I-8196F
Motion Control Module
User Manual

(Version 1.0)

API Library

ICP DAS CO., LTD.

Warranty

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

Warning

ICPDAS Inc. assumes no liability for damages consequent to the use of this product. ICPDAS Inc. reserves the right to change this manual at any time without notice. The information furnished by ICPDAS Inc. is believed to be accurate and reliable. However, no responsibility is assumed by ICPDAS Inc. for its use, or for any infringements of patents or other rights of third parties resulting from its use.

Copyright

Trademark

The names used for identification only maybe registered trademarks of their respective companies.

License

The user can use, modify and backup this software on a single machine. The user may not reproduce, transfer or distribute this software, or any copy, in whole or in part.

Contents

1 Preface

1.1 Function Overview

The I-8196F motion card has been designed for the following PAC (Programmable Automation Controller) series of ICPDAS: XPAC, WinPAC, ViewPAC. The PAC comes with the Windows CE5.0, CE6.0 and Embedded Standard 2009 operation system (Consult ICPDAS catalogue to select the required PAC version).
ICPDAS offers for each OS version a corresponding I-8196F DLL version. The PAC user manual describes in detail the steps to be taken to write a program for the PAC. Before starting to write a program it is necessary to install the SDK on your computer (consult PAC user manual).
The EzGo utility allows the user to configure the I-8196F and execute some basic motion commands.

The following table gives a short overview of the DLL functions supported by I-8196F:

Function	Description
Initialization	
i8196f_registration	Confirms that the I-8196F card is plugged into the specified slot of the PAC and initializes the I-8196F.
i8196f_check_card_in_slot	Checks whether the I-8196F card is plugged into the specified slot of the PAC.
i8196f_reset	Reset and reinitialize the I-8196F module.
Axis Digital I/O Function	
i8196f_set_servo_on	Set the SRV_ON channel output signal to enable/disable the servo drive for pulse command input.
i8196f_set_erc	The ERC signal sets the servo drive deflection counter to zero.
i8196f_set_alarm_reset	Reset the servo drive alarm signal (ALM_RST).
i8196f_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 I-8196F. If the servo alarm function of the I-8196F has been enabled and the servo alarm signal is active then no pulses will be outputted.
i8196f_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 I-8196F 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.
i8196f_set_ready	Enables/Disables the I-8196F to check for the servo drive “ready” state and sets the active level of the servo RDY signal.
i8196f_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.
i8196f_get_mdi_status	Reads the current input state of all axis DI channels.
i8196f_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 “i8196f_set_servo_on()”.
i8196f_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 “i8196f_set_erc()”.
i8196f_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. “i8196f_set_alarm_reset()” sets the output signal.
Motion Control Pulse Setting Function	
i8196f_set_pls_cfg	Set the pulse output mode for each axis.
i8196f_set_enc_cfg	Set the parameters of the encoder pulse input.
i8196f_set_cmdcounter	Set the command counter (position command) value.
i8196f_get_cmdcounter	Get the current command counter (position command) value.
i8196f_set_enccounter	Set the encoder counter value.
i8196f_get_enccounter	Get the current encoder counter value.
i8196f_set_vring_counter	Set the maximum ring counter position for both the encoder and commanded position counter.
i8196f_get_vring_counter	Read the maximum ring counter setting.
i8196f_disable_vring_counter	Disable the ring counter setting.
Automatic Home Search Configuration	
i8196f_set_home_cfg	Set automatic home search parameters.
Automatic Home Execution	
i8196f_home_start	Start searching for the home position.
Read Motion Status	
i8196f_get_motion_done	Read the current motion status of the axis.
i8196f_get_speed	Get the current axis speed.
i8196f_get_acc	Get the axis current axis acceleration.
Single Axis Motion Commands	
i8196f_t_move	Execute a single axis, relative position motion command with a trapezoidal velocity profile (T-curve). The i8196f_t_move instruction moves the axis the specified travel distance from the current position.
i8196f_abs_t_move	Execute a single axis, absolute position motion command with a trapezoidal velocity profile (T-curve). The i8196f_abs_t_move instruction moves the axis to a specified absolute target position. You can execute this instruction even if home is not defined.
i8196f_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.
i8196f_abs_s_move	Execute a single axis, absolute position motion command with

	an S-curve velocity profile.
i8196f_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	
i8196f_t_line2_move	Executes a two axes relative distance linear interpolation motion command with a T-curve velocity profile. The i8196f_t_line2_move instruction performs linear interpolation for two axes. The target position is specified as a relative position.
i8196f_abs_t_line2_move	Executes a two axes absolute position interpolation motion command with a T-Curve velocity profile.
i8196f_s_line2_move	Executes a two axes relative distance interpolation motion command with an S-Curve velocity profile.
i8196f_abs_s_line2_move	Executes a two axis absolute position interpolation motion command with an S-Curve velocity profile.
Three Axis Linear Interpolation Commands	
i8196f_t_line3_move	Executes a three axes linear interpolation motion command with a T-curve velocity profile. The target position is specified as a relative distance to the current position.
i8196f_abs_t_line3_move	Executes a three axes linear interpolation motion command with a T-curve velocity profile. This instruction moves the axes to an absolute position relative to (0) zero.
i8196f_s_line3_move	Executes a three axes linear interpolation motion command with an S-curve velocity profile. The i8196f_s_line3_move instruction moves the axes the specified travel distance from the current position.
i8196f_abs_s_line3_move	Executes a three axes absolute position interpolation motion command with an S-curve velocity profile.
Multi-Dimensional Linear Interpolation Commands	
i8196f_lines_move	Executes a multidimensional relative position motion command. Positioning is performed on up to six axes with linear interpolation at the specified interpolation speed. The number of interpolation axes can be selected.
i8196f_abs_lines_move	Executes a multidimensional absolute position motion command. Positioning is performed on up to six axes with linear interpolation at the specified interpolation speed. The number of interpolation axes can be selected.
Two Dimensional Circular Interpolation Functions	
i8196f_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.
i8196f_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.
Three Dimensional Helical Interpolation Functions	
i8196f_t_helical_move	Executes a three dimensional helical (screw like) movement. The movement is a composition of a rotation by an angle about a helical axis with a translation distance along this axis. The circle center position is the relative distance from the current position.
i8196f_abs_t_helical_move	Executes a three dimensional helical (screw like) movement. The movement is a composition of a rotation by an angle about a helical axis with a translation distance along this axis. The circle center position is an absolute position.

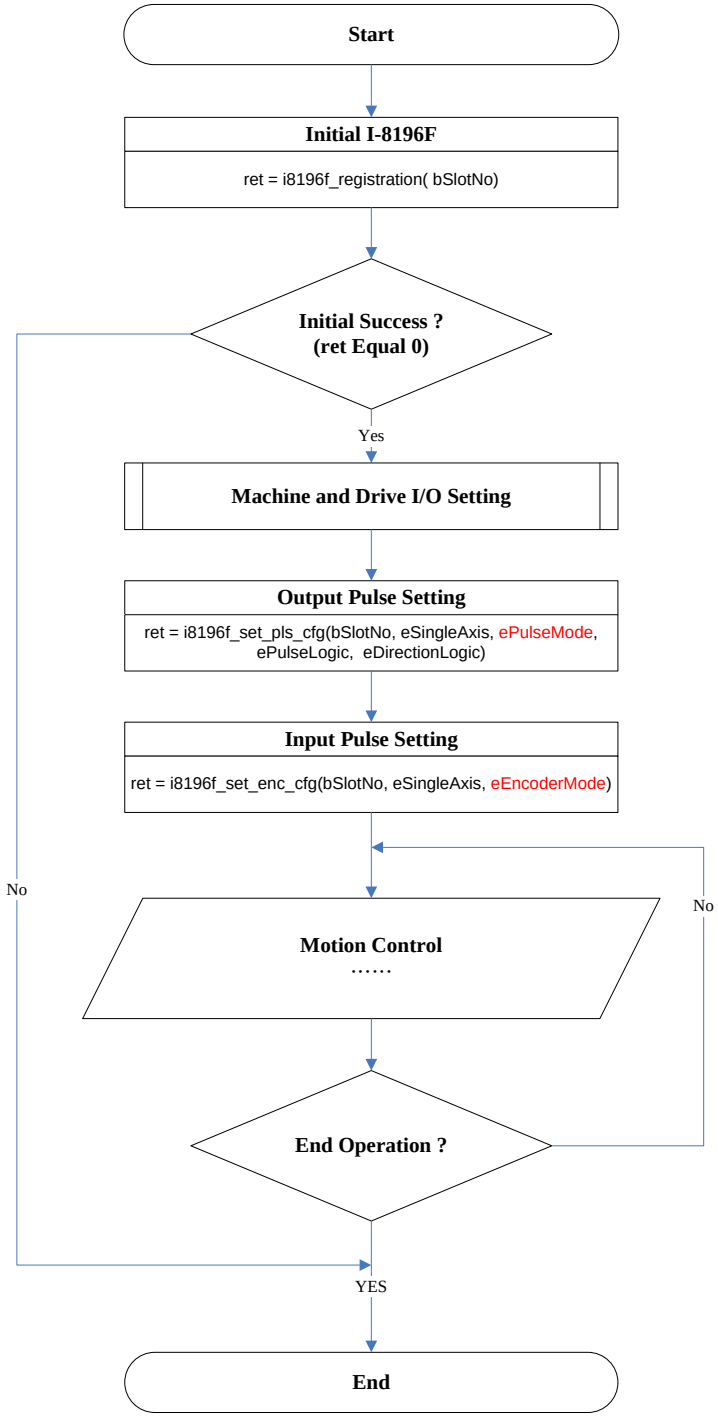
Continuous Interpolation Functions	
i8196f_set_conti_interp_cfg	Assigns axes to an interpolation group and sets the axes group to continuous interpolation mode (see Figure 4, Figure 5 and Figure 6). Once the group has switch to continuous mode, all the arriving commands are being treated as continuous 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.
Motion Stop Functions	
i8196f_stop_move	Stops the current executing motion command for the specified axis. Stops motion before reaching the destination.
i8196f_set_softlimit	Sets the software limits for the positive and negative direction. Once a software limit position is specified, the i8196F will not accept position commands beyond the limit and motion will stop once the limit is hit.
i8196f_set_softlimit_disable	Disables the axis limits settings.
Multi-Axis Hold/Release Functions	
i8196f_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 “i8196f_drv_start()” releases the hold operation.
i8196f_drv_start	Terminates the hold operation. Axes which have been put on hold by “i8196f_drv_hold()” will continue to execute the next motion command stored in the command FIFO buffer.
Compare Function	
i8196f_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	
i8196f_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.
i8196f_get_latch	Reads the present latched position of the specified axis. Returns the captured position triggered by the latch LTC signal.
General Purpose IO	
i8196f_set_general_do	Sets the output state of one general purpose digital output channel.
i8196f_get_general_do	Reads the current output state of the general purpose DO.
i8196f_set_all_general_do	Sets all the general-purpose output signal state.
i8196f_get_all_general_do	Reads the current output state of the general purpose DO.
i8196f_get_general_di	Gets the input state of the general purpose DI channel.
i8196f_get_all_general_di	Gets the input state of all general purpose DIs.
Remote FRnet IO	

i8196f_set_frnet_group_do	Sets the output state of a remote FRnet DO module. Each FRnet DO module has 16 DO channels.
i8196f_get_frnet_group_do	Gets the output state of a remote FRnet DO module. Each FRnet DO module has 16 DO channels.
i8196f_scan_frnet_di	Scans the FRnet network for remote FRnet DI modules and returns which DI modules are connected and active. A maximum number of 8 FRnet DI modules can be used in a FRnet network.
i8196f_get_frnet_di	Gets the channel state of a remote FRnet DI module. Each FRnet DI module has 16 DI channels.
i8196f_get_frnet_group_di	Reads 16 channel states of a remote FRnet DI module. Each FRnet DI module has 16 DI channels.
Hardware Version	
i8196f_get_card_version	Gets the PCB and PLD version.
i8196f_get_fpga_version	Gets the FPGA version.
i8196f_get_dsp_firmware_version	Gets the current DSP firmware version.
i8196f_get_dll_version	Gets the DLL version.

Table 1: I-8196F DLL functions

1.2 Command Flow Chart

This section illustrates the basic function call sequence required for the initialization of the I-8196F and motion command execution.



