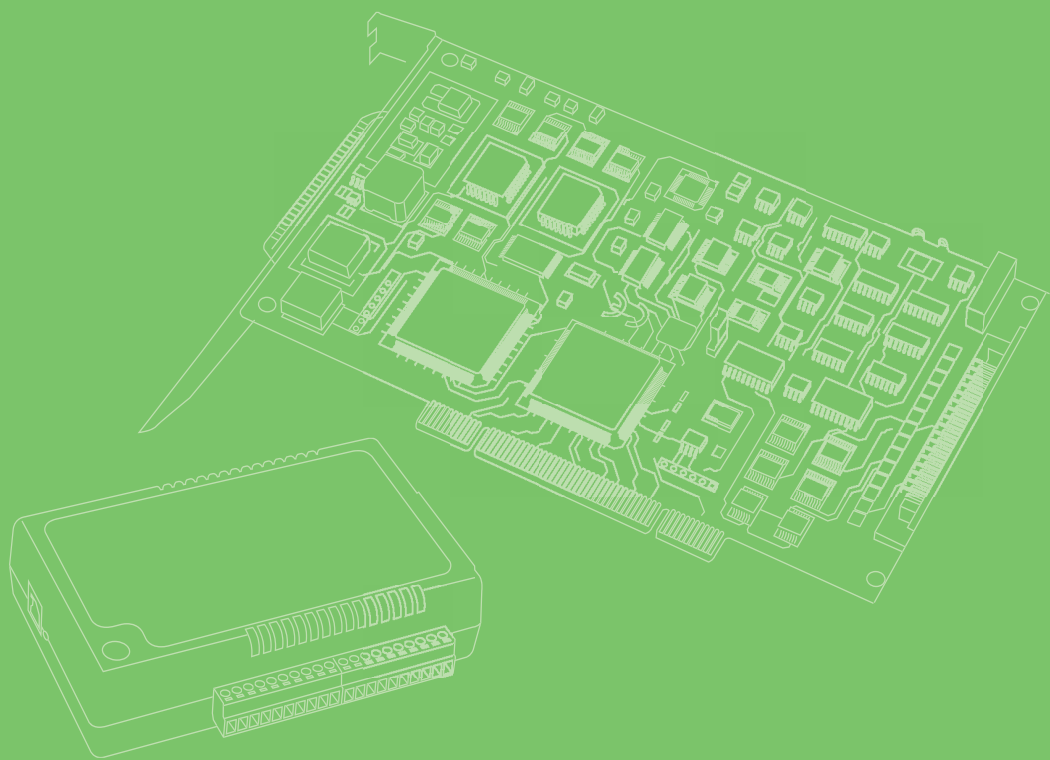


User Manual



PCI-1203

EtherCAT Master PCI Card

ADVANTECH

Enabling an Intelligent Planet

Copyright

The documentation and the software included with this product are copyrighted 2015 by Advantech Co., Ltd. All rights are reserved. Advantech Co., Ltd. reserves the right to make improvements in the products described in this manual at any time without notice. No part of this manual may be reproduced, copied, translated or transmitted in any form or by any means without the prior written permission of Advantech Co., Ltd. Information provided in this manual is intended to be accurate and reliable. However, Advantech Co., Ltd. assumes no responsibility for its use, nor for any infringements of the rights of third parties, which may result from its use.

Acknowledgements

IBM and PC are trademarks of International Business Machines Corporation.

All other product names or trademarks are properties of their respective owners.

Product Warranty (2 years)

Advantech warrants to you, the original purchaser, that each of its products will be free from defects in materials and workmanship for two years from the date of purchase.

This warranty does not apply to any products which have been repaired or altered by persons other than repair personnel authorized by Advantech, or which have been subject to misuse, abuse, accident or improper installation. Advantech assumes no liability under the terms of this warranty as a consequence of such events.

Because of Advantech's high quality-control standards and rigorous testing, most of our customers never need to use our repair service. If an Advantech product is defective, it will be repaired or replaced at no charge during the warranty period. For out-of-warranty repairs, you will be billed according to the cost of replacement materials, service time and freight. Please consult your dealer for more details.

If you think you have a defective product, follow these steps:

1. Collect all the information about the problem encountered. (For example, CPU speed, Advantech products used, other hardware and software used, etc.) Note anything abnormal and list any onscreen messages you get when the problem occurs.
2. Call your dealer and describe the problem. Please have your manual, product, and any helpful information readily available.
3. If your product is diagnosed as defective, obtain an RMA (return merchandise authorization) number from your dealer. This allows us to process your return more quickly.
4. Carefully pack the defective product, a fully-completed Repair and Replacement Order Card and a photocopy proof of purchase date (such as your sales receipt) in a shippable container. A product returned without proof of the purchase date is not eligible for warranty service.
5. Write the RMA number visibly on the outside of the package and ship it prepaid to your dealer.

Declaration of Conformity

CE

This product has passed the CE test for environmental specifications when shielded cables are used for external wiring. We recommend the use of shielded cables. This kind of cable is available from Advantech. Please contact your local supplier for ordering information.

FCC Class A

Note: This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference in which case the user will be required to correct the interference at his own expense.

FM

This equipment has passed the FM certification. According to the National Fire Protection Association, work sites are classified into different classes, divisions and groups, based on hazard considerations. This equipment is compliant with the specifications of Class I, Division 2, Groups A, B, C and D indoor hazards.

Technical Support and Assistance

1. Visit the Advantech web site at www.advantech.com/support where you can find the latest information about the product.
2. Contact your distributor, sales representative, or Advantech's customer service center for technical support if you need additional assistance. Please have the following information ready before you call:
 - Product name and serial number
 - Description of your peripheral attachments
 - Description of your software (OS, version, application software, etc.)
 - A complete description of the problem
 - The exact wording of any error messages

Safety Precaution - Static Electricity

Follow these simple precautions to protect yourself from harm and the products from damage.

- To avoid electrical shock, always disconnect the power from your PC chassis before you work on it. Don't touch any components on the CPU card or other cards while the PC is on.

Disconnect power before making any configuration changes. The sudden rush of power as you connect a jumper or install a card may damage sensitive electronic components.

Safety Instructions

1. Read these safety instructions carefully.
2. Keep this User Manual for later reference.
3. Disconnect this equipment from any AC outlet before cleaning. Use a damp cloth. Do not use liquid or spray detergents for cleaning.
4. For plug-in equipment, the power outlet socket must be located near the equipment and must be easily accessible.
5. Keep this equipment away from humidity.
6. Put this equipment on a reliable surface during installation. Dropping it or letting it fall may cause damage.
7. The openings on the enclosure are for air convection. Protect the equipment from overheating. **DO NOT COVER THE OPENINGS.**
8. Make sure the voltage of the power source is correct before connecting the equipment to the power outlet.
9. Position the power cord so that people cannot step on it. Do not place anything over the power cord.
10. All cautions and warnings on the equipment should be noted.
11. If the equipment is not used for a long time, disconnect it from the power source to avoid damage by transient overvoltage.
12. Never pour any liquid into an opening. This may cause fire or electrical shock.
13. Never open the equipment. For safety reasons, the equipment should be opened only by qualified service personnel.
14. If one of the following situations arises, get the equipment checked by service personnel:
 15. The power cord or plug is damaged.
 16. Liquid has penetrated into the equipment.
 17. The equipment has been exposed to moisture.
 18. The equipment does not work well, or you cannot get it to work according to the user's manual.
 19. The equipment has been dropped and damaged.
 20. The equipment has obvious signs of breakage.
21. **DO NOT LEAVE THIS EQUIPMENT IN AN ENVIRONMENT WHERE THE STORAGE TEMPERATURE MAY GO BELOW -20° C (-4° F) OR ABOVE 60° C (140° F). THIS COULD DAMAGE THE EQUIPMENT. THE EQUIPMENT SHOULD BE IN A CONTROLLED ENVIRONMENT.**
22. **CAUTION: DANGER OF EXPLOSION IF BATTERY IS INCORRECTLY REPLACED. REPLACE ONLY WITH THE SAME OR EQUIVALENT TYPE RECOMMENDED BY THE MANUFACTURER, DISCARD USED BATTERIES ACCORDING TO THE MANUFACTURER'S INSTRUCTIONS.**
23. The sound pressure level at the operator's position according to IEC 704-1:1982 is no more than 70 dB (A).

DISCLAIMER: This set of instructions is given according to IEC 704-1. Advantech disclaims all responsibility for the accuracy of any statements contained herein.

Contents

Chapter 1	Introduction.....	1
1.1	EtherCAT Introduction.....	2
1.1.1	EtherCAT	2
	Figure 1.1 EtherCAT Function Principle	2
	Figure 1.2 EtherCAT Protocol.....	3
	Figure 1.3 EtherCAT Topology	3
	Figure 1.4 EtherCAT Distributed Clock.....	4
	Figure 1.5 EtherCAT Distributed Clock Jitter	5
1.2	PCI-1203 Features	5
1.2.1	Multi Master Mode	5
1.2.2	Multi-Axis Interpolation.....	6
1.2.3	Huge I/O Data Processing Capability	6
1.2.4	On-board Real-Time OS Support	6
1.2.5	Distributed Clocks (DC)	6
1.2.6	Supports Common Motion SDK.....	6
1.2.7	Error Detection and Diagnostic	6
1.3	Specifications	6
1.3.1	EtherCAT	6
1.3.2	Isolated Digital Input	7
1.3.3	Isolated Digital Output.....	7
1.3.4	General	7
Chapter 2	Installation.....	9
2.1	Unpacking	10
2.2	Driver Installation	10
2.3	Hardware Installation	11
Chapter 3	Hardware	13
3.1	Outline	14
	Table 3.1: Point to point.....	14
3.2	Board ID Switch (SW1)	14
	Table 3.2: Board ID Setting	14
3.3	LAN Port (CN20, CN21).....	15
3.4	D-Sub Connector (CN26).....	15
	Table 3.3: D-Sub Pin Assignment	15
3.5	Signal Connection	16
3.5.1	Digital Input Wiring Reference	16
3.5.2	Digital Output Wiring Reference	16
Chapter 4	EtherCAT Utility	17
4.1	Introduction	18
4.2	Main Form	18
4.2.1	Main Menu	18
4.2.2	Toolbar	19
4.3	Master Pages	23
4.3.1	Single-Axis Motion	24
4.3.2	Multi-Axis Motion.....	30
4.3.3	Synchronized Motion	37
4.3.4	DO / DI / AO / AI	43

	4.3.5	IO Mapping Table	44
	4.3.6	Information	48
4.4		Slave Pages	49
	4.4.1	Single-Axis Motion	49
	4.4.2	Digital Input and Digital Output	50
	4.4.3	Motion IO	50
	4.4.4	IO List	51
	4.4.5	Slave Information	52
	4.4.6	ADAM-E5000 Module Pages	53

Chapter 5 Programming Guide 61

5.1		Introduction	62
5.2		About Common Motion API	62
		Figure 5.1 Using API in different device	62
5.3		Header Files of API	63
		Figure 5.2 Header files of API	63
5.3.1		About APIs	63
5.3.2		About Properties	64
		Table 5.1: Naming rules of properties	65
		Table 5.2: APIs of setting and getting property	65
		Figure 5.3 Relationship between data type of property and func- tions	65
5.3.3		About Device Type	66
		Table 5.3: Device type definition	66
5.3.4		About Error Code	66
5.3.5		Data Type Redefinition	67
		Table 5.4: Data type redefinition	67
5.3.6		Getting Started	67
		Figure 5.4 Flow chart of board initialization	69
5.4		Creating a New Application	71
		Figure 5.5 Software Architecture	71
5.4.1		Creating a New Visual C++ 6.0 Application	71
5.4.2		Creating a New Visual Basic 6.0 Application	73
5.4.3		Creating a New Visual C# Application	73
5.4.4		Creating a New Borland C++ Builder Application	75
5.5		Using Common Motion API in Win7	76
	5.5.1	About Elevating Application Privileges	76
	5.5.2	Using Common Motion API in Win7	77
5.6		Multiple Motion Cards Programming	77
	5.6.1	Independence of Multiple Motion Cards	77
	5.6.2	Operating Multiple Motion Cards	78

Chapter 6 Common Motion Architecture 81

6.1		Introduction	82
6.2		Common Motion Architecture	82
		Figure 6.1 Common Motion Architecture	82
6.3		Naming Rules of APIs and Properties	83
		Table 6.1: Abbreviations and Their Meanings	83
6.4		Device Number	84

Chapter 7 Description of Property Configuration Files 85

7.1		Introduction	86
7.2		Basic Concept of System Configuration	86

7.2.1	Hardware Resource Configuration.....	86
7.2.2	Software Source Configuration.....	89
7.2.3	Resource Integration.....	91
7.3	Generating and Downloading Configuration Files.....	92
7.3.1	Downloading Function of the Configuration File.....	92
7.3.2	Description of Important Information in Configuration Files.....	92
7.4	Modifying Configuration Information.....	92
7.5	Configuring Initialization Status through Controller.....	93
7.5.1	Configuring Initialization Status with Hardware Resource.....	93
7.5.2	Configuring Initialization Status with Software Resource.....	95
7.6	Servo Operation.....	96
7.7	Floating Point PPU.....	97
7.7.1	Properties of Floating Point PPU.....	97
7.7.2	Description of Floating Point PPU Functions.....	97
7.7.3	Description of Floating Point PPU.....	98
7.7.4	Example.....	98

Chapter 8 Motion Status99

8.1	Introduction.....	100
8.2	Axis Status.....	100
8.2.1	Axis Status Functions.....	100
8.2.2	Description.....	100
	Table 8.1: Definition of Axis' Current Status.....	100
	Table 8.2: Definition of Axis' Current Motion Status.....	101
8.2.3	Example.....	101
8.3	Axis Velocity.....	102
8.3.1	Function and Property.....	102
8.3.2	Description.....	103
8.3.3	Example.....	103
8.4	Axis' Motion I/O Status.....	104
8.4.1	Function.....	104
8.4.2	Description.....	104
8.4.3	Example.....	105
8.5	Group Status.....	105
8.5.1	Function.....	105
8.5.2	Description.....	105
	Table 8.3: Definition of Group's Current Status.....	105
8.5.3	Example.....	106
8.6	Group Velocity.....	107
8.6.1	Function and Property.....	107
8.6.2	Description.....	107
8.6.3	Example.....	107

Chapter 9 Motion API109

9.1	Introduction.....	110
9.2	Single-Axis Motion.....	110
9.2.1	About this Section.....	110
9.2.2	Point to Point Motion.....	110
	Table 9.1: Point to point motion functions.....	110
	Table 9.2: Point to point motion properties.....	111
	Figure 9.1 Point to point motion flow chart.....	112
9.2.3	Continuous Motion.....	114
	Table 9.3: Continuous motion functions.....	114
	Table 9.4: Continuous motion properties.....	114
	Figure 9.2 Continuous motion flow chart.....	115
9.2.4	Change Position Motion.....	117
	Table 9.5: Change position motion functions.....	117

	Table 9.6: Change position motion properties.....	117
	Figure 9.3 Change position motion flow chart	119
9.2.5	Change Velocity Motion.....	121
	Table 9.7: Change velocity motion functions.....	121
	Table 9.8: Change velocity motion properties	122
	Figure 9.4 Change velocity motion flow chart.....	123
9.2.6	Simultaneous Start/Stop Motion	124
	Table 9.9: Simultaneous Start/Stop Motion functions	124
	Table 9.10: Simultaneous start/stop motion properties	125
	Figure 9.5 Simultaneous start/stop motion flow chart.....	127
9.2.7	Imposed Motion	129
	Table 9.11: Imposed motion functions	129
	Table 9.12: Imposed motion properties.....	129
	Figure 9.6 Imposed motion.....	130
	Figure 9.7 Imposed motion flow chart	131
9.2.8	Jog Motion	132
	Table 9.13: Jog motion functions	132
	Table 9.14: Jog motion properties.....	133
	Figure 9.8 Jog motion flow chart	135
9.2.9	Homing	136
	Table 9.15: Homing motion functions.....	136
	Table 9.16: Homing motion properties	137
	Figure 9.9 Home motion flow chart.....	154
9.2.10	Axis Event.....	156
	Table 9.17: Axis event functions	156
	Table 9.18: Axis event properties.....	157
	Figure 9.10 Axis event flowchart	159
9.2.11	PT Motion	161
	Table 9.19: PT motion functions.....	161
	Figure 9.11 PT static mode flow chart.....	162
	Figure 9.12 PT dynamic mode flow chart	163
9.2.12	PVT Motion	166
	Table 9.20: PVT functions.....	166
	Figure 9.13 PVT motion flow chart	167
9.3	Interpolation Motion	168
9.3.1	About this Section.....	168
9.3.2	Linear Interpolation.....	168
	Table 9.21: Point to point motion functions	168
	Table 9.22: Linear interpolation motion properties	169
	Figure 9.14 Linear interpolation motion flow chart.....	171
9.3.3	Arc Interpolation.....	173
	Table 9.23: Arc interpolation motion functions	173
	Table 9.24: Arc interpolation motion properties.....	174
	Figure 9.15 Arc interpolation motion flow chart	179
9.3.4	Helix Interpolation	181
	Table 9.25: Helical interpolation motion functions.....	181
	Table 9.26: Helical interpolation motion properties	182
	Figure 9.16 Helical interpolation motion flow chart.....	184
9.3.5	Group Event.....	186
	Table 9.27: Group event functions	186
	Figure 9.17 Group event flow chart	188
9.4	Following Motion	190
9.4.1	About this Section.....	190
9.4.2	E-Gear	190
	Table 9.28: E-Gear motion functions.....	190
	Table 9.29: E-Gear motion properties	191
	Figure 9.18 E-Gear flow chart	192
9.4.3	E-CAM	194
	Table 9.30: E-CAM motion functions.....	194
	Table 9.31: E-CAM motion properties	194

9.4.4	Gantry	199
	Table 9.32: Gantry motion functions.....	199
	Table 9.33: Gantry motion properties.....	200
	Figure 9.19 Gantry motion flow chart.....	201
9.4.5	Tangential Following	203
	Table 9.34: Tangential following motion functions.....	203
	Table 9.35: Tangential following motion properties.....	203
	Figure 9.20 Tangential following flow chart	205
9.5	Path Table Motion	207
9.5.1	About this Section	207
9.5.2	Path Table.....	207
	Table 9.36: Path table motion functions	207
	Table 9.37: Path table motion properties.....	208
	Table 9.38: Path table motion mode.....	208
	Table 9.39: Command in path table motion.....	209
	Table 9.40: Parameter description of Acm_GpAddPath	211
9.5.3	Effect of EndPath	212
9.5.4	Move Mode	213
	Table 9.41: Relational tables of move mode	213
9.5.5	Path DO function.....	217
9.5.6	Start PathTable Motion	218
9.5.7	Pause and Resume Path Table Motion	218
9.5.8	Add Path during Moving.....	222
9.5.9	Flow Chart.....	223
	Figure 9.21 Path table motion flow chart	223
9.5.10	Example	224

Chapter 10 IO API.....227

10.1	About this Section	228
10.2	Function and Property.....	228
	Table 10.1: IO control functions.....	228
	Table 10.2: IO control properties	229
10.3	I/O Mapping.....	229
10.4	Flow Charts	230
10.4.1	Basic Flow.....	230
10.4.2	Event.....	231
10.5	Example	232

Appendix A API List235

A.1	Common.....	236
A.2	Device Object.....	237
A.3	DAQ	256
A.3.1	Digital Input/ Output	256
A.3.2	Analog Input/ Output	260
A.4	Axis	265
A.4.1	System	265
A.4.2	Motion IO	266
A.4.3	Motion Status	268
A.4.4	Velocity Motion.....	270
A.4.5	Point-to-Point Motion	273
A.4.6	Simultaneous Motion	275
A.4.7	Home	277
A.4.8	Position/Counter Control.....	278
A.4.9	Latch	279
A.4.10	Ext-Drive	281
A.4.11	Cam/Gear	282
A.4.12	Gantry/Tangent.....	285

	A.4.13	Stop	286
	A.4.14	PT/PVT Motion	287
A.5	Group		290
	A.5.1	System.....	290
	A.5.2	Motion Status.....	291
	A.5.3	MotionStop.....	292
	A.5.4	Interpolation Motion	293
	A.5.5	Path	305
	A.5.6	Pause & Resume.....	309

Appendix B Property List..... 311

B.1	Device Features.....	312	
B.2	Device Configurations	315	
B.3	DAQ Features	317	
B.4	DAQ Configurations	319	
B.5	DAQ Channel Configurations.....	319	
B.6	Axis Features	324	
	B.6.1	System.....	324
	B.6.2	Speed Pattern	325
	B.6.3	Pulse In.....	325
	B.6.4	Pulse Out.....	326
	B.6.5	Alarm	328
	B.6.6	In Position	328
	B.6.7	ERC	328
	B.6.8	Hardware Limit.....	329
	B.6.9	Software Limit.....	329
	B.6.10	Home	330
	B.6.11	Backlash	331
	B.6.12	Compare	332
	B.6.13	Latch	332
	B.6.14	Cam DO.....	333
	B.6.15	Ext-Drive	333
	B.6.16	Jog	334
	B.6.17	Aux/Gen DIO	334
	B.6.18	Simultaneous Starting.....	335
	B.6.19	Trigger Stop	336
B.7	Axis Configurations	337	
	B.7.1	System.....	337
	B.7.2	Speed Pattern	337
	B.7.3	Alarm	338
	B.7.4	In Position	339
	B.7.5	Hardware Limit.....	339
	B.7.6	Software Limit.....	340
	B.7.7	Home	341
	B.7.8	Backlash	342
	B.7.9	Latch	343
	B.7.10	Ext-Drive	344
	B.7.11	Cam Do.....	345
	B.7.12	Module	347
	B.7.13	Simultaneous Starting.....	347
	B.7.14	Gantry	348
	B.7.15	Trigger Stop	348
B.8	Axis Parameters.....	349	
	B.8.1	Speed Pattern	349
	B.8.2	Home	351
B.9	Group Configurations	352	
	B.9.1	System.....	352
	B.9.2	Path	352

B.10	Group Parameters.....	353
	B.10.1 Speed Pattern.....	353
	B.10.2 System.....	354
	B.10.3 Path.....	354

Appendix C Error Codes355

C.1	Error Codes.....	356
C.2	Common Errors.....	356
C.3	DSP Common Errors.....	366
C.4	EtherCAT Common Errors.....	376
C.5	Unusual Errors.....	378

Chapter 1

Introduction

1.1 EtherCAT Introduction

EtherCAT (Ethernet Control Automation Technology) is a high-performance, Ethernet-based fieldbus industrial network system. The protocol is standardized in IEC 61158 and applies to automation applications that need faster and more efficient communications. Short data update times with precise synchronization make EtherCAT suitable for real-time requirements in automation technology.

1.1.1 EtherCAT

1.1.1.1 Functional Principle

EtherCAT is a real time, high speed and flexible Ethernet based protocol. In EtherCAT network, the master sends Ethernet frames through all of the slave nodes. Standard Ethernet packets or frames are no longer received, interpreted, and copied as process data at every node. Instead, slave devices read the data addressed to them and the input data are inserted at the same time while the telegram passes through the device, processing data “on the fly”. Typically the entire network can be addressed with just one frame. In comparison to other Ethernet based communication solutions EtherCAT utilizes the available full duplex bandwidth efficiently.

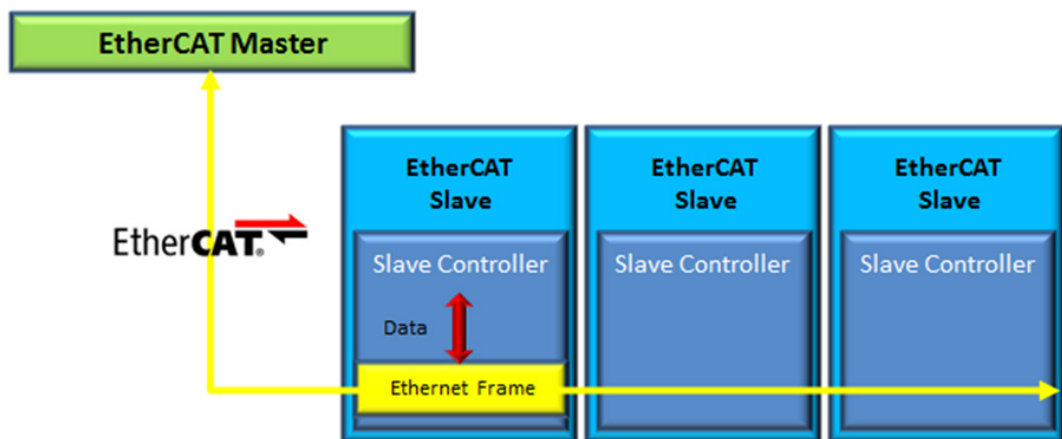


Figure 1.1 EtherCAT Function Principle

1.1.1.2 Protocol

Data exchange are cyclically updated between EtherCAT master and slaves. Data in EtherCAT frames are transported directly within the standard IEEE 802.3 Ethernet frame using Ethertype 0x88a4 and are processed by the EtherCAT Slave Controller on the fly. Each EtherCAT datagram is a command that consists of a header, data and a working counter. The datagram header indicates what type of access the master device would like to execute:

- Read, write, read-write
- Access to a specified slave device through direct addressing
- Access to multiple slave devices through logical addressing

Logical addressing is used for the cyclical exchange of process data. The header and data are used to specify the operation that the slave must perform, and the working counter is updated by the slave to let the master to know that a slave has processed the command.

Every EtherCAT datagram ends with a 16 Bit Working Counter (WKC). The Working Counter counts the number of devices that were successfully addressed by this EtherCAT datagram.

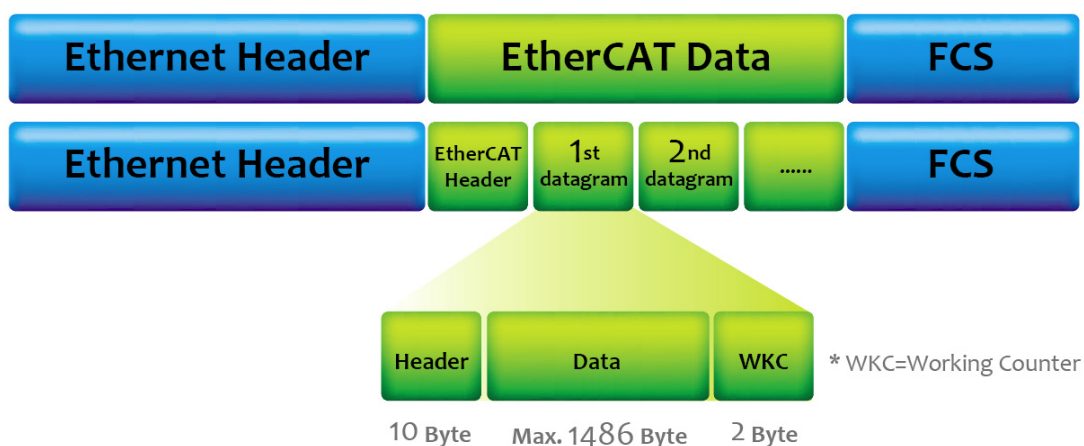


Figure 1.2 EtherCAT Protocol

EtherCAT datagrams are processed before receiving the complete frame.

In case data is invalid, the frame check sum (FCS) is not valid and the slave will not set data valid for the local application.

1.1.1.3 Topology

EtherCAT supports a variety of network topologies, including line, tree, ring and star. The line and tree topologies are more conducive to fieldbus applications because they require fewer connections and utilize a much simpler and more flexible cabling schema that switches and hubs are not necessary for lines or trees topology.

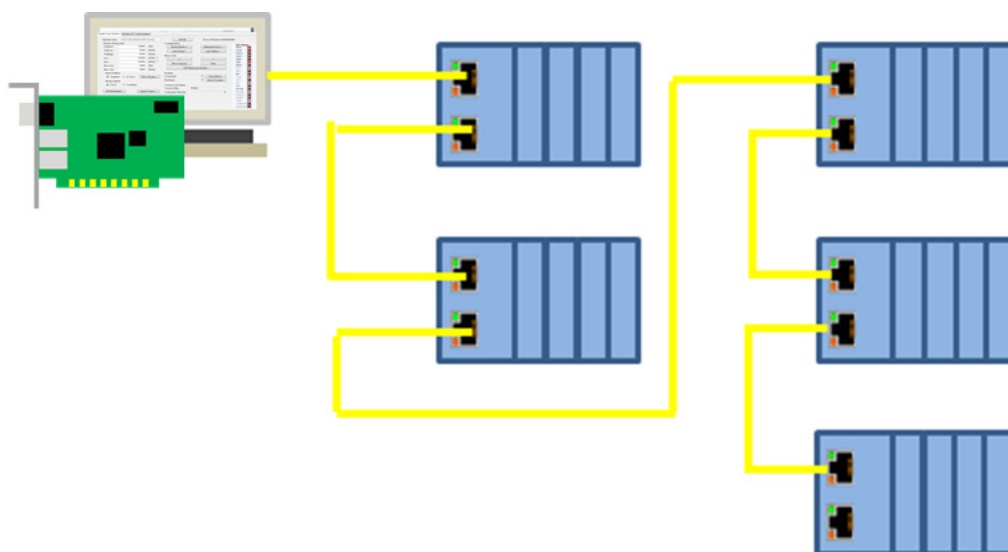


Figure 1.3 EtherCAT Topology

Inexpensive industrial Ethernet cables up to 100m apart, can be used between two nodes in 100BASE-TX mode. EtherCAT makes a pure bus or line topology with hundreds of nodes possible without limitations. Up to 65,535 devices can be connected to EtherCAT, so network expansion is almost unlimited.

EtherCAT supports individual nodes to be connected and disconnected during operation. If one of the slaves in the network is removed, the rest of the network can continue to operate normally. Additionally, EtherCAT also enables other communication features such as cable redundancy or even master redundancy with Hot Standby.

1.1.1.4 Synchronization

A Distributed Clock (DC) mechanism is used to provide highly precise time synchronization between slaves in an EtherCAT network, which is equivalent to the IEEE 1588 Precision Time Protocol standard. By using distributed clocks, EtherCAT is able to synchronize the time in all local bus devices within a very narrow tolerance range. All EtherCAT slaves are provided with an internal clock which named as System Time ($t_{Local\ Time}$). One EtherCAT Slave, usually the first slave, will be used as a Reference Clock and cyclically distributes its Clock.

Possible misalignments between the reference clock and the clocks of the other slaves are usually due to the following reason: when a slave is switched on, the internal free-running register that holds the current time is reset to zero. Unfortunately, this action does not take place exactly at the same time in all the different slaves, and this result in an initial offset (t_{offset}) among clocks that has to be compensated.

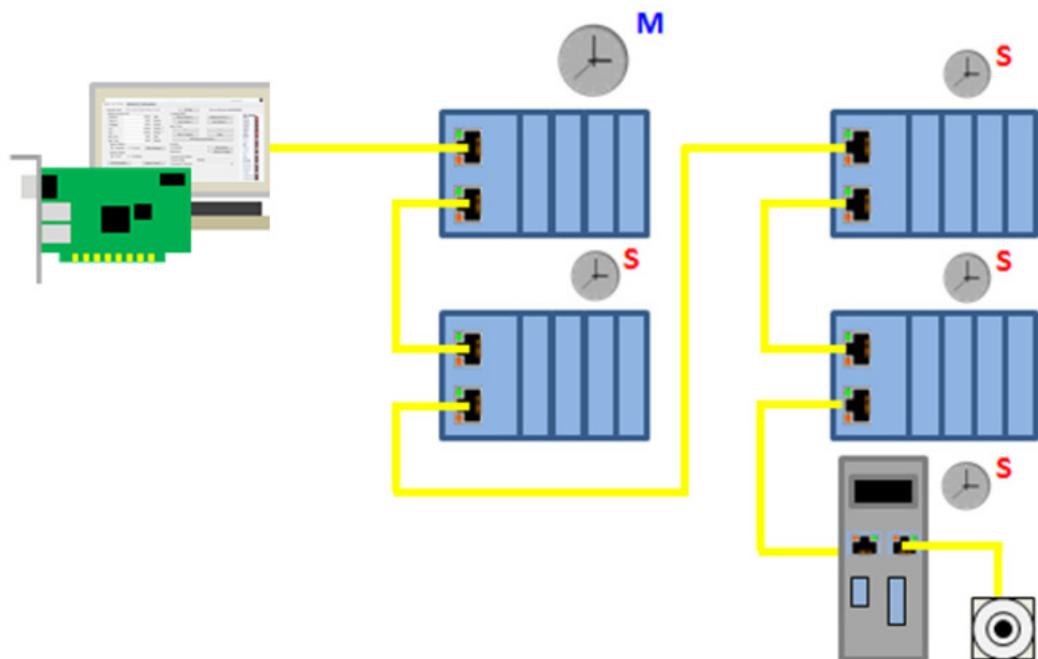


Figure 1.4 EtherCAT Distributed Clock

Typically, the master sends a broadcast to all other slaves in the system. Once receiving the message, slaves will latch the value of their internal clock. There are two latch values, one is receiving and the other is returning back. Thus, the master can read all latched values and calculate the delay for each slave ($t_{Propagation\ Delay}$). Delays will be stored into offset register. In the following, master will send a message periodically to all other slaves in EtherCAT network to make the first slave the reference clock and forcing all other slaves to set their internal clock by the calculated offset.

$$\Delta t = (t_{Local\ Time} + t_{Offset} - t_{Propagation\ Delay}) - t_{Received\ System\ Time}$$

Because synchronization between slaves in DC mode is done by internal clocks in hardware, EtherCAT guarantee the time jitter is less than 1us.

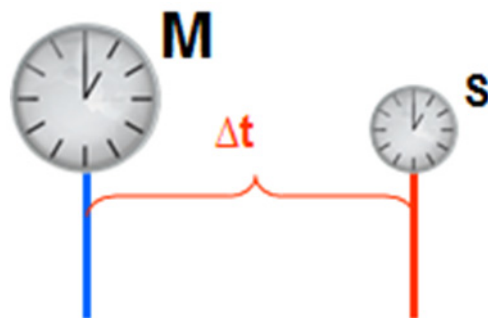


Figure 1.5 EtherCAT Distributed Clock Jitter

1.1.1.5 Diagnosis with exact localization

EtherCAT is an ultra -fast I/O system. To reach the best high-speed communication, high communication accuracy is demanded. EtherCAT comprises a wide range of system-inherent diagnostic features which help detect and locate system errors precisely. Apart from broken wire detection and localization, the protocol, physical layer and topology of the EtherCAT system enable individual quality monitoring of each individual transmission segment.

As mentioned, every EtherCAT datagram ends with a 16 bit Working Counter (WKC) to count the number of devices that were successfully addressed by this EtherCAT datagram. Master can check the data exchange situation by WKC in the same cycle and the error frame can be detected by analyzing the nodes' error counters. The slave application will be executed only as the frame is received correctly. The automatic evaluation of the associated error counters enables precise localization of critical network sections.

Bit errors during transmission are detected reliably by the analysis of the CRC (Cyclic Redundancy Check) check sum. CRC is an error-detecting code commonly used in digital networks and storage devices to detect accidental changes to raw data. In addition to the error detection and localization protocol, transmission physics and topology of the EtherCAT system allow an individual quality monitoring of every single transmission path. There is a very effective monitoring mechanism in EtherCAT.

1.2 PCI-1203 Features

Advantech's EtherCAT hardware master solution is the PCI-1203 with two ethernet ports acting as Motion and I/O master, respectively. There are four versions of PCI-1203: 6/10/16/32-axis. Process of data exchange and memory location in EtherCAT communication can be executed automatically by PCI-1203. User can benefit from EtherCAT high performance communication feature without handling complicated EtherCAT communication protocol.

1.2.1 Multi Master Mode

- Support for separated master instances simultaneously using different EtherCAT ports on PCI-1203.
- Motion master cycle time is less than 500 us
- I/O master cycle time is less than 200 us

1.2.2 Multi-Axis Interpolation

- Supports 32 axes of Panasonic A5B EtherCAT servo motors
- Supports 6 groups and 8-axis linear interpolation per group

1.2.3 Huge I/O Data Processing Capability

- Supports Advantech ADAM-5000/ECAT EtherCAT I/O slaves
- Supports extra on-board 8-CH isolated DI and 4-CH isolated DO

1.2.4 On-board Real-Time OS Support

- Process data by on-board 650MHz dual-core ARM processor without wasting CPU resource in PC
- Supports high-accuracy trajectory planning and fast-response time
- Speed optimized exchange of cyclic data

1.2.5 Distributed Clocks (DC)

- Synchronizes all slave clocks with one slave reference clock
- Delay compensation
- Runtime monitoring and correction of deviation

1.2.6 Supports Common Motion SDK

- Support “Common Motion API” to integrate all Advantech SoftMotion controller
- EtherCAT slaves are completely configurable via the API without ENI data
- Graphical utility interface is easy to use and configure EtherCAT system

1.2.7 Error Detection and Diagnostic

- Lost link monitoring
- Detection and retry of timed out and failed EtherCAT command
- Comprehensive diagnostic data of physical, device and master layer.

1.3 Specifications

1.3.1 EtherCAT

Number of Axes	6/10/16/32
Number of Rings	2
Surge Protection	10kV
Communication Time	Motion: 500us I/O: 200us
Communication Motion Slave	32 Servo Drive Max.(eq. Panasonic A5B)
Communication IO Slave	128 port DI (128 byte) 128 port DO (128 byte) 128 channel AI (256 byte) 128 channel AO (256 byte) *(based on ADAM-5000/ECAT)

1.3.2 Isolated Digital Input

Channels	8
Input Voltage	Logic 0: 5 V max. Logic 1: 6 V min. (24V max.) Needs 24VDC external power
Input Resistance	8.4 k Ω
Input delay time	150us
Isolation Protection	1,000 V _{DC}

1.3.3 Isolated Digital Output

Channels	4
Output Type	Sink
Output Voltage	12 ~ 24 V _{DC}
Sink Current	300mA/ Per channel (23°C) 200mA/ Per channel (60°C)
Switch Frequency	30KHz @ External power 24V _{DC}
Isolation Protection	1,000 V _{DC}

- Note!**
1. External power range: +12V_{DC} to +24V_{DC}
 2. DO switch frequency is estimated based on the resistor load with 4.7K ohm.
 - 30KHz @ External power 24V_{DC}
 - 50KHz @ External power 22V_{DC}
 - 100KHz @ External power 18V_{DC}
 - 200KHz @ External power 12V_{DC}



1.3.4 General

Bus Type	Universal PCI V2.2
Power	+5V _{DC} (from PCI-Bus)
Certification	CE, FCC Class A
Connectors	RJ45 x 2, DB15 x 1
Dimensions (L x H)	175 x 100 mm (6.9" x 3.9")
Power Consumption	5 V _{DC} @ 0.5 A typical
Humidity	5 ~ 95% RH, non-condensing (IEC 60068-2-3)
Operating Temp.	0 ~ 60°C (32 ~ 140°F)
Storage Temp.	-20 ~ 85°C (-4 ~ 185°F)

Chapter 2

Installation

2.1 Unpacking

After receiving your PCI-1203 package, inspect the contents first. The package should include the following items:

- PCI-1203 card
- CD-ROM (DLL driver & user manual included)

The PCI-1203 card has certain electronic components vulnerable to electro static discharge (ESD). ESD could easily damage the integrated circuits and certain components if preventive measures are not carefully taken.

Before removing the card from the antistatic plastic bag, you should take the following precautions to prevent ESD damage:

- Touch the metal part of your computer chassis with your hand to discharge static electricity accumulated on your body. Or one can also use a grounding strap.
- Touch the antistatic bag to a metal part of your computer chassis before opening the bag.
- Hold of the card only by the metal bracket when taking it out of the bag.

After taking out the card, you should first:

- Inspect the card for any possible signs of external damage (loose or damaged components, etc.). If the card is visibly damaged, notify our service department or the local sales representative immediately. Avoid installing a damaged card into your system.

Also pay extra attention to the followings to ensure a proper installation:

- Avoid physical contact with materials that could hold static electricity such as plastic, vinyl and Styrofoam.
- Whenever you handle the card, grasp it only by its edges. DO NOT TOUCH the exposed metal pins of the connector or the electronic components.

2.2 Driver Installation

We recommend you install the driver before you install the PCI-1203 card into your system.

The DLL driver setup program for the card is included on the companion CD-ROM that is shipped with package. Follow the steps below to install the driver software:

1. Insert the companion CD-ROM into your CD-ROM drive.
2. The setup program will be launched automatically if you have the auto-play function enabled on your system.
3. Select the proper Windows OS option according to your operating system. Just follow the installation instructions step by step to complete your DLL driver setup.
4. Then setup the PCI-1203 Motion Utility automatically.

For further information on driver-related issues, an online version of the Device Drivers Manual is available by accessing the following path:

\Program Files\Advantech\Common Motion\Manual

The example source codes could be found under the corresponding installation folder, such as the default installation path:

\Program Files\Advantech\Common Motion\Example_1203

2.3 Hardware Installation

Note! *Make sure you have installed the driver first before you install the card (refer to 2.2 Driver Installation).*



After the DLL driver installation is completed, you can now go on to install the PCI-1203 card in any PCI slot on your computer. But it is suggested that you refer to the computer's user manual or related documentations if you have any doubt. Follow the steps below to install the card on your system.

1. Turn off your computer and remove any accessories connected to the computer.

Warning! *CUT OFF power supply of your computer whenever you install or remove any card, or connect and disconnect cables.*



2. Disconnect the power cord and any other cables from the back of the computer.
3. Remove the cover of the computer.
4. Select an empty +3.3/+5 V PCI slot. Remove the screws that secure the expansion slot cover to the system unit. Save the screws to secure the retaining bracket of interface card.
5. Carefully grasp the upper edge of the PCI-1203. Align the hole in the retaining bracket with the hole on the expansion slot and align the gold striped edge connector with the expansion slot socket. Press the card into the socket gently but firmly. Make sure the card fits the slot tightly. Use of excessive force must be avoided; otherwise the card might be damaged.
6. Fasten the bracket of the PCI card on the back panel rail of the computer with screws.
7. Connect appropriate accessories (cable, wiring terminals, etc. if necessary) to the PCI card.
8. Replace the cover of your computer and connect the cables you removed in step 2.
9. Turn on your computer.

Chapter 3

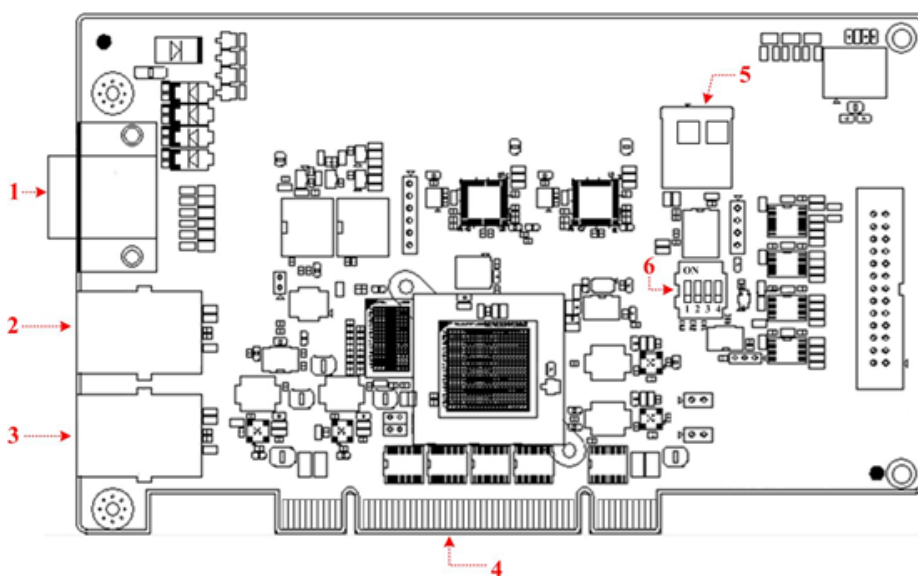
Hardware

3.1 Outline

Table 3.1 shows location of the connectors and switches of the PCI-1203.

Table 3.1: Point to point

No.	Part Reference	Function
1	CN26	Isolated digital input/output connector
2	CN21	RJ45 with Transformer (LAN 1)
3	CN20	RJ45 with Transformer (LAN 0)
4	Gold finger	PCI-Bus
5	CN25	Micro-SD card slot (No support)
6	SW1	Board ID switch
7	CN30	Boot master from flash memory (Short Pin1 and Pin2)

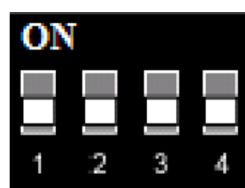


Note! Confirm that Pin1 and Pin2 are short-circuited by a jumper before using PCI-1203.



3.2 Board ID Switch (SW1)

Table 3.2: Board ID Setting



SW1

Pin	Label	Switch ON	Switch OFF
1	ID3	0 (ID3 Value =0)	1 (ID3 Value =1)
2	ID2	0 (ID2 Value =0)	1 (ID2 Value =1)
3	ID1	0 (ID1 Value =0)	1 (ID1 Value =1)
4	ID0	0 (ID0 Value =0)	1 (ID0 Value =1)

Board ID = (8×ID3 Value) + (4×ID2 Value) + (2×ID1 Value) + (ID0 Value)

The defaulted setting is ON, e.g. the defaulted values is 0.

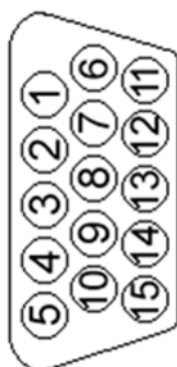
3.3 LAN Port (CN20, CN21)

There two RJ45 10/100/1000 LAN port connectors, which connect to EtherCAT slaves via a Cat.5e LAN cable. CN20 is designed to the EtherCAT motor driver slaves (e.g. Panasonic A5B). CN21 is used for the EtherCAT IO slaves (e.g. ADAM-5000/ECAT).

3.4 D-Sub Connector (CN26)

The connector of the 8 sink type digital inputs (DI) and the 4 sink type digital outputs (DO) on the PCI-1203 is the D-Sub 15-pin connector, which provides us to connect a wiring board via a shielded cable. Table 3.3 shows the pin assignment of CN26. For the shielded cable design, the wire pairs, such as (pin1, pin6), (pin2, pin 7), (pin3, pin8), (pin4, pin9), (pin5, pin14), (pin11, pin12) and (pin14, pin15), should be twisted.

Table 3.3: D-Sub Pin Assignment



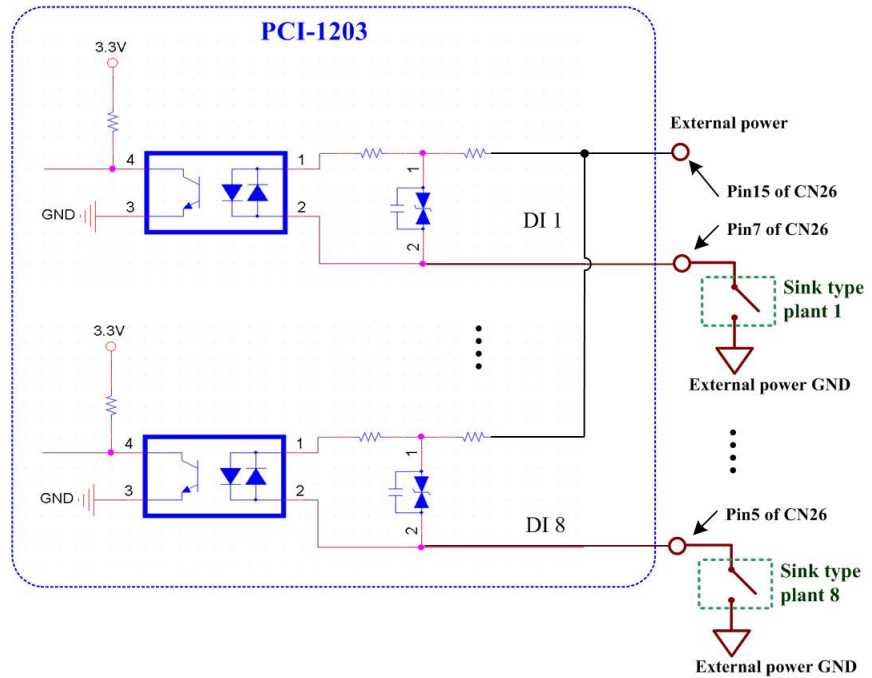
CN26

Pin	Description	Pin	Description
1	DO 3	9	DI 5
2	DI 2	10	DI 7
3	DI 4	11	DO 2
4	DI 6	12	DO 4
5	DI 8	13	External power GND
6	DO 1	14	External power GND
7	DI 1	15	External power (+24V)
8	DI 3		

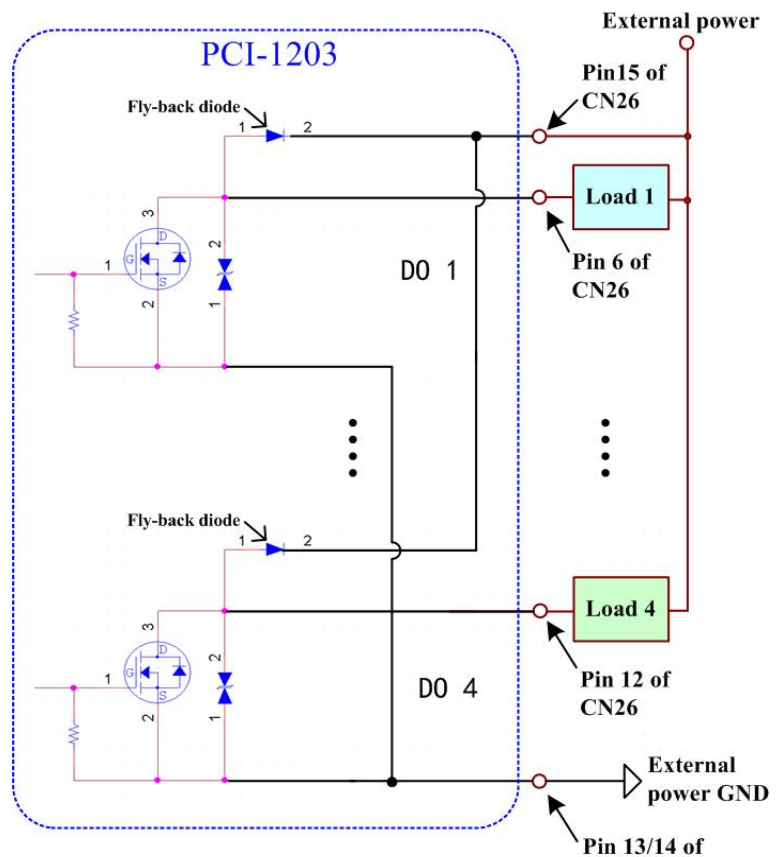
3.5 Signal Connection

This section provides useful information about how to connect input and output signals on PCI-1203.

3.5.1 Digital Input Wiring Reference



3.5.2 Digital Output Wiring Reference



Chapter 4

EtherCAT Utility

4.1 Introduction

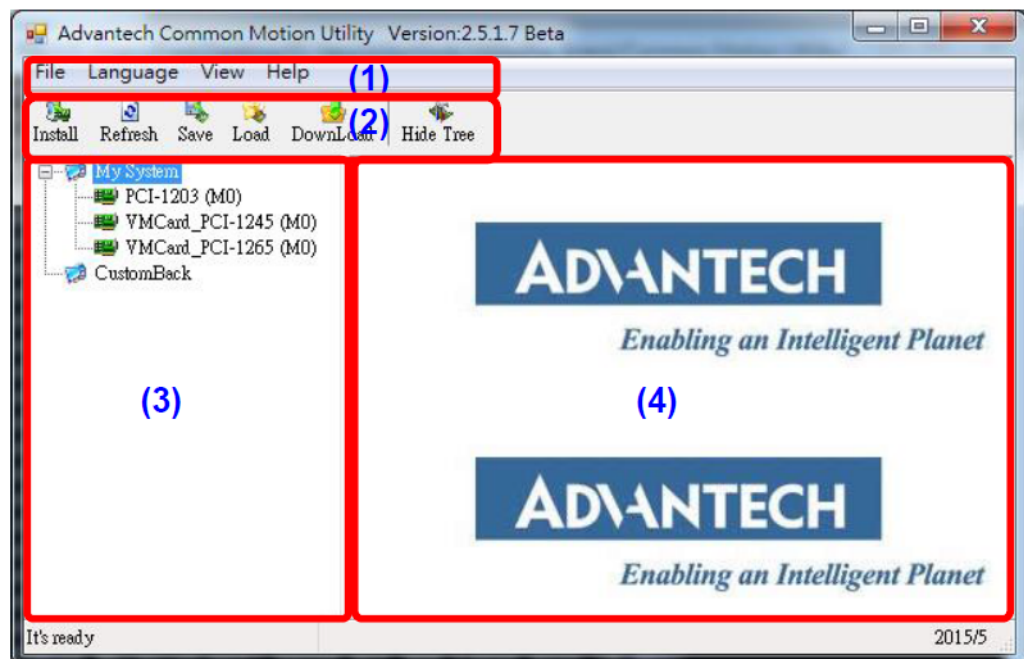
The Common Motion Utility is a tool to test the functions of PCI-1203. It is developed on Common Motion API and Common Motion Component, so it supports all Common Motion based products such as PCI-1240U, PCI-1245 and PCI-1285 series motion cards. Users can use the same Utility to test all Common Motion based cards on PC.

The following interfaces will be introduced in this chapter:

1. Main Form: the main interface of this utility. Includes Main Menu, Toolbar, Device Tree and Control Panel.
2. Device Page: contains most of the operations of PCI-1203. Includes Single Axis Motion, Multi-Axis Motion, Synchronized Motion, IO Operations and IO Mapping Table.
3. Slave Page: contains the operations of the slaves. Includes Single Axis Motion, GPIO Operations and Motion IO Settings.
4. ADAM-E5000 Module Page: contains the operations and configurations of ADAM-E5000 Modules.

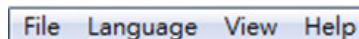
4.2 Main Form

The main form contains (1) Main Menu, (2) Toolbar, (3) Device Tree and (4) Control Panel. Devices would show on the Device Tree if the installation is correct and successful. After the user selects a device from Device Tree, the corresponding control page will show in Control Panel.

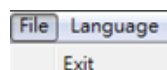


4.2.1 Main Menu

The operations of the Main Menu are described below.

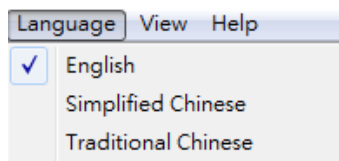


4.2.1.1 File



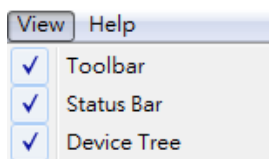
Users can click **[Exit]** to terminate the utility.

4.2.1.2 Language



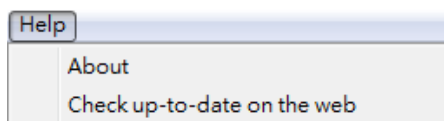
Through this menu, the language in the Utility can be switched. This utility supports three languages: English, Simplified Chinese and Traditional Chinese. After you select a language, the corresponding menu item will be checked. When you close the Utility, the language you selected will be saved to register. When opened next time, the utility's language will be last used one.

4.2.1.3 View



This menu allow users to display/hide the toolbar, status bar and device tree. If Toolbar/Status Bar/Device Tree is visible, the corresponding menu item will be checked.

4.2.1.4 Help



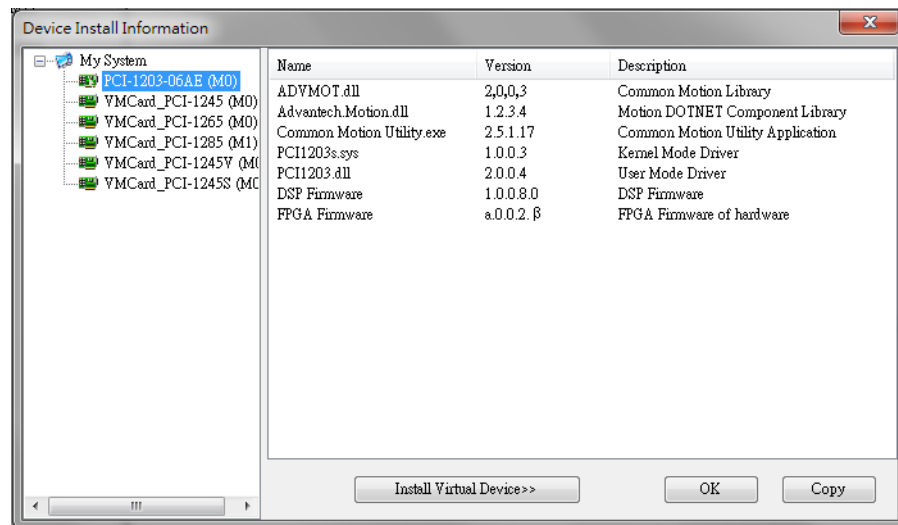
The [**About**] menu item supports the copyright notice of the driver and utility for device. Click [**Check up-to-date on the web**], you can link to company's website to check whether the firmware, driver and utility are the latest ones by comparing version information of Install interface.

4.2.2 Toolbar



4.2.2.1 Install

Click [Install], a new window will pop up as below, which shows the version information of device driver, hardware, firmware and utility.



ADVMOT.dll: The common motion API for development.

Advantech.Motion.dll: The .NET motion control library.

Common Motion Utility.exe: The utility which is running now.

PCI1203s.sys: Kernel-mode driver.

PCI1203.dll: User-mode driver.

4.2.2.2 Refresh

Research the available common motion cards in the computer and refresh the device tree.

4.2.2.3 Save

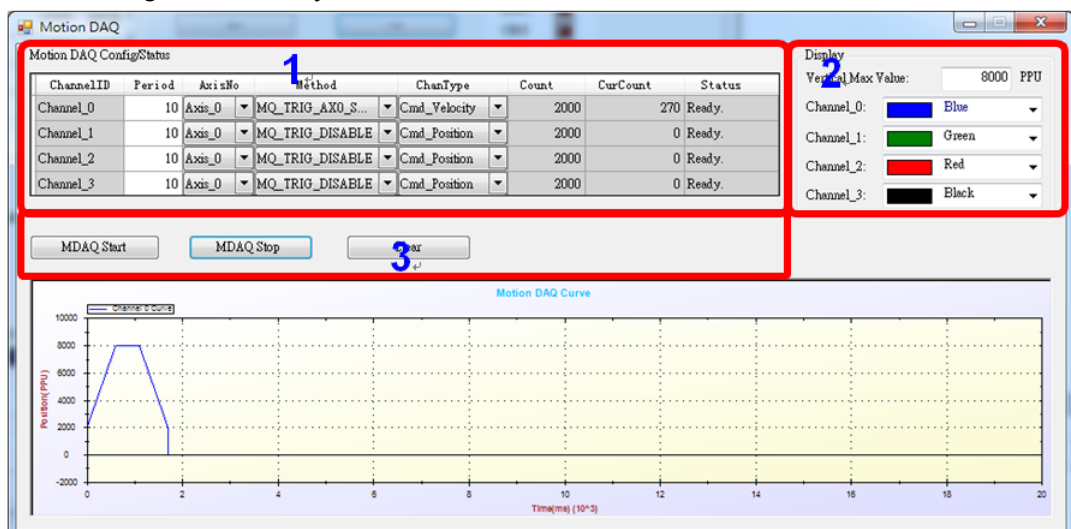
Export all properties and configuration of the axes of the selected device.

4.2.2.4 Load

Import configurations of all axes of the selected device. Select the previously exported configuration to import the setting into the device.

4.2.2.5 Motion DAQ

After clicking the button, you'll see the interface as below:



The tool is designed to show the Motion Data Acquisition function. In the PCI-1203, four channels are provided for real-time motion data acquisition. Each of them can acquire Command/Actual/Lag (the difference between Command and Actual) motion data of any axis, with the max. data count of 2000.

This interface consists of the following parts:

1. Motion DAQ Config/Status

Configure acquisition data of each channel. ChannelID, Count, CurCount and Status row are read-only (the background is grey).

ChannelID	Period	AxisNo	Method	ChanType	Count	CurCount	Status
Channel_0	10	Axis_0	MQ_TRIG_DISABLE	Cmd_Position	2000		
Channel_1	10	Axis_0	MQ_TRIG_DISABLE	Cmd_Position	2000		
Channel_2	10	Axis_0	MQ_TRIG_DISABLE	Cmd_Position	2000		
Channel_3	10	Axis_0	MQ_TRIG_DISABLE	Cmd_Position	2000		

ChannelID: Four channels are provided.

AxisNo: Select the axis for measuring.

Period: Acquisition period, which means the interval between each data is acquired. The range is 1 to 255 ms. In order to unify the max value of horizontal ordinate of Curve window, Period value of each channel will adopt the same value. Therefore, if Period value of one channel has been changed, other channels will change accordingly.

Method: Trigger mode. Trigger modes of data acquisition are as followings:

MQ_TRIG_DISABLE: Disable data acquisition function

MQ_TRIG_SW: Start acquisition after click [MDAQ Start]

MQ_TRIG_AXN_START: Trigger when axis N starts to move

ChanType: The source of data acquisition, the value could be Cmd_Position (Command Position), Act_Position (Actual Position), Lag_Position (The difference between Command Position and Actual Position) and Cmd_Velocity (Command Velocity).

Count: Count of acquired data.

CurCount: Show how many data have been acquired.

Status: Could be "Ready", data acquisition function is not started yet, "Wait Trigger", waiting for trigger, and "Started", motion data is being acquired.

2. Display



Configure the color of each channel curve and the max value of vertical coordinate of picture box. Select a color then the color of corresponding curve will be changed.

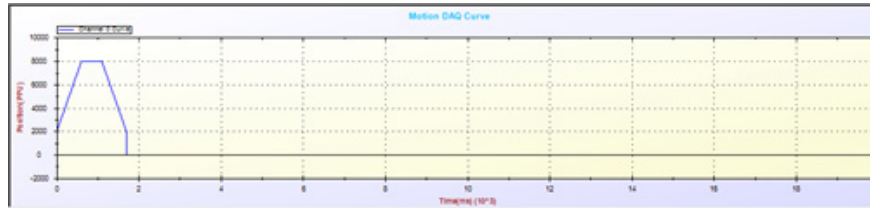
3. Operations

MDAQ Start: Start motion data acquisition function. When trigger conditions are met, the acquisition of motion data will be started

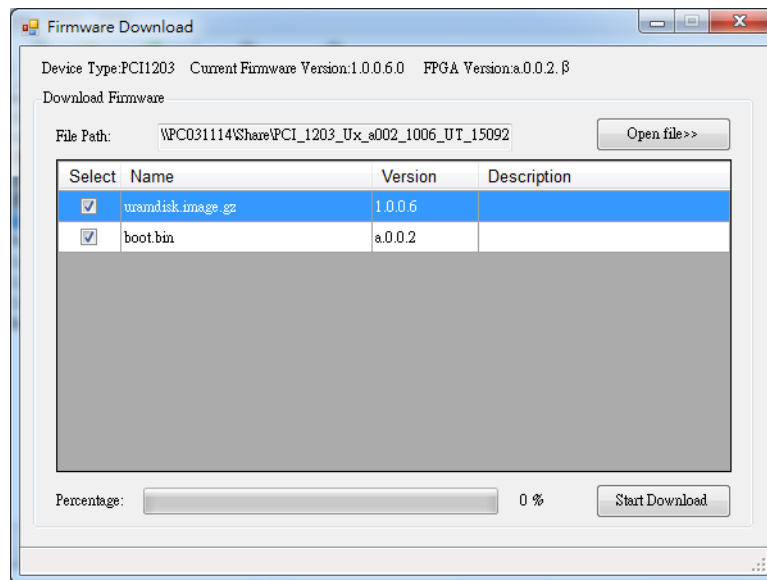
MDAQ Stop: Stop motion data acquisition

Clear: Clear curve of each channel

When data acquisition is finished, the curve of corresponding acquisition data of each channel will be displayed in the below picture box, which is shown as below:



4.2.2.6 Download



The tool upgrades the firmware of PCI-1203. The top of this dialog shows the current device type, name and firmware version. Click **[Open File]** to select the latest firmware file you have acquired. The FPGA and firmware version in this firmware package will be listed in the data list. Click **[Start Download]** to activate the downloading procedure to hardware and the progress bar will show the task progress. After the upgrade procedure finished, please power down your computer and restart it to ensure PCI-1203 loading the newest firmware version.

Note! *DO NOT close the upgrade dialog and utility while downloading the firmware to hardware.*



DO NOT turn off the power while Firmware Upgrade is in progress, damage may occur. If the downloading procedure is interrupted by power outages or other problems, the hardware may need to be sent back to Advantech for firmware update.

4.2.2.7 Hide Tree

This button is provided to hide/show Device Tree.

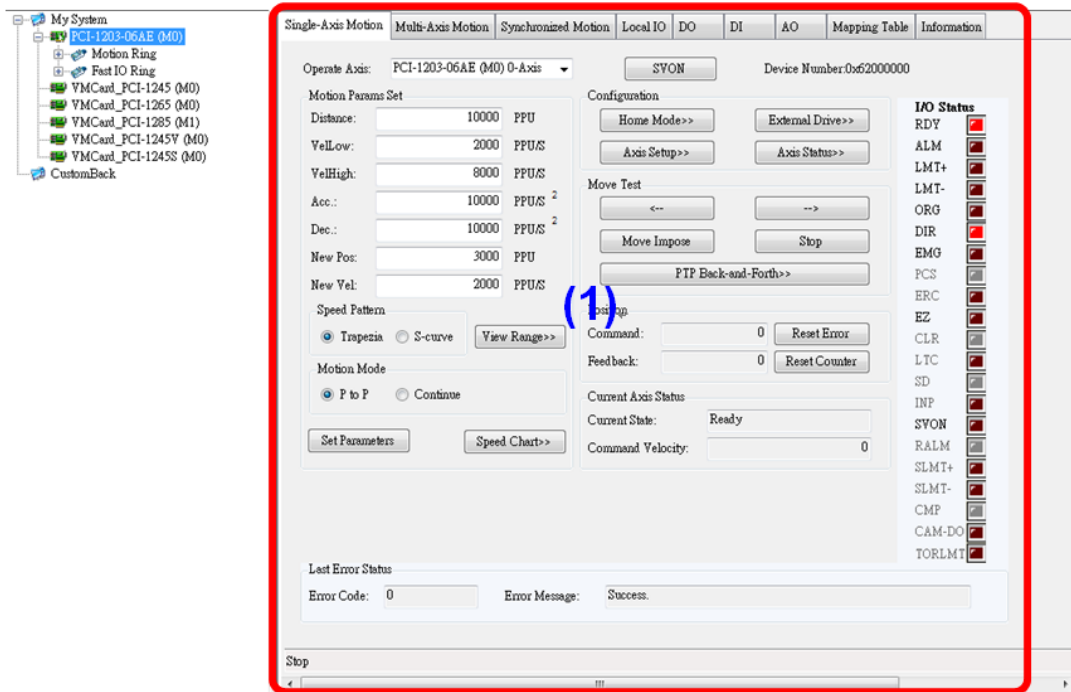
If Device Tree is currently shown, click the button to hide it and the text on the button will change to "Show Tree".

If Device Tree is currently hidden, click the button to show it and the text on the button will change to "Hide Tree".

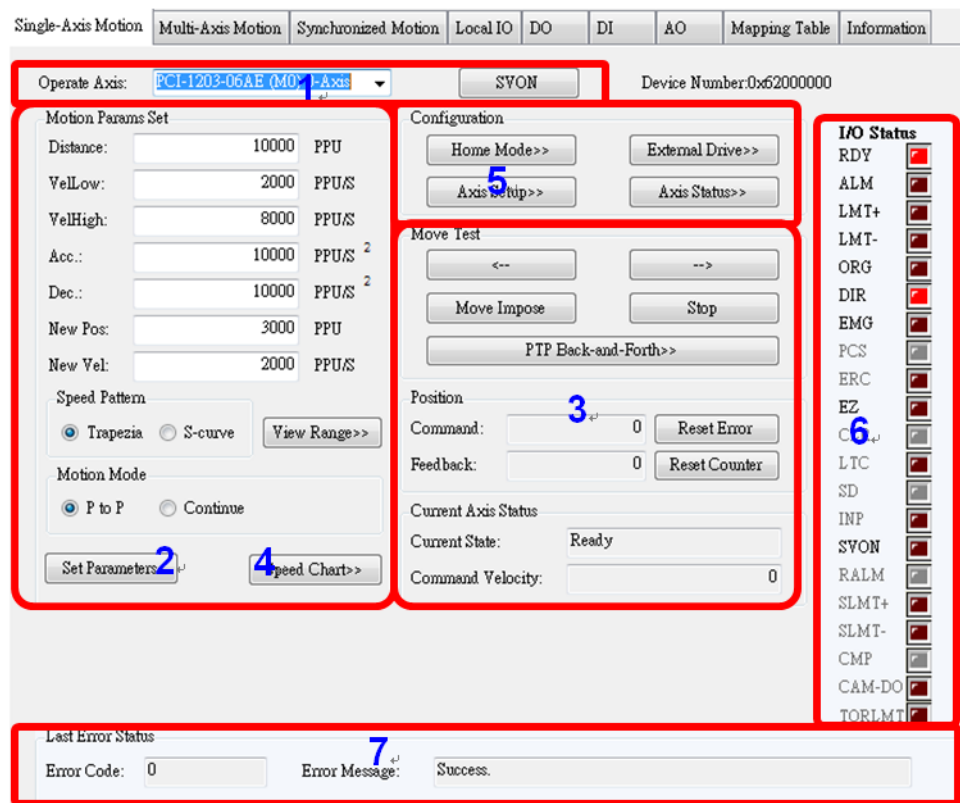
4.3 Master Pages

After selecting PCI-1203 card in the Device Tree, the operating pages will show in the (1) Control Panel and all the slaves connected to the PCI-1203 will be shown as child nodes in the PCI-1203 node.

The control pages of PCI-1203 node include Single-Axis Motion, Multi-Axis Motion, Synchronized Motion, DAQ (including DI, DO, AI, AO), Mapping Table and Device Information pages. For more detailed operation for slaves, users can click the slave node in the Device Tree. The corresponding control pages will show in the Control Panel.



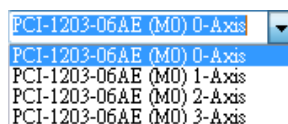
4.3.1 Single-Axis Motion



4.3.1.1 Select Axis and Servo ON

Click on the drop-down combo box and select the axis you want to operate. The axis index is defined by the network connection position. The first axis connected to PCI-1203 will have index 0. The second one will have index 1, and so on. The number of items in combo box is according to how many axes you connected to PCI-1203.

In this sample below, four axes are connected to PCI-1203.

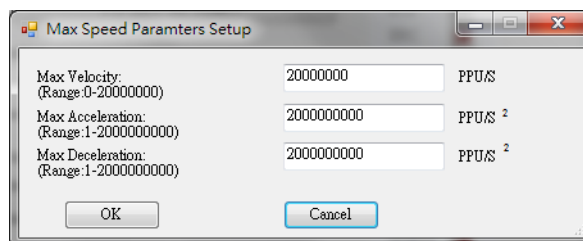


Click **[SVON]** button to servo on / off the selected axis.

4.3.1.2 Set Motion Parameter

Set the distance (Distance), initial velocity (VelLow), running velocity (VelHigh), acceleration (Acc.), deceleration (Dec.) and choose the speed pattern (Trapezia or S-Curve) for P2P and Continuous motion. The New Pos. and New Vel. are for superimposed motion.

Click **[View Range]** to check or set the maximum velocity, acceleration and deceleration. The dialog will show as follow.



Note! *VelHigh in Single-axis Motion cannot be greater than the Max Velocity; Acc. cannot be greater than the Max Acceleration and Dec. cannot be greater than the Max Deceleration.*



After finishing the parameter setting for operation, click **[Set Parameter]** to save and apply the values to device.

4.3.1.3 Move Test



After finishing the parameter setting, click [**<--**] or [**-->**], the axis will do P2P / Continuous motion in negative or positive direction. Click [**Stop**] button to stop the axis. The command position and feedback position will show on the Position panel. Check the current status and command speed in the Current Axis Status panel.

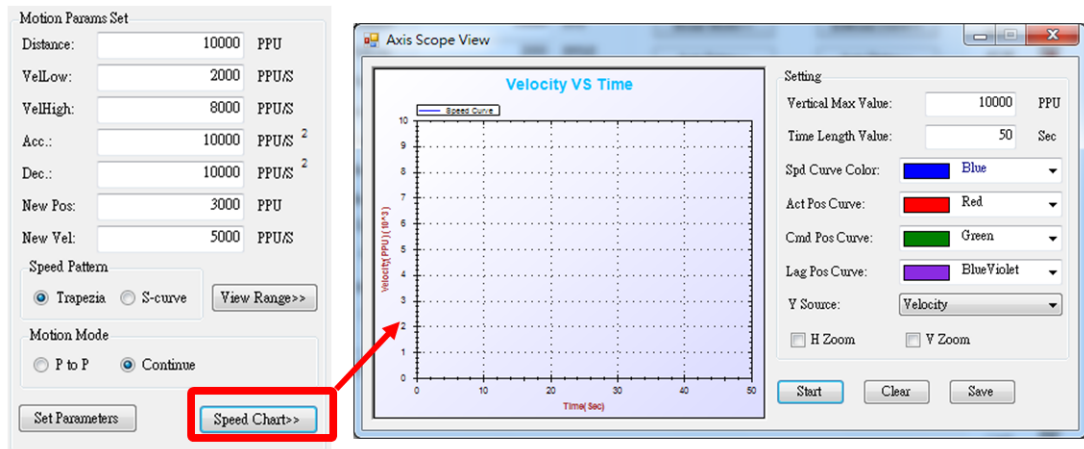
For more details on the axis state, please refer to the `Acm_AxGetState` function listed in Common API of Programming guide.

After the movement velocity reaches VelHigh in P2P motion, users can click [**Move Impose**] to generate a superimposed movement, the distance of the imposed movement is the value of New Pos and the velocity of the imposed movement is the value of New Vel.

Click [**PTP Back-and-Forth**], an advanced P2P motion window will pop out. Users can set P2P motion to multiple axes in the meanwhile in this function.

4.3.1.4 Speed Chart

Users can check the velocity curve by clicking this button and the following window will appear.



Setting items are as follows:

1. Vertical Max Value: maximum vertical coordinate.
2. Time Length Value: maximum horizontal coordinate.
3. Spd Curve Color: the color for speed curve.
4. Act Pos Curve: the color for actual position curve.
5. Cmd Pos Curve: the color for command position curve.
6. Y Source: data source for vertical coordinate. It may be velocity, actual position, command position or all of these three.
7. H Zoom: if it is checked, it indicates horizontal zoom is enabled, users can select appropriate region by the mouse to zoom in.
8. V Zoom: if it is checked, it indicates vertical zoom is enabled, users can select appropriate region by the mouse to zoom in.

Click to draw the curve.

Click to clear the graphic panel.

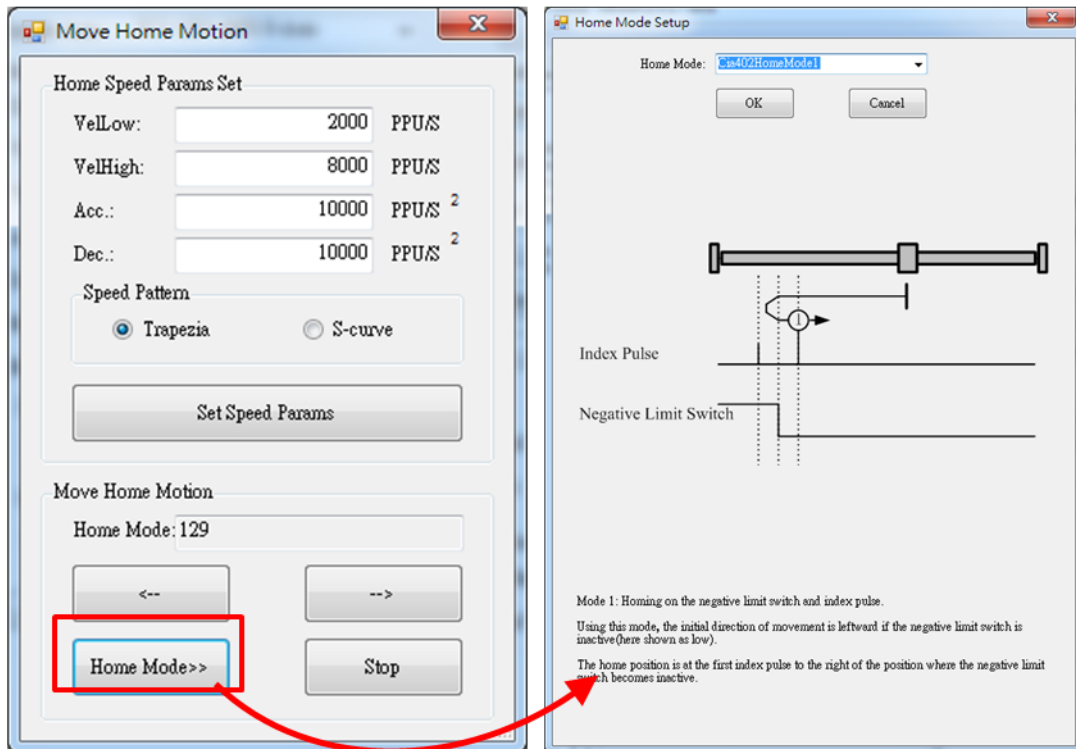
Click to save the curve as .png, .gif, .jpg, .tif or .bmp format file.

4.3.1.5 Configuration



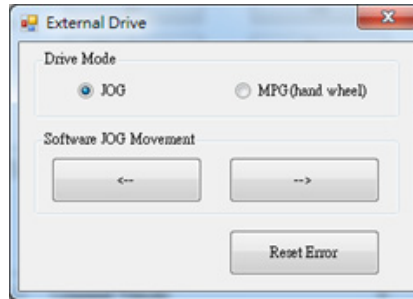
It includes Home Mode configuration, External Drive mode, the axis property configuration and I/O status of the axis.

Home Mode



Before performing home movement, you need to select the home mode first. The PCI-1203 supports all the home modes defined in CiA402. Brief homing procedures are described in the Home Mode Setup window. For detailed information, refer to the description about Home Mode in Common API of Programming guide. Set the home speed parameters in **Home Speed Params Set** Panel and click **[Set Speed Params]** button to apply the speed parameters. After all the setting are complete, click **[←]** or **[→]** to start homing.

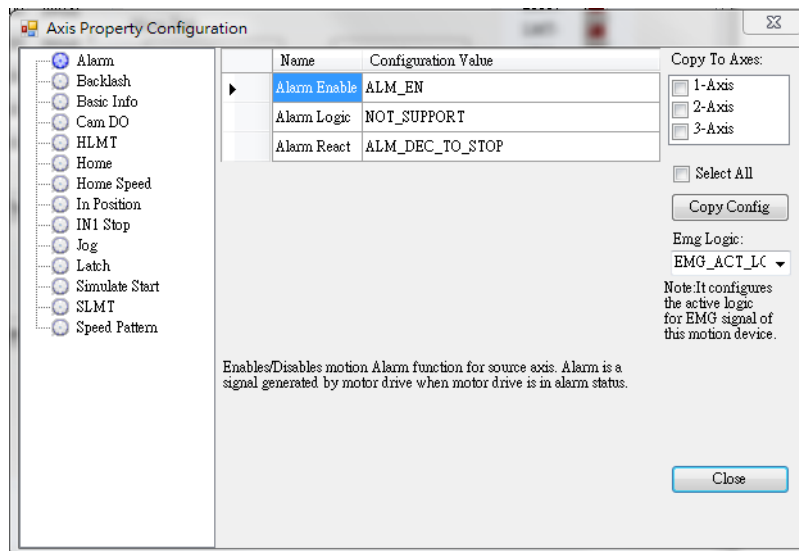
External Drive



Click this button to enable the external drive function. PCI-1203 now only supports JOG drive (MPG is not support). To set the external DI signal to trigger JOG function, please refer to the Motion IO section in Axis Pages.

Axis Setup

Click the button to check/set the axis's attributes and I/O as follows:

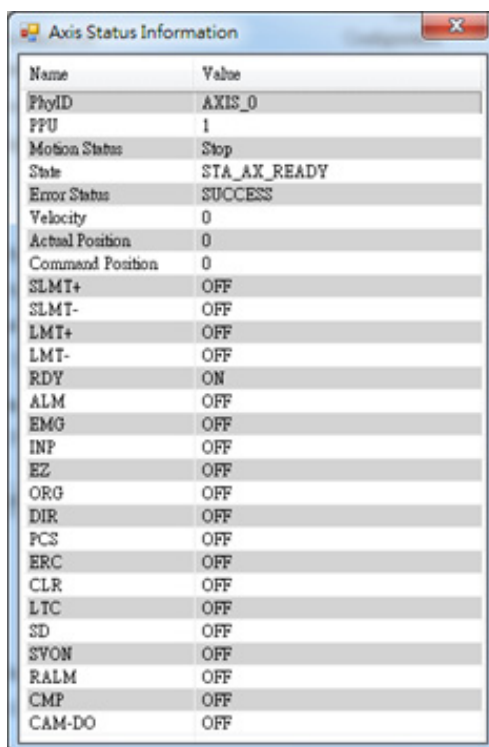


The left tree view shows the classification of axis's properties, when you click the corresponding item, the right side, Data View, will list the properties and corresponding property values in the category. For details, refer to the description about Feature, Configuration and Parameter of axis which are listed in property list of Programming guide.

- Note!**
- In the utility, if no corresponding functions of the selected device, the item will not be shown in the left side Tree View. For example, if the selected device is PCI-1245/1265, and this board does not support slow down (SD) and vibration suppression function, then, you will not see the items in the Tree View. At the same time, because single axis dialog has speed parameter setting, the speed pattern item will not be shown.
 - When "Pulse Out" category is selected, there will be illustration of corresponding mode below the description of "Pulse Out Mode" property.

After editing, the property value will become effective (already set in the device) after the mouse leaves the edit box. To duplicate the attributes to other axes, check the axis list then click **[Copy Config]** to apply the setting to the selected axis.

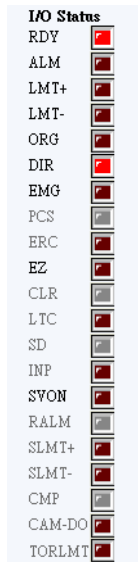
Axis Status



Name	Value
PhyID	AXIS_0
PPU	1
Motion Status	Stop
State	STA_AX_READY
Error Status	SUCCESS
Velocity	0
Actual Position	0
Command Position	0
SLMT+	OFF
SLMT-	OFF
LMT+	OFF
LMT-	OFF
RDY	ON
ALM	OFF
EMG	OFF
INP	OFF
EZ	OFF
ORG	OFF
DIR	OFF
PCS	OFF
ERC	OFF
CLR	OFF
LTC	OFF
SD	OFF
SVON	OFF
RALM	OFF
CMP	OFF
CAM-DO	OFF

Click the button; users can view the assigned axis information. For example, PhyID, PPU, basic status (Motion Status, State, Error Status etc.) and I/O status (Alarm, SLMTP/N and etc.).

4.3.1.6 IO Status



I/O Status	LED Bar
RDY	<input checked="" type="checkbox"/>
ALM	<input checked="" type="checkbox"/>
LMT+	<input checked="" type="checkbox"/>
LMT-	<input checked="" type="checkbox"/>
ORG	<input checked="" type="checkbox"/>
DIR	<input checked="" type="checkbox"/>
EMG	<input checked="" type="checkbox"/>
PCS	<input type="checkbox"/>
ERC	<input checked="" type="checkbox"/>
EZ	<input checked="" type="checkbox"/>
CLR	<input type="checkbox"/>
LTC	<input checked="" type="checkbox"/>
SD	<input type="checkbox"/>
INP	<input checked="" type="checkbox"/>
SVON	<input checked="" type="checkbox"/>
RALM	<input type="checkbox"/>
SLMT+	<input checked="" type="checkbox"/>
SLMT-	<input checked="" type="checkbox"/>
CMP	<input type="checkbox"/>
CAM-DO	<input checked="" type="checkbox"/>
TORLMT	<input checked="" type="checkbox"/>

Users can check the I/O status from the LED bar. Wherein, indicates the device does not support the function or does not have the corresponding I/O; indicates the device support the function, but I/O is not triggered (OFF); indicates the corresponding I/O is triggered (ON).

For details, refer to the description about Status in Acm_AxGetMotionIO function which is listed in Common API of Programming guide.

If no functional or no corresponding item, the text will be displayed in gray color. If the board supports the function, but not enable, the text is also displayed as gray. If the board supports this function and enable this function, the test will be display as normal.

4.3.1.7 Error Status

Last Error Status

Error Code: Error Message:

Users can check the latest error code and error message. If there are no errors, the error code is “0”, error message is “SUCCESS”.

4.3.2 Multi-Axis Motion

The screenshot shows a multi-axis motion control interface with several sections:

- Operate Axes:** A list of axes with checkboxes. Callout 1 points to the list.
- Motion Params Set:** Fields for VelLow, VelHigh, Acc., and Dec., along with Speed Pattern options (Trapezia, S-curve). Callout 2 points to the Speed Pattern options.
- Motion Operation:** Basic Interpolation Motion settings including Movement Mode (Absolute, Relative) and Interpolation Mode (Line, Arc, Helix). Callout 4 points to the Interpolation Mode options.
- Path Motion:** Settings for Speed Forward and Blend Time. Callout 5 points to the Blend Time field.
- Path Status:** Fields for CurIndex, CurCmd, Path Count, Remain, and FreeCnt. Callout 6 points to the CurCmd field.
- Position:** A table showing Command and Feedback for 0-Axis, 1-Axis, 2-Axis, and 3-Axis. Callout 7 points to the 2-Axis Command field.
- Group State:** A dropdown menu showing 'Ready'. Callout 8 points to the Group State dropdown.
- Last Error Status:** A section at the bottom showing Error Code and Error Message.

4.3.2.1 Operate Axes

The Checked List in the form will list all axes of the selected device, check the Check-box of corresponding axis, you can add the axis into the Group. When the number of axis added to the Group is less than 1, Group's State will be “Disable”. When the number of axis added to the Group is greater than or equal to 2, Group's State will be “Ready”. After configuring appropriate parameters, users can do appropriate interpolation operation.

Click **[SVON]** / **[SVOFF]** to servo on / off all the servos in Group.

4.3.2.2 Set Motion Parameters

The parameter set includes Group VelLow, Group VelHigh, Group Acc, Group Dec and Speed Pattern. Click **[Set Parameters]** to apply the settings.

4.3.2.3 Set Target Position

Configure motion's center / end as follows. The dialog will automatically enable the edit box writable by referring to group axis and interpolation mode.

Axis	Line End (PPU)	Arc Center (PPU)	Arc End (PPU)
0-Axis	8000	8000	16000
1-Axis	8000	0	0
2-Axis	8000	8000	16000

As in the figure, 1-axis and 2-axis are added to Group. If Line interpolation mode is selected, the writable edit boxes are the "Line End (PPU)" column of "1-axis" and "2-axis", whose background color is white. The edit boxes with gray background color are not editable.

4.3.2.4 Motion Operation

Basic Interpolation Motion Mode

Basic interpolation motion includes linear interpolation (Line), circular interpolation (Arc) and helical interpolation (Helix) as follows.

■ Movement Mode

Absolute: The interpolation motion will directly use the set position parameters.

Relative: The interpolation motion will add initial offset to the position parameters and then use it.

■ Arc Direction

CW: Clockwise.

CCW: Counter Clockwise.

■ Interpolation Mode

Line: linear interpolation

Arc: circular interpolation

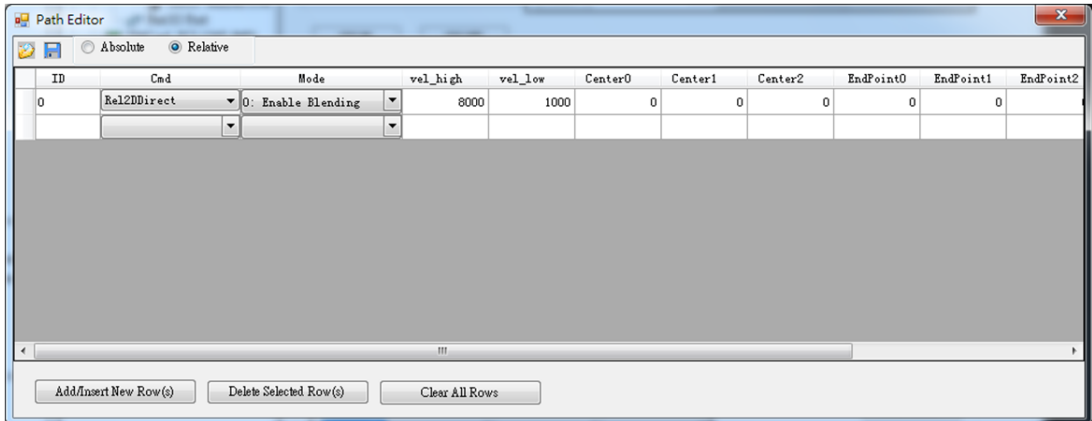
Helix: helical interpolation

After corresponding configuration, click [Move] button, Group will do the specified interpolation. While Group is in interpolation motion, click [Stop] button to stop the motion.

Path Motion

■ Edit Path

Click [Edit Path], the following form will be shown up:



– Open File

Open selected Path file from appropriate file path, which can be a binary file (.bin) or a comma-separated value file (.CSV).

– Save File

Save the edited data to a Path file which can be a binary file (.bin) or a comma-separated value file (.CSV). CSV files can be opened in Excel, so can be checked / modified conveniently later. But if you want to run the Path through [Load Path], you need to save the data as .bin format, because currently device only supports .bin file to import through [Load Path].

– Movement mode Absolute Relative

Absolute or Relative. If you select “Absolute”, then commands listed in the Cmd column will be the commands related to absolute motion; similarly, if you choose “Relative”, then commands listed in the Cmd column will be the commands related to relative motion.

– Path edit items

ID	Cmd	Mode	vel_high	vel_low	Center0	Center1	Center2	EndPoint0	EndPoint1
0	Rel2DDirect	0: Enable Blending	8000	1000	0	0	0	0	0
1	Rel2DLine	0: Enable Blending	8000	1000	0	0	0	0	0
2	EndPath	0: Enable Blending	8000	1000	0	0	0	0	0

Include command (Cmd), Mode (Blending/No Blending), movement velocity (vel_high), initial velocity (vel_low), center (Center), and end (EndPoint). Wherein, there are three axes (Center0/Center1/Center2) in circle interpolation by default, the number of EndPoint is according to the maximum number of axis supported in the selected device interpolation, such as in PCI-1245, the maximum number of axis supported is 4 in Direct interpolation motion, so the end point will be EndPoint0, EndPoint1, EndPoint2, and EndPoint3.

– Add/Insert New Row(s)

After clicking, the following dialog will be shown. You can edit the number of rows to be added/inserted. Click [OK], corresponding number of rows will be added after or inserted into the selected row.

– Delete Selected Row(s)

Delete the selected rows.

– Clear All Rows

Clear all lines.

■ Load Path

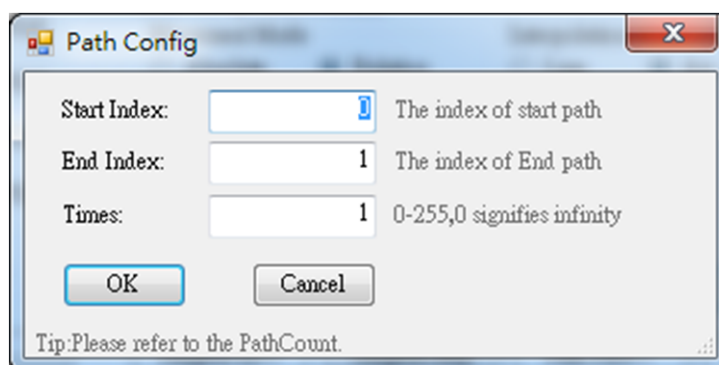
Click [**Load Path**], if group's state is "Ready", you can import selected Path file (. bin format) from the Open File dialog box to the Device.

■ Move Path

After loading Path, if the edited Path is not wrong, The Path will be run one by one according to the serial number. You can observe movement curve and corresponding state from [**Path Status**], [Path Plot] and [**Speed Chart**].

■ Move Sel Path

Does continuous interpolation movement with path(s) selected from loaded Path file. After loading Path, click [Move Sel Path] to activate the dialog:



1. Start Index: select the starting serial number of Path
2. End Index: select the end serial number of Path
3. Times: executable times. The value is 0 to 255. If you set 0, it means an infinite loop until you click "Stop" to terminate the loop.

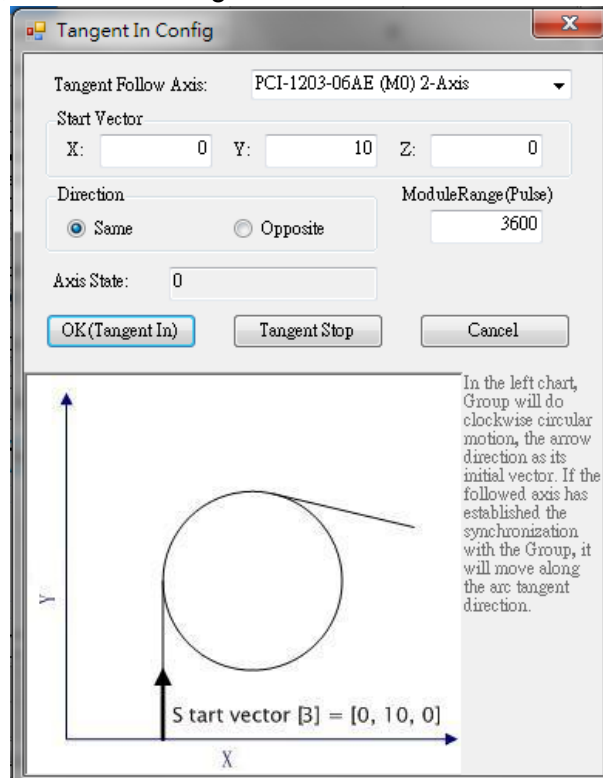
■ Speed Forward and Blending Time

If the value is "Checked", the Group's speed-forward function will be enabled. For detail, refer to the description about CFG_GpSFEnable in Group which is listed in property list of Programming guide.

For details about blending time, refer to the description about CFG_GpBlidTime in Group which is listed in property list of Programming guide.

■ Tangent In and Tangent Stop

Click [**Tangent In**], the follow dialog will be shown.



The following parameters need to be configured:

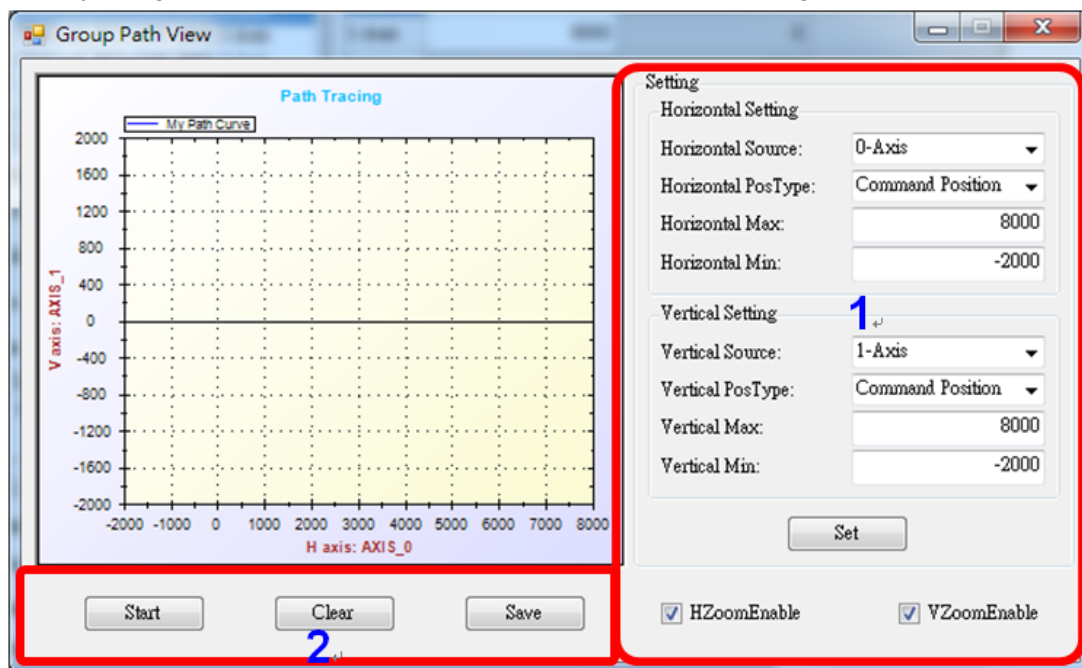
1. **Tangent Follow Axis:** Select tangent follow axis. As the axis cannot be axes added in Group, so the axis listed in the combo box does not includes axes added in Group.
2. **Start Vector:** the start vector of tangent follow motion. In Utility, the default reference plane is X-Y, you only need to set X vector and Y vector. Z axis is not necessary to edit.
3. **Direction:** The direction of tangent follow axis in motion, which can be the same as or opposite to the direction of Group's motion.
4. **ModuleRange:** The module range of tangent follow axis, that is, the pulse number of tangent follow axis's one revolution (360 degrees)

There are related diagram and description below the configuration. Click [**OK (Tangent In)**], the tangent follow axis will establish tangent follow synchronization with the Group. After that, if the Group does interpolation motion, the following axis will move along the tangent direction of the interpolation motion. If the tangent follow axis has established tangent follow synchronization with the Group, click [**Tangent In**] again, the value of parameters in the form will be the configured value, and you can click [**Tangent Stop**] to dissolve the synchronization relationship. Click [**Cancel**], nothing will do but close the form.

Click [**Tangent Stop**] to dissolve the synchronization relationship between tangent follow axis and the Group.

■ Path Plot

Display the group motion curve. Click [**Path Plot**], the following window will appear:



1. Setting

Set the horizontal and vertical coordinates.

- Source: Horizontal data source, 1st axis of group (sorted by the order being added) by default. You can choose any axis in group.
- PosType: Horizontal position type, you can choose command or feedback position.
- Max: Horizontal maximum coordinate.
- Min: Horizontal minimum coordinate.

Click [**Set**] to activate the effectiveness.

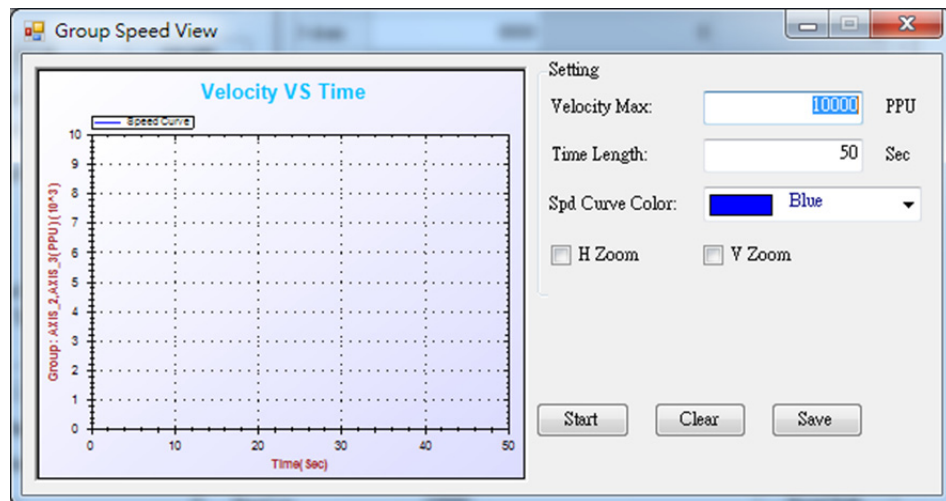
2. Operation

Click [**Start**], the graphic box will be ready to draw the curve. If Group is in motion, you can see the trajectory. After clicked, the text on [**Start**] button will change into "Stop"; click [**Stop**], drawing the curve will be stopped and the text will back to "Start".

Click [**Clear**], the current curve in graphic box will be cleared.

Click [**Save**], the specified path curve will be save as .png, .gif, .jpg, .tif or.bmp format.

■ Speed Chart



The setup and operation are similar to [Speed Chart] in “Single Axis Motion”.

4.3.2.5 Path Status

Path Status			
CurIndex:	<input type="text" value="0"/>	CurCmd:	<input type="text" value="EndPath"/>
Remain:	<input type="text" value="0"/>	FreeCnt:	<input type="text" value="7000"/>
			<input type="button" value="Reset Path"/>

CurIndex: The serial number of path currently running.

CurCmd: The command code of path currently running.

Remain: Unexecuted path number

FreeCnt: The remain space of Path Buffer

Path Count: The total path number in loaded Path file.

4.3.2.6 Axis Position

Position			
Position	0-Axis	1-Axis	2-Axis
Command	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>
Feedback	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>

Display the current command and feedback position for all axes of device. Click [Reset Counter] to reset to 0.

4.3.2.7 Group State and Error Status

Group State:	Ready
Last Error Status	
Error Code:	0
Error Message:	Success.

Group State: Show the current Group's State. For detail, refer to the description about State in Acm_GpGetState function which is listed in Common API of Programming guide.

Last Error Status: Display the latest error message:

Axis Name: The axis which has error.

Error Code: The error code.

Error Message: The specific error message.

4.3.3 Synchronized Motion

The screenshot displays the configuration interface for synchronized motion, divided into two main sections: Slave Axis Operation and Master Axis Operation.

- Slave Axis Operation:**
 - Synchronized Mode:** Radio buttons for CAM (selected), Gear, and Gantry. A red box highlights this section with a blue '1'.
 - CAM Motion Configuration:** Includes CAMTable ID, Camming Type, Master/Slave Movement Mode (Absolute/Relative), Master/Slave Offset, Master/Slave Scaling, and Reference Source (Command Pos/Feedback Pos). A red box highlights this section with a blue '2'.
 - Gear Motion Configuration:** Includes Numerator, Denominator, Reference Source, and Movement Mode (Absolute/Relative). A red box highlights this section with a blue '3'.
 - Gantry Motion Configuration:** Includes Reference Source, Direction (Same/Opposite), and Max Diff. A red box highlights this section with a blue '4'.
- Master Axis Operation:**
 - Motion Params Set:** Fields for Distance, VelLow, VelHigh, Acc., and Dec. with units (PPU, PPU/S, PPU/S²). A red box highlights this section with a blue '5'.
 - Speed Pattern:** Radio buttons for Trapezia (selected) and S-curve.
 - Motion Mode:** Radio buttons for P to P (selected) and Continue.
 - Position:** Fields for Master Axis and Slave Axis Command and Feedback.
 - Current Axis State:** Status indicators for Master and Slave axes.

At the bottom, a **Last Error Status** section shows Axis Name: AXIS_0, Error Code: 0, and Error Message: Success.

4.3.3.1 Select Axis and Synchronized Mode

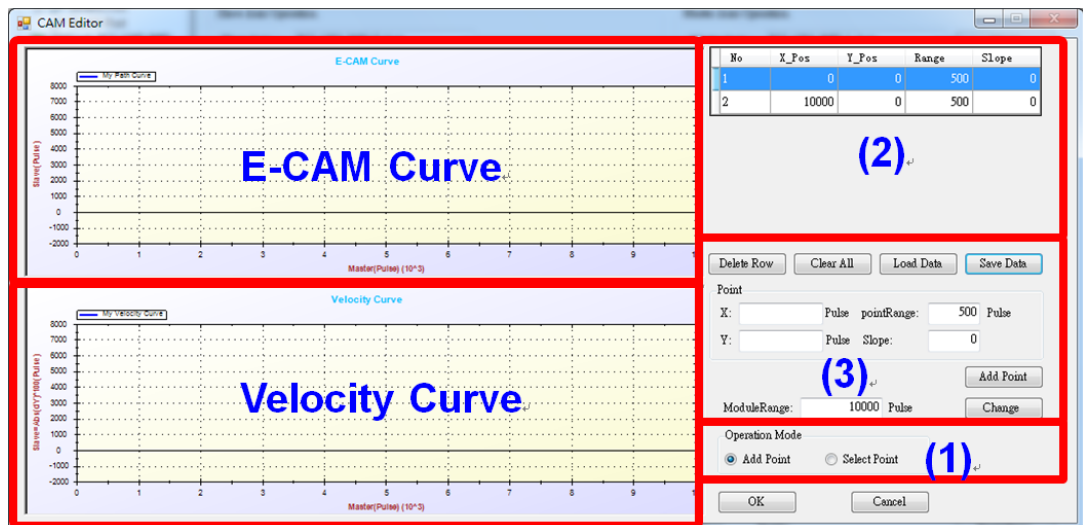
Select any one of Device's axes to be slave axis. Master axis and slave axis cannot be the same one. The default slave axis is 0-axis of the selected device.

