buffering takes place.
• A new command can only be executed if the previous command has finished.</td></tr>
<tr><td>Continuous interpolation</td><td>CONTI_INTERP_CONFIG (1)
• Command buffering; up to 5000 command can be stored
• Use this mode to generate a continuous motion path.</td></tr>
<tr><td>Reserved</td><td>CONTI_FIFO_BUFFER_MODE (2)</td></tr>
</table></td></tr>
</table>

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

5.6 Continuous Interpolation Functions

5.6.1 pcielm4_set_conti_interp_cfg

Assigns the two axes to an interpolation group and sets the group to continuous interpolation mode (see Figure 4, Figure 5 and Figure 6). Once the group has switch to continuous mode, all the arriving commands are being treated as continuous interpolation commands.

In continuous interpolation mode more than one command can be sent at a time. If a new command is being sent while the previous commands is still executing, then the arriving command will first be written to the internal FIFO buffer and starts to executed once the running command has finished. Up to 5000 commands can be stored in the FIFO buffer.

Syntax:

```
I32 pcielm4_set_conti_interp_cfg (     U16 wBoardNo,
                                       U8 bCfgEnable,
                                       U16 wGroupIndex0 );
```

Parameters:

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td><i>wBoardNo:</i></td><td>PCIe board number of PCIELM4 (set via dip switch)</td></tr>
<tr><td><i>bCfgEnable:</i></td><td>Enable continuous interpolation:
<table>
<tr><th>Mode</th><th>Value</th></tr>
<tr><td>Disable</td><td>CANCEL_INTERP_CONFIG (0)</td></tr>
<tr><td>Enable</td><td>START_INTERP_CONFIG (1)</td></tr>
</table></td></tr>
<tr><td><i>wGroupIndex0:</i></td><td>Select the axis which belongs to the first interpolation group
<table>
<tr><th>Axis</th><th>Value</th></tr>
<tr><td>Axis0</td><td>AXIS_0 (0x01)</td></tr>
<tr><td>Axis1</td><td>AXIS_1 (0x02)</td></tr>
</table>
Example:
<code>wGroupIndex0 = AXIS_0 | AXIS_1;</code></td></tr>
</table>

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

- If the user first want to fill the command FIFO buffer before starting to execute the motion commands then follow the following steps (see example 2):
    1. first call “`pcielm4_drv_hold`” to hold the next command from being executed

2. Fill the command buffer with commands
3. Call “`pcielm4_drv_start`” to start executing the command in the buffer

- The start and end speed of each command has to be lower or equal to the driving speed (Figure 14). Figure 15 shows velocity profiles which are currently not supported by continuous interpolation mode.