1.2.1 Axis Digital I/O Setting

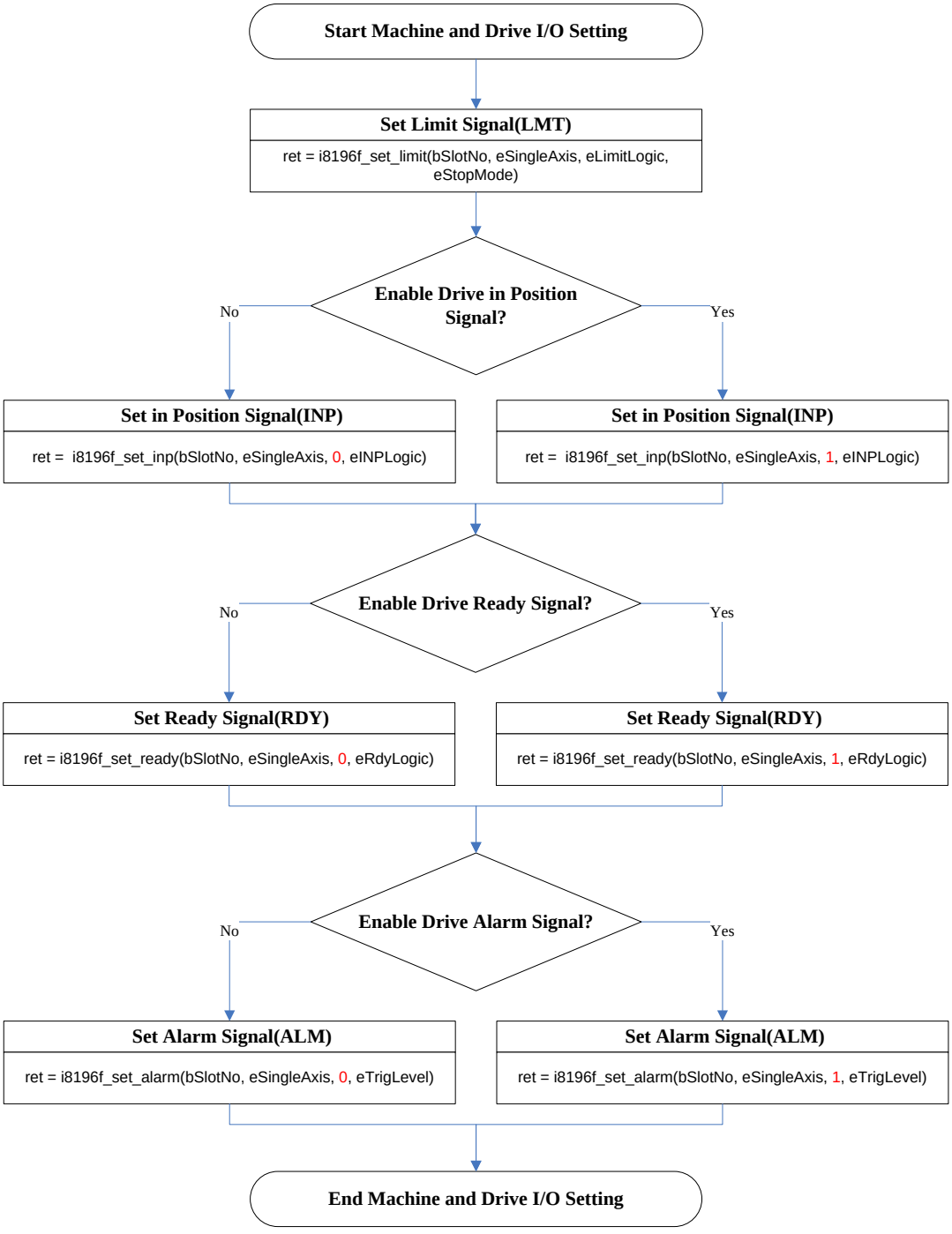


Figure 1: Axis digital I/O setting

1.2.2 Motion Control Initiation

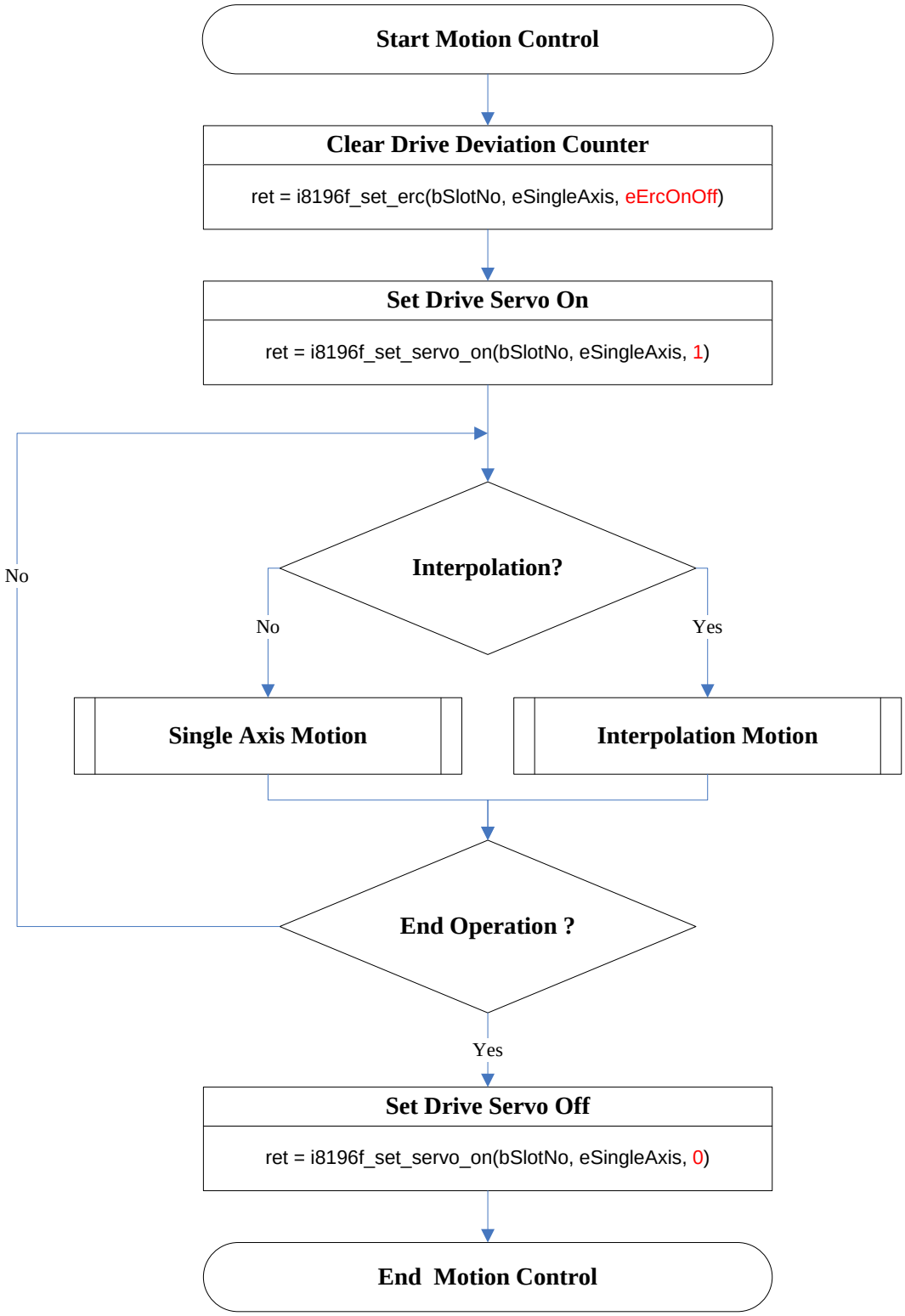


Figure 2: Initiate motion control

1.2.3 Single Axis Motion Control

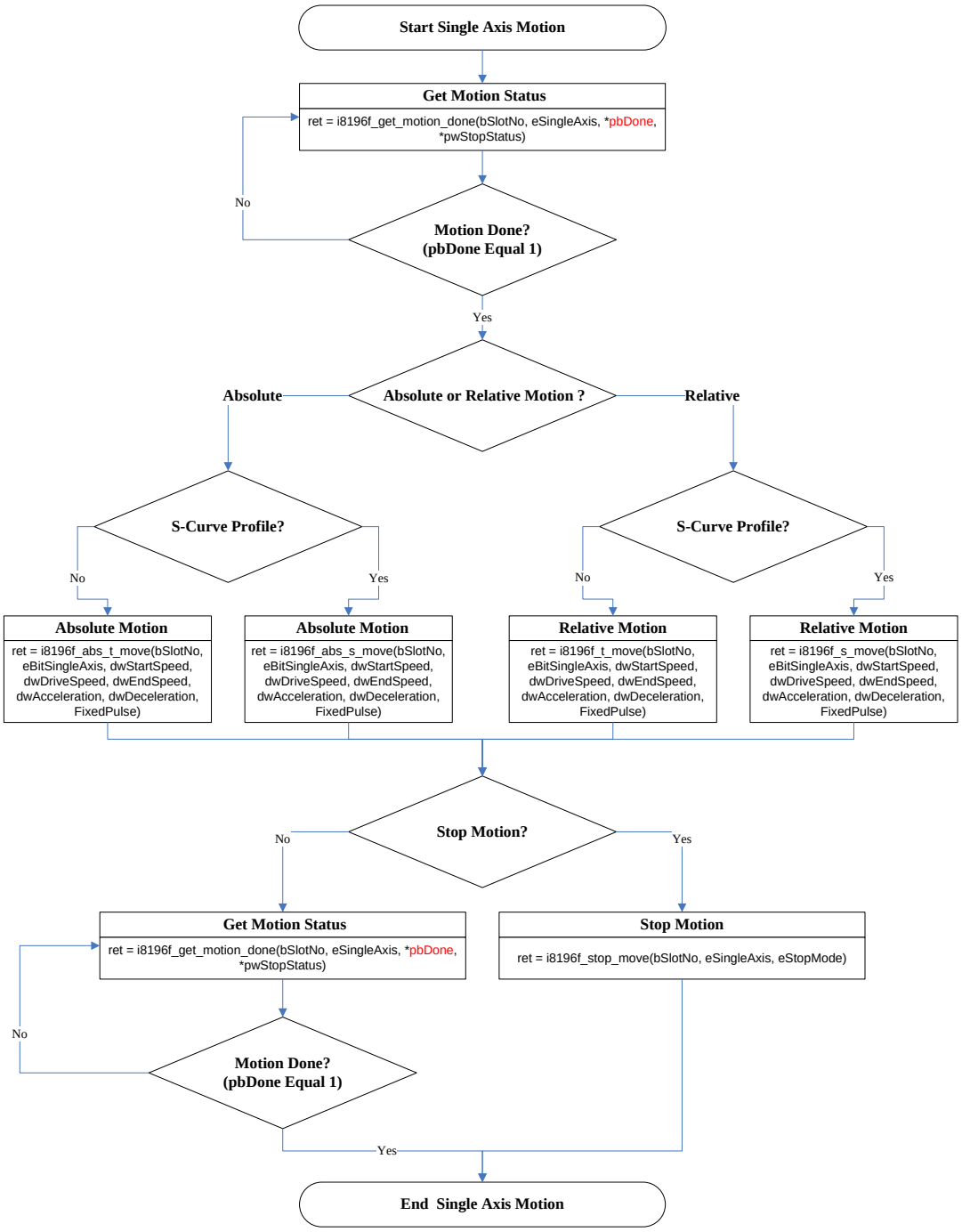


Figure 3: Single axis motion control

1.2.4 Interpolation Motion Control

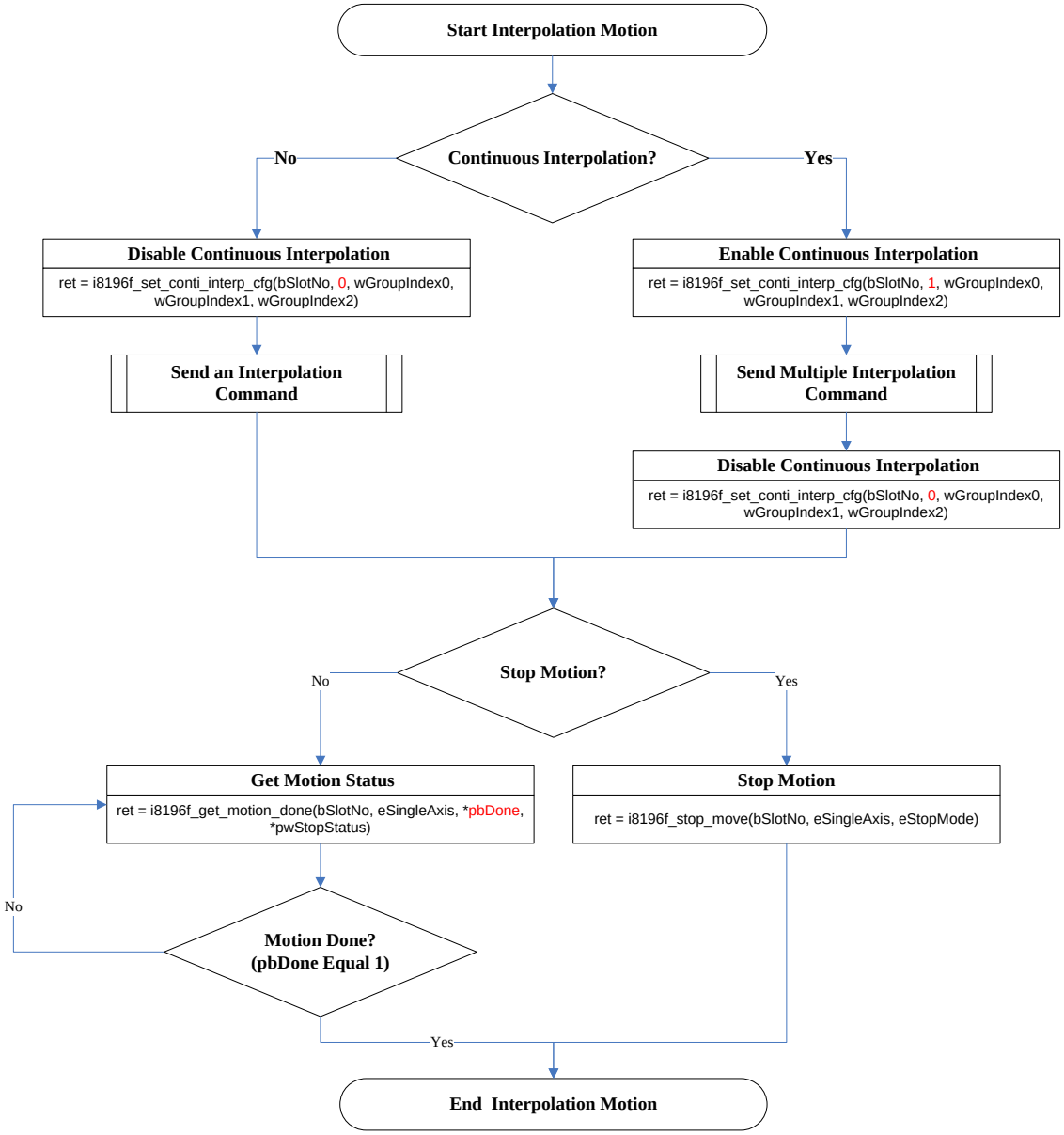


Figure 4: Interpolation motion control settings

1.2.5 Initiate Simple Interpolation Motion

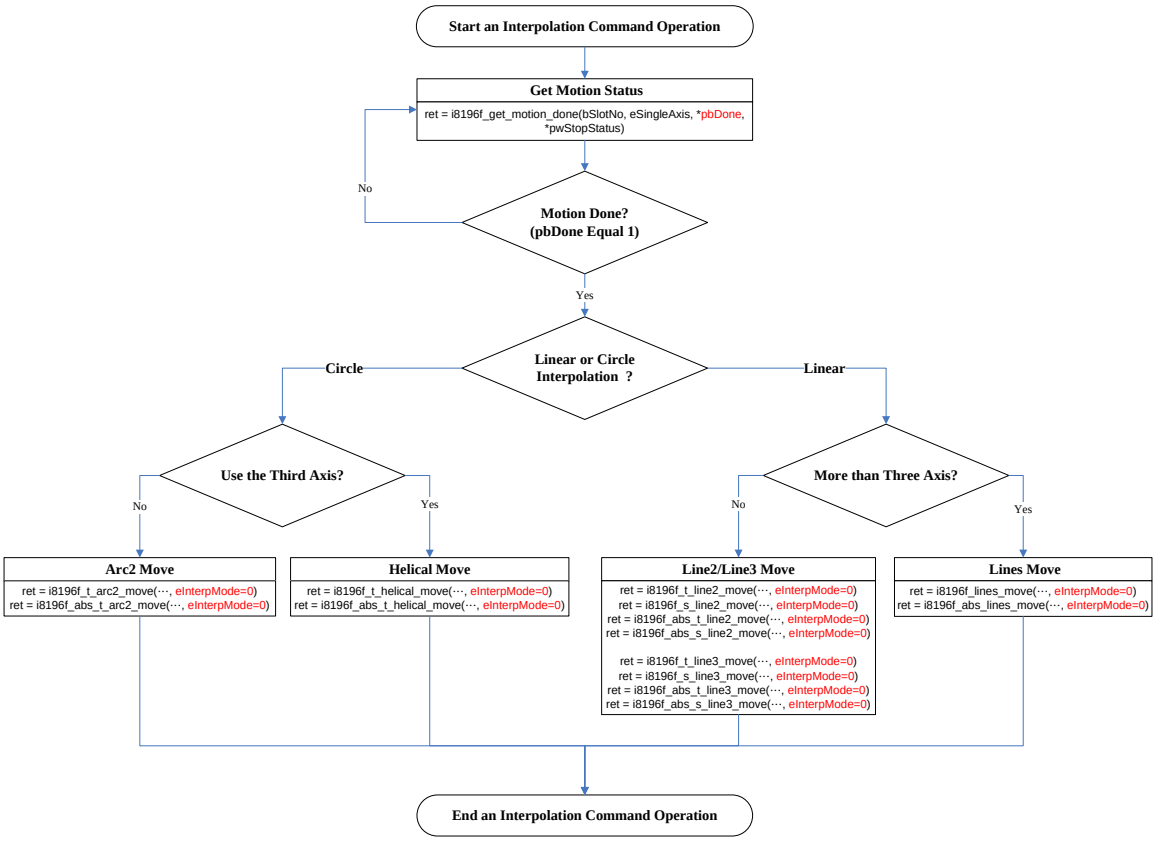


Figure 5: Initiate simple interpolation motion

1.2.6 Initiate Continuous Interpolation Motion

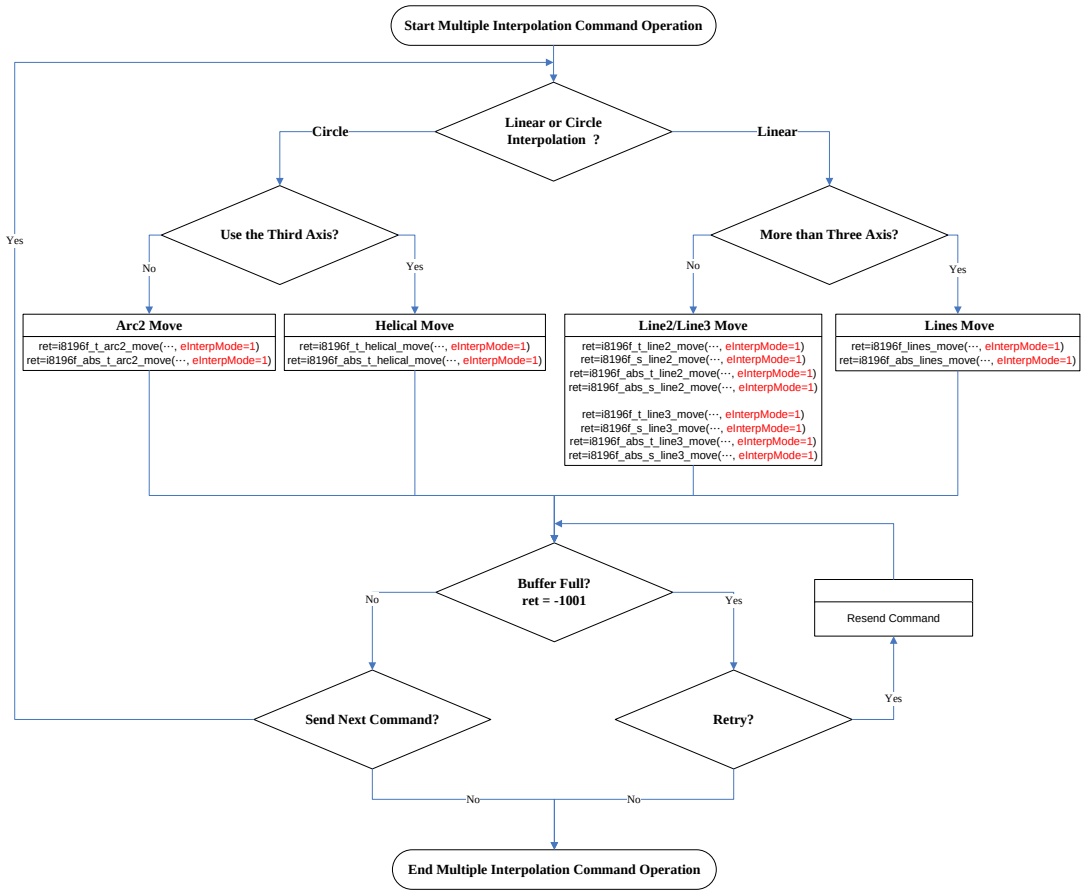


Figure 6: Initiate continuous interpolation motion

2 System Setting APIs

2.1 Initialization

2.1.1 i8196f_registration

Confirms that the I-8196F card is plugged into the specified slot of the PAC and initializes the I-8196F.