Select the synchronized mode: CAM, Gear or Gantry. Users must select synchronized mode first before the configuration and establishment of synchronized motion.

4.3.3.2 CAM Motion

CAM Editor

Click [CAM Editor], the following dialog will show up:



Left upper corner is E-CAM curve graphic box; left lower corner is Velocity curve graphic box; right upper corner is CAM Table; right lower corner is operation panel.

1. Operation Panel

- Add Point: you can directly add CAM points on the E-CAM Curve. Whenever you add one point, CAM Curve will be redrawn. Wherein, CAM Point is expressed with a small red solid circle and CAM Curve is expressed with blue curve. In this operation mode, the shape of the mouse is cross. When the form is opened first time or the CAM curve has not been edited, the operation mode is "Add Point" mode by default.
- Select Point: you can select the corresponding CAM Point to drag and drop. At the same time, CAM Curve will be changed accordingly. In this operation mode, the shape of the mouse is arrow. When the form is opened again or the CAM curve has been edited, the default operation mode is "Select Point" mode.

2. CAM Table

The CAM Table formed by edited CAM Point is shown on the top right. It is noteworthy that, the X_Pos and Y_Pos of the first row of CAM Table, that is the first CAM Point, must be zero, which means the Master axis's starting position is 0, the Slave axis's starting position is 0; The X_Pos and Y_Pos of CAM Table's last line, that is the last CAM Point, must be (ModuleRange, 0), which means the Master axis rotates a circle and the Slave axis backs to the starting position 0.

- No: CAM Point 's serial number.
- X_Pos: Horizontal position coordinate (Master axis's position)
- Y_Pos: Vertical position coordinate (Slave axis's position)
- Range: The distance between reference point and CAM Point. For detail, refer to the description about PointRange in Acm_DevDownloadCAMTable function which is listed in Common API of Programming guide. The default value is the edited value of pointRange. Users can change the value by editing it. When you add more CAM Points, the pointRange will be also changed. If you want to change pointRange of edited CAM Point, directly modify it in CAM Table.

- e. Slope: The slope between two reference points of CAM Point. For detail, refer to the description about PointSlope in Acm_DevDownloadCAMTable function which is listed in Common API of Programming guide. The default value is the value in Slope editing box, which can be modified by editing the value in the edit box, the slope of following added CAM Point will be the modified value in the edit box. If you want to change the Slope of edited CAM Point, directly edit in CAM Table.

Note! *The range of Slope value is from -10 to 10. If the value is less than -10, it will be -10 by default. And if the value is larger than 10, it will be 10 by default.*



3. CAM Table Operation

- a. Delete Row: Delete the selected row(s).
- b. Clear All: Clear all CAM Points (except starting point and final point)
- c. Load Data: Insert the selected CAM Table file. The file format can be binary (.bin) or .csv readable by EXCEL.
- d. Save Data: Save the CAM Table. The file format can be binary (.bin) or .csv readable by EXCEL. But if you want to import CAMTable through [Load CAM-Table File] operation, users need to save the CAM Table as .bin format, because currently the device only support .bin file to import through [Load CAMTable File].
- e. Add Point: To add CAM Point, users can also edit X_Pos, Y_Pos, pointRange and Slope on the lower right and click [Add Point] to add it.
- f. Change (ModuleRange): The master axis's revolution pulse (ModuleRange) is set to be 10,000 pulses by default. If you want to edit, you can edit in ModuleRange box, and then click [Change] to finish. After modified, the horizontal maximum ordinate of E-CAM Curve and Velocity Curve will be the modified value; if there are edited CAM Points before change the value, the X_Pos and pointRange of the CAM Points will become ModuleRange (after modified)/ Pre_ModuleRange (before modified) fold.

Click [OK] to save the CAM Table. Then click [Download CAMTable] button in Synchronized Motion Page to download the CAM Table into hardware. Click [Cancel] to give up the editing.

Load CAMTable File

By clicking [Load CAMTable File] to choose binary file, the CAM Table will be saved in the hard drive.

Note! *Before you save, you should set up the "CAMTableID" first. The value is 0 or 1. After you execute this step, the CAMTableID cannot be changed before you dissolve the synchronization relation.*



CAMTable Configuration

Configure cam motion and establish cam synchronization.

The screenshot shows a 'CAM Motion Configuration' dialog box with the following settings:

- CAMTable ID: 0
- Camming Type: Non periodic
- Master Movement Mode: Absolute, Relative
- Slave Movement Mode: Absolute, Relative
- MasterOffset: 0
- SlaveOffset: 0
- MasterScaling: 1
- SlaveScaling: 1
- Reference Source: Command Pos, Feedback Pos
- Buttons: Download CAMTable, CAM In, Stop

Before the establishment of cam synchronization, users need to configure the following parameters:

1. Camming Type:
 - a. Non periodic: non-circular pattern. If you select this mode, after the Master axis runs a complete cycle, the Slave axis will no longer follow the Master axis according to CAM curve.
 - b. Periodic: circular pattern. If you select this mode, the Slave axis will always follow the Master axis according to CAM curve in cam motion.
2. Master Movement Mode:
 - a. Absolute: If you select this mode, the current position of the Master axis will be served by CAM curve as the starting point of horizontal coordinate.
 - b. Relative: If you select this mode, the Master axis will serve the current Command/Actual Position as a starting point to move in relative mode.
3. Slave Movement Mode:
 - a. Absolute: If you choose this mode, the Slave axis will chase after the Y_Pos value in CAM Table with set speed from the current Command/Actual Position.
 - b. Relative: If you choose this mode, the Slave axis will move in relative mode with the current Command/Actual Position according to CAM curve.
4. Master Offset:

Offset value relative to the Master axis.
5. Slave Offset:

Offset value relative to Slave axis.
6. Master Scaling:

Master axis ratio factor. CAM Curve zoom in/out in the horizontal direction.
7. Slave Scaling:

Slave axis ratio factor. CAM Curve zoom in/out in the vertical direction.
8. Reference Source:

Master axis's position reference source.

 - a. Command Position: reference source is command position.
 - b. Feedback Position: reference source is feedback (actual) position.

Click [**CAM In**], the Slave axis will establish CAM synchronization with the Master axis and the Slave's state will change into "Synchronous Driving". Thereafter, if the master axis is in P to P or Continue motion, the slave axis will follow the Master axis to Move with the CAM Curve and the configuration accordingly.

Click [**Stop**] to dissolve the synchronization relation. The slave axis's state will be ready.

4.3.3.3 Gear Motion

Before the establishment of gear synchronization, users need to configure the following parameters:

1. Numerator: The numerator of gear ratio
2. Denominator: The denominator of gear ratio
3. Reference Source: The Master axis's position reference source, command position or feedback position.
4. Movement Mode:
 - Absolute: If you select this mode, the Slave axis will chase after the Command/Actual Position of the Master axis with set speed.
 - Relative: If you select the mode, the Slave axis will keep initial position difference with the Master axis.

Click [**Gear In**], the Slave axis will establish Gear synchronization with the Master axis and the Slave's state will change into "Synchronous Driving". Thereafter, if master axis is in P to P or Continue motion, the slave axis will follow master axis to move with the configuration accordingly.

Click [**Stop**] to dissolve the synchronization and the slave axis's state will be "Ready".

4.3.3.4 Gantry Motion

Select Gantry in Synchronized Mode, you can configure and operate the gantry movement.

Before the establishment of gantry synchronization, you need to configure the following parameters:

1. Reference Source: the Master axis's position reference source, command position or feedback position.
2. Direction: The Slave axis direction relative to the Master axis.
 - a. Same: Same as the Master axis.
 - b. Opposite: Opposite to the Master axis.

Click [**Gantry In**] to the Slave axis will establish gantry synchronization with the Master axis and the Slave's state will change into "Synchronous Driving". Thereafter, if the Master axis is in P to P or Continue motion, the Slave axis will follow the Master axis to move with the configuration accordingly.

Click [**Stop**] to dissolve the gantry synchronization and Slave axis's state will be "Ready".

4.3.3.5 Master Axis and Synchronized Motion Operation

The screenshot shows a software interface for controlling a master axis in a synchronized motion system. The interface is organized into several functional areas:

- Master Axis:** A dropdown menu is set to "PCI-1203-06AE (M0) 1-Axis".
- Motion Params Set:** A table of parameters for motion control:

Distance:	10000	PPU
VelLow:	2000	PPU/S
VelHigh:	8000	PPU/S
Acc.:	10000	PPU/S ²
Dec.:	10000	PPU/S ²
- Speed Pattern:** Radio buttons for "Trapezia" (selected) and "S-curve".
- Motion Mode:** Radio buttons for "P to P" (selected) and "Continue".
- Position:** Input fields for "Command" and "Feedback" for both "Master Axis" and "Slave Axis", all showing "0".
- Current Axis State:** Status indicators for "Master" and "Slave", both showing "Ready".

Control buttons on the right side include "SVON", "<--", "-->", "Stop", "Path Plot>>", "Reset Counter", and "Reset Error".

After finishing the setup of slave axis, users can use the master operation panel to test the synchronized motion. The operation is similar to the single-axis motion. Select the master axis and set the motion parameters in "Motion Params Set" panel. Click [**Set Parameters**] button to apply the settings. Click [**SVON**] button to servo on the axis and click [**<-**] or [**->**] button to start the motion. While the master starting to move, the slave axis will move automatically according to the synchronized mode settings.

Users can view the command / feedback position in position panel and the current state of the Master axis and the Slave axis through the status bar. For more detailed about the axis state, please refer to the description about State in Acm_AxGetState function which is listed in Common API of Programming guide.

4.3.4 DO / DI / AO / AI

4.3.4.1 Local I/O

D-Sub Pin Assignment			
Pin	Description	Pin	Description
1	DO 3	9	DI 5
2	DI 2	10	DI 7
3	DI 4	11	DO 2
4	DI 6	12	DO 4
5	DI 8	13	External power GND
6	DO 1	14	External power GND
7	DI 1	15	External power (+24V)
8	DI 3		

This page shows the status of local DIO in D-Sub 15-pin connector. or indicates that the IO status is in effect (ON) and the value of the bit is 1; or indicates that the IO status is not in effect (OFF) and the value of the bit is 0. Hex indicates the hexadecimal value of the byte composed by 8 IOs. User can turn on/off the digital output channel by click the or buttons.

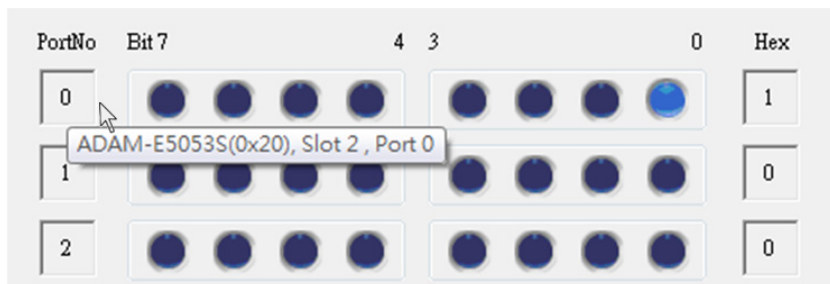
This page also shows the pin assignment of the D-Sub 15-pin connector. Please note that the pin assignment definition is start from 1 and the bit definition is start from 0. For example, the DO bit 0 is DO 1 in pin assignment and DO bit 1 is DO 2 in pin assignment respectively. There's 4 digital output (DO 1 - DO 4) in PCI-1203, so the highest bit number of DO is bit 3.

4.3.4.2 Digital Output

Shows the status of all the Digital Outputs connect to PCI-1203. As the above figure, Bit 7 to 0 from right to left are digital outputs respectively. Wherein, indicates that the DO is in effect (ON) and the value of the bit is 1; indicates that the DO is not in effect (OFF) and the value of the bit is 0. Hex indicates the hexadecimal value of the byte composed by 8 DOs.

PortNo is the port number. To get the actual DO position, just move mouse to the portNo. A tip will show the actual position of this port. In this example, port 0 is the port 0 on the ADAM-E5056S/SO module, slave ID 0x20, slot 0.

4.3.4.3 Digital Input



Shows the status of all the Digital Inputs connect to PCI-1203. As the above figure, Bit 7 to 0 from right to left are digital inputs respectively. Wherein, ● indicates that the DI is in effect (ON) and the value of the bit is 1; ● indicates that the DI is not in effect (OFF) and the value of the bit is 0. Hex indicates the hexadecimal value of the byte composed by 8 DIs.

PortNo is the port number. To get the actual DI position, move your mouse to the portNo, a tip will show the actual position of this port. In this example, port 0 is the port 0 on the ADAM-E5053S module, slave ID 0x20, slot 2.

4.3.4.4 Analog Output

Channel	Value	Range	Raw
CH-0	0.000	-10 ~ 10V	0x0
CH-1	0.000	-10 ~ 10V	0x0
CH-2	0.000	-10 ~ 10V	0x0
CH-3	0.000	-10 ~ 10V	0x0

Shows the status of all the Analog Outputs connect to PCI-1203. It might take time to scan the AO settings from each slave for the first time you open this page. The value is the analog value of this channel and its unit may be mA, mV or V which is depended on the AO range. Raw is the 16-bit raw value of this channel. In this example, channel 0 is on the ADAM-E5024H module, slave ID 0x20, slot 1 and its value is 0V.

4.3.4.5 Analog Input

Channel	Value	Range	Raw
CH-0	0.007	-10 ~ 10V	0x801
CH-1	0.007	-10 ~ 10V	0x801
CH-2	*****		*****
CH-3	*****		*****

Shows the status of all the Analog Inputs connect to PCI-1203. It might take time to scan the AI settings from each slave for the first time you open this page. The value is the analog value of this channel and its unit may be mA, mV or V which is depended on the AI range. Raw is the 16-bit raw value of this channel. In this example, channel 0 is on the ADAM-E5017UH module, slave ID 0x20, slot 1 and its value is 0.007V.

4.3.5 IO Mapping Table

This page shows the physical and logical index mapping table.

DI	DO	AI	AO	Port	Ring	ID	Slot	Description	Edit
				0	0	0x20	2	Port 0 [ADAM-E5053S]	Select
				1	0	0x20	2	Port 1 [ADAM-E5053S]	Select
				2	0	0x20	2	Port 2 [ADAM-E5053S]	Select
				3	0	0x20	2	Port 3 [ADAM-E5053S]	Select
				4	0	0x200	1	Port 0 [ADAM-E5051S]	Select
				5	0	0x200	1	Port 1 [ADAM-E5051S]	Select
				6	0	0x0		Port 0 [MADHT1505BA1]	Select
				7	0	0x0		Port 1 [MADHT1505BA1]	Select
				8	0	0x0		Port 2 [MADHT1505BA1]	Select
				9	0	0x0		Port 3 [MADHT1505BA1]	Select
				10	0	0x102		Port 0 [MADHT1505BA1]	Select
				11	0	0x102		Port 1 [MADHT1505BA1]	Select
				12	0	0x102		Port 2 [MADHT1505BA1]	Select
				13	0	0x102		Port 3 [MADHT1505BA1]	Select
				14	0	0x103		Port 0 [MADHT1505BA1]	Select
				15	0	0x103		Port 1 [MADHT1505BA1]	Select
				16	0	0x103		Port 2 [MADHT1505BA1]	Select
				17	0	0x103		Port 3 [MADHT1505BA1]	Select

Find
 Port
 Slave ID Start Offset :
 Edit Port (0 ~ 19) :

Filename: Default Mapping Settings

For Common Motion API, users should read / write value from / to slaves by a specified number. We called the specified number as logical index or logical port / channel. In this page, users can customize the logical index and export the customized mapping setting to file. Then after import the mapping file to Common Motion API, users can operate these ports and channels by the logical index.

4.3.5.1 How to Read Mapping Table

DI	DO	AI	AO	Port	Ring	ID	Slot	Description	Edit
				0	0	0x20	2	Port 1 [ADAM-E5053S]	Select
				1	0	0x20	2	Port 0 [ADAM-E5053S]	Select
				2	0	0x20	2	Port 2 [ADAM-E5053S]	Select
				3	0	0x20	2	Port 3 [ADAM-E5053S]	Select
				4	0	0x200	1	Port 0 [ADAM-E5051S]	Select

In the above picture, a DI mapping table is shown as an example. The first column, "Port", is the logical port number. The "Ring", "ID", "Slot" and "Description" column describe the actual position of this port. See the first row in this table, the actual position of logical port 0 is the port 1 of ring 0, slave ID 0x20, slot 2, which is belong to an ADAM-E5053S module.

4.3.5.2 Edit Mapping Table

Users can customize the logical port by edit the mapping table. There are two rules to edit the mapping table: First, you have to give all the actual port / channel a unique logical index. It is not allowed if an unmapped port /channel exists. Two actual port share the same logical index is not allowed too. Notice that the mapping table is individual in DI/DO/AI/AO, so DI, DO, AI and AO could have their own logical port 0 without conflict. Second, the logical index should be continuous.

In the following section, we will show how to customize the mapping table.

Edit Logical Index

Users can double click the logical index to edit it. If this index exists, the conflicted item will be highlighted and moved to the bottom of the table.

See the following example.

Double click port number to start edit.

Port	Ring	ID	Slot	Description	Edit
0	0	0x20	2	Port 0 [ADAM-E5053S]	Select
1	0	0x20	2	Port 1 [ADAM-E5053S]	Select
2	0	0x20	2	Port 2 [ADAM-E5053S]	Select
3	0	0x20	2	Port 3 [ADAM-E5053S]	Select

Set this port to 1 than press

Conflict!

Port	Ring	ID	Slot	Description	Edit
1	0	0x20	2	Port 0 [ADAM-E5053S]	Select
0	0	0x20	2	Port 1 [ADAM-E5053S]	Select
2	0	0x20	2	Port 2 [ADAM-E5053S]	Select
3	0	0x20	2	Port 3 [ADAM-E5053S]	Select

The first row now is port 1.

The conflicted item will be moved to bottom.

Port	Ring	ID	Slot	Description	Edit
1	0	0x20	2	Port 0 [ADAM-E5053S]	Select
0	0	0x20	2	Port 1 [ADAM-E5053S]	Select
2	0	0x20	2	Port 2 [ADAM-E5053S]	Select

ADAM-E5053S, Slave ID 0x20, Slot 2, Port 1

Edit Actual Position

Users can double click the **[Select]** button to change the actual position of a logical port. A selection tool will pop out after users click **[Select]** button. If the candidate port is already mapped to a logical index, the conflicted item will be highlighted and moved to the bottom of the table.

See the following example.

1. Click the select button.

2. Double click to select

Conflict!

3. Item changed

4. The conflicted item will be moved to bottom.

Port	Ring	ID	Slot	Description	Edit
0	0	0x20	2	Port 1 [ADAM-E5053S]	Select
1	0	0x20	2	Port 0 [ADAM-E5053S]	Select
2	0	0x20	2	Port 2 [ADAM-E5053S]	Select
3	0	0x20	2	Port 3 [ADAM-E5053S]	Select

Mapping Unmapped Item

See the following example.

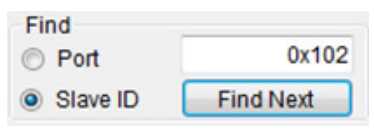
1. Click here to select this row.

2. Click ↑

3. The unmapped item has moved to the selected row.

Port	Ring	ID	Slot	Description	Edit
0	0	0x20	2	Port 0 [ADAM-E5053S]	Select
1	0	0x20	2	Port 2 [ADAM-E5053S]	Select
2	0	0x20	2	Port 3 [ADAM-E5053S]	Select
3	0	0x20	2	Port 3 [ADAM-E5053S]	Select

4.3.5.3 Find Data from Mapping Table



Users can use the **[Find]** panel to search the target item. Type the slave ID (Including "0x") or the logic index in the textbox and click **[Find Next]** button to search.

4.3.5.4 Start Offset

Channel	Ring	ID	Slot	Description	Edit
11	0	0x20	3	Channel 0 [ADAM-E5024H]	Select
12	0	0x20	3	Channel 1 [ADAM-E5024H]	Select
13	0	0x20	3	Channel 2 [ADAM-E5024H]	Select
14	0	0x20	3	Channel 3 [ADAM-E5024H]	Select
15	0	0x200	2	Channel 0 [ADAM-E5024]	Select
16	0	0x200	2	Channel 1 [ADAM-E5024]	Select
17	0	0x200	2	Channel 2 [ADAM-E5024]	Select
18	0	0x200	2	Channel 3 [ADAM-E5024]	Select

Find
 Channel
 Slave ID

Start Offset :

The mapping index should be continuous but the index range could be customized. For example, there are total 8 AO channels in the EtherCAT network and the start offset is 11. The valid logical channel range will be 11 ~ 18.

Enter the start offset you want and click **[Set]** button to apply the setting.

4.3.5.5 Save and Load

Filename: Default Mapping Settings

Click **[Save]** or **[Save As]** button to save the mapping table. Click **[Load]** button to load mapping table to Utility. The mapping table will apply to Utility automatically if the table is valid.


The **[Filename]** shows the current mapping table applied to Utility. If the user opened Utility and didn't load or edit the mapping table, the Filename would be "Default Mapping Settings" which is the system default mapping settings.

Click **[Undo all changes]** to discard all changes and reload mapping table.

Click **[Export CSV]** to export the mapping table to .CSV file. Users can use Excel or any other tool which support .CSV format to check or print the mapping table.

4.3.6 Information

Device Information

Device Name	PCI-1203-06AE (M0)	
Description	Advantech EtherCAT Master Card	
Firmware Version	1.0.0.9.0	
FPGA Version	a.0.0.2.β	
Kernel Driver Version	1.0.0.4	
User Driver Version	2.0.0.5	
System Log	<input type="checkbox"/> Enable	

Master (R0) Topology


Master Name	Motion Ring	
Cycle Time	500μs	
Connected Slave Count	5	
Description	Support EtherCAT servo motor and ADAM-5000/ECAT.	

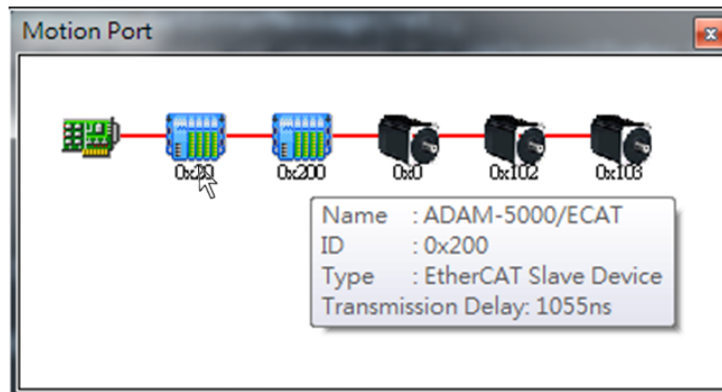
Master (R1) Topology

Master Name	Fast IO Ring	
Cycle Time	200μs	
Connected Slave Count	0	
Description	Support ADAM-5000/ECAT. (Servo motor not support)	

This page shows the card information and master information. If there's no slave connected to the PCI-1203, this page will be the only page users can operate. Users can check the firmware and driver version in this page.

Note that the PCI-1203 supports 2 EtherCAT rings: Motion Ring and Fast IO Ring. The fast IO ring doesn't support servo motor. The servo motor may be ignored if the user connects it to the PCI-1203.

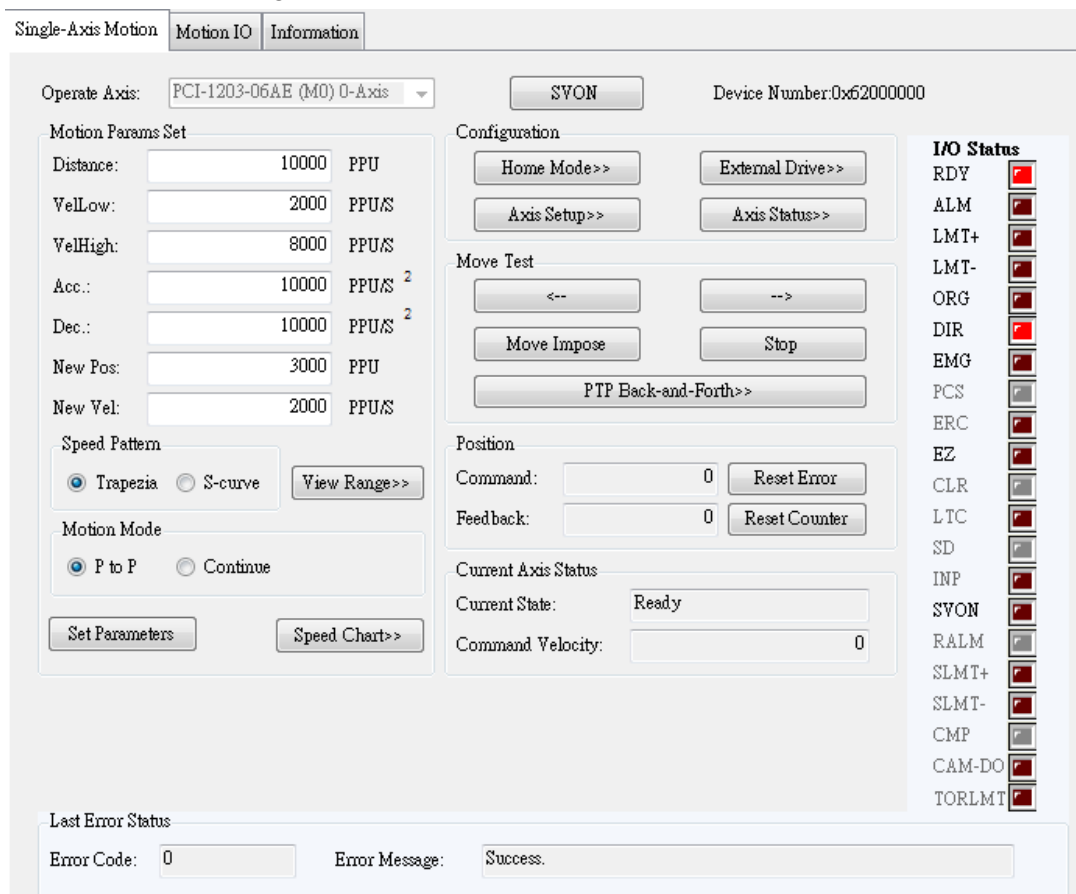
Click **[Topology]** button, the following window will pop out. Check the network connection in this page. If the EtherCAT connection is not correct, for example, users connect the input cable to output port, an  icon will display on the slave with problems.



Click the slave icon, the corresponding operation page will show.

4.4 Slave Pages

Click the slave node in the device tree. The corresponding operating pages will be shown in the operating panel.

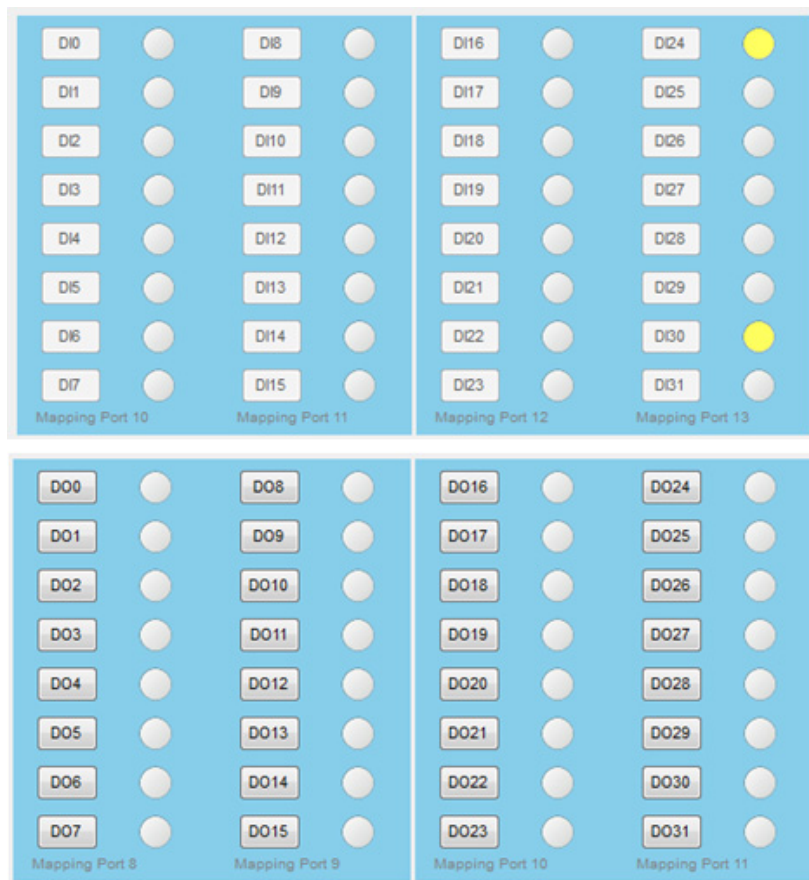


4.4.1 Single-Axis Motion

If the selected slave is servo driver, single-axis motion page will be shown and selectable. The operation of this page is same as the page described in 4.3.1.

4.4.2 Digital Input and Digital Output

If the selected slave supports Digital Input or Digital Output, these pages will be shown and selectable.



● indicates that the signal is in effect (ON) and the value of the bit is 1. ○ indicates that the DO is not in effect (OFF) and the value of the bit is 0. Mapping Port indicates the mapping port number. Users can click DO button to switch on / off the output signal.

4.4.3 Motion IO

Apply Hint: Double click Port / Bit number to edit.

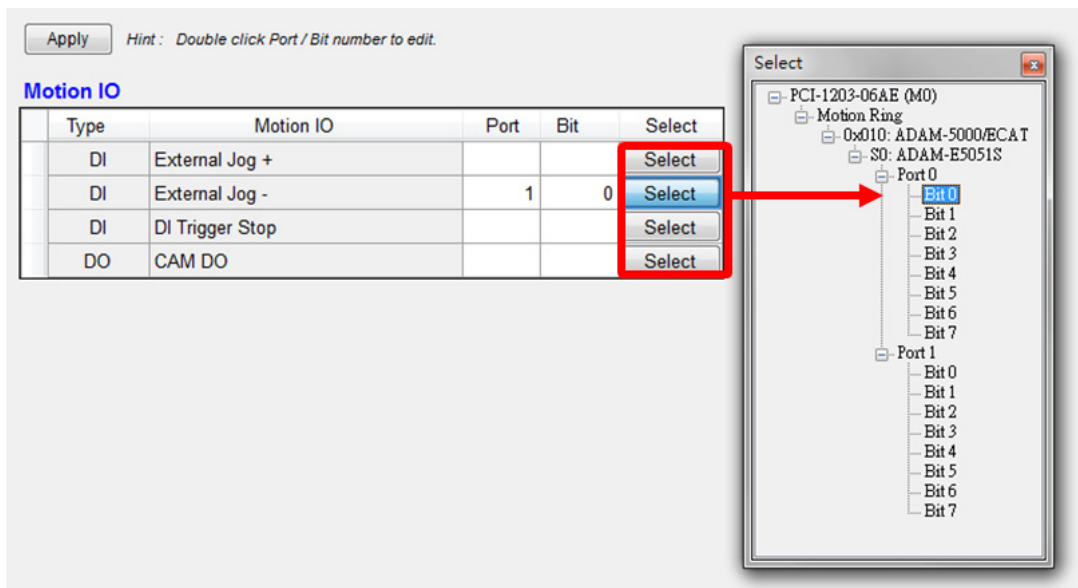
Click this button to apply the settings.

Motion IO

Type	Motion IO	Port	Bit	Select
DI	External Jog +			Select
DI	External Jog -	0	0	Select
DI	DI Trigger Stop	Double click to edit		Select
DO	CAM DO			Select

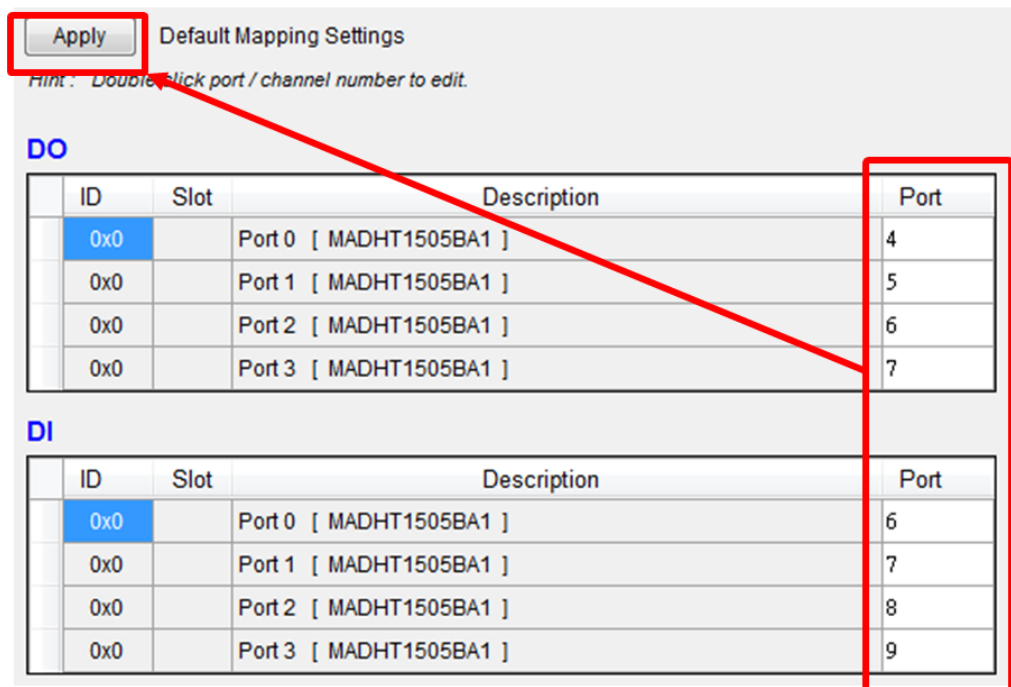
This page is used to map the IO triggered motion function to external IO ports. PCI-1203 supports external DI-jog, external DI-stop and position window output (CAM DO). There are two ways to edit the mapping information: double click the cell to edit

the mapping position or click [**Select**] button, a device list tree window will pop out to select.



In the above picture, the external DI-Jog-trigger stop of this axis is mapped to the bit 0 of DI port 1. The actual location of this DI is defined by mapping table. If the mapping information changed, the actual trigger channel may change. Check the mapping table before using this function is strongly recommended to avoid confusing. After all settings completed, click [Apply] to download the settings to motion card.

4.4.4 IO List



This page shows all the IO of this slave. The port column is the corresponding logical port. Users can double click the port cell to edit the mapping port. After all settings

completed, click **[Apply]** to save the mapping table and apply the settings to motion card.

Note! *We recommend you use the mapping table page in PCI-1203 node to edit the IO mapping and make the final check because the mapping information is more detailed in that page.*



4.4.5 Slave Information

Slave Information

Vendor	Advantech (0x13FE)	
Device Name	ADAM-5000/ECAT	
Device Type	EtherCAT Slave Device	
ProductCode	0x00015530	
Revision No	0x00000000	
Serial No	0x00000000	
ID No	0x <input style="width: 50px;" type="text" value="200"/>	<input type="button" value="Change ID"/>

Link

Port A: <input style="width: 100%;" type="text" value="Link up"/>	Port B: <input style="width: 100%;" type="text" value="Link up"/>
-------------------------------------------------------------------	-------------------------------------------------------------------

Firmware

Current Firmware: V0352_000

Firmware <input style="width: 90%;" type="text"/>	<input type="button" value="Open"/>
Password <input style="width: 100%;" type="text" value="0x00000000"/>	<input type="button" value="Upgrade"/>

This page shows the slave information such as vendor name, vendor ID, device name, device type, product code, revision number, serial number and slave ID which defined in the EEPROM and corresponding ESI file. If vendor logo described in detail in the ESI file, the logo might be shown in front of the vendor name. Users can also check the firmware version of slave and upgrade the firmware.

ID No

The alias ID is defined in the EEPROM and can be modify manually. For Common Motion API, the lowest value of alias ID is 0x0001 and the highest is 0xFFFF. If the slave is ADAM-5000/ECAT, the last 3 digits of alias ID is assigned by 3 locate switch in the left-hand side of slave. When the ADAM-5000/ECAT power on, the last 3 digits value of alias in EEPROM will be covered by the setting of locate switch, but the digit in thousands will keep until the user changes the ID manually. Select the ID No you want to assign to this slave then click **[Change ID]** button to apply setting.

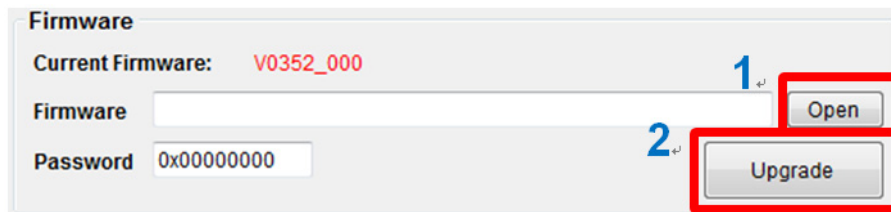
Link Status

It indicates the DLL status (data link layer status) of the individual ports of the EtherCAT slave. The DLL status can have two different states:

Link up: The port is linked up and transmits the EtherCAT datagram.

Loop Closed: The port is closed.

4.4.5.1 Firmware Upgrade

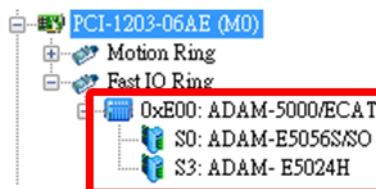


Click **[Open]** to select firmware file you have acquired. Clicking **[Upgrade]** will activate the downloading procedure to hardware and progress bar will show the task process. Firmware Upgrade may take several minutes. DO NOT turn off the power while Firmware Upgrade is in progress, damage may occur.

After the upgrade is finished, reclick the PCI-1203 node on Device Tree to reopen before operation.

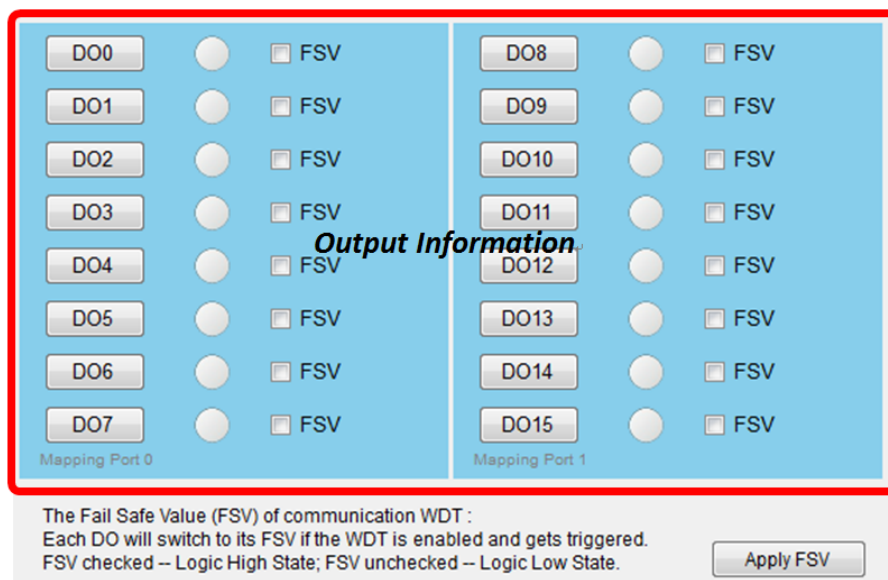
4.4.6 ADAM-E5000 Module Pages

If the EtherCAT slave is ADAM-5000/ECAT, the modules plug in ADAM-5000/ECAT will show in the slave tree node. Click the module node, the corresponding pages will show in the operate panel.



4.4.6.1 Digital Output

This page support ADAM-E5056S 16-ch Isolated Digital Output Module with LED, ADAM-E5057S 32-ch Isolated Digital Output Module and ADAM-E5069 Relay Output Module.



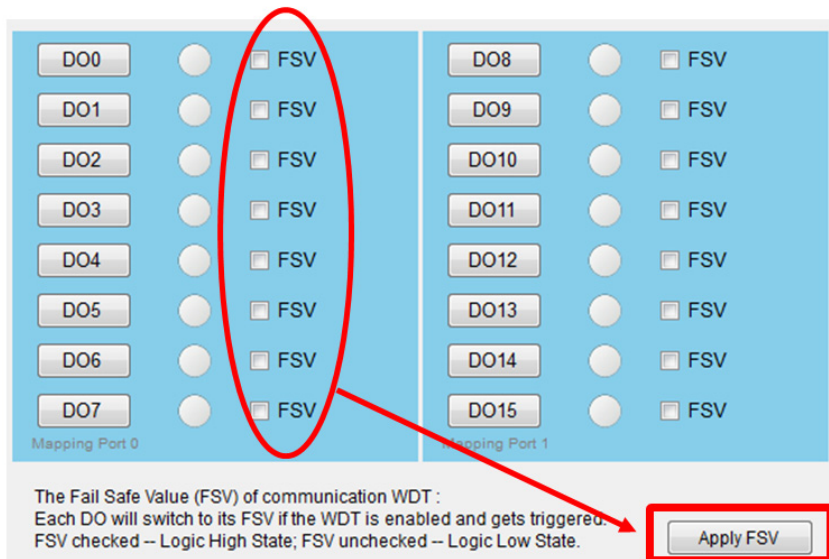
Output Information

There are little differences between each module in the information block due to the channel number.

- indicates that the signal is in effect (ON) and the value of the bit is 1.
- indicates that the DO is not in effect (OFF) and the value of the bit is 0

Mapping Port indicates the mapping port number.

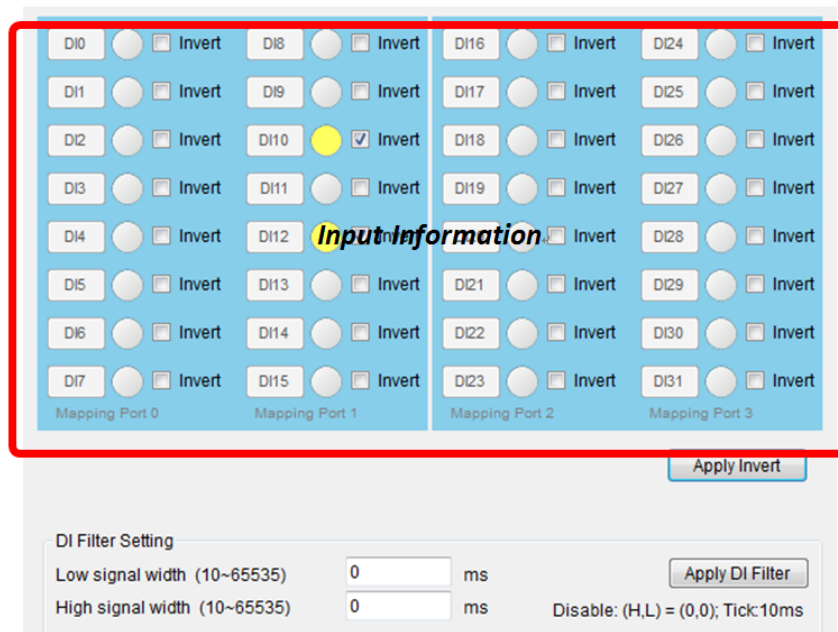
FSV (The Fail Safe Value of communication WDT)



Each DO will switch to its FSV if the module's WDT is enabled and it gets triggered. If you need to enable the FSV, check the FSV check box on each channel and click [ApplyFSV] to apply the setting.

4.4.6.2 Digital Input

This page supports ADAM-E5051S 16-ch Isolated Digital Input Module with LED and ADAM-E5053S 32-ch Isolated Digital Input Module.



Input Information

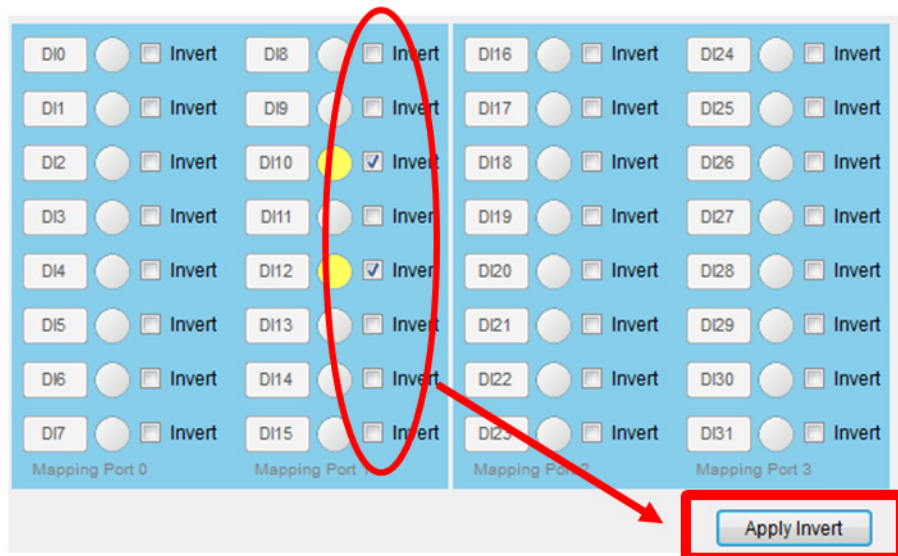
There are little differences between ADAM-E5051S and ADAM-E5053S in the information block due to the channel number. The page of ADAM-E5053S shows 32 channels and the page of ADAM-E5051S shows 16 channels.

- indicates that the signal is in effect (ON) and the value of the bit is 1.
- indicates that the DO is not in effect (OFF) and the value of the bit is 0

Mapping Port indicates the mapping port number.

Invert

If you need to invert the signal of channel, check the Invert check box on each channel and click **[Apply Invert]** to apply the setting.



Digital Filter Scale (Low Signal Width)

The digital input filter is a feature that eliminates noise from input signals. Once the digital input filter is set, it stores data during each sampling and then compares the data of the input terminal state. If all the input signals reach the LOW level within the filtering time, use the state as the input terminal value. If not, use the previous value.

Digital Filter Scale (High Signal Width)

The digital input filter is a feature that eliminates noise from input signals. Once the digital input filter is set, it stores data during each sampling and then compares the data of the input terminal state. If all the input signals reach the HIGH level within the filtering time, use the state as the input terminal value. If not, use the previous value.



The range of the digital filter time is 10 to 65535 in minimum second. To disable the filter, the value of this index must be set to 0. After all settings completed, click **[Apply DI Filter]** to apply the settings to module.

4.4.6.3 Analog Output

This page support ADAM-E5024 4-ch Analog Output Module and ADAM-E5024H 4-ch Analog Output Module

Channel	Value	Output Range	Raw
CH-0	0.000	-10 ~ 10V	0x0
CH-1	0.000	-10 ~ 10V	0x0
CH-2	0.000	-10 ~ 10V	0x0
CH-3	0.000	-10 ~ 10V	0x0

Output Information

Select Channel: 0

Configuration: Output range: -10 ~ 10V, Apply to all, Apply

Calibration: Trim for -10V, Trim for 10V, Trim for 0V, Reset as default

Value: -10, Apply output

Current Value: 0.000, Set current as startup, Set current as FSV

Output Information

The information block shows the current analog output value both in engineering units and raw data. The other setting such as output range of each channel will also be shown.

To get the data in engineering units, the conversion formula and support list are shown below:

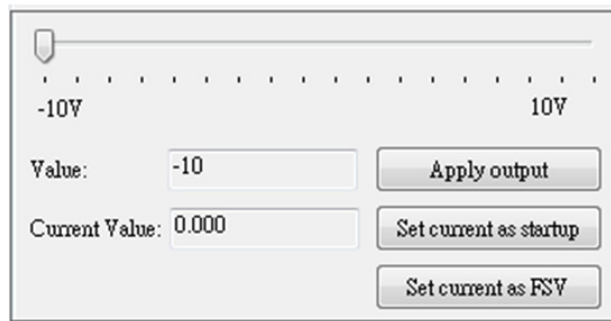
ADAM-E5024

Output Range	Raw data units (R) to Engineering value (E)	Engineering value (E) to Raw data units (R)
0~20 mA	$E = R * 20 / 4095$ (mA)	$R = E * 4095 / 20$
4~20 mA	$E = R * 16 / 4095 + 4$ (mA)	$R = (E - 4) * 4095 / 16$
0~10 V	$E = R * 10 / 4095$ (V)	$R = E * 4095 / 10$

ADAM-E5024H

Output Range	Raw data units (R) to Engineering value (E)	Engineering value (E) to Raw data units (R)
0~20 mA	$E = R * 20 / 65535$ (mA)	$R = E * 65535 / 20$
4~20 mA	$E = R * 16 / 65535 + 4$ (mA)	$R = (E - 4) * 65535 / 16$
0~10 V	$E = R * 10 / 65535$ (V)	$R = E * 65535 / 10$
-5~5 V	$E = R * 10 / 65535$ (V)	$R = E * 65535 / 10$
-10~10 V	$E = R * 20 / 65535$ (V)	$R = E * 65535 / 20$

Set Output Value



There are 4 channels in ADAM-E5024 and ADAM-E5024H. Users can use the scrollbar to set the output value. Before you apply the output of current value, be sure the type of output range is correct.

Power-on Output value (Startup)

In ADAM-E5024H, users can set the current analog output value as startup output value. After the ADAM-5000/ECAT power-on, the voltage/current will output in the corresponding channel. The setting of startup output value will show in the output information block.

Note! ADAM-E5024 doesn't support startup value.



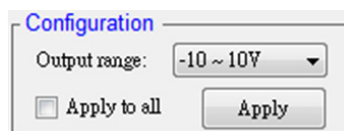
Failed Safe Output Value (FSV)

In ADAM-E5024H, users can set the current analog output value as FSV (Fail Safe Value) output value. If there are some problem to ADAM-5000/ECAT such as network disconnect, each AO channel will output its fail safe value. The setting of FSV output value will show in the output information block.

Note! ADAM-E5024 doesn't support FSV.

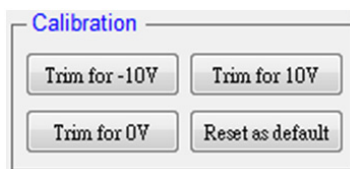


Output Range



The output range can be set to different type for different channel. Select the channel you want to modify or check [Apply to all] to apply setting to all channels. Choose the properly output range, and then click [Apply] to apply the setting.

Calibration



To trim for -Max, Max output value or do Zero calibration, click the button in the Calibration block.

Users must use accurate instrument to calibrate the module, otherwise the channel value will be not accurate. In ADAM-E5024H, if you want to load the default calibration settings, clicking **[Default Setting]** button to reset to the factory setting.

Note! ADAM-E5024 doesn't support factory setting recovery.



4.4.6.4 Analog Input

This page support ADAM-E5017 8-ch Analog Input Module and ADAM-E5017UH 8-ch Ultra High Speed Analog Input Module.

Enable	Channel	Value	Input Range	Raw	Integration Time
<input checked="" type="checkbox"/>	CH-0	0.002	-10 ~ 10V	0x800	60Hz
<input checked="" type="checkbox"/>	CH-1	0.002	-10 ~ 10V	0x800	60Hz
<input checked="" type="checkbox"/>	CH-2	0.007	-10 ~ 10V	0x801	60Hz
<input checked="" type="checkbox"/>	CH-3	0.007	-10 ~ 10V	0x801	60Hz
<input type="checkbox"/>	CH-4	*****		*****	
<input type="checkbox"/>	CH-5	*****	Input Information		*****
<input type="checkbox"/>	CH-6	*****		*****	
<input type="checkbox"/>	CH-7	*****		*****	

Select Channel : 0 Apply Enable

Configuration

Input Range: -10 ~ 10V Apply to all

Integration Time: 60HZ

Calibration

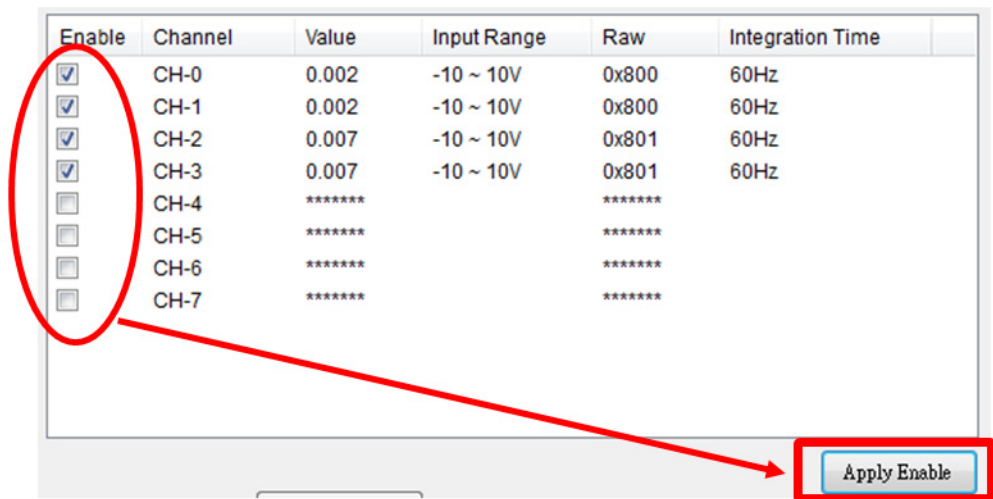
Input Information

The information block shows the current analog input value both in engineering units (Value column) and raw data (Raw column). The other setting such as input range and integration time of each channel will also be shown. If the channel is disabled, the value and raw data.

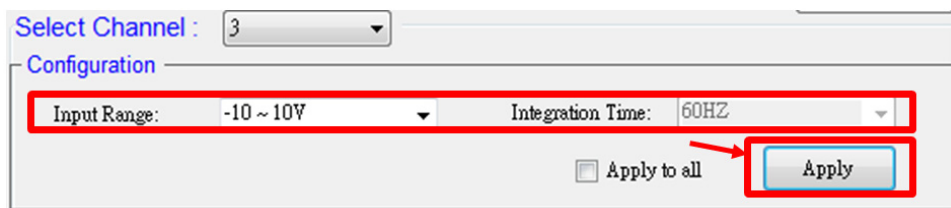
To get the data in engineering units, the conversion formula and support list are shown below:

Input Range	Raw data units (R) to Engineering value (E)	Engineering value (E) to Raw data units (R)	5017	5017 UH
+/- 10 V	$E = R * 20 / 65535 - 10$ (V)	$R = (E + 10) * 65535 / 20$	√	√
+/- 5 V	$E = R * 10 / 65535 - 5$ (V)	$R = (E + 5) * 65535 / 10$		√
+/- 1 V	$E = R * 2 / 65535 - 1$ (V)	$R = (E + 1) * 65535 / 2$		√
+/- 500 mV	$E = R * 1000 / 65535 - 500$ (mV)	$R = (E + 500) * 65535 / 1000$		√
+/- 150 mV	$E = R * 300 / 65535 - 150$ (mV)	$R = (E + 150) * 65535 / 300$		√
+/- 20 mA	$E = R * 40 / 65535 - 20$ (mA)	$R = (E + 20) * 65535 / 40$		√
0~10 V	$E = R * 10 / 65535$ (V)	$R = E * 65535 / 10$	√	
0~500 mV	$E = R * 500 / 65535$ (mV)	$R = E * 65535 / 500$	√	
0~20 mA	$E = R * 20 / 65535$ (mA)	$R = E * 65535 / 20$	√	
4~20 mA	$E = R * 16 / 65535 + 4$ (mA)	$R = (E - 4) * 65535 / 16$	√	√

In the information block, users can disable the analog input channel by uncheck the check box in front of each channel then click **[Apply Enable]** to apply the setting.



Configuration



Users can set the input range and integration in this panel. Select the input range and integration time, then click **[Apply]** button to apply the settings.

Note! There are differences between ADAM-E5017 and ADAM-E5017UH.



For ADAM-E5017, all of channels of module use the same setting of input range. The settings will apply to all channels of this module after the user clicks the **[Apply]** button.

For ADAM-E5017UH, the integration time cannot be set and each channel could have individual input range. Select the channel you want to set in the channel selection combo box or check **[Apply to all]** to apply setting to all channels.

Chapter 5

Programming Guide

5.1 Introduction

This chapter describes two sections:

- Common Motion API

Describes the interface libraries and related content.

- How to use Common Motion API in program:

Creating a new Visual C++ 6.0 application

Creating a new Visual Basic 6.0 application

Creating a new Visual Studio 2005(C#) application

Creating a new Borland C++ Builder application

5.2 About Common Motion API

In order to unify user interfaces of all Advantech motion devices, new software architecture is designed for all Advantech motion devices which is called “Common Motion Architecture”.

Further information on this architecture, refer to Chapter 6.2. In this architecture, users can call the same application programming interface to operate different device.

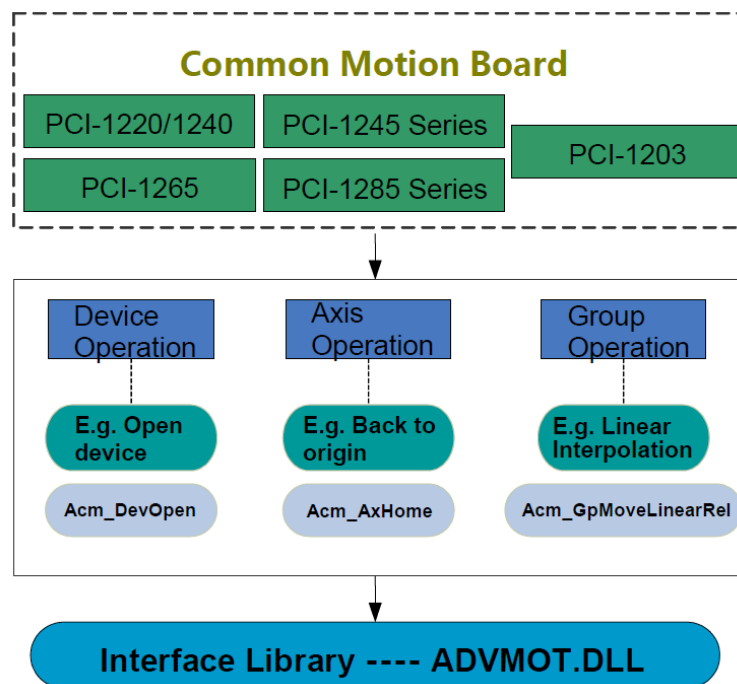


Figure 5.1 Using API in different device

As the figure shown above, all API functions for device are available from the library ADVMOT.DLL.(There are different methods to access in different programming environments, see Chapter 5.4) In order to use these API functions, it is needed to install driver through installation package and the interface library will be installed into Windows system folder.

5.3 Header Files of API

In order to achieve the board functions, it needs to include the following header files to use motion control card interface function. As the figure shows:

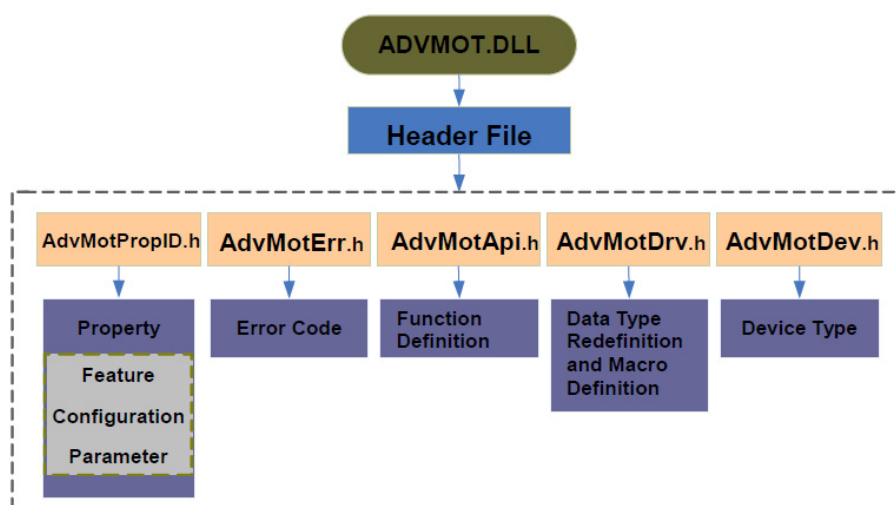


Figure 5.2 Header files of API

The header files are installed on the installation directory “Public” folder. According to the function header file, the following sections will introduce them separately.

5.3.1 About APIs

Advantech Common Motion architecture defines three types of operation objects: device, axis and group. Each type has its own function; the functions are defined in AdvMotApi.h.

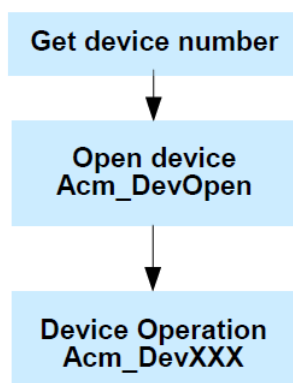
5.3.1.1 Naming Rules of API

According to the difference from these three operating objects, device, axis and group, and AIO, DIO operations, API name follows the rules:

Object	Naming Rules	Description	Example
Device	Acm_DevXXX	Use device function	Acm_DevOpen
AIO/DIO	Acm_DaqXXX	Use DI/DO/AI/AOfunction	Acm_DaqDoGetBit
Axis	Acm_AxXXXX	Use axis function	Acm_AxOpen
Group	Acm_GpXXXX	Use group function	Acm_GpAddAxis

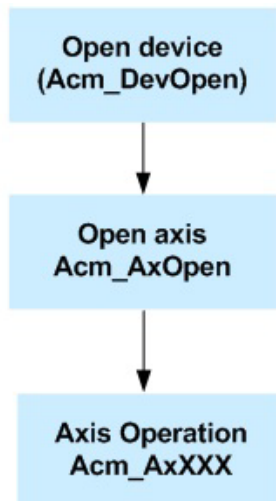
5.3.1.2 How to use API

The basic operation of the device is as follows:

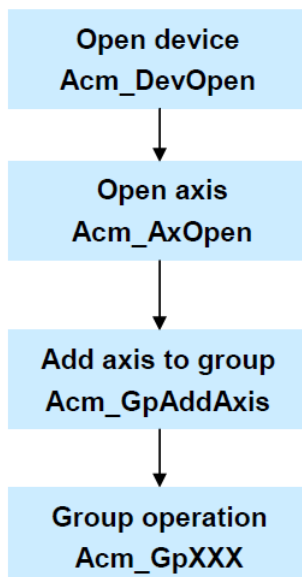



For details about how to get device number, refer to Chapter 5.3.6.1

The basic operation of the axis is as follows:



The basic operation of the group is as follows:



Note!  After opening the device and axis, user can call `Acm_DevLoadConfig` to load the configuration file to set all the configurations such as initial velocity on axis, acceleration, signal of alarm, logic level and so on. The detail information is in the Appendix.

5.3.2 About Properties

Corresponding to the device, axis, group and DIO/AIO, each category has its own property objects, names and values, refer to the header file `AdvMotPropID.h`. About the naming rules and how to get and set these properties, refer to Chapter 5.3.2.1 and 5.3.2.2.

5.3.2.1 Naming Rules of Properties

The properties have three types: feature, configuration and parameter.

Table 5.1: Naming rules of properties

Category	Naming Rules		Classification Rules
Feature	Device	FT_DevXXX	Feature properties are related to the hardware features. These feature properties are read-only.
	Axis	FT_AxXXX	
	DIO/AIO	FT_DaqXXX	
Configuration	Device	CFG_DevXXX	The values of configuration properties may change, but not frequently. Only part of these properties can be set.
	Axis	CFG_AxXXXX	
	Group	CFG_GpXXXX	
	DIO/AIO	CFG_DaqXXX	
Parameter	Device	PAR_DevXXX	The values of parameter properties may change frequently.
	Axis	PAR_AxXXXX	
	Group	PAR_GpXXXX	
	DIO/AIO	PAR_DaqXXX	

5.3.2.2 APIs of Set and GetProperty

Property cannot be called directly, but it can be implemented through API of setting and getting properties. The APIs are shown in the following table:

Table 5.2: APIs of setting and getting property

API	Description
Acm_SetU32Property	Set property (Unsigned 32 -bit integer)
Acm_SetI32Property	Set property (Signed 32 -bit integer)
Acm_SetF64Property	Set property (Double)
Acm_GetU32Property	Get property (Unsigned 32 -bit integer)
Acm_GetI32Property	Get property (Signed 32 -bit integer)
Acm_GetF64Property	Get property (Double)

To use different API in Table 5.2 to set and get property is related to the data type of property, as shown below:

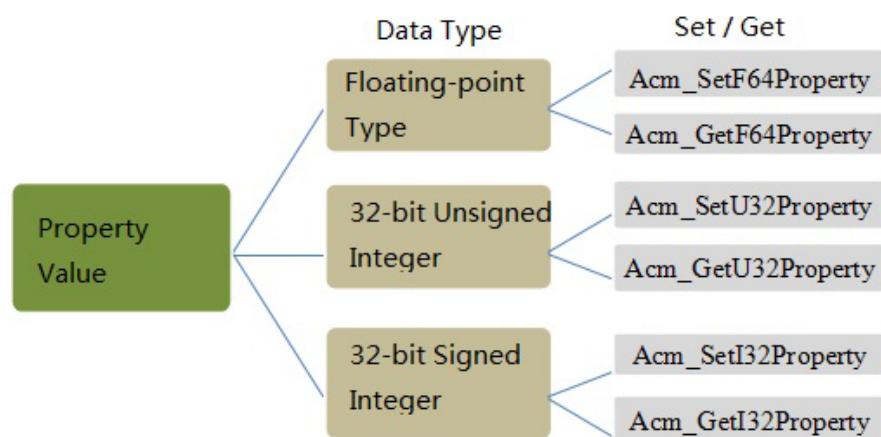


Figure 5.3 Relationship between data type of property and functions

Axis property of operating velocity is defined as the table:

Property	Data Type	Read/Write	Property ID	Description
PAR_AxVelHigh	F64	RW	401	Set/get operating velocity of axis

The data type of this property is F64 (Double), so it needed to call Acm_SetF64Property/Acm_GetF64Property to set/get value of property.

Acm_SetF64Property (HAND Handle, U32 PropertyID, F64 Value)

Parameter	1: Handle	2: PropertyID	3: Value
Device	Device handle from Acm_DevOpen.		
Axis	Axis handle from Acm_AxOpen.	PAR_AxVelHigh	Value of property, the data type is F64
Group	Group handle from Acm_GpAddAxis.		

For example, the operating velocity on axis will be set to 8000, it is set as follow:

Acm_SetF64Property(Axishand,PAR_AxVelHigh,8000);//Operating velocity is 8000

5.3.3 About Device Type

The definition of device type is in the header file AdvMotDev.h, it will be used as the input parameter of Acm_DevOpen. For details about how to get the device number, see Chapter 6.4.

Table 5.3: Device type definition

Device type	Type definition
PCI-1220	Adv_PCI1220
PCI-1240	Adv_PCI1240
PCI-1245	Adv_PCI1245
PCI-1245L	Adv_PCI1245L
PCI-1245E	Adv_PCI1245E
PCI-1245V	Adv_PCI1245V
PCI-1245S	Adv_PCI1245S
PCI-1245HP	Adv_PCI1245HP
PCI-1265	Adv_PCI1265
PCI-1285	Adv_PCI1285
PCI-1285E	Adv_PCI1285E
PCI-1285V	Adv_PCI1285V
MIC-3285	Adv_MIC3285
MIC-3245	Adv_MIC3245
ADAM-5000ECAT (Lan Port)	Adv_ADAM5000
PCI-1203	Adv_PCI1203

5.3.4 About Error Code

Every API in Common Motion Architecture will get a returned code when it is called. The returned code represents a calling result. About the detail error code, see about Appendix. User can get error message according to the returned error code by Acm_GetErrorMessage. According to error message, user can make modification properly. For detailed information about error codes, refer to the Appendix.

5.3.5 Data Type Redefinition

The table of redefinition of data types and windows common data types is as follows:

Table 5.4: Data type redefinition		
New Type	Windows Data Type	Comments
U8	UCHAR	8 bit unsigned integer
U16	USHORT	16 bit unsigned integer
U32	ULONG	32 bit unsigned integer
U64	ULONGLONG	64 bit unsigned integer
I8	CHAR	8 bit signed integer
I16	SHORT	16 bit signed integer
I32	INT	32 bit signed integer
I64	LONGLONG	64 bit signed integer
F32	FLAOT	32 bit Floating point variable
F64	DOUBLE	64 bit Floating point variable
PU8	UCHAR *	Pointer to 8 bit unsigned integer
PU16	USHORT *	Pointer to 16 bit unsigned integer
PU32	ULONG *	Pointer to 32 bit unsigned integer
PU64	ULONGLONG *	Pointer to 64 bit unsigned integer
PI8	CHAR *	Pointer to 8 bit signed integer
PI16	SHORT *	Pointer to 16 bit signed integer
PI32	INT*	Pointer to 32 bit signed integer
PI64	LONGLONG *	Pointer to 64 bit signed integer
PF32	FLAOT *	Pointer to 32 bit Floating point variable
PF64	DOUBLE *	Pointer to 64 bit Floating point variable

The initial character F/I/U represents the data type, and the digital represents the length of data.

5.3.6 Getting Started

For the operation of the motion card, as described in Section 5.3.1.2, you first need to open the motion card according to the device number, get device Handle, and open the axis, then you can do single-axis motion; For doing the group motion, open axis, add the axis to the group, get group Handle, and then you can operate the group.

According to this process, this section focuses on the how to start to control this motion card.

5.3.6.1 Get Device Number

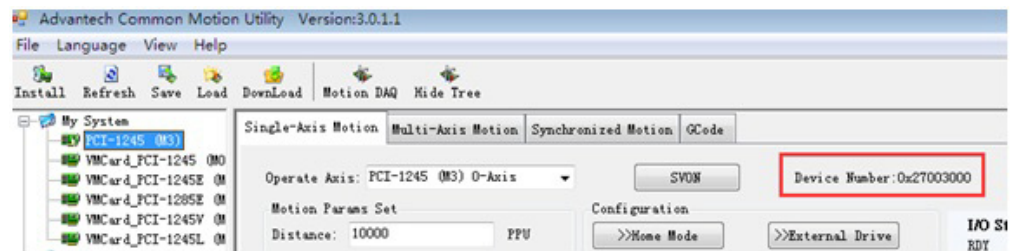
In order to distinguish between different devices, each card will be numbered, that is device number. Each device number of motion card is unique, about the rules of device number, see Chapter 6.4.

To operate the motion control card in Common Motion architecture, user need to call Acm_DevOpen to get the handle of device and this API need the device number as input parameter. There are two ways to get the device number:

- Method 1: Get device number through Utility
- Method 2: Get device number through API

Method 1: Get device number through Utility

In the upper right corner of the page of Single-Axis Motion in Utility, you can see the device number, as shown below:



The Device Number can be entered directly in the open device function to reach the device handle.

Method 2: Get device number through API

It provides two functions to get the device number:

```
Acm_GetAvailableDevs(DEVLIST *DeviceList, U32 MaxEntries, PU32 OutEntries)
```

```
Acm_GetDevNum(U32 DevType, U32 BoardID, PU32 DeviceNumber)
```

For details of Acm_GetAvailableDevs, refer to the example.

Details of how to use Acm_GetDevNum API are shown as follows:

Parameter	Type	IN or OUT	Description
DevType	U32	IN	Device type which defined in header file, see section 5.3.3
BoardID	U32	IN	Board ID
DeviceNumber	PU32	OUT	Return value of device number

Board ID can be directly obtained from the hardware, and it can be adjusted. If the board ID is not adjusted, it generally does not change. BoardID is also available from the Utility. Get device number to open the device as follows:

```
HAND m_Devhand;
```

```
U32 m_DevNum;
```

```
Acm_GetDevNum(Adv_PCI1203,1,&m_DevNum); //Get device number
```

```
Acm_DevOpen(m_DevNum, &m_Devhand); // open device,get device Handle
```


5.3.6.2 Flow Chart of Initialization

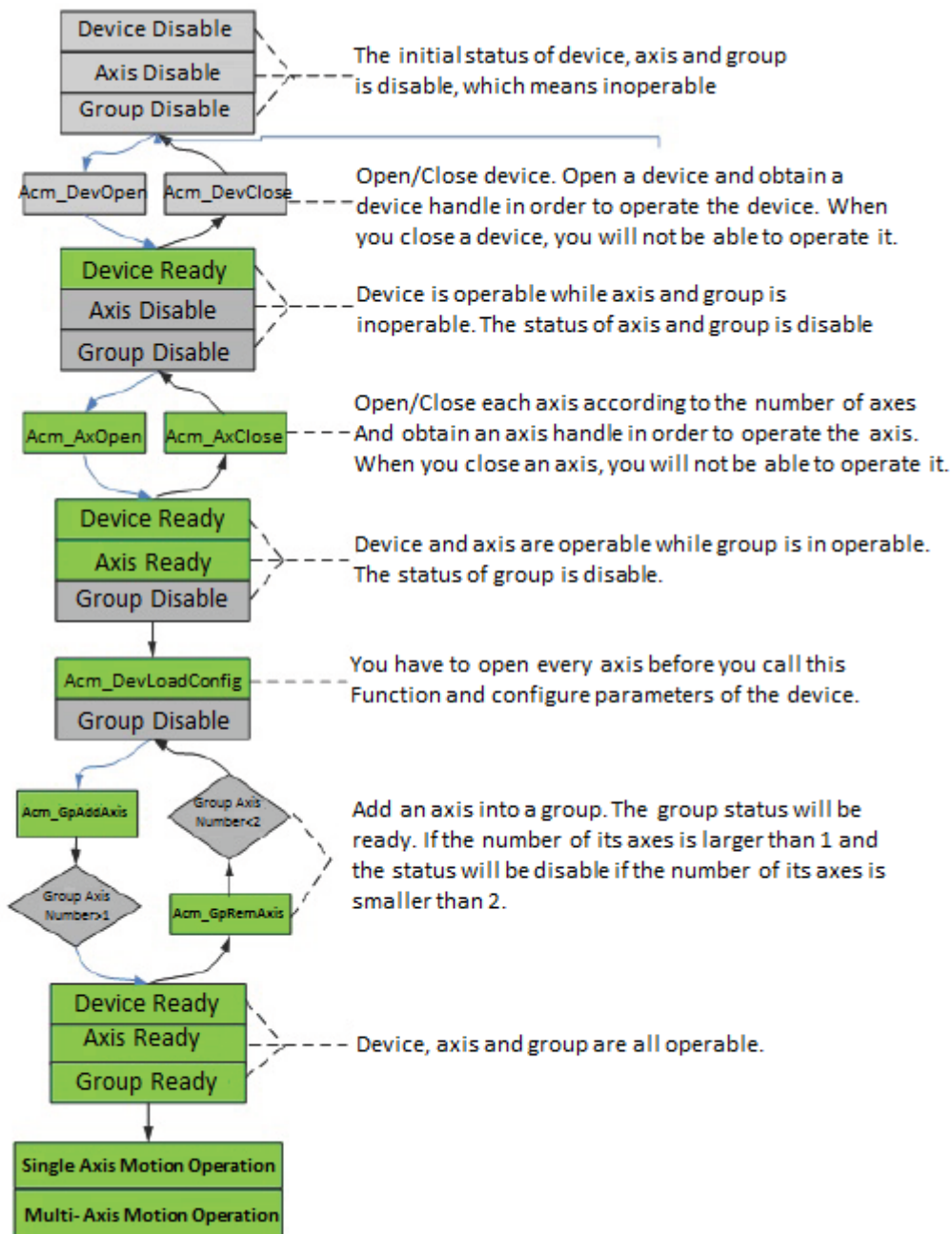


Figure 5.4 Flow chart of board initialization

5.3.6.3 Example

Board initialization example is shown as follow. Before opening the device, you first need to obtain the device number (Section 5.3.6.1). In this example we get it through the API.

VC Code:

```
HAND m_Devhand; //Device Handle
HAND m_Axishand[32]; //Axis handle
HAND m_GpHand; //Grouphandle
U32 m_DevNum; //Device Number
U32 m_ulAxisCount; //Axes Count
U32 Ret;// Function return value
Ret=Acm_GetDevNum(Adv_PCI1203, 15, &m_DevNum); //Get device number
Ret= Acm_DevOpen(m_DevNum, &m_Devhand); //Open devce, get device handle
Ret= Acm_GetU32Property(m_Devhand,FT_DevAxesCount,&m_ulAxisCount); //Get axes count
//According to number of axes, open all of axis
for(uint AxisNumber=0;AxisNumber<m_ulAxisCount;AxisNumber++)
{
    Ret=Acm_AxOpen(m_Devhand,(USHORT)AxisNumber,&m_Axishand[Axis-
    Number]); //Open axis, get axis handle
    Acm_AxSetCmdPosition(m_Axishand[AxisNumber], 0); //Set axis command
    position to 0
    Acm_AxSetActualPosition(m_Axishand[AxisNumber], 0); // Set axis actual posi-
    tion to 0
}
//Add axis 0 and 1into group
Ret =Acm_GpAddAxis(&m_GpHand,m_Axishand[0]);
Ret =Acm_GpAddAxis(&m_GpHand,m_Axishand[1]);
//Do single-axis motion, refer to Chapter 9.2
//Do multi-axis motion such as line interpolation, refer to Chapter 9.3
//After operating these axes, it need to close group, axis and device
Acm_GpClose(&m_GpHand); //Cloas group
for(uint AxisNum=0;AxisNum<m_ulAxisCount;AxisNum++)
    Acm_AxClose(&m_Axishand[AxisNum]); //Close axis
Acm_DevClose(&m_Devhand); //Close device
```

5.4 Creating a New Application

For creating a new application under PCI-1203, users ought to install Common Motion Examples, there are many examples developed in different language in folder Advantech\Common Motion\Example_1203, users can follow these examples to develop a new application.

After installing Common Motion examples, users can find two folders Include and Publicin folder \Advantech\Common Motion, the files in Public folder are supplied for users to create applications in different languages, the relationship between files and developing language is as Figure 5.5.

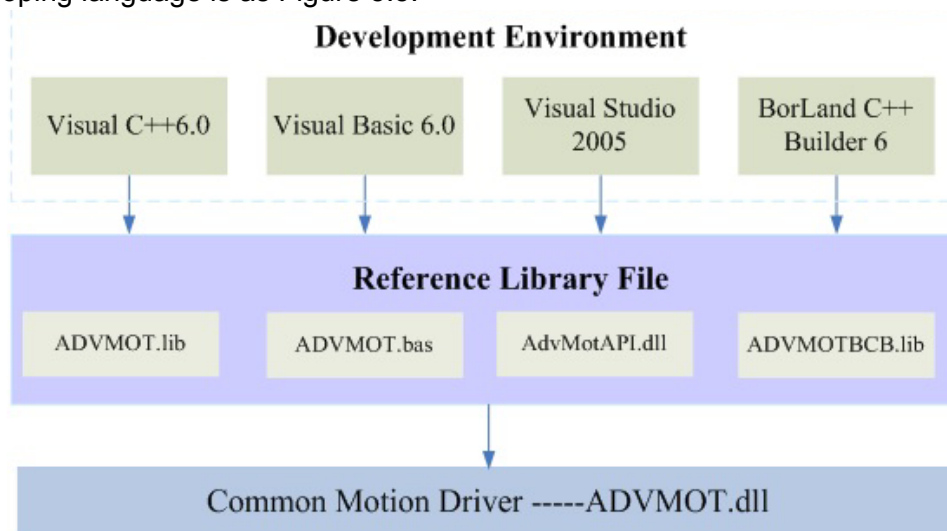


Figure 5.5 Software Architecture

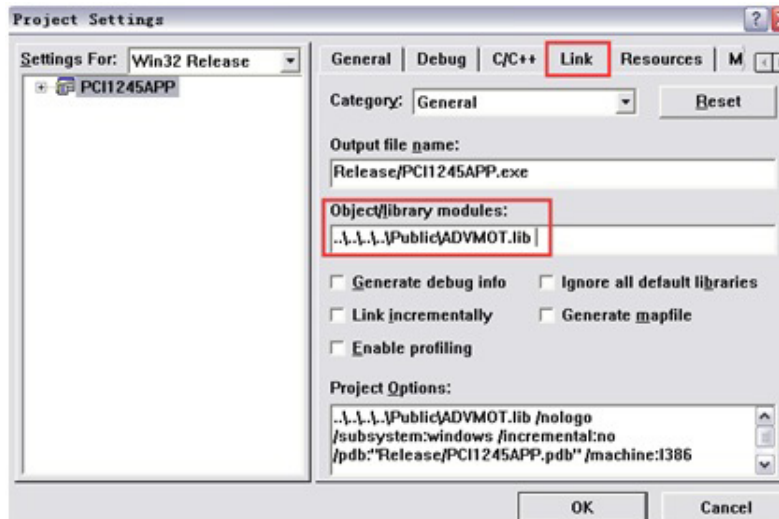
All of API used to implement device functions can be acquired from ADVMOT.DLL which is a common interface for user. The AdvMotAPI.dll, ADVMOT.bas and ADVMOT.lib are created upon ADVMOT.dll for user developing application easily. AdvMotAPI.dll is used for C# application and VB.net application which includes Utility, C# examples and VB.net example. ADVMOT.bas is used to develop VB application. ADVMOT.lib is used to develop VC application. ADVMOTBCB.lib is used to develop BCB application.

5.4.1 Creating a New Visual C++ 6.0 Application

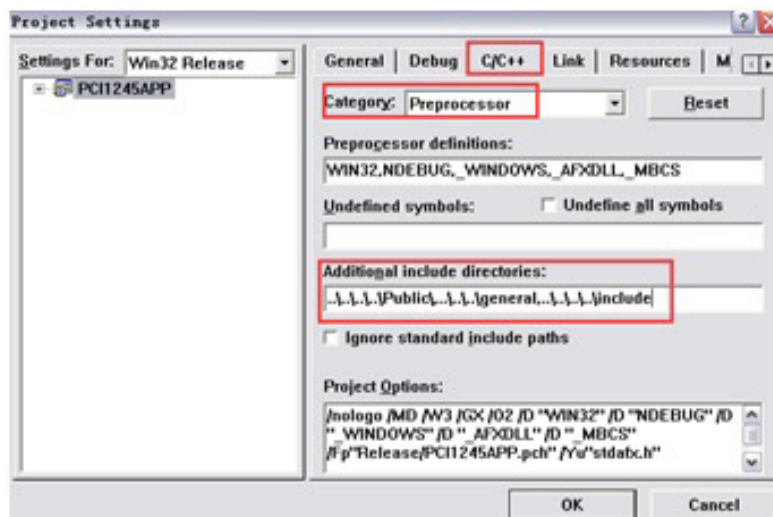
For creating a new application, the procedure is as follow:

1. Create a new project.
2. Open "Project Setting" in Menu - Project - Settings... or right click the new Project and chose "Setting" to open

3. Switch to “Link” page and specify the path of ADVMOT.lib in the column of “Object/library modules”



4. Switch to “C/C++” page and specify the path of header files in the column of “Additional include directories”



5. Include the library header files in the application as follows

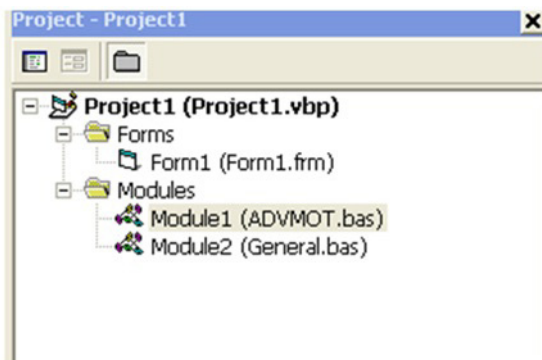
```
#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000
#include "advmotdrv.h"
#include "General.h"
#include "AdvMotApi.h"
```

And then users can call any functions in Visual C++, and start writing applications.

5.4.2 Creating a New Visual Basic 6.0 Application

For creating a new application, the procedure is as follow:

1. Open the Visual Basic 6.0 development program, select the Standard EXE icon and press the “**Open**” button. A new project is created.
2. Adding the module into project. Click on the Project Explorer in the Viewmenu. Add ADVMOT.bas (In the Advantech\Common Motion\Public folder after installing examples package) module and general.bas (In the folder \Advantech\Com-
mon Motion\Example_1203 after installing examples package) by clicking on Add Module in the Project menu.



And then users can call any functions in Visual Basic, refer to the examples.

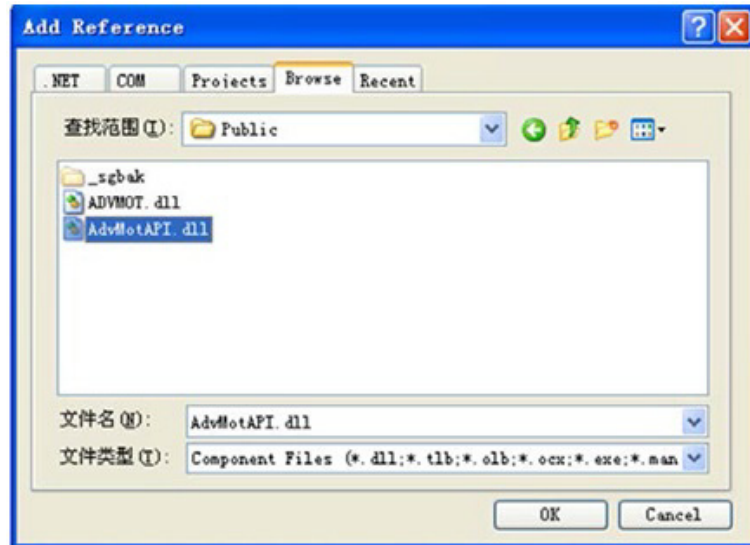
5.4.3 Creating a New Visual C# Application

For creating a new C# application in Visual Studio 2005, the procedure is as follow:

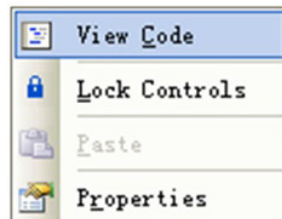
1. Select [**File**] ---> [**New**] ---> [**Project**] in Visual Studio 2005 and create a new C# project.
2. Add relevant .dll
 - a. Click [**References**] on the top right corner of development environment, as follows:



- b. Click **[Browse]** of the **[Add Reference]** dialog box, Select “AdvMotAPI.dll” in the “Public” file folder from search path, then click **[OK]**, as follows:



- c. Right click on the Edit interface; select **[View Code]** to enter the program source code compilation interface, as follows:



- d. Add “using Advantech.Motion” under original referred name spaces, as follows:

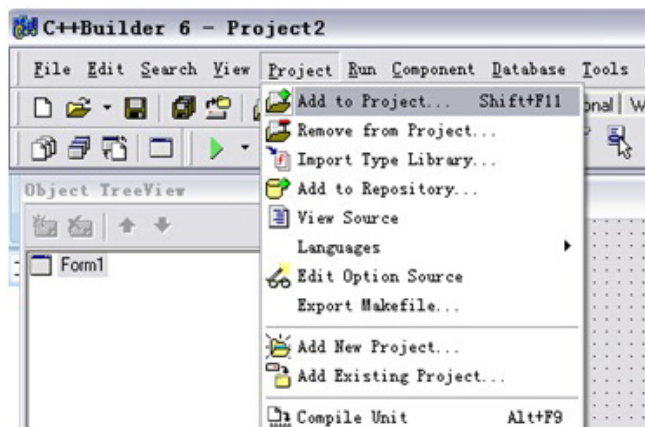
```
1 using System;
2 using System.Collections.Generic;
3 using System.ComponentModel;
4 using System.Data;
5 using System.Drawing;
6 using System.Text;
7 using System.Windows.Forms;
8 using Advantech.Motion;
```

And then users can call any functions in Visual C#, refer to examples. About how to include library files when developing VB .NET application in Visual Studio 2005, the procedure is same as C# project.

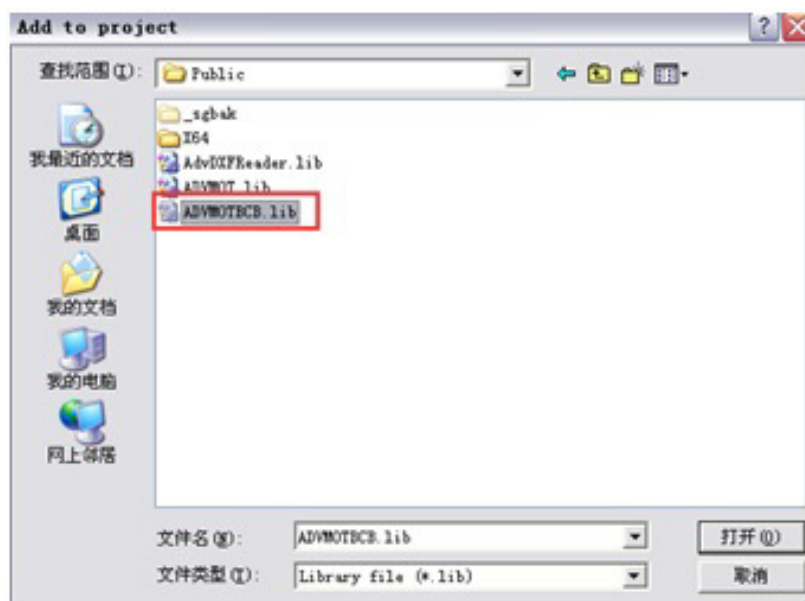
5.4.4 Creating a New Borland C++ Builder Application

For creating a new BCB application in Borland C++ Builder 6.0, the procedure is as follow:

1. Select **[New]** ---> **[Application]** in Borland C++ Builder 6.0 and Create a new project.
2. Include library files as follow



3. Search and select the path of "ADVMOTBCB.lib" in "Public" file, and then click **[OK]**.



And then users can call Common Motion API to develop application.

5.5 Using Common Motion API in Win7

5.5.1 About Elevating Application Privileges

Acm_GetAvailableDevs has to read information from the registry in order to get the information of all boards that are installed in the computer. This operation requires Administrator rights. Therefore, if the application has to call this function, add the corresponding Manifest file and grant administrator rights to the application. (Refer to "About Granting Administrator Rights to Applications".)

1. To develop applications with Microsoft Visual Studio 2005(VS2005)

Copy the Manifest file "app.manifest" from the Properties folder of C#/VB.net examples to the Projects folder of the project. Click "Project"->"Add Existing Item" to add it to the project.

2. To develop applications with Microsoft Visual C++ 6.0

Copy the Manifest file "App.manifest" from VC examples to the path of the project. Import this file to the source. Source type: 24; Source ID: 1.

3. To develop applications with Microsoft Visual Studio 2008/2010

Method 1: Copy app.manifest from examples to the project (as in VS2005);

Method 2: Directly change settings of project privilege management: Click "**Project Properties**"->"**Configuration Properties**"-->"**Linker**"-->"**Manifest File**"-->"**UAC Execution Level**"-->"**require Administrator**".

Method 3: Check the "**Enable ClickOnce Security Setting**" option in "**Security**" column of "**Project properties**", and the Manifest file will be automatically generated under "**Properties**". Open the Manifest file and change the content marked by the red box in the following image to "<requestedExecutionLevel level='requireAdministrator' uiAccess='false' />". Uncheck the "**Enable Click- Once Security Setting**" option in Security column of "**Project properties**".

```

<requestedPrivileges xmlns="urn:schemas-microsoft-com:asm.v3">
  <!-- UAC Manifest Options
    If you want to change the Windows User Account Control level replace the
    requestedExecutionLevel node with one of the following.

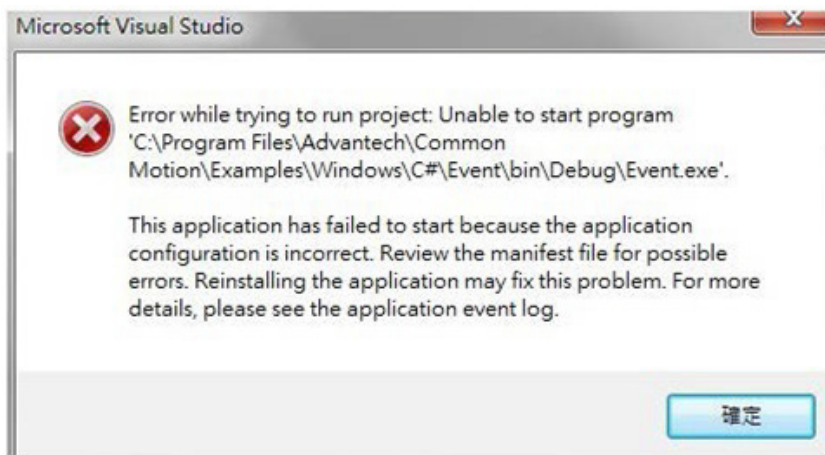
    <requestedExecutionLevel level="asInvoker" uiAccess="false" />
    <requestedExecutionLevel level="requireAdministrator" uiAccess="false" />
    <requestedExecutionLevel level="highestAvailable" uiAccess="false" />

    Specifying requestedExecutionLevel node will disable file and registry virtualization.
    If you want to utilize File and Registry Virtualization for backward
    compatibility then delete the requestedExecutionLevel node.
  -->
  <requestedExecutionLevel level="asInvoker" uiAccess="false" />
</requestedPrivileges>

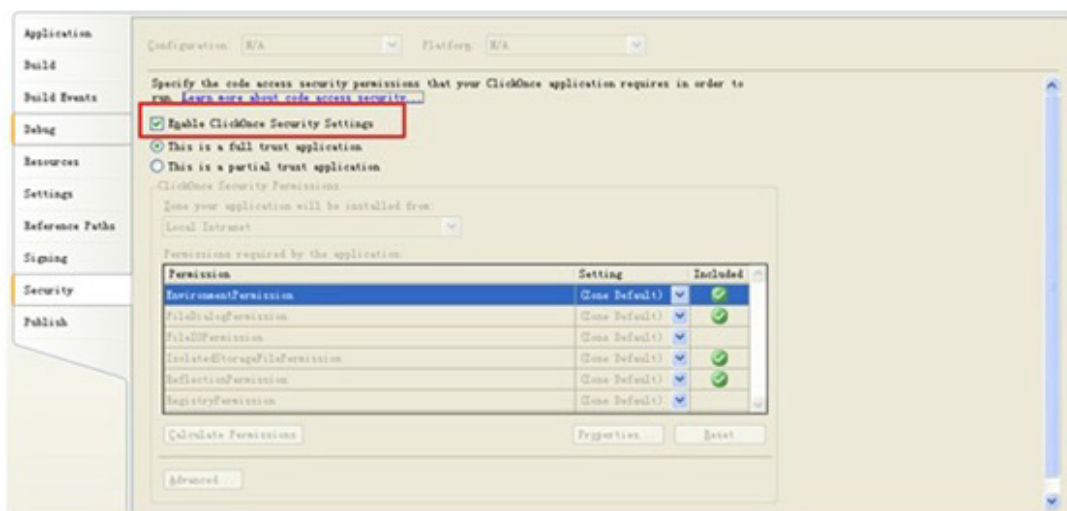
```


5.5.2 Using Common Motion API in Win7

If you open C#/VB.net examples with VS2008 or VS2010 and the following error messages appear:



Uncheck the **“Enable Click Once Security Setting”** option in Security column of Project properties. Recompile and the application will run successfully



5.6 Multiple Motion Cards Programming

When there is multiple (two or more) motion cards plug in the user's computer, using the method described below

5.6.1 Independence of Multiple Motion Cards

When there is multiple motion cards plug in the computer, each motion card is independent. The axes which used in group to implement interpolation motion (Line / Arc) must be in the same motion card, it is not support add axis from different motion card into a group. For example, two PCI-1203 card, which the PCI-1203 (0) and PCI-1203(1) can only control the axis connected to its ring. It cannot select the axis on board (0) and another axis on board (1) to do interpolation motion.

5.6.2 Operating Multiple Motion Cards

In the following example, a computer plugs two PCI-1203 motion cards.

Motion card 1: PCI-1203 BoardID = 0

Motion card 2: PCI-1203 BoardID = 1

1. Device Number:

```
DWORD nDevNum[2]; //Device number of two PCI Card
```

```
Acm_GetDevNum(Adv_PCI1203,0,&nDevNum[0]); //Get device number
```

PCI-1203 Device Type (Chapter 5.3.3)

BoardID = 0

```
Acm_GetDevNum(Adv_PCI1203,1,&nDevNum[1]); //Get device number
```

BoardID = 0

Note! About the information of Device Number, refer to Chapter 5.3.6.1.



2. Open two PCI cards and its axes

After getting the device number of two PCI cards, using `Acm_DevOpen` to open device and get device handle. And then use `Acm_AxOpen` to open all of axis on devices and get axes handle.

```
HAND hDevice[2]; //Device Handle (two PCI-1203 motion cards)
```

```
HAND hAxis[8]; //Axis Handle, for example, there are 4 axes connected to each PCI-1203 (totally 8 axes)
```

```
int nDev=0;
```

```
for( nDev=0; nDev<2; nDev++)
```

```
{
```

```
    Acm_DevOpen(nDevNum[nDev], &hDevice[nDev]); // Open device
```

```
    for( U16 PhyAxisID=0; PhyAxisID<4; PhyAxisID ++)
```

```
        Acm_AxOpen(hDevice[nDev],PhyAxisID,&hAxis[nDev*4 + PhyAxisID]); // Open all of axis
```

```
}
```

3. Motion control

■ Singal-axis motion

After getting axis handle, user can control axis. For example, commanding relative P2P motion of 8 axes:

```
for(int AxisNum =0;AxisNum<8;AxisNum++)
```

```
    Acm_AxMoveRel(hAxis[AxisNum],10000);
```

■ 2-axis linear interpolation

After getting group handle through adding the axis into group, user can control group to do interpolation motion.

```
HAND hGp[2]; //two group Handle
```

```
double m_End[2] = {10000,10000};
```

```
// Group 0: Add 2 axes (hAxis [0], hAxis [1])
```

```
Acm_GpAddAxis(&hGp[0], hAxis [0]);
```

```
Acm_GpAddAxis(&hGp[0], hAxis [1]);
```

```
// Group 1: Add 2 axes (hAxis [4], hAxis [5])
```

```
Acm_GpAddAxis(&hGp[1], hAxis [4]);  
Acm_GpAddAxis(&hGp[1], hAxis [5]);  
Acm_GpMoveLinearRel(hGp[0],m_End, 2);//Group 0: Linear interpolation  
{10000,10000}  
Acm_GpMoveLinearRel(hGp[1],m_End, 2);//Group 1: Linear interpolation  
{10000,10000}
```


Chapter 6

Common Motion
Architecture

6.1 Introduction

This chapter describes the concept and implement of Common Motion Architecture, naming rules of API and properties and device number of Advantech motion devices.

6.2 Common Motion Architecture

In order to unify user interfaces of all Advantech motion devices, new software architecture is designed for all Advantech motion devices which is called “Common Motion Architecture”. This architecture defines all user interfaces and all motion functions that are implemented, including single axis and multiple axes. This unified programming platform enables users to operate devices in the same manner.

There are three layers in this architecture: Device Driver Layer, Integrated Layer and Application Layer. Users do not need to know how to operate the specific driver of a specify device, but only to know the Common Motion Driver. Even though the device which supports this architecture has changed, the application does not need to be modified.

Advantech Common Motion (ACM) Architecture defines three types of operation objects: Device, Axis and Group. Each type has its own methods, properties and states.

To start single axis motion, you have to follow the following steps:

Open device->open one axis of this device->configure instance of this axis->start motion.

All operations can be done by calling corresponding ACM APIs. General calling flows of Device, Axis, and Group are specified by Common Motion Architecture. For detailed information, refer to the Calling Flow section.

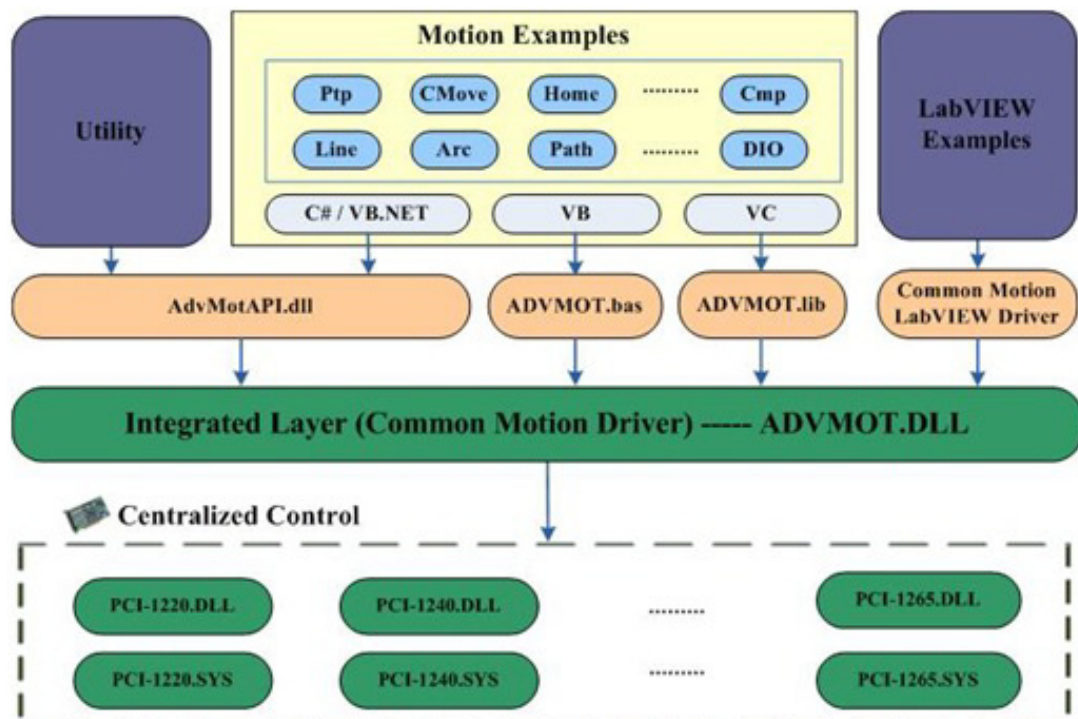


Figure 6.1 Common Motion Architecture

6.3 Naming Rules of APIs and Properties

The naming rule is based on three objects: Device Object, Axis Object and Group Object. User will find many abbreviations in APIs. Table of abbreviations and their meanings is as follow:

Table 6.1: Abbreviations and Their Meanings		
Abbreviations	Full Name	Comments
PPU	Pulse Per Unit	A virtual unit of motion
Dev	Device	
Ax	Axis	
Gp	Group	Multiple axes
Mas	Master	Master Axis or Master Board of device based on communicating mechanism
Daq	Data collection	Common name of AI/AO/DI/DO
Rel	Relative	
Abs	Absolute	
Cmd	Command	
Vel	Velocity	
Acc	Accelerate	
Dec	Decelerate	
Emg	Emergency	Emergency stop
Sd	Slow down	
Info	Information	
Cmp	Compare	
Inp	In position	
EZ	Encode Z	
EI	Hardware Limit	
MeI	Negative Limit	
PeI	Positive Limit	
Org	Origin	
Ext	External	
FT	Feature	Feature properties
CFG	Configuration	Configuration properties
PAR	Parameter	Parameter properties
Ipo	Interpolation	
Chan	Channel	

6.4 Device Number

To operate the motion control card in Common Motion architecture, user need to call Acm_DevOpen to get the handle of device and this API need the device number as input parameter. Device number is composed of 32 bits:

4th byte	3rd byte and 2nd H byte	2nd L byte	1st byte
Master/device type ID	Master/device board ID (or BaseAddr)	Ring	Slave Board ID

4th byte: Master/device type ID (refer to master device type ID table))

3rd& 2nd H byte: Master/device board ID (or base address), the board ID can be modified through switch of the PCI card.

2nd L byte: Master ring number, used by remote device, use 0 as default value for local device.

1st byte: Slave board ID, used by remote device, use 0 as default value for local device.