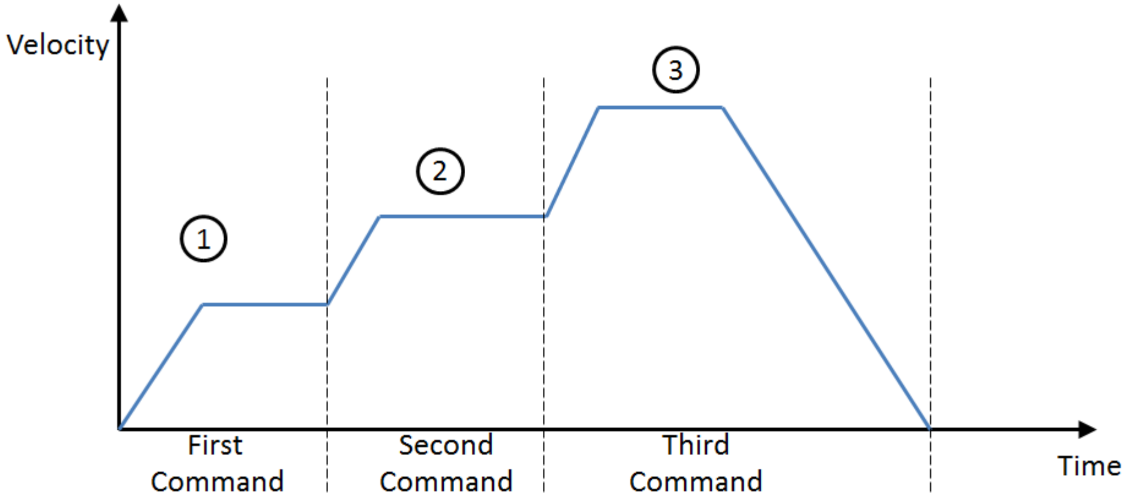


Figure 14: Velocity profile supported in continuous interpolation mode

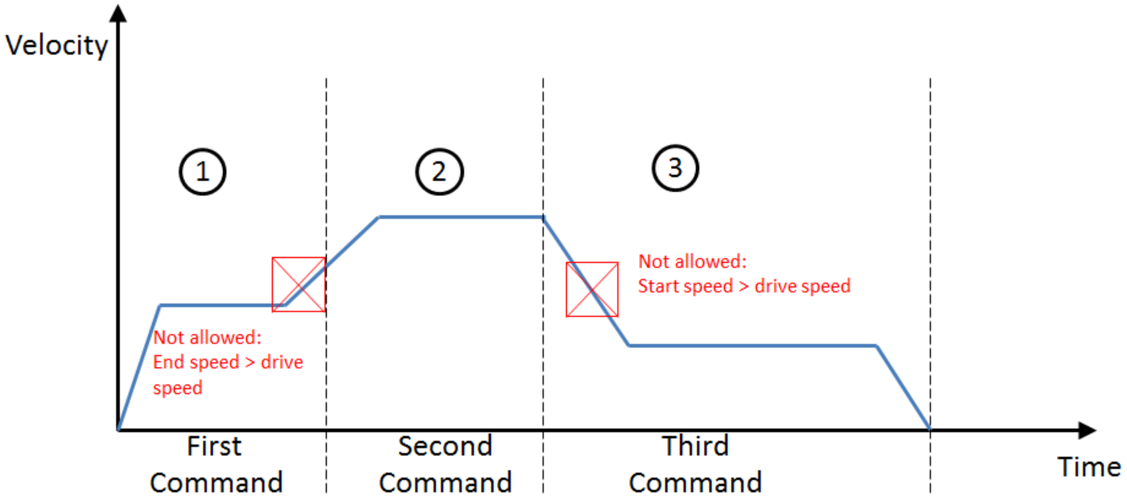


Figure 15: Velocity profiles which are not supported

Example:

Example 1:

```
//================================
// Two Dimensional Continue Interpolation
//================================
I32 eRet;
U32 dwStartSpeed, dwDriveSpeed, dwEndSpeed, dwAcceleration,
dwDeceleration;
```

```
U16 wGroupIndex0;
U8 bDone = 0;
U16 wStopStatus;
I32 x, y;
U16 bSlot = 1;

dwStartSpeed   = 1000;
dwDriveSpeed   = 20000;
dwEndSpeed     = 1000;
dwAcceleration = 5000;
dwDeceleration = 5000;
x = 100;
y = 100;

//Assign two axis to a continuous interpolation group:
wGroupIndex0 = AXIS_0|AXIS_1;
eRet = pcielm4_set_conti_interp_cfg( bSlot, START_INTERP_CONFIG,
wGroupIndex0);

//Write 120 interpolation commands to the buffer for execution
// Once a command arrives at an empty buffer it will be executed
for(int i=0; i<120; i++) //120
{
    x *= i;
    y *= x;
    eRet = pcielm4_abs_t_line2_move( bSlot, AXIS_0, AXIS_1,
            dwStartSpeed, dwDriveSpeed, dwEndSpeed, dwAcceleration,
            dwDeceleration, x, y, CONTI_INTERP_CONFIG);
}

//Wait until the continuous interpolation command have been executed:
while( bDone != 1)
{
    eRet = pcielm4_get_motion_done( bSlot, AXIS_0, &bDone,
&wStopStatus);
    ::Sleep(10);
}

//Disable the continuous interpolation mode:
eRet = pcielm4_set_conti_interp_cfg( bSlot, CANCEL_INTERP_CONFIG,
wGroupIndex0 );
```

5.7 Motion Stop Functions

5.7.1 pcielm4_stop_move

Stops the current executing motion command for the specified axis. Stops motion before reaching the destination.

