
4-axis Motion Control Card
ADT-8949

Basic Information

This Manual is written by Adtech (Shenzhen) Technology Co., Ltd. This Manual is mainly written by: Ai Xiaoyun.

This Manual was first released on May 12, 2013, with version No. 15.05.16.11 and item number BZ001C057A.

Copyright Notice

The property rights of all the parts of the manual belong to Adtech (Shenzhen) Technology Co., Ltd. (Adtech for short), and any form of imitation, copying, transcription or translation by any company or individual without the permission is prohibited. This manual does not include any form of assurance, standpoint expression, or other intimations. Adtech and the stuffs have no responsibility for any direct or indirect disclosure of the information, benefit loss or business termination of this manual of the quoted product information. In addition, the product and the information mentioned in this manual are for reference only, and the content is subject to change without notice.

ALL RIGHTS RESERVED!

Adtech (Shenzhen) Technology Co., Ltd.

Precautions

◆ Transport and storage:

- ☞ Do not stack product package more than six layers;
- ☞ Do not climb, stand on or place heavy stuff on the product package;
- ☞ Do not pull the cable still connecting with machine to move product.
- ☞ Forbid impact and scratch on the panel and display;
- ☞ Prevent the product package from humidity, sun exposure, and rain.

◆ Open-box inspection:

- ☞ Open the package to confirm the product to be purchased by you.
- ☞ Check damages situation after transportation;
- ☞ Confirm the integrity of parts comparing with the parts list or damages situation;
- ☞ Contact our company promptly for discrepant models, shortage accessories, or transport damages.

◆ Wiring

- ☞ Ensure the persons involved into wiring and inspecting are specialized staff;
- ☞ Guarantee the product is grounded with less than 4Ω grounding resistance. Do not use neutral line (N) to substitute earth wire.
- ☞ Ensure grounding to be correct and solid, in order to avoid product failures or unexpected consequences;
- ☞ Connect the surge absorption diodes to the product in the required direction, otherwise, the product will be damaged;
- ☞ Ensure the power switch is OFF before inserting or removing plug, or disassembling chassis.

◆ Overhauling

- ☞ Ensure the power is OFF before overhauling or components replacement;
- ☞ Make sure to check failures after short circuit or overloading,

and then restart the machine after troubleshooting

- ☞ Do not allow to frequently connect and disconnect the power, and at least one minute interval between power-on and power-off.

◆ Miscellaneous

- ☞ Do not open housing without permission;
- ☞ Keep power OFF if not in use for a long time;
- ☞ Pay close attention to keep dust and ferrous powder away from control;
- ☞ Fix freewheel diode on relay coil in parallel if non-solid state relay is used as output relay. Check whether power supply meets the requirement to ensure not burning the control.
- ☞ Install cooling fan if processing field is in high temperature, due to close relationship between service life of the control and environmental temperature. Keep proper operative temperature range for the control: 0℃ ~ 60℃.
- ☞ Avoid using the product in the overheating, humid, dusty, or corrosive environments;
- ☞ Add rubber rails as cushion on the place with strong vibration.

◆ Maintenance

Please implement routine inspection and regular check upon the following items, under the general usage conditions (i.e. environmental condition: daily average 30℃, load rate: 80%, and operating rate: 12 hours/ day)

Table 1 Equipment Inspection Requirements

Routine Inspection	Routine	<ul style="list-style-type: none"> ● Confirm environmental temperature, humidity, dust, or foreign objects. ● Confirm abnormal vibration and noise; ● Check whether vents are blocked by yarn etc.
--------------------	---------	---

Regular Check	One year	<ul style="list-style-type: none"> • Check whether solid components are loose • Confirm whether terminal block is damaged
------------------	-------------	---