Syntax:

```
eRET i8196f_registration(    U8 bSlotNo );
```

Parameters:

Name	Description
bSlotNo:	PAC slot numbering: – 0, 1, 2, …: For WinPAC and ViewPAC the first slot starts at number 0 – 1, 2, 3, …: For XPAC the first slot starts at number 1

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

- This function ensures that the I-8196F module is ready for executing motion commands. It is important to call this function before using the any other APIs (Figure 1).

2.1.2 i8196f_check_card_in_slot

Checks whether the I-8196F card is plugged into the specified slot of the PAC.

Syntax:

```
eRET i8196f_check_card_in_slot (    U8 bSlotNo );
```

Parameters:

Name	Description
bSlotNo:	PAC slot numbering: – 0, 1, 2, …: For WinPAC and ViewPAC the first slot starts at number 0 – 1, 2, 3, …: For XPAC the first slot starts at number 1

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

- This function can be used to scan the slots of the PAC for I-8196F modules.

2.1.3 i8196f_reset

Reset and reinitialize the I-8196F module.

Syntax:

```
eRET i8196f_reset (     U8 bSlotNo );
```

Parameters:

Name	Description
bSlotNo:	PAC slot numbering: – 0, 1, 2, …: For WinPAC and ViewPAC the first slot starts at number 0 – 1, 2, 3, …: For XPAC the first slot starts at number 1

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

- This function is internally power off/on the device for a short period of time. During the reset time the I-8196F device has no control of the servo drive. Therefore do not call this function when the I-8196F needs to be in control of the servo drive or stepping motor at all times.

3 Motion Commands

3.1 Axis Digital I/O Function

The digital IO discussed in this chapter influences the axis operation

3.1.1 i8196f_set_servo_on

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

Syntax:

```
eRET i8196f_set_servo_on (    U8 bSlotNo,
                              eSINGLE_AXIS eSingleAxis,
                              eCHANNEL_STATE eServoOnOff );
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eSingleAxis:	Axis definition: Axis0 AXIS_0 (0x01) Axis1 AXIS_1 (0x02) Axis2 AXIS_2 (0x04) Axis3 AXIS_3 (0x08) Axis4 AXIS_4 (0x10) Axis5 AXIS_5 (0x20)
eServoOnOff:	Status / Value OFF OFF (0x00) ON ON (0x01)

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

3.1.2 i8196f_set_erc

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

Syntax:

```
eRET i8196f_set_erc (    U8 bSlotNo,
                         eSINGLE_AXIS eSingleAxis,
                         eCHANNEL_STATE eErcOnOff);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eSingleAxis:	Axis definition: Axis0 AXIS_0 (0x01) Axis1 AXIS_1 (0x02) Axis2 AXIS_2 (0x04) Axis3 AXIS_3 (0x08) Axis4 AXIS_4 (0x10) Axis5 AXIS_5 (0x20)
eErcOnOff:	OFF OFF (0x00) ON ON (0x01)

Return:

- 0: I8196_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 i8196f_set_alarm_reset

Reset the servo drive alarm signal (ALM_RST).

Syntax:

```
eRET i8196f_set_alarm_reset (    U8 bSlotNo,
                                 eSINGLE_AXIS eSingleAxis,
                                 eCHANNEL_STATE eAlmRstOnOff);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eSingleAxis:	Axis definition: Axis0 AXIS_0 (0x01) Axis1 AXIS_1 (0x02) Axis2 AXIS_2 (0x04) Axis3 AXIS_3 (0x08) Axis4 AXIS_4 (0x10) Axis5 AXIS_5 (0x20)
eAlmRstOnOff:	OFF OFF (0x00) ON ON (0x01)

Return:

- 0: I8196_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 i8196f_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 I-8196F. If the servo alarm function of the I-8196F has been enabled and the servo alarm signal is active then no pulses will be outputted.

Syntax:

```
eRET i8196f_set_alarm (    U8 bSlotNo,
                           eSINGLE_AXIS eSingleAxis,
                           eENABLE_DISABLE eEnableDisable,
                           eLOGIC_ACTIVE_LEVEL eTrigLevel);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eSingleAxis:	Axis definition: Axis0 AXIS_0 (0x01) Axis1 AXIS_1 (0x02) Axis2 AXIS_2 (0x04) Axis3 AXIS_3 (0x08) Axis4 AXIS_4 (0x10) Axis5 AXIS_5 (0x20)
eEnableDisable:	Enable the use of the servo alarm signal Disable DISABLE (0x00) Enable ENABLE (0x01)
eTrigLevel:	Active low LOGIC_ACTIVE_LOW (0x00) Active high LOGIC_ACTIVE_HIGH (0x01)

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

3.1.5 i8196f_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 I-8196F 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.

Syntax:

```
eRET i8196f_set_inp (    U8 bSlotNo,
                         eSINGLE_AXIS eSingleAxis,
                         eENABLE_DISABLE eEnableDisable,
                         eLOGIC_ACTIVE_LEVEL eINPLogic);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eSingleAxis:	Axis definition: Axis0 AXIS_0 (0x01) Axis1 AXIS_1 (0x02) Axis2 AXIS_2 (0x04) Axis3 AXIS_3 (0x08) Axis4 AXIS_4 (0x10) Axis5 AXIS_5 (0x20)
eEnableDisable:	Enable the use of the servo in-position signal Disable DISABLE (0x00) Enable ENABLE (0x01)
eINPLogic:	Active low LOGIC_ACTIVE_LOW (0x00) Active high LOGIC_ACTIVE_HIGH (0x01)

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

- Call “i8196f_get_mdi_status()” to get the INP status and “i8196f_get_motion_done()” to check whether the motion command has finished executing.

3.1.6 i8196f_set_ready

Enables/Disables the I-8196F to check for the servo drive “ready” state and sets the active level of the servo RDY signal.

Syntax:

```
eRET i8196f_set_ready (     U8 bSlotNo,
                            eSINGLE_AXIS eSingleAxis,
                            eENABLE_DISABLE eEnableDisable,
                            eLOGIC_ACTIVE_LEVEL eRdyLogic);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eSingleAxis:	Axis definition: Axis: Axis0 — Value: AXIS_0 (0x01) Axis: Axis1 — Value: AXIS_1 (0x02) Axis: Axis2 — Value: AXIS_2 (0x04) Axis: Axis3 — Value: AXIS_3 (0x08) Axis: Axis4 — Value: AXIS_4 (0x10) Axis: Axis5 — Value: AXIS_5 (0x20)
eEnableDisable:	Enable the use of the servo ready signal State: Disable — Value: DISABLE (0x00) State: Enable — Value: ENABLE (0x01)
eRdyLogic:	Trigger level: Active low — Value: LOGIC_ACTIVE_LOW (0x00) Trigger level: Active high — Value: LOGIC_ACTIVE_HIGH (0x01)

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

- If the servo ready function has been enabled and the RDY signal is OFF then the I-8196F will not execute a motion command until the RDY signal turns ON.
- Call “i8196f_get_mdi_status()” to get the RDY status.

3.1.7 i8196f_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:

```
eRET i8196f_set_limit (    U8 bSlotNo,
                           eSINGLE_AXIS eSingleAxis,
                           eLOGIC_ACTIVE_LEVEL eLimitLogic,
                           eSTOP_MODE eStopMode);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eSingleAxis:	Axis definition: Axis0 AXIS_0 (0x01) Axis1 AXIS_1 (0x02) Axis2 AXIS_2 (0x04) Axis3 AXIS_3 (0x08) Axis4 AXIS_4 (0x10) Axis5 AXIS_5 (0x20)
eLimitLogic:	Active low: LOGIC_ACTIVE_LOW (0x00) Active high: LOGIC_ACTIVE_HIGH (0x01)
eStopMode:	Disable: STOP_NONE (0) Deceleration stop: STOP_SLOWDOWN (1) Immediate stop: STOP_SUDDEN (2)

Return:

- 0: I8196_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 “i8196f_get_mdi_status()” to get the “LMT+” and “LMT-“ status (LMTP, LMTM).

3.1.8 i8196f_get_mdi_status

Reads the current input state of all axis DI channels.

Syntax:

```
eRET i8196f_get_mdi_status (    U8 bSlotNo,
                                eSINGLE_AXIS eSingleAxis,
                                U16* pwDIStatus);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eSingleAxis:	Axis definition: Axis0 AXIS_0 (0x01) Axis1 AXIS_1 (0x02) Axis2 AXIS_2 (0x04) Axis3 AXIS_3 (0x08) Axis4 AXIS_4 (0x10) Axis5 AXIS_5 (0x20)
pwDIStatus:	Bit 0: reserved Bit 1: LMT+ Positive limit switch Bit 2: LMT- Negative limit switch Bit 3: EMG Emergency stop switch Bit 4: ALARM Servo drive alarm signal Bit 5: HOME (ORG) Home switch Bit 6: SLD (NHOME) Slow down switch Bit 7: INP Servo drive in-position signal Bit 8: EZ Servo drive Z phase (Index signal) Bit 9: RDY Servo drive ready signal Bit 10: LTC Latch input Bit 11 ~ 15: reserved If bit is zero: the corresponding signal is OFF If bit is one: the corresponding signal is ON

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

Example:

```
eRET ret = 0;
U16 pwDIStatus = 0;

ret = i8196f_get_mdi_status( 1, AXIS_0, &pwDIStatus );
if (ret == I8196_SUCCESS)
{
      if (pwDIStatus & 0x0008)
      {MessageBox("Emergency Stop!!");}
      else if (pwDIStatus & 0x0002)
      {MessageBox("Reached Positive Limit!!");}
      else if (pwDIStatus & 0x0004)
      {MessageBox("Reached Negative Limit!!");}
}
else
{MessageBox("Get IO Status Error !!!");}
```

3.1.9 i8196f_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 “i8196f_set_servo_on()”.

Syntax:

```
eRET i8196f_get_servo_on_status (   U8 bSlotNo,
                                    eSINGLE_AXIS eSingleAxis,
                                    eCHANNEL_STATE* peServoOn);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eSingleAxis:	Axis definition: Axis0 – AXIS_0 (0x01) Axis1 – AXIS_1 (0x02) Axis2 – AXIS_2 (0x04) Axis3 – AXIS_3 (0x08) Axis4 – AXIS_4 (0x10) Axis5 – AXIS_5 (0x20)
peServoOn:	Status: OFF – OFF (0x00); ON – ON (0x01)

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

3.1.10 i8196f_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 “i8196f_set_erc()”.

Syntax:

```
eRET i8196f_get_servo_erc_status (    U8 bSlotNo,
                                      eSINGLE_AXIS eSingleAxis,
                                      eCHANNEL_STATE* peServoErc);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eSingleAxis:	Axis definition: Axis: Value Axis0: AXIS_0 (0x01) Axis1: AXIS_1 (0x02) Axis2: AXIS_2 (0x04) Axis3: AXIS_3 (0x08) Axis4: AXIS_4 (0x10) Axis5: AXIS_5 (0x20)
peServoErc:	Status: Value OFF: OFF (0x00) ON: ON (0x01)

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

3.1.11 i8196f_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. “i8196f_set_alarm_reset()” sets the output signal.

Syntax:

```
eRET i8196f_get_servo_almrst_status (   U8 bSlotNo,
                                        eSINGLE_AXIS eSingleAxis,
                                        eCHANNEL_STATE* peServoAlarm);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eSingleAxis:	Axis definition: Axis0 AXIS_0 (0x01) Axis1 AXIS_1 (0x02) Axis2 AXIS_2 (0x04) Axis3 AXIS_3 (0x08) Axis4 AXIS_4 (0x10) Axis5 AXIS_5 (0x20)
peServoAlarm:	Status / Value OFF OFF (0x00) ON ON (0x01)

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

3.2 Motion Control Pulse Setting Function

3.2.1 i8196f_set_pls_cfg

Set the pulse output mode for each axis.

Syntax:

```
eRET i8196f_set_pls_cfg (   U8 bSlotNo,
                            eSINGLE_AXIS eSingleAxis,
                            ePULSE_MODE ePulseMode,
                            ePULSE_LOGIC ePulseLogic,
                            eFORWARD_ACTIVE eDirectionLogic);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eSingleAxis:	Axis definition: Axis0 AXIS_0 (0x01) Axis1 AXIS_1 (0x02) Axis2 AXIS_2 (0x04) Axis3 AXIS_3 (0x08) Axis4 AXIS_4 (0x10) Axis5 AXIS_5 (0x20)
ePulseMode:	Mode / Value Pulse/Direction: PULSE_MODE_PULSE_DIRECTION (0) CW/CCW: PULSE_MODE_CW_CCW (1) A/B Phase: PULSE_MODE_AB_DIVID_4 (2)
ePulseLogic:	Logic Level / Value Low: PULSE_LOGIC_ACTIVE_LOW (0x1) High: PULSE_LOGIC_ACTIVE_HIGH (0x0)
eDirectionLogic:	Direction logic-enable signal level: Low: PULSE_FORWARD_ACTIVE_LOW (0x1) High: PULSE_FORWARD_ACTIVE_HIGH (0x0) Note that in "CW / CCW" and "A / B Phase" mode this parameter is invalid

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

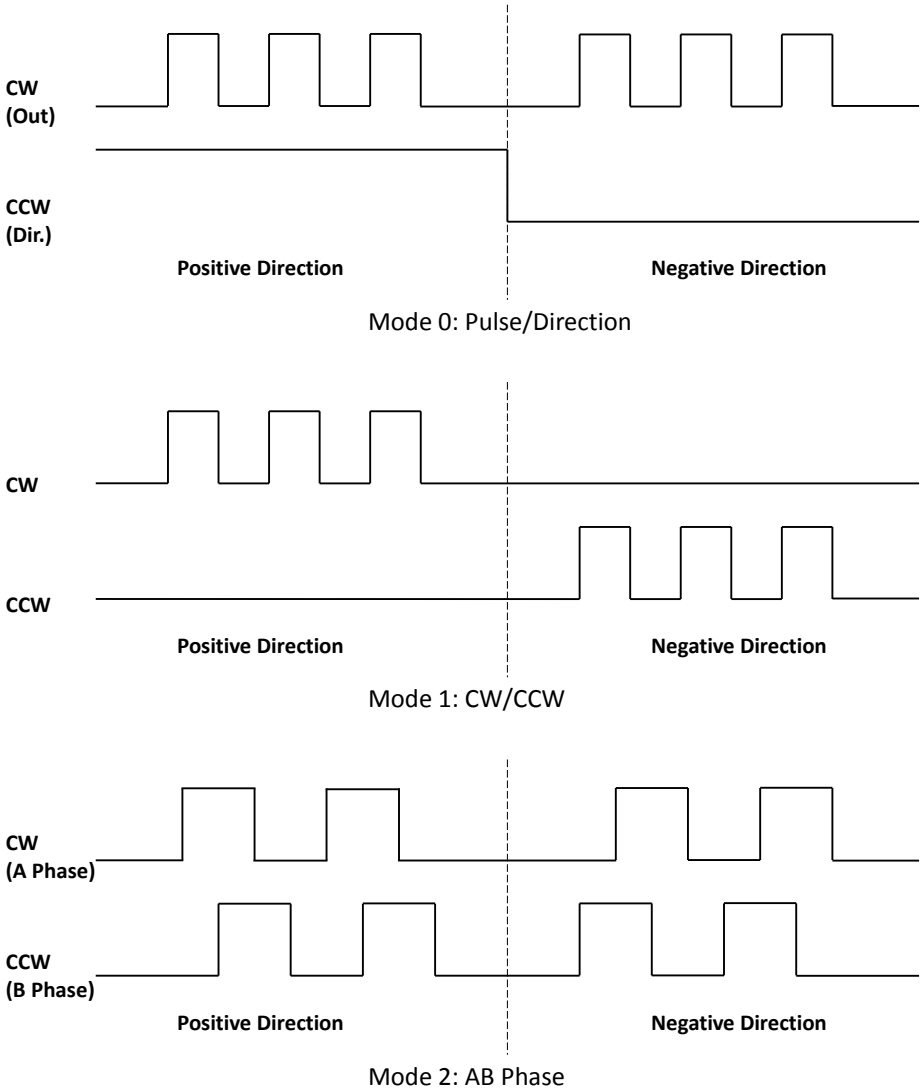


Figure 7: Pulse output modes

3.2.2 i8196f_set_enc_cfg

Set the parameters of the encoder pulse input.