Syntax:

```
I32 pcielm4_stop_move (    U16 wBoardNo,
                           U8 bSingleAxis,
                           U8 bStopMode);
```

Parameters:

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td><i>wBoardNo:</i></td><td>PCIe board number of PCIELM4 (set via dip switch)</td></tr>
<tr><td><i>bSingleAxis:</i></td><td>Axis definition:
<table>
<tr><th>Axis</th><th>Value</th></tr>
<tr><td>Axis0</td><td>AXIS_0 (0x01)</td></tr>
<tr><td>Axis1</td><td>AXIS_1 (0x02)</td></tr>
</table></td></tr>
<tr><td><i>bStopMode:</i></td><td>Stop mode
<table>
<tr><th>Mode</th><th>Value</th></tr>
<tr><td>Deceleration stop</td><td>STOP_SLOWDOWN (1)</td></tr>
<tr><td>Sudden stop</td><td>STOP_SUDDEN (2)</td></tr>
</table></td></tr>
</table>

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

5.7.2 pcielm4_set_softlimit

Sets the software limits for the positive and negative direction. Once a software limit position is specified, the PCIELM4 will not accept position commands beyond the limit and motion will stop once the limit is hit.

Syntax:

```
I32 pcielm4_set_softlimit (   U16 wBoardNo,
                              U8 bSingleAxis,
                              U8 bStopMode,
                              U8 bRefSource,
                              I32 lLimitPositive,
                              I32 lLimitNegative);
```

Parameters:

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td><em>wBoardNo:</em></td><td>PCIe board number of PCIELM4 (set via dip switch)</td></tr>
<tr><td><em>bSingleAxis:</em></td><td>Axis definition:
<table>
<tr><th>Axis</th><th>Value</th></tr>
<tr><td>Axis0</td><td>AXIS_0 (0x01)</td></tr>
<tr><td>Axis1</td><td>AXIS_1 (0x02)</td></tr>
</table></td></tr>
<tr><td><em>bStopMode:</em></td><td>Stop mode
<table>
<tr><th>Stop Mode</th><th>Value</th></tr>
<tr><td>Deceleration stop</td><td>STOP_SLOWDOWN (1)</td></tr>
<tr><td>Sudden stop</td><td>STOP_SUDDEN (2)</td></tr>
</table></td></tr>
<tr><td><em>bRefSource:</em></td><td>Position counter source
<table>
<tr><th>Source</th><th>Value</th></tr>
<tr><td>Encoder Pulse Counter</td><td>FEEDBACK_SRC_ENC (1)</td></tr>
<tr><td>Commanded Position Pulse Counter</td><td>FEEDBACK_SRC_DDA (2)</td></tr>
</table></td></tr>
<tr><td><em>lLimitPositive:</em></td><td>Positive direction soft limit</td></tr>
<tr><td><em>lLimitNegative:</em></td><td>Negative direction soft limit</td></tr>
</table>

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

5.7.3 pcielm4_set_softlimit_disable

Disables the axis limits settings.

Syntax:

```
I32 pcielm4_set_softlimit_disable (     U16 wBoardNo,
                                        U8 bSingleAxis);
```

Parameters:

Name	Description
wBoardNo:	PCIe board number of PCIELM4 (set via dip switch)
bSingleAxis:	Axis definition: Axis | Value Axis0 | AXIS_0 (0x01) Axis1 | AXIS_1 (0x02)

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

5.8 Multi-Axis Hold/Release Functions

5.8.1 pcielm4_drv_hold

This command sets the specified axes in holding mode after the current running command has reached its target position. Therefore this instruction takes effect for the next command. The execution of the next command will be put on hold until the "pcielm4_drv_start()" releases the hold operation.

Syntax:

```
I32 pcielm4_drv_hold (    U16 wBoardNo,
                          U16 wBitMultiAxes);
```

Parameters:

Name	Description
wBoardNo:	PCIe board number of PCIELM4 (set via dip switch)
wBitMultiAxes:	Each bit in the variable represents an axis: Axis | Value Axis0 | AXIS_0 (0x01) Axis1 | AXIS_1 (0x02)

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

- After calling "pcielm4_drv_hold()" the current running interpolation command will finish first, but the next command will not start until "pcielm4_drv_start()" has been called.
- The command will hold the interpolation group if only one or more of the hold axes (*wBitMultiAxes*) belongs to the group.

5.8.2 pcielm4_drv_start

Terminates the hold operation. Axes which have been put on hold by "pcielm4_drv_hold()" will continue to execute the next motion command stored in the command FIFO buffer.