Contents

Contents.....	4
Overview.....	8
Chapter 1 Product Introduction.....	8
1.1 Description:.....	8
1.2 Hardware specifications:.....	8
1.3 Control functions:.....	9
1.4 Software support:.....	11
1.5 Applications:.....	11
1.6 ADT-8949 series features list:.....	11
Hardware.....	13
Chapter 1 Hardware Installation.....	13
1.1 Packing list:.....	13
1.2 Installation size:.....	14
1.3 Installation steps:.....	14
Chapter 2 Electrical Connection.....	15
2.1 Wiring diagram:.....	15
2.2 25-pin port I/O signal definition:.....	15
2.4 Signal definition of J3 interface.....	21
2.5 J4 signal definition, reserved, no specific function.....	23
2.6 J5 signal definition.....	23
2.7 Connecting pulse/direction output signal:.....	23
2.8 Connecting encoder input signal:.....	25
2.9 Connecting digital input:.....	25
2.10 Connecting digital output:.....	29
Chapter 3 Software Installation.....	31
3.1 Installing drivers on Win98:.....	32
3.2 Installing drivers on WinXP:.....	34
3.3 Installing drivers on Win7.....	36
Chapter 4 Electrical Specifications.....	41
4.1 Switch input:.....	41
4.2 Count input:.....	42
<hr/>	
<i>http://www.adtechcn.com</i>	
	4

4.3 Pulse output:	42
4.4 Switching output:	42
4.5 Power output:	42
Chapter 5 Common Servo Wiring Diagrams	43
5.1 Panasonic A5 servo wiring diagram:	43
Chapter 6 Working Environment	44
6.1 Operating temperature:	44
6.2 Storage temperature:	45
6.3 Operating humidity:	45
6.4 Storage humidity:	45
Software Programming	45
Chapter 1 Function Description	45
1.1 Pulse output mode:	45
1.2 Hardware limit signal:	46
1.3 Quantitative driving:	46
1.4 Continuous driving:	47
1.5 Speed curve:	48
1.6 Position latch:	53
1.7 External signal driving:	53
1.8 Large cache small segment:	53
1.9 Speed adaptive model:	53
1.10 Spherical arc interpolation:	55
1.11 NURBS interpolation:	55
1.12 Simultaneous control of multiple processes:	56
1.13 Gantry dual-drive, changing drive speed and target position in motion:	56
1.14 Specify the time for four-axis linear interpolation:	56
1.15 Pulse generator:	56
1.16 Get arc length, set 15 filter levels for input point:	57
1.17 Helical interpolation:	57
Chapter 2 Motion Control Library Function Guide	58
2.1 Introduction of ADT-8949 function library	58
2.2 Calling DLL on Windows:	58
2.3 Calling in VC:	58
2.4 Calling in VB:	58
2.5 Calling in C++Builder:	59
2.6 Return value and meaning of library function:	59

Chapter 3 Key Points for Motion Control Development.....	61
3.1 Card initialization.....	61
3.2 Speed setting.....	62
Chapter 4 System Security Mechanism.....	63
4.1 Monitoring error message:.....	63
4.2 Limit:.....	63
Chapter 5 High Speed Capture of External Signal Homing.....	64
5.1 Homing motion:.....	64
Chapter 6 Linkage Control.....	65
6.1 Single axis quantitative uniform motion:.....	66
6.2 Single axis quantitative symmetry trapezoidal acceleration / deceleration motion:.....	66
6.3 Single axis quantitative asymmetry trapezoidal acceleration / deceleration motion:.....	67
6.4 Single axis quantitative S-curve acceleration / deceleration motion:.....	68
6.5 Single axis quantitative exponential acceleration / deceleration motion:.....	68
6.6 Single axis quantitative trigonometric acceleration and deceleration mode motion:.....	69
6.7 Multi-axis motion.....	70
Chapter 7 Interpolation Motion Control.....	71
7.1 Two-axis linear interpolation (constant speed).....	72
7.2 Two-axis linear interpolation (acceleration / deceleration).....	72
7.3 2D arc interpolation (acceleration/deceleration).....	73
7.4 3D arc interpolation (acceleration/deceleration).....	74
Chapter 8 Track Motion Control.....	76
8.1 Cache interpolation.....	76
Chapter 9 Universal Digital I/O.....	78
9.1 Input port definition: See Hardware CHAPTER 2 Electrical Connection for specific definition.....	78
9.2 Output port definition: See Hardware CHAPTER 2 Electrical Connection for specific definition.....	78
9.3 Output port:.....	78
9.4 Input port:.....	78
Chapter 10 Auxiliary Control.....	79