Syntax:

```
eRET i8196f_set_enc_cfg (    U8 bSlotNo,
                             eSINGLE_AXIS eSingleAxis,
                             eENCODER_MODE eEncoderMode);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eSingleAxis:	Axis definition: Axis0 AXIS_0 (0x01) Axis1 AXIS_1 (0x02) Axis2 AXIS_2 (0x04) Axis3 AXIS_3 (0x08) Axis4 AXIS_4 (0x10) Axis5 AXIS_5 (0x20)
eEncoderMode:	CW/CCW ENCODER_MODE_CW_CCW (1) A/B Phase ENCODER_MODE_AB_DIVID_4 (2) A/B Phase divide 2 ENCODER_MODE_AB_DIVID_2 (3) A/B Phase divide 4 ENCODER_MODE_AB (4)

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

3.2.3 i8196f_set_cmdcounter

Set the command counter (position command) value.

Syntax:

```
eRET i8196f_set_cmdcounter (    U8 bSlotNo,
                                eSINGLE_AXIS eSingleAxis,
                                I32 lLogicPos );
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eSingleAxis:	Axis definition: Axis0 AXIS_0 (0x01) Axis1 AXIS_1 (0x02) Axis2 AXIS_2 (0x04) Axis3 AXIS_3 (0x08) Axis4 AXIS_4 (0x10) Axis5 AXIS_5 (0x20)
lLogicPos:	Command counter value

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

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

3.2.4 i8196f_get_cmdcounter

Get the current command counter (position command) value.

Syntax:

```
eRET i8196f_get_cmdcounter (    U8 bSlotNo,
                                eSINGLE_AXIS eSingleAxis,
                                I32* plLogicPosCount );
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eSingleAxis:	Axis definition: Axis0 — AXIS_0 (0x01) Axis1 — AXIS_1 (0x02) Axis2 — AXIS_2 (0x04) Axis3 — AXIS_3 (0x08) Axis4 — AXIS_4 (0x10) Axis5 — AXIS_5 (0x20)
plLogicPosCount:	Pointer to current command counter (position command) value

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

3.2.5 i8196f_set_enccounter

Set the encoder counter value.

Syntax:

```
eRET i8196f_set_enccounter (    U8 bSlotNo,
                                eSINGLE_AXIS eSingleAxis,
                                I32 lEncPos);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eSingleAxis:	Axis definition: Axis0 AXIS_0 (0x01) Axis1 AXIS_1 (0x02) Axis2 AXIS_2 (0x04) Axis3 AXIS_3 (0x08) Axis4 AXIS_4 (0x10) Axis5 AXIS_5 (0x20)
lEncPos:	Encoder counter value

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

3.2.6 i8196f_get_enccounter

Get the current encoder counter value.

Syntax:

```
eRET i8196f_get_enccounter (    U8 bSlotNo,
                                eSINGLE_AXIS eSingleAxis,
                                I32* plEncoderPosCount);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eSingleAxis:	Axis definition: Axis0 – AXIS_0 (0x01) Axis1 – AXIS_1 (0x02) Axis2 – AXIS_2 (0x04) Axis3 – AXIS_3 (0x08) Axis4 – AXIS_4 (0x10) Axis5 – AXIS_5 (0x20)
plEncoderPosCount:	Pointer to current encoder counter

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

3.2.7 i8196f_set_vring_counter

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

Syntax:

```
eRET i8196f_set_vring_counter (    U8 bSlotNo,
                                   eSINGLE_AXIS eSingleAxis,
                                   U32 dwRingValue);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eSingleAxis:	Axis definition: Axis0 AXIS_0 (0x01) Axis1 AXIS_1 (0x02) Axis2 AXIS_2 (0x04) Axis3 AXIS_3 (0x08) Axis4 AXIS_4 (0x10) Axis5 AXIS_5 (0x20)
dwRingValue:	The upper limit of the encoder counter value (range: 2~2147483647)

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

- The counter setting will disable the software limit setting
- Ring counter function is not support when axis is in compare trigger mode
- Use the function "i8196f_disable_vring_counter()" to turn off the ring counter setting
- The ring position counter operation:

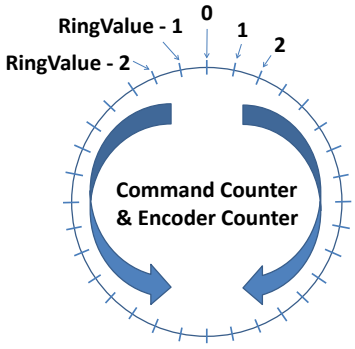


Figure 8: Ring position counter

3.2.8 i8196f_get_vring_counter

Read the maximum ring counter setting.

Syntax:

```
eRET i8196f_get_vring_counter (    U8 bSlotNo,
                                   eSINGLE_AXIS eSingleAxis,
                                   U32* pdwRingValue);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eSingleAxis:	Axis definition: Axis0 – AXIS_0 (0x01) Axis1 – AXIS_1 (0x02) Axis2 – AXIS_2 (0x04) Axis3 – AXIS_3 (0x08) Axis4 – AXIS_4 (0x10) Axis5 – AXIS_5 (0x20)
pdwRingValue:	• Pointer to current ring counter value setting (range: 2~2147483647) • Returns the value set by “i8196f_set_vring_counter()”

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

3.2.9 i8196f_disable_vring_counter

Disable the ring counter setting.

Syntax:

```
eRET i8196f_disable_vring_counter (    U8 bSlotNo,
                                       eSINGLE_AXIS eSingleAxis);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eSingleAxis:	Axis definition: Axis0 AXIS_0 (0x01) Axis1 AXIS_1 (0x02) Axis2 AXIS_2 (0x04) Axis3 AXIS_3 (0x08) Axis4 AXIS_4 (0x10) Axis5 AXIS_5 (0x20)

Return:

- 0: I8196_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 i8196f_set_home_cfg

Set automatic home search parameters.

Syntax:

```
eRET i8196f_set_home_cfg (    U8 bSlotNo,
                              eSINGLE_AXIS eSingleAxis,
                              eLOGIC_ACTIVE_LEVEL eHomeLogic,
                              eLOGIC_ACTIVE_LEVEL eNHomeLogic,
                              eLOGIC_ACTIVE_LEVEL eIndexLogic,
                              U8 bHomeSteps,
                              I32 lStep4Offset);
```

Parameters:

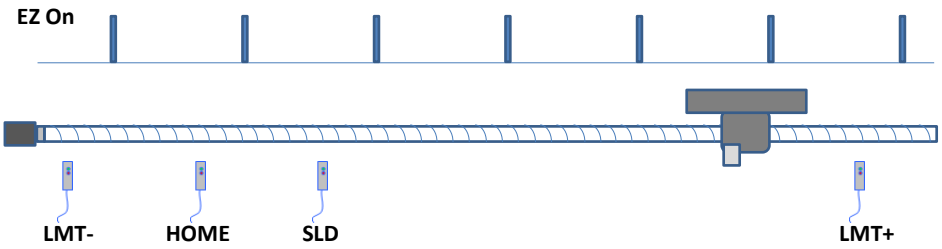
Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eSingleAxis:	Axis definition: Axis0 — AXIS_0 (0x01) Axis1 — AXIS_1 (0x02) Axis2 — AXIS_2 (0x04) Axis3 — AXIS_3 (0x08) Axis4 — AXIS_4 (0x10) Axis5 — AXIS_5 (0x20)
eHomeLogic:	Logic level of origin (HOME) input signal: Active low — LOGIC_ACTIVE_LOW (0x00) Active high — LOGIC_ACTIVE_HIGH (0x01)
eNHomeLogic:	Logic level of the slow down (SLD) input signal: Active low — LOGIC_ACTIVE_LOW (0x00) Active high — LOGIC_ACTIVE_HIGH (0x01)
eIndexLogic:	Logic level of servo drive Z phase (Index signal) Active low — LOGIC_ACTIVE_LOW (0x00) Active high — LOGIC_ACTIVE_HIGH (0x01)
bHomeSteps:	

	Bit Position	Corresponding Home Step
	Bit 1	Step1: High speed near home search(Search SLD)
	Bit 2	Reserved
	Bit 3	Step2: Low speed home search (Search HOME)
	Bit 4	Reserved
	Bit 5	Step3: Low speed servo drive Z phase (Index signal) search (Search EZ)
	Bit 6	Reserved
	Bit 7	Step4: High speed offset drive (Offset)
	Bit 8	Reserved
lStep4Offset:	Offset position	

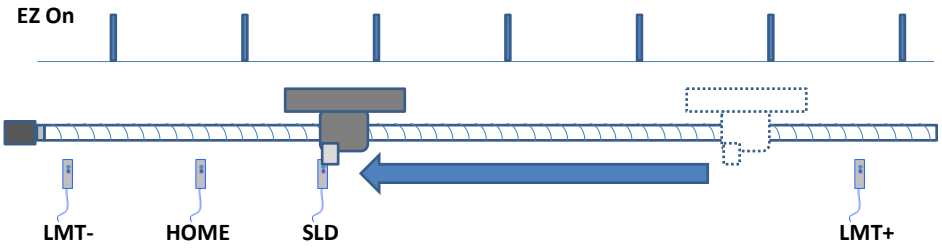
Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

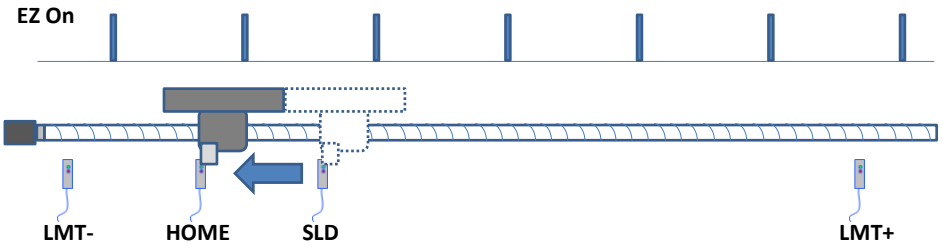
Remarks:



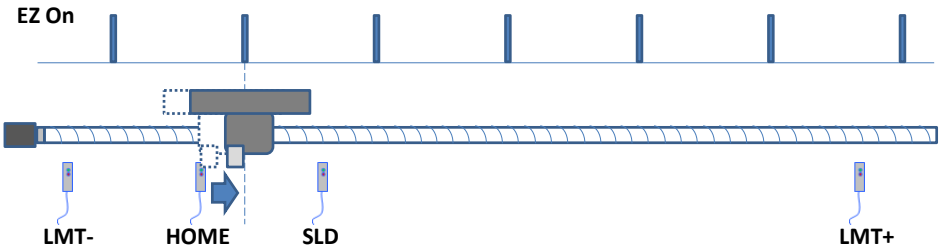
Step 0: Wait auto homing command



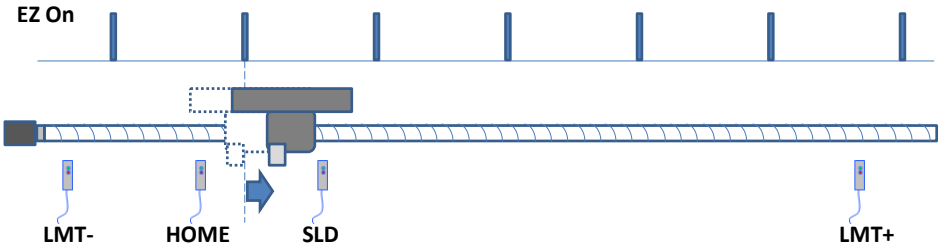
Step 1: Search SLD signal (high speed).



Step 2: Search HOME signal (low speed)



Step 3: Search EZ signal (low speed)



Step 4: Offset (high speed).

Figure 9: Automatic home search

4.2 Automatic Home Execution

4.2.1 i8196f_home_start

Start searching for the home position.

Syntax:

```
eRET i8196f_home_start (    U8 bSlotNo,
                            eSINGLE_AXIS eSingleAxis,
                            U32 dwStartSpeed,
                            U32 dwAcceleration,
                            U32 dwDeceleration,
                            U32 dwNHomeSearchSpeed,
                            U32 dwHomeSearchSpeed,
                            eHOME_DIR eHomingDirection);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eSingleAxis:	Axis definition: Axis0 — AXIS_0 (0x01) Axis1 — AXIS_1 (0x02) Axis2 — AXIS_2 (0x04) Axis3 — AXIS_3 (0x08) Axis4 — AXIS_4 (0x10) Axis5 — AXIS_5 (0x20)
dwStartSpeed:	Start speed (PPS)
dwAcceleration:	Acceleration (PPS/Sec)
dwDeceleration:	Deceleration (PPS/Sec)
dwNHomeSearchSpeed:	Near home search (step 1) and offset drive (step 4) speed (PPS) (*dwNHomeSearchSpeed* > *dwHomeSearchSpeed*)
dwHomeSearchSpeed:	Home search (step 2) and servo drive servo drive Z phase (Index) search (step 3) speed
eHomingDirection:	Home search direction Negative — AUTO_HOME_REVERSE (0) Positive — AUTO_HOME_FORWARD (1)

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

5 Motion Control Instructions

5.1 Read Motion Status

5.1.1 i8196f_get_motion_done

Read the current motion status of the axis.

Syntax:

```
eRET i8196f_get_motion_done (   U8 bSlotNo,
                                eSINGLE_AXIS eSingleAxis,
                                U8* pbDone,
                                U16* pwStopStatus);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eSingleAxis:	Axis definition: Axis0 AXIS_0 (0x01) Axis1 AXIS_1 (0x02) Axis2 AXIS_2 (0x04) Axis3 AXIS_3 (0x08) Axis4 AXIS_4 (0x10) Axis5 AXIS_5 (0x20)
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 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 "i8196f_home_start()") DRIVE_FINISH_WITH_AUTO_HOME Bit 2: Motion command has been interrupted by a stop command ("i8196f_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 "i8196f_set_inp()") DRIVE_FINISH_WAIT_FOR_INPOS

	Bit 4	Motion has been aborted because the maximum positive position has been exceeded (see “i8196f_set_softlimit()”) DRIVE_FINISH_WITH_SW_LIMIT_POSITIVE
	Bit 5	Maximum negative position has been exceeded (see “i8196f_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: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

5.1.2 i8196f_get_speed

Get the current axis speed.

Syntax:

```
eRET i8196f_get_speed (   U8 bSlotNo,
                          eSINGLE_AXIS eSingleAxis,
                          I32* plSpeed);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eSingleAxis:	Axis definition: Axis0 — AXIS_0 (0x01) Axis1 — AXIS_1 (0x02) Axis2 — AXIS_2 (0x04) Axis3 — AXIS_3 (0x08) Axis4 — AXIS_4 (0x10) Axis5 — AXIS_5 (0x20)
plSpeed:	Pointer to current axis speed (PPS)

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

5.1.3 i8196f_get_acc

Get the axis current axis acceleration.

Syntax:

```
eRET i8196f_get_acc (    U8 bSlotNo,
                         eSINGLE_AXIS eSingleAxis,
                         I32* plAcc);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eSingleAxis:	Axis definition: Axis0 — AXIS_0 (0x01) Axis1 — AXIS_1 (0x02) Axis2 — AXIS_2 (0x04) Axis3 — AXIS_3 (0x08) Axis4 — AXIS_4 (0x10) Axis5 — AXIS_5 (0x20)
plAcc:	Pointer to current axis acceleration (PPS/Sec)

Return:

- 0: I8196_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 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.

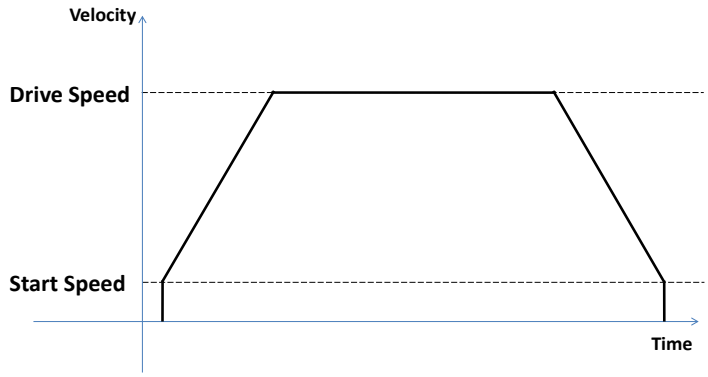


Figure 10: T-Curve velocity profile

S-Curve:

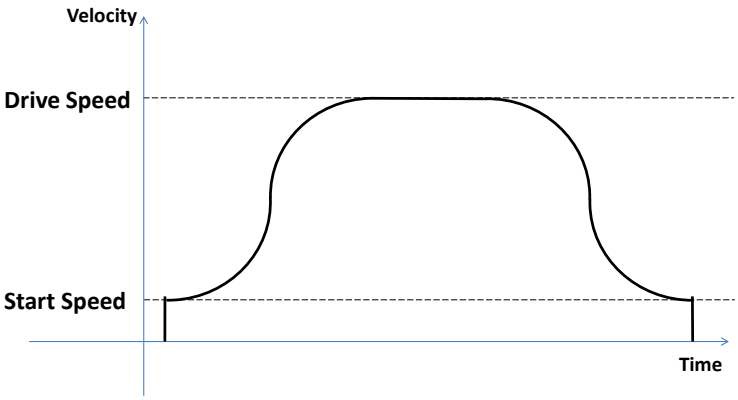


Figure 11: S-Curve velocity profile

5.2.2 i8196f_t_move

Execute a single axis, relative position motion command with a trapezoidal velocity profile (T-curve). The *i8196f_t_move* instruction moves the axis the specified travel distance from the current position.