Syntax:

```
I32 pcielm4_drv_start (     U16 wBoardNo,
                            U16 wBitMultiAxes);
```

Parameters:

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td><i>wBoardNo:</i></td><td>PCIe board number of PCIELM4 (set via dip switch)</td></tr>
<tr><td><i>wBitMultiAxes:</i></td><td>Each bit of the variable represents an axis:
<table>
<tr><th>Axis</th><th>Value</th></tr>
<tr><td>Axis0</td><td>AXIS_0 (0x01)</td></tr>
<tr><td>Axis1</td><td>AXIS_1 (0x02)</td></tr>
</table>
</td></tr>
</table>

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

6 Other Functions

6.1 Compare Function

6.1.1 pcielm4_set_compare_trig_cfg

Configures and enables the compare trigger function. The compare function outputs a signal when the compare condition has been met. Two compare modes are being supported:

1. One time compare mode (Single compare mode)
2. Auto increment compare mode.

Syntax:

```
I32 pcielm4_set_compare_trig_cfg (      U16 wBoardNo,
                                        U8 bSingleAxis,
                                        U8 bCmpTrigEnable,
                                        U8 bOutputLogic,
                                        U16  bTrigPulseWidth,
                                        U8 bMoveDirection,
                                        U8 bCmpIncEnable,
                                        U16 wConstPitch,
                                        I32 lCmpData);
```

Parameters:

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td><em>wBoardNo:</em></td><td>PCIe board number of PCIELM4 (set via dip switch)</td></tr>
<tr><td><em>bSingleAxis:</em></td><td>Axis definition:
<table>
<tr><th>Axis</th><th>Value</th></tr>
<tr><td>Axis0</td><td>AXIS_0   (0x01)</td></tr>
<tr><td>Axis1</td><td>AXIS_1   (0x02)</td></tr>
</table></td></tr>
<tr><td><em>bCmpTrigEnable:</em></td><td>Enable compare function
<table>
<tr><th>State</th><th>Value</th></tr>
<tr><td>Disable</td><td>DISABLE   (0x00)</td></tr>
<tr><td>Enable</td><td>ENABLE   (0x01)</td></tr>
</table></td></tr>
<tr><td><em>bOutputLogic:</em></td><td>Compare (CMP) active level
<table>
<tr><th>Trigger level</th><th>Value</th></tr>
<tr><td>Active low</td><td>LOGIC_ACTIVE_LOW   (0x00)</td></tr>
<tr><td>Active high</td><td>LOGIC_ACTIVE_HIGH   (0x01)</td></tr>
</table></td></tr>
<tr><td><em>bTrigPulseWidth:</em></td><td>Pulse width trigger signals (see remarks)</td></tr>
</table>

<table>
<tr><td><i>bMoveDirection:</i></td><td>Axis moving direction
<table>
<tr><th>Direction</th><th>Value</th></tr>
<tr><td>Negative</td><td>CMPTRIG_REVERSE_MOVE (0)</td></tr>
<tr><td>Positive</td><td>CMPTRIG_FORWARD_MOVE (1)</td></tr>
</table>
The compare function will only trigger if the axis moves in the specified direction and the compare condition is being met.</td></tr>
<tr><td><i>bCmpIncEnable:</i></td><td>Select the compare mode:
<table>
<tr><th>Compare Mode</th><th>Value</th></tr>
<tr><td>One time compare mode: Triggers only one output signal</td><td>DISABLE (0x00)</td></tr>
<tr><td>Auto-increment compare position:
Set the compare trigger to continuously trigger a output signal at equidistant position</td><td>ENABLE (0x01)</td></tr>
</table></td></tr>
<tr><td><i>wConstPitch:</i></td><td>The auto-increment distance; The distance between two compare signal (pulse)
(Only valid if “<i>bCmpIncEnable</i>” is enabled otherwise this parameter will be ignored)</td></tr>
<tr><td><i>lCmpData:</i></td><td>The first position at which the compare function will trigger an output signal.
(If “<i>bCmpIncEnable</i>” is disabled then one output signal will be triggered at the “lCmpData” position)</td></tr>
</table>

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

- Compare function cannot be used when axis is in Vring counter mode (see “pcielm4_set_vring_counter()”
- Pulse width table:

Pulse Width	Minimum trigger period *	Value
160ns~320ns	640ns	TRIG_PULSE_WIDTH_160nsTo320ns (0x01)
320ns~640ns	1.28us	TRIG_PULSE_WIDTH_320nsTo640ns (0x02)
640ns~1.28us	2.56us	TRIG_PULSE_WIDTH_640nsTo1p28us (0x03)
1.28us~2.56us	5.12us	TRIG_PULSE_WIDTH_1p28usTo2p56us (0x04)
2.56us~5.12us	10.24us	TRIG_PULSE_WIDTH_2p56usTo5p12us (0x05)
5.12us~10.24us	20.48us	TRIG_PULSE_WIDTH_5p12usTo10p24us (0x06)
10.24us~20.48us	40.96us	TRIG_PULSE_WIDTH_10p24usTo20p48us (0x07)
20.48us~40.96us	81.92us	TRIG_PULSE_WIDTH_20p48usTo40p96us (0x08)
40.96us~81.92us	163.84us	TRIG_PULSE_WIDTH_40p96usTo81p92us