11.1 Position locking:.....	79
Chapter 11 List of ADT8949 Basic Library Functions.....	80
Chapter 12 Detailed Description of ADT8949 Basic Library Functions.....	86
12.1 Basic parameter setting.....	86
12.2 Reset motion card.....	96
12.3 Driving status checking.....	96
12.4 Motion parameter setting.....	99
12.5 Parameter checking.....	107
12.6 Driving.....	111
12.7 Switch quantity.....	122
12.8 FIFO operation output.....	126
12.9 Position locking.....	130
12.10 Homing module.....	135
12.11 One-dimensional position comparator.....	138
12.12 Helical interpolation.....	143
12.13 Handwheel function (not enabled).....	145
Chapter 13 Troubleshooting.....	146
13.1 Motion control card detections fails.....	146
13.2 Motor running abnormal.....	146
13.3 Switch input abnormal.....	148
13.3 Switch output abnormal.....	149
13.4 Encoder abnormal.....	150

Overview

Chapter 1 Product Introduction

1.1 Description:

ADT-8949 multi-axis motion control card is one of the high-performance four-axis motion control cards of Adtech based on PCI bus. One system supports up to 10 control cards and controls 40 channels of servo / stepper motor. It features T-shaped, S-shaped and E-shaped acceleration and deceleration, point and track motion planning, electronic gear, electronic cam synchronized motion planning, linear interpolation, arc interpolation planning, splines, follow and other functions. It is especially suitable for occasions that have high-speed and high-precision position control requirement, and is widely used in testing, semiconductor packaging, mechanical arm, dispensing, packaging, engraving, and PCB processing.

1.2 Hardware specifications:

- 32-bit PCI bus, plug and play.
- Controllable axes: Four-axis pulse control.
- Maximum pulse output frequency 5MHz.
- Four-axis encoder feedback, frequency up to 4MHz, A/B phase difference pulse input and upper/lower pulse input are available, 32-bit count, 4 ratios.
- Support hardware serial number, allow third-party encryption.
- Four-position DIP switch, allow specifying card number 0-9.
- DSP + FPGA chip dedicated motion technology, provide high-speed and high-performance track smoothing and speed optimization.
- Pulse output types: Pulse + direction (PUL+DIR) or double pulse (CW + CCW).
- 36 channels digital inputs, all optically isolated.

- 32 channels open collector output.
- 2 channels DA output. Range: 0-10V, precision: 0.01V.

1.3 Control functions:

- Hardware cache: large capacity multi-axis hardware interpolation cache, store up to 10,000 interpolation instructions, continuous small line segments and large cache, used for multi-segment continuous track applications, such as engraving machine or cutting machine, so that the discrete data of CAM can be restored to the processing model.
- Speed preview: small line segment pretreatment, speed adaptive model, ensure high speed under high precision, automatic speed optimization, used for milling machine, tooling and other applications requiring high precision control, making the machine run smoothly from speed planning, and track error can be controlled.
- Multi-axis linkage: each axis moves or stops independently in accordance with the set speed and target position, and the driving speed can be changed in real time.
- Any 2-4 axis linear interpolation, 1-4 axis hardware buffer interpolation, 2-3 axis hardware arc interpolation, helical interpolation based on plane arc.
- 3D arc interpolation (spherical interpolation): achieve arc in any spatial plane and spherical arc. 3D hardware-level arc interpolation, support for hardware buffer interpolation, only occupies four data segments, and arc approximation accuracy is determined by interpolation speed dynamically to avoid discrete error of small segment approximation and contradictions of difficult speed tradeoff.
- Various acceleration and deceleration modes: T-shaped, S-shaped,