Local Device Number

4th byte	3rd byte and 2nd H byte	2nd L byte	1st byte
Master/device type ID	Device board ID	0	0

For example, one BoardID of PCI-1203 is 1, the device number (Hexadecimal) is:

62	001	0	0
----	-----	---	---

So the device number is 0x62001000.

As described above, when obtaining board device number, it is also available in the following way:

```
DWORD nDevNum[2]; //Device number of two PCI card
```

```
nDevNum[0]= (Adv_PCI1203<<24) | (0<<12);
```

```
nDevNum[1]= (Adv_PCI1203<<24) | (1<<12);
```

PCI-1203 Device Type (Chapter 5.3.3)

BoardID = 0

BoardID = 1

Chapter 7

Description of
Property
Configuration Files

7.1 Introduction

The user has to configure some parameters, such as speed, ALM signal, left limit and right limit, before utilizing the motion controlling card. The process of configuring card properties is called system configuration process.

In the testing utility, Save and Load options of configuratoin(.cfg) files are provide for the user to save or load the configuration information of all axes of the device selected. During the programming, the user can call API to transfer the information of configuration files to the motion controller, so as to complete the configuration of the controller.

7.2 Basic Concept of System Configuration

Advantech motion controller includes two parts: hardware resources and software resources, which should be perfectly combined for different applications of the controller.

7.2.1 Hardware Resource Configuration

Hardware resource configuration is composed of digital input (DI)/digital output (DO) and analog input (AI)/analog output (AO).

7.2.1.1 Digital Output (DO)

Digital output includes:

- Servo Alarm Clear Signal (ERC)
- Cam Output (CAMDO)

Servo Alarm Clear Signal (ERC)

Property	.CFGFile Abbreviation	Description
CFG_AxErcLogic	ErcLogic	Clears a signal's valid logic level.
CFG_AxErcEnableMode	ErcEnMde	Enables/disables the source axis' alarm clear function.
CFG_AxErcOnTime	ErcOnTime	Sets the period of starting an alarm clear signal.
CFG_AxErcOffTime	ErcOffTime	Sets the period of closing an alarm clear signal.

*So far, motion controlling cards don't support CFG_AxErcOnTime and CFG_AxErcOffTime property functions.

Cam Output (CAMDO)

Property	.CFGFile Abbreviation	Description
CFG_AxCamDOEnable	CamDoEnable	Enables/disables CAMDO.
CFG_AxCamDOLoLimit	CamDOLoLimit	Sets the low limit for CAM DO signal.
CFG_AxCamDOHiLimit	CamDOHiLimit	Sets the high limit for CAM DO signal.
CFG_AxCamDOCmpSrc	CamDoCmpSrc	Sets the compare source for CAM DO signal.
CFG_AxCamDOLogic	CamDoLogic	Cam logic level.

7.2.1.2 Digital Input (DI)

Digital output includes:

- Motion Alarm Digital Input (ALM)
- IN Position Signal Input (IN Position)
- Positive/Negative Limit Digital Input
- ORG Signal Digital Input
- EZ Signal Input
- External Drive Input
- Latch Input Signal
- Stop Signal Input

Motion Alarm Digital Input (ALM)

Property	.CFGFile Abbreviation	Description
CFG_AxAlmEnable	AlmEnable	Enables/disables the alarm function.
CFG_AxAlmLogic	AlmLogic	Sets the active logic of alarm signal.
CFG_AxAlmReact	AlmReact	Sets the stop mode when receiving ALM signal.
CFG_AxALMFilterTime	ALMFilterTime	Sets the filtering time of the axis' ALM signal.

IN Position Signal Input (IN Position)

Property	.CFGFile Abbreviation	Description
CFG_AxInpEnable	InpEnable	Enables/disables the IN-Position function.
CFG_AxInpLogic	InpLogic	Sets the active logic of IN-Position signal.

Positive/Negative Limit Digital Input

Property	.CFGFile Abbreviation	Description
CFG_AxEIEnable	EIEnable	Enables/disables the hardware limit function.
CFG_AxEIReact	EIReact	Sets the reaction mode of EL signal.
CFG_AxEILogic	EILogic	Sets the active logic of hardware limit signal.
CFG_AxLMTFilterTime	LMTFilterTime	Sets the filtering time of the axis' LMT+ signal.
CFG_AxLMTNFilterTime	LMTNFilterTime	Sets the filtering time of the axis' LMT- signal.

ORG Singal Digital Input

Property	.CFGFile Abbreviation	Description
CFG_AxOrgLogic	OrgLogic	Sets the active logic of ORG signal.
CFG_AxEzLogic	OrgReact	Sets the reaction modes for ORG signal.
CFG_AxORGFilterTime	ORGFilterTime	Sets the filtering time for the axis' ORG signal.
PAR_AxHomeCrossDistance	HomeCrossDis	Sets the home cross distance of Home motion.
CFG_AxHomeResetEnable	HomeResetEnable	Enables/disables the logical counter reset function.

PAR_AxHomeExSwitchMode	HomeExSwitchMode	Sets the stopping condition of Home motion.
CFG_AxHomeOffsetDistance	HomeOffsetDistance	Sets the offset distance of Home motion.
CFG_AxHomeOffsetVel	HomeOffsetVel	Sets the offset velocity of Home motion.

EZ Signal Input

Property	.CFGFile Abbreviation	Description
CFG_AxEzLogic	EzLogic	Sets the active logic of EZ signal.

External Drive Input

Property	.CFGFile Abbreviation	Description
CFG_AxExtMasterSrc	ExtMasterSrc	Sets the external drive source.
CFG_AxExtSelEnable	ExtSelEnable	Enables/disables external drive.
CFG_AxExtPulseNum	ExtPulseNum	Sets the pulse count of the external drive.
CFG_AxExtPulseInMode	ExtPulseInMode	Sets the pulse input mode of the external drive.
CFG_AxExtPresetNum	ExtPresetNum	External drive count.

Latch Input Signal

Property	.CFGFile Abbreviation	Description
CFG_AxLatchEnable	LatchEn	Enables/disables latch function.
CFG_AxLatchLogic	LatchLogic	Sets the logic level of latch signal.
CFG_AxLatchBufEnable	LatchBufEnable	Enables/disables continuous latch.
CFG_AxLatchBufMinDistance	LatchBufMinDist	Sets the minimum distance of continuous latch.
CFG_AxLatchBufEventNum	LatchBufEventNum	Sets the number of continuous latch events.
CFG_AxLatchBufSource	LatchBufSource	Sets the source of continuous latch.
CFG_AxLatchBufAxisID	LatchBufAxisID	Continuously latch the axis' ID.
CFG_AxLatchBufEdge	LatchBufEdge	Continuously latch the triggering source.

Stop Signal Input

Property	.CFGFile Abbreviation	Description
CFG_AxIN1StopEnable	IN1StopEnable	Enables/disables IN1 trigger to stop function.
CFG_AxIN1StopLogic	IN1StopLogic	Sets the logic level of IN1 trigger to stop function.
CFG_AxIN1StopReact	IN1StopReact	Sets IN1 trigger to stop mode.

Filtering Signal Input

Property	.CFGFile Abbreviation	Description
CFG_AxIN1FilterTime	IN1FilterTime	Sets the filtering time of the axis' IN1 signal.

Emergency Stop Input

Property	.CFGFile Abbreviation	Description
CFG_DevEmgLogic	EmgLogic	Sets the logic level of the emergency stop signal.

CFG_DevEmgFilterTime	EmgFilterTime	Sets the filtering time of the device's EMG signal.
----------------------	---------------	-----------------------------------------------------

7.2.2 Software Source Configuration

The axis' software source configuration includes:

- Axis' Velocity Configuration
- Axis' PPU
- Axis' Software Limit
- Axis' Backlash
- Axis' Module Number
- Axis' Synchronous Start
- Axis' Velocity Configuration of Home Motion
- JOG Velocity Configuration
- Tolerance
- Axis' Deceleration
- Counting Error

Axis' Velocity Configuration

Property	.CFGFile Abbreviation	Description
PAR_AxVelLow	VelLow	Low velocity of the axis.
PAR_AxVelHigh	VelHigh	High velocity of the axis.
PAR_AxAcc	Acc	Acceleration of the axis.
PAR_AxDec	Dec	Deceleration of the axis.
CFG_AxMaxVel	MaxVel	Sets the max velocity of the axis.
CFG_AxMaxDec	MaxDec	Sets the max deceleration of the axis.
CFG_AxMaxAcc	MaxAcc	Sets the max acceleration of the axis.
CFG_AxMaxJerk	MaxJerk	Sets the max jerk of the axis.

Axis' PPU

Property	.CFGFile Abbreviation	Description
CFG_AxPPU	PlsPerUnit	PPU numerator
CFG_AxPPUDenominator	PPU Denominator	PPU denominator

Axis' Software Limit

Property	.CFGFile Abbreviation	Description
CFG_AxSwMeIEnable	SwMeIEnable	Enables/disables the minus software limit function.
CFG_AxSwPeIEnable	SwPeIEnable	Enables/disables the plus software limit.
CFG_AxSwMeIReact	SwMeIReact	Sets the reacting mode of minus software limit.
CFG_AxSwPeIReact	SwPeIReact	Sets the reacting mode of plus software limit.
CFG_AxSwMeIValue	SwMeIValue	Sets the value of minus software limit.
CFG_AxSwPeIValue	SwPeIValue	Sets the value of plus software limit.

Axis' Backlash

Property	.CFGFile Abbreviation	Description
CFG_AxBacklashEnable	BacklashEnable	Enables/disables corrective backlash.
CFG_AxBacklashPulses	BacklashPulses	Sets the number of compensation pulses.
CFG_AxBacklashVel	BacklashVel	Sets the velocity of corrective backlash.

Axis' Module Number

Property	.CFGFile Abbreviation	Description
CFG_AxModuleRange	ModuleRange	Sets the pulse number when the axis rotates 360 degrees.

Axis' Synchronous Start

Property	.CFGFile Abbreviation	Description
CFG_AxSimStartSource	SimStartSource	Sets simultaneous start/stop mode for current axis.

Axis' Velocity Configuration of Home Motion

Property	.CFGFile Abbreviation	Description
PAR_AxHomeVelLow	HomeVelLow	Sets the low velocity of Home motion.
PAR_AxHomeVelHigh	HomeVelHigh	Sets the high velocity of Home motion.
PAR_AxHomeAcc	HomeAcc	Sets the acceleration of Home motion.
PAR_AxHomeDec	HomeDec	Sets the deceleration of Home motion.
PAR_AxHomeJerk	HomeJerk	Sets the jerk type of Home motion.

JOG Velocity Configuration

Property	.CFGFile Abbreviation	Description
CFG_AxJogVLTime	JogVLTime	Sets the time duration of JOG motion at low velocity (Unit: ms).
CFG_AxJogVelLow	JogVelLow	Sets the low velocity of Jog motion.
CFG_AxJogVelHigh	JogVelHigh	Sets the high velocity of Jog motion.
CFG_AxJogAcc	JogAcc	Sets the acceleration of Jog motion.
CFG_AxJogDec	JogDec	Sets the deceleration of Jog motion.
CFG_AxJogJerk	JogJerk	Sets the jerk type of Jog motion.

Tolerance

Property	.CFGFile Abbreviation	Description
CFG_AxMeiToleranceEnable	MeiToleranceEnable	Enables/disables minus tolerance function.
CFG_AxMeiToleranceValue	MeiToleranceValue	Sets minus tolerance value.
CFG_AxPeiToleranceEnable	PeiToleranceEnable	Enables/disables plus tolerance function.
CFG_AxPeiToleranceValue	PeiToleranceValue	Sets plus tolerance value.
CFG_AxSw MeiToleranceEnable	SwMeiToleranceEnable	Enables/disables software minus tolerance function.

CFG_AxSwMeiToleranceValue	SwMeiToleranceValue	Sets software minus tolerance value.
CFG_AxSwPelToleranceEnable	SwPelToleranceEnable	Enables/disables software plus tolerance function.
CFG_AxSwPelToleranceValue	SwPelToleranceValue	Sets software plus tolerance value.

Axis' Deceleration

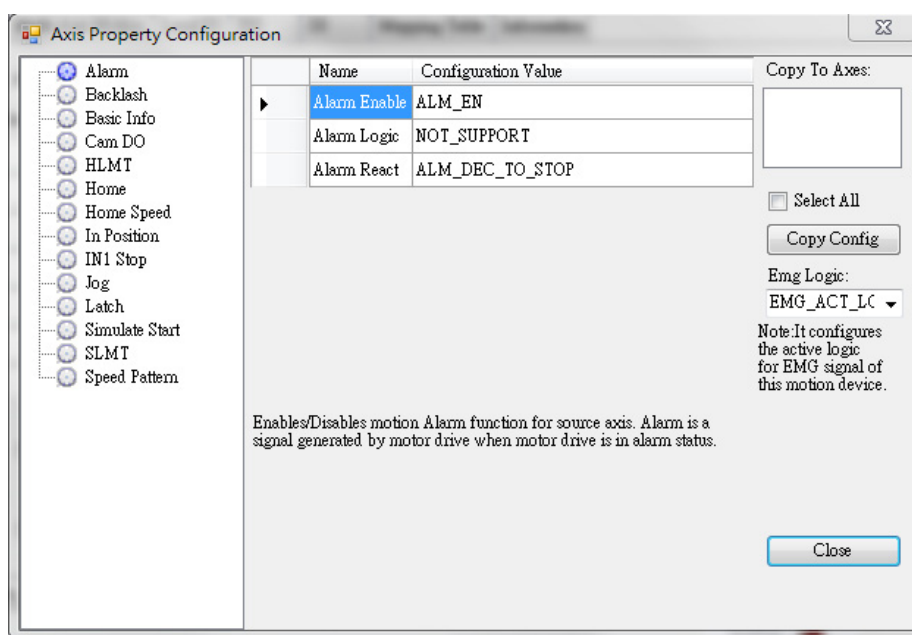
Property	.CFGFile Abbreviation	Description
CFG_AxKillDec	KillDec	Sets the deceleration of DI Stop.

Counting Error

Property	.CFGFile Abbreviation	Description
CFG_AxMaxErrorCnt	MaxErrorCnt	Sets the alarm count for the difference between feedback and Command values.

7.2.3 Resource Integration

The process of combing the above hardware and software resources and then configuring the fundamental properties of all resources is called resource integration, which can be applied to satisfy different function needs. The properties can be configured through testing utility as below. Refer to the Appendix for property configuration description.



```

62007000 - Notepad
File Edit Format View Help
[[Device Config]
EmgLogic=0
EmgFilterTime=0
[Group Config]
GpvelLow=1000
GpvelHigh=8000
GpAcc=50000
GpDec=50000
GpJerk=0
SFEnable=0
BldTime=0
[AXIS 0 Config]
PlsPerUnit=1
Maxvel=200000000
MaxAcc=2000000000
MaxDec=2000000000

```

7.3 Generating and Downloading Configuration Files

Hardware and software resources can be configured through the utility. The configuration file can be saved when the “Save” button in Utility interface is clicked.

At the same time, by clicking on the “Load” button, the configuration file will be loaded to the utility so that the parameter settings in the configuration file are enabled.

7.3.1 Downloading Function of the Configuration File

By calling the following API, the user can import the configuration file into his/her own project.

Property	Description
Acm_DevLoadConfig	Configures all settings of the device according to the loaded configuration file.

7.3.2 Description of Important Information in Configuration Files

As described above, the user configures property files through the utility and imports configuration files into the project by calling the corresponding function. The configuration file could be binary or text files. If the filename extension is “.bin”, then the driver will read the file as a binary file. Or else, the driver will read the file as an “.INI” (text) file.

The following are configuration files generated through the utility. “AXIS 0 Config” indicates the property configuration of Axis 0. For example, PCI-1245 has 4 axes, so that the configuration file includes the property configuration information of AXIS 0 Config ~ AXIS 3 Config. Refer to Chapter 7.2 for detailed information about the corresponding properties of parameter names in the configuration file.

The user can directly modify the corresponding property value in the configuration file, e.g.: if MaxAcc=4,000,000, the user can import the modified configuration file by calling Acm_DevLoadConfig function; the property value will then be valid and the board's max acceleration will be set to 4,000,000.

7.4 Modifying Configuration Information

Besides the configuration file, the user can also modify the corresponding configuration information by properties listed in Chapter 7.2.

7.5 Configuring Initialization Status through Controller

7.5.1 Configuring Initialization Status with Hardware Resource

The following list indicates how to configure initialization status with the controller's hardware resource.

Resource	Configuration Option	Default Status	Relative Command
Board	Emergency stop input level	High Logic Level	CFG_DevEmgLogic
	EMG signal filtering time	5us	CFG_DevEmgFilterTime
Cam DO	Enables/disables DO.	Disabled	CFG_AxCamDOEnable
	Sets the low limit for CAM DO signal.	10,000	CFG_AxCamDOLoLimit
	Sets the high limit for CAM DO signal.	10,000	CFG_AxCamDOHiLimit
	Sets the compare source for CAM DO signal.	Theoretical position	CFG_AxCamDOCmpSrc
	Cam logic level.	High Logic Level	CFG_AxCamDOLogic
	Error Clear Signal Logic Level	High Logic Level	CFG_AxErcLogic
Clear Signal Output of Axis Alarm	Enables/disables clear function.	Disabled	CFG_AxErcEnableMode
	Sets the On time of error clear function.	12us	CFG_AxErcOnTime
	Sets the Off time of error clear function.	0us	CFG_AxErcOffTime
Axis' General DO Function	Enables/disables the axis' general DO.	Disabled	CFG_AxGenDoEnable
	Enables/disables the alarm function.	Disabled	CFG_AxAlmEnable
Motion Alarm Digital Input	Sets the active logic of alarm signal.	High Logic Level	CFG_AxAlmLogic
	Sets the stop mode when receiving ALM signal.	Decelerate and stop	CFG_AxAlmReact
	Sets the filtering time of the axis' ALM signal.	5us	CFG_AxALMFilterTime
	Enables/disables the IN-Position function.	Disabled	CFG_AxInpEnable
IN Position Signal Input	Sets the active logic of IN-Position signal.	High Logic Level	CFG_AxInpLogic
	Enables/disables the hardware limit function.	Enabled	CFG_AxEIEnable
Positive/Negative Limit Digital Input	Sets the reaction mode of EL signal.	Stop immediately	CFG_AxEIReact
	Sets the active logic of hardware limit signal.	Low Logic Level	CFG_AxEILogic
	Sets the filtering time of the axis' LMT+ signal.	5us	CFG_AxLMT+FilterTime
	Sets the filtering time of the axis' LMT- signal.	5us	CFG_AxLMT-FilterTime

ORG Signal Digital Input	Sets the active logic of ORG signal.	Low Logic Level	CFG_AxOrgLogic
	Sets the reaction modes for ORG signal.	Decelerate and stop	CFG_AxEzLogic
	Sets the home cross distance of Home motion.	10,000	PAR_AxHomeCrossDistance
	Enables/disables the logical counter reset function.	Enabled	CFG_AxHomeResetEnable
	Sets the stopping condition of Home motion.	Decelerate and stop	PAR_AxHomeExSwitchMode
EZ Signal Input	Sets the active logic of EZ signal.	Low Logic Level	CFG_AxEzLogic
External Drive Input	Sets the external drive source.	Axis 0	CFG_AxExtMasterSrc
	Enables/disables external drive.	Disabled	CFG_AxExtSelEnable
	Sets the pulse count of the external drive.	1	CFG_AxExtPulseNum
	Sets the pulse input mode of the external drive.	4XAB	CFG_AxExtPulseInMode
	External drive count.	1	CFG_AxExtPresetNum
Latch Input Signal	Enables/disables latch function.	Disabled	CFG_AxLatchEnable
	Sets the logic level of latch signal.	High Logic Level	CFG_AxLatchLogic
	Enables/disables continuous latch.	Disabled	CFG_AxLatchBufEnable
	Sets the minimum distance of continuous latch.	1,000	CFG_AxLatchBufMinDistance
	Sets the number of continuous latch events.	128	CFG_AxLatchBufEventNum
	Sets the source of continuous latch.	Theoretical position	CFG_AxLatchBufSource
	Continuously latch the axis' ID.	Axis 0	CFG_AxLatchBufAxisID
	Continuously latch the triggering source.		CFG_AxLatchBufEdge
Emergency Stop Signal	Enables/disables IN1 trigger to stop function.	Disabled	CFG_AxIN1StopEnable
	Sets the stop mode of IN1 triggering.	Decelerate and stop	CFG_AxIN1StopLogic
	Sets the logic level of IN1 trigger to stop function.	High Logic Level	CFG_AxIN1StopReact
	Sets the filtering time of the axis' IN1 signal.	5us	CFG_AxIN1FilterTime

7.5.2 Configuring Initialization Status with Software Resource

The following list indicates how to configure initialization status with the controller's software resource.

Resource	Configuration Option	Default Status	Relative Command
Axis' Velocity Parameters	Low velocity of the axis.	2,000	PAR_AxVelLow
	High velocity of the axis.	8,000	PAR_AxVelHigh
	Acceleration of the axis.	10,000	PAR_AxAcc
	Deceleration of the axis.	10,000	PAR_AxDec
	Sets the max velocity of the axis.	20,000,000	CFG_AxMaxVel
	Sets the max deceleration of the axis.	2,000,000,000	CFG_AxMaxDec
	Sets the max acceleration of the axis.	2,000,000,000	CFG_AxMaxAcc
Axis' PPU	Sets the max jerk of the axis.	1	CFG_AxMaxJerk
	PPU numerator	1	CFG_AxPPU
	PPU denominator	1	CFG_AxPPUDenominator
Axis' Software Limit	Enables/disables the minus software limit function.	Enabled	CFG_AxSwMelEnable
	Enables/disables the plus software limit.	Enabled	CFG_AxSwPelEnable
	Sets the reacting mode of minus software limit.	Decelerate and stop	CFG_AxSwMelReact
	Sets the reacting mode of plus software limit.	Decelerate and stop	CFG_AxSwPelReact
	Sets the value of minus software limit.	-10,000,000	CFG_AxSwMelValue
	Sets the value of plus software limit.	10,000,000	CFG_AxSwPelValue
Axis' Backlash	Enables/disables corrective backlash.	Disabled	CFG_AxBacklashEnable
	Sets the number of compensation pulses.	10	CFG_AxBacklashPulses
	Sets the velocity of corrective backlash.	1,000	CFG_AxBacklashVel
Axis' Module Number	Sets the module number range.	0	CFG_AxModuleRange
Simultaneous Start and Stop	Sets simultaneous start/Stop mode for current axis.	Disabled	CFG_AxSimStartSource
Home Motion Velocity Parameters	Sets the low velocity of Home motion.	2,000	PAR_AxHomeVelLow
	Sets the high velocity of Home motion.	8,000	PAR_AxHomeVelHigh
	Sets the acceleration of Home motion.	10,000	PAR_AxHomeAcc
	Sets the deceleration of Home motion.	10,000	PAR_AxHomeDec
	Sets the jerk type of Home motion.	T type	PAR_AxHomeJerk

JOG Motion Velocity Parameters	Sets the time duration of JOG motion at low velocity (Unit: ms).	5,000	CFG_AxJogVlTime
	Sets the low velocity of Jog motion.	2,000	CFG_AxJogVelLow
	Sets the high velocity of Jog motion.	8,000	CFG_AxJogVelHigh
	Sets the acceleration of Jog motion.	10,000	CFG_AxJogAcc
	Sets the deceleration of Jog motion.	10,000	CFG_AxJogDec
	Sets the jerk type of Jog motion.	T type	CFG_AxJogJerk
Tolerance	Enables/disables minus tolerance function.	Disabled	CFG_AxMelToleranceEnable
	Sets minus tolerance value.	5,000	CFG_AxMelToleranceValue
	Enables/disables plus tolerance function.	Disabled	CFG_AxPelToleranceEnable
	Sets plus tolerance value.	5,000	CFG_AxPelToleranceValue
	Enables/disables software minus tolerance function.	Disabled	CFG_AxSwMelToleranceEnable
	Sets software minus tolerance value.	5,000	CFG_AxSwMelToleranceValue
	Enables/disables software plus tolerance function.	Disabled	CFG_AxSwPelToleranceEnable
	Sets software plus tolerance value.	5,000	CFG_AxSwPelToleranceValue
Axis' Deceleration	Sets the deceleration of DI Stop.	100,000	CFG_AxKillDec
Counting Error Alarm	Sets the alarm count for the difference between feedback and Command values.	0(no alarm)	CFG_AxMaxErrorCnt

7.6 Servo Operation

The Open/Close Servo Motor function is as below:

Function	Description
Acm_AxSetSvOn	Opens/closes servo driver.

By calling Acm_AxSetSvOn function, the user can open the servo enabling signal of the motor connected to the specified controlling axis which will turn into controlling status then.

Below is the example:

```
U32    Result;
HAND   m_Axishand[32]; // Axis' handle
Result =Acm_AxSetSvOn(m_Axishand[0], 0); //Open the servo of axis 0
Result =Acm_AxSetSvOn(m_Axishand[0], 1); // Open the servo of axis 0
```

7.7 Floating Point PPU

7.7.1 Properties of Floating Point PPU

The properties of floating point PPU is as below:

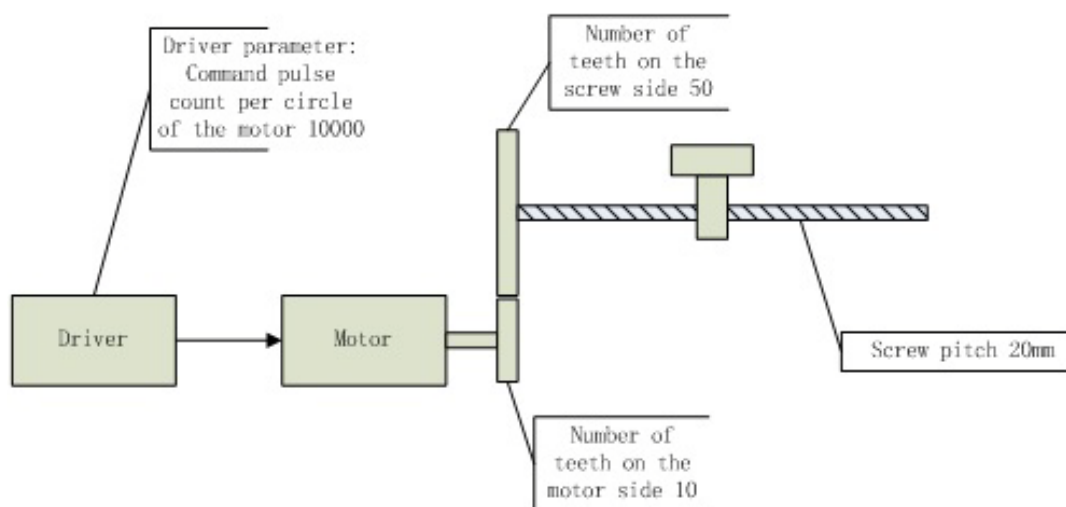
Property	Description
CFG_AxPPU	Sets PPU numerator
CFG_AxPPUDenominator or	Sets PPU denominator.

7.7.2 Description of Floating Point PPU Functions

PPU: Virtual unit of the board, enabling pulse output of different mechanisms.

PPU Value = CFG_AxPPUProperty Value/CFG_AxPPUDenominator Property Value

Refer to the following mechanism:



The PPU is derived for the above mechanism and driver parameters as below:

Mechanism conditions:

Screw pitch: 20mm

If the number of teeth on the screw side is 50 and the number of teeth on the motor side is 10, then the motor needs to spin $50/10=5$ circles when the screw spins 1 circle.

For the driver parameter, the command pulse count per circle of the motor is 10,000.

If the user configures the unit as 1mm, then the following formula is concluded according to the above mechanism conditions:

$$\text{PPU} = (10000 \times 50) / (10 \times 20)$$

The simplified formula is:

$$\text{PPU} = 500000 / 200 = 2500$$

i.e., the number of the board's output pulse should be 2,500 per millimeter as the virtual mechanism moves.

Therefore, the user could set:

CFG_AxPPUProperty value to 2,500

CFG_AxPPUDenominatorProperty value to 1

Floating point PPU is configured when the corresponding PPU of the user's distance unit mustn't be an integer.

For example, if there should be 1,005 pulses if the user moves 10mm, then the pulse count will be 100.5 per 1mm. To fix the distance difference, the PPU denominator

could be set to get the actual PPU of double type for calculations of distance, speed and so on.

7.7.3 Description of Floating Point PPU

The properties of PPU numerator and PPU denominator should be predefined, or else the other properties, such as HomeCrossDistance, will be impacted. PPU numerator and PPU denominator can be used when the actual PPU is a floating point number.

The change of the property's value will have impact on CFG_AxMaxVel, CFG_AxMaxAcc, CFG_AxMaxDec, PAR_AxVelHigh, PAR_AxVelLow, PAR_AxAcc, PAR_AxDec, PAR_GpVelHigh, PAR_GpVelLow, PAR_GpAcc, PAR_GpDec and PAR_HomeCrossDistance, as well as all moving distances (the value of the above are all calculated by the actual PPU).

Both the values of the numerator and the denominator are 1 as default.

7.7.4 Example

To set the actual PPU of axis X to 100.5 pulses:

```
U32 Result;
```

```
Result =Acm_SetU32Property(m_Axishand[0], CFG_AxPPU, 201);
```

```
Result =Acm_SetU32Property(m_Axishand[0], CFG_AxPPUDenominator, 2);
```

Chapter 8

Motion Status

8.1 Introduction

Once the motion controller, driver and motor are connected, the user can get the system's motion status and speed parameters, such as current position, motion speed, acceleration and deceleration, by calling functions. This chapter demonstrates how to get motion information of the board.

8.2 Axis Status

8.2.1 Axis Status Functions

The user can read axis status from the board's status memory so as to get axis status functions. Refer to the following table:

Function	Description
Acm_AxGetState	Gets states of axis.
Acm_AxGetMotionStatus	Gets status of current motion.

8.2.2 Description

8.2.2.1 Current Axis Status

When the user gets the axis' current status by calling Acm_AxGetState, a 16-bit word of axis status will be returned.

The following table defines axis status.

Table 8.1: Definition of Axis' Current Status

Value	Definition
0	STA_AxDisable. Axis is disabled, you can open it to active it.
1	STA_AxReady. Axis is ready and waiting for new command.
2	STA_Stopping. Axis is stopping.
3	STA_AxErrorStop. Axis has stopped because of error.
4	STA_AxHoming. Axis is executing home motion.
5	STA_AxPtpMotion. Axis is executing PTP motion.
6	STA_AxContiMotion. Axis is executing continuous motion.
7	STA_AxSyncMotion. Axis is in one group and the group is executing interpolation motion, or axis is slave axis in E-cam/E-gear/Gantry motion.
8	STA_AX_EXT_JOG. Axis is controlled by external signal and will execute JOG mode motion once external signal is active.
9	STA_AX_EXT_MPG. Axis is controlled by external signal and will execute MPG mode motion once external signal is active.

The axis status should be Ready when the user opens board and axis. The new motion operation (e.g., continuous motion) could be implemented only when the axis status is Ready.

When the board is opened while the axis is not opened yet, the axis status will be disabled (STA_AxDisable); if the user tries to implement motion operation, an error will occur.

8.2.2.2 Axis' Current Motion Status

When the user gets the axis' current motion status by calling `Acm_AxGetMotionStatus`, a 32-bit value of axis' current motion status will be returned. The following shows the definition of axis' current motion status.

Table 8.2: Definition of Axis' Current Motion Status

Bit	Definition
0	Stop. Stop
1	Res1. Reserved
2	WaitERC. Wait ERC finished
3	Res2. Reserved
4	CorrectBksh. Correcting Backlash;
5	Res3. Reserved
6	InFA. Feeding in return velocity= FA
7	InFL. Feeding in StrVel speed=FL;
8	InACC. Accelerating
9	WaitINP. Wait in position.

8.2.3 Example

The following example shows how to get the axis' current status when the board and axis are opened.

```

HAND  m_Axishand;//Axis 0's handle
U32   Ret; //Return value of the function
U16   State;
CString strTemp;
— Initialization process —
Ret =Acm_AxGetState(m_Axishand,&State);
if (Ret ==SUCCESS)
{switch (State)
    {
        case 0:
            strTemp.Format("STA_AX_DISABLE");
            break;
        case 1:
            strTemp.Format("STA_AX_READY");
            break;
        case 2:
            strTemp.Format("STA_AX_STOPPING");
            break;
        case 3:
            strTemp.Format("STA_AX_ERROR_STOP");
            break;
        case 4:
            strTemp.Format("STA_AX_HOMING");
            break;
        case 5:
            strTemp.Format("STA_AX_PTP_MOT");

```

```

        break;
    case 6:
        strTemp.Format("STA_AX_CONTI_MOT");
        break;
    case 7:
        strTemp.Format("STA_AX_SYNC_MOT");
        break;
    case 8:
        strTemp.Format("STA_AX_EXT_JOG ");
        break;
    case 9:
        strTemp.Format("STA_AX_EXT_MPG ");
        break;
    default:
        break;
    }
}

```

8.3 Axis Velocity

8.3.1 Function and Property

The corresponding functions and properties of axis velocity are shown in the table below:

Function	Description
Acm_AxChangeVel	To command axis to change the velocity while axis is in velocity motion.
Acm_AxGetCmdVelocity	Get current command velocity of the specified axis.
Acm_AxChangeVelEx	Get moving velocity, acceleration and deceleration of the motion axis.
Acm_AxChangeVelExByRate	Proportionally change moving velocity, acceleration and deceleration of the motion axis.
Acm_AxChangeVelByRate	Proportionally change the velocity of the motion axis.
Feature	Description
FT_AxMaxVel	Get axis supported max velocity.
FT_AxMaxAcc	Get axis supported max acceleration.
FT_AxMaxDec	Get axis supported max deceleration.
FT_AxMaxJerk	Get axis supported max jerk.
Configuration	Description
CFG_AxMaxVel	Configure the max velocity for the motion axis.
CFG_AxMaxAcc	Configure the max acceleration for the motion axis.
CFG_AxMaxDec	Configure the max deceleration for the motion axis.
CFG_AxMaxJerk	Get max jerk configuration for the motion axis.
Parameter	Description
PAR_AxVelLow	Set/get initial velocity of axis.
PAR_AxVelHigh	Set/get operating velocity of axis.
PAR_AxAcc	Set/get acceleration of this axis.

PAR_AxDec	Set/get deceleration of this axis.
PAR_AxJerk	Set/get the type of velocity profile: T-Curve or S-Curve

8.3.2 Description

When board and axis are opened, the user can set/get the axis' current velocity values including initial velocity, acceleration and moving velocity. Please refer to the corresponding chapter for setting and getting properties.

The specified velocity parameter value shouldn't greater than the axis velocity parameter value of the motion axis, or else an error will occur. For example, if the max velocity of the axis(CFG_AxMaxVel) is set to 10,000, then an error will occur when the user tries to set the moving velocity(PAR_AxVelHigh) to 11,000.

During the motion process of the axis, the user can change the motion axis' moving velocity, acceleration and deceleration by calling functions.

8.3.3 Example

The following example sets the velocity parameters of axis 0 and changes the moving velocity during the motion process.

The VC codes are as below:

```

HAND m_Axishand; //Axis 0's handle
U32 Ret;
— Initialization process —
//Sets velocity parameters by codes below:
Ret =Acm_SetF64Property(m_Axishand,PAR_AxVelLow,2000);//Initial velocity is
2000
Ret = Acm_SetF64Property(m_Axishand,PAR_AxVelHigh,8000); //Moving velocity
is 8000
Ret =Acm_SetF64Property(m_Axishand,PAR_AxAcc,10000);//Acceleration is
10000
Ret =Acm_SetF64Property(m_Axishand,PAR_AxDec,10000);//Deceleration is
10000
Ret =Acm_SetF64Property(m_Axishand,PAR_AxJerk,0);//T-type curve
//Implement point-to-point motion and change the motion velocity during motion pro-
cess
Ret = Acm_AxMoveRel(m_Axishand,10000);//Relative to point-to-point motion, the
target position is 10000
Ret = Acm_AxChangeVel(m_Axishand, 6000);//Change the motion velocity to
6000PPU/S

```

8.4 Axis' Motion I/O Status

8.4.1 Function

The user can get the axis' motion I/O status by the following function:

Function	Description
Acm_AxGetMotionIO	Get the motion I/O status of the axis.

8.4.2 Description

The user can get the axis' motion I/O status by calling Acm_AxGetMotionIO; a 32-bit word of axis status will be returned then. Please refer to the axis status word definition below:

Bit	Definition
0	RDY-- RDY pin input
1	ALM -- Alarm signal input
2	LMT+-- Limit switch+
3	LMT-- Limit switch-
4	ORG-- Original switch
5	DIR -- DIR output
6	EMG -- Emergency signal input
7	PCS -- PCS signal input
8	ERC -- Output offset counter clear signal to servo motor drvier
9	EZ -- Encoder Z signal
10	CLR -- External output toward clear position counter
11	LTC -- Latch signal input
12	SD -- Deceleration signal input
13	INP -- In-position signal input
14	14: SVON -- Servo On (OUT6)
15	ALRM -- Alarm reset output status
16	SLMT+ -- Softwarte limit+
17	SLMT-- Softwarte limit -
18	CMP--Compare signal
19	CAMDO -- Cam DO

The value will not return to zero when driver alarm tag and limit trigger tag are triggered. When the cause of exception is cleared, the axis status will turn to Ready if Acm_AxResetError is called.

8.4.3 Example

The following example demonstrates how to get the motion I/O status of axis 0 when board and axis are opened, and detect signals such as alarm, home, left limit and right limit. Please refer to VC codes below:

```
HAND  m_Axishand;//Axis 0's handle
U32   Ret; //Return value of the function
U16   Status;
CString strTemp;
—Intialization process —
Ret =Acm_AxGetMotionIO (m_Axishand,& Status);
if (Ret ==SUCCESS)
{
    if (Status& AX_MOTION_IO_ALM)//ALM
    { //Alarm signal output}
    if (Status &AX_MOTION_IO_ORG)//ORG
    { //Home motion signal output}
    if (Status &AX_MOTION_IO_LMTP)//+EL
    { //Right limit signal output}
    if (Status & AX_MOTION_IO_LMTN)//-EL
    { //Left limit signal output}
}
}
```

8.5 Group Status

8.5.1 Function

The user can read group status from the board's status memory and get the corresponding functions as shown below:

Function	Description
Acm_GpGetState	Gets the group's current status.

8.5.2 Description

A 16-bit group status word will be returned when the user get the group's current status by calling Acm_GpGetState. The definition of group status is as below:

Table 8.3: Definition of Group's Current Status

Bit	Definition
0	STA_GP_DISABLE. Group status is Disable: group motion is disabled.
1	STA_GP_READY. Group is ready for implementing new command.
2	STA_GP_STOPPING. Group is stopping.
3	STA_GP_ERROR_STOP. Group stops because of error.
4	STA_GP_MOTION. Group is in implementing a motion.
5	STA_GP_AX_MOTION (Not supported)
6	STA_GP_MOTION_PATH. Group is implementing Path motion..

When board and axis are opened, the user has to add the axis to the group by calling the certain function. If the number of axes in group is greater than 2, then the group status should be Ready for group operations. However, if group status is Disable, an error will occur when the user implements group operation.

8.5.3 Example

The following example adds the axis to group after opening board and axis to get the group's current status.

```
HAND m_GpHand;//Group handle
— Intialization process —
U32 Ret; //Return value of the function
U16 GpState;
CString strTemp;
Ret =Acm_GpGetState(m_GpHand, &GpState);
if (Ret ==SUCCESS)
{
    switch (GpState)
    {
    case 0:
        strTemp.Format("STA_GP_DISABLE ");
        break;
    case 1:
        strTemp.Format("STA_GP_READY ");
        break;
    case 2:
        strTemp.Format("STA_GP_STOPPING ");
        break;
    case 3:
        strTemp.Format("STA_GP_ERROR_STOP ");
        break;
    case 4:
        strTemp.Format("STA_GP_AX_MOTION ");
        break;
    case 5:
        strTemp.Format("STA_GP_MOTION_PATH ");
        break;
    }
}
```

8.6 Group Velocity

8.6.1 Function and Property

The following table lists group velocity functions and properties.

Function	Description
Acm_GpGetCmdVel	Get current velocity of the group.
Parameter	Description
PAR_GpVelLow	Set/get initial velocity (starting velocity) of the axis.
PAR_GpVelHigh	Set/get the motion velocity of the axis.
PAR_GpAcc	Set/get the acceleration of the axis.
PAR_GpDec	Set/get the deceleration of the axis.
PAR_GpJerk	Set/get the type of velocity curve: T/S type curve.

8.6.2 Description

After the board is initialized, the user can set/get the current value of velocity parameters, such as initial velocity, acceleration and motion velocity. In motion process, the user can get the current velocity value of group by calling Acm_GpGetCmdVel.

8.6.3 Example

The following example demonstrates how to set velocity parameters of group and get the motion velocity of group in motion process.

Please refer to the VC codes below:

```

HAND m_GpHand;
U32 Ret;
U32 m_CW=0 //Implement clockwise circular motion
F64 CenterArray[2]={8000,0};
F64 EndArray[2]={16000,0};
U32 AxisNum =2; // Motion axis number
F64 CmdVel; //Group motion velocity
— Intialization process —
//The following codes set velocity paramters:
Ret =Acm_SetF64Property(m_GpHand,PAR_GpVelLow,2000);//Initial velocity
2000
Ret = Acm_SetF64Property(m_GpHand,PAR_GpVelHigh,8000); //Motion velocity
8000
Ret =Acm_SetF64Property(m_GpHand,PAR_GpAcc,10000);//Acceleration 10000
Ret =Acm_SetF64Property(m_GpHand,PAR_AxDec,10000);//Deceleration 10000
//Implement circular motion
Ret=Acm_GpMoveCircularRel(m_GpHand,CenterArray,EndArray,&Axis-
Num,m_CW);
//Get group velocity in motion process
Ret = Acm_GpGetCmdVel(m_GpHand, &CmdVel);

```


Chapter 9

Motion API

9.1 Introduction

This chapter describes the single-axis motion, interpolation motion, following motion, path motion.

9.2 Single-Axis Motion

9.2.1 About this Section

Single-axis motion mode means to plan single-axis movement, including the following motion mode:

- Point to point motion
- Continuous motion
- Change position motion
- Change velocity motion
- Simultaneous start/stop motion
- Imposed motion
- JOG move
- Homing

9.2.2 Point to Point Motion

9.2.2.1 Function and Property

Point to point motion functions are shown in Table 9.1, the functions in table can be called directly in program.

Table 9.1: Point to point motion functions

Function	Description
Acm_AxMoveAbs	Command absolute point-to-point motion
Acm_AxMoveRel	Command relative point-to-point motion
Acm_AxSetCmdPosition	Set axis command position
Acm_AxGetCmdPosition	Get axis command position
Acm_AxGetActualPosition	Get axis actual (feedback) position
Acm_AxGetCmdVelocity	Get axis command velocity
Acm_AxStopDec	Decelerated stop
Acm_AxStopEmg	Emergency stop (No deceleration)
Acm_AxStopDecEx	Command the axis to stop and specify the deceleration.
Acm_AxGetState	Get states of axis
Acm_AxResetError	Reset error when axis is error-stop
Acm_SetU32Property	Set property (Unsigned 32 -bit integer)
Acm_SetI32Property	Set property (Signed 32 -bit integer)
Acm_SetF64Property	Set property (Double)
Acm_GetU32Property	Get property (Unsigned 32 -bit integer)
Acm_GetI32Property	Get property (Signed 32 -bit integer)
Acm_GetF64Property	Get property (Double)

Point to point motion related properties as shown in Table 9.2, the property can't be called directly, but set and get property values by calling related property function. Refer to the Appendix for more detail information.

Table 9.2: Point to point motion properties

Parameter	Description
PAR_AxVelLow	Set/get initial velocity of the axis
PAR_AxVelHigh	Set/get operating velocity of the axis
PAR_AxAcc	Set/get acceleration of this axis
PAR_AxDec	Set/get deceleration of this axis
PAR_AxJerk	Set/get the type of velocity profile: T-Curve or S-Curve
Configuration	Description
CFG_AxMaxVel	Configure the max velocity for the motion axis
CFG_AxMaxAcc	Configure the max acceleration for the motion axis
CFG_AxMaxDec	Configure the max deceleration for the motion axis

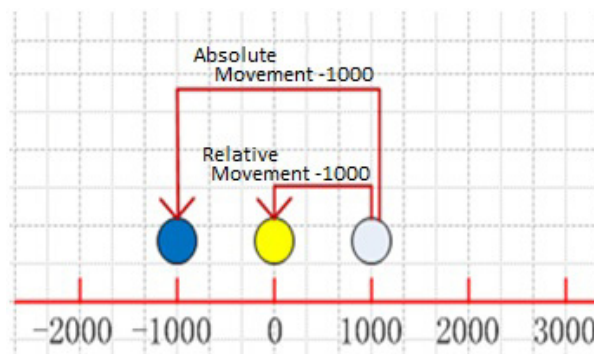
9.2.2.2 Description

When the status of the axis is Ready, it will be able to execute point to point movement. Refer to section 8.2 for axis status.

Point to point movement is divided into relative and absolute motions.

Relative movement: Move position by set the current command position as a reference position

Absolute movement: Move position by set absolute zero position as a reference position.



As shown above, the current control unit stops in the axis position 1000 pulse, If commands the relative motion -1000 pulse, the control unit will stop at 0 pulse position; If commands the absolute motion -1000 pulse, the control unit will stop at -1000 pulse position. It means that the control unit moves 2000 pulse distance in the negative direction.

Users can set the target position, initial velocity and acceleration of each axis individually. Each axis move or stop independently and can change the target position and speed during moving period.(section 9.2.4 and 9.2.5).

9.2.2.3 Flow Chart

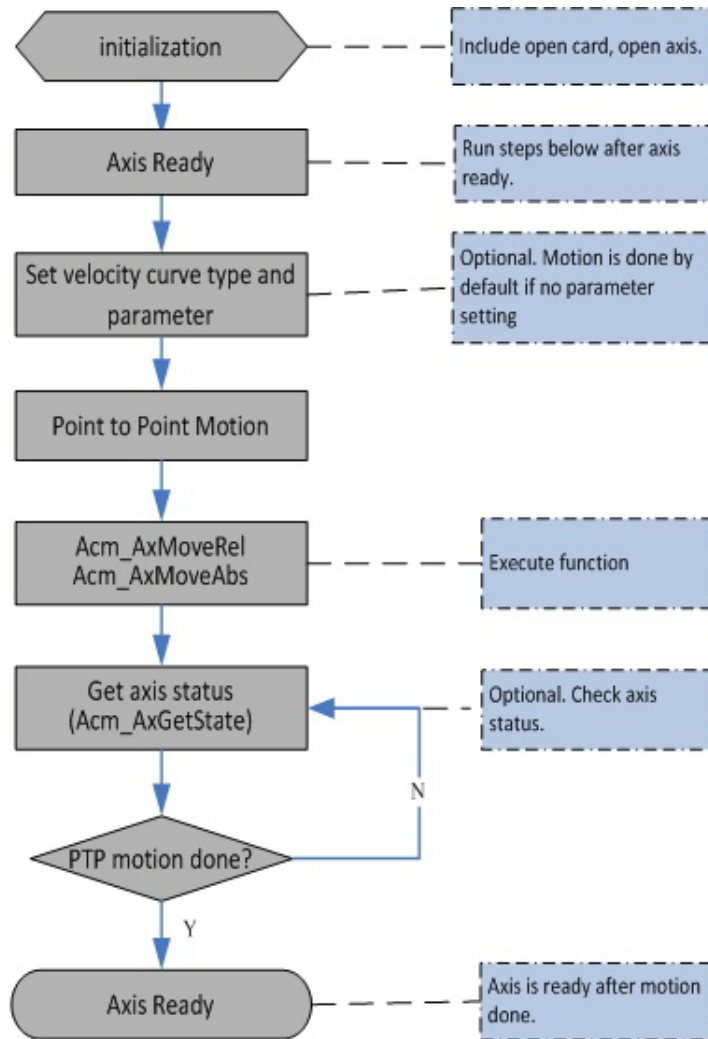


Figure 9.1 Point to point motion flow chart

9.2.2.4 Example

This example executes relative point to point motion on 0 axis. Refer to the table below for setting parameters.

Function	Executes relative point to point motion on 0 axis (PCI-1203)
Initial velocity	2000PPU/S
Operating velocity	8000PPU/S
Acceleration	10000PPU/S ²
Deceleration	10000PPU/S ²
Velocity curve	T-Curve
Target position	10000PPU

VC code:

```

HAND m_Axishand[32]; //Axis handle
U32 Ret; //Function return value
---Initialization---
//Set parameters
Ret =Acm_SetF64Property(m_Axishand[0],PAR_AxVelLow,2000); //Initial velocity
Ret = Acm_SetF64Property(m_Axishand[0],PAR_AxVelHigh,8000); //Operating
velocity
Ret =Acm_SetF64Property(m_Axishand[0],PAR_AxAcc,10000); //Acceleration
Ret =Acm_SetF64Property(m_Axishand[0],PAR_AxDec,10000); //Deceleration
Ret =Acm_SetF64Property(m_Axishand[0],PAR_AxJerk,0); //T-Curve
//Execute Point to Point motion
Ret =Acm_AxMoveRel(m_Axishand[0],10000); //Relative Point to Point motion
//Get axis status and position
F64 CmdPos;
F64 ActPos;
U16 State;
Acm_AxGetCmdPosition(m_Axishand[0],&CmdPos); //Get 0 axis command posi-
tion
Acm_AxGetActualPosition(m_Axishand[0],&ActPos); //Get 0 axis actual position
Acm_AxGetState(m_Axishand[0],&State); //Get 0 axis status

```

Check function returned value to judge the function execution status and establish error handle mechanism is recommended to ensure the correct result. Refer to the PTP example.

9.2.3 Continuous Motion

9.2.3.1 Function and Property

Continuous motion functions are shown in Table 9.3, the functions in table can be called directly in program.

Table 9.3: Continuous motion functions

Function	Description
Acm_AxMoveVel	To command axis to make a never ending movement with a specified velocity
Acm_AxSetCmdPosition	Set axis command position
Acm_AxGetCmdPosition	Get axis command position
Acm_AxSetActualPosition	Set axis actual (feedback) position
Acm_AxGetActualPosition	Get axis actual (feedback) position
Acm_AxGetCmdVelocity	Get axis command velocity
Acm_AxStopDec	Decelerated stop
Acm_AxStopEmg	Emergency stop (No deceleration)
Acm_AxStopDecEx	Command the axis to stop and specify the deceleration.
Acm_AxGetState	Get states of axis
Acm_AxResetError	Reset error when axis is error-stop
Acm_SetU32Property	Set property (Unsigned 32 -bit integer)
Acm_SetI32Property	Set property (Signed 32 -bit integer)
Acm_SetF64Property	Set property (Double)
Acm_GetU32Property	Get property (Unsigned 32 -bit integer)
Acm_GetI32Property	Get property (Signed 32 -bit integer)
Acm_GetF64Property	Get property (Double)

Continuous related properties as shown in Table 9.3, the property can't be called directly, but set and get property values by calling related property function. Refer to Appendix for more detail information.

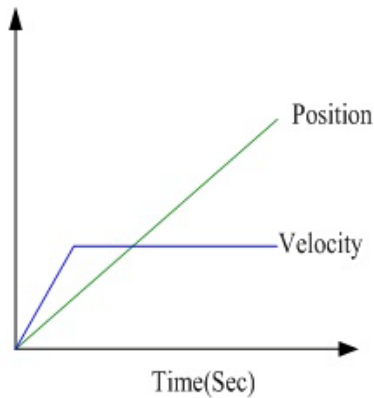
Table 9.4: Continuous motion properties

Parameter	Description
PAR_AxVelLow	Set/get initial velocity of axis
PAR_AxVelHigh	Set/get operating velocity of axis
PAR_AxAcc	Set/get acceleration of this axis
PAR_AxDec	Set/get deceleration of this axis
PAR_AxJerk	Set/get the type of velocity profile: T-Curve or S-Curve
Configuration	Description
CFG_AxMaxVel	Configure the max velocity for the motion axis
CFG_AxMaxAcc	Configure the max acceleration for the motion axis
CFG_AxMaxDec	Configure the max deceleration for the motion axis

9.2.3.2 Description

Continuous motion command movement of axis follows the specified velocity and direction to make a never ending motion.

As shown in the figure below:



Users can set the target position, initial velocity and acceleration of each axis individually. Each axis move or stop independently and can change the speed during moving period. (Section 9.2.5).

When performing continuous motion, status of the axis must be ready. The status of axis status can be accessed by `Acm_AxGetState`. Refer to chapter 8.2.2.1 of Motion Status.

9.2.3.3 Flow Chart

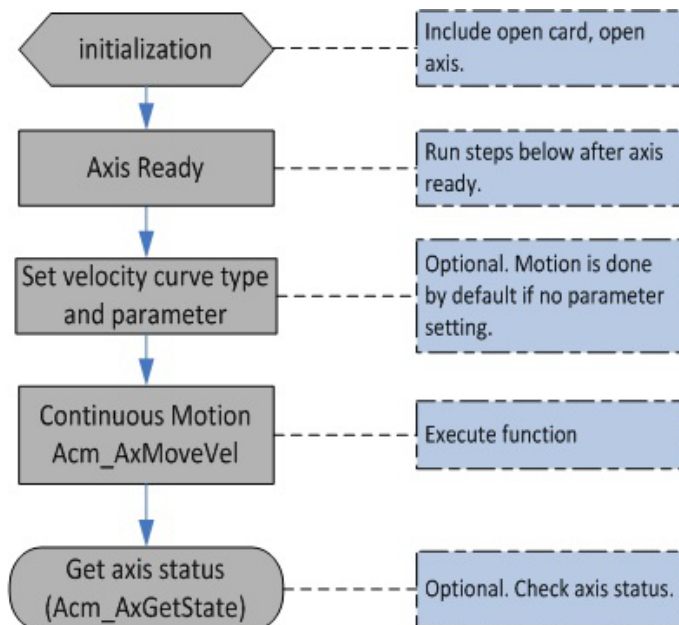


Figure 9.2 Continuous motion flow chart

9.2.3.4 Example

This example executes continuous motion on 0 axis. Refer to the table below for setting parameters.

Function	Executes continuous motion on 0 axis (PCI-1203)
Initial velocity	2000PPU/S
Operating velocity	8000PPU/S
Acceleration	10000PPU/S ²
Deceleration	10000PPU/S ²
Velocity curve	T-Curve
Move direction	Positive

VC code:

```
HAND m_Axishand[32]; //Axis handle
U32 Ret; // Function return value
--- Initialization---
// Set parameters
Ret =Acm_SetF64Property(m_Axishand[0],PAR_AxVelLow,2000); // Initial velocity
Ret = Acm_SetF64Property(m_Axishand[0],PAR_AxVelHigh,8000);// Operating
velocity
Ret =Acm_SetF64Property(m_Axishand[0],PAR_AxAcc,10000); //Acceleration
Ret =Acm_SetF64Property(m_Axishand[0],PAR_AxDec,10000);//Deceleration
Ret =Acm_SetF64Property(m_Axishand[0],PAR_AxJerk,0); // T-Curve
//Execute Continuous motion
Ret =Acm_AxMoveVel(m_Axishand[0],0); //Continuous move in positive direction
//Get axis status and position
F64 CmdPos;
F64 ActPos;
U16 State;
Acm_AxGetCmdPosition(m_Axishand[0],&CmPos); //Get 0 axis command position
Acm_AxGetActualPosition(m_Axishand[0],&ActPos);//Get 0 axis actual position
Acm_AxGetState(m_Axishand[0],&State); //Get 0 axis status
```

Check function returned value to judge the function execution status and establish error handle mechanism is recommended to ensure the correct result. Refer to the CMove example.

9.2.4 Change Position Motion

9.2.4.1 Function and Property

Change position motion functions are shown in Table 9.5, the functions in table can be called directly in program.

Table 9.5: Change position motion functions

Function	Description
Acm_AxChangePos	To command axis to change the end distance while axis is in point to point motion.
Acm_AxMoveAbs	Command absolute point-to-point motion
Acm_AxMoveRel	Command relative point-to-point motion
Acm_AxSetCmdPosition	Set axis command position
Acm_AxGetCmdPosition	Get axis command position
Acm_AxSetActualPosition	Set axis actual (feedback) position
Acm_AxGetActualPosition	Get axis actual (feedback) position
Acm_AxStopDec	Decelerated stop
Acm_AxStopEmg	Emergency stop (No deceleration)
Acm_AxStopDecEx	Command the axis to stop and specify the deceleration.
Acm_AxGetState	Get states of axis
Acm_AxResetError	Reset error when axis is error-stop
Acm_SetU32Property	Set property (Unsigned 32 -bit integer)
Acm_SetI32Property	Set property (Signed 32 -bit integer)
Acm_SetF64Property	Set property (Double)
Acm_GetU32Property	Get property (Unsigned 32 -bit integer)
Acm_GetI32Property	Get property (Signed 32 -bit integer)
Acm_GetF64Property	Get property (Double)

Change position motion related properties as shown in Table 9.6, the property can't be called directly, but set and get property values by calling related property function. Refer to Appendix for more detail information.

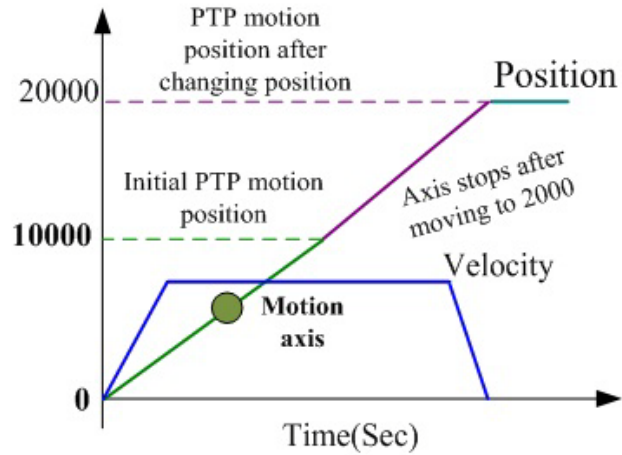
Table 9.6: Change position motion properties

Parameter	Description
PAR_AxVelLow	Set/get initial velocity of axis
PAR_AxVelHigh	Set/get operating velocity of axis
PAR_AxAcc	Set/get acceleration of this axis
PAR_AxDec	Set/get deceleration of this axis
PAR_AxJerk	Set/get the type of velocity profile: T-Curve or S-Curve
Configuration	Description
CFG_AxMaxVel	Configure the max velocity for the motion axis
CFG_AxMaxAcc	Configure the max acceleration for the motion axis
CFG_AxMaxDec	Configure the max deceleration for the motion axis

9.2.4.2 Description

Change position motion means that users can change the target position in point to point motion. The result is dependent on the current position when calling Change position function to change the target position of motion axis:

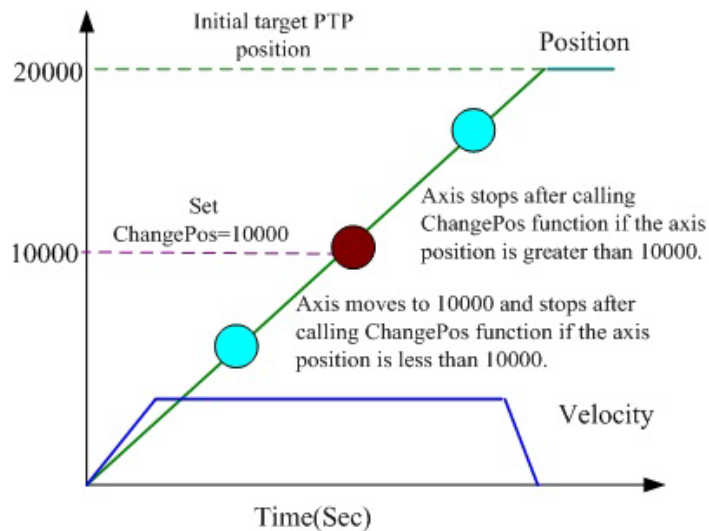
- The end position will be the new target position as the new target position is greater than the current position.



- The end position will be dependent on current position as the new target position is less than the current position.

As shown below, the 0-axis executes relative point to point motion; the initial target position is 20000. Change the target position to 10000 by calling Acm_AxChangePos during moving period.

1. When axis current position is less than 10000, axis will move to 10000 and then stop after changing position.
2. When axis current position is greater than 10000, axis will stop immediately after changing position.



9.2.4.3 Flow Chart

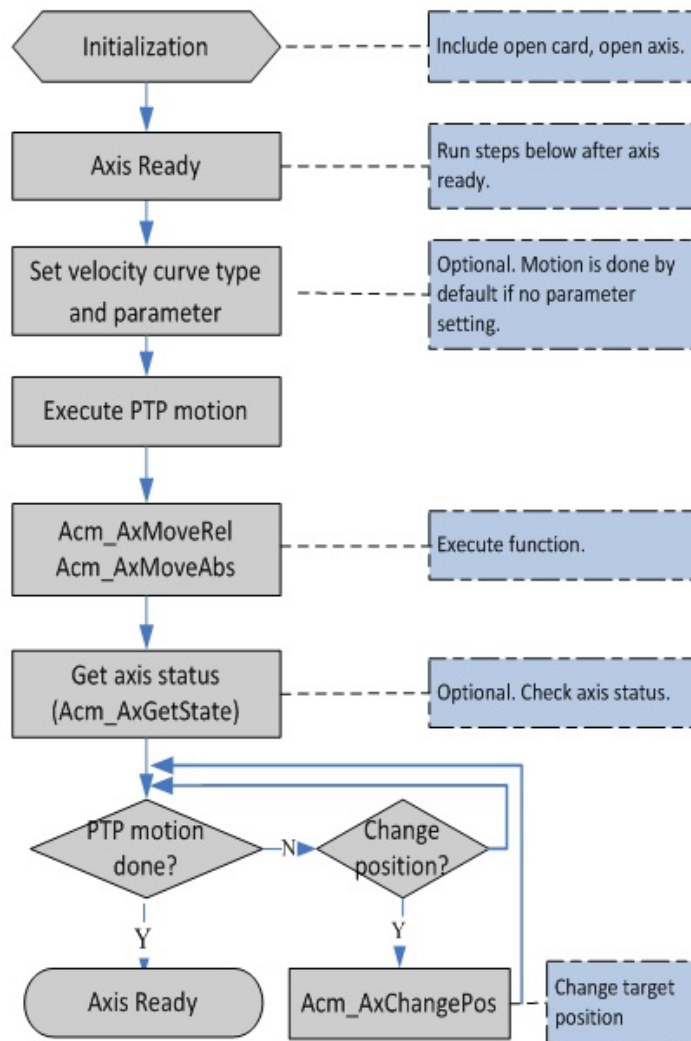


Figure 9.3 Change position motion flow chart

9.2.4.4 Example

This example executes changing initial target position when executing point to point motion on 0 axis. Refer to the table below for setting parameters.

Function	change initial target position when executing point to point motion on 0 axis
Initial velocity	2000PPU/S
Operating velocity	8000PPU/S
Acceleration	10000PPU/S ²
Deceleration	10000PPU/S ²
Velocity curve	T-Curve
Target position	10000PPU
Change target position	20000PPU

VC code:

```
U32 Ret;// Function return value
DWORD m_dwDevNum;
--- Initialization---
// Set parameters
Ret =Acm_SetF64Property(m_Axishand[0],PAR_AxVelLow,2000); //Initial velocity
Ret = Acm_SetF64Property(m_Axishand[0],PAR_AxVelHigh,8000); //Operating
velocity
Ret =Acm_SetF64Property(m_Axishand[0],PAR_AxAcc,10000);//Acceleration
Ret =Acm_SetF64Property(m_Axishand[0],PAR_AxDec,10000);//Deceleration
Ret =Acm_SetF64Property(m_Axishand[0],PAR_AxJerk,0); //T-Curve
//Execute changing target position in PTP motion
Ret = Acm_AxMoveRel(m_Axishand[0],10000);//Relative Point to Point motion
Ret =Acm_AxChangePos(m_Axishand[0], 20000);//Change target position
//Get axis status and position
F64 CmdPos;
F64 ActPos;
U16 State;
Acm_AxGetCmdPosition(m_Axishand[0],&CmPos); //Get 0 axis command position
Acm_AxGetActualPosition(m_Axishand[0],&ActPos); //Get 0 axis actual position
Acm_AxGetState(m_Axishand[0],&State); //Get 0 axis status
```

Check function returned value to judge the function execution status and establish error handle mechanism is recommended to ensure the correct result. Refer to the Change_P example.

9.2.5 Change Velocity Motion

9.2.5.1 Function and Property

Change velocity motion functions are shown in Table 9.7, the functions in table can be called directly in program.

Table 9.7: Change velocity motion functions	
Function	Description
Acm_AxMoveVel	To command axis to make a never ending movement with a specified velocity
Acm_AxMoveAbs	Command absolute point-to-point motion
Acm_AxMoveRel	Command relative point-to-point motion
Acm_AxChangeVel	To command axis to change the velocity while axis is in velocity motion.
Acm_AxChangeVelEx	Change the velocity, acceleration and deceleration simultaneously in motion status.
Acm_AxChangeVelByRate	Change the velocity of current motion according to the given rate
Acm_AxChangeVelExByRate	Change the velocity by rate, acceleration and deceleration simultaneously in motion
Acm_AxGetCmdVelocity	Get axis command velocity
Acm_AxSetCmdPosition	Set axis command position
Acm_AxGetCmdPosition	Get axis command position
Acm_AxGetActualPosition	Get axis actual (feedback) position
Acm_AxStopDec	Decelerated stop
Acm_AxStopEmg	Emergency stop (No deceleration)
Acm_AxStopDecEx	Command the axis to stop and specify the deceleration
Acm_AxGetState	Get states of axis
Acm_AxResetError	Reset error when axis is error-stop
Acm_SetU32Property	Set property (Unsigned 32 -bit integer)
Acm_SetI32Property	Set property (Signed 32 -bit integer)
Acm_SetF64Property	Set property (Double)
Acm_GetU32Property	Get property (Unsigned 32 -bit integer)
Acm_GetI32Property	Get property (Signed 32 -bit integer)
Acm_GetF64Property	Get property (Double)

Change velocity motion related properties as shown in Table 9.8, the property can't be called directly, but set and get property values by calling related property function. Refer to Appendix for more detail information.

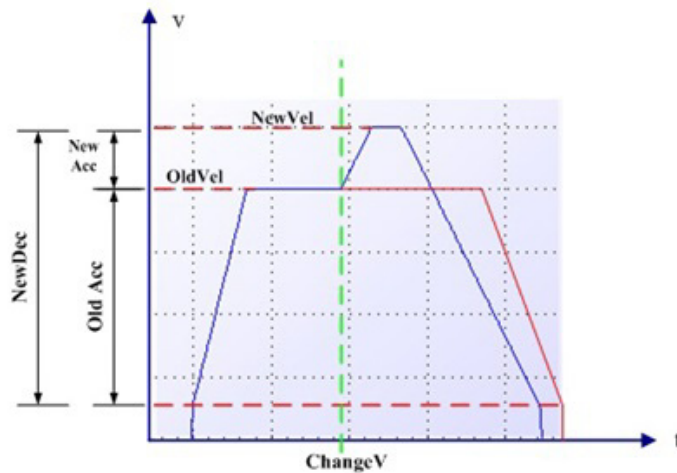
Table 9.8: Change velocity motion properties

Parameter	Description
PAR_AxVelLow	Set/get initial velocity of axis
PAR_AxVelHigh	Set/get operating velocity of axis
PAR_AxAcc	Set/get acceleration of this axis
PAR_AxDec	Set/get deceleration of this axis
PAR_AxJerk	Set/get the type of velocity profile: T-Curve or S-Curve
Configuration	Description
CFG_AxMaxVel	Configure the max velocity for the motion axis
CFG_AxMaxAcc	Configure the max acceleration for the motion axis
CFG_AxMaxDec	Configure the max deceleration for the motion axis

9.2.5.2 Description

Change velocity motion means that changing the current operating velocity, acceleration and deceleration of axis when executing the point to point motion or continuous motion. The new velocity should be greater than the initial velocity. As shown in the figure below:

Change the velocity of the specified axis by calling `Acm_AxChangeVel`. If command has been successfully issued, the new velocity will be applied to the next movement without re-setting the operating velocity.



Change the velocity, acceleration and deceleration of the specified axis by calling `Acm_AxChangeVelEx`. If command has been successfully issued, the new velocity, acceleration and deceleration will be applied to the next movement without re-setting the operating velocity, acceleration and deceleration. If new value of acceleration and deceleration is set to 0, then the acceleration and deceleration will keep the previous value set last time.

Change the velocity according to a given rate and change acceleration and deceleration of the specified axis simultaneously by calling `Acm_AxChangeVelExByRate`. If command has been successfully issued, the new velocity, acceleration and deceleration will be applied to the next movement without re-setting the operating velocity, acceleration and deceleration. If new value of acceleration and deceleration is set to 0, then the acceleration and deceleration will keep the previous value set last time.

Change the velocity according to a given rate by calling `Acm_AxChangeVelByRate`. If command has been successfully issued, the new velocity just be effective for current motion.

9.2.5.3 Flow Chart

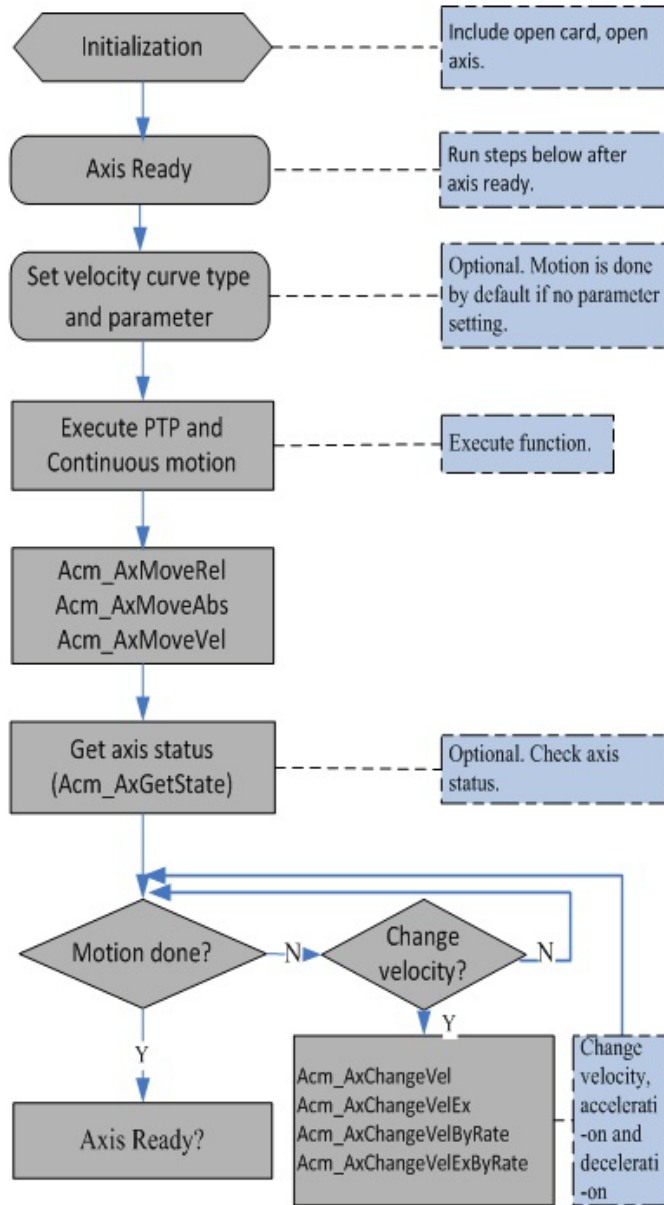


Figure 9.4 Change velocity motion flow chart

9.2.5.4 Example

This example executes changing axis operating velocity when executing point to point motion on 0 axis. Refer to the table below for setting parameters.

Function	Changing axis operating velocity when executing point to point motion on 0 axis.
Initial velocity	2000PPU/S
Operating velocity	8000PPU/S
Acceleration	10000PPU/S ²
Deceleration	10000PPU/S ²
Velocity curve	T-Curve
Change velocity	6000 PPU/S
Target position	10000PPU

VC code:

```
HAND m_Axishand[32];
U32  Ret; // Function return value
--- Initialization---
// Set parameters
Ret =Acm_SetF64Property(m_Axishand[0],PAR_AxVelLow,2000);// Initial velocity
Ret = Acm_SetF64Property(m_Axishand[0],PAR_AxVelHigh,8000); // Operating
velocity
Ret =Acm_SetF64Property(m_Axishand[0],PAR_AxAcc,10000);//Acceleration
Ret =Acm_SetF64Property(m_Axishand[0],PAR_AxDec,10000);//Deceleration
Ret =Acm_SetF64Property(m_Axishand[0],PAR_AxJerk,0); //T-Curve
//Execute changing velocity in PTP motion
Ret = Acm_AxMoveRel(m_Axishand[0],10000); //Relative Point to Point motion
Ret = Acm_AxChangeVel(m_Axishand[0], 6000); //Change operating velocity
//Get axis status and position
F64  CmdPos;
F64  ActPos;
U16  State;
Acm_AxGetCmdPosition(m_Axishand[0],&CmPos); //Get 0 axis command position
Acm_AxGetActualPosition(m_Axishand[0],&ActPos);//Get 0 axis actual position
Acm_AxGetState(m_Axishand[0],&State); //Get 0 axis status
```

Check function returned value to judge the function execution status and establish error handle mechanism is recommended to ensure the correct result. Refer to the Change_V example.

9.2.6 Simultaneous Start/Stop Motion

9.2.6.1 Function and Property

Simultaneous start / stop motion functions are shown in Table 9.9, the functions in table can be called directly in program.

Table 9.9: Simultaneous Start/Stop Motion functions

Function	Description
Acm_AxSimStartSuspend Abs	Set the axis in wait state for simultaneous absolute point to point motion operation
Acm_AxSimStartSuspend Rel	Set the axis in wait state for simultaneous relative point to point motion operation
Acm_AxSimStartSuspend Vel	Set the axis in wait state for simultaneous continuous motion operation
Acm_AxSimStart	Start all axis waiting for start trigger
Acm_AxSimStop	Stop all axis triggered by simultaneous signal
Acm_AxSetCmdPosition	Set axis command position
Acm_AxGetCmdPosition	Get axis command position
Acm_AxGetActualPosition	Get axis actual (feedback) position
Acm_AxGetCmdVelocity	Get axis command velocity
Acm_AxGetState	Get states of axis
Acm_AxResetError	Reset error when axis is error-stop
Acm_SetU32Property	Set property (Unsigned 32 -bit integer)
Acm_SetI32Property	Set property (Signed 32 -bit integer)

Table 9.9: Simultaneous Start/Stop Motion functions

Acm_SetF64Property	Set property (Double)
Acm_GetU32Property	Get property (Unsigned 32 -bit integer)
Acm_GetI32Property	Get property (Signed 32 -bit integer)
Acm_GetF64Property	Get property (Double)

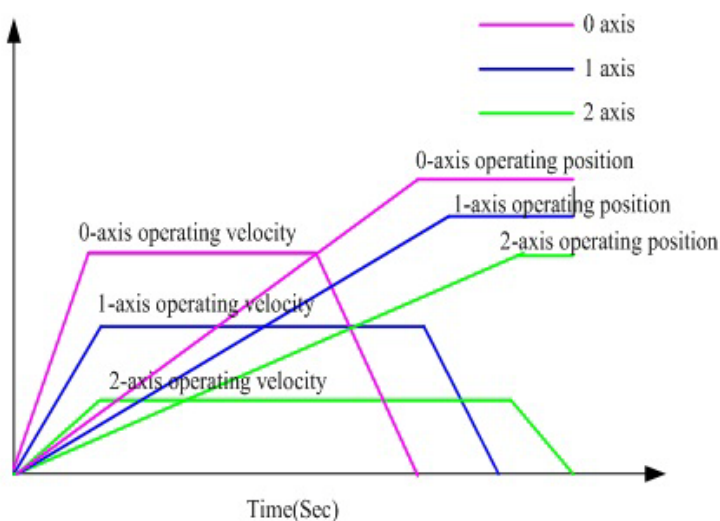
Change velocity motion related properties as shown in Table 9.10, the property can't be called directly, but set and get property values by calling related property function. Refer to Appendix for more detail information.

Table 9.10: Simultaneous start/stop motion properties

Parameter	Description
PAR_AxVelLow	Set/get initial velocity of axis
PAR_AxVelHigh	Set/get operating velocity of axis
PAR_AxAcc	Set/get acceleration of this axis
PAR_AxDec	Set/get deceleration of this axis
PAR_AxJerk	Set/get the type of velocity profile: T-Curve or S-Curve
Configuration	Description
CFG_AxMaxVel	Configure the max velocity for the motion axis
CFG_AxMaxAcc	Configure the max acceleration for the motion axis
CFG_AxMaxDec	Configure the max deceleration for the motion axis
CFG_AxSimStartSource	Set/get simultaneous start/Stop mode for current axis.
Feature	Description
FT_AxSimStartSourceMa p	Simultaneous start/stop mode supportedBy axis

9.2.6.2 Description

Simultaneous start/ stop motion means that axes execute point to point motion or continuous motion after receiving the trigger signal. Each axis moves to the target position with its own velocity set by the user. All axes also can stop simultaneously.



All axes stop simultaneously after executing `Acm_AxSimStop` command during operation. Simultaneous start / stop mode can be set through property `CFG_AxSimStartSource`. Different property value represents different simultaneous start / stop mode. Refer to Appendix for more detail information.

Users need to set axis to suspend through calling different API according to the motion behavior:

1. Absolute point to point motion: `Acm_AxSimStartSuspendAbs`
2. Relative point to point motion: `Acm_AxSimStartSuspendRel`
3. Continuous motion: `Acm_AxSimStartSuspendRel`

Finally, start all axes which are waiting for trigger by calling `Acm_AxSimStart` to issues a simultaneous trigger signal.

9.2.6.3 Flow Chart

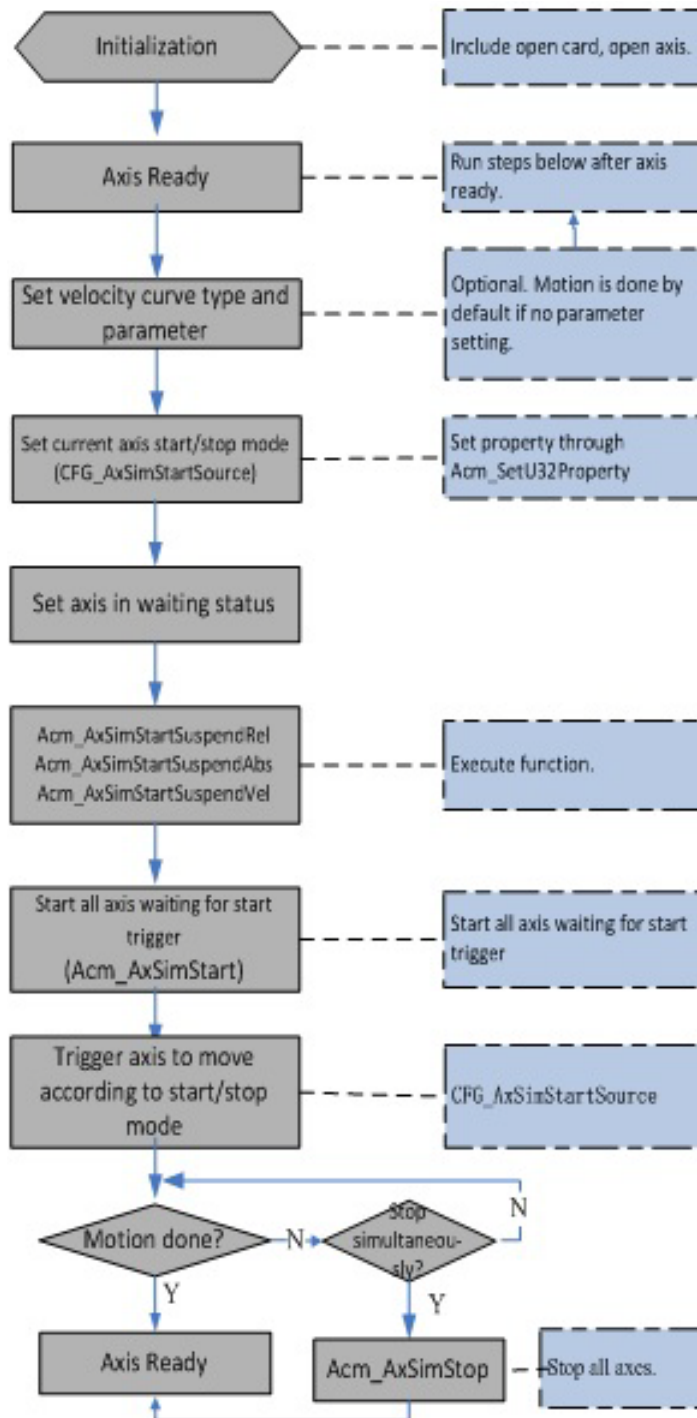


Figure 9.5 Simultaneous start/stop motion flow chart

9.2.6.4 Example

This example executes triggering 1, 2, 3 axis simultaneously to do point to point motion. Refer to the table below for setting parameters.

Function	Trigger 1, 2, 3 axis simultaneously to do point to point motion.
Simultaneous Start/ Stop	SIM_STOP_AX0 (The waiting axes will start simultaneously if 0 axis stops.)
Source axis	0 axis
Initial velocity	2000PPU/S
Operating velocity	8000PPU/S
Acceleration	10000PPU/S ²
Deceleration	10000PPU/S ²
Velocity curve	T-Curve

First, users need to select simultaneous start/stop mode by setting the property value of CFG_AxSimStartSource. Each waiting axis needs to set simultaneous start/stop mode independently.

VC code:

```
HAND m_Axishand[32];
U32 Ret; // Function return value
--- Initialization---
// Set parameters
for (int i = 0; i < 4; i++)
{
    Ret =Acm_SetF64Property(m_Axishand[i],PAR_AxVelLow,2000); // Initial velocity
    Ret = Acm_SetF64Property(m_Axishand[i],PAR_AxVelHigh,8000); //Operating velocity
    Ret =Acm_SetF64Property(m_Axishand[i],PAR_AxAcc,10000);//Acceleration
    Ret = Acm_SetF64Property(m_Axishand[i],PAR_AxDec,10000);//Deceleration
    Ret =Acm_SetF64Property(m_Axishand[i],PAR_AxJerk,0);//T-Curve
}
for (int i = 1; i < 4; i++)
{Ret=Acm_SetU32Property(m_Axishand[i],CFG_AxSimStartSource,SIM_STOP_AX0);}
//Each waiting axis needs to set simultaneous start/stop mode imdependently
for (inti = 1; i < 4; i++)
{
    Ret =Acm_AxSimStartSuspendRel(m_Axishand[i],50000);}//Set axis in waiting status
}
Ret =Acm_AxSimStart(m_Axishand[0]); //Start simultaneous start/stop motion
```

After completing the setting above, axis 1, 2 and 3 will execute Point-to-Point motion simultaneously once 0 axis stops.

Stop all axes motion by calling Acm_AxSimStop. Set properties and APIs of compared events if the user selects simultaneous start/stop mode as compare signal trigger.

Check function returned value to judge the function execution status and establish error handle mechanism is recommended to ensure the correct result. Refer to the NewSimulateOpe example.

9.2.7 Imposed Motion

9.2.7.1 Function and Property

Imposed motion functions are shown in Table 9.11, the functions in the table can be called directly in program.

Table 9.11: Imposed motion functions

Function	Description
Acm_AxMoveImpose	Impose a new motion on current motion
Acm_AxMoveAbs	Command absolute point-to-point motion
Acm_AxMoveRel	Command relative point-to-point motion
Acm_AxStopDec	Decelerated stop
Acm_AxStopEmg	Emergency stop (No deceleration)
Acm_AxStopDecEx	Command the axis to stop and specify the deceleration
Acm_AxGetState	Get states of axis
Acm_AxResetError	Reset error when axis is error-stop
Acm_SetU32Property	Set property (Unsigned 32 -bit integer)
Acm_SetI32Property	Set property (Signed 32 -bit integer)
Acm_SetF64Property	Set property (Double)
Acm_GetU32Property	Get property (Unsigned 32 -bit integer)
Acm_GetI32Property	Get property (Signed 32 -bit integer)
Acm_GetF64Property	Get property (Double)
Acm_AxSetCmdPosition	Set axis command position
Acm_AxGetCmdPosition	Get axis command position
Acm_AxGetActualPosition	Get axis actual (feedback) position

Imposed motion related properties as shown in Table 9.12, the property can't be called directly, but set and get property values by calling related property function. Refer to Appendix for more detail information.

Table 9.12: Imposed motion properties

Parameter	Description
PAR_AxVelLow	Set/get initial velocity of axis
PAR_AxVelHigh	Set/get operating velocity of axis
PAR_AxAcc	Set/get acceleration of this axis
PAR_AxDec	Set/get deceleration of this axis
PAR_AxJerk	Set/get the type of velocity profile: T-Curve or S-Curve
Configuration	Description
CFG_AxMaxVel	Configure the max velocity for the motion axis
CFG_AxMaxAcc	Configure the max acceleration for the motion axis
CFG_AxMaxDec	Configure the max deceleration for the motion axis

9.2.7.2 Description

Imposed motion refers to impose new movement on the current motion, and the end position will be the original end position plus / minus new position. Users may choose appropriate parameters of imposed position and velocity since this function only use the time within continuous section to impose/compensate this position and velocity. It means that the imposed motion will not change (extend) the moment of deceleration in time. If the imposed position is too large to achieve (or the imposed velocity is too small), it will not reach the goal. If the user needs to change end position, use Change Position Motion in Chapter 9.2.4.

The velocity supports T-Curve. The new imposed position and velocity can be set by calling `Acm_AxMoveImpose`. Overall velocity curve is decided by `NewVel`, `PAR_AxVelLow`, `PAR_AxVelHigh`, `PAR_AxAcc`, `PAR_AxDec` and `PAR_AxJerk`. As shown as Figure 9.6.

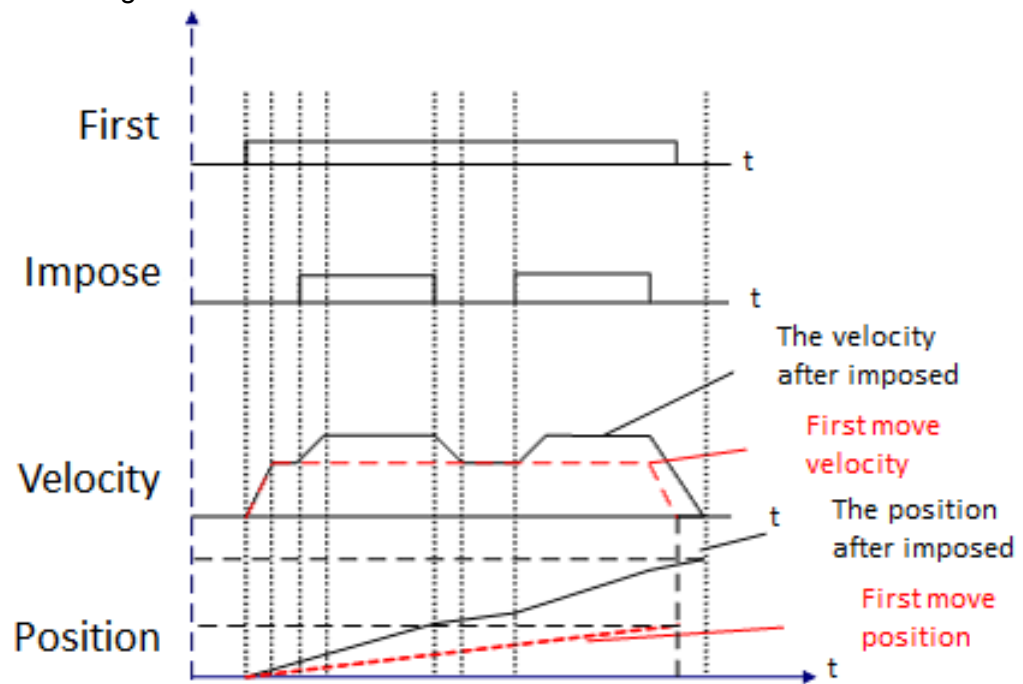


Figure 9.6 Imposed motion

Note! It's not allowed to add new movement on imposed motion.



9.2.7.3 Flow Chart

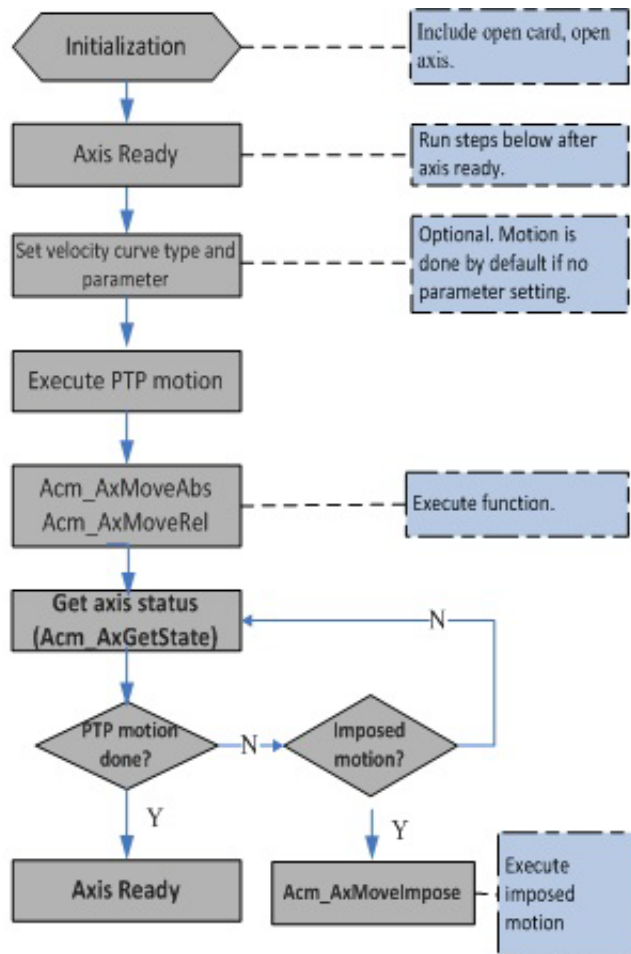


Figure 9.7 Imposed motion flow chart

9.2.7.4 Example

This example executes impose a new motion on original point to point motion on 0 axis. Refer to the table below for setting parameters.

Function	Impose a new motion on original point to point motion on 0 axis.
Target position	30000
Initial velocity	2000PPU/S
Operating velocity	8000PPU/S
Acceleration	10000PPU/S ²
Deceleration	10000PPU/S ²
Velocity curve	T-Curve
Imposed operating velocity	3000 PPU/S
Imposed distance	5000PPU

VC code:

```
HAND m_Axishand[32];
U32 Ret;// Function return value
--- Initialization---
// Set parameters
Ret =Acm_SetF64Property(m_Axishand[0],PAR_AxVelLow,2000); // Initial velocity
Ret = Acm_SetF64Property(m_Axishand[0],PAR_AxVelHigh,8000); //Operating
velocity
Ret =Acm_SetF64Property(m_Axishand[0],PAR_AxAcc,10000); //Acceleration
Ret = Acm_SetF64Property(m_Axishand[0],PAR_AxDec,10000); //Deceleration
Ret =Acm_SetF64Property(m_Axishand[0],PAR_AxJerk,0); //T-Curve
//Impose a new motion on original Point-to point motion
Ret =Acm_AxMoveRel(m_Axishand[0],30000); //Relative Point to Point motion
Ret = Acm_AxMoveImpose(m_Axishand[0],5000,3000); //Impose a new motion
//Get axis status and position
F64 CmdPos;
F64 ActPos;
F64 NewVel;
U16 State;
Acm_AxGetCmdVelocity(m_Axishand[0],&NewVel) //Get 0 axis command velocity
Acm_AxGetCmdPosition(m_Axishand[0],&CmPos); //Get 0 axis command position
Acm_AxGetActualPosition(m_Axishand[0],&ActPos); //Get 0 axis actual position
Acm_AxGetState(m_Axishand[0],&State); //Get 0 axis status
```

The imposed velocity is 11000 and the moving distance is 35000.

Check function returned value to judge the function execution status and establish error handle mechanism is recommended to ensure the correct result. Refer to the MoveImpose example.

9.2.8 Jog Motion

9.2.8.1 Function and Property

Jog motion functions are shown in Table 9.13, the functions in table can be called directly in program.

Table 9.13: Jogmotion functions

Function	Description
Acm_AxSetExtDrive	Enable or disable external drive mode
Acm_AxJog	Assign axis to do Jog move
Acm_AxGetState	Get states of axis
Acm_AxResetError	Reset error when axis is error-stop
Acm_SetU32Property	Set property (Unsigned 32 -bit integer)
Acm_SetI32Property	Set property (Signed 32 -bit integer)
Acm_SetF64Property	Set property (Double)
Acm_GetU32Property	Get property (Unsigned 32 -bit integer)
Acm_GetI32Property	Get property (Signed 32 -bit integer)
Acm_GetF64Property	Get property (Double)
Acm_AxGetCmdPosition	Get axis command position
Acm_AxGetActualPosition	Get axis actual (feedback) position

Jog motion related properties as shown in Table 9.14, the property can't be called directly, but set and get property values by calling related property function. Refer to Appendix for more detail information.

Table 9.14: Jog motion properties

Parameter	Description
PAR_AxVelLow	Set/get initial velocity of axis
PAR_AxVelHigh	Set/get operating velocity of axis
PAR_AxAcc	Set/get acceleration of this axis
PAR_AxDec	Set/get deceleration of this axis
PAR_AxJerk	Set/get the type of velocity profile: T-Curve or S-Curve
Configuration	Description
CFG_AxJogVelLow	Set/get initial velocity of Jog move
CFG_AxJogVTime	Set/get low-speed period of Jog move
CFG_AxJogVelHigh	Set/get operating velocity of Jog move
CFG_AxJogAcc	Set/get acceleration of Jog move
CFG_AxJogDec	Set/get deceleration of Jog move
CFG_AxJogJerk	Set/get T-Curve or S-Curve of Jog move
CFG_AxMaxVel	Configure the max velocity for the motion axis
CFG_AxMaxAcc	Configure the max acceleration for the motion axis
CFG_AxMaxDec	Configure the max deceleration for the motion axis
CFG_AxJogPAssign	Set/get the DI channel as external Jog+
CFG_AxJogNAssign	Set/get the DI channel as external Jog-
Feature	Description
FT_AxExtDriveMap	Get axis supported external drive features
FT_AxJogMap	Get Jog feature supported by axis
FT_AxExtMasterSrcMap	Get Jog source supported by axis

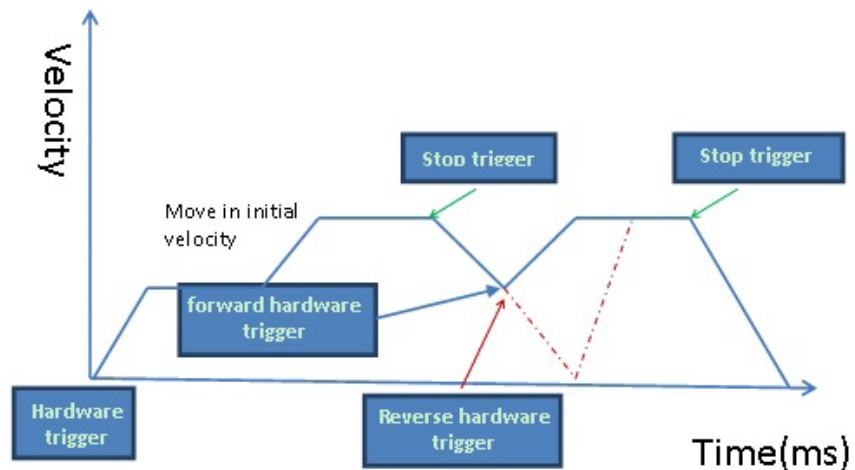
9.2.8.2 Description

The initial velocity, operating velocity and acceleration of each axis can be set independently under Jog / Hand Wheel mode.

In Jog mode, setting initial velocity, operating velocity, acceleration and deceleration by configuring the values of properties: CFG_AxJogVelLow, CFG_AxJogVelHigh, CFG_AxJogAcc, CFG_AxJogDec.

Set low-speed operating time by configuring the property value of CFG_AxJogVLTime in Jog motion. Axis will move in low speed in that period.

Jog motion is shown as figure below. Users can keep trigger Jog move during stop period of Jog. When the trigger is in the same direction, axis will be accelerated to the operating velocity directly; when the trigger is in the opposite direction, axis will be decelerated to zero and then be accelerated to the operating velocity directly.



By setting the property of CFG_AxJogPAssign or CFG_AxJogNAssign, users can assign DI channel which belong to ADAM-5000/ECAT as the signal JOG+ and JOG-.

9.2.8.3 Flow Chart

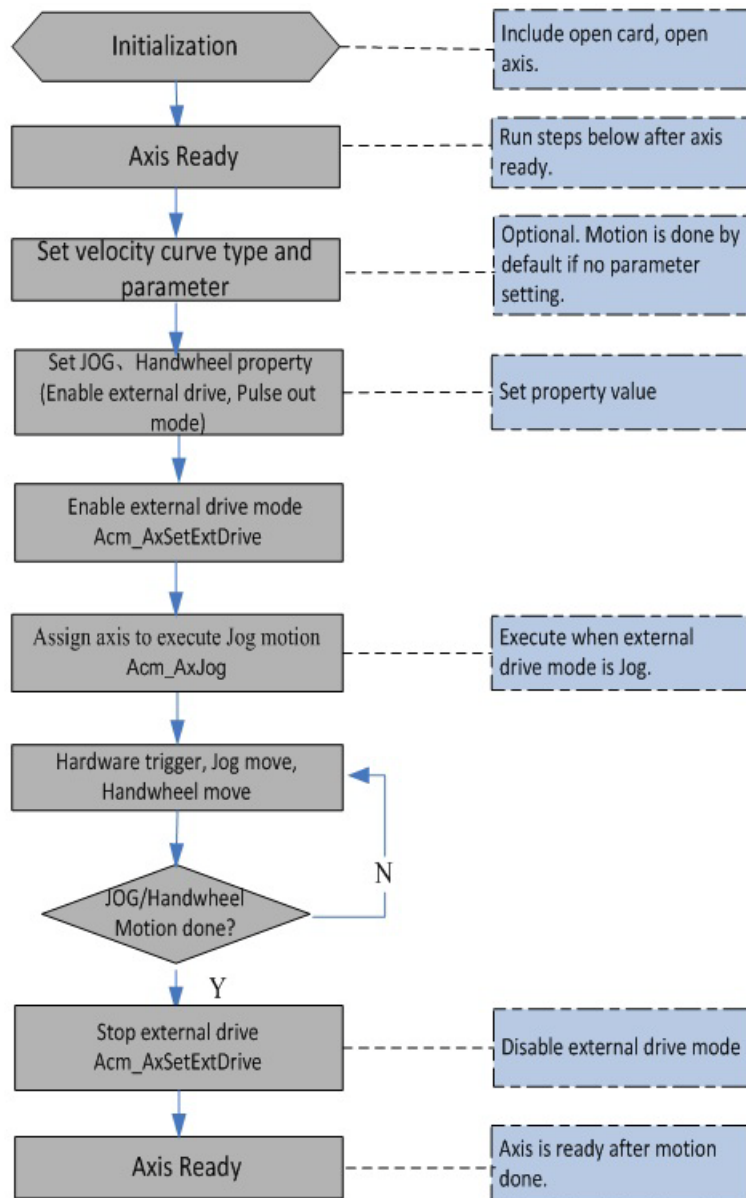


Figure 9.8 Jog motion flow chart

9.2.8.4 Example

This example executes Jog move on 0 axis. Refer to the table below for setting parameters.

Function	Executes Jog move on 0 axis.
Command pulse	1000
Pulde output mode	2000PPU/S
Initial velocity	2000PPU/S
Operating velocity	8000PPU/S
Acceleration	10000PPU/S ²
Deceleration	10000PPU/S ²
Velocity curve	T-Curve

VC code:

```
HAND m_Axishand[32];
U32  Ret; // Function return value
--- Initialization---
// Set parameters
Ret =Acm_SetF64Property(m_Axishand[0],PAR_AxVelLow,2000); // Initial velocity
Ret = Acm_SetF64Property(m_Axishand[0],PAR_AxVelHigh,8000); //Operating
velocity
Ret =Acm_SetF64Property(m_Axishand[0],PAR_AxAcc,10000);//Acceleration
Ret = Acm_SetF64Property(m_Axishand[0],PAR_AxDec,10000); //Deceleration
Ret = Acm_SetF64Property(m_Axishand[0],PAR_AxJerk,0); //T-Curve
Ret = Acm_AxSetExtDrive(m_Axishand[0], 1);//Enable Jog Mode
//Get axis status and position
F64  CmdPos;
F64  ActPos;
U16  State;
Acm_AxGetCmdPosition(m_Axishand[0],&CmPos); //Get 0 axis command position
Acm_AxGetActualPosition(m_Axishand[0],&ActPos);//Get 0 axis actual position
Acm_AxGetState(m_Axishand[0],&State); //Get 0 axis status
```

Check function returned value to judge the function execution status and establish error handle mechanism is recommended to ensure the correct result. Refer to the JOG example.

9.2.9 Homing

9.2.9.1 Function and Property

Homing motion functions are shown in Table 9.15, the functions in table can be called directly in program.

Table 9.15: Homing motion functions

Function	Description
Acm_AxHome	To command axis to start typical home motion
Acm_AxMoveHome	Command specific axis move home
Acm_AxSetCmdPosition	Set axis command position
Acm_AxGetCmdPosition	Get axis command position
Acm_AxGetActualPosition	Get axis actual (feedback) position
Acm_AxStopDec	Decelerated stop
Acm_AxStopEmg	Emergency stop (No deceleration)
Acm_AxStopDecEx	Command the axis to stop and specify the deceleration
Acm_AxGetState	Get states of axis
Acm_AxResetError	Reset error when axis is error-stop
Acm_SetU32Property	Set property (Unsigned 32 -bit integer)
Acm_SetI32Property	Set property (Signed 32 -bit integer)
Acm_SetF64Property	Set property (Double)
Acm_GetU32Property	Get property (Unsigned 32 -bit integer)
Acm_GetI32Property	Get property (Signed 32 -bit integer)
Acm_GetF64Property	Get property (Double)

Homing motion related properties as shown in Table 9.16, the property can't be called directly, but set and get property values by calling related property function. Refer to Appendix for more detail information.

Table 9.16: Homing motion properties

Parameter	Description
PAR_AxHomeCrossDistance	Set the home cross distance (Unit: PPU).
PAR_AxHomeExSwitchMode	Setting the stopping condition of Acm_AxHomeEx
PAR_AxVelLow	Set/get initial velocity of axis
PAR_AxVelHigh	Set/get operating velocity of axis
PAR_AxAcc	Set/get acceleration of this axis
PAR_AxDec	Set/get deceleration of this axis
PAR_AxJerk	Set/get the type of velocity profile: T-Curve or S-Curve
PAR_AxHomeVelLow	Set/get home initial velocity of axis
PAR_AxHomeVelHigh	Set/get home operating velocity of axis
PAR_AxHomeAcc	Set/get home acceleration of this axis
PAR_AxHomeDec	Set/get home deceleration of this axis
PAR_AxHomeJerk	Set/get the type of home velocity profile: T-Curve or S-Curve
Configuration	Description
CFG_AxMaxVel	Configure the max velocity for the motion axis
CFG_AxMaxAcc	Configure the max acceleration for the motion axis
CFG_AxMaxDec	Configure the max deceleration for the motion axis
Feature	Description
FT_AxHomeModeMap	The supported Home return modes

9.2.9.2 Description

Homing refers to axis comes back behavior as receiving ORG, LMT+, LMT-, EZ signals. In Home mode, ORG, LMT+, LMT-, EZ are also called as Home signals. Namely, the signals make motor stop when motor comes back to home. The modes of homing function are defined by CiA402 and the details of how each of the homing modes shall function will be described.

There are two functions provided by Advantech Common Motion to execute Home motion: Acm_AxHome and Acm_AxMoveHome.