Syntax:

```
eRET i8196f_t_move (    U8 bSlotNo,
                        eSINGLE_AXIS eSingleAxis,
                        U32 dwStartSpeed,
                        U32 dwDriveSpeed,
                        U32 dwEndSpeed,
                        U32 dwAcceleration,
                        U32 dwDeceleration,
                        I32  FixedPulse );
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eSingleAxis:	Axis definition: Axis0 AXIS_0 (0x01) Axis1 AXIS_1 (0x02) Axis2 AXIS_2 (0x04) Axis3 AXIS_3 (0x08) Axis4 AXIS_4 (0x10) Axis5 AXIS_5 (0x20)
dwStartSpeed:	Start speed (PPS)
dwDriveSpeed:	Drive speed (PPS)
dwEndSpeed:	End speed (PPS)
dwAcceleration:	Acceleration (PPS/Sec)
dwDeceleration:	Deceleration (PPS/Sec)
FixedPulse:	Relative moving distance (Pulse) > 0: driving in positive direction < 0: driving in negative direction

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

5.2.3 i8196f_abs_t_move

Execute a single axis, absolute position motion command with a trapezoidal velocity profile (T-curve). The *i8196f_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:

```
eRET i8196f_abs_t_move (    U8 bSlotNo,
                            eSINGLE_AXIS eSingleAxis,
                            U32 dwStartSpeed,
                            U32 dwDriveSpeed,
                            U32 dwEndSpeed,
                            U32 dwAcceleration,
                            U32 dwDeceleration,
                            I32  FixedPulse );
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eSingleAxis:	Axis definition: Axis0 AXIS_0 (0x01) Axis1 AXIS_1 (0x02) Axis2 AXIS_2 (0x04) Axis3 AXIS_3 (0x08) Axis4 AXIS_4 (0x10) Axis5 AXIS_5 (0x20)
dwStartSpeed:	Start speed (PPS)
dwDriveSpeed:	Drive speed (PPS)
dwEndSpeed:	End speed (PPS)
dwAcceleration:	Acceleration (PPS/Sec)
dwDeceleration:	Deceleration (PPS/Sec)
FixedPulse:	Absolute position (Pulse)

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

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

5.2.4 i8196f_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.

Syntax:

```
eRET i8196f_s_move (    U8 bSlotNo,
                        eSINGLE_AXIS eSingleAxis,
                        U32 dwStartSpeed,
                        U32 dwDriveSpeed,
                        U32 dwEndSpeed,
                        U32 dwAcceleration,
                        U32 dwDeceleration,
                        I32  FixedPulse );
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eSingleAxis:	Axis definition: Axis0 AXIS_0 (0x01) Axis1 AXIS_1 (0x02) Axis2 AXIS_2 (0x04) Axis3 AXIS_3 (0x08) Axis4 AXIS_4 (0x10) Axis5 AXIS_5 (0x20)
dwStartSpeed:	Start speed (PPS)
dwDriveSpeed:	Drive speed (PPS)
dwEndSpeed:	End speed (PPS)
dwAcceleration:	Acceleration (PPS/Sec)
dwDeceleration:	Deceleration (PPS/Sec)
FixedPulse:	Relative moving distance (Pulse) > 0: driving in positive direction < 0: driving in negative direction

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

5.2.5 i8196f_abs_s_move

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

Syntax:

```
eRET i8196f_abs_s_move (    U8 bSlotNo,
                            eSINGLE_AXIS eSingleAxis,
                            U32 dwStartSpeed,
                            U32 dwDriveSpeed,
                            U32 dwEndSpeed,
                            U32 dwAcceleration,
                            U32 dwDeceleration,
                            I32  FixedPulse );
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eSingleAxis:	Axis definition: Axis0 AXIS_0 (0x01) Axis1 AXIS_1 (0x02) Axis2 AXIS_2 (0x04) Axis3 AXIS_3 (0x08) Axis4 AXIS_4 (0x10) Axis5 AXIS_5 (0x20)
dwStartSpeed:	Start speed (PPS)
dwDriveSpeed:	Drive speed (PPS)
dwEndSpeed:	End speed (PPS)
dwAcceleration:	Acceleration (PPS/Sec)
dwDeceleration:	Deceleration (PPS/Sec)
FixedPulse:	Absolute position (Pulse)

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

5.2.6 i8196f_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.

Syntax:

```
eRET i8196f_velocity_move (     U8 bSlotNo,
                                eSINGLE_AXIS eSingleAxis,
                                U32 dwStartSpeed,
                                U32 dwDriveSpeed,
                                U32 dwAcceleration,
                                eMOVE_DIRECTION  eDirection);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eSingleAxis:	Axis definition: Axis0 AXIS_0 (0x01) Axis1 AXIS_1 (0x02) Axis2 AXIS_2 (0x04) Axis3 AXIS_3 (0x08) Axis4 AXIS_4 (0x10) Axis5 AXIS_5 (0x20)
dwStartSpeed:	Start speed (PPS)
dwDriveSpeed:	Drive speed (PPS)
dwAcceleration:	Acceleration (PPS/Sec)
eDirection:	Driving direction: Negative MOVE_DIRECTION_REVERSE (0) Positive MOVE_DIRECTION_FORWARD (1)

Return:

- 0: I8196_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 i8196f_t_line2_move

Executes a two axes relative distance linear interpolation motion command with a T-curve velocity profile. The *i8196f_t_line2_move* instruction performs linear interpolation for two axes. The target position is specified as a relative position.

Syntax:

```
eRET i8196f_t_line2_move (    U8 bSlotNo,
                              eSINGLE_AXIS eMainAxis,
                              eSINGLE_AXIS eSlaveAxis,
                              U32 dwStartSpeed,
                              U32 dwDriveSpeed,
                              U32 dwEndSpeed,
                              U32 dwAcceleration,
                              U32 dwDeceleration,
                              I32 lMainAxisRelDist,
                              I32 lSlaveAxisRelDist,
                              eINTERPOLATION_MODE eInterpMode);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eMainAxis: *eSlaveAxis:*	Axis definition: Axis0 AXIS_0 (0x01) Axis1 AXIS_1 (0x02) Axis2 AXIS_2 (0x04) Axis3 AXIS_3 (0x08) Axis4 AXIS_4 (0x10) Axis5 AXIS_5 (0x20)
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

<table>
<tr><td></td><td>< 0: relative distance in negative direction</td></tr>
<tr><td>lSlaveAxisRelDist:</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>eInterpMode:</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: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

5.3.2 i8196f_abs_t_line2_move

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

Syntax:

```
eRET i8196f_abs_t_line2_move (  U8 bSlotNo,
                                eSINGLE_AXIS eMainAxis,
                                eSINGLE_AXIS eSlaveAxis,
                                U32 dwStartSpeed,
                                U32 dwDriveSpeed,
                                U32 dwEndSpeed,
                                U32 dwAcceleration,
                                U32 dwDeceleration,
                                I32 lMainAxisFinishPoint,
                                I32 lSlaveAxisFinishPoint,
                                eINTERPOLATION_MODE eInterpMode);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eMainAxis: *eSlaveAxis:*	Axis definition: Axis0 – AXIS_0 (0x01) Axis1 – AXIS_1 (0x02) Axis2 – AXIS_2 (0x04) Axis3 – AXIS_3 (0x08) Axis4 – AXIS_4 (0x10) Axis5 – AXIS_5 (0x20)
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)
lMainAxisFinishPoint:	Absolute end position of the main axis (Pulse)
lSlaveAxisFinishPoint:	Absolute end position of the slave axis (Pulse)
eInterpMode:	Command execution mode: General – GENERAL_INTERP_CONFIG (0) • No command buffering takes place. • A new command can only be executed if the previous command has finished. Continuous – CONTI_INTERP_CONFIG (1)

	interpolation	• 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: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

5.3.3 i8196f_s_line2_move

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

Syntax:

```
eRET i8196f_s_line2_move (    U8 bSlotNo,
                              eSINGLE_AXIS eMainAxis,
                              eSINGLE_AXIS eSlaveAxis,
                              U32 dwStartSpeed,
                              U32 dwDriveSpeed,
                              U32 dwEndSpeed,
                              U32 dwAcceleration,
                              U32 dwDeceleration,
                              I32 lMainAxisRelDist,
                              I32 lSlaveAxisRelDist,
                              eINTERPOLATION_MODE eInterpMode);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eMainAxis: *eSlaveAxis:*	Axis definition: Axis0 AXIS_0 (0x01) Axis1 AXIS_1 (0x02) Axis2 AXIS_2 (0x04) Axis3 AXIS_3 (0x08) Axis4 AXIS_4 (0x10) Axis5 AXIS_5 (0x20)
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
lSlaveAxisRelDist:	Relative distance of the slave axis (Pulse) > 0: relative distance in positive direction < 0: relative distance in negative direction
eInterpMode:	Command execution mode: Mode: General — Value: 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: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

5.3.4 i8196f_abs_s_line2_move

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

Syntax:

```
eRET i8196f_abs_s_line2_move (  U8 bSlotNo,
                                eSINGLE_AXIS eMainAxis,
                                eSINGLE_AXIS eSlaveAxis,
                                U32 dwStartSpeed,
                                U32 dwDriveSpeed,
                                U32 dwEndSpeed,
                                U32 dwAcceleration,
                                U32 dwDeceleration,
                                I32 lMainAxisFinishPoint,
                                I32 lSlaveAxisFinishPoint,
                                eINTERPOLATION_MODE eInterpMode);
```

Parameters:

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>bSlotNo:</td><td>PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)</td></tr>
<tr><td>eMainAxis:

eSlaveAxis:</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>
<tr><td>Axis2</td><td>AXIS_2 (0x04)</td></tr>
<tr><td>Axis3</td><td>AXIS_3 (0x08)</td></tr>
<tr><td>Axis4</td><td>AXIS_4 (0x10)</td></tr>
<tr><td>Axis5</td><td>AXIS_5 (0x20)</td></tr>
</table></td></tr>
<tr><td>dwStartSpeed:</td><td>Magnitude of start velocity vector (PPS)</td></tr>
<tr><td>dwDriveSpeed:</td><td>Magnitude of drive velocity vector (PPS)</td></tr>
<tr><td>dwEndSpeed:</td><td>Magnitude of end velocity vector (PPS)</td></tr>
<tr><td>dwAcceleration:</td><td>Magnitude of acceleration vector (PPS/Sec)</td></tr>
<tr><td>dwDeceleration:</td><td>Magnitude of deceleration vector (PPS/Sec)</td></tr>
<tr><td>lMainAxisFinishPoint:</td><td>Absolute end position of the main axis (Pulse)</td></tr>
<tr><td>lSlaveAxisFinishPoint:</td><td>Absolute end position of the slave axis (Pulse)</td></tr>
<tr><td>eInterpMode:</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</td><td>CONTI_INTERP_CONFIG (1)</td></tr>
</table></td></tr>
</table>

	interpolation	• 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: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

5.4 Three Axis Linear Interpolation Commands

5.4.1 i8196f_t_line3_move

Executes a three axes linear interpolation motion command with a T-curve velocity profile. The target position is specified as a relative distance to the current position.

Syntax:

```
eRET i8196f_t_line3_move (  U8 bSlotNo,
                            eSINGLE_AXIS eMainAxis,
                            eSINGLE_AXIS eSecondAxis,
                            eSINGLE_AXIS eThirdAxis,
                            U32 dwStartSpeed,
                            U32 dwDriveSpeed,
                            U32 dwEndSpeed,
                            U32 dwAcceleration,
                            U32 dwDeceleration,
                            I32 lMainAxisRelDist,
                            I32 lSecondAxisRelDist,
                            I32 lThirdAxisRelDist,
                            eINTERPOLATION_MODE eInterpMode);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eMainAxis: *eSecondAxis:* *eThirdAxis:*	Axis definition: Axis0 – AXIS_0 (0x01) Axis1 – AXIS_1 (0x02) Axis2 – AXIS_2 (0x04) Axis3 – AXIS_3 (0x08) Axis4 – AXIS_4 (0x10) Axis5 – AXIS_5 (0x20)
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
lSecondAxisRelDist:	Relative distance of the second axis (Pulse) > 0: relative distance in positive direction

<table>
<tr><td></td><td>< 0: relative distance in negative direction</td></tr>
<tr><td>lThirdAxisRelDist:</td><td>Relative distance of the third axis (Pulse)
> 0: relative distance in positive direction
< 0: relative distance in negative direction</td></tr>
<tr><td>eInterpMode:</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: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

5.4.2 i8196f_abs_t_line3_move

Executes a three axes linear interpolation motion command with a T-curve velocity profile. This instruction moves the axes to an absolute position relative to (0) zero.

Syntax:

```
eRET i8196f_abs_t_line3_move (    U8 bSlotNo,
                                  eSINGLE_AXIS eMainAxis,
                                  eSINGLE_AXIS eSecondAxis,
                                  eSINGLE_AXIS eThirdAxis,
                                  U32 dwStartSpeed,
                                  U32 dwDriveSpeed,
                                  U32 dwEndSpeed,
                                  U32 dwAcceleration,
                                  U32 dwDeceleration,
                                  I32 lMainAxisFinishPoint,
                                  I32 lSecondAxisFinishPoint,
                                  I32 lThirdAxisFinishPoint,
                                  eINTERPOLATION_MODE eInterpMode);
```

Parameters:

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>bSlotNo:</td><td>PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)</td></tr>
<tr><td>eMainAxis:

eSecondAxis:

eThirdAxis:</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>
<tr><td>Axis2</td><td>AXIS_2 (0x04)</td></tr>
<tr><td>Axis3</td><td>AXIS_3 (0x08)</td></tr>
<tr><td>Axis4</td><td>AXIS_4 (0x10)</td></tr>
<tr><td>Axis5</td><td>AXIS_5 (0x20)</td></tr>
</table></td></tr>
<tr><td>dwStartSpeed:</td><td>Magnitude of start velocity vector (PPS)</td></tr>
<tr><td>dwDriveSpeed:</td><td>Magnitude of drive velocity vector (PPS)</td></tr>
<tr><td>dwEndSpeed:</td><td>Magnitude of end velocity vector (PPS)</td></tr>
<tr><td>dwAcceleration:</td><td>Magnitude of acceleration vector (PPS/Sec)</td></tr>
<tr><td>dwDeceleration:</td><td>Magnitude of deceleration vector (PPS/Sec)</td></tr>
<tr><td>lMainAxisFinishPoint:</td><td>Absolute end position of the main axis (Pulse)</td></tr>
<tr><td>lSecondAxisFinishPoint:</td><td>Absolute end position of the second axis (Pulse)</td></tr>
<tr><td>lThirdAxisFinishPoint:</td><td>Absolute end position of the third axis (Pulse)</td></tr>
<tr><td>eInterpMode:</td><td>Command execution mode:
<table>
<tr><th>Mode</th><th>Value</th></tr>
<tr><td>General</td><td>GENERAL_INTERP_CONFIG (0)</td></tr>
</table></td></tr>
</table>

		• 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: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

5.4.3 i8196f_s_line3_move

Executes a three axes linear interpolation motion command with an S-curve velocity profile. The i8196f_s_line3_move instruction moves the axes the specified travel distance from the current position.

Syntax:

```
eRET i8196f_s_line3_move (    U8 bSlotNo,
                              eSINGLE_AXIS eMainAxis,
                              eSINGLE_AXIS eSecondAxis,
                              eSINGLE_AXIS eThirdAxis,
                              U32 dwStartSpeed,
                              U32 dwDriveSpeed,
                              U32 dwEndSpeed,
                              U32 dwAcceleration,
                              U32 dwDeceleration,
                              I32 lMainAxisRelDist,
                              I32 lSecondAxisRelDist,
                              I32 lThirdAxisRelDist,
                              eINTERPOLATION_MODE eInterpMode);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eMainAxis: *eSecondAxis:* *eThirdAxis:*	Axis definition: Axis0 AXIS_0 (0x01) Axis1 AXIS_1 (0x02) Axis2 AXIS_2 (0x04) Axis3 AXIS_3 (0x08) Axis4 AXIS_4 (0x10) Axis5 AXIS_5 (0x20)
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
lSecondAxisRelDist:	Relative distance of the second axis (Pulse) > 0: relative distance in positive direction < 0: relative distance in negative direction
lThirdAxisRelDist:	Relative distance of the third axis (Pulse)

<table>
<tr><td></td><td>> 0: relative distance in positive direction
< 0: relative distance in negative direction</td></tr>
<tr><td>eInterpMode:</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: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

5.4.4 i8196f_abs_s_line3_move

Executes a three axes absolute position interpolation motion command with an S-curve velocity profile.