		(0x09)
81.92us~163.84us	327.68us	TRIG_PULSE_WIDTH_81p92usTo163p84us (0x0A)
163.84us~327.68us	655.36us	TRIG_PULSE_WIDTH_163p84usTo327p68us (0x0B)
327.68us~655.36us	1.31072ms	TRIG_PULSE_WIDTH_327p68usTo655p36us (0x0C)
655.36us~1.31072ms	2.62144ms	TRIG_PULSE_WIDTH_655p36usTo1p31072ms (0x0D)
1.31072ms~2.62144ms	5.24288ms	TRIG_PULSE_WIDTH_1p31072msTo2p62144ms (0x0E)
2.62144ms~5.24288ms	10.48576ms	TRIG_PULSE_WIDTH_2p62144msTo5p24288ms (0x0F)

* If the trigger signal output period is less than the minimum trigger cycles then sporadically no output signal will be generated.

Table 2: Pulse width setting of the trigger signal

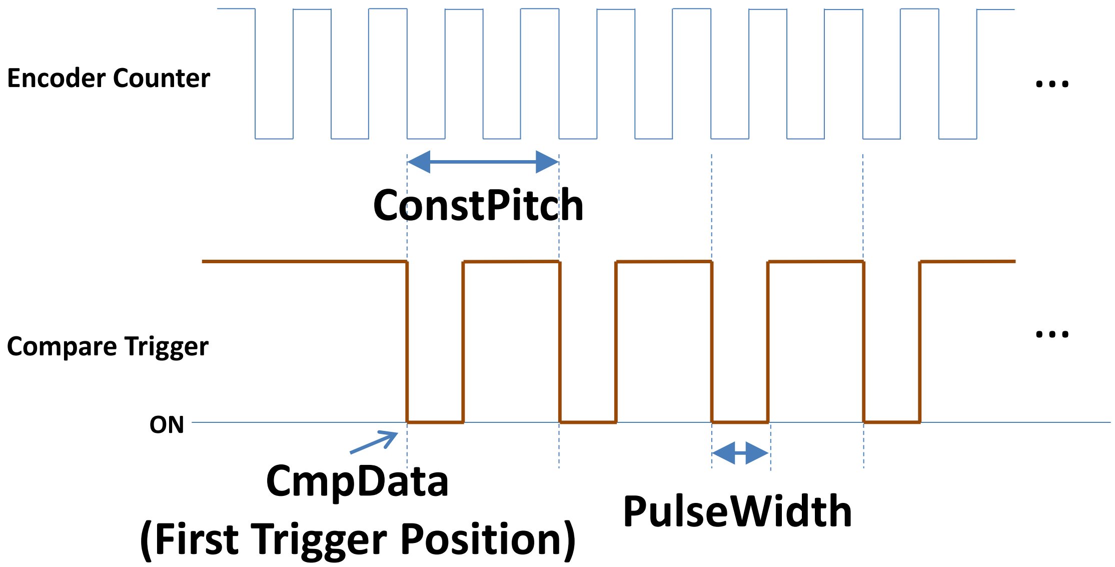


Figure 16: Continuous equidistant spaced trigger output function

6.2 Latch Function

6.2.1 pcielm4_set_latch_cfg

Configures and enables position Latch. The latch function captures the encoder counter value at an instant when the latch signal activates. The LTC channel is used to receive the latch pulse. The latch function is hardware implemented and executes at very high speed.

Syntax:

```
I32 pcielm4_set_latch_cfg (    U16 wBoardNo,
                               U8 bSingleAxis,
                               U8 bLatchEnable,
                               U8 bLatchLogic);
```

Parameters:

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td><i>wBoardNo:</i></td><td>PCIe board number of PCIELM4 (set via dip switch)</td></tr>
<tr><td><i>bSingleAxis:</i></td><td>Axis definition:
<table>
<tr><th>Axis</th><th>Value</th></tr>
<tr><td>Axis0</td><td>AXIS_0 (0x01)</td></tr>
<tr><td></td><td></td></tr>
</table></td></tr>
<tr><td><i>bLatchEnable:</i></td><td>Enable/Disable the latch function
<table>
<tr><th>State</th><th>Value</th></tr>
<tr><td>treat LTC PIN as a general input</td><td>DISABLE (0x00)</td></tr>
<tr><td>treat LTC PIN as a dedicated external trigger to latch input</td><td>ENABLE (0x01)</td></tr>
</table></td></tr>
<tr><td><i>bLatchLogic:</i></td><td>Latch (LTC) active level
<table>
<tr><th>Trigger level</th><th>Value</th></tr>
<tr><td>Active low</td><td>LOGIC_ACTIVE_LOW (0x00)</td></tr>
<tr><td>Active high</td><td>LOGIC_ACTIVE_HIGH (0x01)</td></tr>
</table></td></tr>
</table>

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

- Only when Latch DI activates will the current encoder position be latched.
- Latch function is only supported by AXIS_0

6.2.2 pcielm4_get_latch

Reads the present latched position of the specified axis. Returns the captured position triggered by the latch LTC signal.

Syntax:

```
I32 pcielm4_get_latch (    U16 wBoardNo,
                           U8 bSingleAxis,
                           I32* plLatchData);
```

Parameters:

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td><i>wBoardNo:</i></td><td>PCIe board number of PCIELM4 (set via dip switch)</td></tr>
<tr><td><i>bSingleAxis:</i></td><td>Axis definition:
<table>
<tr><td>Axis</td><td>Value</td></tr>
<tr><td>Axis0</td><td>AXIS_0   (0x01)</td></tr>
<tr><td></td><td></td></tr>
</table></td></tr>
<tr><td><i>plLatchData:</i></td><td>Pointer to the value of the encoder position counter latch</td></tr>
</table>

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

- Only when Latch DI activates will the current encoder position be latched.
- Latch function is only supported by AXIS_0

7 Hardware and Software Version Functions

7.1 Hardware Version

7.1.1 pcielm4_get_card_version

Gets the PCB and PLD version.

Syntax:

```
I32 pcielm4_get_card_version (    U16 wBoardNo,
                                  U16* pwPCBVersion,
                                  U16* pwPLDVersion);
```

Parameters:

Name	Description
wBoardNo:	PCIe board number of PCIELM4 (set via dip switch)
pwPCBVersion:	Pointer to PCB version
pwPLDVersion:	Pointer to PLD version

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

7.1.2 pcielm4_get_fpga_version

Gets the FPGA version.

Syntax:

```
I32 pcielm4_get_fpga_version (    U16 wBoardNo,
                                  U16* pwFpgaVers);
```

Parameters:

Name	Description
wBoardNo:	PCIe board number of PCIELM4 (set via dip switch)
pwFpgaVers:	Pointer to FPGA version

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

7.2 Software Version

7.2.1 pcielm4_get_dsp_firmware_version

Gets the current DSP firmware version.

Syntax:

```
I32 pcielm4_get_dsp_firmware_version (      U16 wBoardNo,
                                            U16* pwDspFirmwareVers);
```

Parameters:

Name	Description
wBoardNo:	PCIe board number of PCIELM4 (set via dip switch)
pwDspFirmwareVers:	Pointer to DSP firmware version

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

7.2.2 pcielm4_get_dll_version

Gets the DLL version.

Syntax:

```
I32 pcielm4_get_dll_version (    U16* pwDllVers  );
```

Parameters:

Name	Description
pwDllVers:	Pointer to DLL version

Return:

- 0: PCIELM4_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

8 Appendix

8.1 Variable data type definition

Type	Bytes	VC++ / BCB	C#	VB.NET	VB6	Delphi
F64	8	double	double	Double	Double	Double
F32	4	float	float	Single	Single	Single
U32	4	unsigned long	uint	UInteger	Long	LongWord
I32	4	long	int	Integer	Long	LongInt
U16	2	unsigned short	ushort	UShort	Integer	Word
I16	2	short	short	Short	Integer	Smallint
U8	1	unsigned char	byte	Byte	Byte	Byte
		char*	string	String	String	WideString/ WideChar
		wchar_t*	string	String	String	AnsiString/ AnsiChar
		HANDLE	IntPtr	IntPtr	Long	Thandle