Before using Acm_AxHome, the cross distance should be set through PAR_AxHomeCrossDistance. The speed curve is decided by PAR_AxVelLow, PAR_AxVelHigh, PAR_AxAcc, PAR_AxDec and PAR_AxJerk.

Before using Acm_AxMoveHome, the cross distance should be set through PAR_AxHomeCrossDistance. The speed curve is decided by PAR_AxHomeVelLow, PAR_AxHomeVelHigh, PAR_AxHomeAcc, PAR_AxHomeDec and PAR_AxHomeJerk.

The Homing mode can be divided into two types, one is the homing mode defined by Advantech and the other is typical Home motion defined by CiA402.

Type 1: Defined by Advantech

To command axis to start typical home motion. The 16 types of typical home motion are composed of extended home.

Comments

During home motion of MODE3_Ref~MODE16_LmtSearchReFind_Ref, the initial velocity will be used in some stages. Therefore, the initial velocity decided by PAR_AxVelLow must be larger than zero.

If property CFG_AxHomeResetEnable is set to be true, command position and actual position will be reset to be zero after home motion ends.

Explanations

The meanings of a, b, c and d in the below figures are:

- a. The velocity will decrease when trapezoid PTP motion meets ORG/EL signal.
- b. Trapezoid PTP motion moves with HomeCrossDistance as distance until the motion finishes. ORG/EL signal is in effective.
- c. Trapezoid PTP take a uniform motion at VelLow. It will stop immediately when it meets ORG/EL signal.
- d. Trapezoid PTP motion moves at VelLow with HomeCrossDistance as distance unit until the motion finishes.

ORG/EL signal is in effective.

•: This solid black dot means the ending point of a motion.

Note! *Features of trapezoid PTP motion: When start, the velocity will increase from VelLow to VelHigh with Acc (If distance is long enough); when end, the velocity will decrease from VelHigh to VelLow with Dec.*



1. MODE1_Abs: Move (Dir) ->touch ORG->Stop.

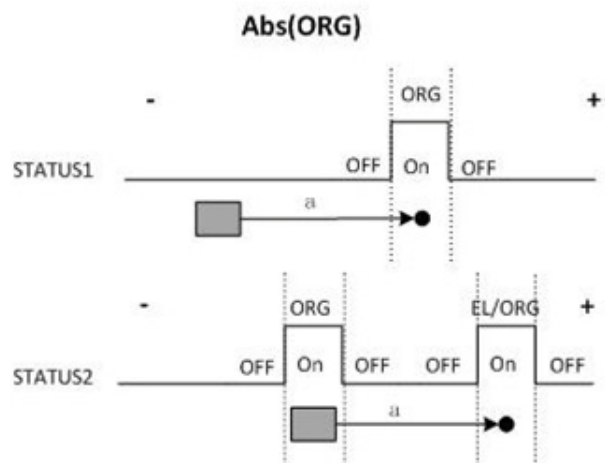
Only according to origin equipment (eg.sensor) to home. The object moves continuously until the origin signal occurring.

For example:

Dir: Positive.

Org Logic (CFG_AxOrgLogic): Active High.

EL (Hard Limit switch) Logic (CFG_AxEILogic): Active High.



- STATUS1: If the object is out of the field of ORG signal, when home command is written, the object will move until ORG signal occurring.

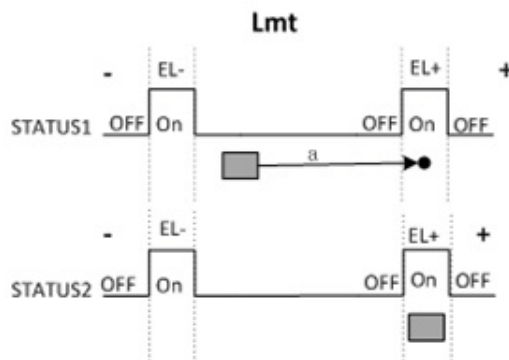
- STATUS2: If the object is in the field of ORG signal or the direction is opposite with ORG switch, the object will move until ORG signal (if there are more than one ORG switch or the axis equipment is occlusive) or EL signal (axis's states is error stop).

2. MODE2_Lmt: Move (Dir)->touch EL->Stop

Only according to limit equipment (eg.sensor) to home. The object moves continuously until the limit signal occurring.

For Example:

Dir: Positive. Limit Logic (CFG_AxEILogic): Active High.



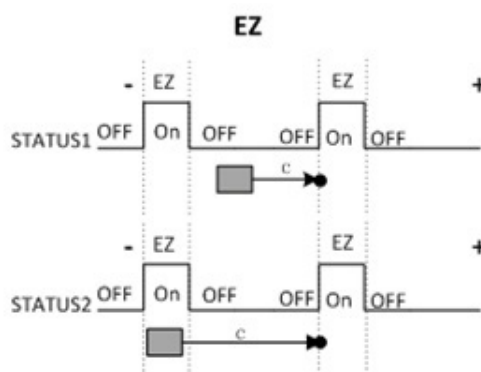
- STATUS1: If the object is out of the field of EL signal, when home command is written, the object will move until EL signal occurring.
- STATUS2: If the object is in the field of EL signal, there will be no response.

3. MODE3_Ref: Move (Dir) ->touch EZ->Stop.

Only according to EZ to home. The object moves continuously until the EZ signal occurring.

For Example:

Dir: Positive. EZ Logic (CFG_AxEzLogic): Active High.



- STATUS1: If the object is out of the field of EZ signal, when home command is written, the object will move until EZ signal occurring.
- STATUS2: If the object is in the field of ORG signal, the object will move until next EZ signal occurring.

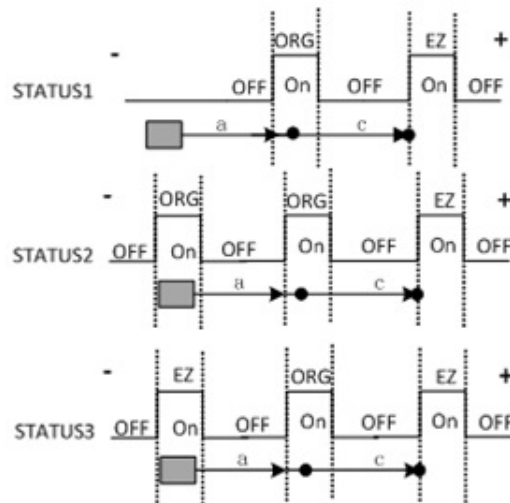
4. MODE4_Abs_Ref: ORG+EZ, Move (Dir) ->touch ORG ->Stop ->Move (Dir)->touch EZ ->Stop

This is a composed home mode. Firstly, the object moves until origin signal occurring, and then continues to move in same direction with ORG until EZ signal occurring.

For Example:

Dir: Positive. ORG logic: Active Logic. EZ Logic: Active Logic.

Abs(ORG)+EZ



- STATUS1: If the object is out of the field of EZ signal and ORG signal, when home command is written. Firstly, the object will move until ORG signal occurring, then continue to move until EZ signal occurring.
- STATUS2: If the object is in the field of ORG signal, the home command is written, the object begins to move. Firstly, the ORG signal disappears, and then next ORG signal occurs. At last, motion is stopped when EZ signal occurring.
- STATUS3: If the object is in the field of EZ signal, the home command is written, the object begins to move. Firstly, the EZ signal disappears, and then ORG signal occurs. At last, motion stops when EZ signal occurring.

Note! Home will stop in case EL signal occurs.

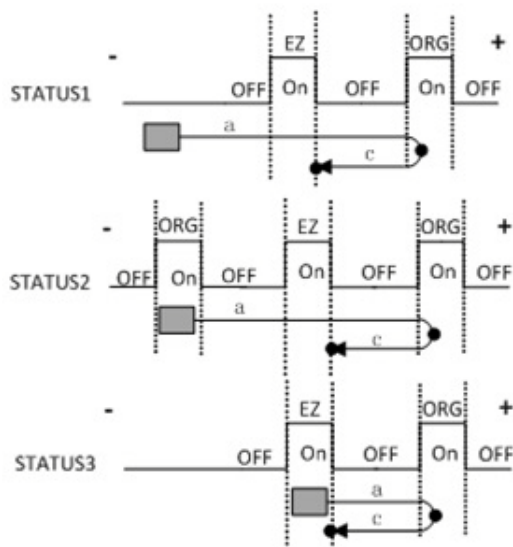


5. MODE5_Abs_NegRef: ORG+EZ, Move (Dir) ->touch ORG ->Stop ->Move (-Dir)->touch EZ ->Stop.

This is a composed home mode. The object moves until origin signal occurring firstly, and then continues to move in opposite direction with ORG until EZ signal is occurred.

For Example:

Dir: Positive. ORG logic: Active Logic. EZ Logic: Active Logic.

Abs (ORG)+NegEZ

- STATUS1: If the object is out of the field of EZ signal and ORG signal, when home command is written. Firstly, the object will move until ORG signal occurring, then continue to move in opposite direction until EZ signal occurring.
- STATUS2: If the object is in the field of ORG signal, the home command is written, the object begins to move. Firstly, the ORG signal disappears, and then next ORG signal occurs, at the same time reverses motion direction. At last, motion is stopped when EZ signal occurring.
- STATUS3: If the object is in the field of EZ signal, the home command is written, the object begins to move. Firstly, the EZ signal disappears, and then ORG signal occurs, at the same time reverses motion direction. At last, motion stops when EZ signal occurring.

Note! Home will stop in case EL signal occurs.

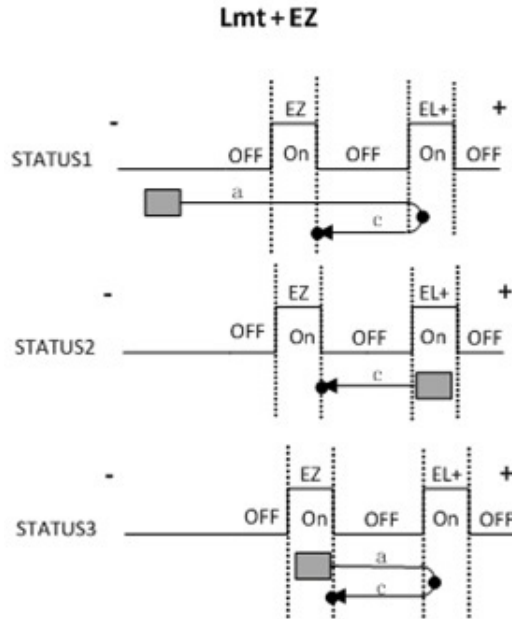


6. MODE6_Lmt_Ref: EL + NegEZ, Move (Dir) ->touch EL ->Stop -> Move (-Dir) ->touch EZ ->Stop.

The object moves until limit signal occurring firstly, and then continues to move in opposite direction until EZ signal is occurred.

For Example:

Dir: Positive. EZ Logic: Active Logic. Limit Logic: Active High.



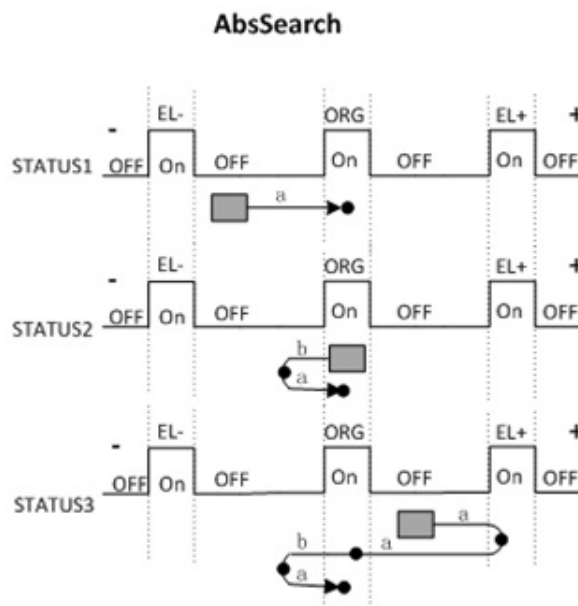
- STATUS1: If the object is out of the field of EZ signal and EL signal, when home command is written. Firstly, the object will move until EL signal occurring, then continue to move in opposite direction until EZ signal occurring.
- STATUS2: If the object is in the field of EL signal, the object will move in opposite direction until EZ signal occurring.
- STATUS3: If the object is in the field of EZ signal, the home command is written, the object begins to move. Firstly, the EZ signal disappears, and then EL signal occurs, at the same time reverses motion direction. At last, motion stops when an EZ signal occurs.

7. MODE7_AbsSearch: Move (Dir) ->Search ORG ->Stop.

This is a mode of searching transformation of ORG signal from no signal to signal occurring.

For Example:

Dir: Positive. ORG logic: Active high. Limit logic: Active High.



- STATUS1: If there is no ORG signal occurring, the object will stop when ORG-signal occurs.

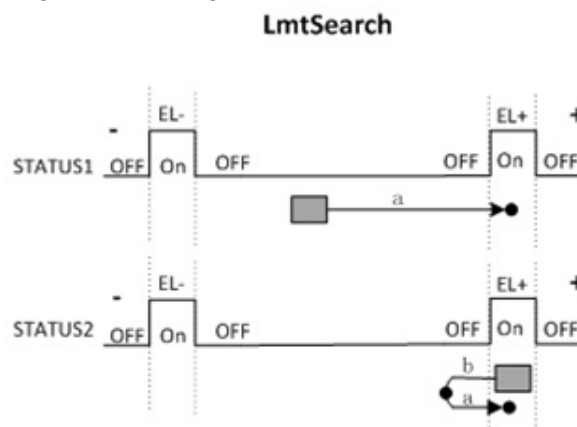
- STATUS2: If the object is in the field of ORG signal. The Object moves in opposite direction until the signal disappears, and then converts direction to move until ORG signal occurring.
- STATUS3: If there is no ORG signal occurring. EL signal happens while moving firstly, the object reverses direction and continues to move, and then the ORG-signal from happening to disappearing. Reverses direction again, and moves until ORG signal occurring. Motion stops.

8. **MODE8_LmtSearch: Move (Dir) ->Search EL ->Stop.**

This is a mode of searching transformation of limit signal from no signal to signal occurring.

For Example:

Dir: Positive. Limit logic: Active High.



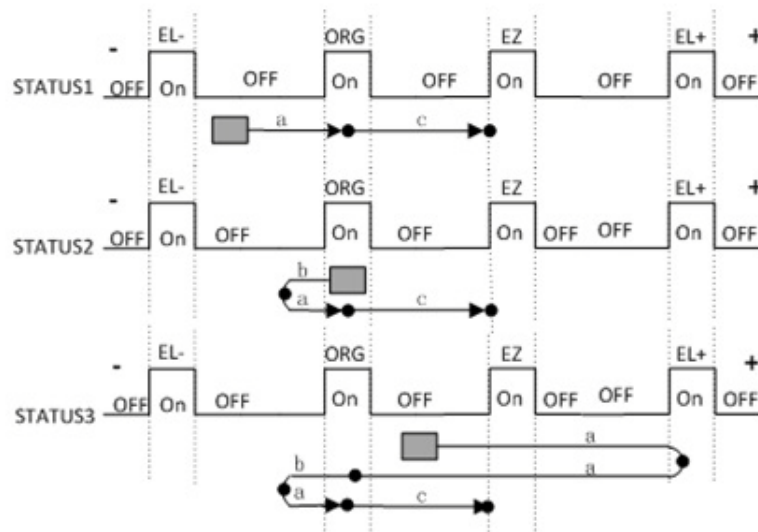
- STATUS1: If the Limit signal is occurred firstly while the object is moving, the home process is end.
 - STATUS2: If the object is in the field of limit signal. The Object moves in opposite direction until the signal disappears, and then converts direction to move until limit signal occurring.
9. **MODE9_AbsSearch_Ref: Search ORG + EZ, Move (Dir) ->Search ORG ->Stop->Move (Dir) ->touch EZ ->Stop.**

Firstly, object moves in the way of MODE7_AbsSearch, and then moves in same direction until EZ signal occurring.

For example:

Dir: Positive. Limit logic: Active High. ORG logic: Active High.

AbsSearch + EZ

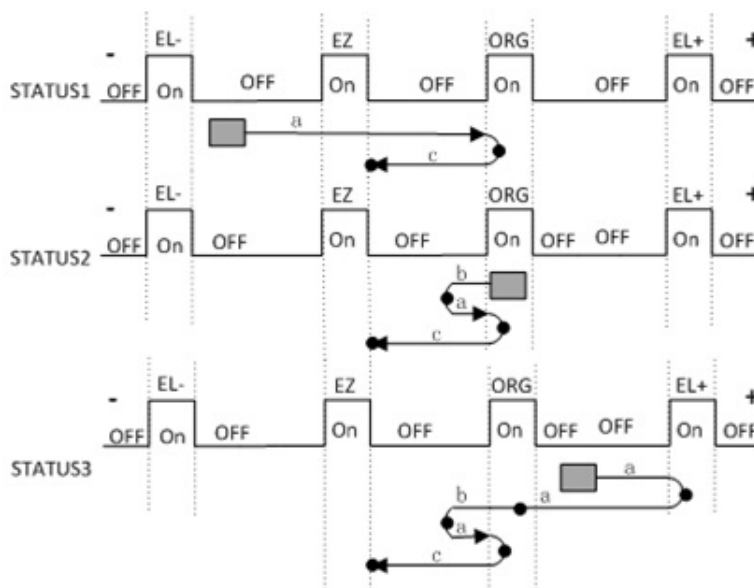


- STATUS1: If the object is out of the field of EZ signal and ORG signal, when home command is written, firstly, the object will move until ORG signal occurring, then continue to move until EZ signal occurring.
 - STATUS2: If the object is in the field of ORG signal, the home command is written. Firstly, the object reserves direction and moves, the ORG signal disappears, then reverses direction again and continues to move, the ORG signal occurs again. At last, motion is stopped when EZ signal occurring.
 - STATUS3: If there is no ORG signal occurring. EL signal happens before ORG-signal, the object reverses direction when EL signal happens and continues to move, and then the ORG signal from happening to disappearing. Reverses direction again, continues to move, the ORG signal will happen and disappear again. At last, motion is stopped when EZ signal occurring.
10. **MODE10_AbsSearch_NegRef: Search ORG + NegEZ, Move (Dir) ->SearchORG ->Stop ->Move (-Dir) ->touch EZ ->Stop.**

Firstly, object moves in the way of MODE7_AbsSearch, and then moves in opposite direction until EZ signal occurring.

For example:

Dir: Positive. Limit logic: Active High. ORG logic: Active High.

AbsSearch + NegEZ

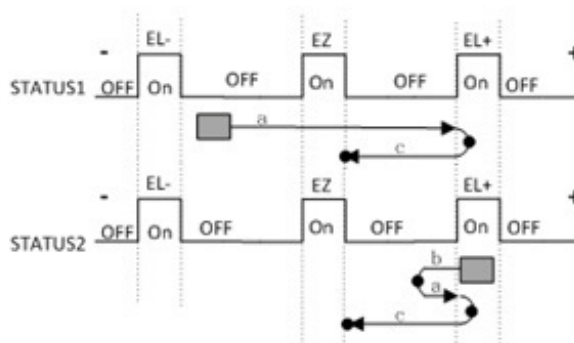
- STATUS1: If the object is out of the field of EZ signal and ORG signal, when home command is written. Firstly, the object will move until ORG signal occurring, then reverse direction and continue to move until EZ signal occurring.
- STATUS2: If the object is in the field of ORG signal, the home command is written, firstly, the object reserves direction and moves, the ORG signal disappears, then reverses direction again and continues to move, the ORG signal occurs again, reverses direction and moves. At last, motion is stopped when EZ signal occurring.
- STATUS3: If there is no ORG signal occurring. EL signal happens before ORG signal, the object reverses direction when EL signal happens and continues to move, and then the ORG signal from happening to disappearing. Reverses direction again, continues to move, the ORG signal will happen again, then reverses direction. At last, motion is stopped when EZ signal occurring.

11. MODE11_LmtSearch_Ref: Search EL +NegEZ, Move (Dir) ->Search EL ->Stop->Move (-Dir) ->touch EZ ->Stop.

Firstly, object moves in the way of MODE8_LmtSearch, and then moves in opposite direction until EZ signal occurring.

For example:

Dir: Positive. Limit logic: Active High.

LmtSearch + NegEZ

- STATUS1: When object is not in field of limit signal. Firstly, the object will move until EL signal occurring, then reverse direction and continue to move until EZ signal occurring.
 - STATUS2: When object is in the field of limit signal. Firstly, the object reserves direction and moves, the EL signal disappears, then reverses direction again and continues to move, the EL signal occurs again, reverses direction again and moves. At last, motion is stopped when EZ signal occurring.
12. **MODE12_AbsSearchRefind: Search ORG +Refind ORG, Move (Dir) ->SearchORG ->Stop->Move (-Dir) ->Leave ORG(FL) ->Stop-> Move (-Dir)->RefindORG(FL)->Stop.**

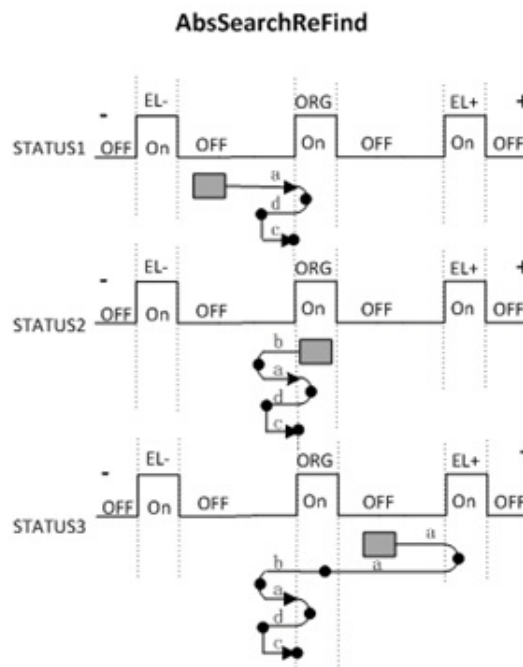
Firstly, axis moves in the way of MODE7_AbsSearch, and then moves uniformly in opposite direction at Vellow until ORG signal disappears. Then, axis reverses the direction again and continues to move uniformly at Vellow until ORG signal occurs.

For example:

Dir: Positive.

ORG Logic: Active High.

Limit Logic: Active High.



AbsSearch process has three situations. For detailed information, see about descriptions in MODE7_AbsSearch.

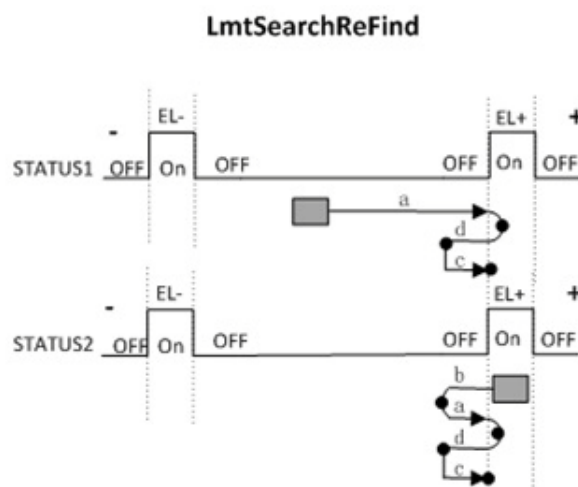
13. **MODE13_LmtSearchRefind: Search EL +Refind EL, Move (Dir) ->Search EL ->Stop->Move (-Dir) ->Leave EL(FL) ->Stop-> Move (-Dir)->Refind EL(FL)->Stop.**

Firstly, axis moves in the way of MODE8_LmtSearch, and then moves uniformly in opposite direction at Vellow until EL signal disappears. Then, axis reverses the direction again and continues to move uniformly at Vellow until EL signal occurs.

For example:

Dir: Positive.

Limit Logic: Active High.



14. **MODE14_AbsSearchRefind_Ref: Search ORG +Refind ORG+EZ, Move (Dir) ->Search ORG ->Stop->Move (-Dir) ->Leave ORG(FL) ->Stop-> Move (-Dir)->Refind ORG(FL)->Stop->Move (Dir) ->touch EZ ->Stop.**

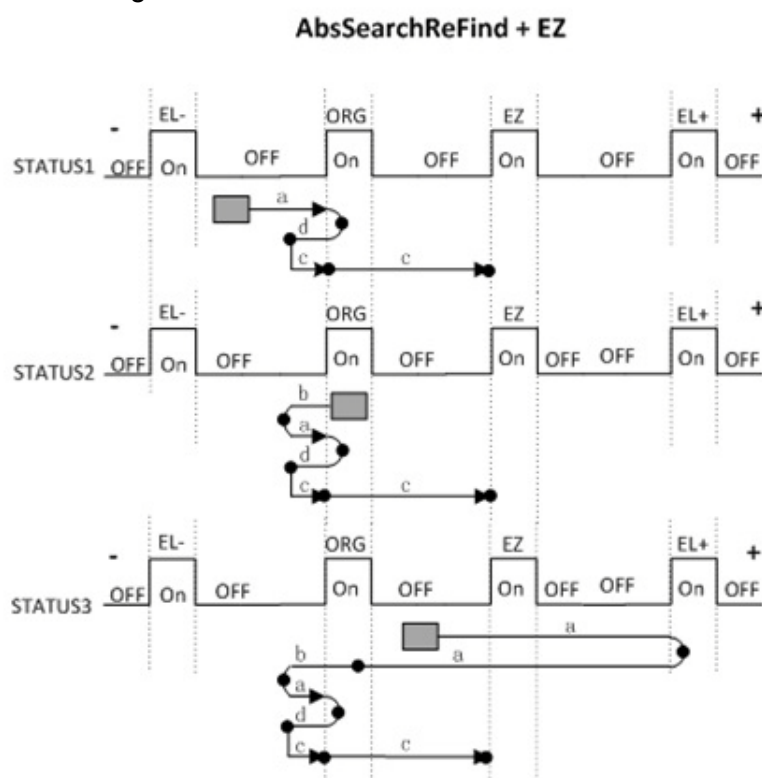
Firstly, axis moves in the way of MODE7_AbsSearch, and then moves uniformly in opposite direction at VeLow until ORG signal disappears. Then, axis reverses the direction again and continues to move uniformly at VeLow until ORG signal occurs. At last, axis moves in the same direction to Z phase.

For example:

Dir: Positive.

Limit Logic: Active High.

ORG Logic: Active High.



AbsSearch process has three situations. For detailed information, see about descriptions in MODE7_AbsSearch.

15. **MODE15_AbsSearchRefind_NegEZ: Search ORG +Refind ORG+NegEZ,Move (Dir) ->Search ORG ->Stop->Move (-Dir) ->Leave ORG (FL)->Stop->Move (-Dir)->Refind ORG(FL)-> Stop-> Move (-Dir) ->touch EZ ->Stop.**

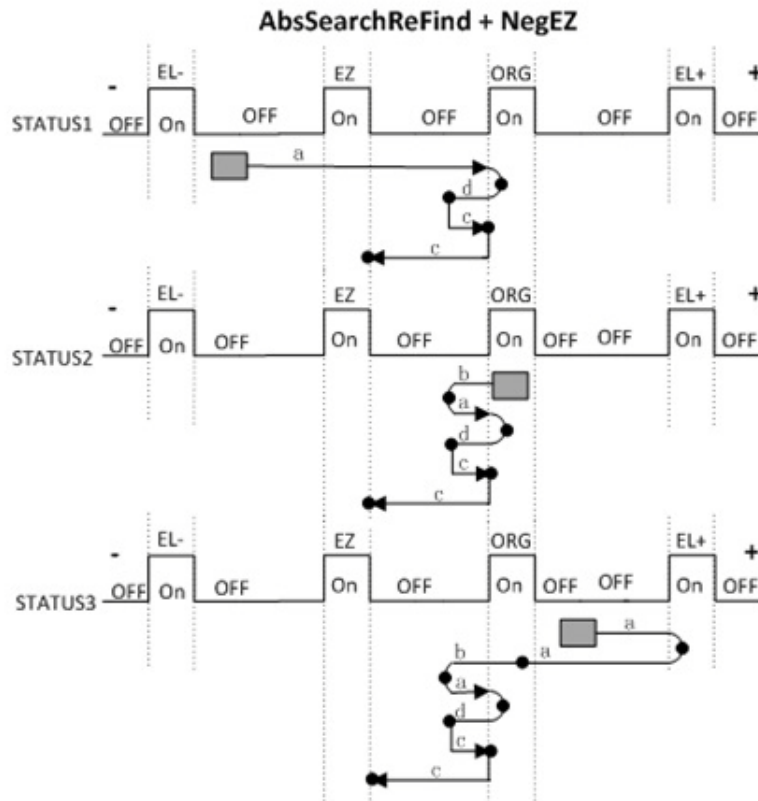
Firstly, axis moves in the way of MODE7_AbsSearch, and then moves uniformly in opposite direction at VelLow until ORG signal disappears. Then, axis reverses the direction again and continues to move uniformly at VelLow until ORG signal occurs. At last, axis moves in opposite direction again until EZ signal occurs.

For example:

Dir: Positive.

Limit Logic: Active High.

ORG Logic: Active High.



AbsSearch process has three situations. For detailed information, see about descriptions in MODE7_AbsSearch.

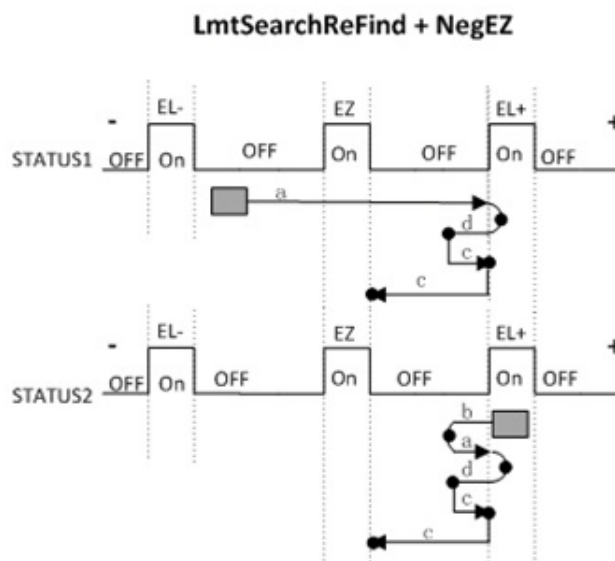
16. **MODE16_LmtSearchRefind_Ref: Search EL +Refind EL+EZ, Move (Dir) ->Search EL ->Stop->Move (-Dir) ->Leave EL(FL) ->Stop-> Move (-Dir)->RefindEL(FL)->Stop->Move (-Dir) ->touch EZ ->Stop.**

Firstly, axis moves in the way of MODE8_LmtSearch, and then moves uniformly in opposite direction at VelLow until EL signal disappears. Then, axis reverses the direction again and continues to move uniformly at VelLow until EL signal occurs. At last, axis moves in opposite direction again until EZ signal occurs.

For example:

Dir: Positive.

Limit Logic: Active High.



LmtSearch process has three situations. For detailed information, see about descriptions in MODE8_LmtSearch.

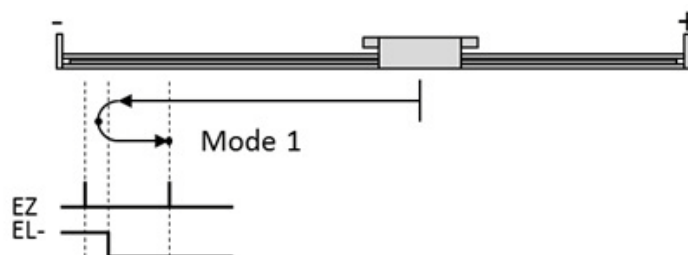
Type 2: Defined by CiA402

The 35 types of typical Home motion defined by CiA402 and can be set through Acm_AxHome and Acm_AxMoveHome functions.

35 Home modes defined by CiA402 (The following introduction of homing mode and figure are referenced by CiA402) and there will be a little difference between different motor driver. To know the correct meaning of home function of motor drive, refer to manuals provided by the motor drive manufacturer.

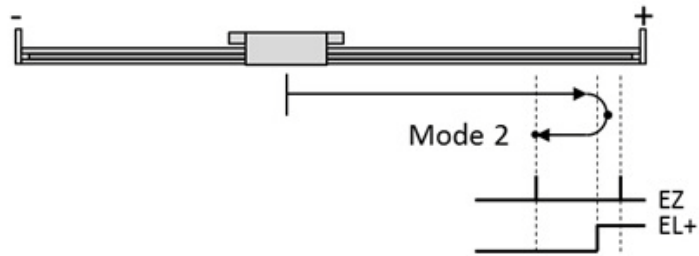
Method 1: Homing on the negative limit switch and index pulse

Using this method the initial direction of movement is leftward if the negative limit switch is inactive (here shown as low). The home position is at the first index pulse to the right of the position where the negative limit switch becomes inactive.



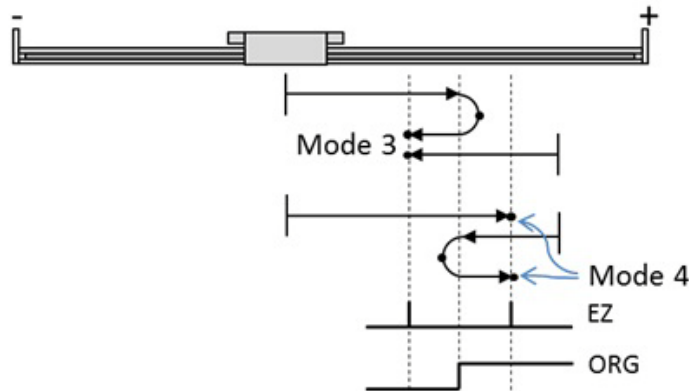
Method 2: Homing on the positive limit switch and index pulse

Using this method the initial direction of movement is rightward if the positive limit switch is inactive (here shown as low). The position of home is at the first index pulse to the left of the position where the positive limit switch becomes inactive.



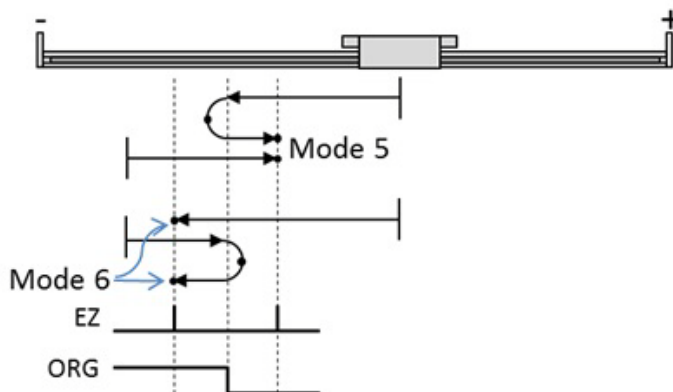
Methods 3 and 4: Homing on the positive home switch and index pulse

Using methods 3 or 4 the initial direction of movement is dependent on the state of the home switch. The home position is at the index pulse to either to the left or the right of the point where the home switch changes state. If the initial position is sited so that the direction of movement must reverse during homing, the point at which the reversal takes place is anywhere after a change of state of the home switch.



Methods 5 and 6: Homing on the negative home switch and index pulse

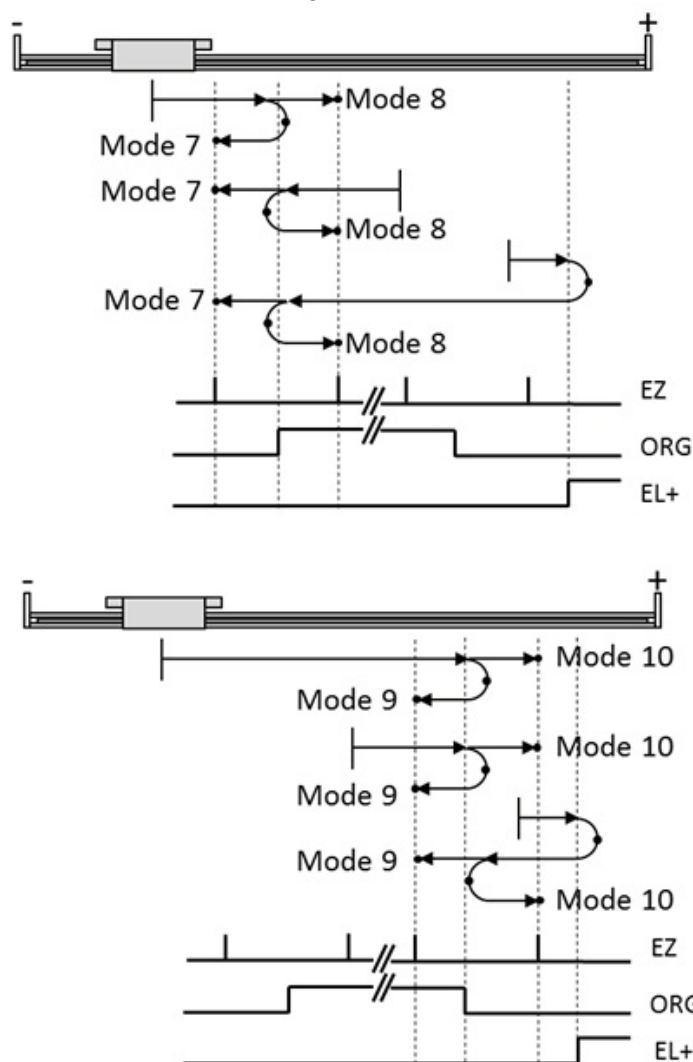
Using methods 5 or 6 the initial direction of movement is dependent on the state of the home switch. The home position is at the index pulse to either to the left or the right of the point where the home switch changes state. If the initial position is sited so that the direction of movement must reverse during homing, the point at which the reversal takes place is anywhere after a change of state of the home switch.

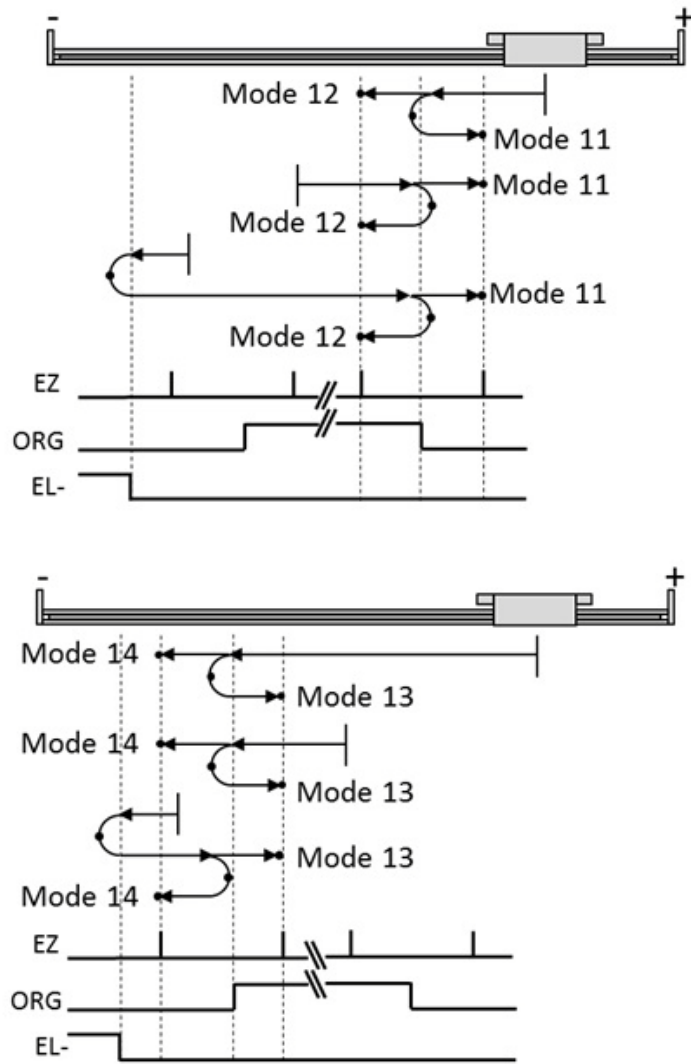


Methods 7 to 14: Homing on the home switch and index pulse

These methods use a home switch which is active over only portion of the travel, in effect the switch has a 'momentary' action as the axle's position sweeps past the switch.

Using methods 7 to 10 the initial direction of movement is to the right, and using methods 11 to 14 the initial direction of movement is to the left except if the home switch is active at the start of the motion. In this case the initial direction of motion is dependent on the edge being sought. The home position is at the index pulse on either side of the rising or falling edges of the home switch, as shown in the following two diagrams. If the initial direction of movement leads away from the home switch, the drive must reverse on encountering the relevant limit switch.



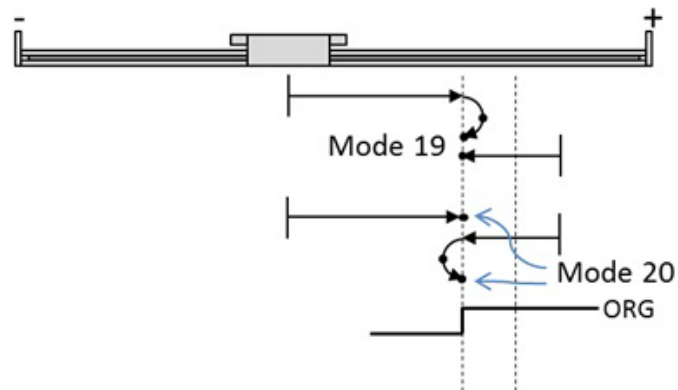


Methods 15 and 16: Reserved

These methods are reserved for future expansion of the homing mode.

Methods 17 to 30: Homing without an index pulse

These methods are similar to methods 1 to 14 except that the home position is not dependent on the index pulse but only dependent on the relevant home or limit switch transitions. For example methods 19 and 20 are similar to methods 3 and 4 as shown in the following diagram.

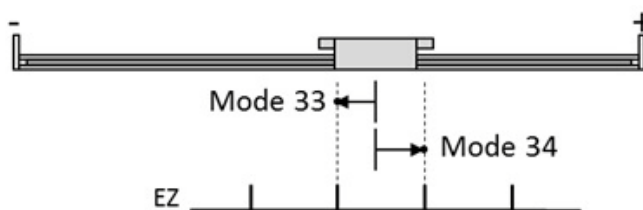


Methods 31 and 32: Reserved

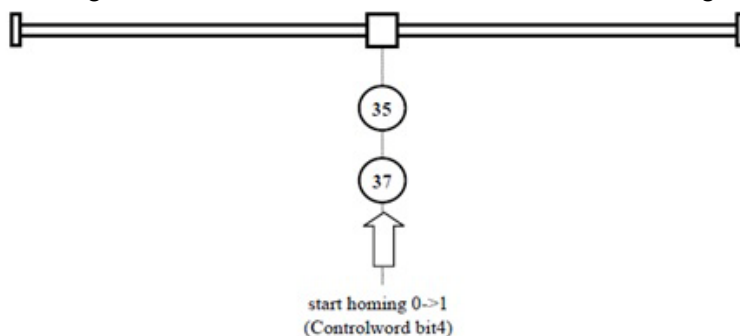
These methods are reserved for future expansion of the homing mode.

Methods 33 to 34: Homing on the index pulse

Using methods 33 or 34 the direction of homing is negative or positive respectively. The home position is at the index pulse found in the selected direction.

**Methods 35 and 37: Homing on the current position**

The method 35 is defined by CiA402 and the method 37 is defined by and can be only used in Panasonic EtherCAT Motor. In method 35 the current position is taken to be the home position. Although Method 35 and 37 are the same functions, use Method 37 according to the ETG standard at the time of a new design.



Method 37 is defined by Panasonic EtherCAT Motor

9.2.9.3 Flow Chart

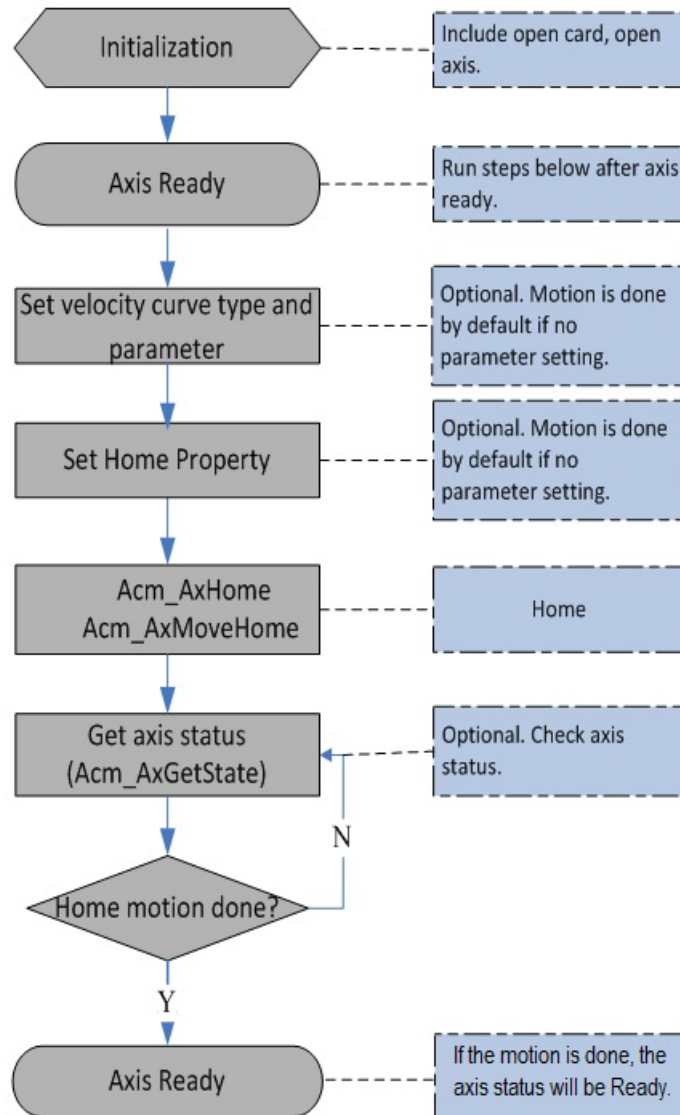


Figure 9.9 Home motion flow chart

9.2.9.4 Example

This example executes home motion on 0 axis. Refer to the table below for setting parameters.

Function	Executes home motion on 0 axis
Home mode	MODE8_LmtSearch. This is a mode of searching transformation of limit signal from no signal to signal occurring.
Cross distance	100
Initial velocity	2000PPU/S
Operating velocity	8000PPU/S
Acceleration	10000PPU/S ²
Deceleration	10000PPU/S ²
Velocity curve	T-Curve

VC code:

```

HAND m_Axishand[32];
U32 Ret; // Function return value
--- Initialization---
// Set parameters
Ret =Acm_SetF64Property(m_Axishand[0],PAR_AxVelLow,2000); // Initial velocity
Ret = Acm_SetF64Property(m_Axishand[0],PAR_AxVelHigh,8000); // Operating
velocity
Ret =Acm_SetF64Property(m_Axishand[0],PAR_AxAcc,10000); // Acceleration
Ret = Acm_SetF64Property(m_Axishand[0],PAR_AxDec,10000); // Deceleration
Ret = Acm_SetF64Property(m_Axishand[0],PAR_AxJerk,0); //T-Curve
Ret = Acm_SetU32Property(m_Axishand[0], CFG_AxEILogic, 0); // LMT actvie low
Ret = Acm_SetU32Property(m_Axishand[0], CFG_AxOrgLogic, 0); //ORG actvie
low
Ret =Acm_SetU32Property(m_Axishand[0], CFG_AxEzLogic, 0); // EZ actvie low
Ret = Acm_SetU32Property(m_Axishand[0],PAR_AxHomeExSwitchMode, 0); //
Enable sensor
Ret = Acm_SetF64Property(m_Axishand[0], PAR_AxHomeCrossDistance, 100);
//Cross distance
Ret = Acm_AxHome(m_Axishand[0], MODE8_LmtSearch, 0);
//HomeMode=MODE8_LmtSearch, come Home in forward direction
//Get axis status and position
F64 CmdPos;
F64 ActPos;
U16 State;
Acm_AxGetCmdPosition(m_Axishand[0],&CmPos); //Get 0 axis command position
Acm_AxGetActualPosition(m_Axishand[0],&ActPos); //Get 0 axis actual position
Acm_AxGetState(m_Axishand[0],&State); //Get 0 axis status

```

Check function returned value to judge the function execution status and establish error handle mechanism is recommended to ensure the correct result. Refer to the Home example.

9.2.10 Axis Event

9.2.10.1 Function and Property

Axis event functions are shown in Table 9.17, the functions in table can be called directly in program.

Table 9.17: Axis event functions

Function	Description
Acm_EnableMotionEvent	Enable / Disable motion event
Acm_CheckMotionEvent	Check axis and groups enabled motion event status.
Acm_AxMoveAbs	Command absolute point-to-point motion
Acm_AxMoveRel	Command relative point-to-point motion
Acm_AxMoveVel	To command axis to make a never ending movement with a specified velocity
Acm_AxSimStartSuspend Abs	Set the axis in waiting state for simultaneous absolute point-to-point motion
Acm_AxSimStartSuspend Rel	Set the axis in waiting state for simultaneous relative point-to-point motion
Acm_AxSimStartSuspend Vel	Set the axis in waiting state for simultaneous continuous motion
Acm_AxSimStart	Start all axis waiting for start trigger
Acm_AxSimStop	Stop all axis triggered by simultaneous signal
Acm_AxSetExtDrive	Enable or disable external drive mode
Acm_AxJog	Assign axis to do Jog move
Acm_AxHome	To command axis to start typical home motion
Acm_AxSetCmdPosition	Set axis command position
Acm_AxGetCmdPosition	Get axis command position
Acm_AxGetActualPosition	Get axis actual (feedback) position
Acm_AxStopDec	Decelerated stop
Acm_AxStopEmg	Emergency stop (No deceleration)
Acm_AxStopDecEx	Command the axis to stop and specify the deceleration
Acm_AxGetState	Get states of axis
Acm_AxResetError	Reset error when axis is error-stop
Acm_SetU32Property	Set property (Unsigned 32 -bit integer)
Acm_SetI32Property	Set property (Signed 32 -bit integer)
Acm_SetF64Property	Set property (Double)
Acm_GetU32Property	Get property (Unsigned 32 -bit integer)
Acm_GetI32Property	Get property (Signed 32 -bit integer)
Acm_GetF64Property	Get property (Double)

Axis event related properties as shown in Table 9.18 the property can't be called directly, but set and get property values by calling related property function. Refer to Appendix for more detail information.

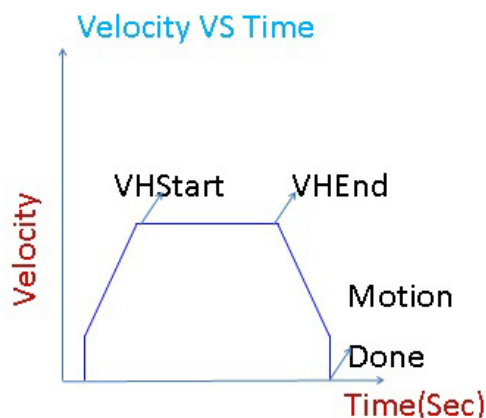
Table 9.18: Axis event properties

Parameter	Description
PAR_AxVelLow	Set/get initial velocity of axis
PAR_AxVelHigh	Set/get operating velocity of axis
PAR_AxAcc	Set/get acceleration of this axis
PAR_AxDec	Set/get deceleration of this axis
PAR_AxJerk	Set/get the type of velocity profile: T-Curve or S-Curve
Configuration	Description
CFG_AxMaxVel	Configure the max velocity for the motion axis
CFG_AxMaxAcc	Configure the max acceleration for the motion axis
CFG_AxMaxDec	Configure the max deceleration for the motion axis
Feature	Description
FT_AxMaxVel	Get axis supported max velocity
FT_AxMaxAcc	Get axis supported max acceleration
FT_AxMaxDec	Get axis supported max deceleration
FT_AxMaxJerk	Get axis supported max jerk

9.2.10.2 Description

Axis event includes Motion Done, VHStart, VHEnd, Compare, Latch, and LatchBuffer. This section focus on Motion Done, VHStart and VHEnd.

Refer to Appendix for detail information of other events.



Motion Done: Interrupt produced by hardware trigger when single-axis motion done.

VHStart: Interrupt produced by hardware trigger when single-axis velocity reaches the operating velocity.

VHSEnd: Interrupt produced by hardware trigger when single-axis velocity slows down.

Note! *VHStart and VHSEnd means the start of operating velocity and the end of operating velocity, respectively.*



If you want to get event status of axis or groups, you should enable these events by calling `Acm_EnableMotionEvent`. Realize Enable / Disable event of each axis by setting parameter `AxEvtStatusArray[N]` which is 32 bits data type, each bit represents different Event types. N represents the axes of device.

For example, if the axis number of motion card is 4, then `AxEvtStatusArray[0]` represents the all events of 0 axis. Each bit represents different Event.

Bit n = 1: Enable Event
 Bit n = 0: Disable Event
 Refer to the table below:

AxEvtStatus	Axis	Axis0	Axis1	Axis2	Axis3
Bit0		Evt_Ax_Motion_Done
Bit1		Evt_Ax_Compared
Bit2		Evt_Ax_Latched
Bit3		Evt_Ax_Error
Bit4		Evt_AxVHSTART
Bit5		Evt_AxVHEnd
Bit6		Evt_Ax_LatchBuffer
Bit7-31		ReServered

Create a new thread after enable axis Event to call Acm_CheckMotionEvent. AxEvt-StatusArray[N] which is 32 bits data type will return all interrupt event status. N represents the axes of device. For example, if the axis number of motion card is 4, then N=4. AxEvtStatusArray[0] represents all event status of 0 axis and each bit represents different Event type.

Bit n = 1: Enable Event
 Bit n = 0: Disable Event
 Refer to the table below:

AxEvtStatusAxis	Axis	Axis0	Axis1	Axis2	Axis3
Bit0		Evt_Ax_Motion_Done
Bit1		Evt_Ax_Compared
Bit2		Evt_Ax_Latched
Bit3		Evt_Ax_Error
Bit4		Evt_AxVHSTART
Bit5		Evt_AxVHEnd
Bit6		Evt_Ax_LatchBuffer
Bit7-31		ReServered

9.2.10.3 Flow Chart

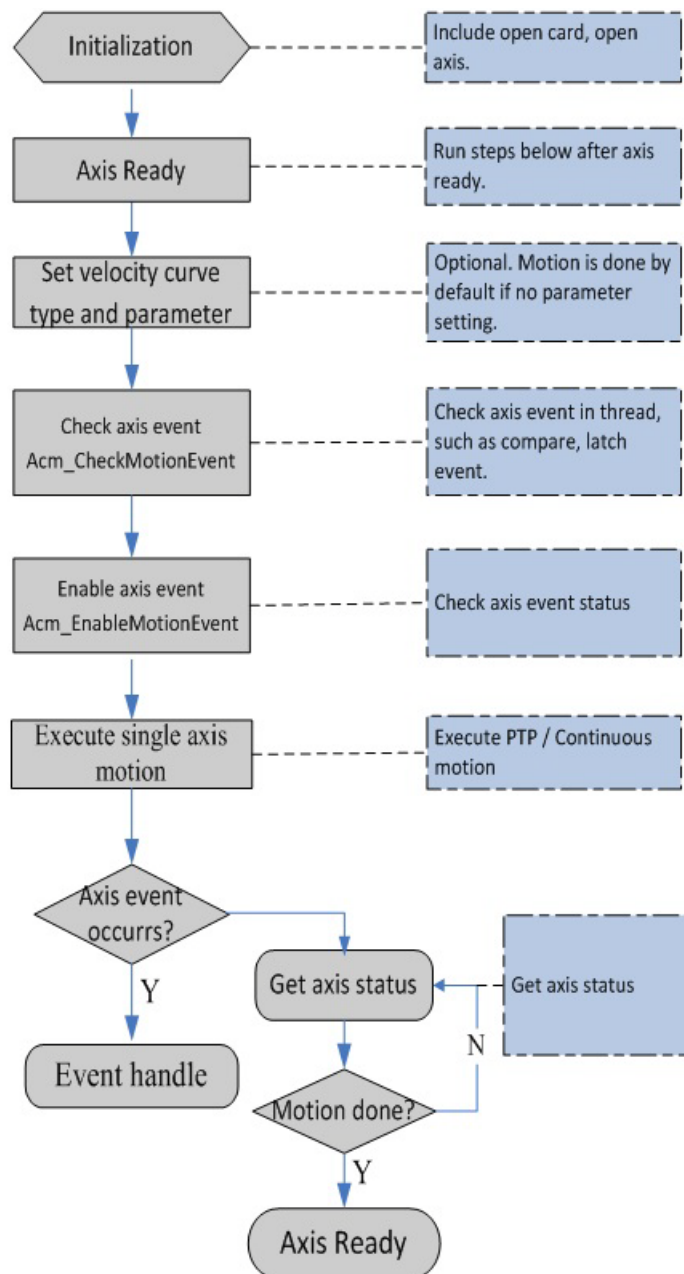


Figure 9.10 Axis event flowchart

9.2.10.4 Example

This example checks event on 0 axis. Refer to the table below for setting parameters.

Function	Check Motion Done, VHStart, VHEnd event on 0 axis
Initial velocity	2000PPU/S
Operating velocity	8000PPU/S
Acceleration	10000PPU/S ²
Deceleration	10000PPU/S ²
Velocity curve	T-Curve
Target position	10000PPU

VC code:

```
HAND m_Axishand[32];
U32 Ret;
U32 AxEnableEvtArray[4];
U32 GpEnableEvt[3];
U32 m_ulAxisCount;
BOOL m_blnit ;
--- Initialization---
//Set parameters
Ret =Acm_SetF64Property(m_Axishand[0],PAR_AxVelLow,2000);// Initial velocity
Ret = Acm_SetF64Property(m_Axishand[0],PAR_AxVelHigh,8000); //Operating
velocity
Ret =Acm_SetF64Property(m_Axishand[0],PAR_AxAcc,10000);// Acceleration
Ret =Acm_SetF64Property(m_Axishand[0],PAR_AxDec,10000);// Deceleration
Ret =Acm_SetF64Property(m_Axishand[0],PAR_AxJerk,0); //T-Curve
pThreadObject = AfxBeginThread( (AFX_THREADPROC)CDlg::CheckEvtThread,
this, THREAD_PRIORITY_TIME_CRITICAL, 0, 0, NULL); //Create thread object
m_ulAxisCount =4; //axis number of motion card
AxEnableEvtArray[0] |=EVT_AX_MOTION_DONE; //Enable motion done
AxEnableEvtArray[0] |=EVT_AX_VH_START; //Enable VHStart
AxEnableEvtArray[0] |=EVT_AX_VH_END; //Enable VHEND
Ret=Acm_EnableMotionEvent(m_Devhand,AxEnableEvtArray,GpEnableEvt,
m_ulAxisCount ,3); //Enable event
m_blnit = true;
UINT CDlg::CheckEvtThread(LPVOID ThreadArg)//Thread function for check event
{
    U32 Ret;
    U32 AxEvtStatusArray[4];
    U32 GpEvtStatusArray[3];
    CDlg *pThreadInfo;
    pThreadInfo = (CDlg*)ThreadArg;
    while (pThreadInfo->m_blnit)
    {
        Ret=Acm_CheckMotionEvent(pThreadInfo->m_Devhand,AxEvtStatusArray,
        GpEvtStatusArray,pThreadInfo->m_ulAxisCount,3,10000);//Checkevent sta-
        tus
    }
}
// Get axis status and position
F64 CmdPos;
F64 ActPos;
U16 State;
Acm_AxGetCmdPosition(m_Axishand[0],&CmPos); //Get 0 axis command position
Acm_AxGetActualPosition(m_Axishand[0],&ActPos); //Get 0 axis actual position
Acm_AxGetState(m_Axishand[0],&State); //Get 0 axis status
```

Check function returned value to judge the function execution status and establish error handle mechanism is recommended to ensure the correct result. Refer to the Event example.

9.2.11 PT Motion

9.2.11.1 Function and Property

PT motion functions are shown in Table 9.19, the functions in table can be called directly in program.

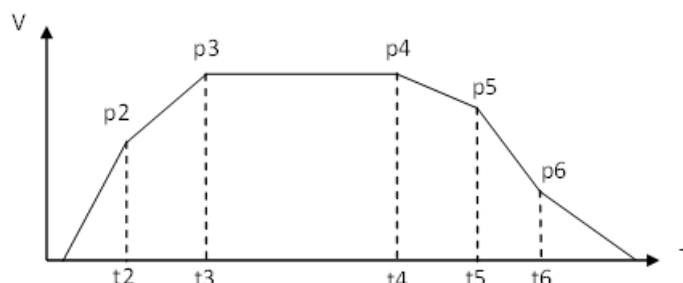
Table 9.19: PT motion functions

Function	Description
Acm_AxAddPTData	Add PT data into PT buffer
Acm_AxStartPT	Command PT motion start
Acm_AxCheckPTBuffer	Check PT buffer availability
Acm_AxResetPTData	Reset PT buffer
Acm_AxGetState	Get states of axis

9.2.11.2 Description

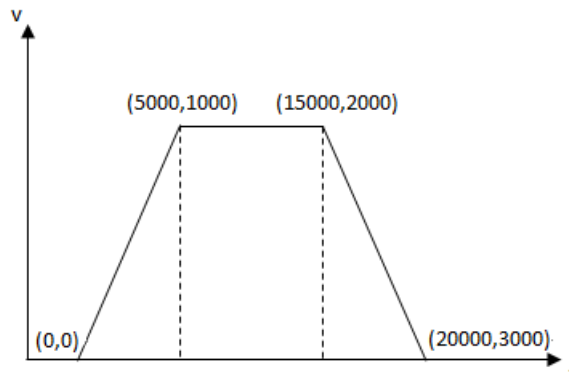
When the status of the axis is Ready, it will be able to execute PT movement. Refer to section 6.2 for axis status.

PT motion use position and time data to shape the velocity profile. Therefore the acceleration and deceleration motion can be manipulated according to the input PT data point.



As shown above, a complete velocity profile can be divided into several motion segments and each segment is based on the first data point (p_1, t_1) to calculate its displacement. Each data point is an absolute value with respect to the start point. The data point example is shown below.

	P(position, PPU)	T(time, ms)
Data1	0	0
Data2	5000	1000
Data3	15000	2000
Data4	20000	3000



PT motion provides two type operations, static mode and dynamic mode, for user to shape the velocity profile. In static mode, users need to load total data into the buffer before the PT motion start. In contrast, user can update data while PT motion is operating. In addition, when buffer is in static mode, user can set the “repeat” parameter to loop the PT data, but it is not support in dynamic operation.

If users want to update new data into the PT buffer after previous PT motion was initiated, users need to reset the buffer to clear the previous information.

9.2.11.3 Flow Chart

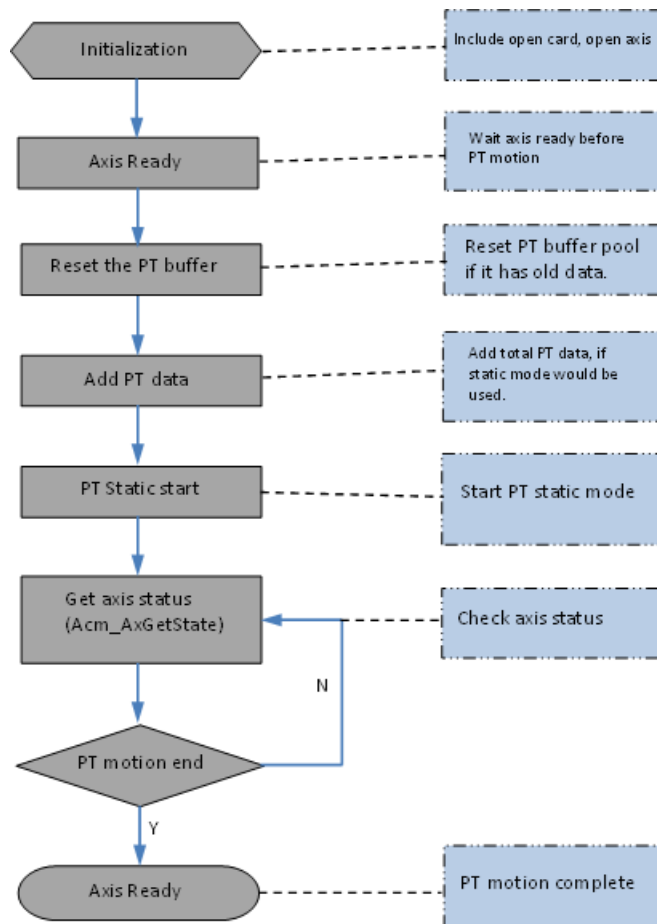


Figure 9.11 PT static mode flow chart

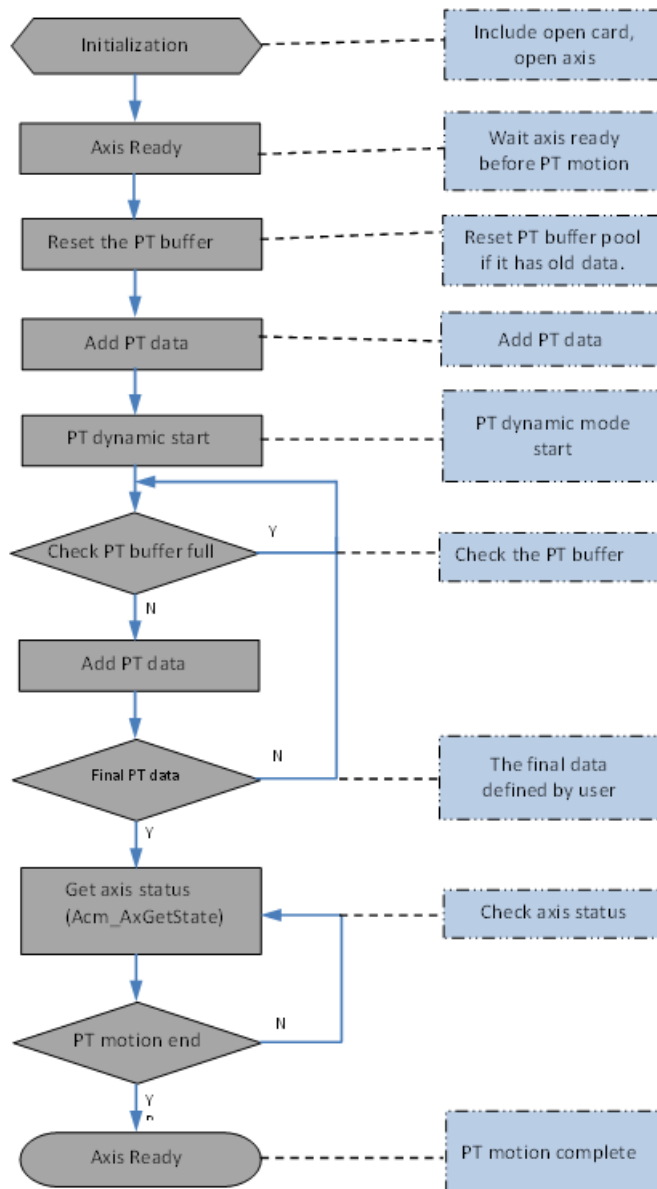


Figure 9.12 PT dynamic mode flow chart

9.2.11.4 Example

This example executes PT static motion on 0 axis. Please refer to the table below for setting parameters.

Function	Executes relative point to point motion on 0 axis (PCI-1203)
Initial velocity	N/A
Operating velocity	N/A
Acceleration	N/A
Deceleration	N/A
Velocity curve	N/A
Target position	N/A

VC code: PT Static mode

```
HAND m_Axishand[24]; //Axis handle
```

```
U32 Ret; //Function return value
```

```
---Initialization---
```

```
//Set parameters
```

```
Ret = Acm_AxResetPTData(m_Axishand[0]); //reset previous data in PT buffer.
```

```
Ret = Acm_AxAddPTData(m_Axishand[0], 0,0); //write first PT data point
```

```
Ret = Acm_AxAddPTData (m_Axishand[0], 5000,1000); //write PT data
```

```
Ret = Acm_AxAddPTData (m_Axishand[0], 15000,2000); //write PT data
```

```
Ret = Acm_AxAddPTData (m_Axishand[0], 20000,3000); //write PT data
```

```
Ret = Acm_AxAddPTData (m_Axishand[0], 21024,4024); //write PT data
```

```
Ret = Acm_AxAddPTData (m_Axishand[0], 23072,5048); //write PT data
```

```
Ret = Acm_AxAddPTData (m_Axishand[0], 24096,6072); //write PT data
```

```
//Execute PT motion in static mode and repeat two times.
```

```
Ret = Acm_AxStartPT(m_Axishand[0], PT_BUFFER_STATIC,2);
```

```
//Get axis status and position
```

```
F64 CmdPos;
```

```
F64 ActPos;
```

```
U16 State;
```

```
Acm_AxGetCmdPosition(m_Axishand[0],&CmdPos); //Get 0 axis command position
```

```
Acm_AxGetActualPosition(m_Axishand[0],&ActPos); // Get 0 axis actual position
```

```
Acm_AxGetState(m_Axishand[0],&State); // Get 0 axis status
```

Check function returned value to judge the function execution status and establish error handle mechanism is recommended to ensure the correct result.

The second example implements PT dynamic motion on 0 axis. Please refer to the table below for setting parameters.

Function	Executes relative point to point motion on 0 axis (PCI-1203)
Initial velocity	N/A
Operating velocity	N/A
Acceleration	N/A
Deceleration	N/A
Velocity curve	N/A
Target position	N/A

VC code: PT Dynamic mode

```

HAND m_Axishand[24]; //Axis handle
U32 Ret; //Function return value
---Initialization---
//Set parameters
Ret = Acm_AxResetPTData(m_Axishand[0]); //reset previous data in PT buffer.
Ret = Acm_AxAddPTData(m_Axishand[0], 0,0); //write first PT data point
Ret = Acm_AxAddPTData (m_Axishand[0], 5000,1000);//write PT data
Ret = Acm_AxAddPTData (m_Axishand[0], 15000,2000); //write PT data
Ret = Acm_AxAddPTData (m_Axishand[0], 20000,3000); //write PT data
//Execute PT motion in static mode and repeat two times.
Ret = Acm_AxStartPT(m_Axishand[0], PT_BUFFER_DYNAMIC,0);
//update PT data continuously.
Ret = Acm_AxAddPTData (m_Axishand[0], 21024,4024); //write PT data
Ret = Acm_AxAddPTData (m_Axishand[0], 23072,5048); //write PT data
Ret = Acm_AxAddPTData (m_Axishand[0], 24096,6072); //write PT data
Ret = Acm_AxAddPTData (m_Axishand[0], 29096,7072); //write PT data
Ret = Acm_AxAddPTData (m_Axishand[0], 39096,8072); //write PT data
Ret = Acm_AxAddPTData (m_Axishand[0], 44096,9072); //write PT data
Ret = Acm_AxAddPTData (m_Axishand[0], 49096,10072); //write PT data
Ret = Acm_AxAddPTData (m_Axishand[0], 59096,11072); //write PT data
Ret = Acm_AxAddPTData (m_Axishand[0], 64096,61272); //write PT data
//Get axis status and position
F64 CmdPos;
F64 ActPos;
U16 State;
Acm_AxGetCmdPosition(m_Axishand[0],&CmdPos); //Get 0 axis command position
Acm_AxGetActualPosition(m_Axishand[0],&ActPos); // Get 0 axis actual position
Acm_AxGetState(m_Axishand[0],&State); // Get 0 axis status

```

Check function returned value to judge the function execution status and establish error handle mechanism is recommended to ensure the correct result.

9.2.12 PVT Motion

9.2.12.1 Function and Property

PVT motion functions are shown in Table 9.20, the functions in table can be called directly in program.

Table 9.20: PVT functions

Function	Description
Acm_AxResetPVTable	Reset PVT table
Acm_AxLoadPVTable	Load PVT table into PVT buffer
Acm_AxStartPVT	Start PVT motion
Acm_AxGetState	Get states of axis

9.2.12.2 Description

PVT motion profiles the velocity according to position, velocity, and time information to satisfy the third order equation shown below, a proper parameters calculated from the equation can get a much smoother velocity motion.

$$p = at^3 + bt^2 + ct + d$$

$$\frac{dp}{dt} = v = 3at^2 + 2bt + c$$

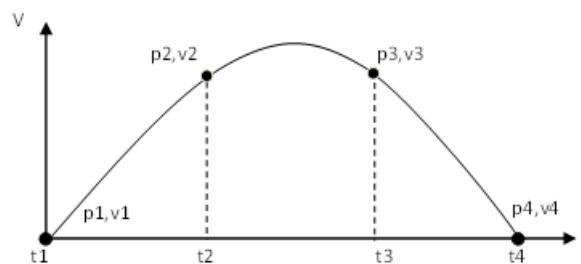
Substitute two adjacent point, (p_1, v_1, t_1) and (p_2, v_2, t_2) , into the equation can determine the parameters a, b, c, d

$$p_1 = at_1^3 + bt_1^2 + ct_1 + d$$

$$v_1 = 3at_1^2 + 2bt_1 + c$$

$$p_2 = at_2^3 + bt_2^2 + ct_2 + d$$

$$v_2 = 3at_2^2 + 2bt_2 + c$$



9.2.12.3 Flow Chart

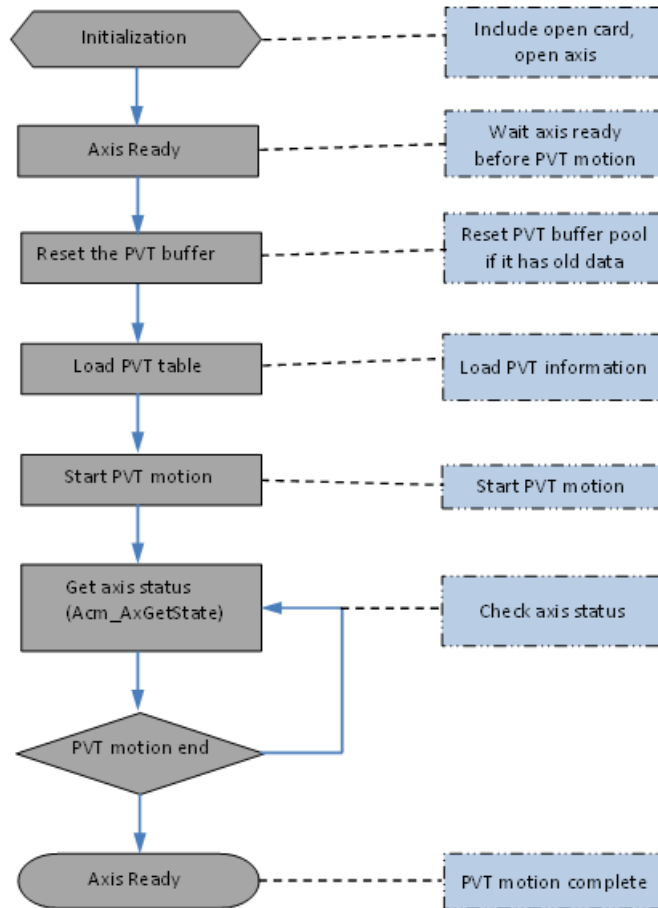


Figure 9.13 PVT motion flow chart

9.2.12.4 Example

This example executes PVT motion on 0 axis. Please refer to the table below for setting parameters.

Function	Executes continuous motion on 0 axis (PCI-1203)
Initial velocity	N/A
Operating velocity	N/A
Acceleration	N/A
Deceleration	N/A
Velocity curve	N/A
Move directions	N/A

VC code

```
HAND m_Axishand[24]; //Axis handle
U32 Ret; // Function return value
F64 pos[4]={0,5000,15000,20000}; //position table
F64 vel[4]={0,7500,7500,0}; //velocity table
F64 t[4]={0,1000,2333,3333}; //time table
--- Initialization---
// Set parameters
Ret = Acm_AxResetPVTTable(m_Axishand[0]); //reset previous data in PVT buffer.
Ret = Acm_AxLoadPVTTable(m_Axishand[0],4,pos,vel,t); //load PVT information into
buffer
// Execute Continuous motion and repeat the PVT dat 2 times
Ret = Acm_AxStartPVT (m_Axishand[0],2);
// Get axis status and position
F64 CmdPos;
F64 ActPos;
U16 State;
Acm_AxGetCmdPosition(m_Axishand[0],&CmdPos); // Get 0 axis command position
Acm_AxGetActualPosition(m_Axishand[0],&ActPos); // Get 0 axis actual position
Acm_AxGetState(m_Axishand[0],&State); // Get 0 axis status
```

Check function returned value to judge the function execution status and establish error handle mechanism is recommended to ensure the correct result.

9.3 Interpolation Motion

9.3.1 About this Section

Interpolation motion can realize multi-axis synchronization and is widely used in CNC devices such as CNC machine tools, high-speed machining technology.

Advantech motion control card provides interpolation modes are as follows:

- Linear interpolation
- Arc interpolation
- Helical interpolation

9.3.2 Linear Interpolation

9.3.2.1 Function and Property

Linear interpolation motion functions are shown in Table 9.19, the functions in table can be called directly in program.

Table 9.21: Point to point motion functions

Function	Description
Acm_GpMoveLinearRel	Command group to execute relative linear interpolation
Acm_GpMoveLinearAbs	Command group to execute absolute linear interpolation.
Acm_GpMoveDirectRel	Command group to execute relative direct linear interpolation
Acm_GpMoveDirectAbs	Command group to execute absolute direct linear interpolation

Table 9.21: Point to point motion functions

Acm_GpAddAxis	Add an axis to the specified group
Acm_GpRemAxis	Remove an axis from the specified group
Acm_GpClose	Remove all axis in the group and close the group handle
Acm_GpResetError	Reset group states
Acm_GpChangeVel	Command group to change the velocity while group is in line-interpolation motion
Acm_GpChangeVelByRate	Change the velocity of the current group motion according to the specified ratio
Acm_GpGetCmdVel	Get the current velocity of the group
Acm_GpStopDec	Command axis in this group to decelerate to stop
Acm_GpGetState	Get the group's current state
Acm_GpStopEmg	Command axis in this group to stop immediately without deceleration
Acm_AxSetCmdPosition	Set axis command position
Acm_AxGetCmdPosition	Get axis command position
Acm_AxGetActualPosition	Get axis actual (feedback) position
Acm_AxGetCmdVelocity	Get axis command velocity
Acm_SetU32Property	Set property (Unsigned 32 -bit integer)
Acm_SetI32Property	Set property (Signed 32 -bit integer)
Acm_SetF64Property	Set property (Double)
Acm_GetU32Property	Get property (Unsigned 32 -bit integer)
Acm_GetI32Property	Get property (Signed 32 -bit integer)
Acm_GetF64Property	Get property (Double)

Linear interpolation motion related properties as shown in Table 9.20, the property can't be called directly, but set and get property values by calling related property function. Refer to Appendix for more detail information.

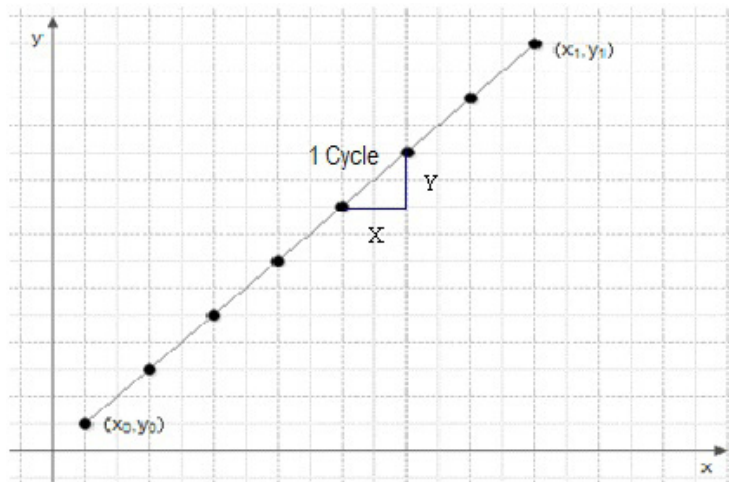
Table 9.22: Linear interpolation motion properties

Parameter	Description
PAR_GpVelLow	Set/get initial velocity of group
PAR_GpVelHigh	Set/get operating velocity of group
PAR_GpAcc	Set/get acceleration of this group
PAR_GpDec	Set/get deceleration of this group
PAR_GpJerk	Set/get the type of velocity profile: T-Curve or S-Curve
PAR_GpGroupID	Get the GroupID through GroupHandle
Configuration	Description
CFG_GpAxesInGroup	Get the specific axis information of group

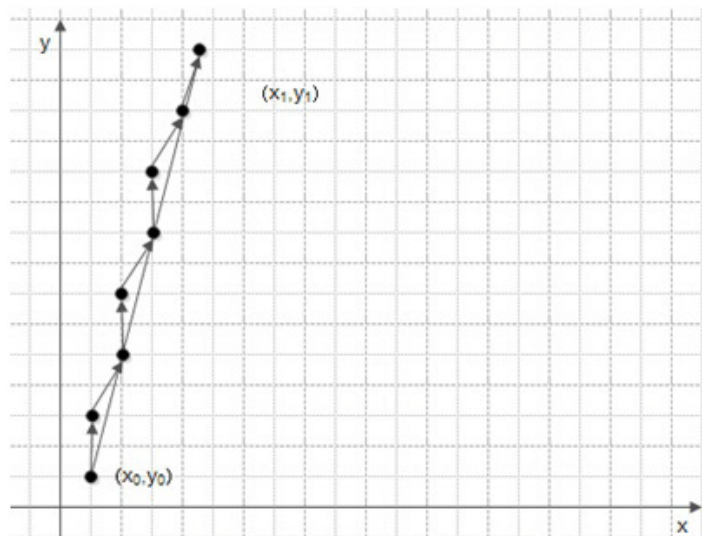
9.3.2.2 Description

Linear interpolation is a curve fitting method using linear polynomials. The algorithm divided the straight line from the starting point to the destination point of the data set into several data segment, and then sends the feed pulses to each coordinate (axis) according to the information after interpolation. The machine is moved some distances which corresponding to each sampling pulse in the specific direction whereby process the work piece to the desired contour shape.

As shown in the figure below, assume that a CNC machine tool is in the xy plane, the starting point coordinates (X_0, Y_0) , the end point coordinates (X_1, Y_1) . First, the space between the two end points will be sampled to a group of data, and then the motion core try to approach the interpolation points on a straight line along the point group. Within a movement cycle, the motion cores end the pulse according to the slope of each coordinate. When the ratio of slope of X-coordinate to that of Y is 1:1, the pulse transmits to the X-axis and Y-axis will be the same. Corresponding to each pulses of each sampling time, the machine is moved certain distance in the corresponding direction, and ultimately achieve the end goal.



When the ratios of slope of X coordinate to that of Y is not 1: 1, for example, the slope of Y coordinate is steeper than X coordinate. In this case, machine will move through the Y coordinate for a certain distance, and then it will start to move another axis to reach the sampling point. And finally, it will achieve the goal of the contour.



There are two methods of linear interpolation in PCI-1203:

Line interpolation: The interpolation's speed is divided into even line speed for every axis, axis moves at this even line speed. Mostly, line interpolation is applied to the axis assembled as right angle.

Direct interpolation: Direct interpolation's line speed is set into master axis (The smallest Physical ID axis) and other axes in group start/stop as the same time as master axis. Mostly, direct interpolation is applied to the axis assembled as oblique-angle.

9.3.2.3 Flow Chart

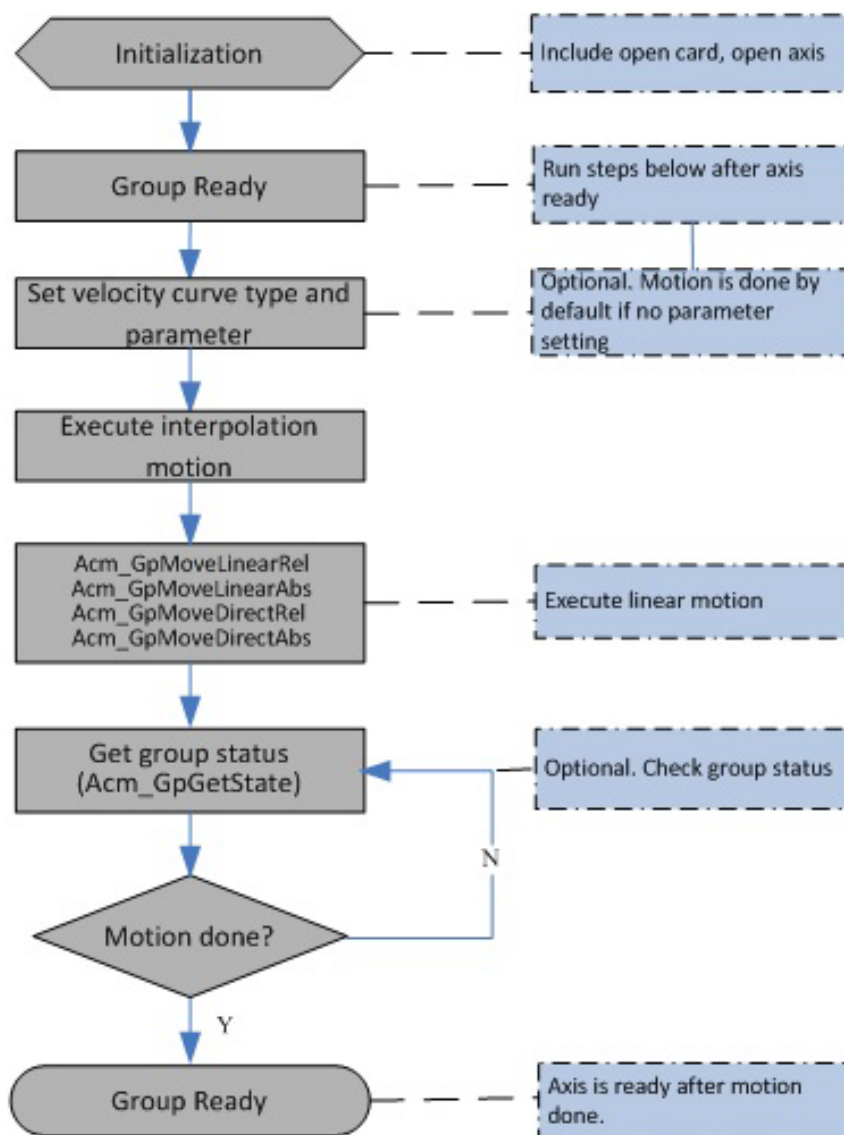


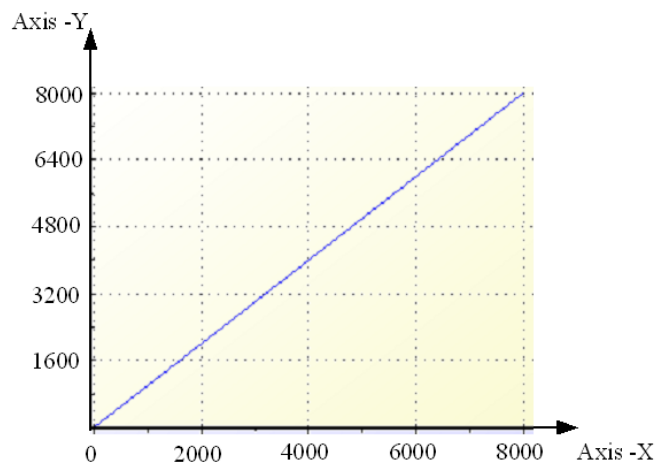
Figure 9.14 Linear interpolation motion flow chart

9.3.2.4 Example

This example executes linear interpolation on X and Y axis. Suppose a tool in CNC motion start from the original point and need to do linear interpolation to the position (8000,8000). Refer to the table below for setting parameters.

Function	Linear interpolation on two axes (PCI-1203)
Initial velocity	2000PPU/S
Operating velocity	8000PPU/S
Acceleration	10000PPU/S ²
Deceleration	10000PPU/S ²
Velocity curve	T-Curve
Initial position	XY(0,0)
Target position	XY(8000,8000)

The result is shown below:



VC Code:

```
HAND m_GpHand;
U32 Ret;
F64 m_End[2] = {8000,8000};
--- Initialization---

//Set parameters
Ret =Acm_SetF64Property(m_GpHand,PAR_GpVelLow,2000);//Initial velocity
Ret =Acm_SetF64Property(m_GpHand,PAR_GpVelHigh,8000);//Operating velocity
Ret =Acm_SetF64Property(m_GpHand,PAR_GpAcc,10000);//Acceleration
Ret =Acm_SetF64Property(m_GpHand,PAR_GpDec,10000);//Deceleration
Ret =Acm_SetF64Property(m_GpHand,PAR_GpJerk,0);//T-Curve
//Linear interpolation
Ret =Acm_GpMoveRel(m_GpHand,m_End);//Relative linear interpolation
//Get group status
U16 GpState;
Acm_GpGetState(m_GpHand,&GpState); //Get group status
```

Check function returned value to judge the function execution status and establish error handle mechanism is recommended to ensure the correct result. Refer to the Line and Direct example.

9.3.3 Arc Interpolation

9.3.3.1 Function and Property

Arc interpolation motion functions are shown in Table 9.21, the functions in table can be called directly in program.

Table 9.23: Arc interpolation motion functions

Function	Description
Acm_GpMoveCircularRel	Command group to execute relative arc interpolation.
Acm_GpMoveCircularAbs	Command group to execute absolute arc interpolation
Acm_GpMoveCircularRel_3P	Command group to execute relative arc interpolation by three specified points.
Acm_GpMoveCircularAbs_3P	Command group to execute absolute arc interpolation by three specified points
Acm_GpMoveArcRel_Angle	Command group to execute relative arc interpolation by center coordinates, angle and direction of rotation.
Acm_GpMoveArcAbs_Angle	Command group to execute absolute arc interpolation by center coordinates, angle and direction of rotation.
Acm_GpMove3DArcRel	Command group to execute relative 3D arc interpolation by specified points
Acm_GpMove3DArcAbs	Command group to execute absolute 3D arc interpolation by specified points
Acm_GpMove3DArcAbs_V	Command group to execute absolute 3D arc interpolation by normal vector
Acm_GpMove3DArcRel_V	Command group to execute relative 3D arc interpolation by normal vector
Acm_GpAddAxis	Add an axis to the specified group
Acm_GpRemAxis	Remove an axis from the specified group
Acm_GpClose	Remove all axis in the group and close the group handle
Acm_GpResetError	Reset group states
Acm_GpChangeVel	Command group to change the velocity while group is in line-interpolation motion
Acm_GpChangeVelByRate	Change the velocity of the current group motion according to the specified ratio
Acm_GpGetCmdVel	Get the current velocity of the group
Acm_GpStopDec	Command axis in this group to decelerate to stop
Acm_GpStopEmg	Command axis in this group to stop immediately without deceleration
Acm_GpGetState	Get the group's current state
Acm_AxSetCmdPosition	Set axis command position
Acm_AxGetCmdPosition	Get axis command position
Acm_AxGetActualPosition	Get axis actual (feedback) position
Acm_AxGetCmdVelocity	Get axis command velocity
Acm_SetU32Property	Set property (Unsigned 32-bit integer)
Acm_SetI32Property	Set property (Signed 32-bit integer)
Acm_SetF64Property	Set property (Double)
Acm_GetU32Property	Get property (Unsigned 32-bit integer)
Acm_GetI32Property	Get property (Signed 32-bit integer)
Acm_GetF64Property	Get property (Double)

Arc interpolation motion related properties as shown in Table 9.22, the property can't be called directly, but set and get property values by calling related property function. Refer to Appendix for more detail information.

Table 9.24: Arc interpolation motion properties

Parameter	Description
PAR_GpVelLow	Set/get initial velocity of group
PAR_GpVelHigh	Set/get operating velocity of group
PAR_GpAcc	Set/get acceleration of this group
PAR_GpDec	Set/get deceleration of this group
PAR_GpJerk	Set/get the type of velocity profile: T-Curve or S-Curve
PAR_GpGroupID	Get the GroupID through GroupHandle
Configuration	Description
CFG_GpAxesInGroup	Get the specific axis information of group

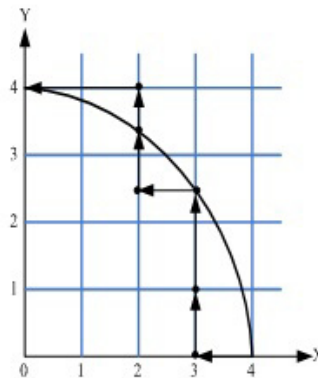
9.3.3.2 Description

The algorithm of arc interpolation is first described the data information of arc segment between two end points and calculates the points approached to the real arc curve. Advantech motion control cards support 2-axis and 3-axis arc interpolation. Multiple input methods are suitable for a variety of applications.

1. 2-Axis Arc Interpolation

The reference plane of 2-axis arc interpolation can be any of XY, YZ, XZ plane.

Assume that a CNC machine tools in the first quadrant of the xy plane run an arc path in the reverse direction, the center of the circle is the origin point, the radius is 4, the coordinate of start point A(4, 0) and end point B(0, 4). The arc interpolation process is shown below

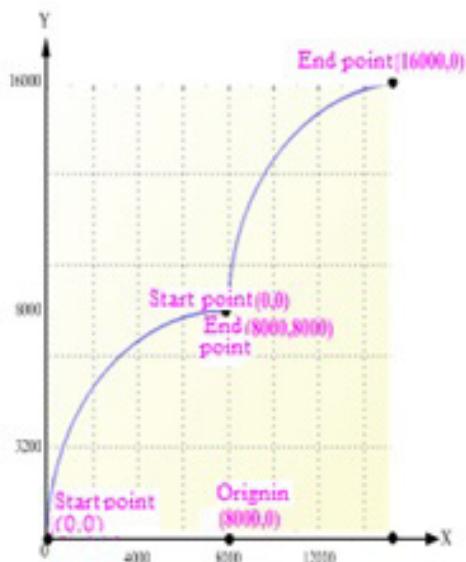


There are three ways to execute 2-axis arc interpolation:

Method1:**Execute arc interpolation according to start, center and end points**

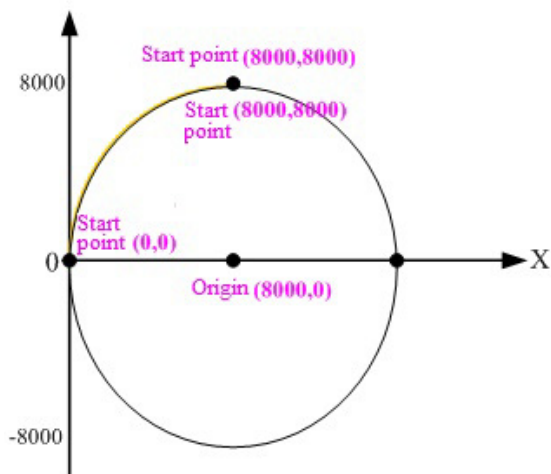
Users can input center point and end point to execute relative / absolute arc interpolation by calling `Acm_GpMoveCircularRel/Acm_GpMoveCircularAbs`.

For example, execute arc interpolation by setting center point (8000, 0), and end point (8000, 8000), then the result of successively executing twice relative arc interpolation is shown as follows. That is, the current command position will be the start point of relative arc interpolation.



For example, execute arc interpolation by setting center point (8000, 0), and end point (8000, 8000), then the result of successively executing twice relative arc interpolation is shown as follows. That is, the current command position will be the start point of relative arc interpolation.

The result of successively executing twice absolute arc interpolation is shown as follows:



The yellow part of the figure is the first implementation result and the start point of the second implementation result is the end point of the first implementation. Absolute arc interpolation sets absolute zero as reference position.

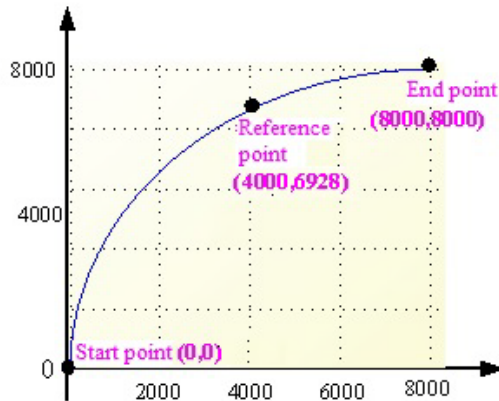
Method2:

Execute arc interpolation according to coordinates of three points

Users can input reference point and end point to execute relative / absolute arc interpolation by calling `Acm_GpMoveCircularRel_3P/Acm_GpMoveCircularAbs_3P`.

For example, execute relative/ absolute arc interpolation by setting reference point (4000,6928) and end point (8000,8000).

The default start point is (0, 0). The result of successively executing twice absolute arc interpolation is shown as follows:



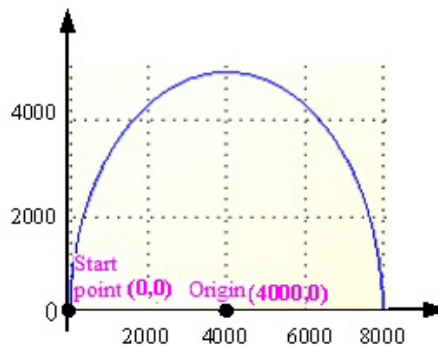
Method3:

Execute arc interpolation according to center point and angle

Users can input center point and angle to execute relative / absolute arc interpolation by calling `Acm_GpMoveArcRel_Angle/Acm_GpMoveArcAbs_Angle`.

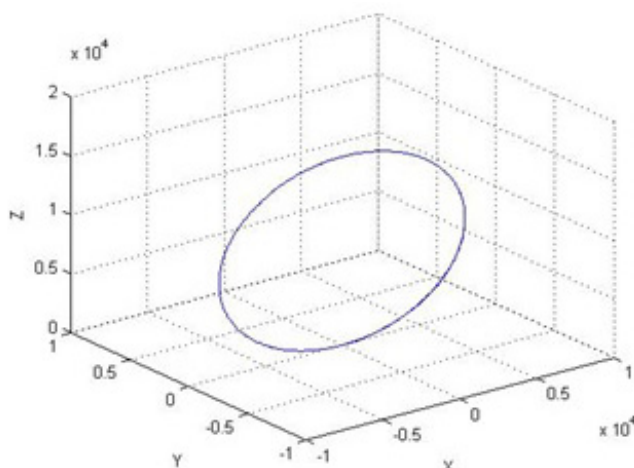
For example, execute relative/absolute arc interpolation by setting reference point (4000, 0) and angle=180°. The default start point is (0, 0).

The result of successively executing twice absolute arc interpolation is shown as follows:



2. 3-Axis Arc Interpolation

3-axis arc interpolation is operating in three dimensional spaces. Refer to the figure below; arc interpolation is executed from X, Y, Z axis in XYZ plane.



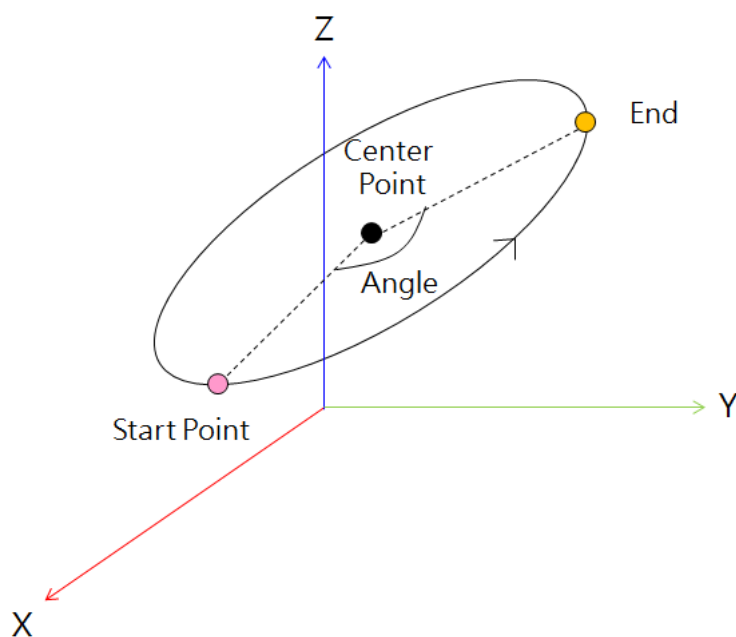
There are two ways to execute 3-axis arc interpolation:

Method1:

Execute 3D arc interpolation according to center point, end point and direction

Users can input center point, end point and direction to execute relative / absolute 3D arc interpolation by calling `Acm_GpMove3DArcRel/Acm_GpMove3DArcAbs`.

3D arc interpolation is shown as below:



There are limitations for 3D arc interpolation:

1. Semicircle with arc angle = 180° is not allowed.
2. Semicircle with arc angle = 360° is not allowed.

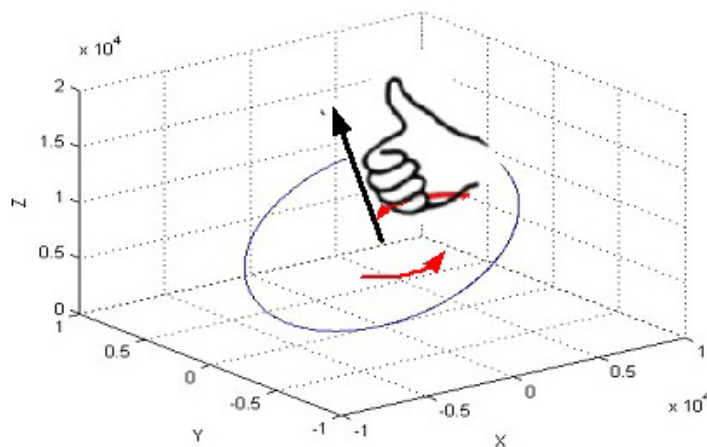
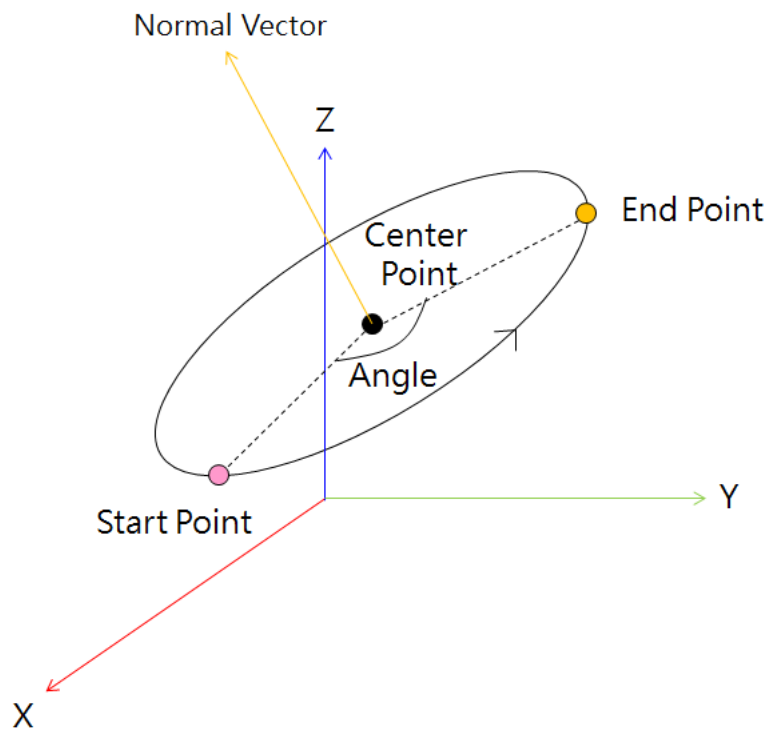
When start point, center point and end point position are in the line and can't decide the plane according to this straight line. In this situation, users can decide the plane by method 2.

Method2:

Execute 3D arc interpolation according to center point, normal vector, angle and direction

Users can input center point, normal vector, angle and direction to execute relative / absolute 3D arc interpolation by calling `Acm_GpMove3DArcRel_V/ Acm_GpMove3DArcAbs_V`. The rotation direction of 3D arc interpolation (CW or CCW) can be decided by the positive direction of normal vector:

- The direction of normal vector is positive If the included angle between the normal vector and Z axis is an acute angle ($<90^\circ$), otherwise, it is negative direction.
- If the normal vector is orthogonal to Z axis, then, the direction is positive if the included angle between the normal vector and Y axis is an acute angle ($<90^\circ$).
- If the normal vector is orthogonal to Z axis and Y axis, then, the direction is positive if the included angle between the normal vector and X axis is an acute angle ($<90^\circ$).