Syntax:

```
eRET i8196f_abs_s_line3_move (    U8 bSlotNo,
                                  eSINGLE_AXIS eMainAxis,
                                  eSINGLE_AXIS eSecondAxis,
                                  eSINGLE_AXIS eThirdAxis,
                                  U32 dwStartSpeed,
                                  U32 dwDriveSpeed,
                                  U32 dwEndSpeed,
                                  U32 dwAcceleration,
                                  U32 dwDeceleration,
                                  I32 lMainAxisFinishPoint,
                                  I32 lSecondAxisFinishPoint,
                                  I32 lThirdAxisFinishPoint,
                                  eINTERPOLATION_MODE eInterpMode);
```

Parameters:

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>bSlotNo:</td><td>PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)</td></tr>
<tr><td>eMainAxis:

eSecondAxis:

eThirdAxis:</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>
<tr><td>Axis2</td><td>AXIS_2 (0x04)</td></tr>
<tr><td>Axis3</td><td>AXIS_3 (0x08)</td></tr>
<tr><td>Axis4</td><td>AXIS_4 (0x10)</td></tr>
<tr><td>Axis5</td><td>AXIS_5 (0x20)</td></tr>
</table></td></tr>
<tr><td>dwStartSpeed:</td><td>Magnitude of start velocity vector (PPS)</td></tr>
<tr><td>dwDriveSpeed:</td><td>Magnitude of drive velocity vector (PPS)</td></tr>
<tr><td>dwEndSpeed:</td><td>Magnitude of end velocity vector (PPS)</td></tr>
<tr><td>dwAcceleration:</td><td>Magnitude of acceleration vector (PPS/Sec)</td></tr>
<tr><td>dwDeceleration:</td><td>Magnitude of deceleration vector (PPS/Sec)</td></tr>
<tr><td>lMainAxisFinishPoint:</td><td>Absolute end position of the main axis (Pulse)</td></tr>
<tr><td>lSecondAxisFinishPoint:</td><td>Absolute end position of the second axis (Pulse)</td></tr>
<tr><td>lThirdAxisFinishPoint:</td><td>Absolute end position of the third axis (Pulse)</td></tr>
<tr><td>eInterpMode:</td><td>Command execution mode:
<table>
<tr><th>Mode</th><th>Value</th></tr>
<tr><td>General</td><td>GENERAL_INTERP_CONFIG (0)</td></tr>
</table></td></tr>
</table>

		• 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: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

5.5 Multi-Dimensional Linear Interpolation Commands

5.5.1 i8196f_lines_move

Executes a multidimensional relative position motion command. Positioning is performed on up to six axes with linear interpolation at the specified interpolation speed. The number of interpolation axes can be selected.

Syntax:

```
eRET i8196f_lines_move (    U8 bSlotNo,
                            U16 wAxes,
                            eACC_DEC_CURVE eAccDecMode,
                            U32 dwStartSpeed,
                            U32 dwDriveSpeed,
                            U32 dwEndSpeed,
                            U32 dwAcceleration,
                            U32 dwDeceleration,
                            const I32 lRelativeDistance [MAX_AXIS_NO],
                            eINTERPOLATION_MODE eInterpMode);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
wAxes:	For multiple axis select the corresponding axis bit: Axis0 – AXIS_0 (0x01) Axis1 – AXIS_1 (0x02) Axis2 – AXIS_2 (0x04) Axis3 – AXIS_3 (0x08) Axis4 – AXIS_4 (0x10) Axis5 – AXIS_5 (0x20) `wAxes= AXIS_0| AXIS_2| AXIS_5;`
eAccDecMode:	T-Curve – I8196F_ACC_DEC_T_CURVE (0x6565) S-Curve – I8196F_ACC_DEC_S_CURVE (0x6666)
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)
lRelativeDistance[]:	Relative distance of selected axes (Pulse) > 0: relative distance in positive direction

	< 0: relative distance in negative direction Array | Relative distance (Pulse) [0] | AXIS_0 relative moving distance [1] | AXIS_1 relative moving distance [2] | AXIS_2 relative moving distance [3] | AXIS_3 relative moving distance [4] | AXIS_4 relative moving distance [5] | AXIS_5 relative moving distance
eInterpMode:	Command execution mode: Mode | Value 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: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

Example:

```
eRET ret;
U16 BitMultiAxes = (0x01 | 0x02 | 0x04 | 0x10); //Axis0,1,2,4
I32 FixedPulse[6] = {10000, 20000, 20000, 0, 10000, 0};

ret = i8196f_lines_move(1, BitMultiAxes, I8196F_ACC_DEC_T_CURVE, 0,
10000, 0, 20000, 20000, FixedPulse, 0);

if (ret != I8196_SUCCESS)
{MessageBox("Lines Move Error!!");}
```

5.5.2 i8196f_abs_lines_move

Executes a multidimensional absolute position motion command. Positioning is performed on up to six axes with linear interpolation at the specified interpolation speed. The number of interpolation axes can be selected.

Syntax:

```
eRET i8196f_abs_lines_move (    U8 bSlotNo,
                                U16 wAxes,
                                eACC_DEC_CURVE eAccDecMode,
                                U32 dwStartSpeed,
                                U32 dwDriveSpeed,
                                U32 dwEndSpeed,
                                U32 dwAcceleration,
                                U32 dwDeceleration,
                                const I32 lAbsolutePosition [MAX_AXIS_NO],
                                eINTERPOLATION_MODE eInterpMode);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
wAxes:	For multiple axis select the corresponding axis bit: Axis0 — AXIS_0 (0x01) Axis1 — AXIS_1 (0x02) Axis2 — AXIS_2 (0x04) Axis3 — AXIS_3 (0x08) Axis4 — AXIS_4 (0x10) Axis5 — AXIS_5 (0x20) `wAxes= AXIS_0| AXIS_2| AXIS_5;`
eAccDecMode:	T-Curve — I8196F_ACC_DEC_T_CURVE (0x6565) S-Curve — I8196F_ACC_DEC_S_CURVE (0x6666)
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) [0] — AXIS_0 absolute position

	[1] AXIS_1 absolute position [2] AXIS_2 absolute position [3] AXIS_3 absolute position [4] AXIS_4 absolute position [5] AXIS_5 absolute position
eInterpMode:	Command execution mode: Mode: General — Value: GENERAL_INTERP_CONFIG (0) • No command buffering takes place. • A new command can only be executed if the previous command has finished. Mode: Continuous interpolation — Value: CONTI_INTERP_CONFIG (1) • Command buffering; up to 5000 command can be stored • Use this mode to generate a continuous motion path. Mode: Reserved — Value: CONTI_FIFO_BUFFER_MODE (2)

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

5.6 Two Dimensional Circular Interpolation Functions

5.6.1 i8196f_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.

Syntax:

```
eRET i8196f_t_arc2_move (   U8 bSlotNo,
                            eSINGLE_AXIS eMainAxis,
                            eSINGLE_AXIS eSlaveAxis,
                            U32 dwStartSpeed,
                            U32 dwDriveSpeed,
                            U32 dwEndSpeed,
                            U32 dwAcceleration,
                            U32 dwDeceleration,
                            eARC_DIR eArcDirection,
                            I32 lMainAxisCenterPoint,
                            I32 lSlaveAxisCenterPoint,
                            I32 lMainAxisFinishPoint,
                            I32 lSlaveAxisFinishPoint,
                            eINTERPOLATION_MODE eInterpMode);
```

Parameters:

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>bSlotNo:</td><td>PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)</td></tr>
<tr><td>eMainAxis:

eSlaveAxis:</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>
<tr><td>Axis2</td><td>AXIS_2 (0x04)</td></tr>
<tr><td>Axis3</td><td>AXIS_3 (0x08)</td></tr>
<tr><td>Axis4</td><td>AXIS_4 (0x10)</td></tr>
<tr><td>Axis5</td><td>AXIS_5 (0x20)</td></tr>
</table></td></tr>
<tr><td>dwStartSpeed:</td><td>Magnitude of start velocity vector (PPS)</td></tr>
<tr><td>dwDriveSpeed:</td><td>Magnitude of drive velocity vector (PPS)</td></tr>
<tr><td>dwEndSpeed:</td><td>Magnitude of end velocity vector (PPS)</td></tr>
<tr><td>dwAcceleration:</td><td>Magnitude of acceleration vector (PPS/Sec)</td></tr>
<tr><td>dwDeceleration:</td><td>Magnitude of deceleration vector (PPS/Sec)</td></tr>
<tr><td>eArcDirection:</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>

lMainAxisCenterPoint:	Relative center point of the main axis (Pulse)
lSlaveAxisCenterPoint:	Relative center point of the slave axis (Pulse)
lMainAxisFinishPoint:	Relative end point of the main axis (Pulse)
lSlaveAxisFinishPoint:	Relative end point of the slave axis (Pulse)
eInterpMode:	Command execution mode: Mode / Value 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: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

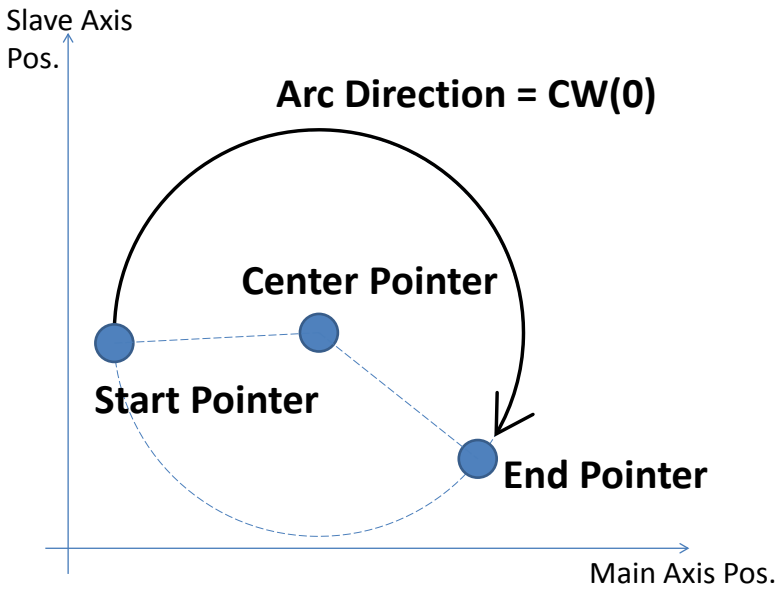


Figure 12: Clockwise circular interpolation

5.6.2 i8196f_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:

```
eRET i8196f_abs_t_arc2_move (   U8 bSlotNo,
                                eSINGLE_AXIS eMainAxis,
                                eSINGLE_AXIS eSlaveAxis,
                                U32 dwStartSpeed,
                                U32 dwDriveSpeed,
                                U32 dwEndSpeed,
                                U32 dwAcceleration,
                                U32 dwDeceleration,
                                eARC_DIR eArcDirection,
                                I32 lMainAxisCenterPoint,
                                I32 lSlaveAxisCenterPoint,
                                I32 lMainAxisFinishPoint,
                                I32 lSlaveAxisFinishPoint,
                                eINTERPOLATION_MODE eInterpMode);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eMainAxis: *eSlaveAxis:*	Axis definition: Axis: Value Axis0: AXIS_0 (0x01) Axis1: AXIS_1 (0x02) Axis2: AXIS_2 (0x04) Axis3: AXIS_3 (0x08) Axis4: AXIS_4 (0x10) Axis5: AXIS_5 (0x20)
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)
eArcDirection:	Rotation direction Direction: Value CW: ARC_DIR_CW (0) CCW: ARC_DIR_CCW (1)
lMainAxisCenterPoint:	Absolute center point of the main axis (Pulse)
lSlaveAxisCenterPoint:	Absolute center point of the slave axis (Pulse)

<table>
<tr><td>lMainAxisFinishPoint:</td><td>Absolute end point of the main axis (Pulse)</td></tr>
<tr><td>lSlaveAxisFinishPoint:</td><td>Absolute end point of the slave axis (Pulse)</td></tr>
<tr><td>eInterpMode:</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: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

5.7 Three Dimensional Helical Interpolation Functions

5.7.1 i8196f_t_helical_move

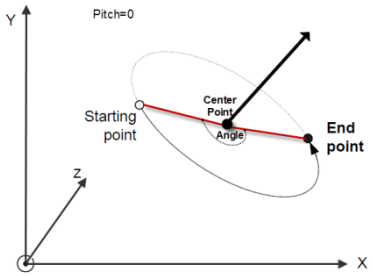
Executes a three dimensional helical (screw like) movement. The movement is a composition of a rotation by an angle about a helical axis with a translation distance along this axis. The circle center position is the relative distance from the current position.

Syntax:

```
eRET i8196f_t_helical_move (    U8 bSlotNo,
                                eSINGLE_AXIS eAxis0,
                                eSINGLE_AXIS eAxis1,
                                eSINGLE_AXIS eAxis2,
                                U32 dwStartSpeed,
                                U32 dwDriveSpeed,
                                U32 dwEndSpeed,
                                U32 dwAcceleration,
                                U32 dwDeceleration,
                                I32 lCenterPoint0,
                                I32 lCenterPoint1,
                                I32 lCenterPoint2,
                                I32 lNormalDir0,
                                I32 lNormalDir1,
                                I32 lNormalDir2,
                                F32 fAngle,
                                F32 fPitch,
                                eINTERPOLATION_MODE eInterpMode);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eAxis0: *eAxis1:* *eAxis2:*	Axis definition: Axis0 AXIS_0 (0x01) Axis1 AXIS_1 (0x02) Axis2 AXIS_2 (0x04) Axis3 AXIS_3 (0x08) Axis4 AXIS_4 (0x10) Axis5 AXIS_5 (0x20)
dwStartSpeed:	Magnitude of start velocity vector (PPS)
dwDriveSpeed:	Magnitude of drive velocity vector (PPS)
dwEndSpeed:	Magnitude of end velocity vector (PPS)

<table>
<tr><td>dwAcceleration:</td><td>Magnitude of acceleration vector (PPS/Sec)</td></tr>
<tr><td>dwDeceleration:</td><td>Magnitude of deceleration vector (PPS/Sec)</td></tr>
<tr><td>lCenterPoint0:</td><td>Relative center point of eAxis0 (Pulse)</td></tr>
<tr><td>lCenterPoint1:</td><td>Relative center point of eAxis1 (Pulse)</td></tr>
<tr><td>lCenterPoint2:</td><td>Relative center point of eAxis2 (Pulse)</td></tr>
<tr><td>lNormalDir0:
lNormalDir1:
lNormalDir2:</td><td>The parameters describe a three dimensional rotation vector (helical axis).
The rotation vector determines:
The positive direction of the translational movement (pitch direction).
The positive direction of rotation. The positive direction of rotation is being determined by the right-hand rule: the right thumb points along the positive direction of the rotation axis and the curl of your fingers represents the direction of rotation  </td></tr>
<tr><td>fAngle:</td><td>Rotation angle in degrees (360 indicates one full revolution, 720 will result in two full revolution, etc.)
> 0: positive direction rotation vector (right-hand rule)
< 0: negative direction of the rotation vector</td></tr>
<tr><td>fPitch:</td><td>The relative distance to move along the helical axis after each full revolution.
If this parameter is zero a three dimensional arc will be executed 
</td></tr>
<tr><td>eInterpMode:</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>

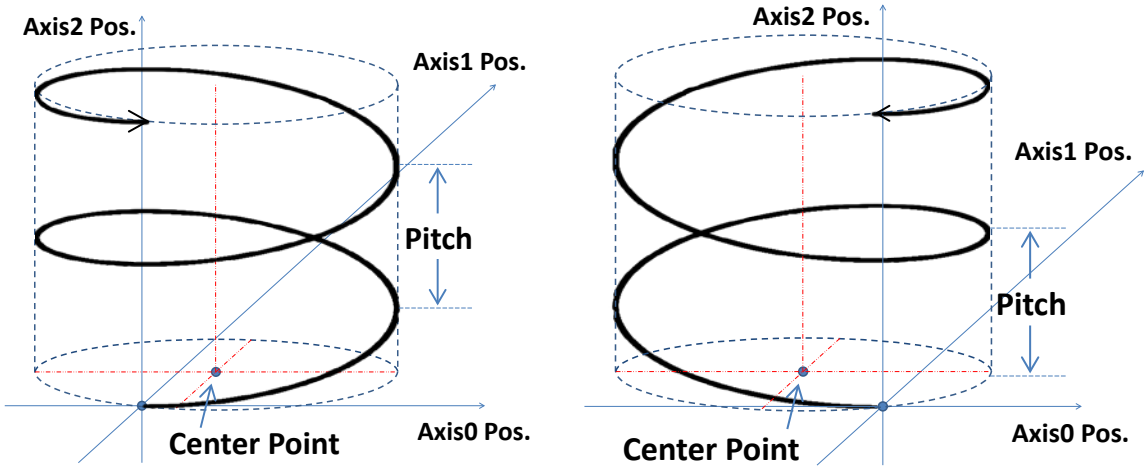
Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

Example:

The Figure 13 shows a helical path which helical direction is perpendicular to the Axis0 - Axis1 plane:
(lCenterPoint2 = 0, lNormalDir0 = 0, lNormalDir1 = 0, lNormalDir2 = 1)



(a) Angle > 0 (b) Angle < 0

Figure 13: Helical axis is perpendicular to the Axis0 and Axis1 plane

5.7.2 i8196f_abs_t_helical_move

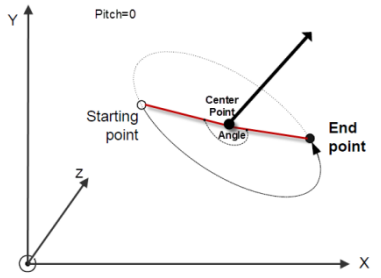
Executes a three dimensional helical (screw like) movement. The movement is a composition of a rotation by an angle about a helical axis with a translation distance along this axis. The circle center position is an absolute position.