Table 3: Variable data type definition

8.2 Function Error Code

Error Code	Name	Description
0	PCIELM4_SUCCESS	Function executed successfully.
1	PCIELM4_ERROR	Reserved
2	ERR_CARD_NOT_FOUND	No PCIELM4 board found with the specified board number.
3	ERR_INVALID_SLOT_NO	Invalid board number
4	ERR_INVALID_PARA	An invalid parameter value has been passed to a function.
5	ERR_INVALID_DRIVING_SPEED	An invalid value has been passed to the driving speed parameter
6	ERR_INVALID_START_SPEED	An invalid value has been passed to the start speed parameter
7	ERR_INVALID_END_SPEED	An invalid value has been passed to the end speed parameter
8	ERR_INVALID_MPG_SPEED	Reserved
9	ERR_INVALID_MOVE_DIRECTION	An invalid value has been passed to the “move direction” parameter
10	ERR_AXES_NOT_MATCH	Internal error
11	ERR_GROUP_INVALID_DIMENSION	Internal error
12	ERR_INVALID_GROUP_AXES	Internal error
13	ERR_GROUP_NOT_CONFIGURED	Internal error
14	ERR_GROUP_OUT_OF_RANGE	Internal error
15	ERR_INCORRECT_GROUP_ASSIGNED	Internal error
16	ERR_GROUP_SAME_AXIS	Internal error
17	ERR_GROUP_RELEASE_REQUIRED	Internal error
18	ERR_MOTION_NOT_FINISHED	• Some parameters cannot be set if a motion command is still executing • If the axis is not in a continuous interpolation mode then the next motion command cannot be sent while the previous motion command is still executing.
19	ERR_CARD_NOT_REGISTERED	The motion functions have not been registered and initialized yet. It is required to first call "pcielm4_registration()" after open the driver before calling any other functions.
20	ERR_INVALID_AXIS	An invalid value has been passed to the axis parameter
21	ERR_AXIS_ERROR	Internal error
22	ERR_CMD_NO_CLOSED_LOOP_SUPPORT	Reserved
23	ERR_INTERPOL_NOT_CONFIGURED	A continuous interpolation command has been sent while the axes are not in continuous interpolation mode. First call “pcielm4_set_conti_interp_cfg()" to set the axes in continuous interpolation mode.

24	ERR_INVALID_HELICAL_MODE	reserved
25	ERR_INVALID_FRNET_SPEED_SETTING	Reserved
26	ERR_INVALID_FRNET_SA_GROUP_ADDR	Reserved
27	ERR_INVALID_FRNET_RA_GROUP_ADDR	Reserved
28	ERR_INVALID_MPG_GAIN	Reserved
29	ERR_MPG_NOT_CONFIGURED	Reserved
30	ERR_MOTION_IS_COMPLETED	Reserved
31	ERR_CMD_NOT_FOR_CONTI_INTERPOL	The called function cannot be called when the axes are in continuous interpolation mode. First call "pcielm4_set_conti_interp_cfg()" to disable the continuous interpolation mode.
32	ERR_MODE_NOT_FOR_CONTI_INTERPOL	Reserved
33	ERR_DPRAM_RTC_BLOCKS_AVAILABLE	Internal error
34	ERR_DPRAM_RTC_BUFFER_FULL	Internal error
35	EMG_ACVTIVATED	Emergency stop has been activated. No new motion commands can be executed while emergency stop is active.
36	ERR_CONFLICT_CONTI_INTERP_CONFIG	A "non-continuous" interpolation command has been sent while the axes are "continuous interpolation" mode. First call "pcielm4_set_conti_interp_cfg()" to disable the continuous interpolation mode.
37	ERR_CONTI_INTERP_INVALID_CONFIG	Reserved
38	ERR_CONTI_INTERP_INTERRUPTED	Reserved
39	ERR_CONTI_INTERP_INVALID_START	Reserved
40	ERR_CONTI_INTERP_FIFO_EMPTY	Reserved
41	ERR_NOT_IN_CONTI_INTERP_MODE	Reserved
42	ERR_CONTI_INTERP_START	Reserved
43	ERR_INVALID_RING_COUNTER	The value passed to the ring counter parameter is not being supported.
44	ERR_ISR_IS_USED_FOR_CMP_TRIG	Compare trigger function is in use, therefore the ring counter mode can not be used. Both compare and ring counter mode can not be enabled together.
45	ERR_RTC_WAIT_ABORTED	Internal error
46	ERR_RTC_TIMEOUT	Internal error
47	ERR_DSP_RESET	Resetting the DSP failed
48	INFO_SET_DDA_NOT_REQUIRED	The DDA has already been set
49	ERR_EXCEED_MAX_POSTION	The value passed to the set position counter parameter exceeds the valid range. Valid range:-2147483646~ 2147483646
50	ERR_NOT_SUPPORTED_BY_VRING_MODE	The axes are in ring counter mode. Some functions are not supported in ring counter mode (e.g. “pcielm4_set_compare_trig_cfg()”)
51	ERR_DLL_DO_NOT_SUPPORT_MULTI_APP	More than one application access the PCIELM4. Only one application is allowed to access the PCIELM4 at a time.
-1001	ERR_DSP_MOTIONPATH_FULL	DSP response error code
-1002	ERR_DSP_REMAIN_STILL	
-1003	ERR_DSP_DIMENSION_OUT_RANGE	