9.3.3.3 Flow Chart

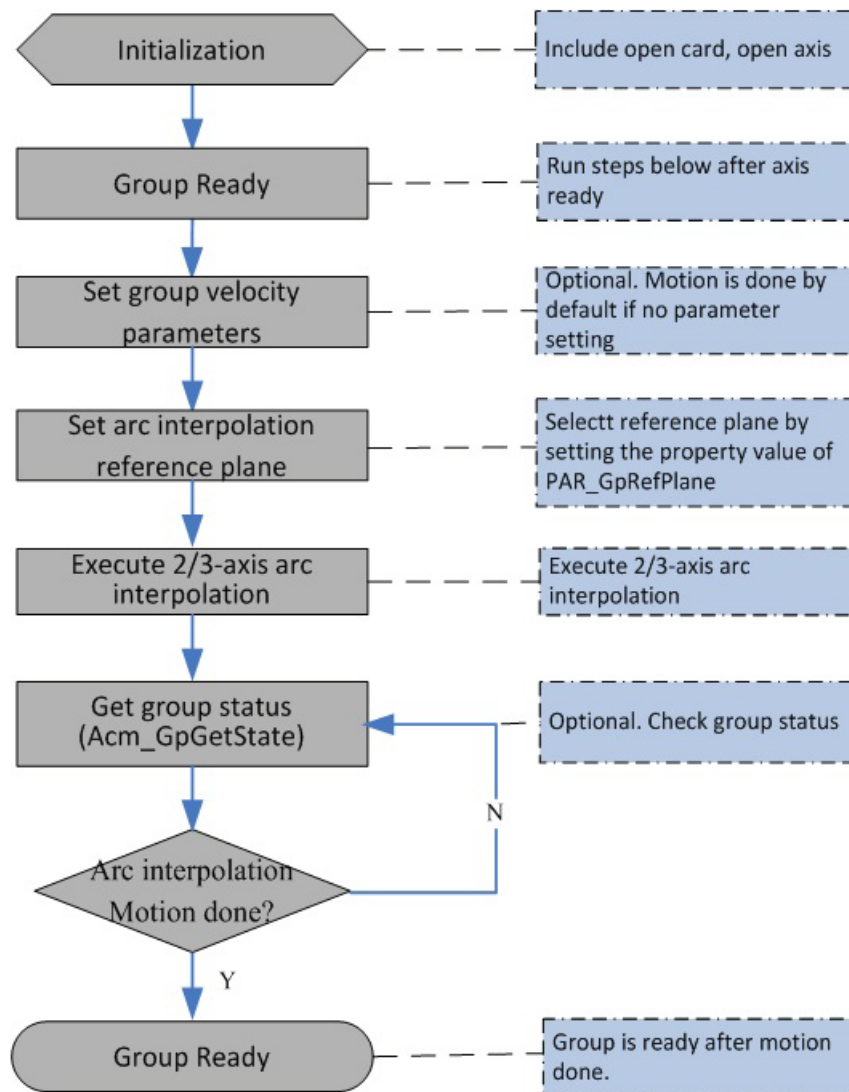


Figure 9.15 Arc interpolation motion flow chart

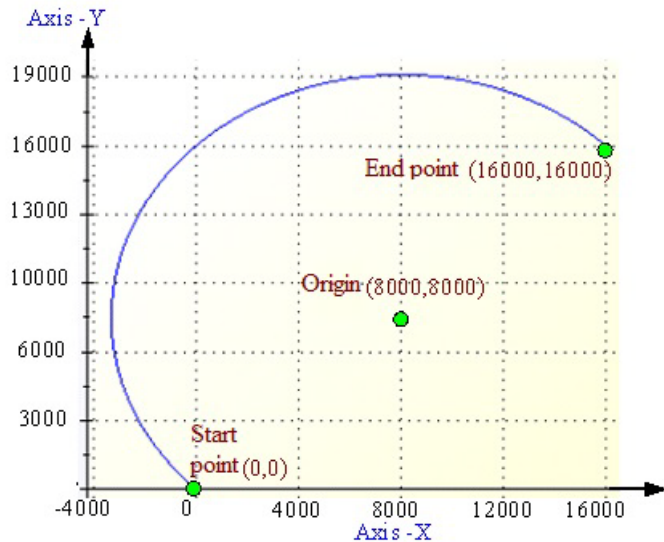
9.3.3.4 Example

This example executes Arc interpolation on XY plane. The center of circle is (8000, 8000), the end point is (16000, 16000), the direction is clockwise and the initial point is (0, 0).

Refer to the table below for setting parameters.

Function	Arc interpolation on XY plane(PCI-1203)
Initial velocity	2000PPU/S
Operating velocity	8000PPU/S
Acceleration	10000PPU/S ²
Deceleration	10000PPU/S ²
Velocity curve	T-Curve
Initial position	(0,0)
Center of circle	(8000,8000)
End point	(16000,16000)
Direction	Clockwise
Reference plane	XY plane

The result is shown below:



VC Code:

```

HAND m_GpHand;
U32 Ret;
U32 m_GpReferencePlane;
double CenterArray[2] ={8000,8000};
double EndArray[2] ={16000,16000};
U32 AxisNum;
I16 Dir;
--- Initialization---
//Set parameters
Ret =Acm_SetF64Property(m_GpHand,PAR_GpVelLow,2000);//Initial velocity
Ret =Acm_SetF64Property(m_GpHand,PAR_GpVelHigh,8000);//Operating velocity
Ret =Acm_SetF64Property(m_GpHand,PAR_GpAcc,10000);//Acceleration
Ret =Acm_SetF64Property(m_GpHand,PAR_GpDec,10000);//Deceleration
Ret =Acm_SetF64Property(m_GpHand,PAR_GpJerk,0);//T-Curve
//Set the reference plane as xy plane
m_GpReferencePlane =0;
Ret=Acm_SetU32Property(m_GpHand,PAR_GpRefPlane,m_GpReferencePlane);
//Arc interpolation
AxisNum =2;
Ret=Acm_GpMoveCircularRel(m_GpHand,CenterArray,EndArray,&AxisNum,0);
//Get group status
U16 GpState;
Acm_GpGetState(m_GpHand,&GpState); //Get group status

```

Check function returned value to judge the function execution status and establish error handle mechanism is recommended to ensure the correct result. Refer to the Arc example.

9.3.4 Helix Interpolation

9.3.4.1 Function and Property

Helical interpolation motion functions are shown in Table 9.23, the functions in table can be called directly in program.

Table 9.25: Helical interpolation motion functions

Function	Description
Acm_GpMoveHelixAbs	Command group to move absolute spiral.
Acm_GpMoveHelixRel	Command group to move relative spiral.
Acm_GpMoveHelixAbs_3P	Command group to move absolute spiral by three specified points.
Acm_GpMoveHelixRel_3P	Command group to move relative spiral by three specified points.
Acm_GpMoveHelixAbs_Angle	Complete helical interpolation through absolute center coordinates, angle, end point and direction of rotation.
Acm_GpMoveHelixRel_Angle	Complete helical interpolation through relative center coordinates, angle, end point and direction of rotation.
Acm_GpAddAxis	Add an axis to the specified group
Acm_GpRemAxis	Remove an axis from the specified group
Acm_GpClose	Remove all axis in the group and close the group handle
Acm_GpResetError	Reset group states
Acm_GpChangeVel	Command group to change the velocity while group is in line-interpolation motion
Acm_GpChangeVelByRate	Change the velocity of the current group motion according to the specified ratio
Acm_GpGetCmdVel	Get the current velocity of the group
Acm_GpStopDec	Command axis in this group to decelerate to stop
Acm_GpStopEmg	Command axis in this group to stop immediately without deceleration
Acm_AxSetCmdPosition	Set axis command position
Acm_AxGetCmdPosition	Get axis command position
Acm_AxGetActualPosition	Get axis actual (feedback) position
Acm_AxGetCmdVelocity	Get axis command velocity
Acm_GpGetState	Get states of axis
Acm_SetU32Property	Set property (Unsigned 32-bit integer)
Acm_SetI32Property	Set property (Signed 32-bit integer)
Acm_SetF64Property	Set property (Double)
Acm_GetU32Property	Get property (Unsigned 32-bit integer)
Acm_GetI32Property	Get property (Signed 32-bit integer)

Helical interpolation motion related properties as shown in Table 9.24, the property can't be called directly, but set and get property values by calling related property function. Refer to Appendix for more detail information.

Table 9.26: Helical interpolation motion properties

Parameter	Description
PAR_GpVelLow	Set/get initial velocity of group
PAR_GpVelHigh	Set/get operating velocity of group
PAR_GpAcc	Set/get acceleration of this group
PAR_GpDec	Set/get deceleration of this group
PAR_GpJerk	Set/get the type of velocity profile: T-Curve or S-Curve
PAR_GpGroupID	Get the GroupID through GroupHandle

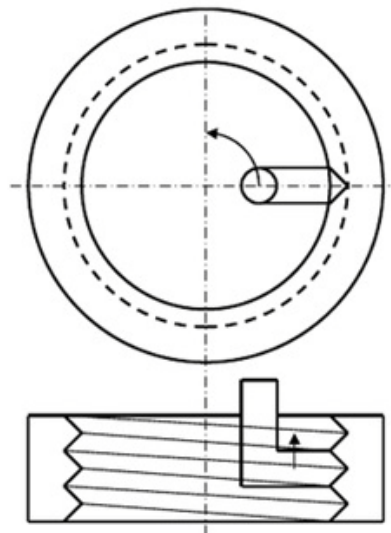
9.3.4.2 Description

Two-axis helical interpolation is specified in the XY, YZ, XZ plane to do a circular interpolation, and do linear interpolation for other axes synchronously to complete the synthesis of movement.

The implementation of the two-axis circular interpolation on which axis added to the group are ID-related. For example, the order of axis added to the group for interpolation is 3,4,2,1-axis, however, the axes which do the circular interpolation is according to the ID axis(0 axis of ID = 0 ...) and in this case, 1 and 2-axis for circular interpolation, and the axis 3, 4 will do the following motion.

In helical interpolation, we use X, Y as circular interpolation generally; Z axis is a straight line perpendicular to the XY plane. If you add a rotation axis C which perpendicular to the circle plane, making the tool and motion path at a fixed angle, and it will ensure that the tool is always perpendicular to the cutting surface.

The following figure shows a front view and a side view of helical interpolation.



When an error occurs in any one axis, all axes stop. The group speed can be set by the attributes of speed of the group. Helical interpolation applied to 3-axis or more than 3-axis linkage system.

There are several methods to do helical interpolation in PCI-1203:

Method 1: Perform helical interpolation according to center of circle, end point and direction.

Method 2: Perform helical interpolation according to center of circle, angle, end point and direction.

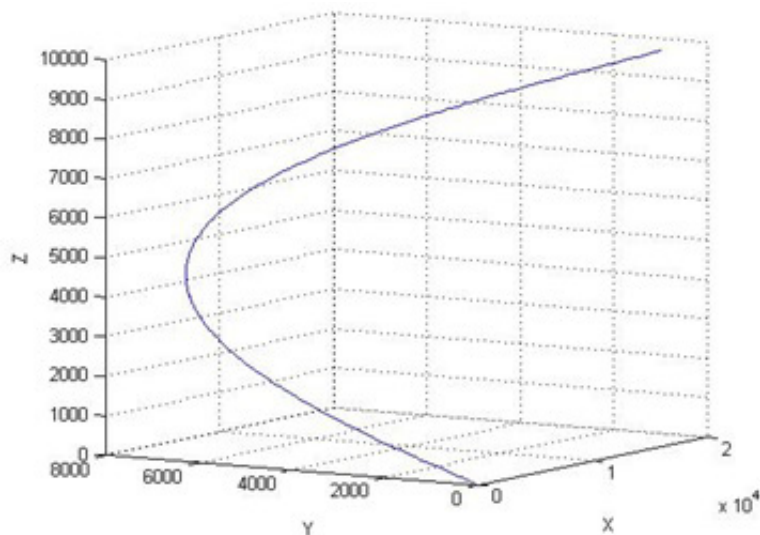
Method 3: Perform helical interpolation by three specified points.

The detail of these methods is shown introduced below:

Method 1:

Perform helical interpolation according to center of circle, angle, end point and direction.

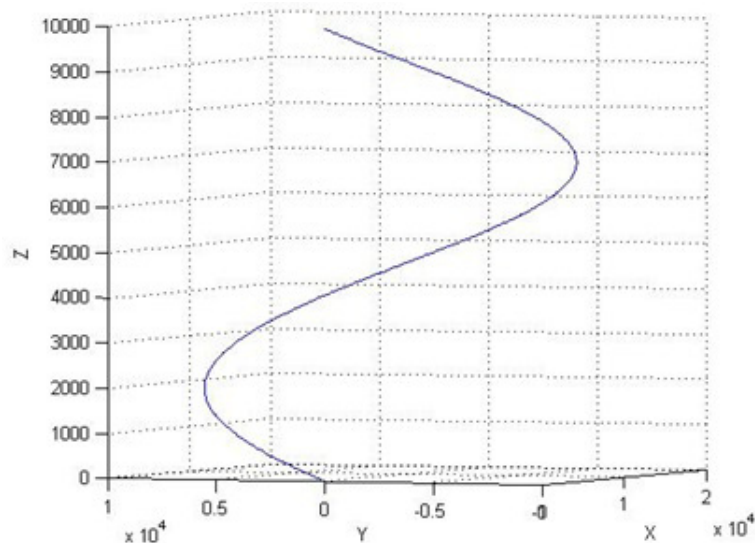
Use the API `Acm_GpMoveHelixRel/ Acm_GpMoveHelixAbs`, and the input parameters are center of circle, end point, the number of axes in group and the direction to perform helical interpolation. In the figure shown below, the center of circle is (8000,0,0), the end point is (16000,0,10000), the axis which do circular interpolation is X and Y axis, the following axis is Z axis.



Method 2:

Perform helical interpolation according to center of circle, angle and direction.

Use the API `Acm_GpMoveHelixRel_Angle/Acm_GpMoveHelixAbs_Angle`, and the input parameters are center of circle, rotation angle, end point, the number of axes in group and the direction to perform helical interpolation. In the figure shown below, the center of circle is (8000,0,0), the end point is (1080, 1080, 10000), where the number 1080 is the angle of rotation, the direction is clockwise, the axis which do circular interpolation is X and Y axis, the following axis is Z axis.



Method 3:

Perform helical interpolation by three specified points.

Use the API `Acm_GpMoveHelixRel_3P/ Acm_GpMoveHelixAbs_3P`, the input parameter are three specified point to perform helical interpolation.

9.3.4.3 Flow Chart

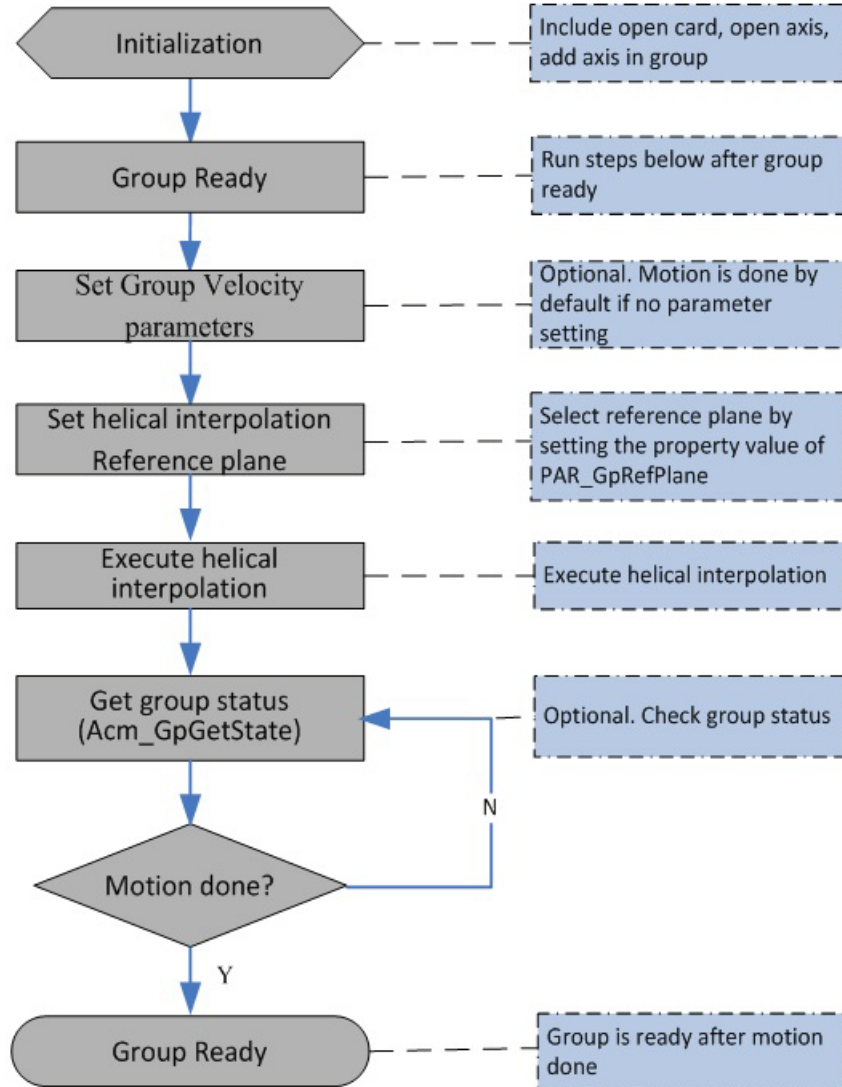


Figure 9.16 Helical interpolation motion flow chart

9.3.4.4 Example

This example executes helical interpolation on XY plane. Specify that the axis 0 and 1 perform circular interpolation and the axis 2 and 3 do point to point motion to follow the axis 0 and 1. The center of circle is (4000, 0), the end point is (8000, 8000), the direction is clockwise, the initial point is (0, 0) and moving distance of point to point motion is 10000. Refer to the table below for setting parameters.

Function	Helical interpolation on XY plane (PCI-1203)
Initial velocity	2000PPU/S
Operating velocity	8000PPU/S
Acceleration	10000PPU/S ²
Deceleration	10000PPU/S ²
Velocity curve	T-Curve
Initial position	(0,0)
Center of circle	(4000,0)
Direction	Clockwise
Reference plane	XY plane

VC Code:

```

HAND m_GpHand;
U32 Ret;
U32 m_GpReferencePlane;
double m_CenterArray[3] = {8000,0,0};
double m_EndArray[4] = {16000,0,10000,10000};
U32 AxisNum;
I16 Dir;
--- Initialization---
//Set parameters
Ret =Acm_SetF64Property(m_GpHand,PAR_GpVelLow,2000);//Initial velocity
Ret =Acm_SetF64Property(m_GpHand,PAR_GpVelHigh,8000);//Operating velocity
Ret =Acm_SetF64Property(m_GpHand,PAR_GpAcc,10000);//Acceleration
Ret =Acm_SetF64Property(m_GpHand,PAR_GpDec,10000);//Deceleration
Ret =Acm_SetF64Property(m_GpHand,PAR_GpJerk,0);//T-Curve
//Set the reference plane as xy plane
m_GpReferencePlane =0;
Dir =0;
Ret =Acm_SetU32Property(m_GpHand,PAR_GpRefPlane,m_GpReferencePlane);
//Helical interpolation
Ret=Acm_GpMoveHelixRel(m_GpHand,m_CenterArray,m_EndArray,&AxisNum,
Dir);
//Get group status
U16 GpState;
Acm_GpGetState(m_GpHand,&GpState); //Get group status

```

Check function returned value to judge the function execution status and establish error handle mechanism is recommended to ensure the correct result. Refer to the Helix example.

9.3.5 Group Event

9.3.5.1 Function and Property

Group event functions are shown in Table 9.25, the functions in table can be called directly in the program.

Table 9.27: Group event functions

Function	Description
Acm_EnableMotionEvent	Enable/disable axis and group event.
Acm_CheckMotionEvent	Check axis and groups enabled motion event status
Acm_GpMoveLinearRel	Command group to execute relative linear interpolation
Acm_GpMoveLinearAbs	Command group to execute absolute linear interpolation.
Acm_GpMoveDirectRel	Command group to execute relative direct linear interpolation
Acm_GpMoveDirectAbs	Command group to execute absolute direct linear interpolation
Acm_GpMoveCircularRel	Command group to execute relative arc interpolation.
Acm_GpMoveCircularAbs	Command group to execute absolute arc interpolation
Acm_GpMoveCircularRel_3P	Command group to execute relative arc interpolation by three specified points.
Acm_GpMoveCircularAbs_3P	Command group to execute absolute arc interpolation by three specified points
Acm_GpMove3DArcRel	Command group to execute relative 3D arc interpolation by specified points
Acm_GpMove3DArcAbs	Command group to execute absolute 3D arc interpolation by specified points
Acm_GpMove3DArcAbs_V	Command group to execute absolute 3D arc interpolation by normal vector
Acm_GpMove3DArcRel_V	Command group to execute relative 3D arc interpolation by normal vector

9.3.5.2 Description

Group events include Motion done, VHStart, VHEnd.

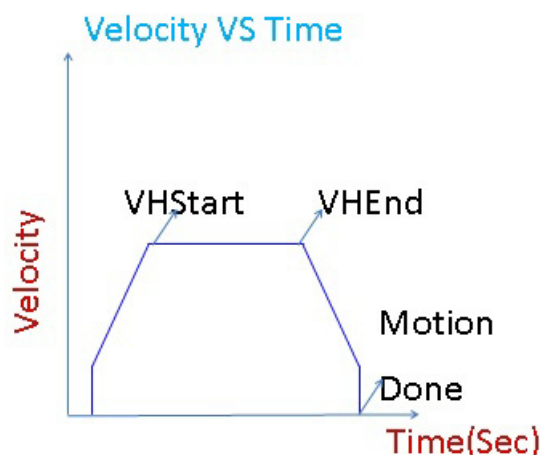
Motion done: Interrupt produced by hardware trigger when single-axis motion done.

VHStart: Interrupt produced by hardware trigger when single-axis velocity reaches the operating velocity.

VHSEnd: Interrupt produced by hardware trigger when single-axis velocity slows down.

Note! *VHStart and VHEnd means the start of operating velocity and the end of operating velocity, respectively.*





If you want to get event status of axis or groups, you should enable these events by calling `Acm_EnableMotionEvent` first. Set the parameter value of `GpEnableEvtArray` to enable / disable the interrupt event.

Parameter description:

`GpEnableEvtArray[N]`: Each element represents interrupt event for each group. This is a 32 bits data type array and the value of N can only be 0, 1, 2, represent `Gp_Motion_Done`, `Gp_VH_START`, `Gp_VH_END`, respectively.

Bitn: Each bit of array element. Set bit value of each element to enable / disable group event.

Bit n = 1: Enable Event

Bit n = 0: Disable Event

Where n is group ID.

For example, if there are three groups can be supported in PCI-1203, then the value of n is 0, 1, 2 and represent interrupt events of group 0, 1, 2, respectively. GroupID can be accessed by property `PAR_GpGroupID`.

	Bit n	Bit n...31 (n: groupID)
<code>GpEnableEvtArray[N]</code>		
<code>GpEnableEvtArray[0]</code>		<code>Evt_GpnMotion_DONE(n=0~31)</code>
<code>GpEnableEvtArray[1]</code>		<code>Evt_GpnVH_START(n=0~31)</code>
<code>GpEnableEvtArray[2]</code>		<code>Evt_GpnVH_END(n=0~31)</code>

Create a new thread to check event by calling `Acm_CheckMotionEvent`. The parameter `GpEvtStatusArray` will return event status of each group.

`GpEvtStatusArray [N]`: Each element represents interrupt event for each group. This is a 32 bits data type array and the value of N can only be 0, 1, 2, represent `Gp_Motion_Done`, `Gp_VH_START`, `Gp_VH_END`, respectively.

Bitn: Each bit of array element. Set bit value of each element to enable / disable group event.

Bit n = 1: Group Motion Done event occurred

Bit n = 0: Event not occurred or disabled.

Where n is group ID.

For example, if there are three groups can be supported in PCI-1203, then the value of n is 0, 1, 2 and represent interrupt events of group 0, 1, 2, respectively. GroupID can be accessed by property `PAR_GpGroupID`.

Bit n	Bit n...31 (n: groupID)
GpEvtStatusArrayN]	
GpEvtStatusArray [0]	Evt_GpnMotion_DONE(n=0~31)
GpEvtStatusArray [1]	Evt_GpnVH_START(n=0~31)
GpEvtStatusArray [2]	Evt_GpnVH_END(n=0~31)

9.3.5.3 Flow Chart

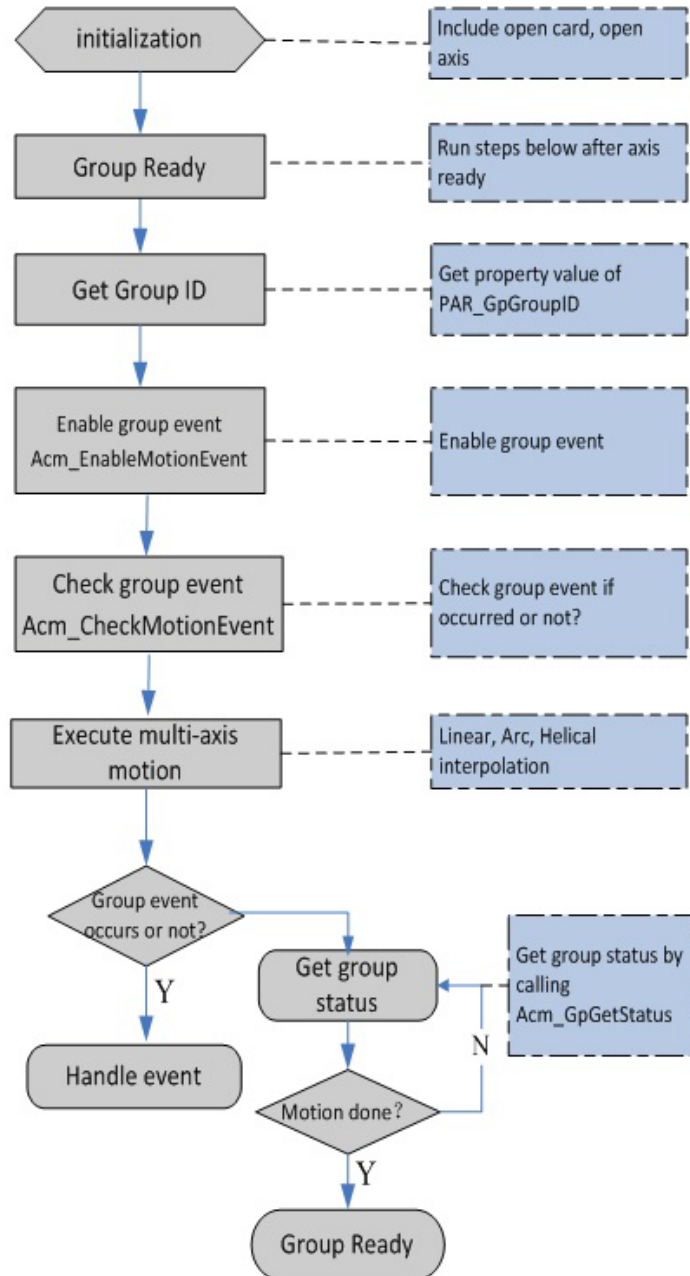


Figure 9.17 Group event flow chart

9.3.5.4 Example

This example shows how to monitor the group event. Refer to the table below for setting parameters.

Function	Monitor group event of Motion Done, VHStart, VHEnd
Motion type	Linear interpolation on two axes(PCI-1203)
Target position	(10000,10000)

VC Code:

```

HAND m_GpHand;
U32 PropertyVal =0;//Group ID
U32 Ret;
U32 AxEnableEvtArray[4];
U32 GpEnableEvt[3];
U32 m_ulAxisCount=4;
BOOL m_blnit ;
CWinThread* pThreadObject;//point of thread
--- Initialization---
Acm_GetU32Property(m_GpHand,PAR_GpGroupID,&PropertyVal); //Get group ID
GpEnableEvt[0]=EVT_GP1_MOTION_DONE << PropertyVal;//Enable motion done
GpEnableEvt[1]=EVT_GP1_VH_START<<PropertyVal;//Enable VHStart
GpEnableEvt[2] |= EVT_GP1_VH_END<<PropertyVal;//Enable VHEnd
Ret=Acm_EnableMotionEvent(m_Devhand,AxEnableEvtArray,GpEnableEvt,
m_ulAxisCount,3);//Enable event
//Open thread to monitor the group event
pThreadObject = AfxBeginThread( (AFX_THREADPROC)CDlg::CheckEvtThread,
this, THREAD_PRIORITY_TIME_CRITICAL, 0, 0, NULL);//Create the thread
UINT CDlg::CheckEvtThread(LPVOID ThreadArg)//Monitor function run in the
thread
{
    U32 Ret;
    U32 AxEvtStatusArray[4];
    U32 GpEvtStatusArray[3];
    CDlg*pThreadInfo;
    pThreadInfo = (CDlg*)ThreadArg;
    while (pThreadInfo->m_blnit)
    {
        Ret=Acm_CheckMotionEvent(pThreadInfo->m_Devhand,AxEvtStatusArray,
        GpEvtStatusArray,pThreadInfo->m_ulAxisCount,3,10000);//Monitor event
        status
        If(Ret == Success)
        {
            ...Determine the event occur or not and do following thing
        }
    }
}

```

Check function returned value to judge the function execution status and establish error handle mechanism is recommended to ensure the correct result. Refer to the Event example.

9.4 Following Motion

9.4.1 About this Section

Multi-axis can be connected together to achieve precise, synchronized motion through following motion mode. The axis which be followed is called the master axis, the others following axes are called the slave axes which operate according to the position of master axis.

Advantech motion control card provides interpolation modes are as follows:

- E-Gear
- E-CAM
- Gantry
- Tangential Following

9.4.2 E-Gear

9.4.2.1 Function and Property

E-Gearmotion functions are shown in Table 9.26, the functions in table can be called directly in program.

Table 9.28: E-Gear motion functions

Function	Description
Acm_AxGearInAx	Command E-Gear.
Acm_AxMoveRel	Command relative point-to-point motion
Acm_AxMoveAbs	Command absolute point-to-point motion
Acm_AxMoveVel	To command axis to make a never ending movement with a specified velocity
Acm_AxStopDec	Decelerated stop
Acm_AxStopEmg	Emergency stop (No deceleration)
Acm_AxStopDecEx	Command the axis to stop and specify the deceleration
Acm_AxGetState	Get states of axis
Acm_AxResetError	Reset error when axis is error-stop
Acm_AxSetCmdPosition	Set axis command position
Acm_AxGetCmdPosition	Get axis command position
Acm_AxGetActualPosition	Get axis actual (feedback) position
Acm_AxGetCmdVelocity	Get axis command velocity
Acm_SetU32Property	Set property (Unsigned 32 -bit integer)
Acm_SetI32Property	Set property (Signed 32 -bit integer)
Acm_SetF64Property	Set property (Double)
Acm_GetU32Property	Get property (Unsigned 32 -bit integer)
Acm_GetI32Property	Get property (Signed 32 -bit integer)
Acm_GetF64Property	Get property (Double)

E-Gear motion related properties as shown in Table 9.27 the property can't be called directly, but set and get property values by calling related property function. Refer to Appendix for more detailed information.

Table 9.29: E-Gear motion properties

Parameter	Description
PAR_AxVelLow	Set/get initial velocity of axis
PAR_AxVelHigh	Set/get operating velocity of axis
PAR_AxAcc	Set/get acceleration of this axis
PAR_AxDec	Set/get deceleration of this axis
PAR_AxJerk	Set/get the type of velocity profile: T-Curve or S-Curve
Configuration	Description
CFG_AxMaxVel	Configure the max velocity for the motion axis
CFG_AxMaxAcc	Configure the max acceleration for the motion axis
CFG_AxMaxDec	Configure the max deceleration for the motion axis

9.4.2.2 Description

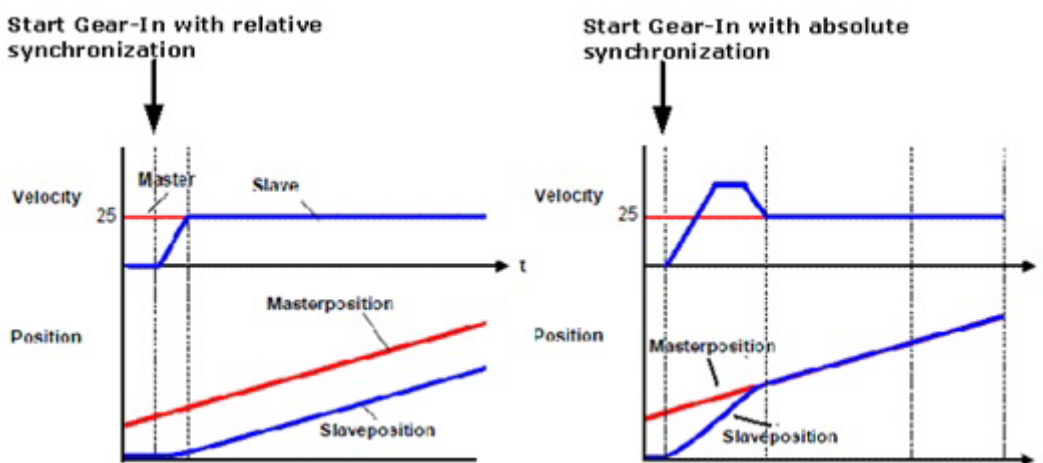
In the E-Gear mode, a master axis can drive multiple slave axes so that these axes can operate by following the planned position and encoder position of master axis.

The electronic gear ratio can be set by calling `Acm_AxGearInAx` and this function starts gear synchronization with a ratio between a slave (following) axis and master (leading) axis. Users can call this function several times according to the actual needs to set the synchronization relationship from slave axes and master axis.

Gear Ratio: Numerator/Denominator. If the value is positive, the slave will move at the same direction with master axis, or else it will move at the opponent direction with mater axis.

Absolute Relationship: Slave axis will compensate the offset with master axis.

Relative relationship: Slave axis will not compensate any offset with master axis. If the initial position is not equal to zero in this mode, the acceleration and deceleration of slave axes need to exceed 1000PPS.



9.4.2.3 Flow Chart

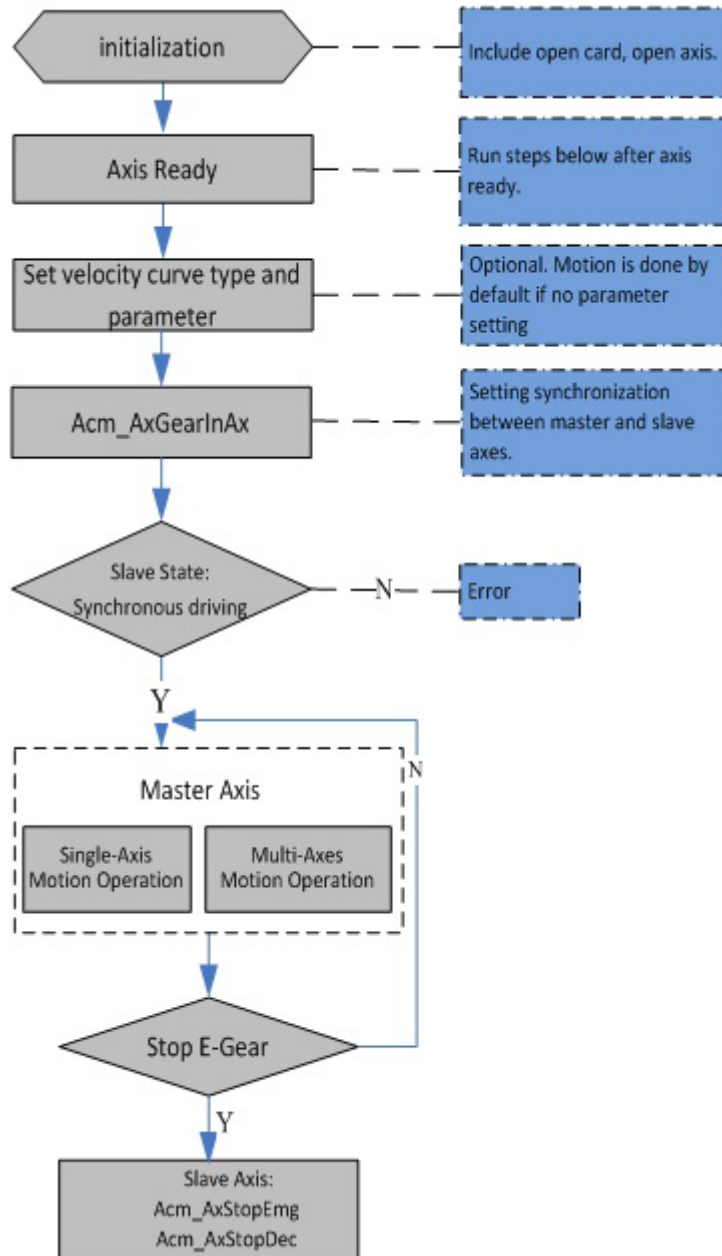


Figure 9.18 E-Gear flow chart

9.4.2.4 Example

This example executes E-Gear motion on 0 (master) and 1 (slave) axis. The gear ratio of master axis to slave axis is 10:1. Refer to the table below for setting parameters.

Function	Axis 1 follow axis 0 to do E-Gear motion
Initial velocity of master axis	2000PPU/S
Operating velocity of master axis	8000PPU/S
Acceleration of master axis	10000PPU/S ²
Deceleration of master axis	10000PPU/S ²
Velocity curve of master axis	T-Curve
Initial position of master axis	0
Target position of master axis	10000
Numerator of gear ratio	10
Denominator of gear ratio	1
Reference source	Slave axis engages to omm and position of master axis
Synchronization relationship	The synchronization is relative to start position (random position values upon reaching synchronization)

VC code:

```

HAND m_Axhand[32]; //Axis handle
U32  Ret; //
int   m_Num =10; //Gear ratio numerator
int   m_Den =1; //Gear ratio denominator
U32   RefSrc=0; //Slave axis engages to command position of master axis
U32   Mode=0; //Synchronization is relative to start position
---Initialization---
//Set parameters of main axis
Ret =Acm_SetF64Property(m_Axhand[0],PAR_AxVelLow,2000); //Initial velocity
Ret =Acm_SetF64Property(m_Axhand[0],PAR_AxVelHigh,8000); //Operating velocity
Ret =Acm_SetF64Property(m_Axhand[0],PAR_AxAcc,10000); //Acceleration
Ret =Acm_SetF64Property(m_Axhand[0],PAR_AxDec,10000); //Deceleration
Ret =Acm_SetF64Property(m_Axhand[0],PAR_AxJerk,0); //T-Curve
//Set gear ratio, reference source and synchronization relationship
Ret=Acm_AxGearInAx(m_Axhand[1],m_Axhand[0],m_Num,m_Den,RefSrc,Mode);
//Point to point motion of master axis
Ret =Acm_AxMoveRel(m_Axhand[0],10000); //Command relative point-to-point motion
//Get axis status and position
F64   CmdPos;
F64   ActPos;
U16   State;
Acm_AxGetCmdPosition(m_Axhand[0],&CmdPos); //Get 0 axis command position
Acm_AxGetActualPosition(m_Axhand[0],&ActPos); //Get 0 axis actual position
Acm_AxGetState(m_Axhand[0],&State); //Get 0 axis status

```

Check function returned value to judge the function execution status and establish error handle mechanism is recommended to ensure the correct result. Refer to the E-Gear example.

9.4.3 E-CAM

9.4.3.1 Function and Property

E-CAM functions are shown in Table 9.28, the functions in table can be called directly in program.

Table 9.30: E-CAM motion functions

Function	Description
Acm_DevDownloadCAMTable	Load data in CamTable
Acm_DevConfigCAMTable	Configure Cam
Acm_DevLoadCAMTableFile	Load CamTable file
Acm_AxCamInAx	Command E-CAM.
Acm_AxMoveRel	Command relative point-to-point motion
Acm_AxMoveAbs	Command absolute point-to-point motion
Acm_AxMoveVel	To command axis to make a never ending movement with a specified velocity
Acm_AxStopDec	Decelerated stop
Acm_AxStopEmg	Emergency stop (No deceleration)
Acm_AxStopDecEx	Command the axis to stop and specify the deceleration
Acm_AxGetState	Get states of axis
Acm_AxResetError	Reset error when axis is error-stop
Acm_AxSetCmdPosition	Set axis command position
Acm_AxGetCmdPosition	Get current command position of the specified axis
Acm_AxGetActualPosition	Get current actual position of the specified axis
Acm_AxGetCmdVelocity	Get current command velocity of the specified axis
Acm_SetU32Property	Set property (Unsigned 32 -bit integer)
Acm_SetI32Property	Set property (Signed 32 -bit integer)
Acm_SetF64Property	Set property (Double)
Acm_GetU32Property	Get property (Unsigned 32 -bit integer)
Acm_GetI32Property	Get property (Signed 32 -bit integer)
Acm_GetF64Property	Get property (Double)

E-CAM related properties as shown in Table 9.29, the property can't be called directly, but set and get property values by calling related property function. Refer to Appendix for more detail information.

Table 9.31: E-CAM motion properties

Parameter	Description
PAR_AxVelLow	Set/get initial velocity of axis
PAR_AxVelHigh	Set/get operating velocity of axis
PAR_AxAcc	Set/get acceleration of this axis
PAR_AxDec	Set/get deceleration of this axis
PAR_AxJerk	Set/get the type of velocity profile: T-Curve or S-Curve
Configuration	Description
CFG_AxCamDOAssign	Set/get the DO channel to output CamDO signal

Table 9.31: E-CAM motion properties

CFG_AxCamDOEnable	Set/get cam DO enable/disable
CFG_AxCamDOLoLimit	Set/get the low limit for CAMDO signal
CFG_AxCamDOHiLimit	Set/get the high limit for CAMDO signal
CFG_AxCamDOCmpSrc	Set/get the compare source
CFG_AxCamDOLogic	Set/get the active logic of CAMDO
CFG_AxModuleRange	Set/get pulse number when axis moves 360 degree
CFG_AxMaxVel	Configure the max velocity for the motion axis
CFG_AxMaxAcc	Configure the max acceleration for the motion axis
CFG_AxMaxDec	Configure the max deceleration for the motion axis

9.4.3.2 Description

The main function of a cam mechanism is used especially in transforming rotary motion into linear motion. It can make driven rod complete a variety of complex motion, including linear motion, swinging, constant movement and non-uniform motion according to the work required.

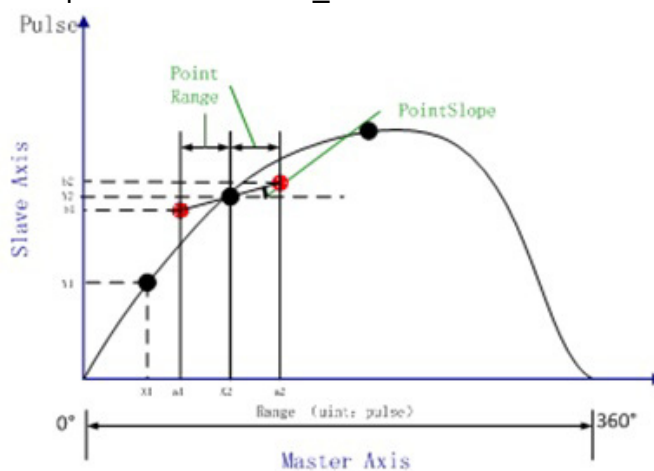
Camming is characterized by dynamic ratio between the leading and following axis, and by the phase shift. The transmission ratio is described by a CamTable. Camming is done with one table (two dimensional - describing master and slave positions together). The table should be strictly monotonic rising or falling, going both reverse and forward with the master.

There are two functions provided by Advantech Common Motion to load the CamTable:

`Acm_DevDownloadCAMTable`: This function downloads a CAM table profile which describes the ratio relationship of leading and following axis.

`Acm_DevLoadCAMTableFile`: Load Cam Table file edited and saved by Utility into device.

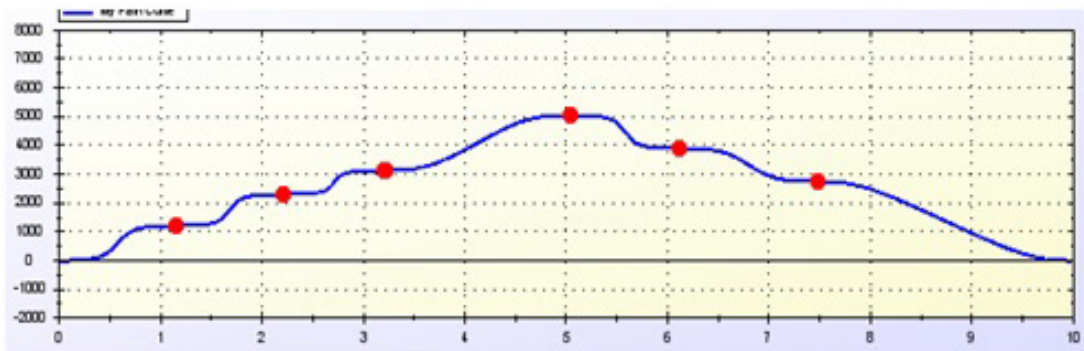
The meanings of the parameters of `Acm_DevDownloadCAMTable` are as follows:



As top-figure, Range is the needed pulse number when master axis rotates 360°. The black points in figure are composed of master values (eg. X1, X2) in MasterArray and corresponding slave values in SlaveArray, and the red points created by black points and assigned PointRange and PointSlope are called reference points. Cam curve is fitted by points in CamTable composed of MasterArray and SlaveArray. The horizontal axis is the pulse number when master axis moves at some angles. The vertical axis is the pulse number of slave axis when master moves at some angles.

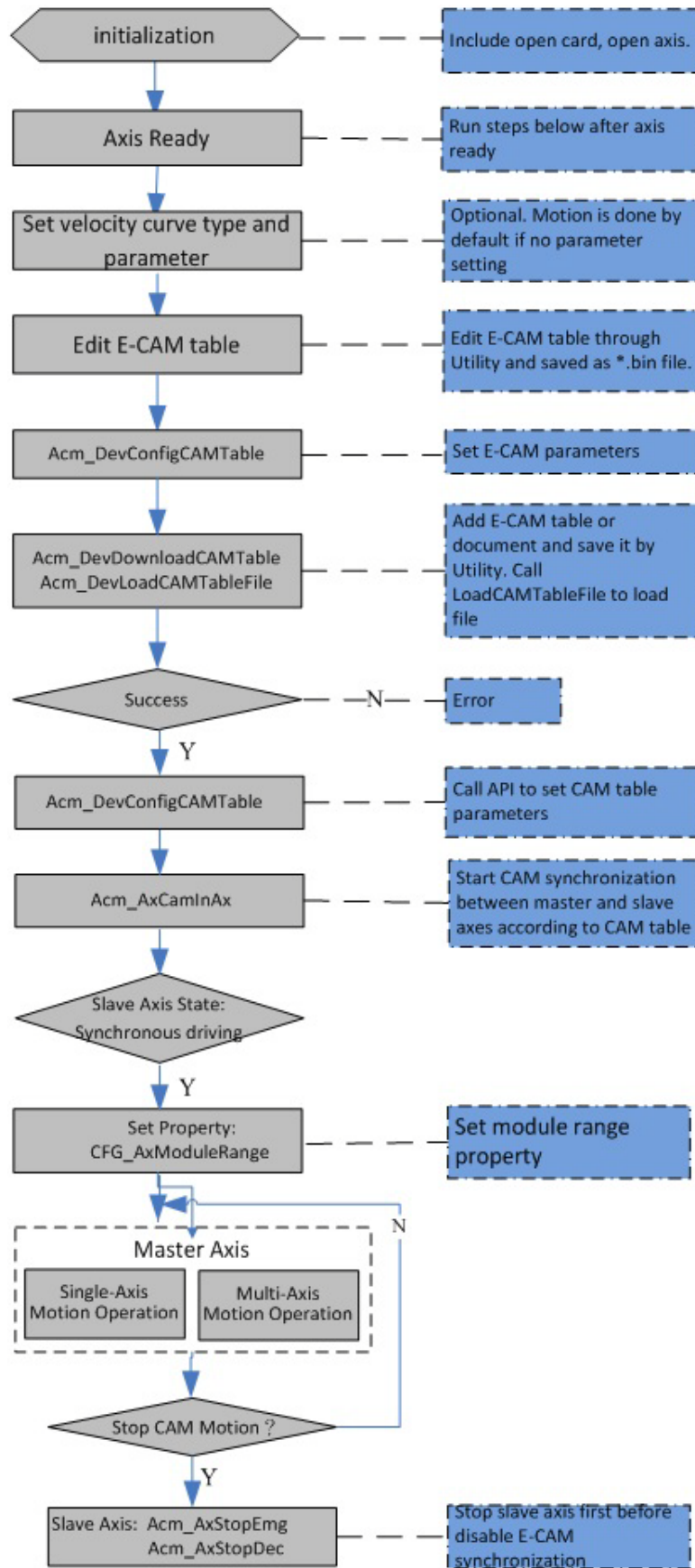
Range must be set into master axis by property `CFG_AxModuleRange`.

The CamTable can be edited by Common Motion Utility. In this figure, users can add the CAM points to form the curve of E-CAM, and saved the data in .bin format through utility. Then the CAM table can be loaded and operated by Acm_DevLoadCACMTableFile API.



The relevant parameters of the cam table can be set through Acm_DevConfigCAMTableAPI.

9.4.3.3 Flow Chart



9.4.3.4 Example

This example executes E-CAM motion on 0 (master) and 1 (slave) axis. The CAM table can be edited from utility.

Refer to the table below for setting parameters.

Function	Axis 1 follow axis 0 to do E-CAM motion
Initial velocity of master axis	2000PPU/S
Operating velocity of master axis	8000PPU/S
Acceleration of master axis	10000PPU/S ²
Deceleration of master axis	10000PPU/S ²
Velocity curve of master axis	T-Curve
Initial position of master axis	0
Target position of master axis	10000
Periodic	CAM curve is executed periodic
MasterAbsolute	Interpret CAM curve absolute to the master axis
SlaveAbsolute	Interpret CAMcurve relative to the slave axis.
Offset of master axis	0
Offsetof slave axis	0
Scaling factor of master axis	1
Scaling factor of slave axis	1
Reference source of CAMTable	Command position

VC code:

```
HAND m_Axhand[32]; //Axis handle
U32 Ret;
char *pstrString=newchar[100]; //Pointer to a string that saves CamTable file's path
int m_CurCamID; //Identifier of Cam table
U32 m_PointsCount = 0; //Points number in CamTable
U32 m_ModuleRange = 0; //Pulse number which master needed in one period
int m_CurCamType = 1; //CAM curve is executed periodic
U32 m_MasterAbsOrRel = 1; //Interpret cam curve absolute to the master axis
U32 m_SlaveAbsOrRel = 0; //Interpret cam curve relative to the slave axis
double m_MasterOffset=0; //Shifting the cam along the coordinates of the master axis
double m_SlaveOffset=0; //Shifting the cam along the coordinates of the slave axis
double m_MasterScale = 1.0; //Scaling factor for the cam in the coordinates of the master axis
double m_SlaveScale = 1.0; //Scaling factor of slave axis
int m_CmdOrFdb = 0; //Cam table's master position reference to command position
---Initialization---
//Set parameters of main axis
Ret =Acm_SetF64Property(m_Axhand[0],PAR_AxVelLow,2000); //Initial velocity
Ret =Acm_SetF64Property(m_Axhand[0],PAR_AxVelHigh,8000); //Operating velocity
Ret =Acm_SetF64Property(m_Axhand[0],PAR_AxAcc,10000); //Acceleration
Ret=Acm_SetF64Property(m_Axhand[0],PAR_AxDec,10000); //Deceleration
Ret =Acm_SetF64Property(m_Axhand[0],PAR_AxJerk,0); //T-Curve
```

```

//Load CAM Table file edited from utility
Ret=Acm_DevLoadCAMTableFile(m_Devhand,pstrString,m_CurCamID,
&m_ModuleRange,&m_PointsCount);
//Set parameters of CAM Table
Ret=Acm_DevConfigCAMTable(m_Devhand,m_CurCamID,m_CurCamType,
m_MasterAbsOrRel, m_SlaveAbsOrRel);
//Set synchronization relationship of E-CAM
Ret=Acm_AxCamInAx(m_Axhand[1],m_Axhand[0],m_MasterOffset,m_SlaveOffset,
m_MasterScale, m_SlaveScale, m_CurCamID, m_CmdOrFdb);
//Point to point motion of master axis
Ret=Acm_AxMoveRel(m_Axhand[0],10000);//Command relative point-to-point
motion
//Get axis status and position
F64 CmdPos;
F64 ActPos;
U16 State;
Acm_AxGetCmdPosition(m_Axhand[0],&CmdPos);//Get 0 axis command position
Acm_AxGetActualPosition(m_Axhand[0],&ActPos);//Get 0 axis actual position
Acm_AxGetState(m_Axhand[0],&State);//Get 0 axis status

```

Check function returned value to judge the function execution status and establish error handle mechanism is recommended to ensure the correct result. Refer to the ECAM example.

9.4.4 Gantry

9.4.4.1 Function and Property

Gantry functions are shown in Table 9.30, the functions in table can be called directly in program.

Table 9.32: Gantry motion functions

Function	Description
Acm_AxGantryInAx	Command gantry
Acm_AxMoveRel	Command relative point-to-point motion
Acm_AxMoveAbs	Command absolute point-to-point motion
Acm_AxMoveVel	To command axis to make a never ending movement with a specified velocity
Acm_AxStopDec	Decelerated stop
Acm_AxStopEmg	Emergency stop (No deceleration)
Acm_AxStopDecEx	Command the axis to stop and specify the deceleration
Acm_AxGetState	Get states of axis
Acm_AxResetError	Reset error when axis is error-stop
Acm_AxSetCmdPosition	Set axis command position
Acm_AxGetCmdPosition	Get current command position of the specified axis
Acm_AxGetActualPosition	Get current actual position of the specified axis
Acm_AxGetCmdVelocity	Get current command velocity of the specified axis
Acm_SetU32Property	Set property (Unsigned 32 -bit integer)
Acm_SetI32Property	Set property (Signed 32 -bit integer)
Acm_SetF64Property	Set property (Double)
Acm_GetU32Property	Get property (Unsigned 32 -bit integer)

Table 9.32: Gantry motion functions

Acm_GetI32Property	Get property (Signed 32 -bit integer)
Acm_GetF64Property	Get property (Double)

Gantry related properties as shown in Table 9.31, the property can't be called directly, but set and get property values by calling related property function. Refer to Appendix for more detail information.

Table 9.33: Gantry motion properties

Parameter	Description
PAR_AxVelLow	Set/get initial velocity of axis
PAR_AxVelHigh	Set/get operating velocity of axis
PAR_AxAcc	Set/get acceleration of this axis
PAR_AxDec	Set/get deceleration of this axis
PAR_AxJerk	Set/get the type of velocity profile: T-Curve or S-Curve
Configuration	Description
CFG_AxGantryMaxDiffValue	Set/get the value of following error protection
CFG_AxMaxVel	Configure the max velocity for the motion axis
CFG_AxMaxAcc	Configure the max acceleration for the motion axis
CFG_AxMaxDec	Configure the max deceleration for the motion axis

9.4.4.2 Description

In the Gantry mode, a master axis can drive multiple slave axes so that these axes can operate by following the planned position and encoder position of master axis.

The reference source and the direction with master axis can be set by calling Acm_AxGantryInAx and this function command two axes to move e-gantry motion.

There are some restrictions about gantry:

- Cannot set any command except Acm_AxStopDec /Acm_AxStopEmg to slave axis.
- Slave axis cannot be added in any group.
- If the axis is already one axis in group, it cannot be slave axis of gantry. If the command/actual position of master axis is reset, command/actual position of slave axis is reset same value too.

The following error protection function is provided in Gantry movement, when the difference value between the feedback position value of master and slave axis outside the maximum range of the following error window, the master and slave axis stops pulse output. This value can be set by CFG_AxGantryMaxDiffValue property.

9.4.4.3 Flow Chart

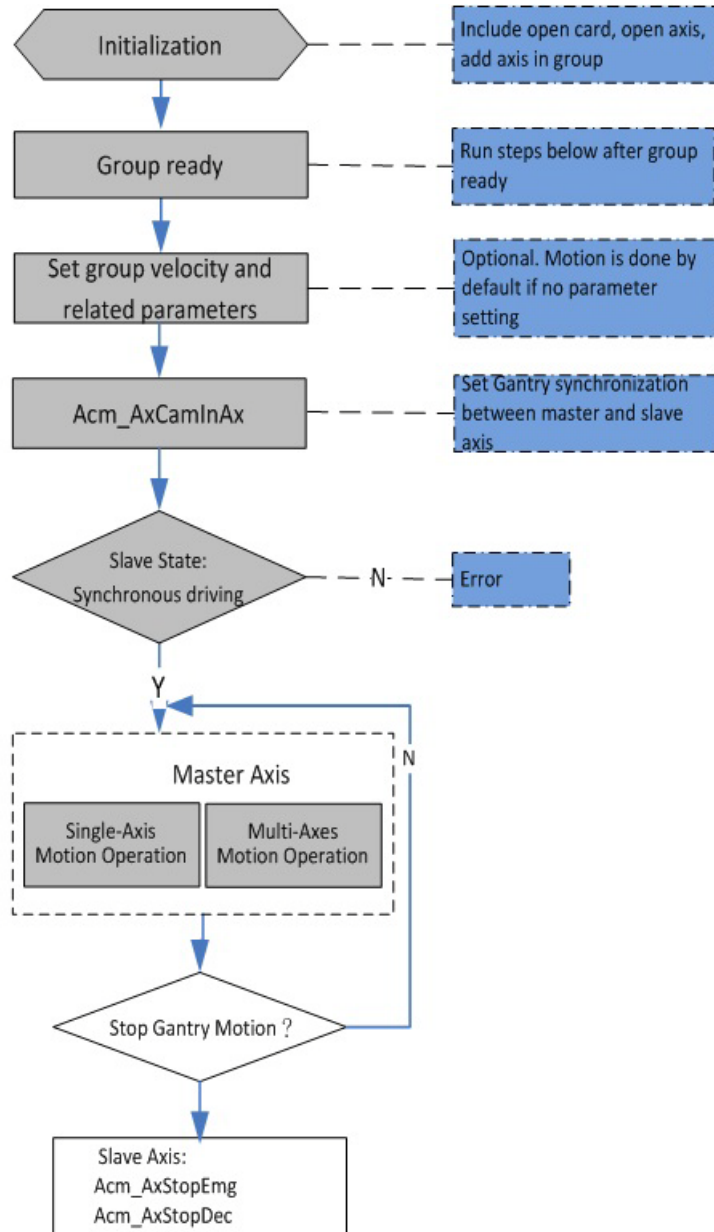


Figure 9.19 Gantry motion flow chart

9.4.4.4 Example

This example executes gantry motion on 0 (master) and 1 (slave) axis. Refer to the table below for setting parameters.

Function	Axis 1 follow axis 0 to do gantry motion
Initial velocity of master axis	2000PPU/S
Operating velocity of master axis	8000PPU/S
Acceleration of master axis	10000PPU/S ²
Deceleration of master axis	10000PPU/S ²
Velocity curve of master axis	T-Curve
Initial position of master axis	0
Target position of master axis	10000
Reference source	Slave axis engages to command position of master axis
Direction with master axis	same

VC code:

```
HAND m_Axhand[32]; //Axis handle
U32  Ret;
I16  RefSrc=0; //Reference source is command position
I16  Dir=0; //Same direction with master axis
---Initialization---
//Set parameters of main axis
Ret =Acm_SetF64Property(m_Axhand[0],PAR_AxVelLow,2000); //Initial velocity
Ret =Acm_SetF64Property(m_Axhand[0],PAR_AxVelHigh,8000); //Operating velocity
Ret =Acm_SetF64Property(m_Axhand[0],PAR_AxAcc,10000); //Acceleration
Ret =Acm_SetF64Property(m_Axhand[0],PAR_AxDec,10000); //Deceleration
Ret =Acm_SetF64Property(m_Axhand[0],PAR_AxJerk,0); //T-Curve
//Set synchronization relationship of gantry
Ret=Acm_AxGantryInAx(m_Axhand[1],m_Axhand[0],RefSrc,Dir);
//Point to point motion of master axis
Ret =Acm_AxMoveRel(m_Axhand[0],10000); //Command relative point-to-point motion
// Get axis status and position
F64  CmdPos;
F64  ActPos;
U16  State;
Acm_AxGetCmdPosition(m_Axhand[0],&CmdPos); //Get 0 axis command position
Acm_AxGetActualPosition(m_Axhand[0],&ActPos); //Get 0 axis actual position
Acm_AxGetState(m_Axhand[0],&State); //Get 0 axis status
```

Check function returned value to judge the function execution status and establish error handle mechanism is recommended to ensure the correct result. Refer to the Gantry example.

9.4.5 Tangential Following

9.4.5.1 Function and Property

Tangential following functions are shown in Table 9.32, the functions in table can be called directly in program.

Table 9.34: Tangential following motion functions

Function	Description
Acm_AxTangentInGp	Command tangent motion follow group.
Acm_GpMoveLinearRel	Command group to execute relative linear interpolation
Acm_GpMoveLinearAbs	Command group to execute absolute linear interpolation.
Acm_GpMoveCircularRel	Command relative arc interpolation.
Acm_GpMoveCircularAbs	Command absolute arc interpolation
Acm_GpMoveHelixRel	Command relative helix interpolation.
Acm_GpMoveHelixAbs	Command absolute helix interpolation.
Acm_AxStopDec	Decelerated stop
Acm_AxStopEmg	Emergency stop (No deceleration)
Acm_AxStopDecEx	Command the axis to stop and specify the deceleration
Acm_AxGetState	Get states of axis
Acm_AxResetError	Reset error when axis is error-stop
Acm_AxSetCmdPosition	Set axis command position
Acm_AxGetCmdPosition	Get current command position of the specified axis
Acm_AxGetActualPosition	Get current actual position of the specified axis
Acm_AxGetCmdVelocity	Get current command velocity of the specified axis
Acm_SetU32Property	Set property (Unsigned 32 -bit integer)
Acm_SetI32Property	Set property (Signed 32 -bit integer)
Acm_SetF64Property	Set property (Double)
Acm_GetU32Property	Get property (Unsigned 32 -bit integer)
Acm_GetI32Property	Get property (Signed 32 -bit integer)
Acm_GetF64Property	Get property (Double)

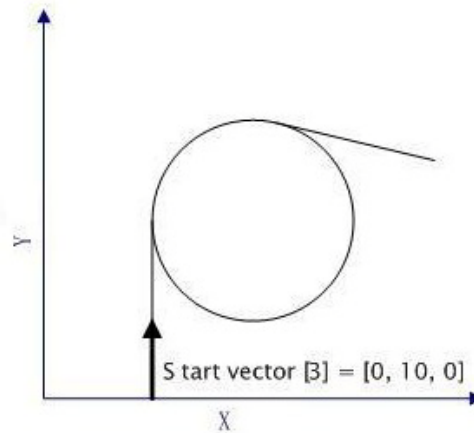
Tangential following related properties as shown in Table 9.33, the property can't be called directly, but set and get property values by calling related property function. Refer to Appendix for more information.

Table 9.35: Tangential following motion properties

Parameter	Description
PAR_AxVelLow	Set/get initial velocity of axis
PAR_AxVelHigh	Set/get operating velocity of axis
PAR_AxAcc	Set/get acceleration of this axis
PAR_AxDec	Set/get deceleration of this axis
PAR_AxJerk	Set/get the type of velocity profile: T-Curve or S-CurveT-Curve or S-Curve
Configuration	Description
CFG_AxMaxVel	Configure the max velocity for the motion axis
CFG_AxMaxAcc	Configure the max acceleration for the motion axis
CFG_AxMaxDec	Configure the max deceleration for the motion axis

9.4.5.2 Description

In cutting machine control system, it must be ensured that the cutting edge of tool and cutting direction of feed is consistent during the whole cutting process, that is, the tool always travel in the tangential direction of the curved profile and this form of machining is called tangential following motion. The continuous movement trajectory and rotational movement of the blade will be considered in the cutting process.



As shown in the figure, the group is doing clockwise arc interpolation motion and the direction of the arrow is the direction of the start vector. The following axis will move along the tangent direction of the interpolation motion if the tangent follow axis has established tangent follow synchronization with the Group.

9.4.5.3 Flow Chart

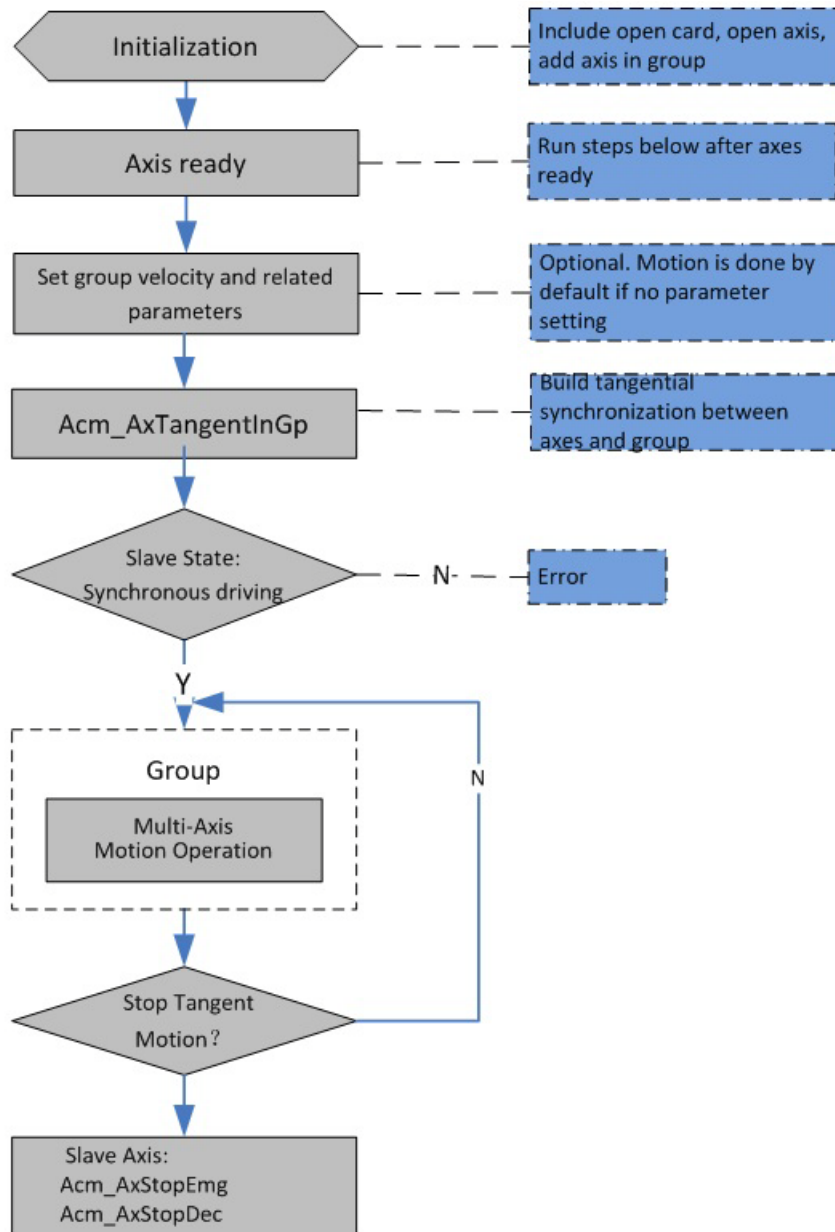


Figure 9.20 Tangential following flow chart

9.4.5.4 Example

This example executes tangential following motion. The axis 0, 1 do arc interpolation motion and the axis 2 follow the axis 0, 1 to do tangential following motion. Refer to the table below for setting parameters.

Function	Tangential following motion
Initial velocity of master axis	2000PPU/S
Operating velocity of master axis	8000PPU/S
Acceleration of master axis	10000PPU/S ²
Deceleration of master axis	10000PPU/S ²
Velocity curve of master axis	T-Curve
Initial position of master axis	0
Target position of master axis	10000
Reference source	Slave axis engages to command position of master axis
Direction with master axis	same

VC code:

```
HAND m_GpHand;//group handle
U32 Ret;
U32 m_GpReferencePlane; //Reference plane
double CenterArray[3] ={8000,8000};
double EndArray[3] ={16000,16000};
U32 AxisNum; //Axis number added to group
short startVector[3] ={0,10,0}; //Initial vector which is starting direction of axis
byte WorkingPlane=0; //Working plane is XY plane
int m_Dir =0; //Same direction with master axis
---Initialization---
//Set parameters
Ret=Acm_SetF64Property(m_GpHand,PAR_GpVelLow,2000); //Initial velocity
Ret =Acm_SetF64Property(m_GpHand,PAR_GpVelHigh,8000); //Operating velocity
Ret =Acm_SetF64Property(m_GpHand,PAR_GpAcc,10000); //Acceleration
Ret =Acm_SetF64Property(m_GpHand,PAR_GpDec,10000); //Deceleration
Ret =Acm_SetF64Property(m_GpHand,PAR_GpJerk,0); //T-Curve
//Set pulse number when axis moves 360 degree
Ret=Acm_SetU32Property(m_Axishand[2],CFG_AxModuleRange,3600);
//Set tangential following relationship of axis and group
Ret=Acm_AxTangentInGp(m_Axishand[2], m_GpHand, startVector, WorkingPlane,
m_Dir);
//Arc interpolation motion
AxisNum =2;
Ret=Acm_GpMoveCircularRel(m_GpHand,CenterArray,EndArray,&AxisNum,0);
//Get group status
U16 GpState;
Acm_GpGetState(m_GpHand,&GpState); //Get group status
Check function returned value to judge the function execution status and establish
error handle mechanism is recommended to ensure the correct result. Refer to the
Tangent example.
```

9.5 Path Table Motion

9.5.1 About this Section

Users can add at most 7,000 paths into the path table for motion in path table mode. In this motion mode, users can set different motion command such as line interpolation, arc interpolation, helix interpolation, group delay and digital output control in different velocity. It also supports the function of speed-forward, pause and resume during the motion process.

9.5.2 Path Table

9.5.2.1 Function and Property

Path table motion functions are shown in Table 9.34.

Table 9.36: Path table motion functions

Function	Description
Acm_GpAddPath	Add one path into system buffer
Acm_GpResetPath	Reset path system buffer
Acm_GpLoadPath	Load a path file
Acm_GpUnloadPath	Unload path
Acm_GpMovePath	Move path in system buffer
Acm_GpMoveAllPath	Start continuous interpolation motion (Path) for selected groups.
Acm_GpGetPathStatus	Get current path status
Acm_GpMoveSelPath	Move assigned range paths
Acm_GpGetPathIndexStatus	Get status of assigned index path
Acm_GpPauseMotion	Pause group movement
Acm_GpResumeMotion	Resume movement after pause
Acm_GpGetState	Get states of group
Acm_GpGetCmdVel	Get current velocity of the group
Acm_GpStopDec	Decelerated stop
Acm_GpStopEmg	Emergency stop
Acm_SetU32Property	Set property (Unsigned 32 -bit integer)
Acm_SetI32Property	Set property (Signed 32 -bit integer)
Acm_SetF64Property	Set property (Double)
Acm_GetU32Property	Get property (Unsigned 32 -bit integer)
Acm_GetI32Property	Get property (Signed 32 -bit integer)
Acm_GetF64Property	Get property (Double)

Path table motion related properties as shown in Table 9.35, the property can't be called directly, but set and get property values by calling related property function. Refer to the Appendix for more detail information.

Table 9.37: Path table motion properties

Parameter	Description
PAR_GpRefPlane	Set/get reference plane for helix motion and arc interpolation
PAR_GpVelLow	Set/get initial velocity of group
PAR_GpVelHigh	Set/get operating velocity of group
PAR_GpAcc	Set/get acceleration of this group
PAR_GpDec	Set/get deceleration of this group
PAR_GpJerk	Set/get the type of velocity profile: T-Curve or S-Curve
PAR_GpGroupID	Get group ID
Configuration	Description
CFG_GpAxesInGroup	Get axes in group
CFG_GpBldTime	Set/get blending time when add a path into system path buffer
CFG_GpSFEnable	Enable/Disable speed forward function

9.5.2.2 Load Path and Path Table Motion Mode

Advantech motion control card support two ways to add the path into path buffer:

Method 1: Add an interpolation path to system path buffer by Acm_GpAddPath API. This function can be called multiple times for loading multiple interpolation paths.

Method 2: Load path data from path file by Acm_GpLoadPath API. It can load up to 600 path data at one time.

Table 9.38: Path table motion mode


Mode	Description
Line interpolation	Relative/ Absolute line interpolation of two-axis
	Relative/ Absolute line interpolation of three-axis
Direct interpolation	It can be selected parts of or all axes in the group to do direct interpolation motion.
Arc interpolation	Relative/ Absolute arc interpolation of two-axis
	Relative/ Absolute arc interpolation of three-axis
Helix interpolation	Relative/ Absolute helical interpolation of three or more than three axes.
Delay command	The group will delay some time to move next path
DO control	Control the digital output during path table motion without affecting the overall speed of the continuous path

Table 9.39: Command in path table motion

Mode	Motion command		Description
Line interpolation	Abs2DLine	Rel2DLine	2 and 3-axis relative/ absolute line interpolation
	Abs3DLine	Rel3DLine	
Direct interpolation	Abs1DDirect	Rel1DDirect	Relative/ Absolute point-to-point motion. 2 to 8-axis relative/ absolute direct line interpolation
	Abs2DDirect	Rel2DDirect	
	Abs3DDirect	Rel3DDirect	
	Abs4DDirect	Rel4DDirect	
	Abs5DDirect	Rel5DDirect	
	Abs6DDirect	Rel6DDirect	
	Abs7DDirect	Rel7DDirect	
	Abs8DDirect	Rel8DDirect	
2-axis arc interpolation	Abs2DArcCW	Rel2DArcCW	2-axis relative/ absolute clockwise/ counterclockwise circular interpolation
	Abs2DArcCCW	Rel2DArcCCW	
	Abs2DArcCWAngle	Rel2DArcCWAngle	
	Abs2DArcCCWAngle	Rel2DArcCCWAngle	
	Abs2DArcCW_3P	Rel2DArcCW_3P	
3-axis arc interpolation	Abs3DArcCW	Rel3DArcCW	3-axis relative/ absolute clockwise/ counterclockwise circular interpolation
	Abs3DArcCW	Rel3DArcCW	
	Abs3DArcCWAngle	Rel3DArcCWAngle	
	Abs3DArcCCWAngle	Rel3DArcCCWAngle	

Table 9.39: Command in path table motion

Helix interpolation	Abs3DSpiralCW	Rel3DSpiralCW	3 to 8-axis relative/ absolute clockwise/ counterclockwise helix interpolation	
	Abs3DSpiralCCW	Rel3DSpiralCCW		
	Abs4DSpiralCW	Rel4DSpiralCW		
	Abs4DSpiralCCW	Rel4DSpiralCCW		
	Abs5DSpiralCW	Rel5DSpiralCW		
	Abs5DSpiralCCW	Rel5DSpiralCCW		
	Abs6DSpiralCW	Rel6DSpiralCW		
	Abs6DSpiralCCW	Rel6DSpiralCCW		
	Abs7DSpiralCW	Rel7DSpiralCW		
	Abs7DSpiralCCW	Rel7DSpiralCCW		
	Abs8DSpiralCW	Rel8DSpiralCW		
	Abs8DSpiralCCW	Rel8DSpiralCCW		
	Abs3DSpiralCWAngle	Rel3DSpiralCWAngle		3 to 8-axis relative/ absolute clockwise/ counterclockwise helix interpolation through absolute center coordinates, angle and direction of rotation.
	Abs3DSpiralCCWAngle	Rel3DSpiralCCWAngle		
	Abs4DSpiralCWAngle	Rel4DSpiralCWAngle		
	Abs4DSpiralCCWAngle	Rel4DSpiralCCWAngle		
	Abs5DSpiralCWAngle	Rel5DSpiralCWAngle		
	Abs5DSpiralCCWAngle	Rel5DSpiralCCWAngle		
	Abs6DSpiralCWAngle	Rel6DSpiralCWAngle		
	Abs6DSpiralCCWAngle	Rel6DSpiralCCWAngle		
Abs7DSpiralCWAngle	Rel7DSpiralCWAngle			
Abs7DSpiralCCWAngle	Rel7DSpiralCCWAngle			
Abs8DSpiralCWAngle	Rel8DSpiralCWAngle			
Abs8DSpiralCCWAngle	Rel8DSpiralCCWAngle			
Delay command	GPDELAY		Delay some time to move next path (uint:ms)	
DO Control	DoControl		Control digital output signal, refer to chapter 9.5.5 for advanced information	
EndPath	EndPath		At the last, the End-Path command must be add in path buffer.	

Note!  The absolute commands and relative commands cannot be mixed in system path buffer except GPDELAY, DoControl and EndPath, or else the error will be returned.

9.5.2.3 Load Path by Acm_GpAdd Path

Method 1: Add an interpolation path to system path buffer by Acm_GpAddPath API. Users can enable/disable speed-forward function by CFG_GpSFEnable and set blending time by CFG_GpBldTime before calling this function.

U32 Acm_GpAddPath (HAND GroupHandle, U16 MoveCmd, U16 MoveMode, F64 FH, F64 FL, PF64 EndPoint_DataArray, PF64 CenPoint_DataArray, PU32 ArrayElements)

Table 9.40: Parameter description of Acm_GpAddPath

Parameter	Description
GroupHandle	The group handle of every path in system buffer must be the same. So, if there are some unexecuted paths in system buffer and you want to add new path into it by call Acm_GpAddPath, the parameter GroupHandle must be the same with the first unexecuted path's group handle.
MoveCmd	The move command list in Table 9.37. If the command is GPDELAY, users can set the delay time by FH. The path with this command cannot be added when speed-forward function is enabled by CFG_GpSFEnable. The unit of delay time is ms.
MoveMode	Blending Mode and Non-Blending Mode. Refer to chapter 9.5.4
FH, FL	Driving velocity and start velocity of the added interpolation path. Users can set different driving and start velocity for each path. If the MoveCmd is GEDELAY, the parameter FH means delay time.
EndPoint_DataArray	End points (Unit: PPU of each axis). <ol style="list-style-type: none"> If the command is (Abs/Rel)2DArc(CW/CCW), it is rotation angle (Unit: degree). If the command is (Abs/Rel)3DArc(CW/CCW)Angle, EndPoint_DataArray[0~2] is normal vector and EndPoint_DataArray[3] is rotation angle (Unit: degree). If the command is (Abs/Rel)(3~8)DSpiral(CW/CCW)Angle, the rotation angle must be filled in EndPoint_DataArray[0~2] determined by reference plane which can be set through Par_GpRefPlane and the others parameter are end points. Take the example of method 2 in chapter 9.3.4.2, the reference plane is XY plane, then the EndPoint_DataArray[0] and EndPoint_DataArray[1] are rotation angle which the value is 1080 and the EndPoint_DataArray[2] is 10000 which is the end point of Z-axis.
CenPoint_DataArray	Center points (Unit: PPU of each axis)
ArrayElements	Number of array element of EndPoint_DataArray and CenPoint_DataArray, this number cannot be less than axis count in group, or else it will be returned axis count in group. Depending on MoveCmd and the axes count in the group, there are two circumstances: <ol style="list-style-type: none"> When the required number of axis in the motion command is less than or equal to the axes in the group, all paths can be loaded into the path buffer. For instance, there are 4 axes in the group and the command Rel2DLine, Rel3DLine and 4DDirect can be loaded into the device. When the required number of axis in the motion command is greater than to the axes in the group, the motion will be achieved in the axes with previous order in the group. If the motion command is 2-axis arc interpolation, users can select two axes of the previous three axes in the group by Par_GpRefPlane to implement that motion.

9.5.2.4 Load Path by Acm_GpLoadPath

Method 2: Load path data from path file by Acm_GpLoadPath API. It can load up to 600 path data at one time.

U32 Acm_GpLoadPath(Hand GroupHandle, P18 FilePath, PHAND PathHandle, PU32 pTotalCount)

The second parameter of this API point to a file path name of the motion path data which needs to be loaded. The path data file (binary) is usually generated by Motion Utility's [PathEditor]. If you are familiar with Advantech motion product, you can create file by yourself. The PathHandle must be unloaded by Acm_GpUnloadPath when the PathHandle does not be used any more or application is closing, and the paths contained in PathHandle are deleted from driver at the same time.

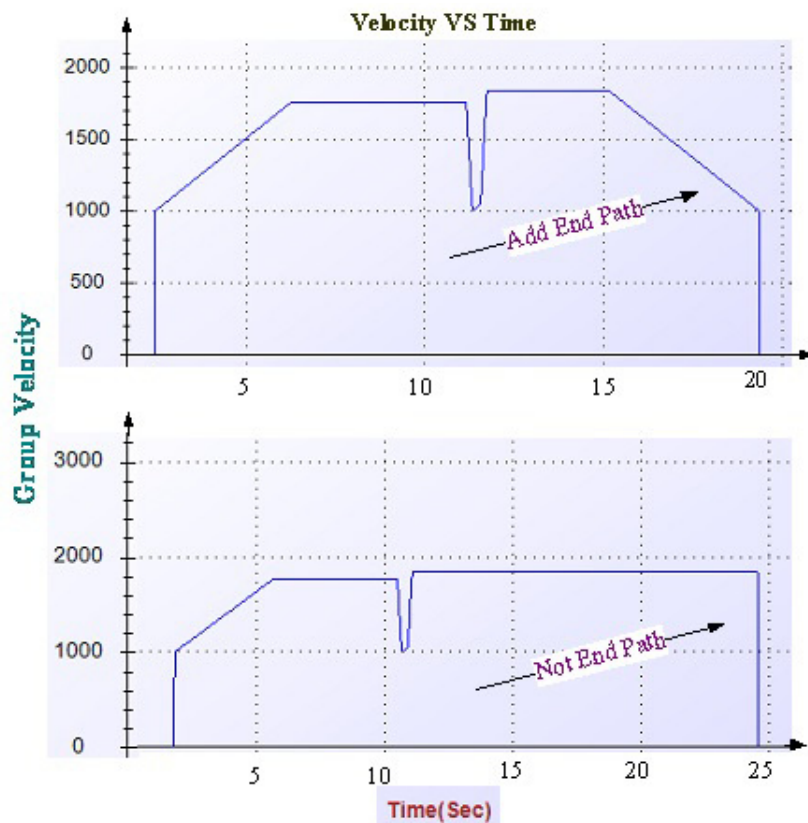
9.5.3 Effect of EndPath

When users load the path by the two methods introduced in chapter 9.5.2.2, the EndPath command must be add in path buffer, otherwise the deceleration of the path will be affected.

In the following table, the acceleration and deceleration is set to 500, enable motion blending and the blending time is 200ms.

ID	MoveCmd	Move-Mode	VelHigh	VelLow	Center0	Center1	EndPoint0	EndPoint1
0	Rel2DLine	Enable Blending	2000	1000	0	0	10000	10000
1	Rel2DArcCW	Enable Blending	2000	1000	4000	0	8000	0
2	EndPath	Enable Blending	1000	400	0	0	0	0

The difference between the EndPath is added to the path table or not is shown in following figure:



The movement velocity of the last segment can be recalculated when the EndPath command is added to the path table; otherwise, the velocity command would be terminated without deceleration.

9.5.4 Move Mode

Move mode means that the velocity blending pattern of each segment when it is running in path table motion. When loading the path, users can set the different parameters of move mode, speed forward (CFG_GpSFEnable) and Blending time (CFG_GpBldTime) to achieve different pattern of velocity blending, the relational table is shown below:

Table 9.41: Relational tables of move mode			
	Buffer mode	Blending mode	Fly mode
Speed forward (CFG_GpSFEnable)	CFG_GpSFEnable should be disabled	CFG_GpSFEnable should be disabled	When speed forward function is enabled by CFG_SFEnable. (T-Curve only)
Blending time (CFG_GpBldTime)	Do not set blending-time	BlendingTime>0	BlendingTime = 0
Move mode (MoveMode)	Non-Blending (MoveMode =1)	Blending (MoveMode =0)	Blending (MoveMode =0)

There will be two move modes when MoveMode enable, they are Blending Mode and FlyMode. If the value of CFG_GpBldTime is greater than zero, the path move in Blending Mode; if the value of CFG_GpBldTime is equal to zero and the type of velocity profile is T-Curve, the path move in Fly Mode.

9.5.4.1 BufferMode

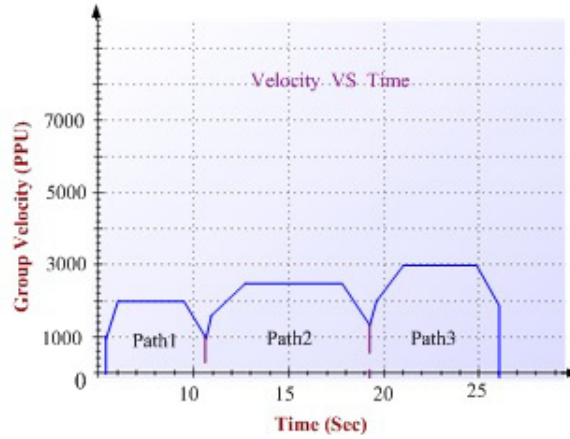
MoveMode	CFG_GpSFEnable	CFG_GpBldTime
1	Invalid	Invalid

When the MoveMode is Non-Blending, the path table will run in this mode. Each path has the whole process of accelerating and decelerating. In this mode, the Speed Foward function cannot be supported, so CFG_GpSFEnable should be disabled.

The path table (can be edited by utility) is shown below, the acceleration and deceleration is 2000.

ID	MoveCmd	MoveMode	VelHigh	VelLow	Center 0	Center 1	Center 2	EndPoint 0	EndPoint 1	EndPoint 2
0	Rel2DLine	Disable Blending	2000	1000	0	0	0	12000	12000	0
1	Rel3DHelixCW	Disable Blending	2500	1500	10000	0	0	20000	0	10000
2	Rel2DArcCW	Disable Blending	3000	2000	10000	0	0	20000	0	0
3	EndPath	Disable Blending	1000	500	0	0	0	0	0	0

And the velocity diagram after calling this API is:



9.5.4.2 Blending Mode

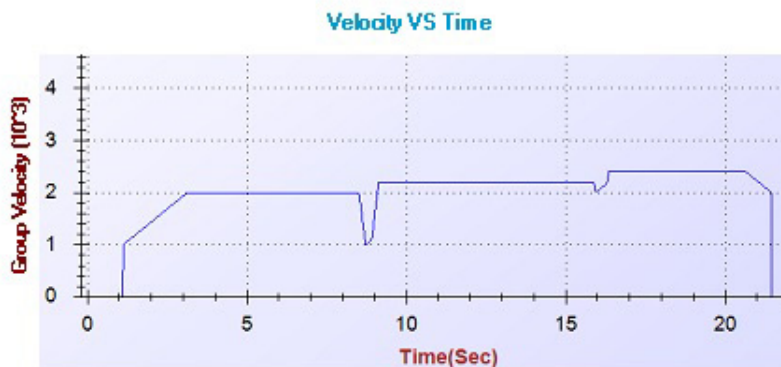
MoveMode	CFG_GpSFEnable	CFG_GpBlidTime
0	Disable	> 0

When the MoveMode is Blending and the value of CFG_GpBlidTime is greater than zero, the path table will run in this mode. Each path has the whole process of accelerating and decelerating. In this mode, the initial speed and final speed will be recalculated each segment of path according to the blending time.

The path table (can be edited by utility) is shown below, the acceleration, deceleration is 500 and the blending time is 400ms.

ID	MoveCmd	MoveMode	VelHigh	VelLow	Center 0	Center 1	Center 2	EndPoint 0	EndPoint 1	EndPoint 2
0	Rel2DLine	Enable Blending	2000	1000	0	0	0	10000	10000	0
1	Rel3DHelixCW	Enable Blending	2500	1500	5000	0	0	10000	0	10000
2	Rel2DArcCW	Enable Blending	3000	2000	4000	0	0	8000	0	0
3	EndPath	Enable Blending	1000	500	0	0	0	0	0	0

And the velocity diagram after calling this API is:



9.5.4.3 FlyMode

MoveMode	CFG_GpSFEnable	CFG_GpBldTime
0	Disable	= 0

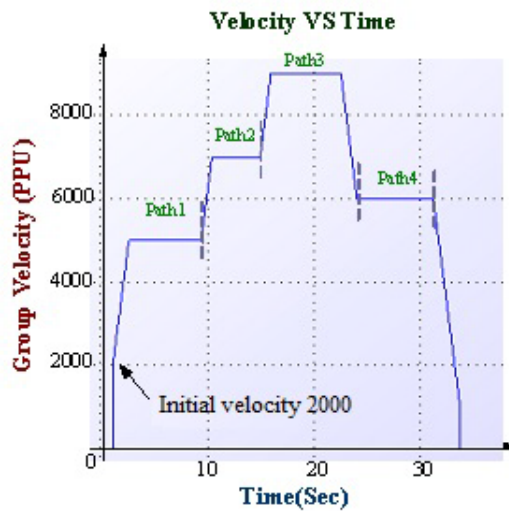
When the MoveMode is Blending and the value of CFG_GpBldTime is equal to zero, the path table will run in this mode. The speed forward function can be enabled during this mode.

1. Disable speed forward function

The path table (can be edited by utility) is shown below, the acceleration, deceleration is 2000 and the blending time is 0ms.

ID	MoveCmd	MoveMode	VelHigh	VelLow	Center 0	Center 1	Center 2	EndPoint 0	EndPoint 1	EndPoint 2
0	Rel2DLine	Enable Blending	5000	2000	0	0	0	30000	25000	0
1	Rel2DLine	Enable Blending	7000	4000	0	0	0	25000	28000	0
2	Rel2DArcCW	Enable Blending	9000	3000	10000	10000	0	20000	0	0
3	Rel2DArcCWAngel	Enable Blending	6000	1000	20000	0	0	180	180	0
4	EndPath	Enable Blending	2000	500	0	0	0	0	0	0

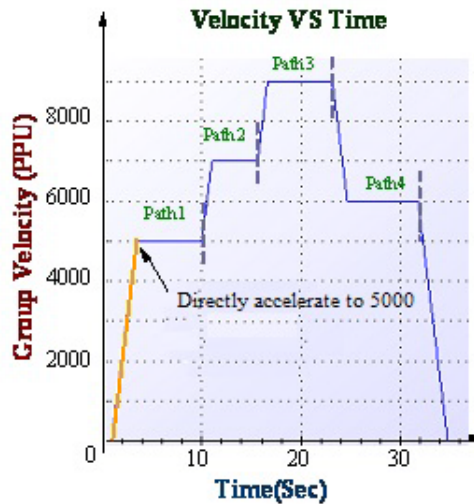
Save the path from utility and load the file through Acm_GpLoadPath API, after that, it can be run by calling Acm_GpMovePath.



And we modify the initial velocity:

ID	MoveCmd	MoveMode	VelHigh	VelLow	Center 0	Center 1	Center 2	EndPoint 0	EndPoint 1	EndPoint 2
0	Rel2DLine	Enable Blending	5000	0	0	0	0	30000	25000	0
1	Rel2DLine	Enable Blending	7000	0	0	0	0	25000	28000	0
2	Rel2DArcCW	Enable Blending	9000	0	10000	10000	0	20000	0	0
3	Rel2DArcCWAngel	Enable Blending	6000	0	20000	0	0	180	180	0
4	EndPath	Enable Blending	2000	500	0	0	0	0	0	0

The velocity diagram is shown below:



The figure shows that in addition to the first path segment, other segments at the same rate of change in velocity.

2. Enable speed forward function

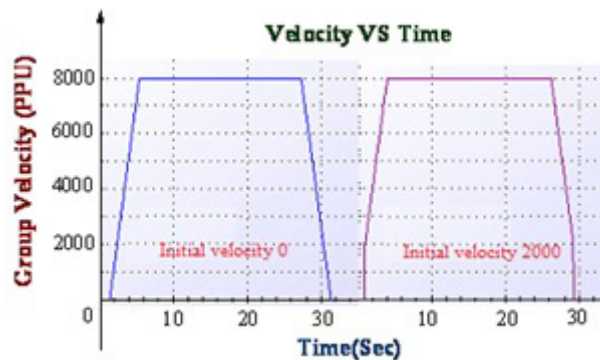
Speed forward means that the driving velocity is calculated according to the moving distance.

Instead of use the velocity parameter of Acm_GpAddPath when the speed forward function enabled, the motion algorithm only use the velocity parameter of group which set by property to run the path table.

The path table (can be edited by utility) is shown below. The parameter PAR_GpVelHigh (driving velocity), PAR_GpAcc (acceleration of group) and PAR_GpDec (deceleration of group) are 8000, 2000 and 2000, and enable speed forward function. Initial velocities of group are 0 and 2000, respectively.

ID	MoveCmd	MoveMode	VelHigh	VelLow	Center 0	Center 1	Center 2	EndPoint 0	EndPoint 1	EndPoint 2
0	Rel2DLine	Disable Blending	2000	1000	0	0	0	12000	12000	0
1	Rel3DHelixCW	Disable Blending	2500	1500	10000	0	0	20000	0	10000
2	Rel2DArcCW	Disable Blending	3000	2000	10000	0	0	20000	0	0
3	EndPath	Disable Blending	1000	500	0	0	0	0	0	0

The velocity diagram is shown below:



9.5.5 Path DO function

9.5.5.1 Description of Path DO

It is needed to use digital output function in the path table motion in some case. Take a dispensing system as an example, when you need to dispense items on a specific path, the user does not need to detect whether the dispenser reach the path or not, just set the DO output to control the dispensing tool to complete the dispensing process.

In the PCI-1203, users can assign the channel of digital output in the network to path buffer to control the digital output signal and it will not affect the overall speed of the continuous path during path table motion.

9.5.5.2 Implement of Path DO

Use the API `Acm_GpAddPath` to set the path DO, the parameters of API are shown below:

Parameter	Description
GroupHandle	Group handle
MoveCmd	Motion command: DoControl
MoveMode	Move mode
FH	The port ID of DO in the network. The port ID can be known in utility.
FL	Always 0
EndPoint_DataArray	Output signal of DO port, only <code>EndPoint_DataArray[0]</code> is valid. Refer to the following table for advanced setting.
CenPoint_DataArray	Always Null
ArrayElements	Number of array element can not be less than axis count in group, or else it will be returned axis count in group.

Bit	EndPoint_DataArray			
	24~31	16~23	8~15	0~7
Port ID of Do need to be set through FH parameter	Reserved	Set DO0~DO7 of this port whether each channel of this port can be controlled or not 0: Disable 1: Enable	Reserved	Enables/disables DO0~DO7 of specific port Enables/disables 0: Disable 1: Enable


9.5.6 Start PathTable Motion

After load the path through the two methods in 9.5.2.2, users can call the API `Acm_GpMovePath` to start the path table motion. The path table status can be known by `Acm_GpGetPathStatus`.

Users can also move path segment in system path buffer from start index and end index by calling `Acm_GpMoveSelPath`.

Use `Acm_GpGetPathIndexStatus` to get the status of specified index path in system path buffer and clear system path buffer through `Acm_GpResetPath` function. If there is group executing path while `Acm_GpResetPath` is called, the path motion will be stopped. For advanced information of each API, refer to the Appendix.

In PCI-1203, you can start multiple groups to do path table motion simultaneously by `Acm_GpMoveAllPath` API. After load the path to each group, users can call this function and all paths in the specific group will run simultaneously.

Note!  Due to the path had been loaded into the system by `Acm_GpLoadPath`, the second parameter of `Acm_GpMovePath` API `PathHandle` might set to `NULL`, otherwise the path will be loaded twice time.

9.5.7 Pause and Resume Path Table Motion

When the Group in the process of movement, it issues a pause command, the board after the receipt of the command will decelerate stopped. It performs the restore command, continuing from where it was before the pause.

Function	Description
<code>Acm_GpPauseMotion</code>	Pause group movement
<code>Acm_GpResumeMotion</code>	Resume movement after pause

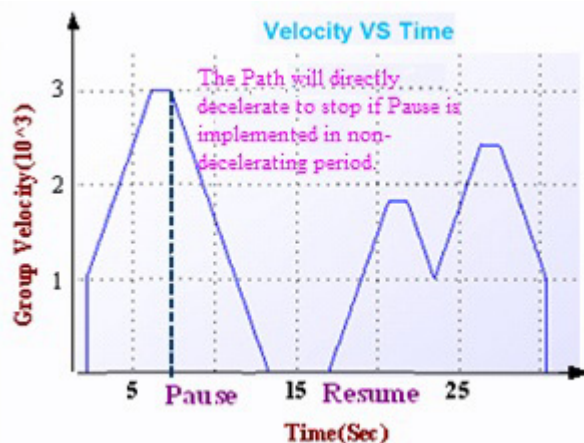
9.5.7.1 Pause / Resume in BufferMode

In `BufferMode`, each segment has its own accelerated and decelerated period. The current segment will decelerate to stop after receiving `Pause` command. Hardware will calculate velocity the rest of pulses can support to precede the rest path.

The path table (can be edited by utility) is shown below, the acceleration, deceleration is 500.

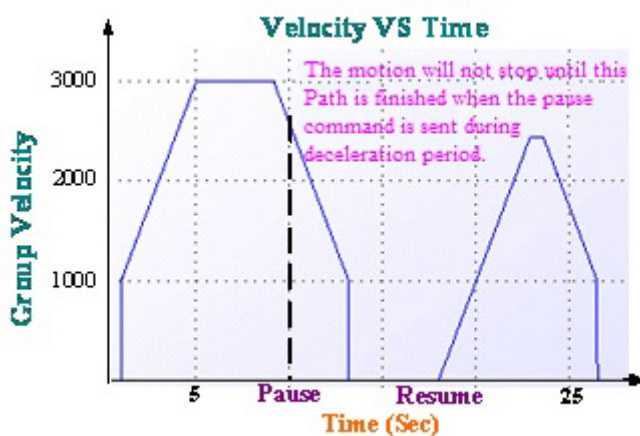
ID	MoveCmd	MoveMode	VelHigh	VelLow	Center0	Center1	EndPoint0	EndPoint1
0	Rel2DLine	Disable Blending	3000	1000	0	0	20000	20000
1	Rel2DArcCW	Disable Blending	3000	1000	4000	0	8000	0
2	EndPath	Disable Blending	1000	400	0	0	0	0

The velocity diagram is shown in the following figure:



If pause command is issued when the path is in the deceleration segment, this pause command will work after the path segment is completed.

The velocity diagram in this case is shown in the following figure:



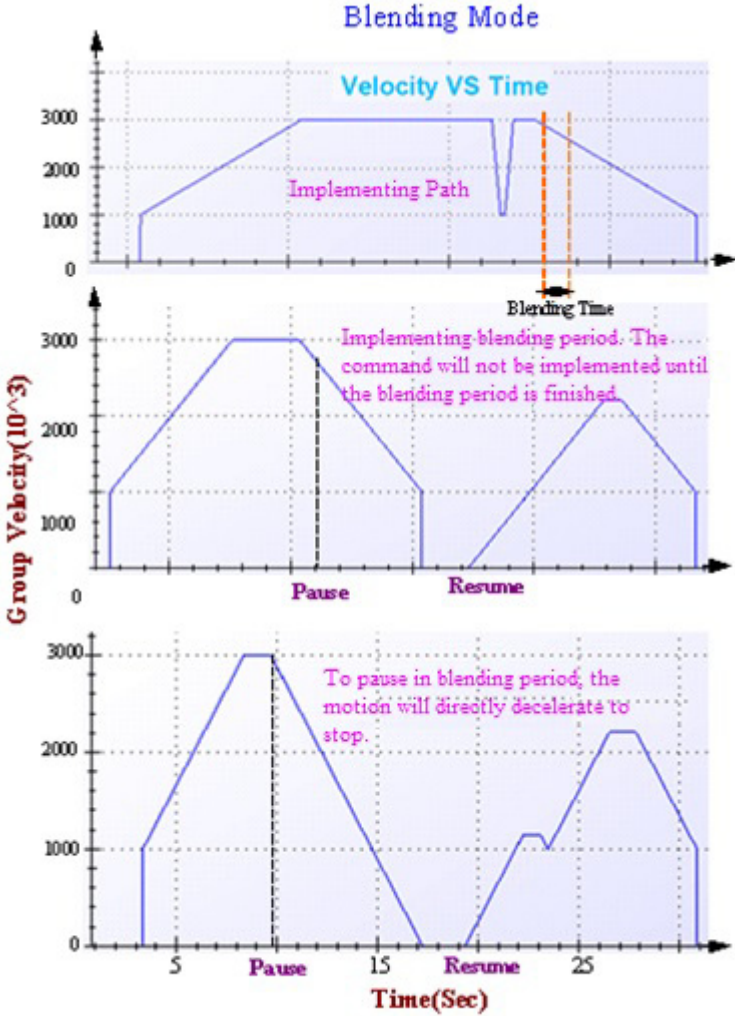
9.5.7.2 Pause / Resume in BlendingMode

In Blending Mode, Pause command will execute after Blending finished if Pause command is issued in the Blending segment.

The path table (can be edited by utility) is shown below, the acceleration, deceleration is 500, and the blending time is 500ms.

ID	MoveCmd	MoveMode	VelHigh	VelLow	Center0	Center1	EndPoint0	EndPoint1
0	Rel2DLine	Enable Blending	3000	1000	0	0	20000	20000
1	Rel2DArcCW	Enable Blending	3000	1000	4000	0	8000	0
2	EndPath	Enable Blending	1000	400	0	0	0	0

The velocity diagram is shown in the following figure:



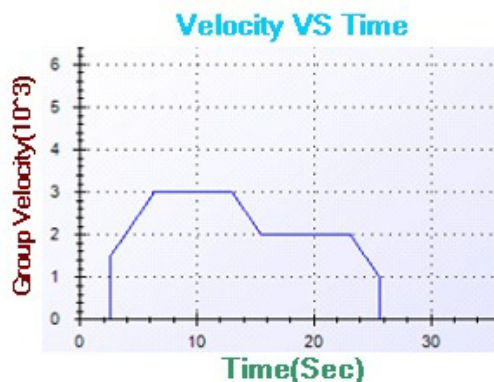
9.5.7.3 Pause / Resume in FlyMode

1. Disable speed forward function

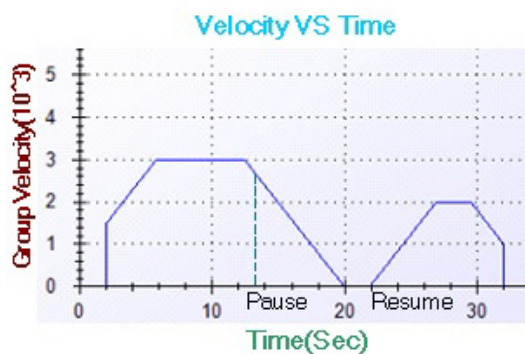
The path table (can be edited by utility) is shown below, the acceleration, deceleration is 500, disable speed forward function and the blending time is 0.