Syntax:

```
eRET i8196f_abs_t_helical_move (    U8 bSlotNo,
                                    eSINGLE_AXIS eAxis0,
                                    eSINGLE_AXIS eAxis1,
                                    eSINGLE_AXIS eAxis2,
                                    U32 dwStartSpeed,
                                    U32 dwDriveSpeed,
                                    U32 dwEndSpeed,
                                    U32 dwAcceleration,
                                    U32 dwDeceleration,
                                    I32 lCenterPoint0,
                                    I32 lCenterPoint1,
                                    I32 lCenterPoint2,
                                    I32 lNormalDir0,
                                    I32 lNormalDir1,
                                    I32 lNormalDir2,
                                    F32 fAngle,
                                    F32 fPitch,
                                    eINTERPOLATION_MODE eInterpMode);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eAxis0: *eAxis1:* *eAxis2:*	Axis definition: Axis0 AXIS_0 (0x01) Axis1 AXIS_1 (0x02) Axis2 AXIS_2 (0x04) Axis3 AXIS_3 (0x08) Axis4 AXIS_4 (0x10) Axis5 AXIS_5 (0x20)
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)
lCenterPoint0:	Relative center point of eAxis0 (Pulse)

<table>
<tr><td>lCenterPoint1:</td><td>Absolute center point of eAxis1 (Pulse)</td></tr>
<tr><td>lCenterPoint2:</td><td>Absolute center point of eAxis2 (Pulse)</td></tr>
<tr><td>lNormalDir0:
lNormalDir1:
lNormalDir2:</td><td>The parameters describe a three dimensional rotation vector.
The rotation vector determines:
The positive direction of the translational movement (pitch direction).
The positive direction of rotation. The positive direction of rotation is being determined by the right-hand rule: the right thumb points along the positive direction of the rotation axis and the curl of your fingers represents the direction of rotation  </td></tr>
<tr><td>fAngle:</td><td>Rotation angle in degrees (360 indicates one full revolution, 720 will result in two full revolution, etc.)
> 0: positive direction rotation vector (right-hand rule)
< 0: negative direction of the rotation vector</td></tr>
<tr><td>fPitch:</td><td>The relative distance to move along the helical axis after each full revolution.
If this parameter is zero a three dimensional arc will be executed 
</td></tr>
<tr><td>eInterpMode:</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: I8196_SUCCESS
- Others: Error (refer to error documentation)

5.8 Continuous Interpolation Functions

5.8.1 i8196f_set_conti_interp_cfg

Assigns axes to an interpolation group and sets the axes 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:

```
eRET i8196f_set_conti_interp_cfg (   U8 bSlotNo,
                                     eINTERP_CONFIG eCfgEnable,
                                     U16 wGroupIndex0,
                                     U16 wGroupIndex1,
                                     U16 wGroupIndex2);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eCfgEnable:	Enable continuous interpolation: Mode: Disable — CANCEL_INTERP_CONFIG (0) Enable — START_INTERP_CONFIG (1)
wGroupIndex0:	Select the axis which belongs to the first interpolation group Axis0 — AXIS_0 (0x01) Axis1 — AXIS_1 (0x02) Axis2 — AXIS_2 (0x04) Axis3 — AXIS_3 (0x08) Axis4 — AXIS_4 (0x10) Axis5 — AXIS_5 (0x20) Example: `wGroupIndex0 = AXIS_0 | AXIS_1;`
wGroupIndex1:	Select the axis which belongs to the second interpolation group Example: `wGroupIndex1 = AXIS_3 | AXIS_5;`
wGroupIndex2:	Select the axis which belongs to the third interpolation group Example: `wGroupIndex2 = AXIS_2 | AXIS_4;`

Return:

- 0: I8196_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 "`i8196f_drv_hold`" to hold the next command from being executed
  2. Fill the command buffer with commands
  3. Call "`i8196f_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.

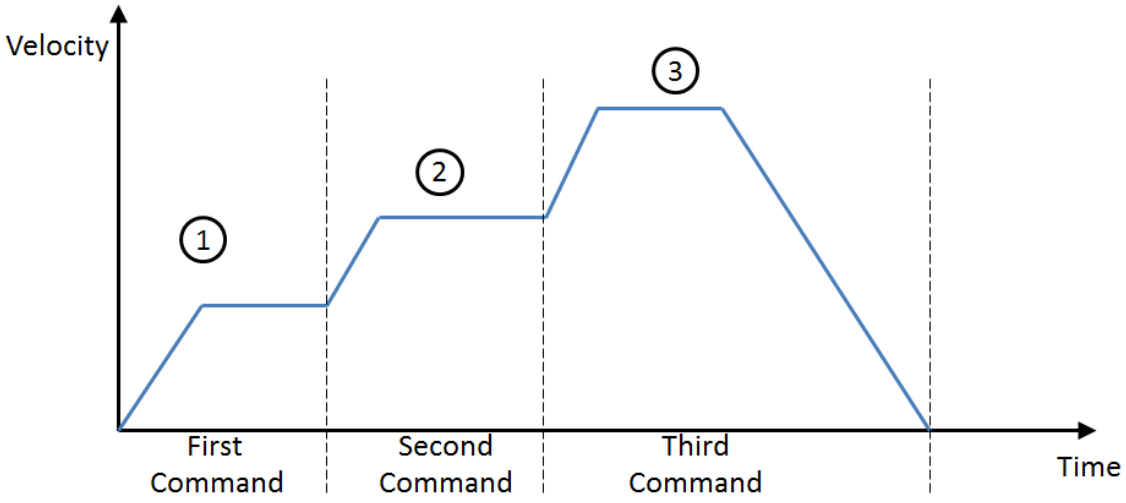


Figure 14: Velocity profile supported in continuous interpolation mode

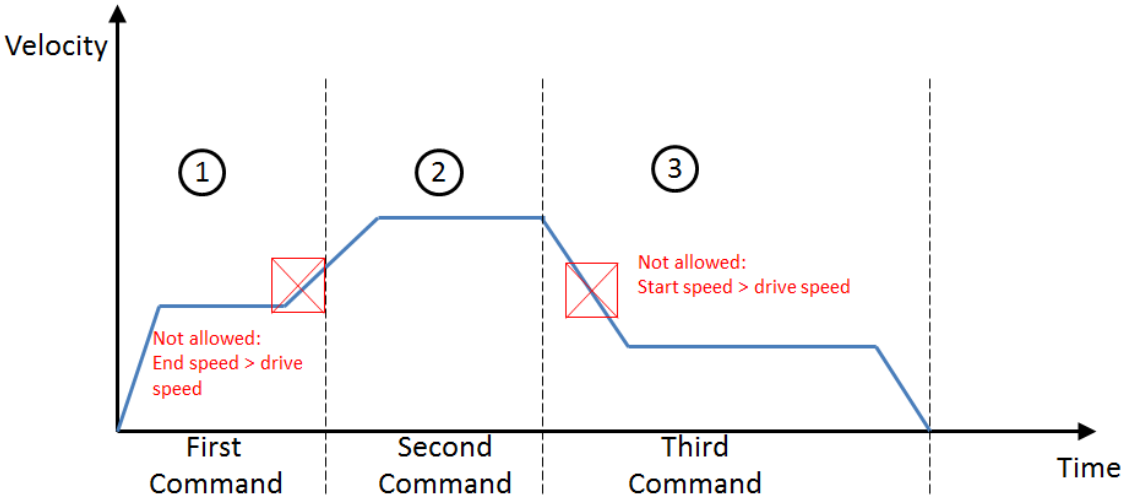


Figure 15: Velocity profiles which are not supported

Example:

Example 1:

```
//================================
// Three Dimensional Continue Interpolation
//================================
eRET eRet;
U32 dwStartSpeed, dwDriveSpeed, dwEndSpeed, dwAcceleration,
dwDeceleration;
U16 wGroupIndex0;
U8 bDone = 0;
U16 wStopStatus;
I32 x, y, z;
U8 bSlot = 1;

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

//Assign three axis to a continuous interpolation group:
wGroupIndex0 = AXIS_0|AXIS_1|AXIS_2;
eRet = i8196f_set_conti_interp_cfg( bSlot, START_INTERP_CONFIG,
wGroupIndex0, 0, 0 );

//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;
    z *= 100;
    eRet = i8196f_abs_t_line3_move( bSlot, AXIS_0, AXIS_1, AXIS_2,
            dwStartSpeed, dwDriveSpeed, dwEndSpeed, dwAcceleration,
            dwDeceleration, x, y, z, CONTI_INTERP_CONFIG);
}

//Wait until the continuous interpolation command have been executed:
while( bDone != 1)
{
    eRet = i8196f_get_motion_done( bSlot, AXIS_0, &bDone,
&wStopStatus);
    ::Sleep(10);
}

//Disable the continuous interpolation mode:
eRet = i8196f_set_conti_interp_cfg( bSlot, CANCEL_INTERP_CONFIG,
wGroupIndex0, 0, 0 );
```

Example 2:

```
//================================
// Three Dimensional Continue Interpolation
//================================
eRET eRet;
U32 dwStartSpeed, dwDriveSpeed, dwEndSpeed, dwAcceleration,
dwDeceleration;
U16 wGroupIndex0;
U8 bDone = 0;
U16 wStopStatus;
I32 x, y, z;
U8 bSlot = 1;

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

//Create two continuous interpolation groups:
wGroupIndex0 = AXIS_0|AXIS_1|AXIS_2;
wGroupIndex1 = AXIS_3|AXIS_4|AXIS_5;
eRet = i8196f_set_conti_interp_cfg( bSlot, START_INTERP_CONFIG,
wGroupIndex0, wGroupIndex1, 0 );

//Put the groups into hold mode
eRet = i8196f_drv_hold( bSlot,
AXIS_0|AXIS_1|AXIS_2|AXIS_3|AXIS_4|AXIS_5 );

//Write 120 interpolation commands to the buffer for execution
for(int i=0; i<120; i++) //120
{
    x *= i;
    y *= x;
    z *= 100;
      eRet = i8196f_abs_t_line3_move( bSlot, AXIS_0, AXIS_1, AXIS_2,
            dwStartSpeed, dwDriveSpeed, dwEndSpeed, dwAcceleration,
            dwDeceleration, x, y, z, CONTI_INTERP_CONFIG);

      eRet = i8196f_abs_t_line3_move( bSlot, AXIS_3, AXIS_4, AXIS_5,
        dwStartSpeed, dwDriveSpeed, dwEndSpeed, dwAcceleration,
        dwDeceleration, x, y, z, CONTI_INTERP_CONFIG);
}

//Start the execution of the commands in the buffer:
eRet = i8196f_drv_start( bSlot,
AXIS_0|AXIS_1|AXIS_2|AXIS_3|AXIS_4|AXIS_5);

//Wait for the first continuous interpolation group to finish:
while( bDone != 1)
{
    eRet = i8196f_get_motion_done( bSlot, AXIS_0, &bDone,
```

```
&wStopStatus);
    ::Sleep(10);
}
//Wait for the second continuous interpolation group to finish:
while( bDone != 1)
{
    eRet = i8196f_get_motion_done( bSlot, AXIS_3, &bDone,
&wStopStatus);
    ::Sleep(10);
}

//Disable the continuous interpolation mode:
eRet = i8196f_set_conti_interp_cfg( bSlot, CANCEL_INTERP_CONFIG,
wGroupIndex0, wGroupIndex1, 0 );
```

5.9 Motion Stop Functions

5.9.1 i8196f_stop_move

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

Syntax:

```
eRET i8196f_stop_move (    U8 bSlotNo,
                           eSINGLE_AXIS eSingleAxis,
                           eSTOP_MODE eStopMode);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eSingleAxis:	Axis definition: Axis0 – AXIS_0 (0x01) Axis1 – AXIS_1 (0x02) Axis2 – AXIS_2 (0x04) Axis3 – AXIS_3 (0x08) Axis4 – AXIS_4 (0x10) Axis5 – AXIS_5 (0x20)
eStopMode:	Stop mode Deceleration stop – STOP_SLOWDOWN (1) Sudden stop – STOP_SUDDEN (2)

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

5.9.2 i8196f_set_softlimit

Sets the software limits for the positive and negative direction. Once a software limit position is specified, the i8196F will not accept position commands beyond the limit and motion will stop once the limit is hit.

Syntax:

```
eRET i8196f_set_softlimit (    U8 bSlotNo,
                               eSINGLE_AXIS eSingleAxis,
                               eSTOP_MODE eStopMode,
                               eFEEDBACK_SRC eFeedbackSrc,
                               I32 lLimitPositive,
                               I32 lLimitNegative);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eSingleAxis:	Axis definition: Axis0 AXIS_0 (0x01) Axis1 AXIS_1 (0x02) Axis2 AXIS_2 (0x04) Axis3 AXIS_3 (0x08) Axis4 AXIS_4 (0x10) Axis5 AXIS_5 (0x20)
eStopMode:	Stop mode Deceleration stop: STOP_SLOWDOWN (1) Sudden stop: STOP_SUDDEN (2)
eFeedbackSrc:	Position counter source Encoder Pulse Counter: FEEDBACK_SRC_ENC (1) Commanded Position Pulse Counter: FEEDBACK_SRC_DDA (2)
lLimitPositive:	Positive direction soft limit
lLimitNegative:	Negative direction soft limit

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

5.9.3 i8196f_set_softlimit_disable

Disables the axis limits settings.

Syntax:

```
eRET i8196f_set_softlimit_disable (   U8 bSlotNo,
                                      eSINGLE_AXIS eSingleAxis);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eSingleAxis:	Axis definition: Axis0 — AXIS_0 (0x01) Axis1 — AXIS_1 (0x02) Axis2 — AXIS_2 (0x04) Axis3 — AXIS_3 (0x08) Axis4 — AXIS_4 (0x10) Axis5 — AXIS_5 (0x20)

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

5.10 Multi-Axis Hold/Release Functions

5.10.1 i8196f_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 "i8196f_drv_start()" releases the hold operation.

Syntax:

```
eRET i8196f_drv_hold (    U8 bSlotNo,
                          U16 wBitMultiAxes);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
wBitMultiAxes:	Each bit in the variable represents an axis:

Axis	Value
Axis0	AXIS_0 (0x01)
Axis1	AXIS_1 (0x02)
Axis2	AXIS_2 (0x04)
Axis3	AXIS_3 (0x08)
Axis4	AXIS_4 (0x10)
Axis5	AXIS_5 (0x20)

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

- After calling "i8196f_drv_hold()" the current running interpolation command will finish first, but the next command will not start until "i8196f_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.

5.10.2 i8196f_drv_start

Terminates the hold operation. Axes which have been put on hold by "i8196f_drv_hold()" will continue to execute the next motion command stored in the command FIFO buffer.