-1004	ERR_DSP_SPEED_VALUE	
-1005	ERR_DSP_SMALLSHAPE_ERROR	
-1006	ERR_DSP_SAWSHAPE_ERROR	
-1007	ERR_DSP_AXIS_INUSE	
-1008	ERR_DSP_NORMALVECTOR	
-1009	ERR_DSP_CIRCLE_RADIUS	
-1011	ERR_DSP_MECH_PARA_VALUE	
-1100	ERR_DSP_MOTIONPATH_ALREADY_FREE	
-1201	ERR_DSP_EXCEED_SPEED_LIMIT	
-1202	ERR_DSP_EXCEED_VRING_LIMIT	
-1203	ERR_DSP_EXCEED_MAX_POSTION	
-2001	ERR_DSP_NO_COMMAND	
-2002	ERR_DSP_CRC_ERROR	
-2003	ERR_DSP_UNKNOWN_COMMAND	
-2004	ERR_DSP_MULTIBLOCK_CMD	
-2005	ERR_DSP_ACC_TYPE	
-2006	ERR_DSP_DEC_TYPE	
-2007	ERR_DSP_CMD_NOT_ALLOWED	
-2011	ERR_DSP_INHIBIT_BY_EMG	
-2012	ERR_DSP_INHIBIT_BY_MPG_EMG	
-2013	ERR_DSP_INHIBIT_BY_PEL	
-2014	ERR_DSP_INHIBIT_BY_MEL	
-2015	ERR_DSP_INHIBIT_BY_ALM	
-2016	ERR_DSP_INHIBIT_BY_RDY	
-2101	ERR_DSP_GROUP_OUT_RANGE	
-2102	ERR_DSP_GROUP_ALREADY_FREE	
-2103	ERR_DSP_GROUP_ASSIGNED	
-2104	ERR_DSP_GROUP_INUSE	
-2105	ERR_DSP_GROUP_NOT_INUSE	
-2106	ERR_DSP_AXIS_ASSIGNED	
-2201	ERR_DSP_CONTROL_MODE	
-2301	ERR_DSP_BUFFER_FULL	
-2302	ERR_DSP_BUFFER_INFO	
-2303	ERR_DSP_BUFFER_TYPE	
-2304	ERR_DSP_BUFFER_SIZE	
-2305	ERR_DSP_BUFFER_INUSE	
-2306	ERR_DSP_BUFFER_NOT_READY	
-2401	ERR_DSP_STOP_BY_P_CHANGE	
-2402	ERR_DSP_CLEAR_STOP	
-2410	ERR_DSP_STOP_BY_GINP	
-2430	ERR_DSP_STOP_BY_AXIS_IO	
-2501	ERR_DSP_NO_AVAILABLE_MACRO	
-2502	ERR_DSP_MACRO_INUSE	
-2503	ERR_DSP_MACRO_EMPTY	
-2601	ERR_DSP_HOMING_IN_PROGRESS	
-3601	ERR_DSP_OUTPUT_SATURATION	
-3611	ERR_DSP_ERR_MSG_BUF_EMPTY	
-3612	ERR_DSP_ERR_MSG_BUF_OVERFLOW	
-1	ERR_DSP_TIMEOUT_ERROR	
-2	ERR_DSP_HW_ID_ERROR	

-3	ERR_DSP_AXIS_OUT_RANGE	
-4	ERR_DSP_ADDR_OUT_RANGE	
-5	ERR_DSP_VALUE_OUT_RANGE	
-6	ERR_DSP_FPGA_DL_FAILED	
-101	ERR_DSP_DA_AUTO_UPDATE	
-102	ERR_DSP_DA_BUSY	
-201	ERR_DSP_CMP_INUSE	
-32767	ERR_DSP_SYSTEM_ERROR	
-32768	ERR_DSP_NOT_IMPLEMENT	

Table 4: Function error code