ID	MoveCmd	MoveMode	VelHigh	VelLow	Center0	Center1	EndPoint0	EndPoint1
0	Rel2DLine	Enable Blending	3000	1500	0	0	20000	20000
1	Rel2DArcCW	Enable Blending	2000	1000	8000	0	16000	0
2	EndPath	Enable Blending	1000	400	0	0	0	0

The velocity diagram is shown in the following figure:

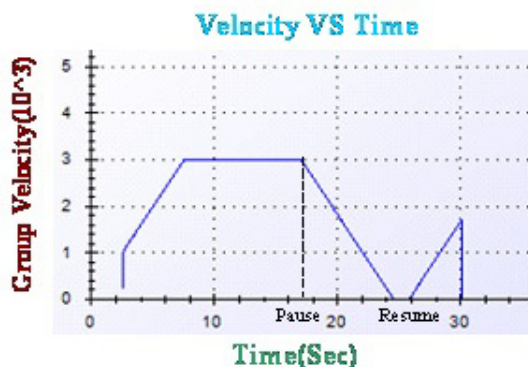


The velocity diagram of the pause and resume command is issued is shown in the following figure:



2. Enable speed forward function

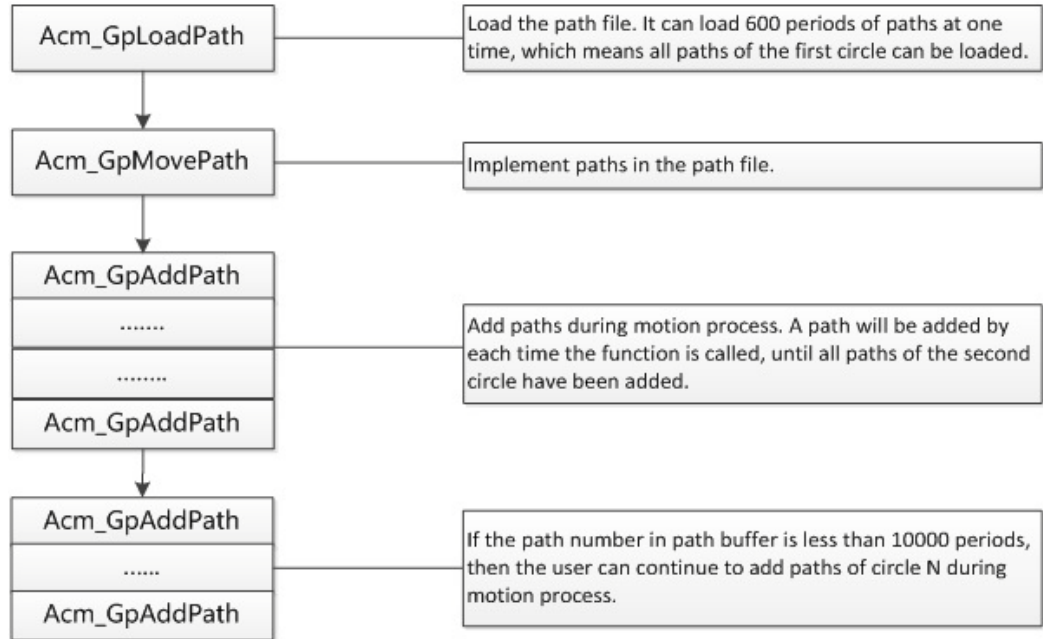
The path table is shown in previous table, the group velocity, the acceleration and deceleration is 500, the driving velocity is 3000 and the initial velocity is 1000. The velocity diagram is shown in the following figure:



9.5.8 Add Path during Moving

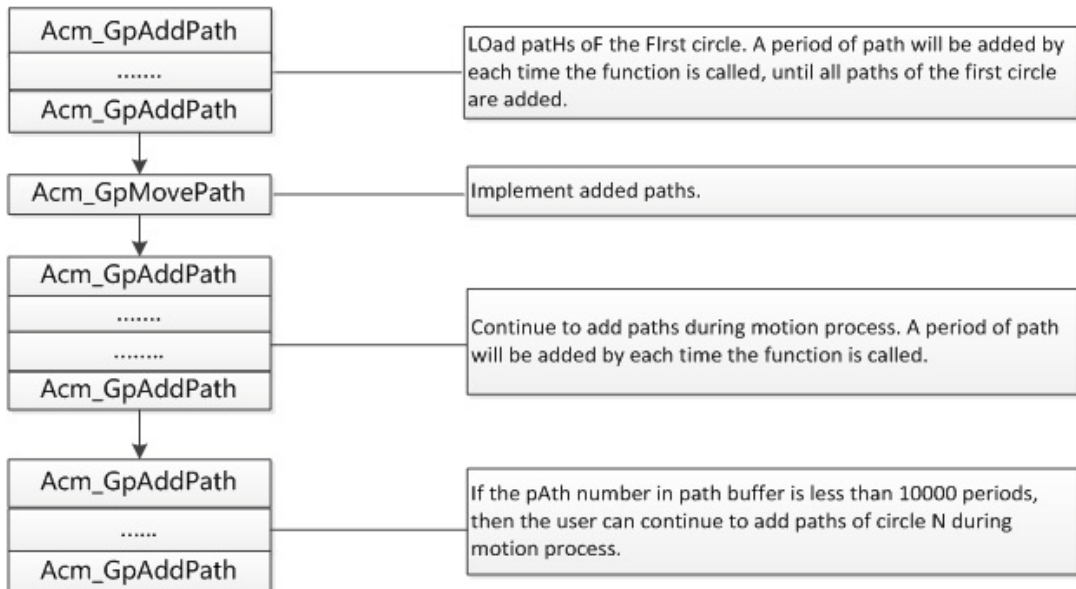
In some case, you will need to add multi-segment path dynamically while running. Take glass deburring process as an example, the whole trajectory is composed of multiple segment of path. After the first segment of path is added and processing this path, the second segment of path need to load into the path buffer. There are two ways to achieve the above process:

Method 1: Load the path by Acm_GpLoadPath



As shown in the figure, it can Load path data from path file by `Acm_GpLoadPath` API dynamically and it should not load more than 600 path data at one time. If the total path data in the system buffer are less than 7000, the path can be loaded continuously. You may know the status of the system path buffer through `Acm_GpGetPathStatus` API.

Method 2: Load the path by Acm_GpAddPath



As shown in the figure, it can add new path by `Acm_GpAddPath` API during moving. If the total path data in the system buffer are less than 7000, the path can be added continuously. You may know the status of the system path buffer through `Acm_GpGetPathStatus` API.

The execution path will continue to release executed path cache. For instance it has already loaded 7000 segment path into the system buffer in the beginning and 10 segment of path are executed, it will release its space in the system. Then you can continue to add the path.

9.5.9 Flow Chart

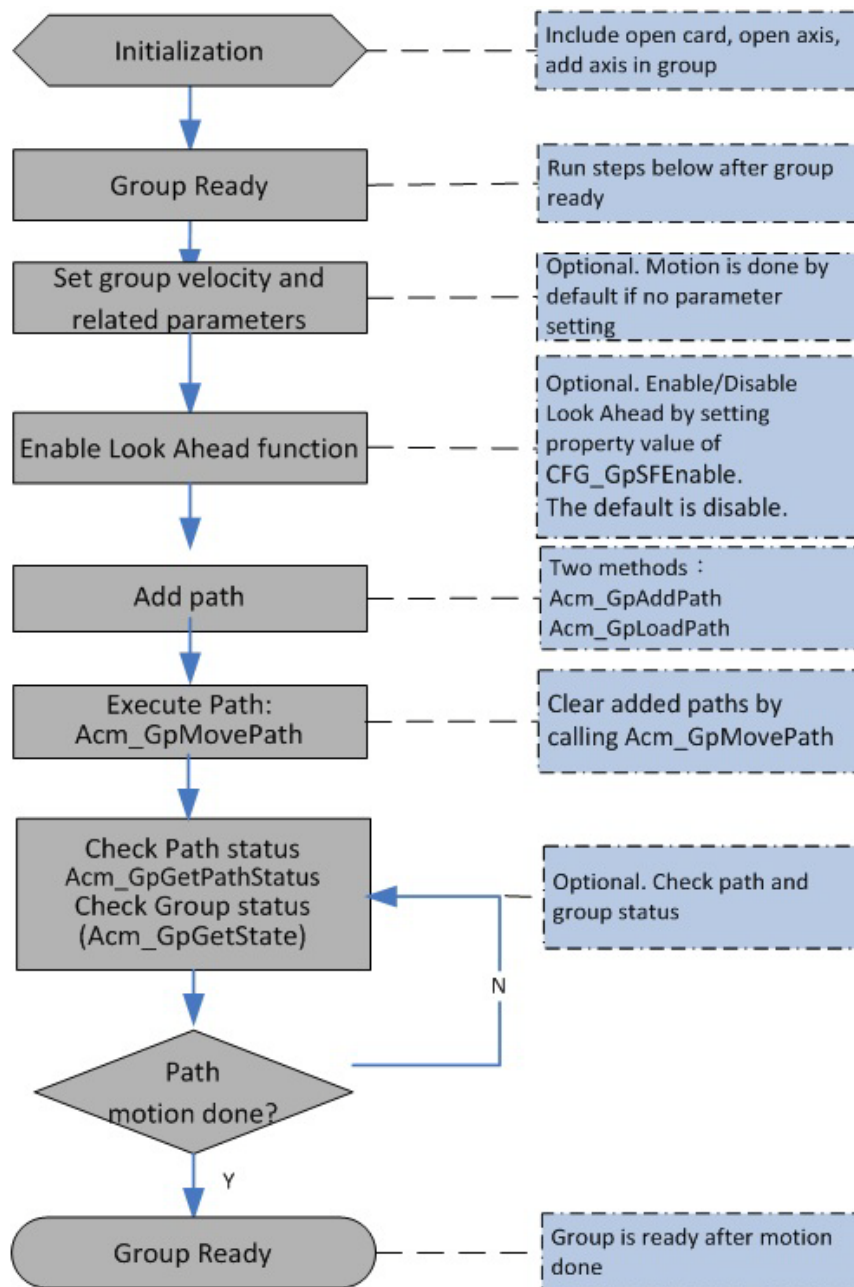


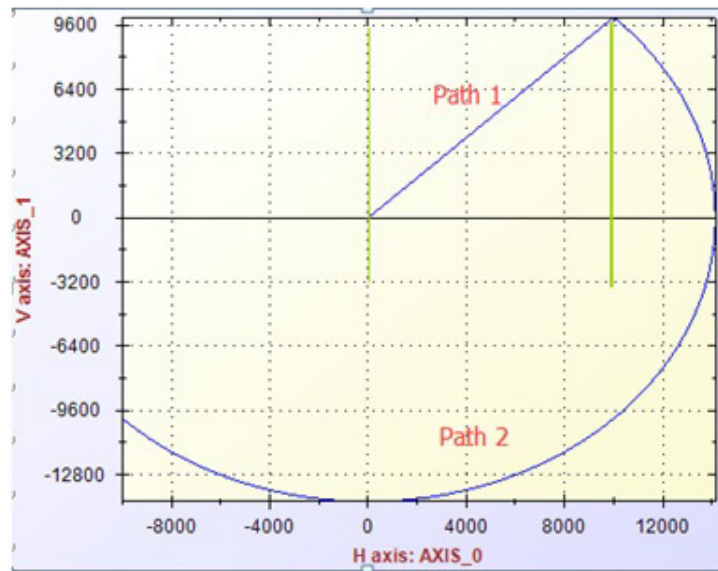
Figure 9.21 Path table motion flow chart

9.5.10 Example

This example executes path table motion on PCI-1203. It enables speed forward function. Refer to the table below for setting parameters.

Parameter\Path	Path1	Path2	Path3
MoveCmd	Rel2DLine	Rel2DArcCW	EndPath
MoveMode	0	0	0
FH	8000	8000	8000
FL	2000	2000	2000
EndPoint_DataArray	(-10000,-20000,0,0)	(-20000,-20000,0,0)	NULL
CenPoint_DataArray	NULL	(-10000,-10000,0,0)	NULL
ArrayElements	4	4	4
Function	Path table motion		
Initial velocity	2000PPU/S		
Operating velocity	8000PPU/S		
Acceleration	10000PPU/S ²		
Deceleration	10000PPU/S ²		
Velocity curve	T-Curve		
Speed forward	Enable		

The result is shown below:



VC Code:

```

HAND m_GpHand;//Group handle
U32  Ret;//Function return value
U16  MoveMode =0;//Eable blending mode
U16  MoveCmd;
F64  FH = 80000;
F64  FL = 2000;
U32  ArrayAxCnt= 4;//Number of array element
F64  CenPosArray[4]; //Center points
F64  EndPosArray[4]; //End points
---Initialization---
//Set parameters
Ret =Acm_SetF64Property(m_GpHand,PAR_GpVelLow,2000);//Initial velocity
Ret =Acm_SetF64Property(m_GpHand,PAR_GpVelHigh,8000);//Operating velocity
Ret =Acm_SetF64Property(m_GpHand,PAR_GpAcc,10000);//Acceleration
Ret =Acm_SetF64Property(m_GpHand,PAR_GpDec,10000);//Deceleration
Ret =Acm_SetF64Property(m_GpHand,PAR_GpJerk,0);//T-Curve
//Add relative linear interpolation motion command
EndPosArray[0] = -10000;
EndPosArray[1] = -20000;
MoveCmd =RelMoveLine;
Ret=Acm_GpAddPath(m_GpHand,MoveCmd,MoveMode,FH,FL,EndPosArray,
NULL,&ArrayAxCnt);
//Add relative arc interpolation motion command
CenPosArray[0] = -10000;
CenPosArray[1] = -10000;
EndPosArray[0] = -20000;
EndPosArray[1] = -20000;
MoveCmd = RelMoveArcCW;
Ret=Acm_GpAddPath(m_GpHand,MoveCmd,MoveMode,FH,FL,EndPosArray,
CenPosArray,&ArrayElements);
MoveMode=0;
//Add EndPath
Ret=Acm_GpAddPath(m_GpHand,MoveCmd,MoveMode,FH,FL,NULL,NULL,&ArrayElements);
//Do path table motion
Ret=Acm_GpMovePath(m_GpHand,NULL);
//Get status of path table motion
U32  Index,FreeCnt,Remain,CurCmd;
Ret = Acm_GpGetPathStatus(m_GpHand,&Index,&CurCmd,&Remain,&FreeCnt);
//Get group status
U16  GpState;
Acm_GpGetState(m_GpHand,&GpState); //Get group status

```

Check the function returned value to judge the function execution status and establish error handle mechanism is recommended to ensure the correct result. Refer to the Path example.

Chapter 10

IO API

10.1 About this Section

The PCI-1203 supports ADAM-5000/ECAT EtherCAT slave.

This chapter introduces programming tools Common Motion API definitions and how to use them for users to operate ADAM-5000/ECAT high speed system.

10.2 Function and Property

IO control functions are shown in Table 10.1, the functions in table can be called directly in the program.

Table 10.1: IO control functions

Function	Description
Acm_GetAvailableDevs	Get the list of available device numbers and device names
Acm_DevOpen	Open a specified device to get device handle
Acm_DevClose	Close a device
Acm_LoadMapFile	Load I/O mapping file in the EtherCAT network
Acm_DevReOpen	Reopen Device
Acm_DevEnableEvent	Enable the event check function
Acm_DevCheckEvent	Check event
Acm_DevGetSlaveInfo	Get the slave information according to the rotate switch on device
Acm_DaqDiGetBit	Get the bit data of specified DI channel
Acm_DaqDiGetByte	Get the byte data of specified DI Port
Acm_DaqDiGetBytes	Get the byte data of continuous DI Port
Acm_DaqDoSetBit	Set the bit data of specified DO channel
Acm_DaqDoSetByte	Set the byte data of specified DO port
Acm_DaqDoSetBytes	Set the byte data of continuous DO Port
Acm_DaqDoGetBit	Get the bit data of specified DO channel
Acm_DaqDoGetByte	Get the byte data of specified DO port
Acm_DaqDoGetBytes	Get the byte data of continuous DO Port
Acm_DaqAiGetRawData	Get the binary value of an analog input channel
Acm_DaqAiGetVoltData	Get the voltage value of an analog input channel when voltage inputs
Acm_DaqAiGetCurrData	Get the current value of an analog input channel when current inputs
Acm_DaqAoSetRawData	Set the binary value of an analog output channel
Acm_DaqAoSetVoltData	Set the voltage output value of an specified analog output channel within the analog voltage output range
Acm_DaqAoSetCurrData	Set the current output value of a specified analog output channel within the analog current output range
Acm_DaqAoGetRawData	Get the binary output value of a specified analog output channel
Acm_DaqAoGetVoltData	Get the voltage output value of a specified analog output channel within the analog voltage output range
Acm_DaqAoGetCurrData	Get the current output value of a specified analog output channel within the analog current output range
Acm_GetProperty	Get the property value through assigned PropertyID
Acm_GetU32Property	Get the property value belonging to unsigned 32 bit integer type

Table 10.1: IO control functions

Acm_GetI32Property	Get the property value belonging to signed 32 bit integer type
Acm_GetF64Property	Get the property value belonging to double type
Acm_GetChannelProperty	Get the DI/DO/AI/AO channel property value
Acm_SetChannelProperty	Set the DI/DO/AI/AO channel property value
Acm_GetMultiChannelProperty	Get the value continuous channels assigned by start channel ID and channel count.
Acm_SetMultiChannelProperty	Set the value continuous channels assigned by start channel ID and channel count.

I/O control related properties as shown in Table 10.2, the property can't be called directly, but set and get property values by calling related property function. Refer to Appendix for more detail information.

Table 10.2: IO control properties

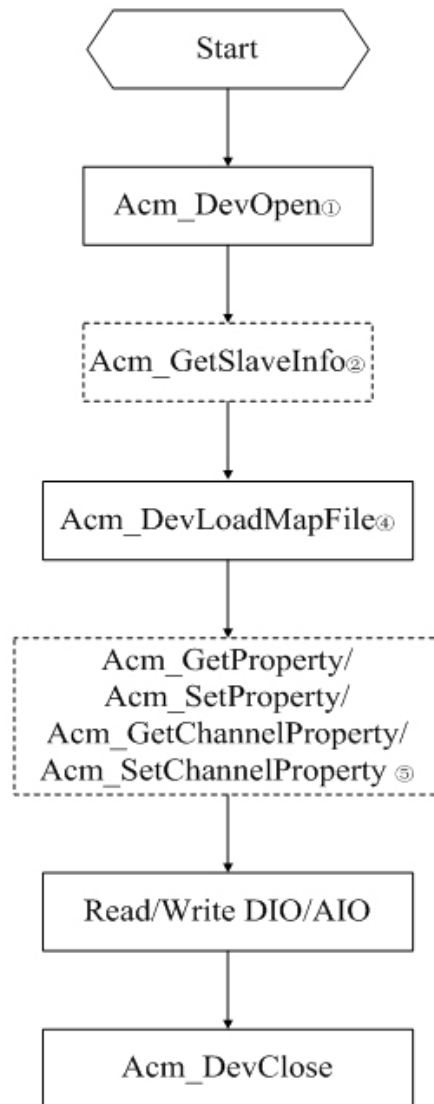
Configuration	Description
CFG_MasCycleTime	Get the cycle time of motion ring for data communication
CFG_IoCycleTime	Get the cycle time of IO ring for data communication
CFG_CH_DaqDiInvertEnable	Set/Get DI Invert
CFG_CH_DaqDiLowFilter	Set/Get down limit value of DI filter
CFG_CH_DaqDiHighFilter	Set/Get upper limit value of DI filter
CFG_CH_DaqDoFsvEnable	Set/Get DO Failure Safe Value
CFG_CH_DaqAoRange	Set/Get AO range
CFG_CH_DaqAiRange	Set/Get AI channel input range
CFG_CH_DaqAiEnable	Enable/Disable this AI channel
CFG_CH_DaqAiIntegrationTime	Set/Get AI setting time
CFG_CH_DaqAoFsv	Get AO Failure Safe Value or Set AO Failure Safe Value as current AO value
CFG_CH_DaqAoStartup	Get AO Startup Value or Set AO Startup Value as current AO value
Feature	Description
FT_DaqDiMaxChan	Get all DI channel count in the network
FT_DaqDoMaxChan	Get all DO channel count in the network
FT_DaqAiMaxDiffChan	Get all differential AI channel count in the network
FT_DaqAoMaxChan	Get all AO channel count in the network
FT_DaqCntMaxChan	Get all CNT channel count in the network

10.3 I/O Mapping

There might be many I/O slaves in EtherCAT network and various DI/DO/AI/AO modules in each slave. Mapping information is important for users to configure and operate these IO modules. For Common Motion API, users should read / write value from / to slaves by a specified number. We called the specified number as logical index or logical port / channel. The logical port / channel can be defined by users in the Common Motion Utility. For more detailed about how to read and how to define the mapping table, refer to chapter 4.3.5.

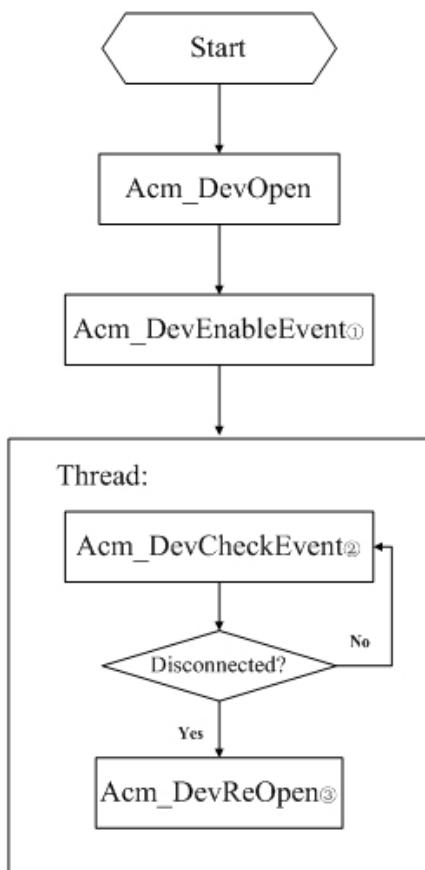
10.4 Flow Charts

10.4.1 Basic Flow



- (1). DeviceNumber need to be passed to Acm_DevOpen. DeviceNumber can be acquired by Acm_GetAvailableDevs.
- (2). Users can check the slave information by Acm_GetSlaveInfo. (Not necessary)
- (3). Users can load mapping file produced by utility, it's not necessary if the users do not change the default mapping index of DI/O and AI/O, but users should know the logical mapping index of DI/O and AI/O by utility.
- (4). Properties of DI/DO/AI/AO can be set by Acm_GetChannelProperty / Acm_SetChannelProperty

10.4.2 Event



- (1). Users need to enable Acm_DevEnableEvent first before checking Event.
- (2). Enable thread for Event and call Acm_DevCheckEvent cyclically in thread
- (3). Users need to call Acm_DevReOpen to reopen device once Event occurs .But the user must make sure the network reconnected before calling this API.

10.5 Example

VC Code:

```
HAND m_DevHand;//Device handle
U32 Ret; //Function return value
U32 DoCount, DiCount, AoCount, AiCount;//Number of DIO/AIO
U8 BitData, ByteData;
---Initialization---
Ret = Acm_GetU32Property(m_DevHand, FT_DaqDoMaxChan, &DoCount);
Ret = Acm_GetU32Property(m_DevHand, FT_DaqDiMaxChan, &DiCount);
Ret = Acm_GetU32Property(m_DevHand, FT_DaqAoMaxChan, &AoCount);
Ret = Acm_GetU32Property(m_DevHand, FT_DaqAiMaxDiffChan, &AiCount);
U8 *DIDataArray =new U8[DiCount/8], *DODataArray =new U8[DoCount/8];
// Set Do and Get Di
for (U32 i = 0; i <DoCount/8; i++)
{
    Ret = Acm_DaqDoSetByte(m_DevHand, i, 0xFF);
    Sleep(100);
    Ret = Acm_DaqDoGetByte(m_DevHand, i, &ByteData);
}
DODataArray[0] = 0x11;
DODataArray[1] = 0xAA;
Ret = Acm_DaqDoSetBytes(m_DevHand, 0, 2, DODataArray);
Ret = Acm_DaqDoGetBytes(m_DevHand, 0, 2, DODataArray);

for (U32 i = 0; i <DiCount; i++)
{
    Ret = Acm_DaqDiGetBit(m_DevHand, i, &BitData);
    if (DiCount%8 == 7)
        Ret = Acm_DaqDiGetByte(m_DevHand, i/8, &ByteData);
}
Ret = Acm_DaqDiGetBytes(m_DevHand, 0, DiCount/8, DIDataArray);

// SetAo and Get Aii
for(U8i = 0; i<AoCount; ++i)
{
    double aorange = 0.;
    float AoData = 0;
    U16 AoRawData = 0;
    Ret = Acm_GetChannelProperty(m_DevHand, i, CFG_CH_DaqAoRange,
&aorange);
    Ret = Acm_DaqAoGetRawData(m_DevHand, i, &AoRawData);
    switch((U32)aorange)
    {
        case CFG_DAQ_AO_NEG_10V_TO_10V:
        case CFG_DAQ_AO_NEG_5V_TO_5V:
```

```

        case CFG_DAQ_AO_NEG_2500MV_TO_2500MV:
        case CFG_DAQ_AO_NEG_1250MV_TO_1250MV:
        case CFG_DAQ_AO_NEG_625MV_TO_625MV:
        case CFG_DAQ_AO_NEG_0V_TO_10V:
        case CFG_DAQ_AO_NEG_0V_TO_5V:
            Ret = Acm_DaqAoSetVoltData(m_DevHand, i, 4.78);
            Ret = Acm_DaqAoGetVoltData(m_DevHand, i, &AoData);
            printf("Ch %d: Raw:%4X Eng:%fV\n",i, AoRawData, AoData);
            break;
        case CFG_DAQ_AO_0MA_TO_20MA:
        case CFG_DAQ_AO_4MA_TO_20MA:
            Ret = Acm_DaqAoSetRawData(m_DevHand, i, 0x0FFF);
            Ret = Acm_DaqAoGetCurrData(m_DevHand, i, &AoData);
            printf("Ch %d: Raw:%4X Eng:%f mA\n",i, AoRawData, AoData);
            break;
    }
}

for(USHORTi = 0; i<AiCount; ++i)
{
    double    airange = 0.;
    float     AiData = 0;
    U16       AiRawData = 0;
    errcde = Acm_GetChannelProperty(devHandle, i, CFG_CH_DaqAiRange,
    &airange);
    errcde = Acm_DaqAiGetRawData(devHandle, i, &AiRawData);
    switch((U32)airange)
    {
        case CFG_DAQ_AI_NEG_10V_TO_10V:
        case CFG_DAQ_AI_NEG_5V_TO_5V:
        case CFG_DAQ_AI_NEG_2500MV_TO_2500MV:
        case CFG_DAQ_AI_NEG_1250MV_TO_1250MV:
        case CFG_DAQ_AI_NEG_625MV_TO_625MV:
        case CFG_DAQ_AI_NEG_1V_TO_1V:
        case CFG_DAQ_AI_NEG_500MV_TO_500MV:
        case CFG_DAQ_AI_NEG_150MV_TO_150MV:
        case CFG_DAQ_AI_NEG_0_TO_10V:
        case CFG_DAQ_AI_NEG_0_TO_500MV:
            Ret = Acm_DaqAiGetVoltData(devHandle, i, &AiData);
            printf("Ch %d: Raw:%4X Eng:%fV\n",i, AiRawData, AiData);
            break;
        case CFG_DAQ_AI_0MA_TO_20MA:
        case CFG_DAQ_AI_4MA_TO_20MA:
        case CFG_DAQ_AI_NEG_20MA_TO_20MA:
            errcde = Acm_DaqAiGetCurrData(devHandle, i, &AiData);
            printf("Ch %d: Raw:%4X Eng:%f mA\n",i, AiRawData, AiData);

```

```
                break;  
            }  
    }
```

Check function returned value to judge the function execution status and establish error handle mechanism is recommended to ensure the correct result. Refer to the EthcatDI, EthcatDO and EthcatAI, EthcatAO example.

Appendix **A**

API List

A.1 Common

Acm_GetAvailableDevs

Format	U32 Acm_GetAvailableDevs (DEVLIST *DeviceList, U32 MaxEntries, PU32 OutEntries)		
Purpose	Get the list of available device numbers and device names		
Return	Error code		
Comments	Get the list of available device numbers and names of devices, of which driver has been loaded successfully.		
Format Parameters			
Name	Type	IN or OUT	Description
DeviceList	DEVLIST*	OUT	Pointer to returned available device info list.
MaxEntries	U32	IN	The max devices count to get.
OutEntries	PU32	OUT	The count of available device.

The structure of DEVLIST is:

```
typedef struct tagPT_DEVLIST
{
    DWORD    DeviceNum;
    CHAR     DeviceName[50];
    SHORT    NumOfSubDevices;
}
```

Acm_GetErrorMessage

Format	BOOL Acm_GetErrorMessage (U32 ErrorCode, LPTSTR lpszError,U32nMaxError)		
Purpose	Get the error message according to error code returned from API.		
Return	Nonzero if the function is successful; otherwise 0 if no error message text is available.		
Comments	Acm_GetErrorMessage will not copy more than nMaxError -1 characters to the buffer and it will always add a trailing null to end the string. If the buffer is too small, the error message may be truncated.		
Format Parameters			
Name	Type	IN or OUT	Description
ErrorCode	U32	IN	The returned error code of API.
lpszError	LPTSTR	OUT	The pointer to the string of error message.
nMaxError	U32	IN	The max length of string to receive error message.

A.2 Device Object

Acm_DevOpen

Format	U32 Acm_DevOpen (U32 DeviceNumber, PHAND DeviceHandle)		
Purpose	Open a specified device to get device handle.		
Return	Error code		
Comments	This function should be called firstly before any operation of the device.		
Format Parameters			
Name	Type	IN or OUT	Description
DeviceNumber	U32	IN	Device Number
DeviceHandle	PHAND	OUT	Return a point to the device handle

Acm_DevClose

Format	U32 Acm_DevClose (PHAND DeviceHandle)		
Purpose	Close a device.		
Return	Error code		
Comments	Last of all, the device must be closed through this function.		
Format Parameters			
Name	Type	IN or OUT	Description
DeviceHandle	PHAND	IN	A pointer to the device handle by Acm_DevOpen

Acm_DevReOpen

Format	U32 Acm_DevReOpen(HAND DeviceHandle)		
Purpose	Reopen Device.		
Return	Error code		
Comments	User needs to call this API to reopen the device in the following situation: (1). Slave is re-connected after disconnection. (2). Communication is not in OP mode. (3). Any of slaves is not in OP mode. (4). Retrieved the slave states do not match the actual connection.		
Format Parameters			
Name	Type	IN or OUT	Description
DeviceHandle	HAND	IN	A pointer to the device handle by Acm_DevOpen

Acm_DevLoadConfig

Format	U32 Acm_DevLoadConfig (HAND DeviceHandle, PI8 ConfigPath)		
Purpose	Set all configurations for the device according to the loaded file.		
Return	Error code		
Comments	Please refer to Chapter 7.1 for detail information		
Format Parameters			
Name	Type	IN or OUT	Description
DeviceHandle	HAND	IN	Device handle from Acm_DevOpen.
ConfigPath	PI8	IN	Pointer to a string that saves configuration file's path.

Acm_DevLoadMapFile

Format	U32 Acm_DevLoadMapFile (HAND DeviceHandle, PI8 FilePath)
Purpose	Load I/O mapping file in the EtherCAT network.
Return	Error code
Comments	<p>User need to load I/O mapping file in their own program before controlling DI/O and AI/O. This file is configured by Utility and records all physical and logical mapping relation in whole EtherCAT network. If user just use the default I/O mapping relationship without any modification, user need not to load mapping file by this API, but user should know the I/O mapping relationship by utility.</p> <p>All DI/O are arranged by "Port" (1 port=8 bits) in sequence. User can re-arrange the port number to map the real DI/O in ADAM-5000/ECAT or driver by themselves.</p> <p>The same, All AI/O arranged by "Channel" in sequence. User can re-arrange the channel ID to map the real AI/O in ADAM-5000/ECAT.</p> <p>Please be noted that you should know all mapping relationship between PortNumber or ChannelID and physical DI/DO/AI/AO.</p>

Format Parameters

Name	Type	IN or OUT	Description
DeviceHandle	HAND	IN	Device handle from Acm_DevOpen.
FilePath	PI8	IN	Pointer to a string that saves I/O mapping file's path.

Acm_GetProperty

Format	U32 Acm_GetProperty(HAND Handle, U32 PropertyID, PVOID Buffer, PU32 BufferLength)
Purpose	Get the property (feature property, configuration property or parameter property) value through assigned PropertyID.
Return	Error code
Comments	<p>User should pay attention on the data type and BufferLength to get the value of Property according to PropertyID.</p> <p>If the BufferLength is not the correct size, the return value will be error code "DataSizeNotCorrect". In this case, Buffer will return the value with the size of the property in BufferLength.</p> <p>About the detail information of PropertyID, please refer to Acm_GetU32Property, Acm_GetI32Property, and Acm_GetF64Property in Property List.</p>

Format Parameters

Name	Type	IN or OUT	Description
Handle	HAND	IN	Object handle. This handle may be device handle from Acm_DevOpen, or axis handle from Acm_AxOpen, or group handle from Acm_GpAddAxis
PropertyID	U32	IN	Property ID to query.
Buffer	PVOID	OUT	Return Property value.
BufferLength	PU32	IN/OUT	Buffer byte size for the property. This value must the same as the length of inquired property, or error will occur and return the actual size of the property in buffer length

Acm_GetU32Property

Format	U32 Acm_GetU32Property(HAND Handle, U32 PropertyID, PU32 Value)
Purpose	Get the unsigned 32 -bit integer property value through assigned PropertyID.
Return	Error code
Comments	HAND: Input Device Handle to get device property. Input Axis Handle to get axis property. Input Group Handle to get group property. PropertyID: Each property only has one property ID and the data type of property value to query must be unsigned 32 -bit integer.

Format Parameters

Name	Type	IN or OUT	Description
Handle	HAND	IN	Object handle. This handle may be device handle from Acm_DevOpen, or axis handle from Acm_AxOpen, or group handle from Acm_GpAddAxis
PropertyID	U32	IN	Property ID to query.
Value	PU32	OUT	Return property value

Acm_GetI32Property

Format	U32 Acm_GetI32Property(HAND Handle, U32 PropertyID, PI32 Value)
Purpose	Get the signed 32 -bit integer property value through assigned PropertyID.
Return	Error code
Comments	HAND: Input Device Handle to get device property. Input Axis Handle to get axis property. Input Group Handle to get group property. PropertyID: Each property only has one property ID and the data type of property value to query must be signed 32 -bit integer.

Format Parameters

Name	Type	IN or OUT	Description
Handle	HAND	IN	Object handle. This handle may be device handle from Acm_DevOpen, or axis handle from Acm_AxOpen, or group handle from Acm_GpAddAxis
PropertyID	U32	IN	Property ID to query.
Value	PI32	OUT	Return property value

Acm_GetF64Property

Format	U32 Acm_GetF64Property(HAND Handle, U32 PropertyID, PF64 Value)
Purpose	Get the double property value through assigned PropertyID.
Return	Error code
Comments	HAND: Input Device Handle to get device property. Input Axis Handle to get axis property. Input Group Handle to get group property. PropertyID: Each property only has one property ID and the data type of property value to query must be double.

Format Parameters

Name	Type	IN or OUT	Description
Handle	HAND	IN	Object handle. This handle may be device handle from Acm_DevOpen, or axis handle from Acm_AxOpen, or group handle from Acm_GpAddAxis
PropertyID	U32	IN	Property ID to query.
Value	PF64	OUT	Return property value

Acm_SetProperty

Format	U32 Acm_SetProperty(HAND Handle, U32 PropertyID, PVOID Buffer, U32 BufferLength)
Purpose	Set the property (configuration property or parameter property) value through assigned PropertyID.
Return	Error code
Comments	User should pay attention that not all of properties in Property List can be set to new property value; only the writable properties can be reset property value. If the BufferLength is not correct, the return value will be error code "DataSizeNotCorrect". About the detail information of PropertyID, please refer to Acm_SetU32Property, Acm_SetI32Property, and Acm_SetF64Property in Property List.

Format Parameters

Name	Type	IN or OUT	Description
Handle	HAND	IN	Object handle. This handle may be device handle from Acm_DevOpen, or axis handle from Acm_AxOpen, or group handle from Acm_GpAddAxis
PropertyID	U32	IN	Property ID to query.
Buffer	PVOID	OUT	Return Property value.
BufferLength	PU32	IN/OUT	Buffer byte size for the property. This value must be the same as the length of inquired property, or error will occur and return the actual size of the property in buffer length

Acm_SetU32Property

Format	U32 Acm_SetU32Property(HAND Handle, U32 PropertyID, U32 Value)
Purpose	Set the unsigned 32-bit integer property value through assigned PropertyID.
Return	Error code
Comments	HAND: Input Device Handle to get device property. Input Axis Handle to get axis property. Input Group Handle to get group property. PropertyID: Each property only has one property ID and the data type of property value to set must be unsigned 32-bit integer.

Format Parameters

Name	Type	IN or OUT	Description
Handle	HAND	IN	Object handle. This handle may be device handle from Acm_DevOpen, or axis handle from Acm_AxOpen, or group handle from Acm_GpAddAxis
PropertyID	U32	IN	Property ID to set.
Value	U32	IN	Set property value

Acm_SetI32Property

Format	U32 Acm_SetI32Property(HAND Handle, U32 PropertyID, I32 Value)
Purpose	Set the signed 32-bit integer property value through assigned PropertyID.
Return	Error code
Comments	HAND: Input Device Handle to get device property. Input Axis Handle to get axis property. Input Group Handle to get group property. PropertyID: Each property only has one property ID and the data type of property value to set must be signed 32-bit integer.

Format Parameters

Name	Type	IN or OUT	Description
Handle	HAND	IN	Object handle. This handle may be device handle from Acm_DevOpen, or axis handle from Acm_AxOpen, or group handle from Acm_GpAddAxis
PropertyID	U32	IN	Property ID to set.
Value	I32	IN	Set property value

Acm_SetF64Property

Format	U32 Acm_SeF64Property(HAND Handle, U32 PropertyID,F64 Value)		
Purpose	Set the double property value through assigned PropertyID.		
Return	Error code		
Comments	HAND: Input Device Handle to get device property. Input Axis Handle to get axis property. Input Group Handle to get group property. PropertyID: Each property only has one property ID and the data type of property value to set must be double.		

Format Parameters

Name	Type	IN or OUT	Description
Handle	HAND	IN	Object handle. This handle may be device handle from Acm_DevOpen, or axis handle from Acm_AxOpen, or group handle from Acm_GpAddAxis
PropertyID	U32	IN	Property ID to set.
Value	F64	IN	Set property value

Acm_GetChannelProperty

Format	U32 Acm_GetChannelProperty (HAND Handle, U32 ChannelID, U32 PropertyID, PF64 Value)		
Purpose	Get the DI/DO/AI/AO channel property value.		
Return	Error code		

Comments	<p>If user wants to get the DI/DO property value by channel. This channelID can be got by calculation: ChannelID= Port number * 8 + Port Index. Take the second DO in Port 3 for an example, the ChannelID= 3*8+2=26. User should check or set the Port number and ChannelID in whole network through mapping Info in Utility before using this API to get the property. User can get the detail information about the property value type in Property List.</p> <p>Note: User need call Acm_DevLoadMapFile(If user uses the default mapping relationship, this API need not to load first before getting Property value by Acm_GetChannelProperty</p>		
-----------------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	--	--

Format Parameters

Name	Type	IN or OUT	Description
Handle	HAND	IN	Device handle from Acm_DevOpen.
ChannelID	U32	IN	DI/DO/AI/AO channel ID
PropertyID	U32	IN	PropertyID
Value	PF64	OUT	Get property value

Acm_SetChannelProperty

Format	U32 Acm_SetChannelProperty (HAND Handle, U32 ChannelID, U32 PropertyID, F64 Value)
Purpose	Set the DI/DO/AI/AO channel property value.
Return	Error code
Comments	<p>If user wants to get the DI/DO property value by channel. This channelID can be got by calculation: ChannelID= Port number * 8 + Port Index. Take the second DO in Port 3 for an example, the ChannelID= 3*8+2=26. User should check or set the Port number and ChannelID in whole network through mapping Info in Utility before using this API to get the property. User can get the detail information about the property value type in Property List.</p> <p>Note: User need call Acm_DevLoadMapFile(If user uses the default mapping relationship, this API need not to load) first before getting Property value by Acm_SetChannelProperty</p>

Format Parameters

Name	Type	IN or OUT	Description
Handle	HAND	IN	Device handle from Acm_DevOpen.
ChannelID	U32	IN	DI/DO/AI/AO channel ID
PropertyID	U32	IN	PropertyID
Value	F64	IN	Set Property value

Acm_GetMultiChannelProperty

Format	U32 Acm_GetMultiChannelProperty (HAND Handle, U32 PropertyID, U32 StartChID, U32 ChCount, PF64 ValueArray)
Purpose	Get the value continuous channels assigned by start channel ID and channel count.
Return	Error code
Comments	<p>If user wants to get the DI/DO property value by channel. This channelID can be got by calculation: ChannelID= Port number * 8 + Port Index. Take the second DO in Port 3 for an example, the ChannelID= 3*8+2=26. User should check or set the Port number and ChannelID in whole network through mapping Info in Utility before using this API to get the property. User can get the detail information about the property value type in Property List.</p> <p>Note:</p> <ol style="list-style-type: none">(1). User needs call Acm_DevLoadMapFile (If user uses the default mapping relationship, this API need not to load) first before getting Property value by this API.(2). The StartChID + ChCount cannot greater than total channel count.

Format Parameters

Name	Type	IN or OUT	Description
Handle	HAND	IN	Device handle from Acm_DevOpen.
PropertyID	U32	IN	PropertyID
StartChID	U32	IN	Start channelID.
ChCount	U32	IN	Channel count
ValueArray	PF64	OUT	Value array.

Acm_SetMultiChannelProperty

Format	U32 Acm_SetMultiChannelProperty (HAND Handle, U32 PropertyID, U32 StartChID, U32 ChCount, PF64 ValueArray)
Purpose	Set the value continuous channels assigned by start channel ID and channel count.
Return	Error code
Comments	<p>If user wants to set the DI/DO property value by channel. This channelID can be got by calculation: ChannelID= Port number * 8 + Port Index.</p> <p>Take the second DO in Port 3 for an example, the ChannelID= 3*8+2=26. User should check or set the Port number and ChannelID in whole network through mapping Info in Utility before using this API to get the property. User can get the detail information about the property value type in Property List.</p> <p>Note:</p> <ol style="list-style-type: none"> (1). User needs call Acm_DevLoadMapFile (If user uses the default mapping relationship, this API need not to load) first before setting Property value by this API. (2). The StartChID + ChCount cannot greater than total channel count.

Format Parameters

Name	Type	IN or OUT	Description
Handle	HAND	IN	Device handle from Acm_DevOpen.
PropertyID	U32	IN	PropertyID
StartChID	U32	IN	Start channelID.
ChCount	U32	IN	Channel count
ValueArray	PF64	IN	Value array

Acm_GetLastError

Format	U32 Acm_GetLastError (HAND ObjectHandle)
Purpose	Get device or axis or group's last error code.
Return	Error code
Comments	To get detail information of error code by Acm_GetErrorMessage.

Format Parameters

Name	Type	IN or OUT	Description
ObjectHandle	HAND	IN	Object handle. This handle may be device handle from Acm_DevOpen, or axis handle from Acm_AxOpen, or group handle from Acm_GpAddAxis

Acm_DevEnableEvent

Format	U32 Acm_DevEnableEvent (HAND DeviceHandle, U32 DevEnableEvt)
Purpose	Enable the event check function
Return	Error code
Comments	This API is used to enable the event check function. User can find the disconnection event occurs by Acm_DevCheckEvent when open the disconnection checks function. Note: Check event need to open threads to call Acm_DevCheckEvent.

Format Parameters

Name	Type	IN or OUT	Description								
DeviceHandle	HAND	IN	Device handle from Acm_DevOpen.								
			Enable Event type:								
			<table border="1"><thead><tr><th>Bit</th><th>Status</th></tr></thead><tbody><tr><td>0</td><td>Motion ring disconnection</td></tr><tr><td>1</td><td>IO ring disconnection</td></tr><tr><td>2~31</td><td>Reserved</td></tr></tbody></table>	Bit	Status	0	Motion ring disconnection	1	IO ring disconnection	2~31	Reserved
Bit	Status										
0	Motion ring disconnection										
1	IO ring disconnection										
2~31	Reserved										
DevEnableEvt	U32	IN									

Acm_DevCheckEvent

Format	U32 Acm_DevCheckEvent (HAND DeviceHandle, PU32 DevEnableEvt, U32 Millisecond)
Purpose	Enable the event check function
Return	Error code
Comments	This API is used to check events status only for those event enabled by Acm_DevEnableEvent. User can find the disconnection event occurs by Acm_DevCheckEvent when open the disconnection checks function. Also, user can get the information about which slaves are disconnected by Acm_DevGetErrorTable. Note: Check event need to open threads to call Acm_DevCheckEvent.

Format Parameters

Name	Type	IN or OUT	Description
DeviceHandle	HAND	IN	Device handle from Acm_DevOpen.
DevEnableEvt	PU32	OUT	Check enabled event status
Millisecond	U32	IN	Specify the time out value in millisecond for each checking (INFINITE as usual)

Acm_CheckMotionEvent

Format	U32 Acm_CheckMotionEvent (HAND DeviceHandle, PU32 AxEvtStatusArray, PU32 GpEvtStatusArray, U32 AxArrayElements, U32 GpArrayElements, U32 Millisecond)
Purpose	Check axis and groups enabled motion event status.
Return	Error code
Comments	If you want to get event status of axis or groups, you should enable these events by calling Acm_EnableMotionEvent. User should create a new thread to check event status.

Format Parameters

Name	Type	IN or OUT	Description																																				
DeviceHandle	HAND	IN	Device handle from Acm_DevOpen.																																				
AxEn-ableEvtArray	PU32	IN	<p>Array[n]: Returned interrupt event status of each axis. n is the axis count of motion device. Each array element is 32 bits data type, each bit represents different event types:</p> <table border="1"> <thead> <tr> <th>Bit</th> <th>31...6</th> <th>5</th> <th>4</th> <th>3</th> <th>2</th> <th>1</th> <th>0</th> </tr> </thead> <tbody> <tr> <td rowspan="3">Description</td> <td rowspan="3">Reserved</td> <td>EVT_A</td> <td>EVT_A</td> <td>EVT_A</td> <td>EVT_A</td> <td>EVT_A</td> <td>EVT_A</td> </tr> <tr> <td>X_VH_</td> <td>X_VH_</td> <td>X_ERR</td> <td>X_LAT</td> <td>X_CO</td> <td>X_MO</td> </tr> <tr> <td>END</td> <td>START</td> <td>OR</td> <td>CHED</td> <td>MPAR</td> <td>TION_</td> </tr> <tr> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td>ED</td> <td>DONE</td> </tr> </tbody> </table> <p>Bit n = 1: Axis Motion Done event occurred; Bit n = 0: Event not occurred or disabled.</p>	Bit	31...6	5	4	3	2	1	0	Description	Reserved	EVT_A	EVT_A	EVT_A	EVT_A	EVT_A	EVT_A	X_VH_	X_VH_	X_ERR	X_LAT	X_CO	X_MO	END	START	OR	CHED	MPAR	TION_							ED	DONE
Bit	31...6	5	4	3	2	1	0																																
Description	Reserved	EVT_A	EVT_A	EVT_A	EVT_A	EVT_A	EVT_A																																
		X_VH_	X_VH_	X_ERR	X_LAT	X_CO	X_MO																																
		END	START	OR	CHED	MPAR	TION_																																
						ED	DONE																																
GpEn-ableEvtArray	PU32	IN	<p>Array[n]: Returned interrupt event status for each group. n is just 1. GpEvtStatus is 32 bits data type array and currently the values of n can only be 1.</p> <table border="1"> <thead> <tr> <th>Bit</th> <th>Data</th> <th>31...n</th> <th>1</th> <th>0</th> </tr> </thead> <tbody> <tr> <td rowspan="3">Description</td> <td>GpEnable</td> <td>EVT_GPn_MO</td> <td>EVT_GP1_MO</td> <td>EVT_GP0_MO</td> </tr> <tr> <td>EvtArray[0]</td> <td>TION_DONE</td> <td>TION_DONE</td> <td>TION_DONE</td> </tr> <tr> <td>GpEnable</td> <td>EVT_GPn_VH</td> <td>EVT_GP1_VH</td> <td>EVT_GP0_VH_</td> </tr> <tr> <td>EvtArray[1]</td> <td>_START</td> <td>_START</td> <td>START</td> </tr> <tr> <td>GpEnable</td> <td>EVT_GPn_VH</td> <td>EVT_GP1_VH</td> <td>EVT_GP0_VH_</td> </tr> <tr> <td>EvtArray[2]</td> <td>_END</td> <td>_END</td> <td>END</td> </tr> </tbody> </table> <p>Bit n = 1: Group Motion Done event occurred; Bit n = 0: Event not occurred or disabled.</p> <p>Note: EVT_GPn_MOTION_DONE/ EVT_GPn_VH_START/EVT_GPn_VH_END, n is GroupID. It can be get from PAR_GpGroupID property.</p>	Bit	Data	31...n	1	0	Description	GpEnable	EVT_GPn_MO	EVT_GP1_MO	EVT_GP0_MO	EvtArray[0]	TION_DONE	TION_DONE	TION_DONE	GpEnable	EVT_GPn_VH	EVT_GP1_VH	EVT_GP0_VH_	EvtArray[1]	_START	_START	START	GpEnable	EVT_GPn_VH	EVT_GP1_VH	EVT_GP0_VH_	EvtArray[2]	_END	_END	END						
Bit	Data	31...n	1	0																																			
Description	GpEnable	EVT_GPn_MO	EVT_GP1_MO	EVT_GP0_MO																																			
	EvtArray[0]	TION_DONE	TION_DONE	TION_DONE																																			
	GpEnable	EVT_GPn_VH	EVT_GP1_VH	EVT_GP0_VH_																																			
EvtArray[1]	_START	_START	START																																				
GpEnable	EVT_GPn_VH	EVT_GP1_VH	EVT_GP0_VH_																																				
EvtArray[2]	_END	_END	END																																				
AxArrayElements	U32	IN	Number of AxEvtStatusArray elements.																																				
GpArrayElements	U32	IN	Number of GpEvtStatusArray elements. It should be 1.																																				
Millisecond	U32	IN	Specify the time out value in millisecond for each checking																																				

Acm_EnableMotionEvent

Format	U32 Acm_EnableMotionEvent (HAND DeviceHandle, PU32 AxEnableEvtArray, PU32 GpEnableEvtArray, U32 AxArrayElements, U32 GpArrayElements)
Purpose	Enable motion event.
Return	Error code
Comments	After enable some events of axis or groups, the event status can be get from Acm_CheckMotionEvent.

Format Parameters

Name	Type	IN or OUT	Description																		
DeviceHandle	HAND	IN	Device handle from Acm_DevOpen.																		
AxEnableEvtArray	PU32	IN	<p>Array[n], enable interrupt event for each axis, n is the axis count of motion device. Array is of 32 bits data type, each bit represents different Event types:</p> <table border="1"> <thead> <tr> <th>Bit</th> <th>31...6</th> <th>5</th> <th>4</th> <th>3</th> <th>2</th> <th>1</th> <th>0</th> </tr> </thead> <tbody> <tr> <td>Description</td> <td>Reserved</td> <td>EVT_AX_VH_END</td> <td>EVT_AX_VH_START</td> <td>EVT_AX_ERR_OR</td> <td>EVT_AX_LAT_CHED</td> <td>EVT_AX_CO_MPAR</td> <td>EVT_AX_MOTION_DONE</td> </tr> </tbody> </table> <p>Bit n = 1: Enable event; Bit n = 0: Disable event.</p>	Bit	31...6	5	4	3	2	1	0	Description	Reserved	EVT_AX_VH_END	EVT_AX_VH_START	EVT_AX_ERR_OR	EVT_AX_LAT_CHED	EVT_AX_CO_MPAR	EVT_AX_MOTION_DONE		
Bit	31...6	5	4	3	2	1	0														
Description	Reserved	EVT_AX_VH_END	EVT_AX_VH_START	EVT_AX_ERR_OR	EVT_AX_LAT_CHED	EVT_AX_CO_MPAR	EVT_AX_MOTION_DONE														
GpEnableEvtArray	PU32	IN	<p>Array[n], enable interrupt event for each group. GpEnableEvtArray is 32 bits data type array and currently the value of n can only be 1.</p> <table border="1"> <thead> <tr> <th>Bit</th> <th>Data</th> <th>31...n</th> <th>1</th> <th>0</th> </tr> </thead> <tbody> <tr> <td rowspan="3">Description</td> <td>GpEnable EvtArray[0]</td> <td>EVT_GPn_MO_TION_DONE</td> <td>EVT_GP1_MO_TION_DONE</td> <td>EVT_GP0_MO_TION_DONE</td> </tr> <tr> <td>GpEnable EvtArray[1]</td> <td>EVT_GPn_VH_START</td> <td>EVT_GP1_VH_START</td> <td>EVT_GP0_VH_START</td> </tr> <tr> <td>GpEnable EvtArray[2]</td> <td>EVT_GPn_VH_END</td> <td>EVT_GP1_VH_END</td> <td>EVT_GP0_VH_END</td> </tr> </tbody> </table> <p>Bit n = 1: Enable event; Bit n = 0: Disable event.</p> <p>Note: For EVT_GPn_MOTION_DONE, n is GroupID. It can be got from PAR_GpGroupID property.</p>	Bit	Data	31...n	1	0	Description	GpEnable EvtArray[0]	EVT_GPn_MO_TION_DONE	EVT_GP1_MO_TION_DONE	EVT_GP0_MO_TION_DONE	GpEnable EvtArray[1]	EVT_GPn_VH_START	EVT_GP1_VH_START	EVT_GP0_VH_START	GpEnable EvtArray[2]	EVT_GPn_VH_END	EVT_GP1_VH_END	EVT_GP0_VH_END
Bit	Data	31...n	1	0																	
Description	GpEnable EvtArray[0]	EVT_GPn_MO_TION_DONE	EVT_GP1_MO_TION_DONE	EVT_GP0_MO_TION_DONE																	
	GpEnable EvtArray[1]	EVT_GPn_VH_START	EVT_GP1_VH_START	EVT_GP0_VH_START																	
	GpEnable EvtArray[2]	EVT_GPn_VH_END	EVT_GP1_VH_END	EVT_GP0_VH_END																	
AxArrayElements	U32	IN	number of AxEvtStatusArray elements																		
GpArrayElements	U32	IN	number of GpEvtStatusArray elements																		

Acm_DevGetMasInfo

Format	U32 Acm_DevGetMasInfo(HAND DeviceHandle, PVOID pMasInfo, PU16 SlaveIPArray, PU32 SlvCnt)		
Purpose	Get the master information according to the ring on device		
Return	Error code		
Comments			
Format Parameters			
Name	Type	IN or OUT	Description
Handle	HAND	IN	Device handle from Acm_DevOpen.
pMasInfo	PVOID	IN	0: Motion ring 1: IO ring
SlaveIPArray	PU16	OUT	Returned slave ID list from the specified ring.
SlvCnt	PU32	OUT	Slave count

Acm_DevGetSlaveInfo

Format	U32 Acm_DevGetSlaveInfo(HAND DeviceHandle, U16 RingNo, U16 SlaveIP, PVOID plInfo)		
Purpose	Get the slave information according to the slave ID on device		
Return	Error code		
Comments			
Format Parameters			
Name	Type	IN or OUT	Description
Handle	HAND	IN	Device handle from Acm_DevOpen.
RingNo	U16	IN	0: Motion ring 1: IO ring
SlaveIP	U16	IN	Slave ID Range: 0x001~0xffff
plInfo	PVOID	OUT	Returned information from slave. It should be a pointer to the struct type below: typedef struct _ADVAPI_SLAVE_INFO_ { ULONG SlaveID; ULONG Position; ULONG VendorID; ULONG ProductID; ULONG RevisionNo; ULONG SerialNo; ULONG driverCnts; port_info ports[4]; ULONG transmission_delay; char DeviceName[64]; }ADVAPI_SLAVE_INFO, *PADVAPI_SLAVE_INFO;

Acm_DevGetModuleInfo

Format	U32 Acm_DevGetModuleInfo(HAND DeviceHandle, U16 RingNo, U16 SlaveIP, PU32 ModIDArray, PU32 ModCnt)		
Purpose	Get the module information according to the rotate switch on device		
Return	Error code		
Comments			
Format Parameters			
Name	Type	IN or OUT	Description
Handle	HAND	IN	Device handle from Acm_DevOpen.
RingNo	U16	IN	0: Motion ring 1: IO ring
SlaveIP	U16	IN	Slave ID Range: 0x001~0xffff
ModIDArray	PU32	OUT	Returned module ID list from the specified slave.
ModCnt	PU32	OUT	Module count

Acm_DevSetSlaveID

Format	U32 Acm_DevSetSlaveID(HAND DeviceHandle, U16 RingNo, U16 SlaveIP, U16 SlaveNewIP)		
Purpose	Set the slave new ID		
Return	Error code		
Comments			
Format Parameters			
Name	Type	IN or OUT	Description
Handle	HAND	IN	Device handle from Acm_DevOpen.
RingNo	U16	IN	0: Motion ring 1: IO ring
SlaveIP	U16	IN	Slave ID Range:0x001~0xffff
Slave-NewIP	U16	IN	New slave ID, the new slave ID cannot be duplicated with other slave ID in the ring. Range: 0x001~0xffff

Acm_DevDownloadCAMTable

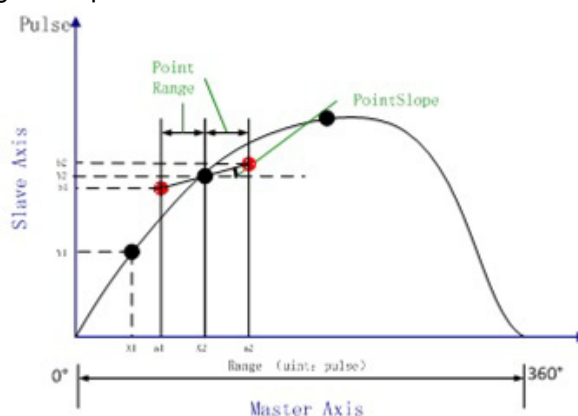
Format	U32 Acm_DevDownloadCAMTable (HAND DeviceHandle, U32 CamTableID, PF64 pMasterArray, PF64 pSlaveArray, PF64 pPointRangeArray, PF64 pPointSlopeArray, U32 ArrayElements)
Purpose	This function downloads a CAM table profile which describes the ratio relationship of leading and following axis.
Return	Error code

Camming is characterized by dynamic ratio between the leading and following axis, and by the phase shift. The transmission ratio is described by a CamTable.

Camming is done with one table (two dimensional - describing master and slave positions together). The table should be strictly monotonic rising or falling, going both reverse and forward with the master.

PCI-1285/1285E do not support this API.

The meaning of the parameters is as follow:

Comments

As top-figure, Range is the needed pulse number when master axis rotates 360°. The black points in figure are composed of master values (eg. X1, X2) in MasterArray and corresponding slave values in SlaveArray, and the red points created by black points and assigned PointRange and PointSlope are called reference points. Cam curve is fitted by points in CamTable composed of MasterArray and SlaveArray. The horizontal axis is the pulse number when master axis moves at some angles. The vertical axis is the pulse number of slave axis when master moves at some angels.

Range must be set into master axis by property CFG_AxModuleRange.

About cam operation, see about E-Cam Flow Chart in Chapter 9.4.3.

Format Parameters

Name	Type	IN or OUT	Description
DeviceHandle	HAND	IN	Device handle form Acm_DevOpen.
CamTableID	U32	IN	Identifier of CAM table. PCI-1245, PCI-1265 and PCI-1203 reserves 2 sets of cam table. So the ID can be 0, 1.
pMasterArray	PF64	IN	Master position array of CAM table points.
pSlaveArray	PF64	IN	Slave position array of CAM table points.
pPointRange-Array	PF64	IN	Point range of CAM table point.
pPointSlope-Array	PF64	IN	Point slope of CAM table point.
ArrayElements	U32	IN	Element count in the pMasterArray/ pSlaveArray/ pPointRangeArray/ pPointSlopeArray array. The Max. Element count is 128.

Acm_DevConfigCAMTable

Format	U32 Acm_DevConfigCAMTable (HAND DeviceHandle, U32 Cam-TableID,U32 Periodic, U32 MasterAbsolute, U32 SlaveAbsolute)
Purpose	This function sets the relevant parameters of the cam table.
Return	Error code

Camming is done with one table (two dimensional - describing master and slave positions together).

There are two types of Camming:

■ Periodic

Periodic: Once a curve is executed the camming immediately starts again at the beginning of the curve. As follow:



Non-periodic: After a curve is executed the execution stops.



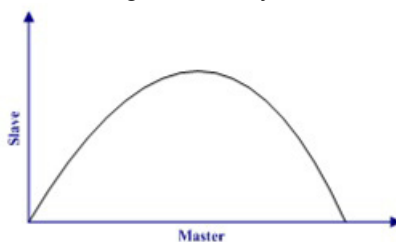
Comments

■ MasterAbsolute and SlaveAbsolute

Absolute: When absolute camming is set, the control values or the slave values based on the CamTable are interpreted as being absolute. The system compensates any offset developing between the leading and following axis during synchronization. When synchronism is reached, a defined phase relationship is established between the control value and the slave value.

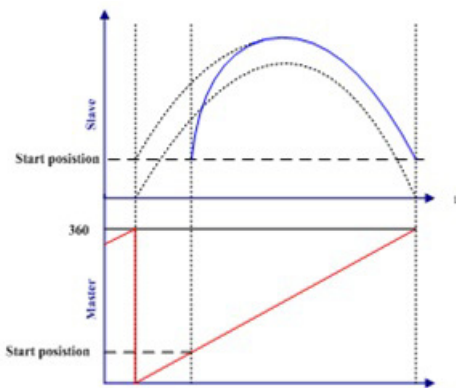
Relative: When relative camming is set, this means that any offsets between the control value and the slave value are not compensated for during synchronization.

For example, sectional drawing from Utility as follows:



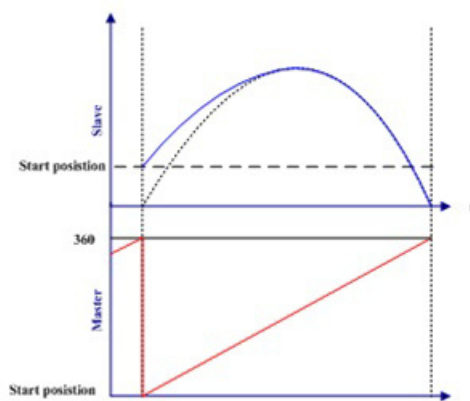
The initial cam curve is as follow:

- Master axis: Absolute. Slave axis: Relative.



Comments

- Master axis: Relative. Slave axis: Absolute.



Format Parameters			
Name	Type	IN or OUT	Description
DeviceHandle	HAND	IN	Device handle. This handle is device handle from Acm_DevOpen.
CamTableID	U32	IN	Identifier of Cam table. It is assigned by Acm_DevDownloadCAMTable. Device reserves 2 cam tables. So the ID can be 0, 1.
Periodic	U32	IN	CAM curve is executed periodic or non-periodic. 0: non-periodic, 1: periodic
MasterAbsolute	U32	IN	Interpret cam curve relative (0) or absolute (1) to the master axis. 0: relative, 1: absolute
SlaveAbsolute	U32	IN	Interpret cam curve relative (0) or absolute (1) to the slave axis. 0: relative, 1: absolute

Acm_DevLoadCAMTableFile

Format	U32 Acm_DevLoadCAMTableFile (HAND DeviceHandle, PI8 FilePath,U32 CamTableID, PU32 Range, PU32 PointsCount)
Purpose	Load Cam Table file edited and saved by Utility into device.
Return	Error code
Comments	The CamTable file is saved in .bin format in utility. When it is loaded successfully by this API, the API Acm_DevDownLoadCACMTable need not be called. About E-cam operation, see about E-Cam Flow Chart in Chapter 9.4.3.3

Format Parameters

Name	Type	IN or OUT	Description
DeviceHandle	HAND	IN	Device handle from Acm_DevOpen.
FilePath	PI8	IN	Pointer to a string that saves CamTable file's path.
CamTableID	U32	IN	CamTableID: 0 or 1.
Range	PU32	OUT	The pulse number which master needed in one period.
PointsCount	PU32	OUT	The points number in CamTable.

Acm_DevMDaqConfig

Format	U32 Acm_DevMDaqConfig (HAND DeviceHandle, U16 ChannelID,U32 Period, U32 AxisNo, U32 Method, U32 ChanType, U32 Count)
Purpose	Set MotionDAQ related configurations.
Return	Error code
Comments	

Format Parameters

Name	Type	IN or OUT	Description
DeviceHandle	HAND	IN	Device handle form Acm_DevOpen.
ChannelID	U16	IN	Specify a Channel ID. The range is 0 ~ 3.
Period	U32	IN	Set an interval time between each data. The range is 0 ~ 255 and unit is ms.
AxisNo	U32	IN	Specify an axis. The range is: 0 ~ 3 (PCI-1245/45E), 0 ~ 5 (PCI-1265), 0 ~ 31 (PCI-1203).

Method	U32	IN	<p>Methods to trigger MDAQ:</p> <p>0: Disable;</p> <p>1: Software trigger;</p> <p>2: DI trigger;</p> <p>3: Axis 0 starts to trigger;</p> <p>4: Axis 1 starts to trigger;</p> <p>5: Axis 2 starts to trigger;</p> <p>6: Axis 3 starts to trigger;</p> <p>7: Axis 4 starts to trigger;(PCI1245(E) does not support)</p> <p>8: Axis 5 starts to trigger;(PCI1245(E) does not support)</p> <p>9: Axis 6 starts to trigger;(PCI1265/PCI1245(E) does not support)</p> <p>10: Axis 7 starts to trigger;(PCI1265/PCI1245(E) does not support)</p> <p>11: Axis 8 starts to trigger;(PCI1265/PCI1245(E) does not support)</p> <p>12: Axis 9 starts to trigger;(PCI1265/PCI1245(E) does not support)</p> <p>13: Axis 10 starts to trigger;(PCI1265/PCI1245(E) does not support)</p> <p>14: Axis 11 starts to trigger;(PCI1265/PCI1245(E) does not support)</p>
ChanType	U32	IN	<p>Get Channel Type:</p> <p>0: Command Position;</p> <p>1: Actual Position;</p> <p>2: Lag Position (Difference value between Command Position and Actual Position);</p> <p>3: Command Velocity (PCI1265/PCI1245(E) does not support)</p>
Count	U32	IN	Specify a data count. The range is 0 ~ 2000.

Acm_DevMDaqGetConfig

Format	U32 Acm_DevMDaqGetConfig (HAND DeviceHandle, U16 ChannelID, PU32 Period, PU32 AxisNo, PU32 Method, PU32 ChanType, PU32 Count)		
Purpose	Get MotionDAQ related values of a specified ChannelID.		
Return	Error code		
Comments			
Format Parameters			
Name	Type	IN or OUT	Description
DeviceHandle	HAND	IN	Device handle form Acm_DevOpen.
ChannelID	U16	IN	Specify to a MDAQ Channel to get the configurations. The range is 0 ~ 3.
Period	PU32	OUT	Get interval time between each data. The range is 0 ~ 255 and the unit is ms.
AxisNo	PU32	OUT	Get related information of an axis. The range is 0 ~ 3 (PCI-1245/45E), 0 ~ 5 (PCI-1265), 0 ~ 31 (PCI-1203).

Method	PU32	OUT	Methods to trigger MDAQ: 0: Disable; 1: Software trigger; 2: DI trigger; 3: Axis 0 starts to trigger; 4: Axis 1 starts to trigger; 5: Axis 2 starts to trigger; 6: Axis 3 starts to trigger; 7: Axis 4 starts to trigger; 8: Axis 5 starts to trigger; 9: Axis 6 starts to trigger; 10: Axis 7 starts to trigger; 11: Axis 8 starts to trigger; 12: Axis 9 starts to trigger; 13: Axis 10 starts to trigger; 14: Axis 11 starts to trigger.
ChanType	PU32	OUT	Get Channel Type: 0: Command Position; 1: Actual Position; 2: Lag Position (Difference value between Command Position and Actual Position); 3: Command Velocity.
Count	PU32	OUT	Get the max count. The range is 0 ~ 2000.

Acm_DevMDaqStart

Format	Acm_DevMDaqStart(HAND DeviceHandle)		
Purpose	Enable MotionDAQ function to record related data.		
Return	Error code		
Comments	When Method in Acm_DevMDaqConfig is set to MQ_TRIG_SW, command can be sent through this API to trigger MotionDAQ function.		

Format Parameters

Name	Type	IN or OUT	Description
DeviceHandle	HAND	IN	Device handle form Acm_DevOpen.

Acm_DevMDaqStop

Format	U32 Acm_DevMDaqStop (HAND DeviceHandle)		
Purpose	Stop recording MotionDAQ related data.		
Return	Error code		
Comments			

Format Parameters

Name	Type	IN or OUT	Description
DeviceHandle	HAND	IN	Device handle form Acm_DevOpen.

Acm_DevMDaqReset

Format	U32 Acm_DevMDaqReset (HAND DeviceHandle, U16 ChannelID)		
Purpose	Clear MotionDAQ related data of corresponding ChannelID.		
Return	Error code		
Comments			

Format Parameters

Name	Type	IN or OUT	Description
DeviceHandle	HAND	IN	Device handle form Acm_DevOpen.
ChannelID	U16	IN	Channel ID. The range is 0 ~ 3.

Acm_DevMDaqGetStatus

Format	U32 Acm_DevMDaqGetStatus(HAND DeviceHandle, U16 ChannelID, PU16 CurrentCnt, PU8 Status)		
Purpose	Get MotionDAQ status of corresponding ChannelID.		
Return	Error code		
Comments			
Format Parameters			
Name	Type	IN or OUT	Description
DeviceHandle	HAND	IN	Device handle form Acm_DevOpen.
ChannelID	U16	IN	Channel ID. The range is 0 ~ 3.
CurrentCnt	PU16	OUT	Return current count that has been recorded.
Status	PU8	OUT	Return current status: 0: Ready; 1: Waiting Trigger; 2: Started.

Acm_DevMDaqGetData

Format	U32 Acm_DevMDaqGetData(HAND DeviceHandle, U16 ChannelID, U16 StartIndex, U16 MaxCount, PI32 DataBuffer)		
Purpose	Get all data in the range specified by MotionDAQ that has been recorded by corresponding channel.		
Return	Error code		
Comments			
Format Parameters			
Name	Type	IN or OUT	Description
DeviceHandle	HAND	IN	Device handle form Acm_DevOpen.
ChannelID	U16	IN	Channel ID. The range is 0 ~ 3.
StartIndex	U16	IN	Set start position to get data.
MaxCount	U16	IN	Set data count.
DataBuffer	PI32	OUT	Return data in the specified range.

A.3 DAQ

A.3.1 Digital Input/ Output

A.3.1.1 Acm_DaqDiGetByte

Format	U32 Acm_DaqDiGetByte (HAND DeviceHandle, U16 DiPort, PU8 ByteData)		
Purpose	Get the data of specified DI port in one byte.		
Return	Error code		
Comments	<p>Get the DI value according to the specified Port Number. User should check or set the Port number and ChannelID in whole network through mapping Info in Utility before using this API to get the property. The DI value will be affected by CFG_CH_DaqDiInvertEnable, CFG_CH_DaqDiLowFilter, CFG_CH_DaqDiHighFilter. Confirm that the property value is correct first. (Please refer to the Property chapter)</p> <p>Note:</p> <ol style="list-style-type: none">(1). DI port number must be consistent to the mapping info in Utility.(2). User need call Acm_DevLoadMapFile(If user uses the default mapping relationship, this API need not to load) first before calling Acm_DaqDiGetByte.		
Format Parameters			
Name	Type	IN or OUT	Description
DeviceHandle	HAND	IN	Device handle form Acm_DevOpen.
DiPort	U16	IN	DI port number.
ByteData	PU8	OUT	Pointer of returned byte data.

A.3.1.2 Acm_DaqDiGetBit

Format	U32 Acm_DaqDiGetBit (HAND DeviceHandle, U16 DiChannel, PU8 Bit-Data)		
Purpose	Get the bit data of specified DI channel.		
Return	Error code		
Comments	<p>Get the DI value according to the specified Port Number and ChannelID This channelID can be getting by calculation: ChannelID= Port number * 8 + Port Index. Take the second DI channel in Port 3 for an example, the ChannelID= 3*8+2=26. User should check or set the Port number and ChannelID in whole network through mapping Info in Utility before using this API to get the property. The DI value will be affected by CFG_CH_DaqDiInvertEnable, CFG_CH_DaqDiLowFilter, CFG_CH_DaqDiHighFilter. Confirm that the property value is correct first. (Please refer to the Property chapter) Note! User need call Acm_DevLoadMapFile(If user uses the default mapping relationship, this API need not to load) first before calling Acm_DaqDiGetBit.</p>		
Format Parameters			
Name	Type	IN or OUT	Description
DeviceHandle	HAND	IN	Device handle form Acm_DevOpen.
DiChannel	U16	IN	DI channel ID Range:0~Max DI count -1
BitData	PU8	OUT	Returned bit data. 0 or 1.

A.3.1.3 Acm_DaqDiGetBytes

Format	U32 Acm_DaqDiGetBytes (HAND DeviceHandle, U16 StartPort, U16 NumPort, PU8 ByteDataArray)		
Purpose	Get the data of DI ports from StartPort to (StartPort + NumPort – 1).		
Return	Error code		
Comments	<p>Get the continuous DI byte value according to the specified Port Number. User should check or set the Port number and ChannelID in whole network through mapping Info in Utility before using this API to get the property. The DI value will be affected by CFG_CH_DaqDiInvertEnable, CFG_CH_DaqDiLowFilter, CFG_CH_DaqDiHighFilter. Confirm that the property value is correct first. (Please refer to the Property chapter)</p> <p>Note:</p> <ol style="list-style-type: none"> (1). (StartPort + NumPort) must less than the total port number. (2). User need call Acm_DevLoadMapFile(If user uses the default mapping relationship, this API need not to load) first before calling Acm_DaqDiGetBytes. 		
Format Parameters			
Name	Type	IN or OUT	Description
DeviceHandle	HAND	IN	Device handle form Acm_DevOpen.
StartPort	U16	IN	First port of DI.
NumPort	U16	IN	Number of DI port to get.
ByteDataArray	PU8	OUT	Pointer of returned bytes data.

A.3.1.4 Acm_DaqDoSetByte

Format	U32 Acm_DaqDoSetByte (HAND DeviceHandle, U16 DoPort, U8 ByteData)		
Purpose	Set value to specified DO port.		
Return	Error code		
Comments	<p>Get the DO value according to the specified Port Number. User should check or set the Port number and ChannelID in whole network through mapping Info in Utility before using this API to get the property. The DO value will be affected by CFG_CH_DaqDoFsvEnable. Confirm that the property value is correct first. (Please refer to the Property chapter)</p> <p>Note:</p> <p>User need call Acm_DevLoadMapFile(If user uses the default mapping relationship, this API need not to load) first before calling Acm_DaqDoSetByte.</p>		
Format Parameters			
Name	Type	IN or OUT	Description
DeviceHandle	HAND	IN	Device handle form Acm_DevOpen.
DoPort	U16	IN	DO port.
ByteData	U8	IN	Value to be set.

A.3.1.5 Acm_DaqDoSetBit

Format	U32 Acm_DaqDoSetBit (HAND DeviceHandle, U16 DoChannel, U8 BitData)		
Purpose	Set the value to specified DO channel.		
Return	Error code		
Comments	<p>Get the DO value according to the specified Port Number and ChannelID This channelID can be got by calculation: ChannelID= Port number * 8 + Port Index. Take the second DO channel in Port 3 for an example, the ChannelID= 3*8+2=26. User should check or set the Port number and ChannelID in whole network through mapping Info in Utility before using this API to get the property. The DO value will be affected by CFG_CH_DaqDoFsvEnable. Confirm that the property value is correct first. (Please refer to the Property chapter)</p> <p>Note: User need call Acm_DevLoadMapFile(If user uses the default mapping relationship, this API need not to load) first before calling Acm_DaqDoSetBit.</p>		
Format Parameters			
Name	Type	IN or OUT	Description
DeviceHandle	HAND	IN	Device handle form Acm_DevOpen.
DoChannel	U16	IN	DO channel ID Range:0~Max DO count -1
BitData	U8	IN	Value to be set. 0 or 1.

A.3.1.6 Acm_DaqDoSetBytes

Format	U32 Acm_DaqDoSetBytes (HAND DeviceHandle, U16 StartPort, U16 NumPort, PU8 ByteDataArray)		
Purpose	Set the data of DO ports from StartPort to (StartPort + NumPort – 1).		
Return	Error code		
Comments	<p>Set the continuous DO byte value according to the specified Port Number. User should check or set the Port number and ChannelID in whole network through mapping Info in Utility before using this API to get the property. The DO value will be affected by CFG_CH_DaqDoFsvEnable. Confirm that the property value is correct first. (Please refer to the Property chapter)</p> <p>Note:</p> <ol style="list-style-type: none"> (1). (StartPort + NumPort) must less than the total port number. (2). User need call Acm_DevLoadMapFile(If user uses the default mapping relationship, this API need not to load) first before calling Acm_DaqDoSetBytes. 		
Format Parameters			
Name	Type	IN or OUT	Description
DeviceHandle	HAND	IN	Device handle form Acm_DevOpen.
StartPort	U16	IN	First port of DO.
NumPort	U16	IN	Number of DO port to set.
ByteDataArray	PU8	IN	Value to be set.

A.3.1.7 Acm_DaqDoGetByte

Format	U32 Acm_DaqDoGetByte(HAND DeviceHandle, U16 DoPort, PU8 Byte-Data)		
Purpose	Get the byte data of specified DO port.		
Return	Error code		
Comments			
Format Parameters			
Name	Type	IN or OUT	Description
DeviceHandle	HAND	IN	Device handle form Acm_DevOpen.
DoPort	U16	IN	DO port.
ByteData	PU8	Out	Returned value.

A.3.1.8 Acm_DaqDoGetBit

Format	U32 Acm_DaqDoGetBit (HAND DeviceHandle, U16 DoChannel, PU8 Bit-Data)		
Purpose	Get the bit value of specified DO channel.		
Return	Error code		
Comments	<p>Get the DO value according to the specified ChannelID This channelID can be got by calculation: ChannelID= Port number * 8 + Port Index. Take the second DO channel in Port 3 for an example, the ChannelID= 3*8+2=26.</p> <p>User should check or set the Port number and ChannelID in whole network through mapping Info in Utility before using this API to get the property. The DO value will be affected by CFG_CH_DaqDoFsvEnable. Confirm that the property value is correct first. (Please refer to the Property chapter)</p> <p>Note: User need call Acm_DevLoadMapFile(If user uses the default mapping relationship, this API need not to load) first before calling Acm_DaqDoGetBit.</p>		
Format Parameters			
Name	Type	IN or OUT	Description
DeviceHandle	HAND	IN	Device handle form Acm_DevOpen.
DoChannel	U16	IN	DO channel ID Range:0~Max DO count -1
BitData	PU8	Out	Returned value 0 or 1.

A.3.1.9 Acm_DaqDoGetBytes

Format	U32 Acm_DaqDoGetBytes (HAND DeviceHandle, U16 StartPort, U16 NumPort, PU8 BytedataArray)		
Purpose	Get the data of DO ports from StartPort to (StartPort + NumPort – 1).		
Return	Error code		
Comments	<p>Get the continuous DO byte value according to the specified Port Number. User should check or set the Port number and ChannelID in whole network through mapping Info in Utility before using this API to get the property. The DO value will be affected by CFG_CH_DaqDoFsvEnable. Confirm that the property value is correct first. (Please refer to the Property chapter)</p> <p>Note:</p> <ol style="list-style-type: none"> (1). (StartPort + NumPort) must less than the total port number. (2). User need call Acm_DevLoadMapFile(If user uses the default mapping relationship, this API need not to load) first before calling Acm_DaqDoGetBytes. 		
Format Parameters			
Name	Type	IN or OUT	Description
DeviceHandle	HAND	IN	Device handle form Acm_DevOpen.
StartPort	U16	IN	First port of DO.
NumPort	U16	IN	Number of DO port to get.
BytedataArray	PU8	OUT	Pointer of returned bytes data.

A.3.2 Analog Input/ Output

A.3.2.1 Acm_DaqAiGetRawData

Format	U32 Acm_DaqAiGetRawData(HAND DeviceHandle, U16 AiChannel,PU16 AiData)		
Purpose	Get the binary value of an analog input channel		
Return	Error code		
Comments	<p>Get the AI binary value according to the specified ChannelID. User should check or set the Port number and ChannelID in whole network through mapping Info in Utility before using this API to get the property. The AI value will be affected by CFG_CH_DaqAiRange, CFG_CH_DaqAiEnable, CFG_CH_DaqAiIntegrationTime. Confirm that the property value is correct first. (Please refer to the Property chapter)</p> <p>Note:</p> <p>User need call Acm_DevLoadMapFile(If user uses the default mapping relationship, this API need not to load) first before calling Acm_DaqAiGetRawData.</p>		
Format Parameters			
Name	Type	IN or OUT	Description
DeviceHandle	HAND	IN	Device handle form Acm_DevOpen.
AiChannel	U16	IN	AI channel ID Range:0~Max AI count -1
AiData	PU16	Out	Pointer to the returned AI binary value

A.3.2.2 Acm_DaqAiGetVoltData

Format	U32 Acm_DaqAiGetVoltData(HAND DeviceHandle, U16 AiChannel, PF32 AiData)		
Purpose	Get the voltage value of an analog input channel when voltage inputs		
Return	Error code		
Comments	<p>Get the AI voltage value according to the specified ChannelID User should check or set the Port number and ChannelID in whole network through mapping Info in Utility before using this API to get the property. As the AI channel is set to voltage level by CFG_CH_DaqAiRange, user can get the input voltage value; Error will occur when calling this API as the AI channel is set to current level.</p> <p>The AI value will be affected by CFG_CH_DaqAiRange, CFG_CH_DaqAiEnable, CFG_CH_DaqAiIntegrationTime. Confirm that the property value is correct first. (Please refer to the Property chapter)</p> <p>Note: User need call Acm_DevLoadMapFile(If user uses the default mapping relationship, this API need not to load) first before calling Acm_DaqAiGetVoltData.</p>		
Format Parameters			
Name	Type	IN or OUT	Description
DeviceHandle	HAND	IN	Device handle form Acm_DevOpen.
AiChannel	U16	IN	AI channel ID Range:0~Max AI count -1
AiData	PF32	Out	Pointer to the returned AI voltage value

A.3.2.3 Acm_DaqAiGetCurrData

Format	U32 Acm_DaqAiGetCurrData(HAND DeviceHandle, U16 AiChannel, PF32 AiData)		
Purpose	Get the current value of an analog input channel when current inputs		
Return	Error code		
Comments	<p>Get the AI current value according to the specified ChannelID. User should check or set the Port number and ChannelID in whole network through mapping Info in Utility before using this API to get the property. As the AI channel is set to current level by CFG_CH_DaqAiRange, user can get the input current value; Error will occur when calling this API as the AI channel is set to voltage level.</p> <p>The AI value will be affected by CFG_CH_DaqAiRange, CFG_CH_DaqAiEnable, CFG_CH_DaqAiIntegrationTime. Confirm that the property value is correct first. (Please refer to the Property chapter)</p> <p>Note: User need call Acm_DevLoadMapFile(If user uses the default mapping relationship, this API need not to load) first before calling Acm_DaqAiGetCurrData.</p>		
Format Parameters			
Name	Type	IN or OUT	Description
DeviceHandle	HAND	IN	Device handle form Acm_DevOpen.
AiChannel	U16	IN	AI channel ID Range:0~Max AI count -1
AiData	PF32	Out	Pointer to the returned AI current value

A.3.2.4 Acm_DaqAoSetRawData

Format	U32 Acm_DaqAoSetRawData (HAND DeviceHandle, U16 AoChannel, U16 AoData)		
Purpose	Set the binary value of an analog output channel		
Return	Error code		
Comments	<p>Set the AO binary value according to the specified ChannelID. User should check or set the Port number and ChannelID in whole network through mapping Info in Utility before using this API to get the property. The AO value will be affected by CFG_CH_DaqAoRange. Confirm that the property value is correct first. (Please refer to the Property chapter)</p> <p>Note: User need call Acm_DevLoadMapFile (If user uses the default mapping relationship, this API need not to load) first before calling Acm_DaqAoSetRawData.</p>		
Format Parameters			
Name	Type	IN or OUT	Description
DeviceHandle	HAND	IN	Device handle form Acm_DevOpen.
AoChannel	U16	IN	AO channel ID Range:0~Max AO count -1
AoData	U16	IN	AO binary value

A.3.2.5 Acm_DaqAoSetVoltData

Format	U32 Acm_DaqAoSetVoltData (HAND DeviceHandle, U16 AoChannel, F32 AoData)		
Purpose	Set the voltage output value of a specified analog output channel within the analog voltage output range.		
Return	Error code		
Comments	<p>Set the AO voltage value according to the specified ChannelID. User should check or set the Port number and ChannelID in whole network through mapping Info in Utility before using this API to get the property. As the AO channel is set to voltage level by CFG_CH_DaqAoRange, user can set the output voltage value; Error will occur when calling this API as the AO channel is set to current level. The AO value will be affected by CFG_CH_DaqAoRange. Confirm that the property value is correct first. (Please refer to the Property chapter)</p> <p>Note: User need call Acm_DevLoadMapFile (If user uses the default mapping relationship, this API need not to load) first before calling Acm_DaqAoSetVoltData.</p>		
Format Parameters			
Name	Type	IN or OUT	Description
DeviceHandle	HAND	IN	Device handle form Acm_DevOpen.
AoChannel	U16	IN	AO channel ID Range:0~Max AO count -1
AoData	F32	IN	AO voltage value

A.3.2.6 Acm_DaqAoSetCurrData

Format	U32 Acm_DaqAoSetCurrData (HAND DeviceHandle, U16 AoChannel, F32 AoData)		
Purpose	Set the current output value of a specified analog output channel within the analog current output range.		
Return	Error code		
Comments	<p>Set the AO current value according to the specified ChannelID. User should check or set the Port number and ChannelID in whole network through mapping Info in Utility before using this API to get the property. As the AO channel is set to current level by CFG_CH_DaqAoRange, user can set the output current value; Error will occur when calling this API as the AO channel is set to voltage level.</p> <p>The AO value will be affected by CFG_CH_DaqAoRange. Confirm that the property value is correct first. (Please refer to the Property chapter)</p> <p>Note: User need call Acm_DevLoadMapFile(If user uses the default mapping relationship, this API need not to load) first before calling Acm_DaqAoSetCurrData.</p>		
Format Parameters			
Name	Type	IN or OUT	Description
DeviceHandle	HAND	IN	Device handle form Acm_DevOpen.
AoChannel	U16	IN	AO channel ID Range:0~Max AO count -1
AoData	F32	IN	AO current value

A.3.2.7 Acm_DaqAoGetRawData

Format	U32 Acm_DaqAoGetRawData (HAND DeviceHandle, U16 AoChannel, PU16 AoData)		
Purpose	Get the binary value of an analog output channel.		
Return	Error code		
Comments	<p>Get the AO binary value according to the specified ChannelID. User should check or set the Port number and ChannelID in whole network through mapping Info in Utility before using this API to get the property. The AO value will be affected by CFG_CH_DaqAoRange. Confirm that the property value is correct first. (Please refer to the Property chapter)</p> <p>Note: User need call Acm_DevLoadMapFile(If user uses the default mapping relationship, this API need not to load) first before calling Acm_DaqAoGetRawData.</p>		
Format Parameters			
Name	Type	IN or OUT	Description
DeviceHandle	HAND	IN	Device handle form Acm_DevOpen.
AoChannel	U16	IN	AO channel ID Range:0~Max AO count -1
AoData	PU16	OUT	Pointer to the returned AO binary value

A.3.2.8 Acm_DaqAoGetVoltData

Format	U32 Acm_DaqAoGetVoltData (HAND DeviceHandle, U16 AoChannel, PF32 AoData)		
Purpose	Get the voltage output value of a specified analog output channel within the analog voltage output range.		
Return	Error code		
Comments	<p>Get the AO voltage value according to the specified ChannelID User should check or set the Port number and ChannelID in whole network through mapping Info in Utility before using this API to get the property. As the AO channel is set to voltage level by CFG_CH_DaqAoRange, user can set the output voltage value; Error will occur when calling this API as the AO channel is set to current level. The AO value will be affected by CFG_CH_DaqAoRange. Confirm that the property value is correct first. (Please refer to the Property chapter)</p> <p>Note: User need call Acm_DevLoadMapFile(If user uses the default mapping relationship, this API need not to load) first before calling DaqAoGetVoltData.</p>		
Format Parameters			
Name	Type	IN or OUT	Description
DeviceHandle	HAND	IN	Device handle form Acm_DevOpen.
AoChannel	U16	IN	AO channel ID Range:0~Max AO count -1
AoData	F32	OUT	Pointer to the AO voltage value

A.3.2.9 Acm_DaqAoGetCurrData

Format	U32 Acm_DaqAoGetCurrData (HAND DeviceHandle, U16 AoChannel, PF32 AoData)		
Purpose	Get the current output value of a specified analog output channel within the analog current output range.		
Return	Error code		
Comments	<p>Get the AO current value according to the specified ChannelID User should check or set the Port number and ChannelID in whole network through mapping Info in Utility before using this API to get the property. As the AO channel is set to current level by CFG_CH_DaqAoRange, user can set the output current value; Error will occur when calling this API as the AO channel is set to voltage level. The AO value will be affected by CFG_CH_DaqAoRange. Confirm that the property value is correct first. (Please refer to the Property chapter)</p> <p>Note: User need call Acm_DevLoadMapFile(If user uses the default mapping relationship, this API need not to load) first before calling Acm_DaqAoGetCurrData.</p>		
Format Parameters			
Name	Type	IN or OUT	Description
DeviceHandle	HAND	IN	Device handle form Acm_DevOpen.
AoChannel	U16	IN	AO channel ID Range:0~Max AO count -1
AoData	PF32	OUT	Pointer to the AO current value

A.4 Axis

A.4.1 System

A.4.1.1 Acm_AxOpen

Format	U32 Acm_AxOpen (HAND DeviceHandle, U16 PhyAxis, PHAND AxisHandle)		
Purpose	Open specified axis and get this axis object's handle.		
Return	Error code		
Comments	Before any axis operation, this API should be called firstly. The physical axis-number in PCI-1203: 0, 1, 2...31.		
Format Parameters			
Name	Type	IN or OUT	Description
DeviceHandle	HAND	IN	Device handle form Acm_DevOpen.
PhyAxis	U16	IN	Physical Axis Number.
AxisHandle	PHAND	Out	Returns a pointer to the axis handle.

A.4.1.2 Acm_AxOpenbyID

Format	U32 Acm_AxOpenbyID (HAND DeviceHandle, U16 SlaveIP, PHAND AxisHandle)		
Purpose	Open specified axis and get this axis object's handle.		
Return	Error code		
Comments	Only PCI-1203 supported this function. Before any axis operation, this API should be called firstly. User can specify the driver ID to open the axis and get the axis handle.		
Format Parameters			
Name	Type	IN or OUT	Description
DeviceHandle	HAND	IN	Device handle form Acm_DevOpen.
SlaveIP	U16	IN	Driver ID.
AxisHandle	PHAND	Out	Returns a pointer to the axis handle.

A.4.1.3 Acm_AxClose

Format	U32 Acm_AxClose (PHAND AxisHandle)		
Purpose	Close axis which has been opened.		
Return	Error code		
Comments	After calling this API, the axis handle cannot be used again.		
Format Parameters			
Name	Type	IN or OUT	Description
AxisHandle	PHAND	IN	Pointer to the axis handle

A.4.1.4 Acm_AxResetError

Format	U32 Acm_AxResetError (HAND AxisHandle)		
Purpose	Reset the axis' state. If the axis is in ErrorStop state, the state will be changed to Ready after calling this function.		
Return	Error code		
Comments			
Format Parameters			
Name	Type	IN or OUT	Description
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen.

A.4.2 Motion IO

A.4.2.1 Acm_AxSetSvOn

Format	U32 Acm_AxSetSvOn (HAND AxisHandle, U32 OnOff)		
Purpose	Set servo Driver ON or OFF.		
Return	Error code		
Comments			
Format Parameters			
Name	Type	IN or OUT	Description
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen.
OnOff	U32	IN	Setting the action of SVON signal. 0: Off; 1: On

A.4.2.2 Acm_AxGetMotionIO

Format	U32 Acm_AxGetMotionIO (HAND AxisHandle, PU32 Status)																																												
Purpose	Get the motion I/O status of the axis.																																												
Return	Error code																																												
Comments																																													
Format Parameters																																													
Name	Type	IN or OUT	Description																																										
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen.																																										
Status	PU32	OUT	<table border="1"> <thead> <tr> <th>Bit</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0:RDY</td> <td>RDY pin input</td> </tr> <tr> <td>1:ALM</td> <td>Alarm Signal input</td> </tr> <tr> <td>2:LMT+</td> <td>Limit Switch+</td> </tr> <tr> <td>3:LMT-</td> <td>Limit Switch-</td> </tr> <tr> <td>4:ORG</td> <td>Origin Switch</td> </tr> <tr> <td>5:DIR</td> <td>DIR output</td> </tr> <tr> <td>6:EMG</td> <td>Emergency signal input</td> </tr> <tr> <td>7:PCS</td> <td>PCS signal</td> </tr> <tr> <td>8:ERC</td> <td>Output deflection counter clear signal to a servomotor driver; (OUT7)</td> </tr> <tr> <td>9:EZ</td> <td>Encoder Z signal</td> </tr> <tr> <td>10:CLR</td> <td>ext. input to Clear position counter</td> </tr> <tr> <td>11:LTC</td> <td>Latch signal input</td> </tr> <tr> <td>12:SD</td> <td>Slow Down signal input;</td> </tr> <tr> <td>13:INP</td> <td>In-Position signal input</td> </tr> <tr> <td>14:SVON</td> <td>Servo-ON (OUT6)</td> </tr> <tr> <td>15:ALRM</td> <td>Alarm Reset output status</td> </tr> <tr> <td>16:SLMT+</td> <td>Software Limit+</td> </tr> <tr> <td>17:SLMT-</td> <td>Software Limit-</td> </tr> <tr> <td>18:CMP</td> <td>Compare signal(OUT5)</td> </tr> <tr> <td>19:CAMDO</td> <td>position window do(OUT4)</td> </tr> </tbody> </table>	Bit	Description	0:RDY	RDY pin input	1:ALM	Alarm Signal input	2:LMT+	Limit Switch+	3:LMT-	Limit Switch-	4:ORG	Origin Switch	5:DIR	DIR output	6:EMG	Emergency signal input	7:PCS	PCS signal	8:ERC	Output deflection counter clear signal to a servomotor driver; (OUT7)	9:EZ	Encoder Z signal	10:CLR	ext. input to Clear position counter	11:LTC	Latch signal input	12:SD	Slow Down signal input;	13:INP	In-Position signal input	14:SVON	Servo-ON (OUT6)	15:ALRM	Alarm Reset output status	16:SLMT+	Software Limit+	17:SLMT-	Software Limit-	18:CMP	Compare signal(OUT5)	19:CAMDO	position window do(OUT4)
Bit	Description																																												
0:RDY	RDY pin input																																												
1:ALM	Alarm Signal input																																												
2:LMT+	Limit Switch+																																												
3:LMT-	Limit Switch-																																												
4:ORG	Origin Switch																																												
5:DIR	DIR output																																												
6:EMG	Emergency signal input																																												
7:PCS	PCS signal																																												
8:ERC	Output deflection counter clear signal to a servomotor driver; (OUT7)																																												
9:EZ	Encoder Z signal																																												
10:CLR	ext. input to Clear position counter																																												
11:LTC	Latch signal input																																												
12:SD	Slow Down signal input;																																												
13:INP	In-Position signal input																																												
14:SVON	Servo-ON (OUT6)																																												
15:ALRM	Alarm Reset output status																																												
16:SLMT+	Software Limit+																																												
17:SLMT-	Software Limit-																																												
18:CMP	Compare signal(OUT5)																																												
19:CAMDO	position window do(OUT4)																																												

A.4.3 Motion Status

A.4.3.1 Acm_AxGetMotionStatus

Format	U32 Acm_AxGetMotionStatus (HAND AxisHandle, PU32 Status)																												
Purpose	Get current motions status of the axis.																												
Return	Error code																												
Comments																													
Format Parameters																													
Name	Type	IN or OUT	Description																										
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen.																										
Status	PU32	OUT	<table border="1"> <thead> <tr> <th>Bit</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0: Stop</td> <td>Stop</td> </tr> <tr> <td>1: Res1</td> <td>Reserved</td> </tr> <tr> <td>2: WaitERC</td> <td>Wait ERC finished</td> </tr> <tr> <td>3: Res2</td> <td>Reserved</td> </tr> <tr> <td>4: CorrectBksh</td> <td>Correcting Backlash</td> </tr> <tr> <td>5: Res3</td> <td>Reserved</td> </tr> <tr> <td>6: InFA</td> <td>Feeding in return velocity = FA</td> </tr> <tr> <td>7: InFL</td> <td>Feeding in StrVel speed =FL</td> </tr> <tr> <td>8: InACC</td> <td>Accelerating</td> </tr> <tr> <td>9: InFH</td> <td>Feeding in MaxVel speed = FH</td> </tr> <tr> <td>10: InDEC</td> <td>Decelerating</td> </tr> <tr> <td>11:WaitINP</td> <td>Wait in position</td> </tr> </tbody> </table>	Bit	Description	0: Stop	Stop	1: Res1	Reserved	2: WaitERC	Wait ERC finished	3: Res2	Reserved	4: CorrectBksh	Correcting Backlash	5: Res3	Reserved	6: InFA	Feeding in return velocity = FA	7: InFL	Feeding in StrVel speed =FL	8: InACC	Accelerating	9: InFH	Feeding in MaxVel speed = FH	10: InDEC	Decelerating	11:WaitINP	Wait in position
Bit	Description																												
0: Stop	Stop																												
1: Res1	Reserved																												
2: WaitERC	Wait ERC finished																												
3: Res2	Reserved																												
4: CorrectBksh	Correcting Backlash																												
5: Res3	Reserved																												
6: InFA	Feeding in return velocity = FA																												
7: InFL	Feeding in StrVel speed =FL																												
8: InACC	Accelerating																												
9: InFH	Feeding in MaxVel speed = FH																												
10: InDEC	Decelerating																												
11:WaitINP	Wait in position																												

A.4.3.2 Acm_AxGetState

Format	U32 Acm_AxGetState (HAND AxisHandle, PU16 State)																								
Purpose	Get the axis's current state.																								
Return	Error code																								
Comments																									
Format Parameters																									
Name	Type	IN or OUT	Description																						
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen.																						
			<table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0: STA_AxDisable</td> <td>Axis is disabled, you can open it to active it.</td> </tr> <tr> <td>1: STA_AxReady</td> <td>Axis is ready and waiting for new command</td> </tr> <tr> <td>2: STA_Stopping</td> <td>Axis is stopping</td> </tr> <tr> <td>3: STA_AxErrorStop</td> <td>Axis has stopped because of error.</td> </tr> <tr> <td>4: STA_AxHoming</td> <td>Axis is executing home motion.</td> </tr> <tr> <td>5: STA_AxPtpMotion</td> <td>Axis is executing PTP motion.</td> </tr> <tr> <td>6: STA_AxContiMotion</td> <td>Axis is executing continuous motion.</td> </tr> <tr> <td>7: STA_AxSyncMotion</td> <td>Axis is in one group and the group is executing interpolation motion, or axis is slave axis in E-cam/Egear/Gantry motion.</td> </tr> <tr> <td>8: STA_AX_EXT_JOG</td> <td>Axis is controlled by external signal and will execute JOG mode motion once external signal is active.</td> </tr> <tr> <td>9: STA_AX_EXT_MPG</td> <td>Axis is controlled by external signal and will execute MPG mode motion once external signal is active.</td> </tr> </tbody> </table>	Value	Description	0: STA_AxDisable	Axis is disabled, you can open it to active it.	1: STA_AxReady	Axis is ready and waiting for new command	2: STA_Stopping	Axis is stopping	3: STA_AxErrorStop	Axis has stopped because of error.	4: STA_AxHoming	Axis is executing home motion.	5: STA_AxPtpMotion	Axis is executing PTP motion.	6: STA_AxContiMotion	Axis is executing continuous motion.	7: STA_AxSyncMotion	Axis is in one group and the group is executing interpolation motion, or axis is slave axis in E-cam/Egear/Gantry motion.	8: STA_AX_EXT_JOG	Axis is controlled by external signal and will execute JOG mode motion once external signal is active.	9: STA_AX_EXT_MPG	Axis is controlled by external signal and will execute MPG mode motion once external signal is active.
Value	Description																								
0: STA_AxDisable	Axis is disabled, you can open it to active it.																								
1: STA_AxReady	Axis is ready and waiting for new command																								
2: STA_Stopping	Axis is stopping																								
3: STA_AxErrorStop	Axis has stopped because of error.																								
4: STA_AxHoming	Axis is executing home motion.																								
5: STA_AxPtpMotion	Axis is executing PTP motion.																								
6: STA_AxContiMotion	Axis is executing continuous motion.																								
7: STA_AxSyncMotion	Axis is in one group and the group is executing interpolation motion, or axis is slave axis in E-cam/Egear/Gantry motion.																								
8: STA_AX_EXT_JOG	Axis is controlled by external signal and will execute JOG mode motion once external signal is active.																								
9: STA_AX_EXT_MPG	Axis is controlled by external signal and will execute MPG mode motion once external signal is active.																								
Status	PU16	IN																							

A.4.4 Velocity Motion

A.4.4.1 Acm_AxMoveVel

Format	U32 Acm_AxMoveVel (HAND AxisHandle, U16 Direction)		
Purpose	To command axis to make a never ending movement with a specified velocity.		
Return	Error code		
Comments	The speed curve is decided by properties: PAR_AxVelLow, PAR_AxVelHigh, PAR_AxAcc, PAR_AxDec, and PAR_AxJerk.		
Format Parameters			
Name	Type	IN or OUT	Description
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen.
Direction	U16	IN	Direction: 0: Positive direction; 1: Negative direction.

A.4.4.2 Acm_AxChangeVel

Format	U32 Acm_AxChangeVel (HAND AxisHandle, F64 NewVelocity)		
Purpose	To command axis to change the velocity while axis is in velocity motion.		
Return	Error code		
Comments	The speed curve is decided by properties: PAR_AxVelLow, NewVelocity, PAR_AxAcc, PAR_AxDec, and PAR_AxJerk. The range of NewVelocity is: 0 ~ CFG_AxMaxVel. If this command runs successfully, then NewVelocity will be used in next motion in case the velocity is not specified before the motion.		
Format Parameters			
Name	Type	IN or OUT	Description
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen.
NewVelocity	F64	IN	New velocity. (unit = PPU/s)

A.4.4.3 Acm_AxGetCmdVelocity

Format	U32 Acm_AxGetCmdVelocity (HAND AxisHandle, PF64 Velocity)		
Purpose	Get current command velocity of the specified axis.		
Return	Error code		
Comments			
Format Parameters			
Name	Type	IN or OUT	Description
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen.
Velocity	PF64	OUT	Return the command velocity. (unit = PPU/s)

A.4.4.4 Acm_AxChangeVelEx

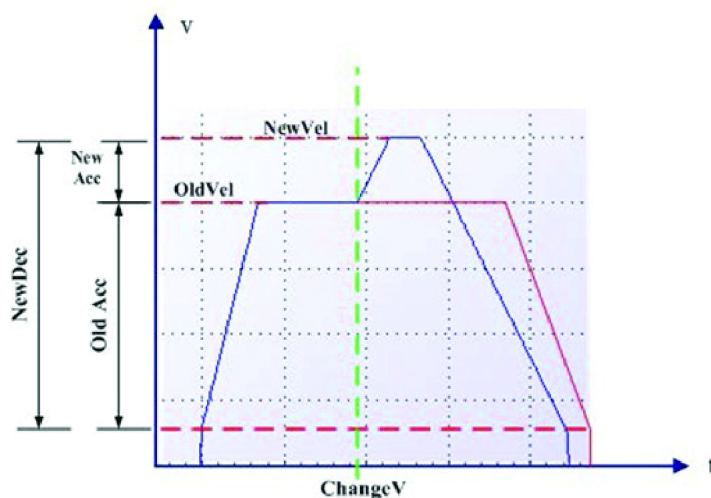
Format	U32 Acm_AxChangeVelEx (HAND AxisHandle, F64 NewVelocity, F64 NewAcc, F64 NewDec)
Purpose	Change the velocity, acceleration and deceleration simultaneously in motion status.
Return	Error code

NewVelocity should not exceed the maximum specified by CFG_AxMaxVel, NewAcc should not exceed the maximum acceleration specified by CFG_AxMaxAcc, and NewDec should not exceed the maximum deceleration specified by CFG_AxMaxDec.