Syntax:

```
eRET i8196f_drv_start (    U8 bSlotNo,
                           U16 wBitMultiAxes);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
wBitMultiAxes:	Each bit of the variable represents an axis: Axis0 — AXIS_0 (0x01) Axis1 — AXIS_1 (0x02) Axis2 — AXIS_2 (0x04) Axis3 — AXIS_3 (0x08) Axis4 — AXIS_4 (0x10) Axis5 — AXIS_5 (0x20)

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

6 Other Functions

6.1 Compare Function

6.1.1 i8196f_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:

```
eRET i8196f_set_compare_trig_cfg (   U8 bSlotNo,
                                     eSINGLE_AXIS eSingleAxis,
                                     eENABLE_DISABLE eCmpTrigEnable,
                                     eLOGIC_ACTIVE_LEVEL eOutputLogic,
                                     eTRIG_PULSE_WIDTH  ePulseWidth,
                                     eCMP_TRIG_MOVE_DIR eMoveDirection,
                                     eENABLE_DISABLE eCmpIncEnable,
                                     U16 wConstPitch,
                                     I32 lCmpData);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eSingleAxis:	Axis definition: Axis0 — AXIS_0 (0x01) Axis1 — AXIS_1 (0x02) Axis2 — AXIS_2 (0x04) Axis3 — AXIS_3 (0x08) Axis4 — AXIS_4 (0x10) Axis5 — AXIS_5 (0x20)
eCmpTrigEnable:	Enable compare function Disable — DISABLE (0x00) Enable — ENABLE (0x01)
eOutputLogic:	Compare (CMP) active level Active low — LOGIC_ACTIVE_LOW (0x00)

<table>
<tr><td></td><td><table><tr><td>Active high</td><td>LOGIC_ACTIVE_HIGH (0x01)</td></tr></table></td></tr>
<tr><td>ePulseWidth:</td><td>Pulse width trigger signals (see remarks)</td></tr>
<tr><td>eMoveDirection:</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>eCmpIncEnable:</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>wConstPitch:</td><td>The auto-increment distance; The distance between two compare signal (pulse)
(Only valid if “eCmpIncEnable” is enabled otherwise this parameter will be ignored)</td></tr>
<tr><td>lCmpData:</td><td>The first position at which the compare function will trigger an output signal.
(If “eCmpIncEnable” is disabled then one output signal will be triggered at the “lCmpData” position)</td></tr>
</table>

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

- Compare function cannot be used when axis is in Vring counter mode (see “i8196f_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

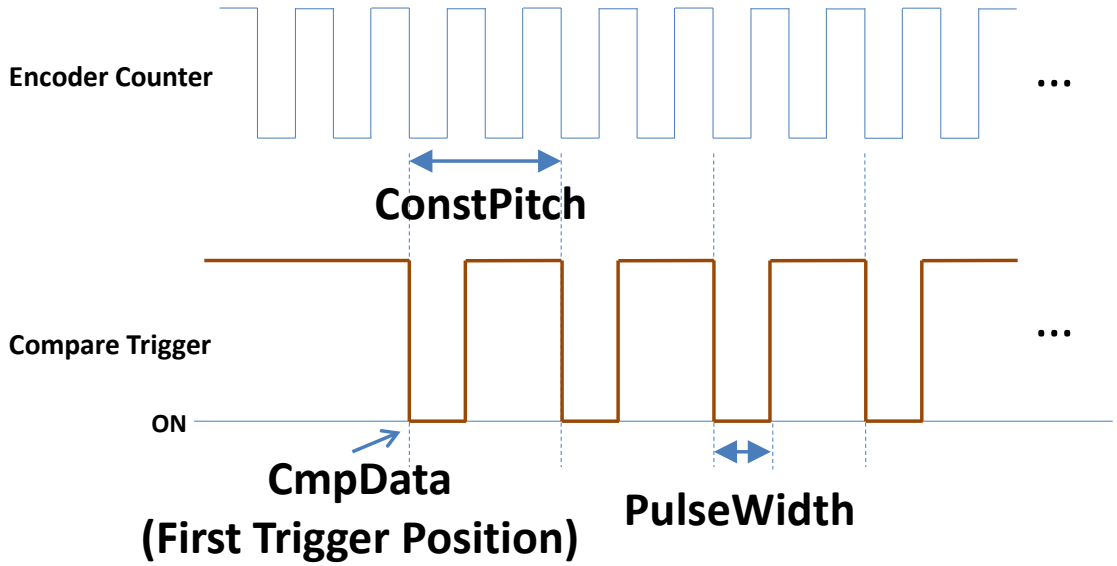


Figure 16: Continuous equidistant spaced trigger output function

6.2 Latch Function

6.2.1 i8196f_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:

```
eRET i8196f_set_latch_cfg (     U8 bSlotNo,
                                eSINGLE_AXIS eSingleAxis,
                                eENABLE_DISABLE eLatchEnable,
                                eLOGIC_ACTIVE_LEVEL eLatchLogic);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eSingleAxis:	Axis definition: Axis0 AXIS_0 (0x01) Axis1 AXIS_1 (0x02) Axis2 AXIS_2 (0x04) Axis3 AXIS_3 (0x08) Axis4 AXIS_4 (0x10) Axis5 AXIS_5 (0x20)
eLatchEnable:	Enable/Disable the latch function treat LTC PIN as a general input: DISABLE (0x00) treat LTC PIN as a dedicated external trigger to latch input: ENABLE (0x01)
eLatchLogic:	Latch (LTC) active level Active low: LOGIC_ACTIVE_LOW (0x00) Active high: LOGIC_ACTIVE_HIGH (0x01)

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

- Only when Latch DI activates will the current encoder position be latched.

6.2.2 i8196f_get_latch

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

Syntax:

```
eRET i8196f_get_latch (     U8 bSlotNo,
                            eSINGLE_AXIS eSingleAxis,
                            I32* plLatchData);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eSingleAxis:	Axis definition: Axis0 AXIS_0 (0x01) Axis1 AXIS_1 (0x02) Axis2 AXIS_2 (0x04) Axis3 AXIS_3 (0x08) Axis4 AXIS_4 (0x10) Axis5 AXIS_5 (0x20)
plLatchData:	Pointer to the value of the encoder position counter latch

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

Only when Latch DI activates will the current encoder position be latched.

7 General Purpose IO and Remote FRnet IO Functions

7.1 General Purpose IO

7.1.1 i8196f_set_general_do

Sets the output state of one general purpose digital output channel.

Syntax:

```
eRET i8196f_set_general_do (    U8 bSlotNo,
                                eGENERAL_DO eDoChannelNo,
                                eCHANNEL_STATE eOnOff);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eDoChannelNo:	DO channel number: Channel: Value DO Channel 0: DO_GDO0_CH (0) DO Channel 1: DO_GDO1_CH (1) DO Channel 2: DO_GDO2_CH (2)
eOnOff:	DO State: State: Value OFF: OFF (0x00) ON: ON (0x01)

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

7.1.2 i8196f_get_general_do

Reads the current output state of the general purpose DO.

Syntax:

```
eRET i8196f_get_general_do (    U8 bSlotNo,
                                eGENERAL_DO eDoChannelNo,
                                eCHANNEL_STATE *peOnOff);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eDoChannelNo:	DO channel number: Channel: Value DO Channel 0: DO_GDO0_CH (0) DO Channel 1: DO_GDO1_CH (1) DO Channel 2: DO_GDO2_CH (2)
peOnOff:	Pointer to the current DO output state: State: Value OFF: OFF (0x00) ON: ON (0x01)

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

7.1.3 i8196f_set_all_general_do

Sets all the general-purpose output signal state.

Syntax:

```
eRET i8196f_set_all_general_do (    U8 bSlotNo,
                                    U16 wMultiGdoStatus);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
wMultiGdoStatus:	Each bit represents a DO channel number. Bit / Channel No Bit 0 / DO Channel 0 Bit 1 / DO Channel 1 Bit 2 / DO Channel 2 Bit 3~15 / reserved If the bit is 0: indicates that the corresponding signal is OFF If the bit is 1: indicates that the corresponding signal is ON

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

7.1.4 i8196f_get_all_general_do

Reads the current output state of the general purpose DO.

Syntax:

```
eRET i8196f_get_all_general_do (    U8 bSlotNo,
                                    U16* pwMultiGdoStatus);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
pwMultiGdoStatus:	Pointer to 16 bit variable. Each bit represents a DO channel number. Bit 0 – DO Channel 0 Bit 1 – DO Channel 1 Bit 2 – DO Channel 2 Bit 3~15 – reserved If the bit is 0: indicates that the corresponding signal is OFF If the bit is 1: indicates that the corresponding signal is ON

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

7.1.5 i8196f_get_general_di

Gets the input state of the general purpose DI channel.

Syntax:

```
eRET i8196f_get_general_di (    U8 bSlotNo,
                                eGENERAL_DI eDiChannelNo,
                                eCHANNEL_STATE *peOnOff);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eDiChannelNo:	DI channel number: Channel / Value DI Channel 0 — DI_GDI0_CH (0) DI Channel 1 — DI_GDI1_CH (1) DI Channel 2 — DI_GDI2_CH (2) DI Channel 3 — DI_GDI3_CH (3) DI Channel 4 — DI_GDI4_CH (4) DI Channel 5 — DI_GDI5_CH (5) DI Channel 6 — DI_GDI6_CH (6) DI Channel 7 — DI_GDI7_CH (7) DI Channel 8 — DI_GDI8_CH (8) DI Channel 9 — DI_GDI9_CH (9) DI Channel 10 — DI_GDI10_CH (10) DI Channel 11 — DI_GDI11_CH (11)
peOnOff:	Pointer to a DI state: State / Value OFF — OFF (0x00) ON — ON (0x01)

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

7.1.6 i8196f_get_all_general_di

Gets the input state of all general purpose DIs.

Syntax:

```
eRET i8196f_get_all_general_di (    U8 bSlotNo,
                                    U16* pwMultiGdiStatus);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
pwMultiGdiStatus:	Pointer to 16 bit variable. Each bit represents a DI channel. Bit / Channel No Bit 0 / DI Channel 0 Bit 1 / DI Channel 1 … / … Bit 10 / DI Channel 10 Bit 11 / DI Channel 11 Bit 12~15 / reserved If the bit is 0: indicates that the corresponding signal is OFF If the bit is 1: indicates that the corresponding signal is ON

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

7.2 Remote FRnet IO

7.2.1 i8196f_set_frnet_group_do

Sets the output state of a remote FRnet DO module. Each FRnet DO module has 16 DO channels.

Syntax:

```
eRET i8196f_set_frnet_group_do (    U8 bSlotNo,
                                    eFRNET_DO_ADDR eRA,
                                    U16 wDOData);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eRA:	Address of the remote FRnet DO module: Address / Value DO Group 0 — FRNET_RA0 (0) DO Group 1 — FRNET_RA1 (1) DO Group 2 — FRNET_RA2 (2) DO Group 3 — FRNET_RA3 (3) DO Group 4 — FRNET_RA4 (4) DO Group 5 — FRNET_RA5 (5) DO Group 6 — FRNET_RA6 (6) DO Group 7 — FRNET_RA7 (7)
wDOData:	A 16 bit parameter where each bit represents the output state of a DO channel. Bit 0 — CH_0 Bit 1 — CH_1 … — … Bit 14 — CH_14 Bit 15 — CH_15 If the bit is 0: indicates that the corresponding signal is OFF If the bit is 1: indicates that the corresponding signal is ON

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

7.2.2 i8196f_get_frnet_group_do

Gets the output state of a remote FRnet DO module. Each FRnet DO module has 16 DO channels.

Syntax:

```
eRET i8196f_get_frnet_group_do (    U8 bSlotNo,
                                    eFRNET_DO_ADDR eRA,
                                    U16 *pwStatus);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eRA:	Address of the remote FRnet DO module: Address / Value DO Group 0 — FRNET_RA0 (0) DO Group 1 — FRNET_RA1 (1) DO Group 2 — FRNET_RA2 (2) DO Group 3 — FRNET_RA3 (3) DO Group 4 — FRNET_RA4 (4) DO Group 5 — FRNET_RA5 (5) DO Group 6 — FRNET_RA6 (6) DO Group 7 — FRNET_RA7 (7)
pwStatus:	A pointer to a 16 bit parameter. Each bit represents the output state of a DO channel. Bit / Channel Number Bit 0 — CH_0 Bit 1 — CH_1 … — … Bit 14 — CH_14 Bit 15 — CH_15 If the bit is 0: indicates that the corresponding signal is OFF If the bit is 1: indicates that the corresponding signal is ON

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

7.2.3 i8196f_scan_frnet_di

Scans the FRnet network for remote FRnet DI modules and returns which DI modules are connected and active. A maximum number of 8 FRnet DI modules can be used in a FRnet network.

Syntax:

```
eRET i8196f_scan_frnet_di (    U8 bSlotNo,
                               U16 *pwDIModules);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
pwDIModules*:*	Pointer to a16 bit variable. Each bit represents an FRnet DI module. A set bit indicates that the remote FRnet DI module is connected and active.

Bit	DI Module Address
Bit 0	Group 8
Bit 1	Group 9
Bit 2	Group 10
Bit 3	Group 11
Bit 4	Group 12
Bit 5	Group 13
Bit 6	Group 14
Bit 7	Group 15
Bit 8 ~ 15	reserved

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

7.2.4 i8196f_get_frnet_di

Gets the channel state of a remote FRnet DI module. Each FRnet DI module has 16 DI channels.

Syntax:

```
eRET i8196f_get_frnet_di (    U8 bSlotNo,
                              eFRNET_DI_ADDR eSA,
                              U8 bDiChannelNo,
                              eCHANNEL_STATE *peOnOff);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eSA:	Address of the remote FRnet DI module: Module / Value DI Group 8 — FRNET_SA8 (8) DI Group 9 — FRNET_SA9 (9) DI Group 10 — FRNET_SA10 (10) DI Group 11 — FRNET_SA11 (11) DI Group 12 — FRNET_SA12 (12) DI Group 13 — FRNET_SA13 (13) DI Group 14 — FRNET_SA14 (14) DI Group 15 — FRNET_SA15 (15)
bDiChannelNo:	DI channel number (0 ~ 15)
peOnOff:	Pointer to a channel DI state: State / Value OFF — OFF (0x00) ON — ON (0x01)

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

7.2.5 i8196f_get_frnet_group_di

Reads 16 channel states of a remote FRnet DI module. Each FRnet DI module has 16 DI channels.

Syntax:

```
eRET i8196f_get_frnet_group_di (    U8 bSlotNo,
                                    eFRNET_DI_ADDR eSA,
                                    U16 *pwStatus);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
eSA:	Address of the remote FRnet DI module: Address / Value DI Group 8 — FRNET_SA8 (8) DI Group 9 — FRNET_SA9 (9) DI Group 10 — FRNET_SA10 (10) DI Group 11 — FRNET_SA11 (11) DI Group 12 — FRNET_SA12 (12) DI Group 13 — FRNET_SA13 (13) DI Group 14 — FRNET_SA14 (14) DI Group 15 — FRNET_SA15 (15)
pwStatus:	A pointer to a 16 bit parameter. Each bit represents the input state of a DI channel. Bit 0 — CH_0 Bit 1 — CH_1 … — … Bit 14 — CH_14 Bit 15 — CH_15 If the bit is 0: indicates that the corresponding signal is OFF If the bit is 1: indicates that the corresponding signal is ON

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

8 Hardware and Software Version Functions

8.1 Hardware Version

8.1.1 i8196f_get_card_version

Gets the PCB and PLD version.

Syntax:

```
eRET i8196f_get_card_version (    U8 bSlotNo,
                                  U16* pwPCBVersion,
                                  U16* pwPLDVersion);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
pwPCBVersion:	Pointer to PCB version
pwPLDVersion:	Pointer to PLD version

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

8.1.2 i8196f_get_fpga_version

Gets the FPGA version.

Syntax:

```
eRET i8196f_get_fpga_version (    U8 bSlotNo,
                                  U16* pwFpgaVers);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
pwFpgaVers:	Pointer to FPGA version

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

8.2 Software Version

8.2.1 i8196f_get_dsp_firmware_version

Gets the current DSP firmware version.

Syntax:

```
eRET i8196f_get_dsp_firmware_version (    U8 bSlotNo,
                                          U16* pwDspFirmwareVers);
```

Parameters:

Name	Description
bSlotNo:	PAC slot number (WinPAC and ViewPAC: 0, 1, 2, …; XPAC: 1, 2, 3, …)
pwDspFirmwareVers:	Pointer to DSP firmware version

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

8.2.2 i8196f_get_dll_version

Gets the DLL version.

Syntax:

```
eRET i8196f_get_dll_version (    U16* pwDllVers  );
```

Parameters:

Name	Description
pwDllVers:	Pointer to DLL version

Return:

- 0: I8196_SUCCESS
- Others: Error (refer to error documentation)

Remarks:

9 Appendix

9.1 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

9.2 Function Error Code

Error Code	Name	Description
0	I8196_SUCCESS	Function executed successfully.
1	I8196_ERROR	Reserved
2	ERR_CARD_NOT_FOUND	No I-8196F card detected in the specified PAC slot.
3	ERR_INVALID_SLOT_NO	Specified slot is not being support by the PAC Remember: • For WinPAC and ViewPAC the number of the first slot is 0 • For XPAC the number of the first slot is 1
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 in 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 I-8196F card has not been registered and initialized yet. It is required to first call "i8196f_registration()" 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 “i8196f_set_conti_interp_cfg()" to set the axes in continuous interpolation mode.
24	ERR_INVALID_HELICAL_MODE	reserved
25	ERR_INVALID_FRNET_SPEED_SETTING	Invalid value has been passed to the FRnet speed parameter. Only the following FRnet speeds are being supported: • FRNET_SPEED_1M • FRNET_SPEED_250K
26	ERR_INVALID_FRNET_SA_GROUP_ADDR	Invalid address has been passed to the FRnet DI module address parameter.
27	ERR_INVALID_FRNET_RA_GROUP_ADDR	Invalid value has been passed to the FRnet DO module address parameter.
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 "i8196f_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 "i8196f_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, can't use ring counter mode.
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.

		“i8196f_set_compare_trig_cfg()”)
51	ERR_DLL_DO_NOT_SUPPORT_MULTI_APP	More than one application access the I-8196F. Only one application is allowed to access the I-8196F 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