If NewAcc or NewDec is "0", then the previous acceleration or deceleration will be used.

If this command runs successfully, then NewVelocity will be used in next motion in case the velocity is not specified before the motion.

Comments



Format Parameters

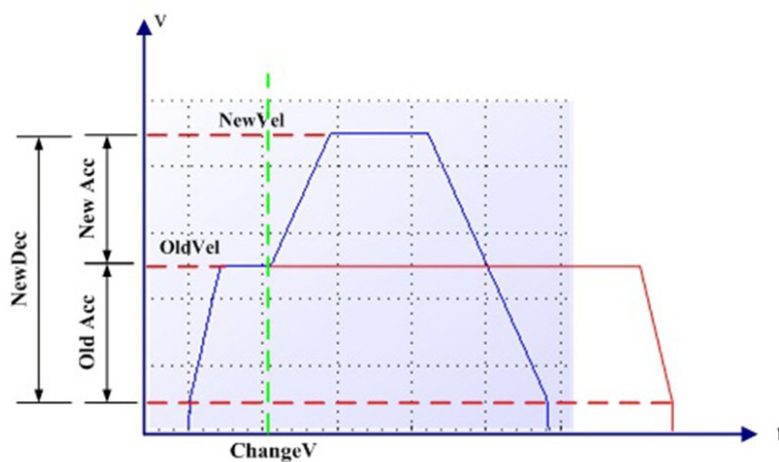
Name	Type	IN or OUT	Description
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen.
NewVelocity	F64	IN	New velocity. (unit = PPU/s)
NewAcc	F64	IN	New acceleration. (unit = PPU/s ²)
NewDec	F64	IN	New deceleration. (unit = PPU/s ²)

A.4.4.5 Acm_AxChangeVelExByRate

Format	U32 Acm_AxChangeVelExByRate (HAND AxisHandle, U32 Rate, F64 NewAcc, F64 NewDec)
Purpose	Change the velocity, acceleration and deceleration simultaneously in motion status.
Return	Error code

$NewVel = OldVel * Rate * 0.01$. The NewVel value calculated by Rate should not exceed the maximum specified by CFG_AxMaxVel, NewAcc should not exceed the maximum acceleration specified by CFG_AxMaxAcc, and NewDec should not exceed the maximum deceleration specified by CFG_AxMaxDec.
If NewAcc or NewDec is "0", then the previous acceleration or deceleration will be used.

Comments



Format Parameters

Name	Type	IN or OUT	Description
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen.
Velocity	U32	IN	Percentage of velocity change. $NewVel = OldVel * Rate * 0.01$
NewAcc	F64	IN	New acceleration. (unit = PPU/s ²)
NewDec	F64	IN	New deceleration. (unit = PPU/s ²)

A.4.4.6 Acm_AxChangeVelByRate

Format	U32 Acm_AxChangeVelByRate (HAND AxisHandle, U32 Rate)
Purpose	Change the velocity of current motion according to the given rate.
Return	Error code

Comments $NewVel = OldVel * Rate * 0.01$. Rate must be more than "0" and lower than the rate of CFG_AxMaxVel to the previous velocity. The new velocity is only valid for the current motion.

Format Parameters

Name	Type	IN or OUT	Description
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen.
Rate	U32	IN	Percentage of velocity change.

A.4.5 Point-to-Point Motion

A.4.5.1 Acm_AxMoveRel

Format	U32 Acm_AxMoveRel (HAND AxisHandle, F64 Distance)		
Purpose	Start single axis's relative position motion.		
Return	Error code		
Comments	The speed curve is decided by properties: PAR_AxVelLow, PAR_AxVelHigh, PAR_AxAcc, PAR_AxDec, and PAR_AxJerk. The range of Distance is: -2,147,483,647 ~ 2,147,483,647, if PPU is 1.		
Format Parameters			
Name	Type	IN or OUT	Description
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen.
Distance	F64	IN	Relative distance.(unit = PPU)

A.4.5.2 Acm_AxMoveAbs

Format	U32 Acm_AxMoveAbs (HAND AxisHandle, F64 Position)		
Purpose	Start single axis's absolute position motion.		
Return	Error code		
Comments	The speed curve is decided by properties: PAR_AxVelLow, PAR_AxVelHigh, PAR_AxAcc, PAR_AxDec, and PAR_AxJerk. The range of Distance is: -2,147,483,647 ~ 2,147,483,647, if PPU is 1.		
Format Parameters			
Name	Type	IN or OUT	Description
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen.
Position	F64	IN	Absolute position (unit = PPU)

A.4.5.3 Acm_AxChangePos

Format	U32 Acm_AxChangePos (HAND AxisHandle, F64 NewDistance)		
Purpose	To command axis to change the end distance while axis is in point to point motion.		
Return	Error code		
Comments	This function will change the end position to specified position on current PTP motion. The range of Distance is: -2,147,483,647~2,147,483,647, if PPU is 1.		
Format Parameters			
Name	Type	IN or OUT	Description
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen.
NewDistance	F64	IN	New relative distance.(unit = PPU)

A.4.5.4 Acm_AxMoveImpose

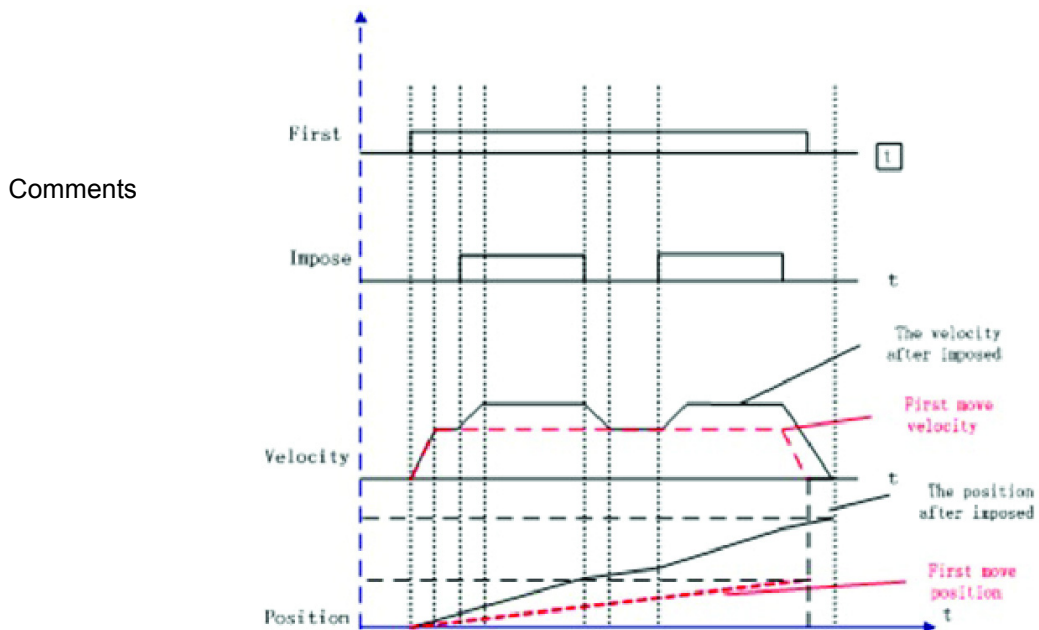
Format	U32 Acm_AxMoveImpose (HAND AxisHandle, F64 Position, F64 NewVel)
Purpose	Impose a new motion on current motion.
Return	Error code

This function just supports trapezoidal profile on this imposed motion. The end position when motion stops will be the original position added/ subtracted Position. And the driver speed will be changed too.

The whole speed curve is decided by NewVel of this motion, and PAR_AxVelLow, PAR_AxVelHigh, PAR_AxAcc, PAR_AxDec, PAR_AxJerk of original motion.

The range of Position + original position is: -2,147,483,647/PPU ~ 2,147,483,647/ PPU, and the range of NewVel + original FH is 0~ CFG_AxMaxVel.

For example, as follow:



Note:

Can not impose new motion on imposed motion.

Format Parameters			
Name	Type	IN or OUT	Description
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen.
Position	F64	IN	The relative position of new motion.(unit = PPU)
NewVel	F64	IN	The velocity of this imposed motion. (unit =PPU/s)

A.4.6 Simultaneous Motion

A.4.6.1 Acm_AxSimStartSuspendAbs

Format	U32 Acm_AxSimStartSuspendAbs (HAND AxisHandle, F64 EndPoint)		
Purpose	Set the axis in wait state for simultaneous operation. When started by start trigger, the axis will start point-to-point absolute moving to the assigned end position.		
Return	Error code		
Comments	If more than one axis wanted to do simultaneous operation, the user should call this function for each axis. The range of EndPoint is: -2,147,483,647/ PPU ~ 2,147,483,647/ PPU.		
Format Parameters			
Name	Type	IN or OUT	Description
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen.
EndPoint	F64	IN	The absolute position to move.(unit = PPU)

A.4.6.2 Acm_AxSimStartSuspendRel

Format	U32 Acm_AxSimStartSuspendRel (HAND AxisHandle, F64 Distance)		
Purpose	Set the axis in wait state for simultaneous operation. When started by start trigger, the axis will start point-to-point relative moving to the assigned end position.		
Return	Error code		
Comments	If more than one ax is wanted to do simultaneous operation, the user should call this function for each axis. The range of EndPoint is: -2,147,483,647/ PPU ~ 2,147,483,647/ PPU.		
Format Parameters			
Name	Type	IN or OUT	Description
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen.
EndPoint	F64	IN	The relative position to move.(unit = PPU)

A.4.6.3 Acm_AxSimStartSuspendVel

Format	U32 Acm_AxSimStartSuspendVel (HAND AxisHandle, U16 DriDir)		
Purpose	Set the axis in wait state for simultaneous operation. When started by start-trigger, the axis will start continuously moving.		
Return	Error code		
Comments	If more than one axis wanted to do simultaneous operation, the user should call this function for each axis.		
Format Parameters			
Name	Type	IN or OUT	Description
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen.
DriDir	U16	IN	Direction: 0: Positive direction; 1: Negative direction.

A.4.6.4 **Acm_AxSimStart**

Format	U32 Acm_AxSimStart (HAND AxisHandle)		
Purpose	Simultaneous start axis and make it output simultaneous start signal to start all axis that are waiting for start trigger.		
Return	Error code		
Comments	If more than one axis waiting on start trigger, user should set the mode of simultaneous starting /stopping by CFG_AxSimStartSource before this API.		
Format Parameters			
Name	Type	IN or OUT	Description
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen.

A.4.6.5 **Acm_AxSimStop**

Format	U32 Acm_AxSimStop (HAND AxisHandle)		
Purpose	Stop the axis and make the axis output a simultaneous stop trigger to stop all axes that are waiting for the stop trigger.		
Return	Error code		
Comments	When doing simultaneous operation, you can do this operation on any axis to stop all axes if the Simultaneous starting mode is on STA. Or else every simultaneous axis needs to call this API to stop simultaneous motion. About simultaneous starting/stopping mode, see about CFG_AxSimStartSource.		
Format Parameters			
Name	Type	IN or OUT	Description
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen.

A.4.7 Home

A.4.7.1 Acm_AxHome

Format	U32 Acm_AxHome (HAND AxisHandle, U32 HomeMode, U32 Dir)		
Purpose	To command axis to start typical home motion.		
Return	Error code		
Comments	The speed curve is decided by PAR_AxVelLow, PAR_AxVelHigh, PAR_AxAcc, PAR_AxDec and PAR_AxJerk. Please refer to Chapter 9.2.9.		
Format Parameters			
Name	Type	IN or OUT	Description
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen.
HomeMode	U32	IN	HomeMode: 0: MODE1_Abs 1: MODE2_Lmt 2: MODE3_Ref 3: MODE4_Abs_Ref 4: MODE5_Abs_NegRef 5: MODE6_Lmt_Ref 6: MODE7_AbsSearch 7: MODE8_LmtSearch 8: MODE9_AbsSearch_Ref 9: MODE10_AbsSearch_NegRef 10: MODE11_LmtSearch_Ref 11: MODE12_AbsSearchReFind 12: MODE13_LmtSearchReFind 13: MODE14_AbsSearchReFind_Ref 14: MODE15_AbsSearchReFind_NegRef 15: MODE16_LmtSearchReFind_Ref 101~137: CiA402_MODE1 ~ CiA402_MODE37
Dir	U32	IN	0: Positive direction; 1: Negative direction.

A.4.7.2 Acm_AxMoveHome

Format	U32 Acm_AxMoveHome (HAND AxisHandle, U32 HomeMode, U32 Dir)		
Purpose	Command specific axis move home.		
Return	Error code		
Comments	The speed curve is decided by PAR_AxHomeVelLow, PAR_AxHomeVelHigh, PAR_AxHomeAcc, PAR_AxHomeDec and PAR_AxHomeJerk. Please refer to Chapter 9.2.9.		
Format Parameters			
Name	Type	IN or OUT	Description
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen.
HomeMode	U32	IN	HomeMode: 0: MODE1_Abs 1: MODE2_Lmt 2: MODE3_Ref 3: MODE4_Abs_Ref 4: MODE5_Abs_NegRef 5: MODE6_Lmt_Ref 6: MODE7_AbsSearch 7: MODE8_LmtSearch 8: MODE9_AbsSearch_Ref 9: MODE10_AbsSearch_NegRef 10: MODE11_LmtSearch_Ref 11: MODE12_AbsSearchReFind 12: MODE13_LmtSearchReFind 13: MODE14_AbsSearchReFind_Ref 14: MODE15_AbsSearchReFind_NegRef 15: MODE16_LmtSearchReFind_Ref 101~137: CiA402_MODE1 ~ CiA402_MODE37
Dir	U32	IN	0: Positive direction; 1: Negative direction.

A.4.8 Position/Counter Control

A.4.8.1 Acm_AxSetCmdPosition

Format	U32 Acm_AxSetCmdPosition (HAND AxisHandle, F64 Position)		
Purpose	Set command position for the specified axis.		
Return	Error code		
Comments			
Format Parameters			
Name	Type	IN or OUT	Description
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen.
Position	F64	IN	New command position(uint:PPU)

A.4.8.2 Acm_AxGetCmdPosition

Format	U32 Acm_AxGetCmdPosition (HAND AxisHandle, PF64 Position)		
Purpose	Get current command position of the specified axis.		
Return	Error code		
Comments			
Format Parameters			
Name	Type	IN or OUT	Description
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen.
Position	PF64	OUT	Return the command position.(unit:PPU)

A.4.8.3 Acm_AxGetActualPosition

Format	U32 Acm_AxSetActualPosition (HAND AxisHandle, F64 Position)		
Purpose	Set actual position for the specified axis.		
Return	Error code		
Comments	The PCI-1203 does not support this API.		
Format Parameters			
Name	Type	IN or OUT	Description
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen.
Position	PF64	IN	New actual position (unit:PPU)

A.4.9 Latch

A.4.9.1 Acm_AxGetLatchData

Format	U32 Acm_AxGetLatchData (HAND AxisHandle, U32 PositionNo, F64 Position)		
Purpose	Get the latch data in device after triggering latch.		
Return	Error code		
Comments			
Format Parameters			
Name	Type	IN or OUT	Description
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen.
PositionNo	U32	IN	0: Command position 1: Actual position.
Position	PF64	OUT	Latch data.(uint:PPU)

A.4.9.2 Acm_AxTriggerLatch

Format	U32 Acm_AxTriggerLatch (HAND AxisHandle)		
Purpose	Trigger the hardware to latch position data.		
Return	Error code		
Comments			
Format Parameters			
Name	Type	IN or OUT	Description
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen.

A.4.9.3 Acm_AxResetLatch

Format	U32 Acm_AxResetLatch (HAND AxisHandle)		
Purpose	Clear the latch data and latch flag in device.		
Return	Error code		
Comments			
Format Parameters			
Name	Type	IN or OUT	Description
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen.

A.4.9.4 Acm_AxGetLatchFlag

Format	U32 Acm_AxGetLatchFlag (HAND AxisHandle, PU8 LatchFlag)		
Purpose	Get the latch flag in device if data is latched.		
Return	Error code		
Comments			
Format Parameters			
Name	Type	IN or OUT	Description
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen.
LatchFlag	PU8	OUT	The flag data. 1: Data is latched. 0: not latched.

A.4.10 Ext-Drive

A.4.10.1 Acm_AxSetExtDrive

Format	U32 Acm_AxSetExtDrive (HAND AxisHandle, U16 ExtDrvMode)		
Purpose	Enable or disable external drive mode.		
Return	Error code		
Comments			
Format Parameters			
Name	Type	IN or OUT	Description
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen.
ExtDrvMode	U16	IN	0: Disabled (Stop command) 1: JOG Mode 2: MPG Mode (Not support) 3: JOG Step mode(Reserved)

A.4.10.2 Acm_AxJog

Format	U32Acm_AxJog(HANDAxisHandle,U16Direction)		
Purpose	This function starts Jog motion		
Return	Error code		
Comments	Before calling this API, it must call Acm_AxSetExtDrive to set the external drive mode to JOG Mode		
Format Parameters			
Name	Type	IN or OUT	Description
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen.
Direction	U16	IN	Jog direction 0: Positive 1: Negative

A.4.11 Cam/Gear

A.4.11.1 Acm_AxCamInAx

Format	U32 Acm_AxCamInAx (HAND AxisHandle, HAND MasAxisHandle, F64 MasterOffset, F64 SlaveOffset, F64 MasterScaling, F64 SlaveScaling, U32 CamTableID, U32 RefSrc)
Purpose	This function starts cam synchronization with a cam table between a slave(following) axis and master (leading) axis. Camming is done with one table (two dimensional - describing master and slave positions together). The table should be strictly monotonic rising or falling, going both reverse and forward with the master.
Return	Error code

If this command is set successfully, slave axis will move according to cam curve fitted by CamTable when master axis is moving continuously or point to point.

Cannot set command/actual position for slave axis and master axis by Acm_AxSetCmdPosition or Acm_AxSetActualPosition.

To terminate follow relationship of slave axis, user can call Acm_AxStopDec, Acm_AxStopEmg, and the slave axis will be Ready status.

The pulse number of master axis rotated 360 degree should be set by property CFG_AxModuleRange. The edited CamTable needs to set by Acm_DevDownloadCAMTable, and related E-cam configure can be set by Acm_DevConfigCAMTable.

After all of above, slave axis will be synchronous status if calls Acm_AxCamInAx successfully. Then slave axis will follow mater axis to move.

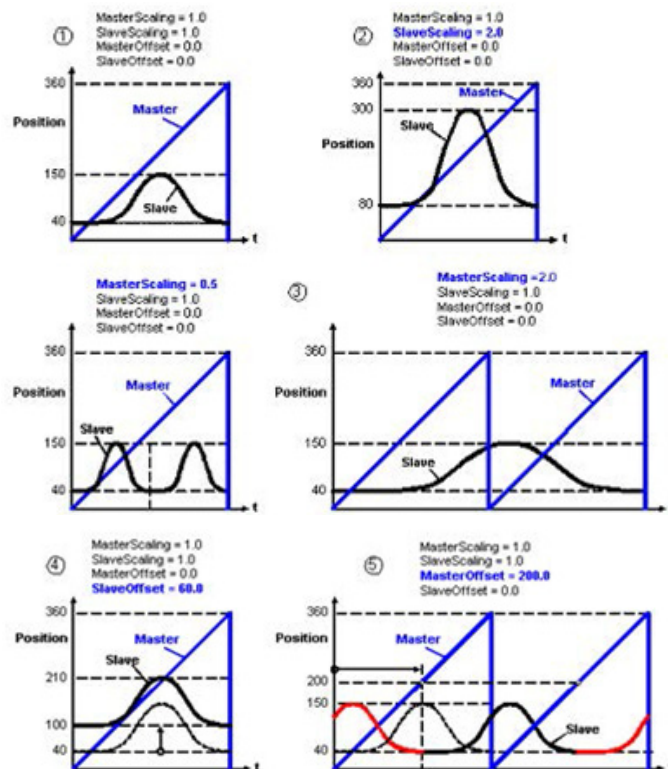
The parameters MasterScaling, SlaveScaling, MasterOffset, SlaveOffset are used to adjust current Camtable based on edited CamTable.

The figures about scalling and offset are as follow.

About detail E-Cam operation, see E-Cam Flow Chart in Chapter 9.4.3.3.

See also Acm_DevDownLoadCAMTable, Acm_DevConfigCAMTable, Acm_DevLoadCAMTableFile.

Comments



Format Parameters			
Name	Type	IN or OUT	Description
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen, this handle should be slave (following) axis handle.
MasAxisHandle	HAND	IN	Axis handle from Acm_AxOpen, this handle should be master (leading) axis handle.
MasterOffset	F64	IN	Shifting the cam along the coordinates of the master axis.
SlaveOffset	F64	IN	Shifting the cam along the coordinates of the slave axis.
MasterScaling	F64	IN	Scaling factor for the cam in the coordinates of the master axis. The overall master profile is multiplied by this factor. This should be larger than zero.
SlaveScaling	F64	IN	Scaling factor for the cam in the coordinates of the slave axis. The overall slave profile is multiplied by this factor. This should be larger than zero.
CamTableID	U32	IN	Acm_DevDownloadCAMTable. The PCI-1203 reserves 2 cam tables. So the ID can be 0, 1.
RefSrc	U32	IN	Cam table's master position reference to command-position (0) or actual-position(1). 0: Command position. 1: Actual position.

A.4.11.2 Acm_AxGearInAx

Format	U32 Acm_AxGearInAx (HAND AxisHandle, HAND MasAxisHandle, I32 Numerator, I32 Denominator, U32 RefSrc, U32 Absolute)
Purpose	This function starts gear synchronization with a ratio between a slave(following) axis and master (leading) axis.
Return	Error code

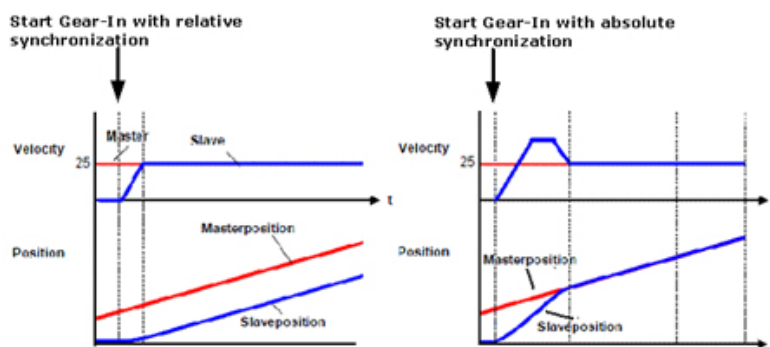
Slave axis will follow mater axis motion if this function is called successfully. The master axis and slave axis can not be reset command/actual position by Acm_AxSetCmdPosition / Acm_AxSetActualPosition in gear motion. The relationship of master and slave can be terminated by calling the Acm_AxStopDec and Acm_AxStopEmg, the axis will return Ready status. Gear Ratio: Numerator/Denominator. If the value is positive, the slave will move at the same direction with master axis, or else it will move at the oppnent direction with mater axis.

Absolute:

Absolute=1: Absolute relationship. Slave axis will compensate the offset with master axis.

Absolute=0: Relative relationship. Slave axis will not compensate any offset with master axis.

Comments



About the E-gear operation, see about E-Gear flow chart in Chapter 9.4.2.3.

Format Parameters			
Name	Type	IN or OUT	Description
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen, this handle should be slave (following) axis handle.
MasAxisHandle	HAND	IN	Axis handle from Acm_AxOpen, this handle should be master (leading) axis handle.
Numerator	I32	IN	Gear ratio numerator.
Denominator	I32	IN	Gear ratio denominator
RefSrc	U32	IN	Slave axis engages to master axis's command-position (0) or actual-position (1).
Absolute	U32	IN	The synchronization is relative to start position (random position values upon reaching synchronization) or absolute. 0: relative, 1: absolute

A.4.11.3 Acm_AxPhaseAx

Format	U32 Acm_AxPhaseAx(HAND AxisHandle, F64 Acc, F64 Dec, F64 PhaseSpeed, F64 PhaseDist)		
Purpose	Enable the phase lead or phase lag motion of the slave axis during the process of electronic cam or electronic gear.		
Return	Error code		
Comments	<p>The API enables the slave axis to pull ahead or lag behind the previous motion during the process of electronic cam or electronic gear. If PhaseDist>0, it enables the phase lead motion; if PhaseDist<0, it enables the phase lag motion.</p> <p>If the remaining Pulse is not enough, then the slave axis could not reach the specified phase. An error will be retrieved if this command is re-sent before the phase lead/lag motion ends.</p> <p>Because of the floating number calculation error, the maximum acceleration/ deceleration specified through CFG_AxMaxAcc/CFG_AxMaxDec by the slave axis must exceed Acc/Dec by 100,000 at least.</p>		
Format Parameters			
Name	Type	IN or OUT	Description
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen, this handle should be slave (following) axis handle.
Acc	HAND	IN	Acceleration of phase lead/lag motion. (Unit: PPU/s ²)
Dec	F64	IN	Deceleration of phase lead/lag motion. (Unit: PPU/s ²)
PhaseSpeed	F64	IN	Speed of phase lead/lag motion. (Unit: PPU/s)
PhaseDist	F64	IN	Distance of phase lead/lag motion. (Unit: PPU)

A.4.12 Gantry/Tangent

A.4.12.1 Acm_AxTangentInGp

Format	U32 Acm_AxTangentInGp (HAND AxisHandle, HAND MasGroupHandle, P16 StartVectorArray, U8 Working_plane, I16 Direction)
Purpose	Command axis to move at same direction with tangent of group path.
Return	Error code

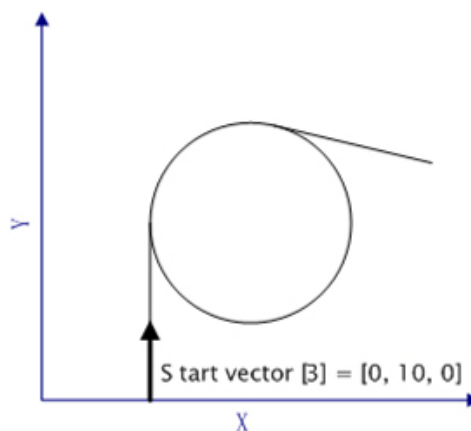
The axis will be synchronous state and will follow group motion at the direction of tangent of group's path if this function is called successfully. The axis and group cannot be reset command/actual position. Using Acm_AxStopDec/Acm_AxStopEmg will terminate synchronous state of the axis, and it will return Ready state.

The axis cannot be one axis in group.

StartVectorArray is the initial vector which is starting direction of axis.

For example, StartVectorArray[3] = {0, 10, 0}, working plane = 0: XY plane.

Comments



Note:

The start vector should be as close as possible with group's motion starting direction, or else there may be the error of motion acceleration greater than max acceleration of device. And if the angle between two conjoint paths is too large, the error will happen too, so user should pay attention to angle between paths. The formula of calculating max angle deviation is as follow:
For example:

Setting:

Module Range (CFG_AxModuleRange): 3,600 pulse.

Max Acceleration (CFG_AxMaxAcc): 10^7 .

Max angle of slope transformation: $10^7 \times 10^{-6} \times 360^\circ / 3,600 = 1^\circ$.

Format Parameters

Name	Type	IN or OUT	Description
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen.
Mas-GroupHandle	HAND	IN	Group handle from Acm_GpAddAxis.
StartVectorArray	P16	IN	Must be 3 dimensions.
Working_plane	U8	IN	0:XY; 1:YZ; 2:XZ
Direction	I16	IN	Same:0; Opposite:1

A.4.12.2 Acm_AxGantryInAx

Format	U32 Acm_AxGantryInAx (HAND AxisHandle, HAND MasAxisHandle, I16 RefMasterSrc, I16 Direction)		
Purpose	Command two axes to move e-gantry motion.		
Return	Error code		
Comments	<p>The slave axis will move synchronously with mater axis. The relationship of master and slave can be terminated by calling the Acm_AxStopDec / Acm_AxStopEmg; the axis will return Ready status.</p> <p>There are some restrictions about gantry:</p> <ol style="list-style-type: none"> (1). Cannot set any command except Acm_AxStopDec /Acm_AxStopEmg to slave axis; (2). Slave axis cannot be add in any group; (3). If the axis is already one axis in group, it cannot be slave axis of gantry. <p>If the command/actual position of master axis is reset, command/actual position of slave axis is reset same value too.</p> <p>About the gantry operation, see about gantry flow chart in Chapter 9.4.4.3.</p>		
Format Parameters			
Name	Type	IN or OUT	Description
AxisHandle	HAND	IN	Slave axis handle from Acm_AxOpen.
MasAxisHandle	HAND	IN	Master axis handle from Acm_AxOpen.
RefMasterSrc	I16	IN	The reference source: 0: Command position 1: Actual position (Reserved)
Direction	I16	IN	The direction with master axis. 0:same; 1:opposite

A.4.13 Stop

A.4.13.1 Acm_AxStopDec

Format	U32 Acm_AxStopDec (HAND AxisHandle)		
Purpose	Command the axis to decelerate and stop.		
Return	Error code		
Comments	If the axis is in synchronous drive mode, for example, the slave axis of E-cam/E-gear/Tangent motion, then the API can be used to terminate the synchronization.		
Format Parameters			
Name	Type	IN or OUT	Description
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen.

A.4.13.2 Acm_AxStopEmg

Format	U32 Acm_AxStopEmg (HAND AxisHandle)		
Purpose	Command the axis to stop (without decelerating).		
Return	Error code		
Comments	If the axis is in synchronous drive mode, for example, the slave axis of E-cam/E-gear/Tangent motion, then the API can be used to terminate the synchronization.		
Format Parameters			
Name	Type	IN or OUT	Description
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen.

A.4.13.3 Acm_AxStopDecEx

Format	U32 Acm_AxStopDecEx (HAND AxisHandle, F64 NewDec)		
Purpose	Command the axis to stop and specify the deceleration.		
Return	Error code		
Comments	If the decelerating command is sent and the remaining pulse is not enough for supporting the specified NewDec, then pulse break will occur.		
Format Parameters			
Name	Type	IN or OUT	Description
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen.
NewDec	F64	IN	Deceleration for decelerating. (Unit: PPU/s ²)

A.4.14 PT/PVT Motion

A.4.14.1 Acm_AxResetPTData

Format	U32 Acm_AxResetPTData(HAND AxisHandle)		
Purpose	Reset the PT buffer.		
Return	Error code		
Comments	This function need to execute before the new data is written.		
Format Parameters			
Name	Type	IN or OUT	Description
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen.

A.4.14.2 Acm_AxAddPTData

Format	U32 Acm_AxAddPTData(HAND AxisHandle, F64 Pos, F64 time)		
Purpose	Add PT data into PT buffer.		
Return	Error code		
Comments	The Pos and Time value must begin from 0. It needs 4 points (Pos and time) to shape a formal Trapezoid velocity profile.		
Format Parameters			
Name	Type	IN or OUT	Description
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen.
Pos	F64	IN	Position (PPU)
Time	F64	IN	Time (ms)

A.4.14.3 Acm_AxStartPT

Format	U32 Acm_AxStartPT (HAND AxisHandle, U8 BfMode, U16 rNum)		
Purpose	Start PT motion		
Return	Error code		
Comments	The PT buffer needs to write more than 2 pieces data before Acm_AxStartPT is executed. Only Static mode supports repeat operation.		
Format Parameters			
Name	Type	IN or OUT	Description
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen.
			Buffer mode Value Description Marco
BfMode	U8	IN	Buffer mode Value:0 Description: Set PT buffer in static operation mode Macro:PT_BUFFER_STATIC Value:1 Description: Set PT buffer in dynamic operation mode Macro:PT_BUFFER_DYNAMIC
rNum	U16	IN	Repeat number Static mode: rNum value must be larger than 0. Dynamic mode: Not supported, write 0 when using Dynamic mode.

A.4.14.4 Acm_AxCheckPTBuffer

Format	U32 Acm_AxCheckPTBuffer(HAND AxisHandle, PU8 freeblock)		
Purpose	Check the free space of PT buffer. The maximum number provided by PT buffer is 256.		
Return	Error code		
Comments	The remaining free data block in PT buffer.		
Format Parameters			
Name	Type	IN or OUT	Description
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen.
freeblock	PU8	OUT	Number of free block

A.4.14.5 Acm_AxResetPVTTable

Format	U32 Acm_AxResetPVTTable(HAND AxisHandle)		
Purpose	Reset the PVT data in PVT table		
Return	Error code		
Comments	PVT table need to be reset before a new table is loaded into buffer.		
Format Parameters			
Name	Type	IN or OUT	Description
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen.

A.4.14.6 Acm_AxLoadPVTTable

Format	U32 Acm_AxLoadPVTTable(HAND AxisHandle, U8 Num, PF64 Pos, PF64 Vel, PF64 Time)		
Purpose	Load PVT table into PVT buffer. There are 42 data block provided by PVT buffer.		
Return	Error code		
Comments	The Pos and Time value must begin from 0. It needs 4 points (Pos and time) to shape a formal Trapezoid velocity profile.		
Format Parameters			
Name	Type	IN or OUT	Description
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen.
Num	U8	IN	Number of data point
Pos	PF64	IN	Position information(PPU)
Vel	PF64	IN	Velocity information(PPU/ms)
Time	PF64	IN	Time information(ms)

A.4.14.7 Acm_AxStartPVT

Format	U32 Acm_AxStartPVT(HAND AxisHandle, U16 rNum)		
Purpose	Start PVT motion		
Return	Error code		
Comments	If need to add more than 4 data points before calling this API		
Format Parameters			
Name	Type	IN or OUT	Description
AxisHandle	HAND	IN	Axis handle from Acm_AxOpen.
rNum	U16	IN	Repeat number

A.5 Group

A.5.1 System

A.5.1.1 Acm_GpAddAxis

Format	U32 Acm_GpAddAxis (PHAND GpHandle, HAND AxHandle)		
Purpose	Add an axis to the specified group.		
Return	Error code		
Comments	If GpHandle points to NULL, driver will create a new group handle and add the axis to this new group. If GpHandle points to a valid group handle, driver will just add the axis to the group. At most, there are 6 groups in PCI-1203.No more than 8 axes in each group for PCI-1203. The same axis cannot be added in different groups. The master axis in group is the minimal PhysicalID one. The parameters of group are initialized when the first axis is added. Such as CFG_GpPPU, PAR_GpVelLow, PAR_GpVelHigh, PAR_GpAcc, PAR_GPDec and PAR_GpJerk.		
Format Parameters			
Name	Type	IN or OUT	Description
GpHandle	PHAND	IN/OUT	Point to group handle (NULL or not).
AxHandle	HAND	IN	Axis handle from Acm_AxOpen.

A.5.1.2 Acm_GpRemAxis

Format	U32 Acm_GpRemAxis (HAND GpHandle, HAND AxHandle)		
Purpose	Remove an axis from the specified group.		
Return	Error code		
Comments	After Acm_GpRemAxis is called and no axis is in group, the GpHandle can still be used. You can use this group handle to add other axes. But if you have called Acm_GpClose to close this group handle, the group handle can't be used again.		
Format Parameters			
Name	Type	IN or OUT	Description
GpHandle	HAND	IN	Group handle from Acm_GpAddaxis.
AxHandle	HAND	IN	Axis handle from Acm_AxOpen.

A.5.1.3 Acm_GpClose

Format	U32 Acm_GpClose (PHAND pGroupHandle)		
Purpose	Remove all axes in the group and close the group handle.		
Return	Error code		
Comments	If the group number is greater than maximal group number of device, new group cannot be created. At the time, you must close one existing group if you want to create new group.		
Format Parameters			
Name	Type	IN or OUT	Description
pGroupHandle	PHAND	IN	Group handle from Acm_GpAddaxis.

A.5.1.4 Acm_GpResetError

Format	U32 Acm_GpResetError (HAND GroupHandle)		
Purpose	Reset group states.		
Return	Error code		
Comments	If the group is in STA_GP_ERROR_STOP state, the state will be changed to STA_GP_READY after calling this function.		
Format Parameters			
Name	Type	IN or OUT	Description
GroupHandle	HAND	IN	Group handle from Acm_GpAddaxis.

A.5.2 Motion Status

A.5.2.1 Acm_GpGetState

Format	U32 Acm_GpGetState (HAND GroupHandle, PU16 pState)		
Purpose	Get the group's current state.		
Return	Error code		
Comments	If an axis of group is implementing command of single-axis motion, the group's state will be unchanged.		
Format Parameters			
Name	Type	IN or OUT	Description
GroupHandle	HAND	IN	Group handle from Acm_GpAddaxis.
pState	PU16	OUT	Group states: 0: STA_GP_DISABLE 1: STA_GP_READY 2: STA_GP_STOPPING 3: STA_GP_ERROR_STOP 4: STA_GP_MOTION 5: STA_GP_AX_MOTION (Not support) 6: STA_GP_MOTION_PATH

A.5.2.2 Acm_GpGetCmdVel

Format	U32 Acm_GpGetCmdVel(HAND GroupHandle, PF64 CmdVel)		
Purpose	Get the current velocity of the group.		
Return	Error code		
Comments	Get the current velocity during interpolation or continuous interpolation of the group through API.		
Format Parameters			
Name	Type	IN or OUT	Description
GroupHandle	HAND	IN	Group handle from Acm_GpAddaxis.
CmdVel	PF64	OUT	Return the current velocity of the group. Unit: PPU/s. (PPU is of the axis with the lowestID.)

A.5.3 MotionStop

A.5.3.1 Acm_GpStopDec

Format	U32 Acm_GpStopDec (HAND GroupHandle)		
Purpose	Command axis in this group to decelerate to stop.		
Return	Error code		
Comments			
Format Parameters			
Name	Type	IN or OUT	Description
GroupHandle	HAND	IN	Group handle from Acm_GpAddaxis.

A.5.3.2 Acm_GpStopEmg

Format	U32 Acm_GpStopEmg(HAND GroupHandle)		
Purpose	Command axis in this group to stop immediately without deceleration.		
Return	Error code		
Comments			
Format Parameters			
Name	Type	IN or OUT	Description
GroupHandle	HAND	IN	Group handle from Acm_GpAddaxis.

A.5.4 Interpolation Motion

A.5.4.1 Acm_GpMoveLinearRel

Format	U32 Acm_GpMoveLinearRel(HAND GroupHandle, PF64 DistanceArray, PU32 pArrayElements)		
Purpose	Command group to execute relative line interpolation.		
Return	Error code		
Comments	<p>The sequence of data in DistanceArray must follow the order of X axis, Y axis, Z axis, U axis. For example, if one group has two axes: Y axis and U axis. The first data in DistanceArray means Y axis' relative distance and the second data means U axis' relative distance. The unit of distance in DistanceArray is PPU of each axis in group.</p> <p>The difference between line interpolation and direct interpolation: line interpolation's speed is divided into even line speed for every axis, axis moves at this even line speed. Mostly, line interpolation is applied to the axis assembled as right angle. But direct interpolation's line speed is set into master axis (The smallest Physical ID axis) and other axes in group start/stop as the same time as master axis. Mostly, direct interpolation is applied to the axis assembled as oblique-angle.</p>		
Format Parameters			
Name	Type	IN or OUT	Description
GroupHandle	HAND	IN	Group handle from Acm_GpAddaxis.
DistanceArray	PF64	IN	Distance array of axis in group, each value of array elements represent the axis relative position.
pArrayElements	PU32	IN/OUT	Element count in the array(This count must equal to the axis count in this group, or else it will be returned axis count in group)

A.5.4.2 Acm_GpMoveLinearAbs

Format	U32 Acm_GpMoveLinearAbs (HAND GroupHandle, PF64 PositionArray, PU32 pArrayElements)		
Purpose	Command group to execute absolute line interpolation.		
Return	Error code		
Comments	<p>The sequence of data in PositionArray must follow the order of X axis, Y axis, Z axis, U axis. For example, if one group has two axes: Y axis and U axis. The first data in PositionArray means Y axis' absolute position and the second data means U axis' absolute position. The unit of distance in PositionArray is PPU of each axis in group.</p> <p>The difference between line interpolation and direct interpolation: line interpolation's speed is divided into even line speed for every axis, axis moves at this even line speed. Mostly, line interpolation is applied to the axis assembled as right angle. But direct interpolation's line speed is set into master axis (The smallest Physical ID axis) and other axes in group start/stop as the same time as master axis. Mostly, direct interpolation is applied to the axis assembled as oblique-angle.</p>		
Format Parameters			
Name	Type	IN or OUT	Description
GroupHandle	HAND	IN	Group handle from Acm_GpAddaxis.
PositionArray	PF64	IN	Position array of axis in group, each value of array elements represent the axis absolute position.

pArrayElements	PU32	IN/OUT	Element count in the array(This count must equal to the axis count in this group, or else it will be returned axis count in group)
----------------	------	--------	------------------------------------------------------------------------------------------------------------------------------------

A.5.4.3 Acm_GpMoveDirectRel

Format	U32 Acm_GpMoveDirectRel (HAND GroupHandle, PF64 DistanceArray, PU32 ArrayElements)		
Purpose	Command group to execute relative direct line interpolation.		
Return	Error code		
Comments	<p>The sequence of data in DistanceArray must follow the order of X axis, Y axis, Z axis, U axis and so on. For example, if one group has two axes: Y axis and U axis. The first data in DistanceArray means Y axis' relative distance and the second data means U axis' relative distance. The unit of distance in DistanceArray is PPU of each axis in group.</p> <p>The difference between line interpolation and direct interpolation: line interpolation's speed is divided into even line speed for every axis, axis moves at this even line speed. Mostly, line interpolation is applied to the axis assembled as right angle. But direct interpolation's line speed is set into master axis (The smallest Physical ID axis) and other axes in group start/stop as the same time as master axis. Mostly, direct interpolation is applied to the axis assembled as oblique-angle.</p>		
Format Parameters			
Name	Type	IN or OUT	Description
GroupHandle	HAND	IN	Group handle from Acm_GpAddaxis.
DistanceArray	PF64	IN	Distance array of axis in group, each value of array elements represent the axis relative position
ArrayElements	PU32	IN/OUT	Element count in the array(This count must equal to the axis count in this group, or else it will be returned axis count in group)

A.5.4.4 Acm_GpMoveDirectAbs

Format	U32 Acm_GpMoveDirectAbs (HAND GroupHandle, PF64 PositionArray, PU32 ArrayElements)		
Purpose	Command group to execute absolute direct line interpolation.		
Return	Error code		
Comments	<p>The sequence of data in PositionArray must follow the order of X axis, Y axis, Z axis, U axis and so on. For example, if one group has two axes: Y axis and U axis. The first data in PositionArray means Y axis' absolute position and the second data means U axis' absolute position. The unit of distance in PositionArray is PPU of each axis in group.</p> <p>The difference between line interpolation and direct interpolation: line interpolation's speed is divided into even line speed for every axis, axis moves at this even line speed. Mostly, line interpolation is applied to the axis assembled as right angle. But direct interpolation's line speed is set into master axis (The smallest Physical ID axis) and other axes in group start/stop as the same time as master axis. Mostly, direct interpolation is applied to the axis assembled as oblique-angle.</p>		
Format Parameters			
Name	Type	IN or OUT	Description
GroupHandle	HAND	IN	Group handle from Acm_GpAddaxis.

PositionArray	PF64	IN	Distance array of axis in group, each value of array elements represent the axis absolute position.
ArrayElements	PU32	IN/OUT	Element count in the array (This count must equal to the axis count in this group, or else it will be returned axis count in group).

A.5.4.5 Acm_GpMoveCircularRel

Format	U32 Acm_GpMoveCircularRel (HAND GroupHandle, PF64 CenterArray, PF64 EndArray, PU32 pArrayElements, I16 Direction)		
Purpose	Command group to execute relative ARC interpolation.		
Return	Error code		
Comments	The sequence of data in CenterArray and EndArray must follow the order of X axis, Y axis, Z axis, U axis and so on. For example, if one group has Y axis and U axis, the first data in CenterArray means Y axis' center distance and the second data means U axis' center distance. The unit of distance in CenterArray and EndArray is PPU of each axis in group.		
Format Parameters			
Name	Type	IN or OUT	Description
GroupHandle	HAND	IN	Group handle from Acm_GpAddaxis.
CenterArray	PF64	IN	Axis relative distance of center point.
EndArray	PF64	IN	Axis relative distance of end point.
pArrayElements	PU32	IN/OUT	must equal to the axis count in this group, or else it will be returned axis count in group).
Direction	I16	IN	Direction: 0: DIR_CW(clockwise) 1: DIR_CCW(counterclockwise)

A.5.4.6 Acm_GpMoveCircularAbs

Format	U32 Acm_GpMoveCircularAbs (HAND GroupHandle, PF64 CenterArray, PF64 EndArray, PU32 pArrayElements, I16 Direction)		
Purpose	Command group to execute absolute ARC interpolation.		
Return	Error code		
Comments	The sequence of data in CenterArray and EndArray must follow the order of X axis, Y axis, Z axis, U axis. For example, if one group has Y axis and U axis, the first data in CenterArray means Y axis' center position and the second data means U axis' center position. The unit of distance in CenterArray and EndArray is PPU of each axis in group.		
Format Parameters			
Name	Type	IN or OUT	Description
GroupHandle	HAND	IN	Group handle from Acm_GpAddaxis.
CenterArray	PF64	IN	Absolute distance of center point.
EndArray	PF64	IN	Absolute distance of end point.
pArrayElements	PU32	IN/OUT	Element count in the array (This count must equal to the axis count in this group, or else it will be returned axis count in group).
Direction	I16	IN	Direction: 0: DIR_CW(clockwise) 1: DIR_CCW(counterclockwise)

A.5.4.7 Acm_GpMoveCircularRel_3P

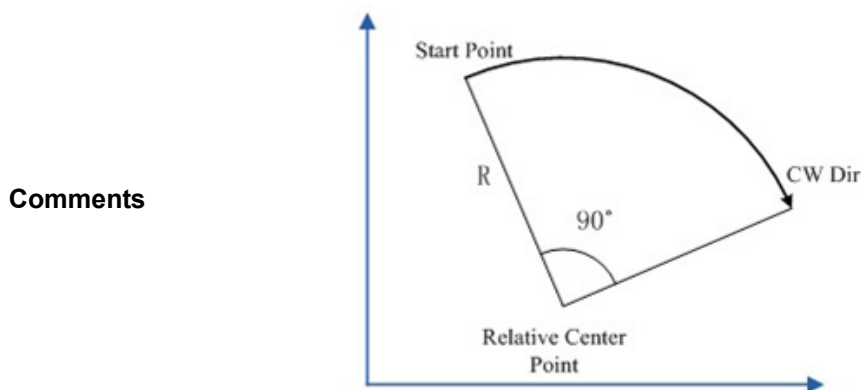
Format	U32 Acm_GpMoveCircularRel_3P (HAND GroupHandle, PF64 RefArray, PF64 EndArray, PU32 pArrayElements, I16 Direction)		
Purpose	Command group to execute relative ARC interpolation by three specified points.		
Return	Error code		
Comments	The sequence of data in RefArray and EndArray must follow the order of X axis, Y axis, Z axis, U axis and so on. For example, if one group has Y axis and U axis, the first data in RefArray means Y axis' reference distance and the second data means U axis' reference distance. The unit of distance in RefArray and EndArray is PPU of each axis in group.		
Format Parameters			
Name	Type	IN or OUT	Description
GroupHandle	HAND	IN	Group handle from Acm_GpAddaxis.
RefArray	PF64	IN	Relative distance of reference point.
EndArray	PF64	IN	Relative distance of end point.
pArrayElements	PU32	IN/OUT	Element count in the array (This count must equal to the axis count in this group, or else it will be returned axis count in group).
Direction	I16	IN	Direction: 0: DIR_CW(clockwise) 1: DIR_CCW (counterclockwise)

A.5.4.8 Acm_GpMoveCircularAbs_3P

Format	U32 Acm_GpMoveCircularAbs_3P (HAND GroupHandle, PF64 RefArray, PF64 EndArray, PU32 pArrayElements, I16 Direction)		
Purpose	Command group to execute absolute ARC interpolation by three specified points.		
Return	Error code		
Comments	The sequence of data in RefArray and EndArray must follow the order of X axis, Y axis, Z axis, U axis and so on. For example, if one group has Y axis and U axis, the first data in RefArray means Y axis' reference position and the second data means U axis' reference position. The unit of distance in RefArray and EndArray is PPU of each axis in group.		
Format Parameters			
Name	Type	IN or OUT	Description
GroupHandle	HAND	IN	Group handle from Acm_GpAddaxis.
RefArray	PF64	IN	Absolute position of reference point.
EndArray	PF64	IN	Absolute position of end point.
pArrayElements	PU32	IN/OUT	Element count in the array (This count must be equal to the axis count in this group, or else it will be returned axis count in group).
Direction	I16	IN	Direction: 0: DIR_CW(clockwise) 1: DIR_CCW (counterclockwise)

A.5.4.9 **Acm_GpMoveArcRel_Angle**

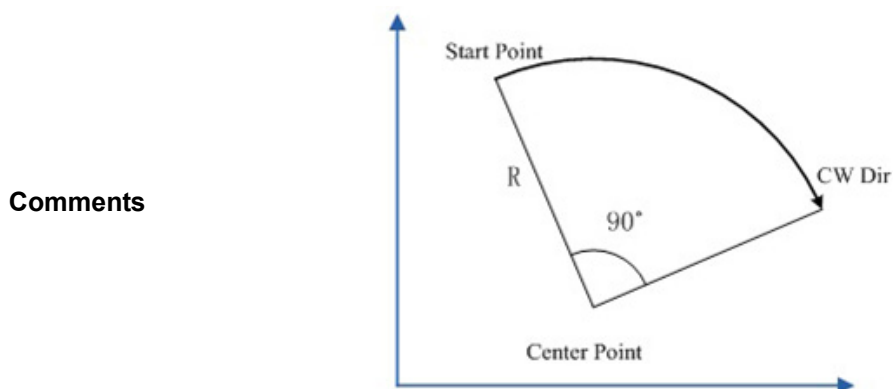
Format	U32 Acm_GpMoveArcRel_Angle (HAND GroupHandle, PF64 CenterArray, F64 Degree, PU32 ArrayElements, I16 Direction)
Purpose	Complete circular interpolation through relative center coordinates, angle and direction of rotation.
Return	Error code

**Format Parameters**

Name	Type	IN or OUT	Description
GroupHandle	HAND	IN	Group handle from Acm_GpAddaxis.
CenterArray	PF64	IN	Relative distance between center point and start point.
Degree	F64	IN	Angle of rotation. (Range: 0~360)
pArrayElements	PU32	IN	Axis count.
Direction	I16	IN	Direction: 0: CW-Dir 1: CCW_Dir

A.5.4.10 **Acm_GpMoveArcAbs_Angle**

Format	U32 Acm_GpMoveArcAbs_Angle (HAND GroupHandle, PF64 CenterArray, F64 Degree, PU32 ArrayElements, I16 Direction)
Purpose	Complete circular interpolation through absolute center coordinates, angle and direction of rotation.
Return	Error code

**Format Parameters**

Name	Type	IN or OUT	Description
------	------	-----------	-------------

GroupHandle	HAND	IN	Group handle from Acm_GpAddaxis.
CenterArray	PF64	IN	Center coordinates.
Degree	F64	IN	Angle of rotation. (Range: 0~360)
ArrayElements	PU32	IN	Axis count.
Direction	I16	IN	Direction: 0: CW-Dir 1: CCW_Dir

A.5.4.11 Acm_GpMove3DArcRel

Format	U32 Acm_GpMove3DArcRel(HAND GroupHandle, PF64 CenterArray, PF64 EndArray, PU32 pArrayElements, I16 Direction)		
Purpose	Command group to execute relative 3D arc interpolation by specified points		
Return	Error code		
Comments	User can input center point, end point and direction to execute relative 3D arc interpolation. Please refer to 3-Axis Arc Interpolation section of Chapter 9.3.3.2.		
Format Parameters			
Name	Type	IN or OUT	Description
GroupHandle	HAND	IN	Group handle from Acm_GpAddaxis.
CenterArray	PF64	IN	Axis relative distance of center point.
EndArray	PF64	IN	Axis relative distance of end point.
pArrayElements	PU32	IN/OUT	Element count in the array (This count must equal to the axis count in this group, or else it will be returned axis count in group).
Direction	I16	IN	Direction: 0: DIR_CW(clockwise) 1: DIR_CCW(counterclockwise)

A.5.4.12 Acm_GpMove3DArcAbs

Format	U32 Acm_GpMove3DArcAbs(HAND GroupHandle, PF64 CenterArray, PF64 EndArray, PU32 pArrayElements, I16 Direction)		
Purpose	Command group to execute absolute ARC interpolation.		
Return	Error code		
Comments	User can input center point, end point and direction to execute absolute 3D arc interpolation. Please refer to 3-Axis Arc Interpolation section of Chapter 9.3.3.2.		
Format Parameters			
Name	Type	IN or OUT	Description
GroupHandle	HAND	IN	Group handle from Acm_GpAddaxis.
CenterArray	PF64	IN	Absolute distance of center point.
EndArray	PF64	IN	Absolute distance of end point.
pArrayElements	PU32	IN/OUT	Element count in the array (This count must equal to the axis count in this group, or else it will be returned axis count in group).
Direction	I16	IN	Direction: 0: DIR_CW(clockwise) 1: DIR_CCW(counterclockwise)

A.5.4.13 **Acm_GpMove3DArcRel_V**

Format	U32 Acm_GpMove3DArcRel_V(HAND GroupHandle, PF64 CenterArray, PF64 NVectorArray, F64 Degree, PU32 pArrayElements, I16 Direction)		
Purpose	Command group to execute relative 3D arc interpolation by normal vector.		
Return	Error code		
Comments	User can input center point, normal vector, angle and direction to execute relative 3D arc interpolation. Please refer to 3-Axis Arc Interpolation section of Chapter 9.3.3.2.		
Format Parameters			
Name	Type	IN or OUT	Description
GroupHandle	HAND	IN	Group handle from Acm_GpAddaxis.
CenterArray	PF64	IN	Axis relative distance of center point.
NVectorArray	PF64	IN	Normal vector of reference plane.
Degree	F64	IN	Angle of rotation. (Range: 0~360)
pArrayElements	PU32	IN/OUT	Element count in the array (This count must equal to the axis count in thisgroup, or else it will be returned axiscount in group).
Direction	I16	IN	Direction: 0: DIR_CW(clockwise) 1: DIR_CCW(counterclockwise)

A.5.4.14 **Acm_GpMove3DArcAbs_V**

Format	U32 Acm_GpMove3DArcAbs_V(HAND GroupHandle, PF64 CenterArray, PF64 NVectorArray, F64 Degree, PU32 pArrayElements, I16 Direction)		
Purpose	Command group to execute absolute 3D arc interpolation by normal vector		
Return	Error code		
Comments	User can input center point, normal vector, angle and direction to execute absolute 3D arc interpolation. Please refer to 3-Axis Arc Interpolation section of Chapter 9.3.3.2.		
Format Parameters			
Name	Type	IN or OUT	Description
GroupHandle	HAND	IN	Group handle from Acm_GpAddaxis.
CenterArray	PF64	IN	Absolute distance of center point.
NVectorArray	PF64	IN	Normal vector of reference plane.
Degree	F64	IN	Angle of rotation. (Range: 0~360)
pArrayElements	PU32	IN/OUT	Element count in the array (This count must equal to the axis count in this group, or else it will be returned axis count in group).
Direction	I16	IN	Direction: 0: DIR_CW(clockwise) 1: DIR_CCW(counterclockwise)

A.5.4.15 Acm_GpMoveHelixRel

Format	U32 Acm_GpMoveHelixRel (HAND GroupHandle, PF64 CenterArray, PF64 EndArray, PU32 pArrayElements, I16 Direction)		
Purpose	Command group to move relative spiral.		
Return	Error code		
Comments	See about Acm_GpMoveHelixAbs.		
Format Parameters			
Name	Type	IN or OUT	Description
GroupHandle	HAND	IN	Group handle from Acm_GpAddaxis.
CenterArray	PF64	IN	Relative distance of center point.
EndArray	PF64	IN	Relative distance of end point.
pArrayElements	PU32	IN/OUT	Element count in the array (This count must equal to the axis count in this group, or else it will be returned axis count in group).
Direction	I16	IN	Direction: 0: DIR_CW (clockwise) 1: DIR_CCW (counterclockwise)

A.5.4.16 Acm_GpMoveHelixAbs

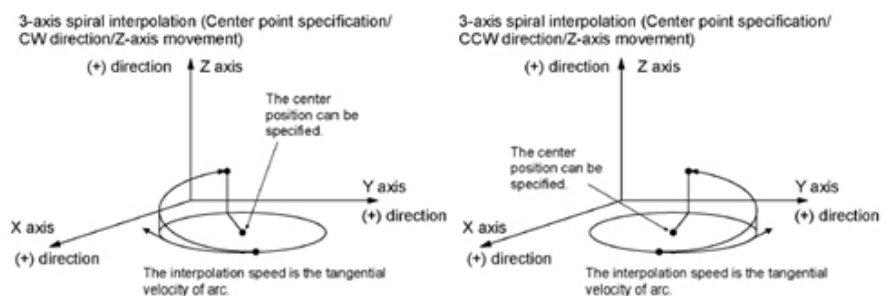
Format	U32 Acm_GpMoveHelixAbs (HAND GroupHandle, PF64 CenterArray, PF64 EndArray, PU32 pArrayElements, I16 Direction)		
Purpose	Command group to move absolute spiral.		
Return	Error code		

This command just supports 3 axes. The elements in CenterArray and EndArray must be in order with PhysicalID of axis. User can choose two axes in group to move arc interpolation by property PAR_GpRefPlane; the other axes decides the height of helix. The unit of distance in CenterArray and EndArray is PPU of each axis in group.

For example:

Group (Y, Z, U), CenterArray (Y, Z, U), EndArray (Y, Z, U). If PAR_GpRefPlane =1(YZ_Plane), Z axis and U axis will move arc interpolation, and Y value in EndArray is the height of helix.

Comments



Format Parameters

Name	Type	IN or OUT	Description
GroupHandle	HAND	IN	Group handle from Acm_GpAddaxis.
CenterArray	PF64	IN	Absolute distance of center point.
EndArray	PF64	IN	Absolute distance of end point.
pArrayElements	PU32	IN/OUT	Element count in the array (This count must be equal to the axis count in this group, or else it will be returned axis count in group).

Direction	I16	IN	Direction: 0: DIR_CW (clockwise) 1: DIR_CCW (counterclockwise)
-----------	-----	----	----------------------------------------------------------------------

A.5.4.17 Acm_GpMoveHelixRel_3P

Format	U32 Acm_GpMoveHelixRel_3P (HAND GroupHandle, PF64 RefArray, PF64 EndArray, PU32 pArrayElements, I16 Direction)		
Purpose	Command group to move relative spiral by three specified points.		
Return	Error code		
Comments	See about Acm_GpMoveHelixAbs_3P.		
Format Parameters			
Name	Type	IN or OUT	Description
GroupHandle	HAND	IN	Group handle from Acm_GpAddaxis.
RefArray	PF64	IN	Relative distance of reference point.
EndArray	PF64	IN	Relative distance of end point.
pArrayElements	PU32	IN/OUT	Element count in the array (This count must equal to the axis count in this group, or else it will be returned axis count in group).
Direction	I16	IN	Direction: 0: DIR_CW(clockwise) 1: DIR_CCW (counterclockwise)

A.5.4.18 Acm_GpMoveHelixAbs_3P

Format	U32 Acm_GpMoveHelixAbs_3P (HAND GroupHandle, PF64 RefArray, PF64 EndArray, PU32 pArrayElements, I16 Direction)		
Purpose	Command group to move absolute spiral by three specified points.		
Return	Error code		
Comments	<p>There must be 3 axes in group. Command helix motion by assigning three points. The orders of value in parameter RefArray, CenterArray and EndArray must follow the order of axis PhysicalID. The unit of distance in RefArray and EndArray is PPU of each axis in group. User can choose two axes in group to move arc interpolation by PAR_GpRefPlane. For example: Group(Y, Z, U), RefArray(Y, Z, U), CenterArray(Y, Z, U), EndArray(Y, Z, U), PAR_GpRefPlane =1(YZ_Plane). Z axis and U axis will move arc interpolation, and Y value in EndArray is the height of helix.</p>		
Format Parameters			
Name	Type	IN or OUT	Description
GroupHandle	HAND	IN	Group handle from Acm_GpAddaxis.
RefArray	PF64	IN	Absolute distance of reference point.
EndArray	PF64	IN	Absolute distance of end point.
pArrayElements	PU32	IN/OUT	Element count in the array (This count must equal to the axis count in this group, or else it will be returned axis count in group).
Direction	I16	IN	Direction: 0: DIR_CW (clockwise) 1: DIR_CCW (counterclockwise)

A.5.4.19 Acm_GpMoveHelixRel_Angle

Format	U32 Acm_GpMoveHelixRel_Angle(HAND GroupHandle,PF64 CenterArray, PF64 EndArray, PU32 ArrayElements,I16 Direction)		
Purpose	Complete helical interpolation through relative center coordinates, angle, end point and direction of rotation.		
Return	Error code		
Comments	The rotation angle must be filled in EndArray[0~2] determined by reference plane which can be set through Par_GpRefPlane and the others parameter are end points. Take the example of method 2 in chapter 9.3.4.2, the reference plane is XY plane, then the EndArray[0] and EndArray[1] are rotation angle which the value is 1,080 and the EndArray[2] is 10,000 which is the end point of Z-axis.		
Format Parameters			
Name	Type	IN or OUT	Description
GroupHandle	HAND	IN	Group handle from Acm_GpAddaxis.
CenterArray	PF64	IN	Relative distance between center point and start point.
EndArray	PF64	IN	Angle of rotation (Unit: degree) and relative distance of end point.
pArrayElements	PU32	IN	Axis count.
Direction	I16	IN	Direction: 0: DIR_CW(clockwise) 1: DIR_CCW (counterclockwise)

A.5.4.20 Acm_GpMoveHelixAbs_Angle

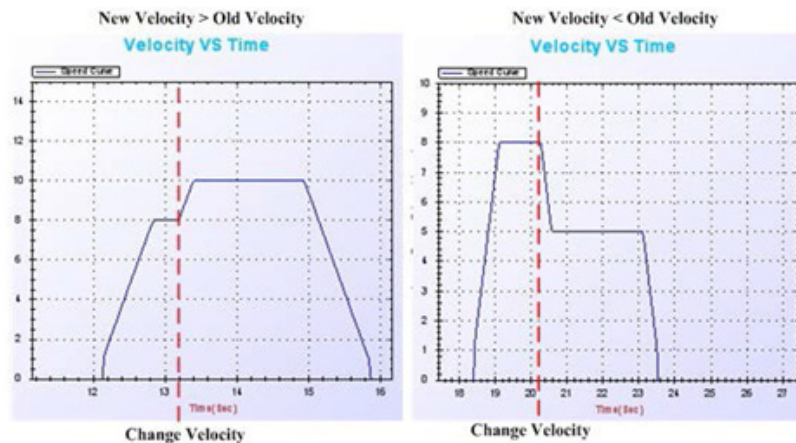
Format	U32 Acm_GpMoveHelixAbs_Angle(HAND GroupHandle,PF64 CenterArray, PF64 EndArray, PU32 ArrayElements,I16 Direction)		
Purpose	Complete helical interpolation through relative center coordinates, angle, end point and direction of rotation.		
Return	Error code		
Comments	The rotation angle must be filled in EndArray[0~2] determined by reference plane which can be set through Par_GpRefPlane and the others parameter are end points. Take the example of method 2 in chapter 9.3.4.2, the reference plane is XY plane, then the EndArray[0] and EndArray[1] are rotation angle which the value is 1080 and the EndArray[2] is 10,000 which is the end point of Z-axis.		
Format Parameters			
Name	Type	IN or OUT	Description
GroupHandle	HAND	IN	Group handle from Acm_GpAddaxis.
CenterArray	PF64	IN	Center coordinates.
EndArray	PF64	IN	Angle of rotation (Unit: degree) and absolute distance of end point.
ArrayElements	PU32	IN	Axis count.
Direction	I16	IN	Direction: 0: CW-Dir 1: CCW_Dir

A.5.4.21 Acm_GpChangeVel

Format	U32 Acm_GpChangeVel (HAND GroupHandle, F64 NewVelocity)
Purpose	Command group to change the velocity while group is in line interpolation motion.
Return	Error code

When group is in motion status, the velocity changing command can be sent, in order for the motion card to change the velocity accordingly. When group is in line interpolation, circular interpolation, helical interpolation or continuous interpolation motion, this command can be sent to change the velocity. If the command runs successfully, NewVelocity will be used in next motion if the velocity is not specified before the motion.

- (1). In single step interpolation motion, the velocity changing process is shown below:



If the remaining pulse is not enough for getting to the new velocity, the card will automatically calculate the available velocity.

- (2). In continuous interpolation motion

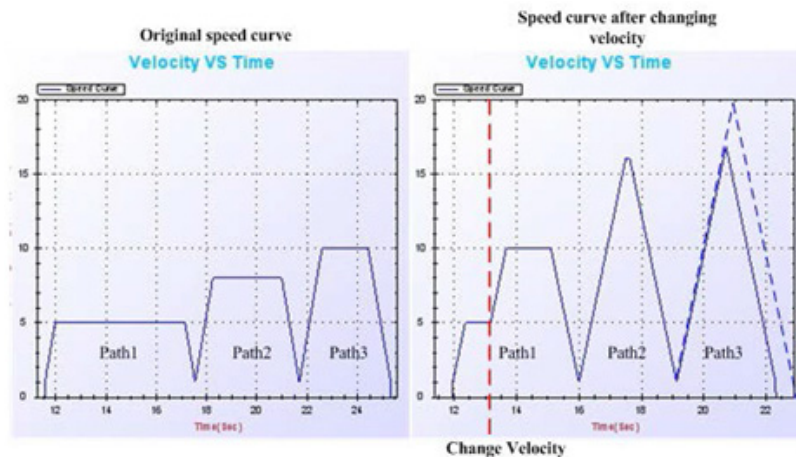
Comments

- BufferMode: Blending Disabled

In this mode, each path in Path Buffer has its own process of acceleration/ deceleration. If the ChangeV function is run in motion status, then the velocity of path of the current phase will accelerate/decelerate to new velocity, and thus the velocity of each phase will increase/decrease proportionally. If the remaining pulse is not enough when the velocity of each phase is being changed, then the new velocity will be automatically calculated.

For example: path1: VL = 1,000, VH = 5,000
 path 2: VL = 1,000, VH = 8,000
 path 3: VL = 1,000, VH = 10,000

During Path1, if ChangeV is run and New Velocity = 10,000, then velocity of the second path should be 16,000, and velocity of the third path should be 20,000.



However, the real status is as below:

- BlendingMode: Blending Enable, BlendingTime >0

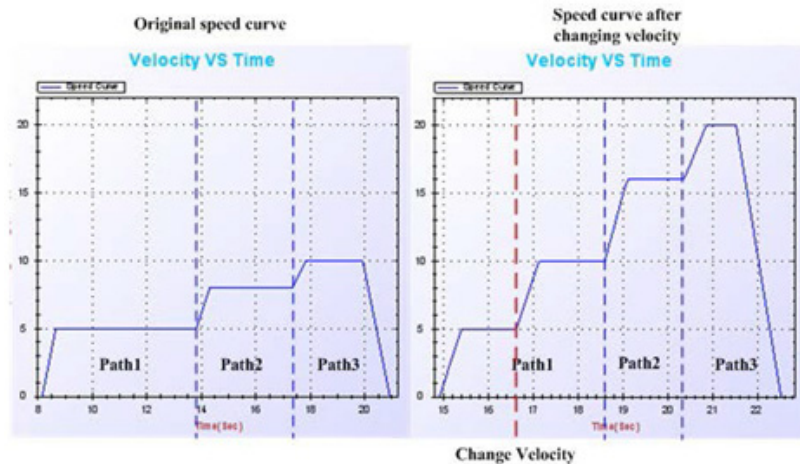
In this mode, each path in Path Buffer doesn't have a complete process of acceleration/deceleration. Its velocity path is decided by BlendingTime and the FL of each phase. For details, please refer to Acm_GpAddPath. If the ChangeV function is run in motion status, then the velocity of path of the current phase will accelerate/decelerate to new velocity, and thus the velocity of each phase will increase/decrease proportionally. If the remaining pulse is not enough when the velocity of each phase is being changed, then the new velocity will be automatically calculated.

For example: path1: VL = 1,000, VH = 5,000

path 2: VL = 1,000, VH = 8,000

path 3: VL = 1,000, VH = 10,000

During Path1, if ChangeV is run and New Velocity = 10,000, then velocity of the second path should be 16,000, and velocity of the third path should be 20,000. However, the real status is as below:



If the velocity changing command is sent in Blending phase, then the ChangeV function will be delayed to next path.

- FlyMode: Blending Enable, BlendingTime=0

The velocity path in this mode is similar to that in Blending mode, while the velocity between two paths will not decelerate to FL. For details, please refer to Acm_GpAddPath. If the ChangeV function is run in motion status, then the velocity of path of the current phase will accelerate/decelerate to new velocity, and thus the velocity of each phase will increase/decrease proportionally. If the remaining pulse is not enough when the velocity of each phase is being changed, then the new velocity will be automatically calculated.

Note:

The ChangeV function is not supported by deceleration phase.

Format Parameters

Name	Type	IN or OUT	Description
GroupHandle	HAND	IN	Group handle from Acm_GpAddaxis.
NewVelocity	F64	IN	New velocity. (unit: PPU/s)

A.5.4.22 Acm_GpChangeVelByRate

Format	U32 Acm_GpChangeVelByRate(HAND GroupHandle, U32 Rate)		
Purpose	Change the velocity of the current group motion according to the specified ratio.		
Return	Error code		
Comments	New velocity = Former velocity of group * rate *0.01. Rate must be larger than zero, and less than the ratio of the maximum velocity to current group velocity of the axis with the lowest ID. New velocity is valid for the current motion only. For more details about changing velocity in interpolation or continuous interpolation motion, please refer to Acm_GpChangeVel.		
Format Parameters			
Name	Type	IN or OUT	Description
GroupHandle	HAND	IN	Group handle from Acm_GpAddaxis.

A.5.5 Path

A.5.5.1 Acm_GpAddPath

Format	U32 Acm_GpAddPath (HAND GroupHandle, U16 MoveCmd, U16 Move-Mode,F64 FH, F64 FL, PF64 EndPoint_DataArray, PF64 CenPoint_DataArray,PU32 ArrayElements)		
Purpose	Add an interpolation path to system path buffer.		
Return	Error code		
Comments	Please refer to Chapter 9.5.		
Format Parameters			
Name	Type	IN or OUT	Description
GroupHandle	HAND	IN	Group handle from Acm_GpAddaxis.
MoveCmd	U16	IN	Move command, please refer to Chapter 9.5.
MoveMode	U16	IN	Move mode: 0: No blending 1: Blending
FH	F64	IN	High velocity / delay time for GPDELAY move command. (driving velocity) (Unit:PPU/s of group)
FL	F64	IN	Low velocity (start velocity) (Unit:PPU/s of group)
EndPoint_DataArray	PF64	IN	End points (Unit: PPU of each axis)
CenPoint_DataArray	PF64	IN	Center points (Unit: PPU of each axis)
ArrayElements	PU32	IN/OUT	Number of array element cannot be less than axis count in group, or else it will be returned axis count in group.

A.5.5.2 Acm_GpResetPath

Format	U32 Acm_GpResetPath (PHAND GroupHandle)		
Purpose	Clear system path buffer. If there is group executing path, the path motion will be stopped.		
Return	Error code		
Comments			
Format Parameters			
Name	Type	IN or OUT	Description
GroupHandle	PHAND	IN	Group handle from Acm_GpAddaxis.

A.5.5.3 Acm_GpLoadPath

Format	U32 Acm_GpLoadPath(HAND GroupHandle, PI8 FilePath,PHAND PathHandle, PU32 pTotalCount)		
Purpose	Load path data from path file. It can load up to 600 path data one time.		
Return	Error code		
Comments	The path data file (binary) is usually generated by Motion Utility's [Path Editor]. If you are familiar with Advantech motion product, you can create file by yourself. The PathHandle must be unloaded by Acm_GpUnloadPath when the PathHandle does not be used any more or application is closing, and the paths contained in PathHandle are deleted from driver at the same time.		
Format Parameters			
Name	Type	IN or OUT	Description
GroupHandle	HAND	IN	Group handle from Acm_GpAddaxis.
FilePath	PI8	IN	Point to a file path name of the motion path data to be loaded.
PathHandle	PHAND	OUT	Return the pointer to path handle
pTotalCount	PU32	OUT	Return actual to tal count of path data in the path file

A.5.5.4 Acm_GpUnloadPath

Format	U32 Acm_GpUnloadPath (HAND GroupHandle, PHAND PathHandle)		
Purpose	Unload path data.		
Return	Error code		
Comments			
Format Parameters			
Name	Type	IN or OUT	Description
GroupHandle	HAND	IN	Group handle from Acm_GpAddaxis.
PathHandle	PI8	IN	Pointer to path handle from Acm_GpLoadPath.

A.5.5.5 Acm_GpMovePath

Format	U32 Acm_GpMovePath (HAND GroupHandle, HAND PathHandle)		
Purpose	Start continuous interpolation motion (Path).		
Return	Error code		
Comments	<p>If the PathHandle is returned by Acm_GpLoadPath, the path data will be passed to system path buffer first, then driver start path motion. If the PathHandle is NULL, the path data in system path buffer will be executed directly.</p> <p>Note: Due to the path had been loaded into the system by Acm_GpLoadPath, the second parameter of Acm_GpMovePath API PathHandle might set to NULL, otherwise the path will be loaded twice time.</p>		
Format Parameters			
Name	Type	IN or OUT	Description
GroupHandle	HAND	IN	Group handle from Acm_GpAddaxis.
PathHandle	PHAND	IN	Pointer to path handle from Acm_GpLoadPath.

A.5.5.6 Acm_GpMoveAllPath

Format	U32 Acm_GpMoveAllPath(PHAND GroupHandle, U32 ArrayElements)		
Purpose	Start continuous interpolation motion (Path) for selected groups.		
Return	Error code		
Comments	Start multiple groups to do path table motion simultaneously. After load the path to each group, user can call this function and all paths in the specific group will run simultaneously		
Format Parameters			
Name	Type	IN or OUT	Description
GroupHandle	PHAND	IN	Group handle list from Acm_GpAddaxis.
ArrayElements	U32	IN	Count of groups which need to start to move simultaneously.

A.5.5.7 Acm_GpGetPathStatus

Format	U32 Acm_GpGetPathStatus (HAND GroupHandle, PU32 pCurIndex, PU32 pCurCmdFunc, PU32 pRemainCount, PU32 pFreeSpaceCount)		
Purpose	Get current status of path buffer.		
Return	Error code		
Comments	You must input the GroupHandle, and then get path status of this group.		
Format Parameters			
Name	Type	IN or OUT	Description
GroupHandle	HAND	IN	Group handle from Acm_GpAddaxis.
pCurIndex	PU32	OUT	Return current index of path data in path buffer
pCurCmdFunc	PU32	OUT	Return current command function in executing.
pRemainCount	PU32	OUT	Return number of unexecuted path data in path.
pFreeSpace-Count	PU32	OUT	Return number of free space in path buffer.

A.5.5.8 Acm_GpMoveSelPath

Format	U32 Acm_GpMoveSelPath (HANDGroupHandle, HAND PathHandle, U32 StartIndex, U32EndIndex, U8Repeat)		
Purpose	Move path segment in system path buffer from start index and end index.		
Return	Error code		
Comments	<p>Command to move paths which index is between StartIndex and EndIndex. If PathHandle is null, it will move specified paths in system path buffer; If PathHandle is not null, the paths in PathHandle will be loaded in system path buffer firstly, then move specified paths.</p> <p>If the value of Repeat is zero, the specified paths will be executed continuously and repeatedly until stopping group motion.</p> <p>If value of EndIndex is greater than path count in system path buffer, it will move paths between StartIndex path and last index path in system path buffer.</p>		
Format Parameters			
Name	Type	IN or OUT	Description
GroupHandle	HAND	IN	Group handle from Acm_GpAddaxis.
PathHandle	HAND	IN	Pointer to path handle from Acm_GpLoadPath.
StartIndex	U32	IN	Start Index.(0~6,999)
EndIndex	U32	IN	End Index.(0~6,999)
Repeat	U8	IN	Repeat count. (0~255)

A.5.5.9 Acm_GpGetPathIndexStatus

Format	U32 Acm_GpGetPathIndexStatus (HAND GroupHandle, U32 Index, PU16 CmdFunc, PU16 MoveMode, PF64 FH, PF64 FL, F64 EndPoint_DataArray, PF64 CenPoint_DataArray, PU32 ArrayElements)		
Purpose	Get the status of specified index path in system path buffer.		
Return	Error code		
Comments	If you want to know the setting of a path, you can call this API.		
Format Parameters			
Name	Type	IN or OUT	Description
GroupHandle	HAND	IN	Group handle from Acm_GpAddaxis.
Index	U32	IN	Index of path.
CmdFunc	PU16	OUT	Return current command function in executing
MoveMode	PU16	OUT	Move mode: 0: No blending 1: Blending
FH	PF64	OUT	Unit: PPU of master Axis(the minimal PhysicalID Axis)
FL	PF64	OUT	Unit: PPU of master Axis(the minimal PhysicalID Axis)
EndPoint_DataArray	PF64	OUT	Unit: PPU of master Axis(the minimal PhysicalID Axis)
CenPoint_DataArray	PF64	OUT	Unit: PPU of master Axis(the minimal PhysicalID Axis)
ArrayElements	PU32	IN/OUT	Return axis count

A.5.6 Pause & Resume

A.5.6.1 Acm_GpPauseMotion

Format	U32 Acm_GpPauseMotion(HAND GroupHandle)		
Purpose	Pause group movement.		
Return	Error code		
Comments	Please refer to Chapter 9.5.7		
Format Parameters			
Name	Type	IN or OUT	Description
GroupHandle	HAND	IN	Group handle from Acm_GpAddaxis.

A.5.6.2 Acm_GpResumeMotion

Format	U32 Acm_GpResumeMotion(HAND GroupHandle)		
Purpose	Resume movement after pause		
Return	Error code		
Comments			
Format Parameters			
Name	Type	IN or OUT	Description
GroupHandle	HAND	IN	Group handle from Acm_GpAddaxis.

Appendix **B**

Property List

B.1 Device Features

FT_DevelopmentMap

Property Type	R/W	Property ID	Default Value
U32	R	0	-
Description	Get device supported interpolation types		

1: support, 0: Not support

Bits	Description	Macro (defined in AdvMotDrv.h)
1	Line interpolation, 3 axes	DEV_IPO_LINE_3AX
2	Line interpolation, 4 axes	DEV_IPO_LINE_4AX
3	Line interpolation, 5 axes	DEV_IPO_LINE_5AX
4	Line interpolation, 6 axes	DEV_IPO_LINE_6AX
5	Line interpolation, 7 axes	DEV_IPO_LINE_7AX
6	Line interpolation, 8 axes	DEV_IPO_LINE_8AX
7	Not defined.	
8	Arc interpolation, 2 axes	DEV_IPO_ARC_2AX
9	Arc interpolation, 3 axes	DEV_IPO_ARC_3AX
10	Spiral.	DEV_IPO_SPIRAL
11	Blending	DEV_IPO_BLENDED
12	Speed forward	DEV_IPO_SPEED_FORWARD
13	DO control	DEV_IPO_DO_CONTROL
14~15	Not defined.	
16	Synchronous electronic gear	DEV_IPO_GEAR
17	Synchronous electronic cam	DEV_IPO_CAM
18	Synchronous gantry	DEV_IPO_GANTRY
19	Synchronous tangent	DEV_IPO_TANGENT
20~23	Not defined.	
24	Select path.	DEV_IPO_SELPTH
25~31	Not defined.	

Comments

Example: Get device supported interpolation types

```

HAND m_Devhand; //Device Handle
U32 DevIpoType; //Property value
Acm_GetU32Property(m_Devhand,FT_DevelopmentMap,&DevIpoType);
If( DevIpoType& DEV_IPO_LINE_2AX)
{ //Support 2-axis line interpolation}
If(DevIpoType & DEV_IPO_LINE_3AX)
{ // Support 3-axis line interpolation}
If().....

```

FT_DevelopmentAxesCount

Property Type	R/W	Property ID	Default Value
U32	R	1	-
Description	Get axis number of this device.		
Comments	Read-only		

FT_DevFunctionMap

Property Type	R/W	Property ID	Default Value
U32	R	2	-
Description	Get device supported functions.		
Comments	Bits	Description	Macro (defined in AdvMotDrv.h)
	0	Motion	DEV_FUNC_MOT
	1	DI	DEV_FUNC_DI
	2	DO	DEV_FUNC_DO
	3	AI	DEV_FUNC_AI
	4	AO	DEV_FUNC_AO
	5	Timer	DEV_FUNC_TMR
	6	Counter	DEV_FUNC_CNT
	7	DAQ DI	DEV_FUNC_DAQDI
	8	DAQ DO	DEV_FUNC_DAQDO
	9	DAQ AI	DEV_FUNC_DAQAI
	10	DAQ AO	DEV_FUNC_DAQAO
	11	Emg	DEV_FUNC_EMG
	12	DEV_FUNC_MDAQ	DEV_FUNC_MDAQ
13~31	No definition		

Example: Get device supported functions

```

HAND m_Devhand; //Device Handle
U32 DevFunMap; //Property value
Acm_GetU32Property(m_Devhand,FT_DevFunctionMap,& DevFunMap);
If(DevFunMap & DEV_FUNC_MOT)
{ //Support motion function}
If(DevFunMap & DEV_FUNC_DI)
{ //Support DI function }
If().....

```

FT_DevOverflowCnt

Property Type	R/W	Property ID	Default Value
U32	R	3	2,147,483,647
Description	The maximum data count of position counter.		
Comments	The maximum data count is 2,147,483,647.		

FT_MasCyclicCnt_R0

Property Type	R/W	Property ID	Default Value
U32	R	4	-
Description	Get slave counts connected on the Motion ring.		
Comments			

FT_MasCyclicCnt_R1

Property Type	R/W	Property ID	Default Value
U32	R	5	-
Description	Get slave counts connected on the IO ring.		
Comments			

FT_DevMDAQTypeMap

Property Type	R/W	Property ID	Default Value
U32	R	6	-
Description	Data types that MotionDAQ supports.		
	Read-only		

Comments

Bits	Description	Macro (defined in AdvMotDrv.h)
0	Command Position.	MQ_TYPE_CMDPOSI
1	Actual Position.	MQ_TYPE_ACTPOSI
2	Lag Position (difference between Command Position and Actual Position).	MQ_TYPE_LAGPOSI
3	Command Velocity.	MQ_TYPE_CMDVEL

Example: Get MotionDAQ supports Data types

```
HAND m_Devhand; //Device Handle
U32  MDAQTypeMap; //Property value
Acm_GetU32Property(m_Devhand,FT_DevMDAQTypeMap,& MDAQTypeMap);
If(MDAQTypeMap & MQ_TYPE_CMDPOSI)
{ //Support command position}
If(MDAQTypeMap & MQ_TYPE_ACTPOSI)
{ //Support actual position}
If(MDAQTypeMap & MQ_TYPE_LAGPOSI)
{ //Support lag position}
```

FT_DevMDAQTrigMap

Property Type	R/W	Property ID	Default Value
U32	R	7	-
Description	The methods to trigger MotionDAQ function.		
	Read-only		

Comments

Bits	Description	Macro (defined in AdvMotDrv.h)
0	Disable MotionDAQ function.	MP_MQ_TRIG_DISABLE
1	Software command trigger mode (i.e. Start command is issued to trigger).	MP_MQ_TRIG_SW
2	DI trigger mode.	MP_MQ_TRIG_DI
3	Specify axis motion to start, i.e. trigger MotionDAQ function.	MP_MQ_TRIG_AX_START

Example: GetMotionDAQ triggermethods

```

HAND m_Devhand; //Device Handle
U32 MDAQTrigMap; //Property value
Acm_GetU32Property(m_Devhand,FT_DevMDAQTrigMap,& MDAQTrigMap);
If(MDAQTrigMap & MP_MQ_TRIG_DISABLE)
{ //Support disable MotionDAQ function}
If (MDAQTrigMap & MP_MQ_TRIG_SW)
{ //Support software command trigger mode}
If (MDAQTrigMap & MP_MQ_TRIG_DI)
{ //Support DI trigger mode}

```

FT_DevMDAQMaxChan

Property Type	R/W	Property ID	Default Value
U32	R	8	-
Description	Record max channel count of MotionDAQ data.		
Comments	Read-only		

FT_DevMDAQMaxBufCount

Property Type	R/W	Property ID	Default Value
U32	R	9	2000
Description	The max data count of MotionDAQ that each MotionDAQ channel can record.		
Comments	Read-only.		

B.2 Device Configurations**CFG_DevBoardID**

Property Type	R/W	Property ID	Default Value
U32	R	201	-
Description	Get Device ID.		
Comments	This property value will be 0~15		

CFG_DevBaseAddress

Property Type	R/W	Property ID	Default Value
U32	R	203	-
Description	Return IO base address.		
Comments	Read-only		

CFG_DevInterrupt

Property Type	R/W	Property ID	Default Value
U32	R	204	-
Description	Get Device interrupt number.		
Comments	Read-only		

CFG_DevBusNumber

Property Type	R/W	Property ID	Default Value
U32	R	205	-
Description	Get device bus number.		
Comments	Read-only		

CFG_DevSlotNumber

Property Type	R/W	Property ID	Default Value
U32	R	206	-
Description	Get device slot number.		
Comments	Read-only		

CFG_DevDriverVersion

Property Type	R/W	Property ID	Default Value
char*	R	207	-
Description	Get SYS driver's version.		
Comments	The format is: 1.0.0.1		

CFG_DevDllVersion

Property Type	R/W	Property ID	Default Value
char*	R	208	-
Description	Get DLL driver's version.		
Comments	The format is: 1.0.0.1.		

CFG_DevFwVersion

Property Type	R/W	Property ID	Default Value
char*	R	214	-
Description	Get the firmware version.		
Comments	The format is: 1.0.0.1.		

CFG_DevLogMsg

Property Type	R/W	Property ID	Default Value						
U32	R/W	229	false						
Description	Choose whether you want to log an error to file or not								
Comments	<table border="1"><thead><tr><th>Bits</th><th>Description</th></tr></thead><tbody><tr><td>0</td><td>No log</td></tr><tr><td>1</td><td>Log to .txt file</td></tr></tbody></table>			Bits	Description	0	No log	1	Log to .txt file
Bits	Description								
0	No log								
1	Log to .txt file								

B.3 DAQ Features

FT_DaqDiMaxChan

Property Type	R/W	Property ID	Default Value
U32	R	50	-
Description	Get all DI channel count in the network.		
Comments	Read-only. If user uses the default mapping relationship, the Acm_DevLoadMapFile need not to load, or else user should download mapfile use Acm_DevLoadMapFile before get this property.		

FT_DaqDoMaxChan

Property Type	R/W	Property ID	Default Value
U32	R	51	-
Description	Get all DO channel count in the network.		
Comments	Read-only. If user uses the default mapping relationship, the Acm_DevLoadMapFile need not to load, or else user should download mapfile use Acm_DevLoadMapFile before get this property.		

FT_DaqAiRangeMap

Property Type	R/W	Property ID	Default Value																																																
U32	R	52	-																																																
Description	Get the supported AI range. 1: support, 0: not support.																																																		
Comments	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> <th>Macro (defined in AdvMotDrv.h)</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>+/- 10V</td> <td>DAQ_AI_NEG_10V_TO_10V</td> </tr> <tr> <td>1</td> <td>+/- 5V</td> <td>DAQ_AI_NEG_5V_TO_5V</td> </tr> <tr> <td>2</td> <td>+/- 2.5V</td> <td>DAQ_AI_NEG_2500MV_TO_2500MV</td> </tr> <tr> <td>3</td> <td>+/- 1.25V</td> <td>DAQ_AI_NEG_1250MV_TO_1250MV</td> </tr> <tr> <td>4</td> <td>+/- 0.625V</td> <td>DAQ_AI_NEG_625MV_TO_625MV</td> </tr> <tr> <td>5</td> <td>+/-1V</td> <td>DAQ_AI_NEG_1V_TO_1V</td> </tr> <tr> <td>6</td> <td>+/-0.5V</td> <td>DAQ_AI_NEG_500MV_TO_500MV</td> </tr> <tr> <td>7</td> <td>+/-0.15V</td> <td>DAQ_AI_NEG_150MV_TO_150MV</td> </tr> <tr> <td>8</td> <td>0 ~ 10V</td> <td>DAQ_AI_NEG_0_TO_10V</td> </tr> <tr> <td>9</td> <td>0 ~ 500 mv</td> <td>DAQ_AI_NEG_0_TO_500MV</td> </tr> <tr> <td>10~15</td> <td>Not defined.</td> <td></td> </tr> <tr> <td>16</td> <td>0 ~ 20 mA</td> <td>DAQ_AI_0MA_TO_20MA</td> </tr> <tr> <td>17</td> <td>4 ~ 20 mA</td> <td>DAQ_AI_4MA_TO_20MA</td> </tr> <tr> <td>18</td> <td>+/-20mA</td> <td>DAQ_AI_NEG_20MA_TO_20MA</td> </tr> <tr> <td>19~31</td> <td>Not defined.</td> <td></td> </tr> </tbody> </table>			Bits	Description	Macro (defined in AdvMotDrv.h)	0	+/- 10V	DAQ_AI_NEG_10V_TO_10V	1	+/- 5V	DAQ_AI_NEG_5V_TO_5V	2	+/- 2.5V	DAQ_AI_NEG_2500MV_TO_2500MV	3	+/- 1.25V	DAQ_AI_NEG_1250MV_TO_1250MV	4	+/- 0.625V	DAQ_AI_NEG_625MV_TO_625MV	5	+/-1V	DAQ_AI_NEG_1V_TO_1V	6	+/-0.5V	DAQ_AI_NEG_500MV_TO_500MV	7	+/-0.15V	DAQ_AI_NEG_150MV_TO_150MV	8	0 ~ 10V	DAQ_AI_NEG_0_TO_10V	9	0 ~ 500 mv	DAQ_AI_NEG_0_TO_500MV	10~15	Not defined.		16	0 ~ 20 mA	DAQ_AI_0MA_TO_20MA	17	4 ~ 20 mA	DAQ_AI_4MA_TO_20MA	18	+/-20mA	DAQ_AI_NEG_20MA_TO_20MA	19~31	Not defined.	
Bits	Description	Macro (defined in AdvMotDrv.h)																																																	
0	+/- 10V	DAQ_AI_NEG_10V_TO_10V																																																	
1	+/- 5V	DAQ_AI_NEG_5V_TO_5V																																																	
2	+/- 2.5V	DAQ_AI_NEG_2500MV_TO_2500MV																																																	
3	+/- 1.25V	DAQ_AI_NEG_1250MV_TO_1250MV																																																	
4	+/- 0.625V	DAQ_AI_NEG_625MV_TO_625MV																																																	
5	+/-1V	DAQ_AI_NEG_1V_TO_1V																																																	
6	+/-0.5V	DAQ_AI_NEG_500MV_TO_500MV																																																	
7	+/-0.15V	DAQ_AI_NEG_150MV_TO_150MV																																																	
8	0 ~ 10V	DAQ_AI_NEG_0_TO_10V																																																	
9	0 ~ 500 mv	DAQ_AI_NEG_0_TO_500MV																																																	
10~15	Not defined.																																																		
16	0 ~ 20 mA	DAQ_AI_0MA_TO_20MA																																																	
17	4 ~ 20 mA	DAQ_AI_4MA_TO_20MA																																																	
18	+/-20mA	DAQ_AI_NEG_20MA_TO_20MA																																																	
19~31	Not defined.																																																		

Example: Get device supported AI range

```
HAND m_Devhand; //Device Handle
U32 AIRangeMap; //Property value
Acm_GetU32Property(m_Devhand,FT_DaqAiRangeMap,& AIRangeMap);
If(AIRangeMap & CFG_DAQ_AI_NEG_10V_TO_10V)
{ //Support +/- 10V }
If (AIRangeMap & CFG_DAQ_AI_NEG_5V_TO_5V)
{ //Support +/-5V}
If (AIRangeMap & CFG_DAQ_AI_NEG_2500MV_TO_2500MV)
{ //Support +/- 2.5V }
.....
```

FT_DaqAoRangeMap

Property Type	R/W	Property ID	Default Value																																				
U32	R	53	-																																				
Description	Get the supported AO range. 1: support, 0: not support.																																						
Comments	<table border="1"><thead><tr><th>Bit</th><th>Description</th><th>Macro (defined in AdvMotDrv.h)</th></tr></thead><tbody><tr><td>0</td><td>+/- 10V</td><td>DAQ_AO_NEG_10V_TO_10V</td></tr><tr><td>1</td><td>+/- 5V</td><td>DAQ_AO_NEG_5V_TO_5V</td></tr><tr><td>2</td><td>+/- 2.5V</td><td>DAQ_AO_NEG_2500MV_TO_2500MV</td></tr><tr><td>3</td><td>+/- 1.25V</td><td>DAQ_AO_NEG_1250MV_TO_1250MV</td></tr><tr><td>4</td><td>+/- 0.625V</td><td>DAQ_AO_NEG_625MV_TO_625MV</td></tr><tr><td>5</td><td>0 ~ 10 V</td><td>DAQ_AO_NEG_0V_TO_10V</td></tr><tr><td>6</td><td>0 ~ 5 V</td><td>DAQ_AO_NEG_0V_TO_5V</td></tr><tr><td>7~15</td><td>Reserved</td><td></td></tr><tr><td>16</td><td>0 ~ 20 mA</td><td>DAQ_AO_0MA_TO_20MA</td></tr><tr><td>17</td><td>4 ~ 20 mA</td><td>DAQ_AO_4MA_TO_20MA</td></tr><tr><td>18~31</td><td>Reserved</td><td></td></tr></tbody></table>			Bit	Description	Macro (defined in AdvMotDrv.h)	0	+/- 10V	DAQ_AO_NEG_10V_TO_10V	1	+/- 5V	DAQ_AO_NEG_5V_TO_5V	2	+/- 2.5V	DAQ_AO_NEG_2500MV_TO_2500MV	3	+/- 1.25V	DAQ_AO_NEG_1250MV_TO_1250MV	4	+/- 0.625V	DAQ_AO_NEG_625MV_TO_625MV	5	0 ~ 10 V	DAQ_AO_NEG_0V_TO_10V	6	0 ~ 5 V	DAQ_AO_NEG_0V_TO_5V	7~15	Reserved		16	0 ~ 20 mA	DAQ_AO_0MA_TO_20MA	17	4 ~ 20 mA	DAQ_AO_4MA_TO_20MA	18~31	Reserved	
Bit	Description	Macro (defined in AdvMotDrv.h)																																					
0	+/- 10V	DAQ_AO_NEG_10V_TO_10V																																					
1	+/- 5V	DAQ_AO_NEG_5V_TO_5V																																					
2	+/- 2.5V	DAQ_AO_NEG_2500MV_TO_2500MV																																					
3	+/- 1.25V	DAQ_AO_NEG_1250MV_TO_1250MV																																					
4	+/- 0.625V	DAQ_AO_NEG_625MV_TO_625MV																																					
5	0 ~ 10 V	DAQ_AO_NEG_0V_TO_10V																																					
6	0 ~ 5 V	DAQ_AO_NEG_0V_TO_5V																																					
7~15	Reserved																																						
16	0 ~ 20 mA	DAQ_AO_0MA_TO_20MA																																					
17	4 ~ 20 mA	DAQ_AO_4MA_TO_20MA																																					
18~31	Reserved																																						

FT_DaqAiMaxSingleChan

Property Type	R/W	Property ID	Default Value
U32	R	54	-
Description	Get all single-end AI channel count in the network.		
Comments	Read-only. If user uses the default mapping relationship, the Acm_DevLoadMapFile need not to load, or else user should download mapfile use Acm_DevLoadMapFile before getting this property.		

FT_DaqAiMaxDiffChan

Property Type	R/W	Property ID	Default Value
U32	R	55	-
Description	Get all differential AI channel count in the network.		
Comments	Read-only. If user uses the default mapping relationship, the Acm_DevLoadMapFile need not to load, or else user should download mapfile use Acm_DevLoadMapFile before getting this property.		

FT_DaqAoMaxChan

Property Type	R/W	Property ID	Default Value
U32	R	57	-
Description	Get all AO channel count in the network.		
Comments	Read-only. If user uses the default mapping relationship, the Acm_DevLoadMapFile need not to load, or else user should download mapfile use Acm_DevLoadMapFile before getting this property.		

FT_DaqCntMaxChan

Property Type	R/W	Property ID	Default Value
U32	R	59	-
Description	Get all CNT channel count in the network.		
Comments	Read-only. If user uses the default mapping relationship, the Acm_DevLoadMapFile need not to load, or else user should download mapfile use Acm_DevLoadMapFile before getting this property.		

B.4 DAQ Configurations**CFG_MasCycleTime**

Property Type	R/W	Property ID	Default Value
U32	R	261	500
Description	Get the cycle time of motion ring for data communication. (Unit: us)		
Comments	Read-only		

CFG_IoCycleTime

Property Type	R/W	Property ID	Default Value
U32	R	262	200
Description	Get the cycle time of IO ring for data communication. (Unit: us)		
Comments	Read-only		

B.5 DAQ Channel Configurations

Properties value can be gotten through

Acm_GetChannelProperty

Acm_GetMultiChannelProperty

and set properties value by

Acm_SetChannelProperty

Acm_SetMultiChannelProperty.

Channel ID of DI/DO/AI/AO should be passed into property.

All properties in this section, user should know the DI/DO port and AI/AO/CNT channel mapping information by utility. If user uses the default mapping relationship, the Acm_DevLoadMapFile need not to load, or else user should download mapfile using Acm_DevLoadMapFile.

CFG_CH_DaqDiInvertEnable

Property Type	R/W	Property ID	Default Value						
F64	R/W	1500	0						
Description	Set/Get DI Invert.								
Comments	<table border="1"><thead><tr><th>Value</th><th>Description</th></tr></thead><tbody><tr><td>0</td><td>Not Invert</td></tr><tr><td>1</td><td>Invert</td></tr></tbody></table>			Value	Description	0	Not Invert	1	Invert
	Value	Description							
	0	Not Invert							
1	Invert								

CFG_CH_DaqDiLowFilter

Property Type	R/W	Property ID	Default Value
F64	R/W	1501	0
Description	Set/Get down limit value of DI filter. Range: 0~65,535		
Comments	DI filter down limit value will copy to all DI as one of them has been successfully set for ADAM-E5017.		

CFG_CH_DaqDiHighFilter

Property Type	R/W	Property ID	Default Value
F64	R/W	1502	0
Description	Set/Get upper limit value of DI filter. Range: 0~65535		
Comments	DI filter upper limit value will copy to all DI as one of them has been successfully set for ADAM-E5017.		

CFG_CH_DaqDoFsvEnable

Property Type	R/W	Property ID	Default Value						
F64	R/W	1503	0						
Description	Set/Get DO Failure Safe Value								
Comments	<table border="1"><thead><tr><th>Value</th><th>Description</th></tr></thead><tbody><tr><td>0</td><td>As communication error, DO will be in Low Logic state</td></tr><tr><td>1</td><td>As communication error, DO will be in High Logic state</td></tr></tbody></table>			Value	Description	0	As communication error, DO will be in Low Logic state	1	As communication error, DO will be in High Logic state
	Value	Description							
	0	As communication error, DO will be in Low Logic state							
1	As communication error, DO will be in High Logic state								

CFG_CH_DaqAoRange

Property Type	R/W	Property ID	Default Value
F64	R/W	1504	-

Set/Get AO range

Description

Value	Description
0	+/- 10V
1	+/- 5V
2	+/- 2.5V (Not Support)
3	+/- 1.25V (Not Support)
4	+/- 0.625V (Not Support)
5	0~10V
6	0~5V
7~15	Reserved
16	0 ~ 20 mA
17	4~ 20 mA

Macro definition

Comments

Value	Description	Macro (defined in AdvMotDrv.h)
0x0	+/- 10 V	CFG_DAQ_AO_NEG_10V_TO_10V
0x1	+/- 5 V	CFG_DAQ_AO_NEG_5V_TO_5V
0x2	+/- 2.5 V	CFG_DAQ_AO_NEG_2500MV_TO_2500MV
0x3	+/- 1.25 V	CFG_DAQ_AO_NEG_1250MV_TO_1250MV
0x4	+/- 0.625V	CFG_DAQ_AO_NEG_625MV_TO_625MV
0x5	0~10V	CFG_DAQ_AO_NEG_0V_TO_10V
0x6	0~5V	CFG_DAQ_AO_NEG_0V_TO_5V
0x10	0~20mA	CFG_DAQ_AO_0MA_TO_20MA
0x11	4~20mA	CFG_DAQ_AO_4MA_TO_20MA

CFG_CH_DaqAiRange

Property Type	R/W	Property ID	Default Value
F64	R/W	1505	-

Set/Get AI channel input range

Value	Description
0	+/- 10V
1	+/- 5V (Only ADAM-E5017 Support)
2	+/- 2.5V (Not Support)
3	+/- 1.25V (Not Support)
4	+/- 0.625V (Not Support)
5	+/- 1V (Only ADAM-E5017 Support)
6	+/- 0.5V (Only ADAM-E5017 Support)
7	+/- 0.15V (Only ADAM-E5017 Support)
8	0~10V (Only ADAM-E5017UH Support)
9	0~500mV (Only ADAM-E5017UH Support)
10~15	Reserved
16	0 ~ 20 mA (Only ADAM-E5017UH Support)
17	4~ 20 mA
18	+/- 20 mA (Only ADAM-E5017 Support)

Description

AI range setting will copy to all channels for ADAM-E5017; Different from ADAM-E5017, input ranges can be set on each channel individually for ADAM-E5017UH

Value	Description	Macro (defined in AdvMotDrv.h)
0x0	+/- 10 V	CFG_DAQ_AI_NEG_10V_TO_10V
0x1	+/- 5 V	CFG_DAQ_AI_NEG_5V_TO_5V
0x2	+/- 2.5 V	CFG_DAQ_AI_NEG_2500MV_TO_2500MV
0x3	+/- 1.25 V	CFG_DAQ_AI_NEG_1250MV_TO_1250MV
0x4	+/- 0.625V	CFG_DAQ_AI_NEG_625MV_TO_625MV
0x5	+/-1V	CFG_DAQ_AI_NEG_1V_TO_1V
0x6	+/-0.5V	CFG_DAQ_AI_NEG_500MV_TO_500MV
0x7	+/-0.15V	CFG_DAQ_AI_NEG_150MV_TO_150MV
0x8	0 ~ 10V	CFG_DAQ_AI_NEG_0_TO_10V
0x9	0 ~ 500 mV	CFG_DAQ_AI_NEG_0_TO_500mV
0x10	0~20mA	CFG_DAQ_AI_0MA_TO_20MA
0x11	4~20mA	CFG_DAQ_AI_4MA_TO_20MA
0x12	+/-20mA	CFG_DAQ_AI_NEG_20MA_TO_20MA

Comments

CFG_CH_DaqAiEnable

Property Type	R/W	Property ID	Default Value
F64	R/W	1506	1

Description Enable/Disable this AI

Value	Description
0	Disable this AI channel
1	Enable this AI channel

Comments

CFG_CH_DaqAiIntegrationTime

Property Type	R/W	Property ID	Default Value						
F64	R/W	1507	0						
Description	Set/Get AI setting time								
Comments	<table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>60Hz</td> </tr> <tr> <td>1</td> <td>50Hz</td> </tr> </tbody> </table>			Value	Description	0	60Hz	1	50Hz
Value	Description								
0	60Hz								
1	50Hz								

CFG_CH_DaqAoFsv

Property Type	R/W	Property ID	Default Value									
F64	R/W	1508	-									
Description	Get AO Failure Safe Value or Set AO Failure Safe Value as current AO value. This property is only available for ADAM-E5024H											
Comments	<table border="1"> <thead> <tr> <th></th> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>Read</td> <td>--</td> <td>The AO output value when communication error</td> </tr> <tr> <td>Set</td> <td>1</td> <td>After set 1 as input value of CFG_CH_DaqAoFsv, the AO Failure Safe Value will be the current AO value of specific channel. As communication error, AO will output the Failure Safe Value.</td> </tr> </tbody> </table>				Value	Description	Read	--	The AO output value when communication error	Set	1	After set 1 as input value of CFG_CH_DaqAoFsv, the AO Failure Safe Value will be the current AO value of specific channel. As communication error, AO will output the Failure Safe Value.
	Value	Description										
Read	--	The AO output value when communication error										
Set	1	After set 1 as input value of CFG_CH_DaqAoFsv, the AO Failure Safe Value will be the current AO value of specific channel. As communication error, AO will output the Failure Safe Value.										

CFG_CH_DaqAoStartup

Property Type	R/W	Property ID	Default Value									
F64	R/W	1509	-									
Description	Get AO Startup Value or Set AO Startup Value as current AO value. This property is only available for ADAM-E5024H											
Comments	<table border="1"> <thead> <tr> <th></th> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>Read</td> <td>--</td> <td>The AO output value after ADAM-5000/ECAT power-on</td> </tr> <tr> <td>Set</td> <td>1</td> <td>After set 1 as input value of CFG_CH_DaqAoStartup, the AO Startup Value will be the current AO value of specific channel. As the ADAM-5000/ECAT power-on, AO will output the Startup Value.</td> </tr> </tbody> </table>				Value	Description	Read	--	The AO output value after ADAM-5000/ECAT power-on	Set	1	After set 1 as input value of CFG_CH_DaqAoStartup, the AO Startup Value will be the current AO value of specific channel. As the ADAM-5000/ECAT power-on, AO will output the Startup Value.
	Value	Description										
Read	--	The AO output value after ADAM-5000/ECAT power-on										
Set	1	After set 1 as input value of CFG_CH_DaqAoStartup, the AO Startup Value will be the current AO value of specific channel. As the ADAM-5000/ECAT power-on, AO will output the Startup Value.										

B.6 Axis Features

B.6.1 System

B.6.1.1 FT_AxFunctionMap

Property Type	R/W	Property ID	Default Value
U32	R	301	-
Description	Get the axis supported function.		

1: support, 0: not support

Comments

Bits	Description	Macro (defined in AdvMotDrv.h)
0	In position.	AX_FUNC_INP
1	Alarm	AX_FUNC_ALM
2	Clear the deflection counter in the servo driver.	AX_FUNC_ERC
3	Slow down	AX_FUNC_SD
4	Hardware limit switch	AX_FUNC_EL
5	Software limit switch	AX_FUNC_SW_EL
6	Home sensor	AX_FUNC_ORG
7	Encode Z phase sensor	AX_FUNC_EZ
8	Backlash corrective.	AX_FUNC_BACKLASH_CORRECT
9	Suppress vibration.	AX_FUNC_SUPPRESS_VIBRATION
10	Home	AX_FUNC_HOME
11	Impose	Ax_FUNC_IMPOSE
12	Compare	Ax_FUNC_CMP
13	Latch	Ax_FUNC_LATCH
14	CAMDO	Ax_FUNC_CAMDO
15	Ext-Drive	Ax_FUNC_EXTRDRV
16	Simultaneous start/stop	Ax_FUNC_SIMSTART
17	IN1 stop	AX_FUNC_IN1_STOP
18	IN1 stop	AX_FUNC_IN2_STOP
19	IN3 stop	AX_FUNC_IN3_STOP
20	IN4 stop	AX_FUNC_IN4_STOP
21	IN5 stop	AX_FUNC_IN5_STOP
22~31	Not defined.	

Example: Get the axis supported function

```
HAND m_Axhand; //Axis Handle
U32 AxFunctionMap; //Property value
Acm_GetU32Property(m_Devhand,FT_AxFunctionMap, AxFunctionMap);//Set AI
range
If(AxFunctionMap & AX_FUNC_INP)
{ //Support in position function}
```


B.6.2 Speed Pattern

B.6.2.1 FT_AxMaxVel

Property Type	R/W	Property ID	Default Value
F64	R	302	20,000,000
Description	Get axis supported max velocity. (Unit: Pulse/s)		
Comments			

B.6.2.2 FT_AxMaxAcc

Property Type	R/W	Property ID	Default Value
F64	R	303	2,000,000,000
Description	Get axis supported max acceleration. (Unit: Pulse/s ²)		
Comments			

B.6.2.3 FT_AxMaxDec

Property Type	R/W	Property ID	Default Value
F64	R	304	2,000,000,000
Description	Get axis supported max deceleration (Unit: Pulse/s ²)		
Comments			

B.6.2.4 FT_AxMaxJerk

Property Type	R/W	Property ID	Default Value
F64	R	305	1
Description	Get axis supported max jerk. (Unit: Pulse/s ³)		
Comments			

B.6.3 Pulse In

B.6.3.1 FT_AxPulseInMap

Property Type	R/W	Property ID	Default Value										
U32	R	306	0										
Description	Get the pulse input features supported by this motion device. 1: support, 0: Not support												
Comments	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Mode</td> </tr> <tr> <td>1</td> <td>Logic</td> </tr> <tr> <td>2</td> <td>Source</td> </tr> <tr> <td>3~31</td> <td>Not defined.</td> </tr> </tbody> </table>			Bits	Description	0	Mode	1	Logic	2	Source	3~31	Not defined.
Bits	Description												
0	Mode												
1	Logic												
2	Source												
3~31	Not defined.												

B.6.3.2 FT_AxPulseInModeMap

Property Type	R/W	Property ID	Default Value																					
U32	R	307	0																					
Description	Get axis supported pulse input mode.																							
	1: support, 0: Not support																							
Comments	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> <th>Macro (defined in AdvMotDrv.h)</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>1X A/B</td> <td>AB_1X</td> </tr> <tr> <td>1</td> <td>2X A/B</td> <td>AB_2X</td> </tr> <tr> <td>2</td> <td>4X A/B</td> <td>AB_4X</td> </tr> <tr> <td>3</td> <td>CW/CCW</td> <td>I_CW_CCW</td> </tr> <tr> <td>4</td> <td>PULSE/DIR</td> <td>I_PULSE_DIR</td> </tr> <tr> <td>5 ~ 31</td> <td>Not defined.</td> <td></td> </tr> </tbody> </table>			Bits	Description	Macro (defined in AdvMotDrv.h)	0	1X A/B	AB_1X	1	2X A/B	AB_2X	2	4X A/B	AB_4X	3	CW/CCW	I_CW_CCW	4	PULSE/DIR	I_PULSE_DIR	5 ~ 31	Not defined.	
	Bits	Description	Macro (defined in AdvMotDrv.h)																					
	0	1X A/B	AB_1X																					
	1	2X A/B	AB_2X																					
	2	4X A/B	AB_4X																					
	3	CW/CCW	I_CW_CCW																					
4	PULSE/DIR	I_PULSE_DIR																						
5 ~ 31	Not defined.																							

Example: Get axis supported pulse input mode

```

HAND m_Axhand; //Axis Handle
U32 AxPulseInMode; //Property value
Acm_GetU32Property(m_Axhand,FT_AxPulseInModeMap, AxPulseInMode);//Get
pulse input mode
If(AxPulseInMode& AB_1X)
{ //Support 1X A/Bpulse input mode }

```

B.6.4 Pulse Out

B.6.4.1 FT_AxPulseOutMap

Property Type	R/W	Property ID	Default Value						
U32	R	308	0						
Description	Get the pulse output features supported by this motion device.								
Comments	<table border="1"> <thead> <tr> <th>Bits</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Mode</td> </tr> <tr> <td>1~31</td> <td>Not defined.</td> </tr> </tbody> </table>			Bits	Description	0	Mode	1~31	Not defined.
	Bits	Description							
	0	Mode							
1~31	Not defined.								

B.6.4.2 FT_AxPulseOutModeMap

Property Type	R/W	Property ID	Default Value
U32	R	309	0
Description		Get pulse output modes supported by this motion device.	

1: support, 0: Not support

Bits	Description	Macro (defined in AdvMotDrv.h)
0	OUT/DIR	OUT_DIR
1	OUT/DIR, OUT negative logic	OUT_DIR_OUT_NEG
2	OUT/DIR, DIR negative logic	OUT_DIR_DIR_NEG
3	OUT/DIR, OUT&DIR negative	OUT_DIR_ALL_NEG
4	CW/CCW	O_CW_CCW
5	CW/CCW, CW&CCW negative	CW_CCW_ALL_NEG
6	A/B Phase	AB_PHASE
7	B/A Phase	BA_PHASE
8	CW/CCW, OUT negative logic.(Not support)	CW_CCW_OUT_NEG
9	CW/CCW, DIR negative logic.(Not support)	CW_CCW_DIR_NEG
10~31	Not defined.	

Bits	Description	Positive direction		Negative direction	
		OUT output	DIR output	OUT output	DIR output
0	OUT/DIR		High		Low
1	OUT/DIR, OUT negative logic		High		Low
2	OUT/DIR, DIR negative logic		Low		High
3	OUT/DIR, OUT&DIR negative logic		Low		High
4	CW/CCW		High	High	
5	CW/CCW, CW&CCW negative logic		Low	Low	
6	A/B Phase				
7	B/A Phase				

Example: Get axis supported pulse output mode

```

HAND m_Axhand; //Axis Handle
U32 AxPulseOutMode; //Property value
Acm_GetU32Property(m_Axhand,FT_AxPulseOutModeMap, AxPulseOutMode);//
Get pulse output mode
If(AxPulseOutMode& OUT_DIR)
{//Support OUT/DIRpulse output mode}

```

B.6.5 Alarm

B.6.5.1 FT_AxAlmMap

Property Type	R/W	Property ID	Default Value
U32	R	310	-
Description	Get the alarm features supported by this motion axis.		
Comments	Bits	Description	
	0	Enabled	
	1	Logic	
	2	React	
	3~31	Not defined.	

B.6.6 In Position

B.6.6.1 FT_AxInpMap

Property Type	R/W	Property ID	Default Value
U32	R	311	-
Description	Get the In-Position features supported by this motion axis.		
Comments	Bits	Description	
	0	Mode	
	1	Logic	
	2~31	Not defined.	

B.6.7 ERC

B.6.7.1 FT_AxErcMap

Property Type	R/W	Property ID	Default Value
U32	R	312	0
Description	Get the ERC features supported by this motion axis.		
Comments	Bits	Description	
	0	Enable mode	
	1	Logic	
	2	On time(not support)	
	3	Off time(not support)	
	4~31	Not defined.	

B.6.7.2 FT_AxErcEnableModeMap

Property Type	R/W	Property ID	Default Value
U32	R	313	0
Description	Get axis supported ERC mode.		
Comments	Bits	Description	
	0	ERC Output when home finish	
	1	ERC Output when EMG/ALM/EL active	
	2	ERC Output when home finish or EMG/ALM/EL active	
	3~31	Not defined.	

B.6.8 Hardware Limit**B.6.8.1 FT_AxEIMap**

Property Type	R/W	Property ID	Default Value
U32	R	317	-
Description	Get the hardware end limit (EL) features supported by this motion axis.		
Comments	Bits	Description	
	0	Enabled	
	1	Logic	
	2	React	
	3~31	Not defined.	

B.6.9 Software Limit**B.6.9.1 FT_AxSwMeIMap**

Property Type	R/W	Property ID	Default Value
U32	R	318	-
Description	Get the software minus limit features supported by the motion axis.		
Comments	Bits	Description	
	0	Enabled	
	1	Logic	
	2	Value	
	3~31	Not defined.	

B.6.9.2 FT_AxSwPelMap

Property Type	R/W	Property ID	Default Value
U32	R	319	-
Description	Get the software plus limit features supported by the motion axis.		
Comments	Bits	Description	
	0	Enabled	
	1	Logic	
	2	Value	
	3~31	Not defined.	

B.6.10 Home

B.6.10.1 FT_AxHomeMap

Property Type	R/W	Property ID	Default Value
U32	R	320	127
Description	Get the home features supported by this motion axis.		
Comments	Bits	Description	
	0	Home mode	
	1	ORG logic	
	2	EZ logic	
	3	Reset Enable	
	4	ORG high low level	
	5	HomeOffsetDistance	
	6	HomeSpeed	

B.6.10.2 FT_AxHomeModeMap

Property Type	R/W	Property ID	Default Value
U32	R	332	-
Description	The supported Home return modes.		

About detailed information about each mode, see about chapter 9.2.9

Comments	Bits	Description	Macro (defined in AdvMotDrv.h)
	0	MP_MODE1_Abs	MODE1_Abs
	1	MP_MODE2_Lmt	MODE2_Lmt
	2	MP_MODE3_Ref	MODE3_Ref
	3	MP_MODE4_Abs_Ref	MODE4_Abs_Ref
	4	MP_MODE5_Abs_NegRef	MODE5_Abs_NegRef
	5	MP_MODE6_Lmt_Ref	MODE6_Lmt_Ref
	6	MP_MODE7_AbsSearch	MODE7_AbsSearch
	7	MP_MODE8_LmtSearch	MODE8_LmtSearch
	8	MP_MODE9_AbsSearch_Ref	MODE9_AbsSearch_Ref
	9	MP_MODE10_AbsSearch_NegRef	MODE10_AbsSearch_NegRef
	10	MP_MODE11_LmtSearch_Ref	MODE11_LmtSearch_Ref
	11	MP_MODE12_AbsSearchReFind	MODE12_AbsSearchReFind
	12	MP_MODE13_LmtSearchReFind	MODE13_LmtSearchReFind
	13	MP_MODE14_AbsSearchReFind_Ref	MODE14_AbsSearchReFind_Ref
	14	MP_MODE15_AbsSearchReFind_NegRef	MODE15_AbsSearchReFind_NegRef
	15	MP_MODE16_LmtSearchReFind_Ref	MODE16_LmtSearchReFind_Ref

Example: Get supported Home return modes

```

HAND m_Axhand; //Axis Handle
U32 AxHomeMode; //Property value
Acm_GetU32Property(m_Axhand,FT_AxHomeModeMap, AxHomeMode);//Get
Home mode
If(AxHomeMode&MODE1_Abs)
{//Support MP_MODE1_Abs homing mode}
If(AxHomeMode& MODE2_Lmt)
{//Support MP_MODE2_Lmt homing mode}

```

B.6.11 Backlash

B.6.11.1 FT_AxBackLashMap

Property Type	R/W	Property ID	Default Value
U32	R	321	-
Description	Get the backlash feature supported by this motion axis.		

Comments	Bits	Description
	0	Enabled
	1	Value
	2~31	Not defined.

B.6.12 Compare

B.6.12.1 FT_AxCompareMap

Property Type	R/W	Property ID	Default Value
U32	R	324	0
Description	Get axis supported compare features.		
Comments	Bits	Description	
	0	Enabled	
	1	Logic	
	2	CmpSrc	
	3	CmpMethod	
	4	CmpPulseMode	
	5	CmpPulseWidth	
	6~31	Not defined.	

B.6.13 Latch

B.6.13.1 FT_AxLatchMap

Property Type	R/W	Property ID	Default Value
U32	R	325	-
Description	Get axis supported latch features.		
Comments	Bits	Description	
	0	Enabled	
	1	Logic	
	2~31	Not defined.	

B.6.14 Cam DO

B.6.14.1 FT_AxCamDOMap

Property Type	R/W	Property ID	Default Value
U32	R	326	-
Description	Get axis supported CamDO features.		
Comments	Bits	Description	
	0	Enabled	
	1	CamDOLogic	
	2	CamDOCmpSrc	
	3	CamDOLoLimit	
	4	CamDOHiLimit	
	5	CamDOMode	
	6	CamDODir	
	7	CamDOAssign	
8~31	Not defined.		

B.6.15 Ext-Drive

B.6.15.1 FT_AxExtDriveMap

Property Type	R/W	Property ID	Default Value
U32	R	327	0
Description	Get axis supported external drive features.		
Comments	Bits	Description	
	0	ExtMasterSrc	
	1	ExtSelEnable	
	2	ExtPulseNum	
	3	ExtPulseMode	
	4	ExtPresetNum	
	5~31	Not defined.	

B.6.15.2 FT_AxExtMasterSrcMap

Property Type	R/W	Property ID	Default Value
U32	R	328	0
Description	Get axis supported external drive master source.		
Comments	Bits	Description	
	0	axis 0	
	1	axis 1	
	2	axis 2	
	3	axis 3	
	4~31	Not defined.	

B.6.16 Jog

B.6.16.1 FT_AxJogMap

Property Type	R/W	Property ID	Default Value										
U32	R	339	7										
Description	Get axis supported jog features. 1: support, 0: Not support												
Comments	<table border="1"><thead><tr><th>Bits</th><th>Description</th></tr></thead><tbody><tr><td>0</td><td>Jog speed</td></tr><tr><td>1</td><td>JogVltime</td></tr><tr><td>2</td><td>Soft jog</td></tr><tr><td>3</td><td>Jog pin assign</td></tr></tbody></table>			Bits	Description	0	Jog speed	1	JogVltime	2	Soft jog	3	Jog pin assign
Bits	Description												
0	Jog speed												
1	JogVltime												
2	Soft jog												
3	Jog pin assign												

B.6.17 Aux/Gen DIO

B.6.17.1 FT_AxGenDOMap

Property Type	R/W	Property ID	Default Value												
U32	R	329	0												
Description	Get axis supported general output from OUT4 to OUT7.														
Comments	<table border="1"><thead><tr><th>Bits</th><th>Description</th></tr></thead><tbody><tr><td>0</td><td>OUT4/CAM_DO</td></tr><tr><td>1</td><td>OUT5/TRIG_Position</td></tr><tr><td>2</td><td>OUT6/SVON</td></tr><tr><td>3</td><td>OUT7/ERC</td></tr><tr><td>4~31</td><td>Not defined.</td></tr></tbody></table>			Bits	Description	0	OUT4/CAM_DO	1	OUT5/TRIG_Position	2	OUT6/SVON	3	OUT7/ERC	4~31	Not defined.
Bits	Description														
0	OUT4/CAM_DO														
1	OUT5/TRIG_Position														
2	OUT6/SVON														
3	OUT7/ERC														
4~31	Not defined.														

B.6.17.2 FT_AxGenDIMap

Property Type	R/W	Property ID	Default Value												
U32	R	330	0												
Description	Get axis supported general input IN1,IN2,IN4,IN5														
Comments	<table border="1"><thead><tr><th>Bits</th><th>Description</th></tr></thead><tbody><tr><td>0</td><td>IN1/LTC</td></tr><tr><td>1</td><td>IN2/RDY</td></tr><tr><td>2</td><td>IN4/JOG+</td></tr><tr><td>3</td><td>IN5/JOG-</td></tr><tr><td>4~31</td><td>Not defined.</td></tr></tbody></table>			Bits	Description	0	IN1/LTC	1	IN2/RDY	2	IN4/JOG+	3	IN5/JOG-	4~31	Not defined.
Bits	Description														
0	IN1/LTC														
1	IN2/RDY														
2	IN4/JOG+														
3	IN5/JOG-														
4~31	Not defined.														

B.6.18 Simultaneous Starting

B.6.18.1 FT_AxSimStartSourceMap

Property Type	R/W	Property ID	Default Value
U32	R	331	-
Description	Get axis supported simultaneous starting mode. See about CFG_AxSimStartSource.		
Comments	The Mode of simultaneous starting that axis supported.		
	Bits	Description	Macro (defined in AdvMotDrv.h)
	0	Start Simultaneous Starting on signal from STA Pin on device.	SimSrc_STA
	1~7	Not defined.	
	8	Start Simultaneous Starting with axis_0's compare signal.	SimSrc_TRIGP_AX0
	9	Start Simultaneous Starting with axis_1's compare signal	SimSrc_TRIGP_AX1
	10	Start Simultaneous Starting with axis_2's compare signal	SimSrc_TRIGP_AX2
	11	Start Simultaneous Starting with axis_3's compare signal	SimSrc_TRIGP_AX3
	12	Start Simultaneous Starting with axis_4's compare signal	SimSrc_TRIGP_AX4
	13	Start Simultaneous Starting with axis_5's compare signal	SimSrc_TRIGP_AX5
	14	Start Simultaneous Starting with axis_6's compare signal	SimSrc_TRIGP_AX6
	15	Start Simultaneous Starting with axis_7's compare signal	SimSrc_TRIGP_AX7
	16	Start Simultaneous Starting when axis_0 is stopped.	SimSrc_STOP_AX0
	17	Start Simultaneous Starting when axis_1 is stopped.	SimSrc_STOP_AX1
	18	Start Simultaneous Starting when axis_2 is stopped.	SimSrc_STOP_AX2
	19	Start Simultaneous Starting when axis_3 is stopped.	SimSrc_STOP_AX3
	20	Start Simultaneous Starting when axis_4 is stopped.	SimSrc_STOP_AX4
	21	Start Simultaneous Starting when axis_5 is stopped.	SimSrc_STOP_AX5
	22	Start Simultaneous Starting when axis_6 is stopped.	SimSrc_STOP_AX6
	23	Start Simultaneous Starting when axis_7 is stopped.	SimSrc_STOP_AX7
	24~31	Not defined.	

B.6.19 Trigger Stop

B.6.19.1 FT_AxIN1Map

Property Type	R/W	Property ID	Default Value
U32	R	333	-
Description	IN1 trigger stop function property.		
Comments	Bits	Description	
	0	Enable/Disable stop function	
	1	Stop logic: high stop or low stop	
	2	Stop mode: decelerating/sudden stop	

B.6.19.2 FT_AxIN2Map

Property Type	R/W	Property ID	Default Value
U32	R	334	0
Description	IN2 trigger stop function property.		
Comments	Bits	Description	
	0	Enable/Disable stop function	
	1	Stop logic: high stop or low stop	
	2	Stop mode: decelerating/sudden stop	

B.6.19.3 FT_AxIN4Map

Property Type	R/W	Property ID	Default Value
U32	R	336	0
Description	IN4 trigger stop function property.		
Comments	Bits	Description	
	0	Enable/Disable stop function	
	1	Stop logic: high stop or low stop	
	2	Stop mode: decelerating/sudden stop	

B.6.19.4 FT_AxIN5Map

Property Type	R/W	Property ID	Default Value
U32	R	337	0
Description	IN5 trigger stop function property.		
Comments	Bits	Description	
	0	Enable/Disable stop function	
	1	Stop logic: high stop or low stop	
	2	Stop mode: decelerating/sudden stop	

B.7 Axis Configurations

B.7.1 System

B.7.1.1 CFG_AxPPU

Property Type	R/W	Property ID	Default Value
U32	R/W	551	1
Description	Pulse per unit (PPU)		
Comments	Pulse per unit (PPU), a virtual unit. This property value must be greater than 0. This property value's change will affect CFG_AxMaxVel, CFG_AxMaxAcc, CFG_AxMaxDec, PAR_AxVelHigh, PAR_AxVelLow, PAR_AxAcc, PAR_AxDec, PAR_GpVelHigh, PAR_GpVelLow, PAR_GpAcc, PAR_GpDec and PAR_HomeCrossDistance.		

B.7.1.2 CFG_AxPhyID

Property Type	R/W	Property ID	Default Value				
U32	R	552	-				
Description	Get physical ID of the axis.						
Comments	<table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0~31</td> <td>0-axis ~ 31-axis</td> </tr> </tbody> </table>			Value	Description	0~31	0-axis ~ 31-axis
Value	Description						
0~31	0-axis ~ 31-axis						

B.7.2 Speed Pattern

B.7.2.1 CFG_AxMaxVel

Property Type	R/W	Property ID	Default Value
F64	R/W	553	20,000,000
Description	Configure the max velocity for the motion axis (Unit: PPU/s).		
Comments	This property's max value = FT_AxMaxVel / CFG_AxPPU and min value = 1 / CFG_AxPPU.		

B.7.2.2 CFG_AxMaxAcc

Property Type	R/W	Property ID	Default Value
F64	R/W	554	2,000,000,000
Description	Configure the max acceleration for the motion axis (Unit: PPU/s ²).		
Comments	This property's max value = FT_AxMaxAcc / CFG_AxPPU and min value = 1 / CFG_AxPPU.		

B.7.2.3 CFG_AxMaxDec

Property Type	R/W	Property ID	Default Value
F64	R/W	555	2,000,000,000
Description	Configure the max deceleration for the motion axis (Unit: PPU/s ²).		
Comments	This property's max value = FT_AxMaxDec / CFG_AxPPU and min value = 1/ CFG_AxPPU.		

B.7.2.4 CFG_AxMaxJerk

Property Type	R/W	Property ID	Default Value
F64	R/W	556	1
Description	Get max jerk configuration for the motion axis.		
Comments			

B.7.3 Alarm

B.7.3.1 CFG_AxAlmEnable

Property Type	R/W	Property ID	Default Value						
U32	R/W	561	0						
Description	Enable/disable motion alarm function. Alarm is a signal generated by motor drive when motor drive is in alarm status.								
Comments	<table border="1"><thead><tr><th>Value</th><th>Description</th></tr></thead><tbody><tr><td>0</td><td>Disabled</td></tr><tr><td>1</td><td>Enabled</td></tr></tbody></table>			Value	Description	0	Disabled	1	Enabled
Value	Description								
0	Disabled								
1	Enabled								

B.7.3.2 CFG_AxAlmLogic

Property Type	R/W	Property ID	Default Value						
U32	R/W	562	1						
Description	Set/get active logic of alarm signal.								
Comments	<table border="1"><thead><tr><th>Value</th><th>Description</th></tr></thead><tbody><tr><td>0</td><td>Low active</td></tr><tr><td>1</td><td>High active</td></tr></tbody></table>			Value	Description	0	Low active	1	High active
Value	Description								
0	Low active								
1	High active								

B.7.3.3 CFG_AxAlmReact

Property Type	R/W	Property ID	Default Value						
U32	R/W	563	1						
Description	Set/get the stop modes when receiving ALARM signal.								
Comments	<table border="1"><thead><tr><th>Value</th><th>Description</th></tr></thead><tbody><tr><td>0</td><td>Motor immediately stops</td></tr><tr><td>1</td><td>Motor decelerates then stops</td></tr></tbody></table>			Value	Description	0	Motor immediately stops	1	Motor decelerates then stops
Value	Description								
0	Motor immediately stops								
1	Motor decelerates then stops								

B.7.3.4 CFG_AxCmdErrorProtection

Property Type	R/W	Property ID	Default Value
U32	R/W	725	1,000
Description	Set/Get the command error protection value.		
Comments	When switching the operation mode such as switching from homing mode to P2P mode, the error occurred if the difference between command position and actual position is large than the command error protection value.		

B.7.4 In Position**B.7.4.1 CFG_AxInpEnable**

Property Type	R/W	Property ID	Default Value						
U32	R/W	564	0						
Description	Enable/disable In-Position function.								
Comments	<table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Disabled</td> </tr> <tr> <td>1</td> <td>Enabled</td> </tr> </tbody> </table>			Value	Description	0	Disabled	1	Enabled
Value	Description								
0	Disabled								
1	Enabled								

B.7.4.2 CFG_AxInpLogic

Property Type	R/W	Property ID	Default Value						
U32	R/W	565	1						
Description	Set/get the active logic for In-Position signal.								
Comments	<table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Low active</td> </tr> <tr> <td>1</td> <td>High active</td> </tr> </tbody> </table>			Value	Description	0	Low active	1	High active
Value	Description								
0	Low active								
1	High active								

B.7.5 Hardware Limit**B.7.5.1 CFG_AxEIEnable**

Property Type	R/W	Property ID	Default Value						
U32	R/W	574	1						
Description	Set/get the reacting mode of EL signal.								
Comments	Please modify CFG_AxEIReact and CFG_AxEILogic before modifying the value of CFG_AxEIEnable.								
Comments	<table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Disabled</td> </tr> <tr> <td>1</td> <td>Enabled</td> </tr> </tbody> </table>			Value	Description	0	Disabled	1	Enabled
Value	Description								
0	Disabled								
1	Enabled								

B.7.5.2 CFG_AxEILogic

Property Type	R/W	Property ID	Default Value
U32	R/W	575	0
Description	Set/get active logic for hardware limit signal.		
Comments	Value	Description	
	0	Disabled	
	1	Enabled	

B.7.5.3 CFG_AxEIReact

Property Type	R/W	Property ID	Default Value
U32	R/W	576	0
Description	Set/get the reacting mode of EL signal.		
Comments	Value	Description	
	0	Motor immediately stops	
	1	Motor decelerates then stops	

B.7.6 Software Limit

B.7.6.1 CFG_AxSwMeIEnable

Property Type	R/W	Property ID	Default Value
U32	R/W	577	0
Description	Enable/Disable the minus software limit function.		
Comments	Value	Description	
	0	Disabled	
	1	Enabled	

B.7.6.2 CFG_AxSwPeIEnable

Property Type	R/W	Property ID	Default Value
U32	R/W	578	0
Description	Enable/Disable the plus software limit.		
Comments	Value	Description	
	0	Disabled	
	1	Enabled	

B.7.6.3 CFG_AxSwMeIReact

Property Type	R/W	Property ID	Default Value
U32	R/W	579	1
Description	Set/get the reacting mode of minus software limit.		
Comments	Value		Description
	0	Motor immediately stops	
	1	Motor decelerates then stops	

B.7.6.4 CFG_AxSwPeIReact

Property Type	R/W	Property ID	Default Value
U32	R/W	580	1
Description	Set/get the reacting mode of plus software limit.		
Comments	Value		Description
	0	Motor immediately stops	
	1	Motor decelerates then stops	

B.7.6.5 CFG_AxSwMeIValue

Property Type	R/W	Property ID	Default Value
I32	R/W	581	-
Description	Set/get the value of minus software limit.		
Comments	The property value's range is: -2,147,483,647 ~ +2,147,483,647.		

B.7.6.6 CFG_AxSwPeIValue

Property Type	R/W	Property ID	Default Value
I32	R/W	582	-
Description	Set/get the value of plus software limit.		
Comments	The property value's range is: -2,147,483,647 ~ +2,147,483,647.		

B.7.7 Home**B.7.7.1 CFG_AxOrgLogic**

Property Type	R/W	Property ID	Default Value
U32	R/W	589	0
Description	Set/get the active logic for ORG signal.		
Comments	Value		Description
	0	Low active	
	1	High active	

B.7.7.2 CFG_AxEzLogic

Property Type	R/W	Property ID	Default Value
U32	R/W	591	0
Description	Set/get the active logic for EZ signal.		
Comments	Value		Description
	0	Low active	
	1	High active	

B.7.7.3 CFG_AxHomeResetEnable

Property Type	R/W	Property ID	Default Value
U32	R/W	602	1
Description	Enable or Disable logical counter reset function after finish Home.		
Comments	Value		Description
	0	Disabled	
	1	Enabled	

B.7.7.4 CFG_AxOrgReact

Property Type	R/W	Property ID	Default Value
U32	R/W	634	1
Description	Set the ending reaction mode after finishing Home.		
Comments	Value		Description
	0	Stop immediately.	
	1	Decelerate and stop.	

B.7.8 Backlash

B.7.8.1 CFG_AxBacklashEnable

Property Type	R/W	Property ID	Default Value
U32	R/W	593	0
Description	Enable/Disable corrective backlash.		
Comments	Value		Description
	0	Disabled	
	1	Enabled	

B.7.8.2 CFG_AxBacklashPulses

Property Type	R/W	Property ID	Default Value
U32	R/W	594	10
Description	Set/get the compensation pulse numbers. (Uint: pulse)		
Comments	This value should be between 0 and 4,095. Whenever direction change occurs, the axis outputs backlash corrective pulses before sending commands.		

B.7.8.3 CFG_AxBacklashVel

Property Type	R/W	Property ID	Default Value
U32	R/W	630	1,000
Description	Set /get the velocity of corrective backlash. (Uint: pulse/s)		
Comments			

B.7.9 Latch**B.7.9.1 CFG_AxLatchLogic**

Property Type	R/W	Property ID	Default Value					
U32	R/W	601	1					
Description	Set/get the active logic for Latch signal.							
Comments	When the Latch is triggered, the command position, actual position and lag distance will be latched.							
		<table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Low active</td> </tr> <tr> <td>1</td> <td>High active</td> </tr> </tbody> </table>	Value	Description	0	Low active	1	High active
Value	Description							
0	Low active							
1	High active							

B.7.9.2 CFG_AxLatchEnable

Property Type	R/W	Property ID	Default Value					
U32	R/W	601	0					
Description	Enable/disable latch function.							
Comments		<table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Disabled</td> </tr> <tr> <td>1</td> <td>Enabled</td> </tr> </tbody> </table>	Value	Description	0	Disabled	1	Enabled
Value	Description							
0	Disabled							
1	Enabled							

B.7.10 Ext-Drive

B.7.10.1 CFG_AxJogPAssign

Property Type	R/W	Property ID	Default Value
U32	R/W	721	-
Description	Set/get the DI channel as external Jog+. In PCI-1203, user can select the DI channel which connected to Motion ring as the JOG+ pin and operate it like other Advantech motion card.		
Comments	It can only select DI channel connected to Motion ring.		

B.7.10.2 CFG_AxJogNAssign

Property Type	R/W	Property ID	Default Value
U32	R/W	722	-
Description	Set/get the DI channel as external Jog-. In PCI-1203, user can select the DI channel which connected to Motion ring as the JOG- pin and operate it like other Advantech motion card.		
Comments	It can only select DI channel connected to Motion ring.		

B.7.10.3 CFG_AxJogVTime

Property Type	R/W	Property ID	Default Value
U32	R/W	692	5,000
Description	Set/get low-speed operating time in Jog motion, it accelerated after the end of time. (Unit: ms)		
Comments	The property value's range is: 0 ~ +4,294,967,295.		

B.7.10.4 CFG_AxJogVelLow

Property Type	R/W	Property ID	Default Value
F64	R/W	695	2,000
Description	Set/get low velocity (start velocity) of this axis in Jog motion (Unit: PPU/S).		
Comments	The property value's range is: 1~ MaxVel		

B.7.10.5 CFG_AxJogVelHigh

Property Type	R/W	Property ID	Default Value
F64	R/W	696	8,000
Description	Set/get high velocity (driving velocity) of this axis in Jog motion (Unit: PPU/S).		
Comments	The property value's range is: 1 ~ MaxVel		

B.7.10.6 CFG_AxJogAcc

Property Type	R/W	Property ID	Default Value
F64	R/W	697	10,000
Description	Set/get acceleration of this axis in Jog motion (Unit: PPU/s ²).		
Comments	The property value's range is: 1 ~ MaxAcc		

B.7.10.7 CFG_AxJogDec

Property Type	R/W	Property ID	Default Value
F64	R/W	698	10,000
Description	Set/get deceleration of this axis in Jog motion (Unit: PPU/s ²).		
Comments	The property value's range is: 1 ~ MaxDec		

B.7.10.8 CFG_AxJogJerk

Property Type	R/W	Property ID	Default Value						
F64	R	699	0						
Description	Set/get the type of velocity profile in Jog motion: T-Curve or S-Curve. It only supports T-Curve profile.								
Comments	<table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>T-curve</td> </tr> <tr> <td>1</td> <td>S-curve (Not support)</td> </tr> </tbody> </table>			Value	Description	0	T-curve	1	S-curve (Not support)
Value	Description								
0	T-curve								
1	S-curve (Not support)								

B.7.11 Cam Do**B.7.11.1 CFG_AxCamDOAssign**

Property Type	R/W	Property ID	Default Value
U32	R/W	723	-
Description	Set/get the DO channel to output CamDO signal. In PCI-1203, user can select the DO channel which connected to Motion ring as the CAM_DO pin and operate it like other Advantech motion card.		
Comments	It can only select DO channel connected to Motion ring.		

B.7.11.2 CFG_AxCamDOEnable

Property Type	R/W	Property ID	Default Value						
U32	R/W	622	-						
Description	Set/get cam DO enable/disable.								
Comments	<table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Disabled</td> </tr> <tr> <td>1</td> <td>Enabled</td> </tr> </tbody> </table>			Value	Description	0	Disabled	1	Enabled
Value	Description								
0	Disabled								
1	Enabled								

B.7.11.3 CFG_AxCamDOLoLimit

Property Type	R/W	Property ID	Default Value
U32	R/W	623	10,000
Description	Set/get the low limit for CAMDO signal.		
Comments	If CamDo is enabled, when command/actual position is between the low limit value and high limit value, the CamDo signal will triggered. Range: -2,147,483,647~+2,147,483,647.		

B.7.11.4 CFG_AxCamDOHiLimit

Property Type	R/W	Property ID	Default Value
U32	R/W	624	10,000
Description	Set/get the high limit for CamDo signal.		
Comments	If CamDo is enabled, when command/actual position is between the low limit value and high limit value, the CamDo signal will triggered. Range: -2,147,483,647~+2,147,483,647.		

B.7.11.5 CFG_AxCamDOCmpSrc

Property Type	R/W	Property ID	Default Value						
U32	R/W	627	0						
Description	Set/get the compare source.								
Comments	<table border="1"><thead><tr><th>Value</th><th>Description</th></tr></thead><tbody><tr><td>0</td><td>Command position</td></tr><tr><td>1</td><td>Feedback position (Not support)</td></tr></tbody></table>			Value	Description	0	Command position	1	Feedback position (Not support)
Value	Description								
0	Command position								
1	Feedback position (Not support)								

B.7.11.6 CFG_AxCamDOLogic

Property Type	R/W	Property ID	Default Value						
U32	R/W	628	1						
Description	Set/get the active logic of CamDO.								
Comments	<table border="1"><thead><tr><th>Value</th><th>Description</th></tr></thead><tbody><tr><td>0</td><td>Low active</td></tr><tr><td>1</td><td>High active</td></tr></tbody></table>			Value	Description	0	Low active	1	High active
Value	Description								
0	Low active								
1	High active								

B.7.12 Module

B.7.12.1 CFG_AxModuleRange

Property Type	R/W	Property ID	Default Value
U32	R/W	629	0
Description	Set/get pulse number when axis moves 360 degree.		
Comments	This value must be multiple of 4. It is used in tangent motion and E-cam motion. See about Acm_AxTangentInGp, Acm_DevDownLoadCAMTable and Acm_AxCamInAx. Range: 0 ~ 8,000,000		

B.7.13 Simultaneous Starting

B.7.13.1 CFG_AxSimStartSource

Property Type	R/W	Property ID	Default Value																																						
U32	R/W	633	1																																						
Description	Set/get simultaneous starting mode for current axis.																																								
Comments	<table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>Disabled</td> </tr> <tr> <td>1</td> <td>Start Simultaneous Starting on signal from STA Pin on device.</td> </tr> <tr> <td>256</td> <td>Start Simultaneous Starting with axis_0's compare signal.</td> </tr> <tr> <td>512</td> <td>Start Simultaneous Starting with axis_1's compare signal</td> </tr> <tr> <td>1024</td> <td>Start Simultaneous Starting with axis_2's compare signal</td> </tr> <tr> <td>2048</td> <td>Start Simultaneous Starting with axis_3's compare signal</td> </tr> <tr> <td>4096</td> <td>Start Simultaneous Starting with axis_4's compare signal</td> </tr> <tr> <td>8192</td> <td>Start Simultaneous Starting with axis_5's compare signal</td> </tr> <tr> <td>16384</td> <td>Start Simultaneous Starting with axis_6's compare signal</td> </tr> <tr> <td>32768</td> <td>Start Simultaneous Starting with axis_7's compare signal</td> </tr> <tr> <td>65536</td> <td>Start Simultaneous Starting when axis_0 is stopped.</td> </tr> <tr> <td>131072</td> <td>Start Simultaneous Starting when axis_1 is stopped.</td> </tr> <tr> <td>262144</td> <td>Start Simultaneous Starting when axis_2 is stopped.</td> </tr> <tr> <td>524288</td> <td>Start Simultaneous Starting when axis_3 is stopped.</td> </tr> <tr> <td>1048576</td> <td>Start Simultaneous Starting when axis_4 is stopped.</td> </tr> <tr> <td>2097152</td> <td>Start Simultaneous Starting when axis_5 is stopped.</td> </tr> <tr> <td>4194304</td> <td>Start Simultaneous Starting when axis_6 is stopped.</td> </tr> <tr> <td>8388608</td> <td>Start Simultaneous Starting when axis_7 is stopped.</td> </tr> </tbody> </table>			Value	Description	0	Disabled	1	Start Simultaneous Starting on signal from STA Pin on device.	256	Start Simultaneous Starting with axis_0's compare signal.	512	Start Simultaneous Starting with axis_1's compare signal	1024	Start Simultaneous Starting with axis_2's compare signal	2048	Start Simultaneous Starting with axis_3's compare signal	4096	Start Simultaneous Starting with axis_4's compare signal	8192	Start Simultaneous Starting with axis_5's compare signal	16384	Start Simultaneous Starting with axis_6's compare signal	32768	Start Simultaneous Starting with axis_7's compare signal	65536	Start Simultaneous Starting when axis_0 is stopped.	131072	Start Simultaneous Starting when axis_1 is stopped.	262144	Start Simultaneous Starting when axis_2 is stopped.	524288	Start Simultaneous Starting when axis_3 is stopped.	1048576	Start Simultaneous Starting when axis_4 is stopped.	2097152	Start Simultaneous Starting when axis_5 is stopped.	4194304	Start Simultaneous Starting when axis_6 is stopped.	8388608	Start Simultaneous Starting when axis_7 is stopped.
Value	Description																																								
0	Disabled																																								
1	Start Simultaneous Starting on signal from STA Pin on device.																																								
256	Start Simultaneous Starting with axis_0's compare signal.																																								
512	Start Simultaneous Starting with axis_1's compare signal																																								
1024	Start Simultaneous Starting with axis_2's compare signal																																								
2048	Start Simultaneous Starting with axis_3's compare signal																																								
4096	Start Simultaneous Starting with axis_4's compare signal																																								
8192	Start Simultaneous Starting with axis_5's compare signal																																								
16384	Start Simultaneous Starting with axis_6's compare signal																																								
32768	Start Simultaneous Starting with axis_7's compare signal																																								
65536	Start Simultaneous Starting when axis_0 is stopped.																																								
131072	Start Simultaneous Starting when axis_1 is stopped.																																								
262144	Start Simultaneous Starting when axis_2 is stopped.																																								
524288	Start Simultaneous Starting when axis_3 is stopped.																																								
1048576	Start Simultaneous Starting when axis_4 is stopped.																																								
2097152	Start Simultaneous Starting when axis_5 is stopped.																																								
4194304	Start Simultaneous Starting when axis_6 is stopped.																																								
8388608	Start Simultaneous Starting when axis_7 is stopped.																																								

B.7.14 Gantry

B.7.14.1 CFG_AxGantryMaxDiffValue

Property Type	R/W	Property ID	Default Value
U32	R/W	658	-
Description	Set/get the value of following error protection		
Comments	The slave axis will move synchronously with mater axis. Set the following error protection value to slave axis, if the distance between master and slave axis is larger than this value, the synchronously motion will stop.		

B.7.15 Trigger Stop

B.7.15.1 CFG_AxIN1StopAssign

Property Type	R/W	Property ID	Default Value
U32	R/W	724	-
Description	Set/get the DI channel as IN1. In PCI-1203, user can select the DI channel which connected to Motion ring as the general input IN1 and operate it like other Advantech motion card.		
Comments	It can only select DI channel connected to Motion ring.		

B.7.15.2 CFG_AxIN1StopEnable

Property Type	R/W	Property ID	Default Value						
U32	R/W	635	0						
Description	Enable/disable INI trigger stop function.								
Comments	<table border="1"><thead><tr><th>Value</th><th>Description</th></tr></thead><tbody><tr><td>0</td><td>Enabled</td></tr><tr><td>1</td><td>Disabled</td></tr></tbody></table>			Value	Description	0	Enabled	1	Disabled
Value	Description								
0	Enabled								
1	Disabled								

B.7.15.3 CFG_AxIN1StopReact

Property Type	R/W	Property ID	Default Value						
U32	R/W	636	1						
Description	Set/get INI trigger stop mode.								
Comments	<table border="1"><thead><tr><th>Value</th><th>Description</th></tr></thead><tbody><tr><td>0</td><td>Sudden stop</td></tr><tr><td>1</td><td>Decelerating</td></tr></tbody></table>			Value	Description	0	Sudden stop	1	Decelerating
Value	Description								
0	Sudden stop								
1	Decelerating								

B.7.15.4 CFG_AxIN1StopLogic

Property Type	R/W	Property ID	Default Value
U32	R/W	637	0
Description	Set/get the active logic of IN1 trigger stop function.		
Comments	Value		Description
	0		Active low
	1		Active high

B.8 Axis Parameters**B.8.1 Speed Pattern****B.8.1.1 PAR_AxVelLow**

Property Type	R/W	Property ID	Default Value
F64	R/W	401	2,000
Description	Set/get low velocity (start velocity) of this axis (Unit: PPU/S).		
Comments	This property value must be smaller than or equal to PAR_AxVelHigh.		

B.8.1.2 PAR_AxVelHigh

Property Type	R/W	Property ID	Default Value
F64	R/W	402	8,000
Description	Set/get high velocity (driving velocity) of this axis (Unit: PPU/s).		
Comments	This property value must be smaller than CFG_AxMaxVel and greater than PAR_AxVelLow.		

B.8.1.3 PAR_AxAcc

Property Type	R/W	Property ID	Default Value
F64	R/W	403	10,000
Description	Set/get acceleration of this axis (Unit: PPU/s^2).		
Comments	This property value must be smaller than or equal to CFG_AxMaxAcc.		

B.8.1.4 PAR_AxDec

Property Type	R/W	Property ID	Default Value
F64	R/W	404	10,000
Description	Set/get deceleration of this axis (Unit: PPU/s^2).		
Comments	This property value must be smaller than or equal to CFG_AxMaxDec.		

B.8.1.5 PAR_AxJerk

Property Type	R/W	Property ID	Default Value						
F64	R/W	405	0						
Description	Set/get the type of velocity profile: T-Curve or S-Curve.								
Comments	<table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>T-curve</td> </tr> <tr> <td>1</td> <td>S-curve</td> </tr> </tbody> </table>			Value	Description	0	T-curve	1	S-curve
	Value	Description							
0	T-curve								
1	S-curve								
	<p>The actual jerk is calculated by driver. If PAR_AxJerk is set to be 1, the PAR_AxAcc does not means acceleration but max acceleration and PAR_AxDec does not means deceleration but max deceleration.</p>								

B.8.1.6 PAR_AxJerkFactor

Property Type	R/W	Property ID	Default Value
F64	R/W	443	50.0
Description	Set/get the first section time percentage of S-Curve.		
Comments	<p>The velocity / acceleration / jerk profile is shown below. The velocity profile can be divided by 7 sections and this property specific the ratio of first section time (S1 or S5) to acceleration section time (S1+S2+S3) or deceleration section time (S5+S6+S7). This time of S1 and S3 (S5 and S7) are the same and the ratio value is between 0 to 50. If PAR_AxJerkFactor is set to be 50, the velocity profile will be a pure S-Curve with its acceleration profile becomes a triangle.</p>		

B.8.2 Home

B.8.2.1 PAR_AxHomeVelLow

Property Type	R/W	Property ID	Default Value
F64	R/W	415	2,000
Description	Set/get low velocity (start velocity) of this axis in home motion (Unit: PPU/S).		
Comments	This property value must be smaller than or equal to CFG_AxMaxVel.		

B.8.2.2 PAR_AxHomeVelHigh

Property Type	R/W	Property ID	Default Value
F64	R/W	416	8,000
Description	Set/get high velocity (driving velocity) of this axis in home motion (Unit: PPU/s).		
Comments	This property value must be smaller than or equal to CFG_AxMaxVel.		

B.8.2.3 PAR_AxHomeAcc

Property Type	R/W	Property ID	Default Value
F64	R/W	417	10,000
Description	Set/get acceleration of this axis in home motion (Unit: PPU/s ²).		
Comments	This property value must be smaller than or equal to CFG_AxMaxAcc.		

B.8.2.4 PAR_AxHomeDec

Property Type	R/W	Property ID	Default Value
F64	R/W	418	10,000
Description	Set/get deceleration of this axis in home motion (Unit: PPU/s ²).		
Comments	This property value must be smaller than or equal to CFG_AxMaxDec.		

B.8.2.5 PAR_AxHomeJerk

Property Type	R/W	Property ID	Default Value						
F64	R/W	419	0						
Description	Set/get the type of velocity profile in home motion.								
Comments	<table border="1"> <thead> <tr> <th>Value</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>T-curve</td> </tr> <tr> <td>1</td> <td>S-curve</td> </tr> </tbody> </table>			Value	Description	0	T-curve	1	S-curve
Value	Description								
0	T-curve								
1	S-curve								

B.8.2.6 PAR_AxHomeJerkFactor

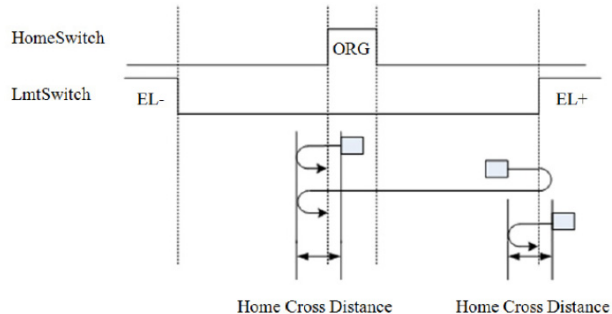
Property Type	R/W	Property ID	Default Value
F64	R/W	444	50.0
Description	Set/get the first section time percentage of S-Curve in home motion.		
Comments	This property only used in Acm_AxMoveHome.		

B.8.2.7 PAR_AxHomeCrossDistance

Property Type	R/W	Property ID	Default Value
F64	R/W	408	10,000
Description	Set the home cross distance (Unit: PPU).		

This property must be greater than 0.

Comments



B.8.2.8 PAR_AxHomeExSwitchMode

Property Type	R/W	Property ID	Default Value
F64	R/W	407	0
Description	Setting the stopping condition of Acn_AxHomeEx.		

Comments

Value	Define	Description
0	Level On	When sensor is ON(Active)
1	Level Off	When sensor is OFF(Non-active)
2	Rising Edge	When OFF to ON transition in sensor
3	Falling Edge	When ON to OFF transition in sensor

B.9 Group Configurations

B.9.1 System

B.9.1.1 CFG_GpAxesInGroup

Property Type	R/W	Property ID	Default Value
U32	R	806	-
Description	Get information about which axis is (are) in this group.		

Comments

Bits	Description
0~31	Axis 0 ~ Axis 31

B.9.2 Path

B.9.2.1 CFG_GpBlidTime

Property Type	R/W	Property ID	Default Value
U32	R/W	808	10,000
Description	Set/get blending time when add a path into system path buffer.		
Comments	It should be 0~65,535 and multiple of 2. (Unit: ms)		

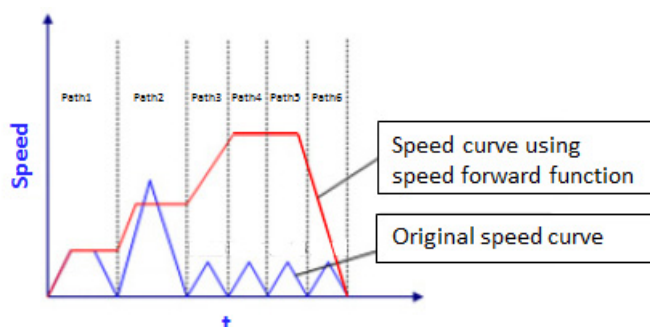
B.9.2.2 CFG_GpSFEnable

Property Type	R/W	Property ID	Default Value
U32	R/W	809	
Description	Enable/Disable speed forward function.		

Value	Description
0	Disabled
1	Enabled

It cannot support S profile speed curve. In this mode, the speed parameter of Acm_AddPath is useless, just speed setting of group is used.

Comments



B.10 Group Parameters

B.10.1 Speed Pattern

B.10.1.1 PAR_GpVelLow

Property Type	R/W	Property ID	Default Value
F64	R/W	701	The first added axis' PAR_AxVelLow
Description	Set low velocity (start velocity) of this group (Unit: PPU/s).		
Comments	This property value must be smaller than or equal to PAR_GpVelHigh.		

B.10.1.2 PAR_GpVelHigh

Property Type	R/W	Property ID	Default Value
F64	R/W	702	The first added axis' PAR_AxVelHigh
Description	Set high velocity (driving velocity) of this group (Unit: PPU/s).		
Comments	This property value must be smaller than first added axis' CFG_AxMaxVel and greater than PAR_GpVelLow.		

B.10.1.3 PAR_GpAcc

Property Type	R/W	Property ID	Default Value
F64	R/W	703	The first added axis' PAR_AxAcc
Description	Set acceleration of this group (Unit: PPU/s ²).		
Comments	This property value must be smaller than or equal to first added axis' CFG_AxMaxAcc.		

B.10.1.4 PAR_GpDec

Property Type	R/W	Property ID	Default Value
F64	R/W	704	The first added axis' PAR_AxDec
Description	Set deceleration of this group (Unit: PPU/s^2).		
Comments	This property value must be smaller than or equal to first added axis' CFG_AxMaxDec.		

B.10.1.5 PAR_GpJerk

Property Type	R/W	Property ID	Default Value
F64	R/W	705	The first added axis jerk
Description	Set the type of velocity profile: T-Curve or S-Curve.		
Comments	If PAR_GpJerk is set to 1, the PAR_GpAcc doesn't mean acceleration but max acceleration and PAR_GpDec doesn't means deceleration but max deceleration.		

B.10.1.6 PAR_GpJerkFactor

Property Type	R/W	Property ID	Default Value
F64	R/W	710	50.0
Description	Set/get the first section time percentage of S-Curve.		
Comments			

B.10.2 System

B.10.2.1 PAR_GpGroupID

Property Type	R/W	Property ID	Default Value
U32	R/W	706	-
Description	Get the GroupID through GroupHandle.		
Comments	In PCI-1203, there are only six GroupID to use. They are 0,1,...4 and 5. You cannot handle more than six groups at the same time, you must close one group to create new group if there are already six groups.		

B.10.3 Path

B.10.3.1 PAR_GpRefPlane

Property Type	R/W	Property ID	Default Value								
U32	R/W	709	0								
Description	Set/get reference plane for helix motion and arc interpolation.										
Comments	<table border="1"><thead><tr><th>Value</th><th>Description</th></tr></thead><tbody><tr><td>0</td><td>XY PLANE</td></tr><tr><td>1</td><td>YZ PLANE</td></tr><tr><td>2</td><td>XZ PLANE</td></tr></tbody></table>			Value	Description	0	XY PLANE	1	YZ PLANE	2	XZ PLANE
Value	Description										
0	XY PLANE										
1	YZ PLANE										
2	XZ PLANE										

Appendix **C**

Error Codes

C.1 Error Codes

Each function will return a return code while calling function in API list and this return code indicates the call result. User can know the message of this error code by calling `Acm_GetErrorMessage` API for further modifications.

Return	0x00000000
Message	SUCCESS

The error message and description of the possible reason of error occurs are shown below.

C.2 Common Errors

Error Code	0x80000000
Message	InvalidDevNumber
Description	Invalid device number
Function	<code>Acm_AxCamInAx</code> , <code>Acm_AxTangentInGp</code> , <code>Acm_AxGantryInAx</code> , <code>Acm_DevLoadConfig...</code>
Example	While using <code>Acm_AxCamInAx</code> API to start cam synchronization and an error occurs when the slave axis does not belong to this device.
Error Code	0x80000004
Message	MemAllocateFailed
Description	Windows does not contain sufficient free space to allocate
Function	<code>Acm_DevOpen</code> , <code>Acm_GpLoadPath</code> , <code>Acm_DevLoadConfig</code> <code>Acm_DevLoadCAMTableFile</code> , <code>Acm_GpAddAxis</code> , <code>Acm_AxOpen</code>
Example	
Error Code	0x80000005
Message	InvalidHandle
Description	Invalid handle
Function	All function with device/axis/group handle input
Example	While using <code>Acm_AxOpen</code> API to open axis and an error occurs when device handle does not exist or is not correct.
Error Code	0x80000006
Message	CreateFileFailed
Description	Failed to create file
Function	<code>Acm_DevOpen</code>
Example	
Error Code	0x80000009
Message	InvalidInputParam
Description	Invalid input parameter
Function	<code>Acm_EnableMotionEvent</code> , <code>Acm_DevDownloadCAMTable</code> , <code>Acm_DaqDiGetByte...</code>
Example	While using <code>Acm_DaqDoSetByte</code> API to set DO value and an error occurs when the input parameter <code>DOPort</code> is more than the maximum DO Port (<code>FT_DaqDoMaxChan/8</code>) connected to the device
Error Code	0x8000000a
Message	PropertyIDNotSupport
Description	PropertyID is not supported
Function	All set/get property function

Example	An error occurs when device does not support this property ID
Error Code	0x8000000b
Message	PropertyIDReadOnly
Description	PropertyID is read only
Function	All set property function
Example	While using API to set FT_DevEmgMap property and an error occurs because the property is read only.
Error Code	0x8000000d
Message	InvalidAxCfgVel
Description	Invalid Max-Velocity configuration on axis
Function	Acm_SetF64Property
Example	While using Acm_SetF64Property API to set the maximum driving velocity (CFG_AxMaxVel) on axis to 20,000,001 and an error occurs because the maximum of this value is 20,000,000.
Error Code	0x8000000e
Message	InvalidAxCfgAcc
Description	Invalid Max-Acceleration configuration on axis
Function	Acm_SetF64Property
Example	While using Acm_SetF64Property API to set the maximum acceleration (CFG_AxMaxAcc) on axis to 2,000,000,001 and an error occurs because the maximum of this value is 2,000,000,000.
Error Code	0x8000000f
Message	InvalidAxCfgDec
Description	Invalid Max-Deceleration configuration on axis
Function	Acm_SetF64Property
Example	While using Acm_SetF64Property API to set the maximum deceleration (CFG_AxMaxDec) on axis to 2,000,000,001 and an error occurs because the maximum of this value is 2,000,000,000.
Error Code	0x80000011
Message	InvalidAxParVelLow
Description	Invalid initial velocity parameter on axis
Function	Acm_SetF64Property
Example	While using Acm_SetF64Property API to set the initial velocity on axis (PAR_AxVelLow) and an error occurs when the value is less than 0.
Error Code	0x80000012
Message	InvalidAxParVelHigh
Description	Invalid driving velocity parameter on axis
Function	Acm_SetF64Property
Example	While using Acm_SetF64Property API to set the driving velocity on axis (PAR_AxVelHigh) and an error occurs when the value is less than 0 or the value is greater than maximum driving velocity (CFG_AxMaxVel).
Error Code	0x80000013
Message	InvalidAxParAcc
Description	Invalid acceleration setting on axis
Function	Acm_SetF64Property
Example	While using Acm_SetF64Property API to set the acceleration on axis (PAR_AxAcc) and an error occurs when the value is less than 0 or the value is greater than maximum acceleration (CFG_AxMaxAcc).
Error Code	0x80000014
Message	InvalidAxParDec

Description	Invalid deceleration setting on axis
Function	Acm_SetF64Property
Example	While using Acm_SetF64Property API to set the acceleration on axis (PAR_AxDec) and an error occurs when the value is less than 0 or the value is greater than maximum acceleration (CFG_AxMaxDec).
Error Code	0x80000015
Message	InvalidAxParJerk
Description	Invalid jerk value on axis
Function	Acm_SetF64Property
Example	While using Acm_SetF64Property API to set the type of velocity profile on axis (PAR_AxJerk) and an error occurs when the set value is not 0 or 1.
Error Code	0x80000016
Message	InvalidAxPulseInMode
Description	Invalid Pulse-in mode on axis
Function	Acm_SetU32Property
Example	While using API to set the value of property CFG_AxPulseInMode and an error occurs when the set value is not within the input range.
Error Code	0x80000017
Message	InvalidAxPulseOutMode
Description	Invalid Pulse-out mode on axis
Function	Acm_SeU32Property
Example	While using API to set the value of property CFG_AxPulseOutMode and an error occurs when the set value is not within the input range.
Error Code	0x8000002a
Message	InvalidAxState
Description	Invalid axis states
Function	Acm_AxGantryInAx, Acm_AxTangentInGp, Acm_AxGearInAx, Acm_AxCamInAx, Acm_AxMoveHome, Acm_AxHomeEx, Acm_AxSimStart, Acm_AxSimStartSuspendVel, Acm_AxSimStartSuspendRel, Acm_AxSimStartSuspendAbs, Acm_AxChangeVelExByRate, Acm_AxChangeVelEx, Acm_AxChangeVel, Acm_AxChangeVelByRate
Example	While using Acm_AxHome API to command axis to start typical home motion and an error occurs when the axis state is not Ready.
Error Code	0x8000002d
Message	InvalidAxDistance
Description	Invalid axis distance
Function	Acm_AxMoveRel, Acm_AxMoveAbs...
Example	While using Acm_AxMoveRel API to command axis to start relative point-to-point motion and an error occurs when the Distance value is not within the input range (-2,147,483,647/PPU ~ 2,147,483,647/PPU).
Error Code	0x80000030
Message	InvalidAxCntInGp
Description	Invalid axis count in group
Function	Acm_GpMoveArcAbs_Angle, Acm_GpMoveArcRel_Angle...
Example	Command group to do 3-axis arc interpolation and an error occurs when only two axes in the group.
Error Code	0x80000031
Message	AxInGpNotFound
Description	Cannot find axis in group
Function	Acm_GpRemAxis

Example	While using Acm_GpRemAxis API to remove the axis from group and an error occurs when the axis is not in the group.
Error Code	0x80000032
Message	AxisInOtherGp
Description	Axis is already in other group
Function	Acm_GpAddAxis
Example	While using Acm_GpAddAxis API to add the axis into group and an error occurs when the axis is already in other group. Each axis can only exist within a group
Error Code	0x80000033
Message	AxCannotIntoGp
Description	Axis cannot be set into group
Function	Acm_GpAddAxis
Example	While using Acm_GpAddAxis API to add the 9th axis into group and an error occurs because the axis count exceeds the maximum count (up to 8 axes) in one group.
Error Code	0x80000039
Message	InvalidGpParVelLow
Description	Invalid initial velocity parameter on group
Function	Acm_SetF64Property
Example	While using Acm_SetF64Property API to set the initial velocity on group (PAR_GpVelLow) and an error occurs when the value is less than 0 or the value is greater than maximum driving velocity (CFG_AxMaxVel).
Error Code	0x8000003a
Message	InvalidGpParVelHigh
Description	Invalid driving velocity parameter on group
Function	Acm_GpMoveCircularRel, Acm_GpMoveCircularAbs, Acm_GpChangeVelByRate...
Example	While using Acm_GpChangeVel API to command group to change the velocity while group is in line interpolation motion and an error occurs when the value is less than 0 or the value is greater than maximum driving velocity (CFG_AxMaxVel).
Error Code	0x8000003b
Message	InvalidGpParAcc
Description	Invalid acceleration parameter on group
Function	Acm_SetF64Property
Example	While using Acm_SetF64Property API to set the property PAR_GpAcc and an error occurs when the value is less than 0 or the value is greater than maximum acceleration (CFG_AxMaxAcc).
Error Code	0x8000003c
Message	InvalidGpParDec
Description	Invalid deceleration parameter on group
Function	Acm_SetF64Property
Example	While using Acm_SetF64Property API to set the property PAR_GpDec and an error occurs when the value is less than 0 or the value is greater than maximum deceleration (CFG_AxMaxDec).
Error Code	0x8000003d
Message	InvalidGpParJerk
Description	Invalid jerk parameter on group

Function	Acm_GpMoveCircularRel_Angle, Acm_GpMoveCircularAbs_Angle Acm_GpMoveArcRel_Angle, Acm_GpMoveArcAbs_Angle Acm_GpMoveHelixAbs_Angle, Acm_GpMoveHelixRel_Angle Acm_GpMove3DArcAbs_V, Acm_GpMove3DArcRel_V
Example	While using Acm_GpMoveCircularRel_Angle API to complete circular interpolation and an error occurs when the type of velocity profile (PAR_GpJerk) is S-Curve.
Error Code	0x800003e
Message	JerkNotSupport
Description	S-Curve is not support
Function	Acm_GpAddPath
Example	While using Acm_GpAddPath API to set value of MoveMode to Blending Mode or FlyMode and an error occurs because these two modes do not support S-Curve type of velocity profile.
Error Code	0x8000041
Message	InvalidGpState
Description	Invalid group states
Function	Acm_GpClose, Acm_GpChangeVelByRate, Acm_GpChangeVel
Example	While using Acm_GpClose API to close group and an error occurs when the group state is not equal to STA_GP_DISABLE, STA_GP_READY or STA_GP_ERROR_STOP.
Error Code	0x8000042
Message	OpenFileFailed
Description	Failed to open file
Function	Acm_DevLoadConfig, Acm_GpLoadPath, Acm_DevLoadCAMTableFile
Example	
Error Code	0x8000043
Message	InvalidPathCnt
Description	Invalid path count
Function	Acm_GpLoadPath, Acm_DevLoadCAMTableFile
Example	While using Acm_GpLoadPath API to load path file and an error occurs when the path count is less than 2 or greater than 600. Or use Acm_DevLoadCAMTableFile API to load Cam table file and an error occurs when the count of table is less than 2 or greater than 100,000.
Error Code	0x8000044
Message	InvalidPathHandle
Description	Invalid path handle
Function	Acm_GpLoadPath, Acm_GpUnloadPath
Example	While using Acm_GpLoadPath and Acm_GpUnloadPath API to load and unload path and an error occurs when the path handle does not exist or is incorrect.
Error Code	0x800004A
Message	InvalidSlaveIP
Description	Invalid slave IP
Function	All function with SlaveIP input
Example	While using Acm_DevGetSlaveInfo API to get slave information and an error occurs when the SlaveIP is not in the device.
Error Code	0x8000054
Message	FunctionNotSupport
Description	Function is not supported
Function	All function

Example	An error occurs when the motion card does not support this function.
Error Code	0x80000055
Message	InvalidPhysicalAxis
Description	Invalid physical axis
Function	Acm_DevMDaqConfig
Example	While using Acm_DevMDaqConfig API to set MDAQ properties and an error occurs when the specified axis number is greater than the number of axes in the device.
Error Code	0x80000069
Message	InvalidPath
Description	For PCI-1240, the following condition will occur error: <ol style="list-style-type: none"> The axis count in the group is greater than 2 and the command of path is 2-axis line interpolation or 2-axis arc interpolation. The axis count in the group is less than 3 and the command of path is multi-axis (greater than or equal to 3-axis) line interpolation. For other PCI motion card, an error occurs when the axis count in group is less than demand of axis count to execute the command of path.
Function	Acm_GpLoadPath
Example	While using Acm_GpLoadPath API to load path file and an error occurs when there is 3-axis interpolation command and only 2 axes are in the group.
Error Code	0x80000073
Message	InvalidGpHandle
Description	Invalid group handle
Function	All function with group handle input
Example	While using Acm_GpGetPathStatus API to get path status and an error occurs when the handle is incorrect.
Error Code	0x80000075
Message	InvalidCmpMethod
Description	Invalid compare method
Function	Acm_SetU32Property
Example	While using API to set CFG_AxCmpMethod property and an error occurs when the set value is not 0 (\geq Position Counter) or 1 (\leq Position Counter).
Error Code	0x8000007b
Message	SpeedFordFunNotSpported
Description	Speed forward function is not supported in current mode
Function	Acm_GpAddPath
Example	While using Acm_GpAddPath API to add path into path buffer and an error occurs when the speed forward function is enabled and MoveMode is in BufferMode or BlendingMode.
Error Code	0x80000081
Message	InvalidAxParHomeVelLow
Description	Invalid initial velocity parameter of home on axis
Function	Acm_SetF64 Property
Example	While using Acm_SetF64Property API to set initial velocity parameter of home on axis (Par_AxHomeVelLow) and an error occurs when the value is less than 0 or the value is greater than maximum driving velocity (CFG_AxMaxVel).
Error Code	0x80000082
Message	InvalidAxParHomeVelHigh

Description	Invalid driving velocity parameter of home on axis
Function	Acm_SetF64 Property
Example	While using Acm_SetF64Property API to set driving velocity parameter of home on axis (Par_AxHomeVelHigh) and an error occurs when the value is less than 0 or the value is greater than maximum driving velocity (CFG_AxMaxVel).
Error Code	0x80000083
Message	InvalidAxParHomeAcc
Description	Invalid acceleration parameter of home on axis
Function	Acm_SetF64 Property
Example	While using Acm_SetF64Property API to set acceleration parameter of home on axis (Par_AxHomeAcc) and an error occurs when the value is less than 0 or the value is greater than maximum acceleration (CFG_AxMaxAcc).
Error Code	0x80000084
Message	InvalidAxParHomeDec
Description	Invalid deceleration parameter of home on axis
Function	Acm_SetF64 Property
Example	While using Acm_SetF64Property API to set deceleration parameter of home on axis (Par_AxHomeDec) and an error occurs when the value is less than 0 or the value is greater than maximum deceleration (CFG_AxMaxDec).
Error Code	0x80000085
Message	InvalidAxParHomeJerk
Description	Invalid jerk parameter of home on axis
Function	Acm_SetF64 Property
Example	While using Acm_SetF64Property API to set the type of velocity profile on axis (PAR_AxHomeJerk) and an error occurs when the set value is not 0 or 1.
Error Code	0x80000086
Message	InvalidAxCfgJogVelLow
Description	Invalid initial velocity parameter of jog on axis
Function	Acm_SetF64 Property
Example	While using Acm_SetF64Property API to set initial velocity parameter of jog on axis (Par_AxJogVelLow) and an error occurs when the value is less than 0 or the value is greater than maximum driving velocity (CFG_AxMaxVel).
Error Code	0x80000087
Message	InvalidAxCfgJogVelHigh
Description	Invalid driving velocity parameter of jog on axis
Function	Acm_SetF64 Property
Example	While using Acm_SetF64Property API to set driving velocity parameter of jog on axis (Par_AxJogVelHigh) and an error occurs when the value is less than 0 or the value is greater than maximum driving velocity (CFG_AxMaxVel).
Error Code	0x80000088
Message	InvalidAxCfgJogAcc
Description	Invalid acceleration parameter of jog on axis
Function	Acm_SetF64 Property

Example	While using Acm_SetF64Property API to set acceleration parameter of jog on axis (CFG_AxJogVelAcc) and an error occurs when the value is less than 0 or the value is greater than maximum acceleration (CFG_AxMaxAcc).
Error Code	0x80000089
Message	InvalidAxCfgJogDec
Description	Invalid deceleration parameter of jog on axis
Function	Acm_SetF64 Property
Example	While using Acm_SetF64Property API to set deceleration parameter of jog on axis (CFG_AxJogVelDec) and an error occurs when the value is less than 0 or the value is greater than maximum deceleration (CFG_AxMaxDec).
Error Code	0x8000008A
Message	InvalidAxCfgJogJerk
Description	Invalid jerk parameter of jog on axis
Function	Acm_SetF64 Property
Example	While using Acm_SetF64Property API to set the type of velocity profile on axis (CFG_AxJogJerk) and an error occurs when the set value is not 0.
Error Code	0x8000008B
Message	InvalidAxCfgKillDec
Description	Invalid axis DI stop deceleration
Function	Acm_SetF64 Property
Example	While using Acm_SetF64Property API to set deceleration parameter of DI stop on axis (CFG_AxKillDec) and an error occurs when the value is 0 and below or the value is greater than maximum deceleration (FT_AxMaxDec).
Error Code	0x8000008C
Message	NotOpenAllAxes
Description	Not all the axes open
Function	Acm_DevLoadConfig
Example	While using Acm_DevLoadConfig API to load the configuration file and an error occurs when not all the axes connected to the motion card have been opened.
Error Code	0x800000d0
Message	ReadFileFailed
Description	Failed to read file
Function	Acm_DevLoadConfig, Acm_DevLoadMapFile
Example	While using API to load configuration or mapping table file and an error occurs because the file is incorrect or open file fail due to operating system error.
Error Code	0x800000d3
Message	GetAuthorityFailed
Description	SDO handshaking time out. PCI-1203 scans network in DevOpen and changes the ESM status
Function	Acm_DevOpen, Acm_DevReOpen
Example	
Error Code	0x800000d6
Message	InvalidGpRadius
Description	Invalid radius setting on group
Function	All function about arc interpolation

Example	An error occurs due to incorrect radius setting when commanding are interpolation.
Error Code	0x800000d9
Message	InvalidGpMovCmd
Description	Invalid group command
Function	Acm_GpLoadPath, Acm_GpAddPath
Example	While using Acm_GpLoadPath, Acm_GpAddPath API and an error occurs when this motion card does not support this command or the command is not supported in some circumstance.
Error Code	0x800000dc
Message	InvalidPathMoveMode
Description	Invalid path move mode
Function	Acm_GpAddPath
Example	While using Acm_GpAddPath API to set the MoveMode parameter and an error occurs when the set value is not 0 or 1.
Error Code	0x800000de
Message	InvalidCamTableID
Description	Invalid Cam table ID
Function	Acm_DevDownloadCAMTable
Example	The Cam table ID of Acm_DevDownloadCAMTable API must be set to 0 or 1, otherwise, an error occurs.
Error Code	0x800000df
Message	InvalidCamPointRange
Description	Invalid Cam Point range
Function	Acm_DevDownloadCAMTable
Example	While using Acm_DevDownloadCAMTable API and an error occurs when the value of parameter pPointRangeArray[i] is greater than (pMasterArray[i] - pMasterArray[i-1])*0.5).
Error Code	0x800000e0
Message	CamTableIsEmpty
Description	Cam table is empty
Function	Acm_DevConfigCAMTable
Example	While using Acm_DevConfigCAMTable API to configure cam table and an error occurs because the cam table is empty.
Error Code	0x800000e1
Message	InvalidPlaneVector
Description	Invalid plane vector
Function	Acm_AxTangentInGp
Example	While using Acm_AxTangentInGp API to establish tangent following synchronization with the Group and an error occurs when the parameter of StartVectorArray is not 3-dimension or all the vector value of StartVectorArray are 0.
Error Code	0x800000e2
Message	MasAxIDSameSlvAxID
Description	Master axis ID cannot be the same with slave axis ID
Function	Acm_AxCamInAx, Acm_AxGantryInAx, Acm_AxGearInAx
Example	While using Acm_AxCamInAx API to establish relations of master and slave axis and an error occurs because the setting of slave axis is the same as master axis.
Error Code	0x800000e3
Message	InvalidGpRefPlane

Description	Invalid group reference plane
Function	All function about arc, helix interpolation and add path
Example	While using Acm_GpMoveCircularRel_3P API to command group to do 3-axis arc interpolation and an error occurs when the setting of reference plane is YZ and the axis count in group is less than 3.
Error Code	0x800000e4
Message	InvalidAxModuleRange
Description	Invalid axis module range
Function	Acm_SetU32Property
Example	While using Acm_SetU32Property API to set axis module range (CFG_AxModuleRange) and an error occurs when the set value is greater than 8,000,000 or the value cannot be divisible by 4.
Error Code	0x800000e5
Message	DownloadFileFailed
Description	Failed to down load file
Function	
Example	
Error Code	0x800000e6
Message	InvalidFileLength
Description	Invalid file length
Function	
Example	
Error Code	0x800000e7
Message	InvalidCmpCnt
Description	Invalid compare count
Function	Acm_AxSetCmpTable, Acm_AxSetCmpAuto
Example	While using Acm_AxSetCmpTable API to set compare table and an error occurs when the set value of ArrayCount is greater than 100,000 or less than 0.
Error Code	0x80004001
Message	DevEvtTimeOut
Description	Device event time out
Function	Acm_CheckMotionEvent
Example	While using Acm_CheckMotionEvent API to detect motion event and an error occurs when there is no event happened within the specified time.
Error Code	0x80004002
Message	DevNoEvt
Description	There is no event in device
Function	Acm_CheckMotionEvent
Example	While using Acm_CheckMotionEvent API to detect motion event and an error occurs because the event is not enabled.

C.3 DSP Common Errors

Error Code	0x10000001
Message	Warning_AxWasInGp
Description	Axis is already in group
Function	Acm_GpAddAxis
Example	While using Acm_GpAddAxis API to add the axis into group and an error occurs when the axis is already in this group.
Error Code	0x80005001
Message	ERR_SYS_TIME_OUT
Description	System timed out
Function	
Example	
Error Code	0x80005002
Message	Dsp_PropertyIDNotSupport
Description	PropertyID is not supported
Function	All set/get property function
Example	An error occurs when device does not support this property ID
Error Code	0x80005003
Message	Dsp_PropertyIDReadOnly
Description	PropertyID is read only
Function	All set property function
Example	While using API to set FT_DevEmgMap property and an error occurs because the property is read only.
Error Code	0x80005004
Message	Dsp_InvalidParameter
Description	Invalid parameter
Function	Acm_DaqDoSetByte...
Example	While using Acm_DaqDoSetByte API to set DO value and an error occurs when the input parameter DOPort is more than the maximum DOPort (FT_DaqDoMaxChan/8) connected to the device
Error Code	0x80005005
Message	Dsp_DataOutBufExceeded
Description	Data is out of buffer
Function	
Example	
Error Code	0x80005006
Message	Dsp_FunctionNotSupport
Description	Function is not supported
Function	All function
Example	An error occurs when the motion card does not support this function.
Error Code	0x80005007
Message	Dsp_InvalidConfigFile
Description	Invalid configuration file
Function	
Example	
Error Code	0x80005008
Message	Dsp_InvalidIntervalData

Description	Invalid interval data
Function	
Example	
Error Code	0x80005009
Message	Dsp_InvalidTableSize
Description	Invalid table size
Function	
Example	
Error Code	0x8000500a
Message	Dsp_InvalidTableID
Description	Invalid Cam table ID or invalid compare table ID
Function	Acm_DevDownloadCAMTable
Example	The Cam table ID of Acm_DevDownloadCAMTable API must be set to 0 or 1, otherwise, an error occurs.
Error Code	0x8000500b
Message	Dsp_DataIndexExceedBufSize
Description	Data index exceeded maximum number (100,000) of compare table or maximum number (100,000) of maximum number (32) of one-time read
Function	Acm_AxSetCmpTable, Acm_AxSetCmpAuto, Acm_AxReadLatchBuffer
Example	While using Acm_AxSetCmpTable API to set compare table and an error occurs when the compare table data index exceeded the maximum number (100,000)
Error Code	0x8000500c
Message	Dsp_InvalidCompareInterval
Description	Invalid compare interval
Function	Acm_AxSetCmpAuto
Example	
Error Code	0x8000500d
Message	Dsp_InvalidCompareRange
Description	Difference from the first data and last data of the compare list is less than the compare interval
Function	Acm_AxSetCmpAuto
Example	While using API to set compare range and an error occurs when the difference from the first data and last data of the compare list is less than the compare interval.
Error Code	0x8000500e
Message	Dsp_PropertyIDWriteOnly
Description	PropertyID is write-only
Function	All get property function
Example	While using API to get property information and an error occurs because the property is write only.
Error Code	0x8000500f
Message	Dsp_NcError
Description	NC is error
Function	
Example	
Error Code	0x80005010
Message	Dsp_CamTableIsInWhile using
Description	CamTable is in use
Function	Acm_AxCamInAx

Example	
Error Code	0x80005011
Message	Dsp_EraseBlockFailed
Description	Failed to erase block
Function	
Example	
Error Code	0x80005012
Message	Dsp_ProgramFlashFailed
Description	Failed to flash program
Function	
Example	
Error Code	0x80005101
Message	Dsp_InvalidAxCfgVel
Description	Invalid velocity configuration on axis
Function	Acm_SetF64Property
Example	While using Acm_SetF64Property API to set the maximum driving velocity (CFG_AxMaxVel) on axis to 20,000,001 and an error occurs because the maximum of this value is 20,000,000.
Error Code	0x80005102
Message	Dsp_InvalidAxCfgAcc
Description	Invalid acceleration configuration on axis
Function	Acm_SetF64Property
Example	While using Acm_SetF64Property API to set the maximum acceleration (CFG_AxMaxAcc) on axis to 2,000,000,001 and an error occurs because the maximum of this value is 2,000,000,000.
Error Code	0x80005103
Message	Dsp_InvalidAxCfgDec
Description	Invalid deceleration configuration on axis
Function	Acm_SetF64Property
Example	While using Acm_SetF64Property API to set the maximum deceleration (CFG_AxMaxDec) on axis to 2,000,000,001 and an error occurs because the maximum of this value is 2,000,000,000.
Error Code	0x80005104
Message	Dsp_InvalidAxCfgJerk
Description	Invalid jerk configuration on axis
Function	Acm_SetF64Property
Example	
Error Code	0x80005105
Message	Dsp_InvalidAxParVelLow
Description	Invalid initial velocity parameter on axis
Function	Acm_SetF64Property
Example	While using Acm_SetF64Property API to set the initial velocity on axis (PAR_AxVelLow) and an error occurs when the value is less than 0 or greater than maximum driving velocity (CFG_AxMaxVel).
Error Code	0x80005106
Message	Dsp_InvalidAxParVelHigh
Description	Invalid driving velocity parameter on axis
Function	Acm_SetF64Property

Example	While using Acm_SetF64Property API to set the driving velocity on axis (PAR_AxVelHigh) and an error occurs when the value is less than 0 or greater than maximum driving velocity (CFG_AxMaxVel).
Error Code	0x80005107
Message	Dsp_InvalidAxParAcc
Description	Invalid acceleration parameter on axis
Function	Acm_SetF64Property
Example	While using Acm_SetF64Property API to set the acceleration on axis (PAR_AxAcc) and an error occurs when the value is less than 0 or the value is greater than maximum acceleration (CFG_AxMaxAcc).
Error Code	0x80005108
Message	Dsp_InvalidAxParDec
Description	Invalid deceleration parameter on axis
Function	Acm_SetF64Property
Example	While using Acm_SetF64Property API to set the deceleration on axis (PAR_AxDec) and an error occurs when the value is less than 0 or the value is greater than maximum deceleration (CFG_AxMaxDec).
Error Code	0x80005109
Message	Dsp_InvalidAxParJerk
Description	Invalid jerk parameter on axis
Function	Acm_SetF64Property
Example	While using Acm_SetF64Property API to set the type of velocity profile on axis (PAR_AxJerk) and an error occurs when the set value is not 0 or 1.
Error Code	0x8000510a
Message	Dsp_InvalidAxPptValue
Description	Invalid property value
Function	
Example	
Error Code	0x8000510b
Message	Dsp_InvalidAxState
Description	Invalid axis states
Function	Acm_AxMoveVel, Acm_AxStopDec...
Example	While using API to command axis to do continuous motion and an error occurs when the axis state is not Ready.
Error Code	0x8000510c
Message	Dsp_InvalidAxSvOnOff
Description	Invalid servo On/Off
Function	
Example	
Error Code	0x8000510d
Message	Dsp_InvalidAxDistance
Description	Invalid distance
Function	Acm_AxMoveImpose
Example	
Error Code	0x8000510e
Message	Dsp_InvalidAxPosition
Description	Invalid position
Function	
Example	
Error Code	0x8000510f

Message	Dsp_InvalidAxHomeMode
Description	Invalid home mode
Function	Acm_AxHome
Example	An error occurs when the homing mode is not supported.
Error Code	0x80005110
Message	Dsp_InvalidPhysicalAxis
Description	Invalid physical ID of axis
Function	Acm_DevMDaqConfig...
Example	While using Acm_DevMDaqConfig API to set MDAQ properties and an error occurs when the specified axis number is greater than the number of axes in the device.
Error Code	0x80005111
Message	Dsp_HLmtPExceeded
Description	Positive hardware limit has been exceeded
Function	
Example	
Error Code	0x80005112
Message	Dsp_HLmtNExceeded
Description	Negative hardware limit has been exceeded
Function	
Example	
Error Code	0x80005113
Message	Dsp_SLmtPExceeded
Description	Positive software limit has been exceeded
Function	
Example	
Error Code	0x80005114
Message	Dsp_SLmtNExceeded
Description	Negative software limit has been exceeded
Function	
Example	
Error Code	0x80005115
Message	Dsp_AlarmHappened
Description	Alarm happened
Function	
Example	
Error Code	0x80005116
Message	Dsp_EmgHappened
Description	Emergency happened
Function	
Example	
Error Code	0x80005117
Message	Dsp_CmdValidOnlyInConstSec
Description	This command is only valid in continuous section
Function	Acm_AxMoveImpose
Example	An error occurs while imposing a new motion but the current motion is not in continuous section.
Error Code	0x80005118
Message	Dsp_InvalidAxCmd

Description	Invalid axis command
Function	
Example	
Error Code	0x80005119
Message	Dsp_InvalidAxHomeDirMode
Description	Invalid axis HomeDirMode
Function	Acm_AxHome
Example	
Error Code	0x8000511a
Message	Dsp_AxisMustBeModuloAxis
Description	Axis must be modulo axis
Function	Acm_AxTangentInGp
Example	
Error Code	0x8000511b
Message	Dsp_AxIdCantSameAsMasId
Description	Axis ID cannot be same with master axis ID
Function	Acm_AxCamInAx, Acm_AxGearInAx, Acm_AxGantryInAx
Example	While using Acm_AxCamInAx API to establish relations of master and slave axis and an error occurs because the setting of slave axis is the same as master axis.
Error Code	0x8000511c
Message	Dsp_CantResetPosiOfMasAxis
Description	Cannot set command or actual position
Function	Acm_AxSetCmdPosition, Acm_AxSetActualPosition
Example	It cannot reset the position while axis is in synchronization status.
Error Code	0x8000511d
Message	Dsp_InvalidAxExtDrvOperation
Description	Invalid axis operation of external driver
Function	Acm_AxSetExtDrive
Example	
Error Code	0x8000511e
Message	Dsp_AxAccExceededMaxAcc
Description	Axis acceleration has exceeded max acceleration configuration
Function	
Example	
Error Code	0x8000511f
Message	Dsp_AxVelExceededMaxVel
Description	Axis velocity has exceeded max velocity configuration
Function	Acm_AxChangeVel
Example	While using API to set new velocity on axis and an error occurs when the set value is greater than maximum driving velocity.
Error Code	0x80005201
Message	Dsp_InvalidAxCntInGp
Description	The number of operating axes is inconsistent with the number of axes in group
Function	Acm_GpMoveLinearRel, Acm_GpMoveCircularRel...
Example	Command group to do 3-axis arc interpolation and an error occurs when only two axes in the group.
Error Code	0x80005202

Message	Dsp_AxInGpNotFound
Description	Axis not found in group
Function	Acm_GpRemAxis
Example	While using Acm_GpRemAxis API to remove the axis from group and an error occurs when the axis is not in the group.
Error Code	0x80005203
Message	Dsp_AxisInOtherGp
Description	Axis is already in other group
Function	Acm_GpAddAxis
Example	While using Acm_GpAddAxis API to add the axis into group and an error occurs when the axis is already in other group. Each axis can only exist within a group
Error Code	0x80005204
Message	Dsp_AxCannotIntoGp
Description	Axis cannot be added into group. Invalid axis or the number of axes in the group has reached the upper limit
Function	Acm_GpAddAxis
Example	While using Acm_GpAddAxis API to add the 9th axis into group and an error occurs because the axis count exceed the maximum count (up to 8 axes) in one group.
Error Code	0x80005205
Message	Dsp_GpInDevNotFound
Description	Group not found in device
Function	
Example	
Error Code	0x80005206
Message	Dsp_InvalidGpCfgVel
Description	Invalid velocity configuration on group
Function	
Example	
Error Code	0x80005207
Message	Dsp_InvalidGpCfgAcc
Description	Invalid acceleration configuration on group
Function	
Example	
Error Code	0x80005208
Message	Dsp_InvalidGpCfgDec
Description	Invalid deceleration configuration on group
Function	
Example	
Error Code	0x80005209
Message	Dsp_InvalidGpCfgJerk
Description	Invalid jerk configuration on group
Function	
Example	
Error Code	0x8000520a
Message	Dsp_InvalidGpParVelLow
Description	Invalid initial velocity parameter on group
Function	Acm_SetF64Property

Example	While using Acm_SetF64Property API to set the initial velocity on group (PAR_GpVelLow) and an error occurs when the value is less than 0 or the value is greater than maximum driving velocity (CFG_AxMaxVel).
Error Code	0x8000520b
Message	Dsp_InvalidGpParVelHigh
Description	Invalid driving velocity parameter on group
Function	Acm_SetF64Property
Example	While using Acm_GpChangeVel API to command group to change the velocity while group is in line interpolation motion and an error occurs when the value is less than 0 or the value is greater than maximum driving velocity (CFG_AxMaxVel).
Error Code	0x8000520c
Message	Dsp_InvalidGpParAcc
Description	Invalid acceleration parameter on group
Function	Acm_SetF64Property
Example	While using Acm_SetF64Property API to set the property PAR_GpAcc and an error occurs when the value is less than 0 or the value is greater than maximum acceleration (CFG_AxMaxAcc).
Error Code	0x8000520d
Message	Dsp_InvalidGpParDec
Description	Invalid deceleration parameter on group
Function	Acm_SetF64Property
Example	While using Acm_SetF64Property API to set the property PAR_GpDec and an error occurs when the value is less than 0 or the value is greater than maximum deceleration (CFG_AxMaxDec).
Error Code	0x8000520e
Message	Dsp_InvalidGpParJerk
Description	Invalid jerk parameter on group
Function	Acm_GpMoveCircularRel_Angle, Acm_GpMoveCircularAbs_Angle Acm_GpMoveArcRel_Angle, Acm_GpMoveArcAbs_Angle Acm_GpMoveHelixAbs_Angle, Acm_GpMoveHelixRel_Angle Acm_GpMove3DArcAbs_V, Acm_GpMove3DArcRel_V
Example	While using Acm_GpMoveCircularRel_Angle API to complete circular interpolation and an error occurs when the type of velocity profile (PAR_GpJerk) is S-Curve.
Error Code	0x8000520f
Message	Dsp_JerkNotSupport
Description	Jerk is not supported
Function	
Example	
Error Code	0x80005210
Message	Dsp_ThreeAxNotSupport
Description	Three axes are not supported
Function	
Example	
Error Code	0x80005211
Message	Dsp_DevlpoNotFinished
Description	Device interpolation is not finished
Function	
Example	
Error Code	0x80005212

Message	Dsp_InvalidGpState
Description	Invalid group state
Function	Acm_GpMoveLinearRel, Acm_GpMoveCircularRel...
Example	While using API to command group to do line interpolation motion and an error occurs when the group state is not Ready.
Error Code	0x80005213
Message	Dsp_OpenFileFailed
Description	Failed to open file
Function	
Example	
Error Code	0x80005214
Message	Dsp_InvalidPathCnt
Description	Invalid path count
Function	Acm_GpMovePath
Example	While using Acm_GpLoadPath API to load path file and an error occurs when the path count is less than 2.
Error Code	0x80005215
Message	Dsp_InvalidPathHandle
Description	Invalid path handle
Function	
Example	
Error Code	0x80005216
Message	Dsp_InvalidPath
Description	Invalid path
Function	
Example	
Error Code	0x80005217
Message	Dsp_GpSlavePositionOverMaster
Description	Slave position is over master position
Function	
Example	
Error Code	0x80005219
Message	Dsp_GpPathBufferOverflow
Description	Path buffer in group is overflow(7,000)
Function	Acm_GpAddPath
Example	While using Acm_GpAddPath API to add path and an error occurs when the path count is greater than 7,000.
Error Code	0x8000521a
Message	Dsp_InvalidPathFunctionID
Description	Invalid path function ID
Function	
Example	
Error Code	0x8000521b
Message	Dsp_SysBufAllocateFailed
Description	Failed to allocate system buffer
Function	
Example	
Error Code	0x8000521c
Message	Dsp_InvalidGpCenterPosition

Description	Invalid center position of group
Function	Acm_GpMove3DArcRel, Acm_GpMove3DArcAbs, Acm_GpMove3DArcAbs_V, Acm_GpMove3DArcAbs_V
Example	While using API to do 3-dimension arc interpolation and an error occurs when the parameter of center position of group is incorrect.
Error Code	0x8000521d
Message	Dsp_InvalidGpEndPosition
Description	Invalid end position of group
Function	Acm_GpMove3DArcRel, Acm_GpMove3DArcAbs
Example	While using API to do 3-dimension arc interpolation and an error occurs when the parameter of end position of group is incorrect.
Error Code	0x8000521e
Message	Dsp_InvalidGpCmd
Description	Invalid group command
Function	
Example	
Error Code	0x8000521f
Message	Dsp_AxHasBeenInGp
Description	Axis is already in group
Function	Acm_GpAddAxis, Acm_AxGearInAx...
Example	While using Acm_AxGearInAx API to establish the gear relations and an error occurs when the axis has established gear relations with other axis.
Error Code	0x80005220
Message	Dsp_InvalidPathRange
Description	Invalid path range
Function	Acm_GpMoveSelPath
Example	There are only 10 paths in path buffer(it can be known by calling Acm_GpGetPathStatus to retrieve the rest count of path) and an error occurs when the set value of parameter StartIndex or EndIndex is greater than 10 or the set value of StartIndex is greater than EndIndex.

C.4 EtherCAT Common Errors

Error Code	0x830002B
Message	EC_SlaveIDConflicted
Description	The slave ID is conflicted
Function	Acm_DevOpen, Acm_DevReOpen
Example	An error occurs when two slaves with same ID connected to the same ring.
Error Code	0x830003D
Message	EC_AxCntExceeded
Description	Connected axis count exceed maximum support axis count.
Function	Acm_DevOpen, Acm_DevReOpen
Example	An error occurs when count of motor drivers connected PCI-1203 exceeded the maximum support count.
Error Code	0x830003E
Message	EC_FWUpgraded
Description	FW upgraded, turn off and on the computer to finish the FW update process.
Function	Acm_DevOpen, Acm_DevReOpen
Example	
Error Code	0x83010001
Message	ECAT_MasterEcatError
Description	EtherCAT slave alarm
Function	
Example	
Error Code	0x83010002
Message	ECAT_SlaveDisconnect_R0
Description	Slave disconnect on Ring 0
Function	
Example	
Error Code	0x83010003
Message	ECAT_SlaveDisconnect_R1
Description	Slave disconnect on Ring 1
Function	
Example	
Error Code	0x83030004
Message	ECAT_MOTION_SLAVE_NOT_SUPPORT
Description	Slave is not supported on Ring 0
Function	Acm_DevOpen, Acm_DevReOpen
Example	
Error Code	0x83040004
Message	ECAT_IO_SLAVE_NOT_SUPPORT
Description	Slave is not supported on Ring 1. IO ring does not support motor drive
Function	Acm_DevOpen, Acm_DevReOpen
Example	
Error Code	0x83050001
Message	ECAT_AxRetryError
Description	Slave is not response (timeout) when reset error or reset position counter

Function	Acm_AxResetError, Acm_AxSetCmdPosition
Example	
Error Code	0x83050002
Message	ECAT_AxResetCounterError
Description	Slave error when reset position counter
Function	Acm_AxSetCmdPosition
Example	
Error Code	0x83050003
Message	ECAT_AxCmdErrorProtection
Description	Command error protection
Function	
Example	When switching the operation mode such as switching from homing mode to P2P mode, the error occurred if the difference between command position and actual position is large than the command error protection value.
Error Code	0x83050004
Message	ECAT_AxSlaveALM
Description	EtherCAT slave alarm
Function	
Example	
Error Code	0x83050005
Message	ECAT_AxFollowError
Description	Following error protection
Function	
Example	An error occurs when the difference between command position and actual position of the motor is large than the setting of drive.
Error Code	0x83050006
Message	ECAT_AxHomeError
Description	Slave error when operating homing motion
Function	Acm_AxMoveHome, Acm_AxHomeEx
Example	
Error Code	0x8310xxxx
Message	ECAT_DriveError
Description	Drive error (0x8310xxxx), please refer to drive manual or drive LED panel for error xxxx.
Function	
Example	

C.5 Unusual Errors

Error Code	Message	Description
0x80000001	DevRegDataLost	
0x80000002	LoadDIIFailed	
0x80000003	GetProcAddrFailed	
0x80000007	OpenEventFailed	
0x80000008	EventTimeOut	
0x8000000C	ConnectWinIrqFailed	
0x80000018	InvalidAxAlarmEn	
0x80000019	InvalidAxAlarmLogic	
0x8000001A	InvalidAxInPEn	
0x8000001B	InvalidAxInPLogic	
0x8000001C	InvalidAxHLmtEn	
0x8000001D	InvalidAxHLmtLogic	
0x8000001E	InvalidAxHLmtReact	
0x8000001F	InvalidAxSLmtPEn	
0x80000020	InvalidAxSLmtPReact	
0x80000021	InvalidAxSLmtPValue	
0x80000022	InvalidAxSLmtMEn	
0x80000023	InvalidAxSLmtMReact	
0x80000024	InvalidAxSLmtMValue	
0x80000025	InvalidAxOrgLogic	
0x80000026	InvalidAxOrgEnable	
0x80000027	InvalidAxEzLogic	
0x80000028	InvalidAxEzEnable	
0x80000029	InvalidAxEzCount	
0x8000002B	InvalidAxInEnable	
0x8000002C	InvalidAxSvOnOff	
0x8000002E	InvalidAxPosition	
0x8000002F	InvalidAxHomeModeKw	
0x80000034	GpInDevNotFound	
0x80000035	InvalidGpCfgVel	
0x80000036	InvalidGpCfgAcc	
0x80000037	InvalidGpCfgDec	
0x80000038	InvalidGpCfgJerk	
0x80000040	DevIpoNotFinished	
0x80000046	IoctlError	
0x80000047	AmnetRingWhile usingd	
0x80000048	DeviceNotOpened	
0x80000049	InvalidRing	
0x8000004A	InvalidSlaveIP	
0x8000004B	InvalidParameter	
0x8000004C	InvalidGpCenterPosition	
0x8000004D	InvalidGpEndPosition	
0x8000004E	InvalidAddress	
0x8000004F	DeviceDisconnect	

0x80000050	DataOutBufExceeded
0x80000051	SlaveDeviceNotMatch
0x80000052	SlaveDeviceError
0x80000053	SlaveDeviceUnknow
0x80000056	InvalidVelocity
0x80000057	InvalidAxPulseInLogic
0x80000058	InvalidAxPulseInSource
0x80000059	InvalidAxErcLogic
0x8000005A	InvalidAxErcOnTime
0x8000005B	InvalidAxErcOffTime
0x8000005C	InvalidAxErcEnableMode
0x8000005D	InvalidAxSdEnable
0x8000005E	InvalidAxSdLogic
0x8000005F	InvalidAxSdReact
0x80000060	InvalidAxSdLatch
0x80000061	InvalidAxHomeResetEnable
0x80000062	InvalidAxBacklashEnable
0x80000063	InvalidAxBacklashPulses
0x80000064	InvalidAxVibrationEnable
0x80000065	InvalidAxVibrationRevTime
0x80000066	InvalidAxVibrationFwdTime
0x80000067	InvalidAxAlarmReact
0x80000068	InvalidAxLatchLogic
0x80000069	InvalidFwMemoryMode
0x8000006A	InvalidConfigFile
0x8000006B	InvalidAxEnEvtArraySize
0x8000006C	InvalidAxEnEvtArray
0x8000006D	InvalidGpEnEvtArraySize
0x8000006E	InvalidGpEnEvtArray
0x8000006F	InvalidIntervalData
0x80000070	InvalidEndPosition
0x80000071	InvalidAxisSelect
0x80000072	InvalidTableSize
0x80000074	InvalidCmpSource
0x80000076	InvalidCmpPulseMode
0x80000077	InvalidCmpPulseLogic
0x80000078	InvalidCmpPulseWidth
0x80000079	InvalidPathFunctionID
0x8000007A	SysBufAllocateFailed
0x80000096	SlaveIOUpdateError
0x80000097	NoSlaveDevFound
0x80000098	MasterDevNotOpen
0x80000099	MasterRingNotOpen
0x800000C8	InvalidDIPort
0x800000C9	InvalidDOPort
0x800000CA	InvalidDOValue
0x800000CC	CreateThreadFailed
0x800000CD	InvalidHomeModeEx

0x800000CE	InvalidDirMode
0x800000CF	AxHomeMotionFailed
0x800000D1	PathBufIsFull
0x800000D2	PathBufIsEmpty
0x800000D3	GetAuthorityFailed
0x800000D4	GplDAllocatedFailed
0x800000D5	FirmWareDown
0x800000D7	InvalidAxCmd
0x800000D8	InvalidaxExtDrv
0x800000DA	SpeedCurveNotSupported
0x800000DB	InvalidCounterNo
0x800000DD	PathSelStartCantRunInSpeed- ForewareMode
0x80002000	HLmtPExceeded
0x80002001	HLmtNExceeded
0x80002002	SLmtPExceeded
0x80002003	SLmtNExceeded
0x80002004	AlarmHappened
0x80002005	EmgHappened
0x80002006	TimeLmtExceeded
0x80002007	DistLmtExceeded
0x80002008	InvalidPositionOverride
0x80002009	OperationErrorHappened
0x8000200A	SimultaneousStopHappened
0x8000200B	OverflowInPAPB
0x8000200C	OverflowInIPO
0x8000200D	STPHappened
0x8000200E	SDHappened
0x8000200F	AxsiNoCmpDataLeft

Error Code	Message	Description
0x83000000	EC_GetNICNumberFailed	
0x83000001	EC_GetNICInfoFailed	
0x83000002	EC_OpenMasterDevFailed	
0x83000003	EC_GetSlaveFailed	
0x83000004	EC_StartOpModeFailed	
0x83000005	EC_CloseDeviceFailed	
0x83000006	EC_MemAllocateFailed	
0x83000007	EC_InvalidNicIndex	
0x83000008	EC_OpenDevFailed	
0x83000009	EC_ReadFileFailed	
0x8300000A	EC_GetNicInfoFailed	
0x8300000B	EC_GetSDOFailed	
0x8300000C	EC_InvalidParameter	
0x8300000D	EC_GetPDOOffsetFailed	
0x8300000E	EC_InitialMappingInfoFalied	
0x8300000F	EC_InitResourceFailed	
0x83000010	EC_SetSDOFailed	

0x83000011	EC_InvalidPortType
0x83000012	EC_SetCycleTimeFailed
0x83000013	EC_InvalidAoRange
0x83000014	EC_InvalidAiRange
0x83000015	EC_GetSlaveInfoFailed
0x83000016	EC_GetNetWorkStateFailed
0x83000017	EC_RegisterEventFailed
0x83000018	EC_InvalidIntegrationTime
0x83000019	EC_InvalidAIEnable
0x8300001A	EC_InvalidDIFilter
0x8300001B	EC_SetSlaveStateFailed
0x8300001C	EC_ZeroCalibrationFailed
0x8300001D	EC_InvalidMasterHandle
0x8300001E	EC_InvalidENIFile
0x8300001F	EC_InvalidCaliType
0x83000020	EC_SetCaliValueFailed
0x83000021	EC_AOCalibrationFailed
0x83000022	EC_InvalidIOMapping
0x83000023	EC_PortIndexGreaterThanPort Num
0x83000024	EC_ChannelIDGreaterThanCha nnelNum
0x83000025	EC_InputIndexGreaterThanInpu tNum
0x83000026	EC_OutputIndexGreaterThanO utputNum
0x83000027	EC_SetEnableFailed
0x83000028	EC_SetAIRangeFailed
0x83000029	EC_SetIntegrationTimeFailed
0x8300002A	EC_PropertyNotSupported
0x8300002B	EC_SlaveIDConflicted
0x8300002C	EC_SpanCalibrationFailed
0x8300002D	EC_InvalidAiValue
0x8300002E	EC_InvalidAoValue
0x8300002F	EC_GetModuleFailed
0x83000030	EC_InvalidCntEnable
0x83000031	EC_InvalidCntPulseInMode
0x83000032	EC_InvalidCntInitValue
0x83000033	EC_InvalidCntMaxValue
0x83000034	EC_InvalidCntOverflowMode
0x83000035	EC_InvalidCntLatchEnable
0x83000036	EC_InvalidCntLatchEdge
0x83000037	EC_InvalidCntCmpEnable
0x83000038	EC_InvalidCntCmpMethod
0x83000039	EC_InvalidCntCmpPulseEnable
0x8300003A	EC_InvalidCntCmpPulseMode
0x8300003B	EC_InvalidCntCmpPulseLogic
0x8300003C	EC_InvalidCntCmpPulseWidth

www.advantech.com

Please verify specifications before quoting. This guide is intended for reference purposes only.

All product specifications are subject to change without notice.

No part of this publication may be reproduced in any form or by any means, electronic, photocopying, recording or otherwise, without prior written permission of the publisher.

All brand and product names are trademarks or registered trademarks of their respective companies.

© Advantech Co., Ltd. 2016