E-exponential and C-cosine, support asymmetric acceleration and deceleration, smooth running, quiet motor

- NURBS interpolation: Compared to the traditional linear interpolation and arc interpolation, achieve curve and surface track; NURBS interpolation can express various curve and surface tracks more accurately to improve curve and surface control accuracy.
- Synchronous dual drive: Two-axis pulse strict synchronization for dual-drive control in gantry structure.
- Position locking: In locking mode, when the locking signal is triggered, the hardware locks the logic location and actual position of the rising edge and falling edge, and can be used for quick homing, position measurement and other applications.
- Hardware cache IO event: In hardware cache function, not only the motion instructions but also IO output action and pulse generator can be cached. After the set axis of pulse generator reaches the specified location, the specified output port flips the level, and the flip times and frequency can be set. IO output cache can easily achieve precise position comparison output.
- Signal filtering: 15 filter levels can be set for input point to increase the anti-jamming capability of input signal
- Simultaneous control of multiple processes: You can open two programs to control one card. (A single monitoring program run simultaneously with the execution program; the execution program doesn't need to switch a lot of time for display, which makes the display more real-time.
- Various homing modes: Before precise motion, it is necessary to set the home of mechanical coordinates, usually perform mechanical homing; the action is to reset. The system provides a variety of homing modes to facilitate efficient homing.
- Hardware upgrade: Upgrade dynamic library to complete hardware

upgrade, which facilitate hardware system upgrade and client features customization.

1.4 Software support:

Operating system: DOS, WINDOWS95/98/NT/2000/XP, WINCE, WIN7

Programming environment: C/BC++/VC/VB/C#/C++Builder/ Delphi/ LabVIEW/ EVC

Application examples of open-DOS and Windows

1.5 Applications:

- Machine vision, automatic test equipment, AOI;
- Biological, medical automatic sampling equipment;
- Cutting equipment: diamond cutter, sponge cutting machine;
- Dispensing industry;
- Semiconductor packaging industry: Bonder;
- Advertising industry: CNC bending machine;
- Packaging and printing equipment: printer, pad printer;
- Engraving equipment;
- Industrial robot equipment;
- PCB processing, SMT and other industries;

1.6 ADT-8949 series features list:

√ Yes, - No, * optional

No.	Function Name	ADT-8949A1 (Universal)	ADT-8949B1 (Track)	ADT-8949G1 (Enhanced)
1	Pulse output mode: Pulse + direction, pulse + pulse	√	√	√

2	4 channels incremental encoder input	√	√	√
3	2 channels DA (0V~+10V)	√	√	√
4	36 channels digital signal input	√	√	√
5	32 channels digital signal output	√	√	√
6	Home signal input	√	√	√
7	Limit signal input	√	√	√
8	Hardware emergency stop input	√	√	√
9	Drive alarm signal input	√	√	√
10	Drive enable signal output	√	√	√
11	Driving reset signal output	√	√	√
12	Multi-axis linkage (point motion)	√	√	√
13	2-4 axis linear interpolation	√	√	√
14	2-axis hardware arc interpolation	√	√	√
15	3-axis hardware arc interpolation	-	√	√
16	1-4 axis hardware cache interpolation	√	√	√
17	Hardware cache IO event	√	√	√

18	Speed preview	-	√	√
19	Various acceleration and deceleration modes (T, S, E, C)	√	√	√
20	Asymmetric acceleration and deceleration	-	√	√
21	Online changing of drive speed	√	√	√
22	Position locking	√	√	√
23	Various homing mode	√	√	√
24	Signal filtering	√	√	√
25	Multiple processes simultaneously control	√	√	√
26	NURBS interpolation	-	-	√
27	Synchronous dual-drive (semi-closed)	-	-	√
28	Hardware upgrade	-	-	√

Hardware

Chapter 1 Hardware Installation

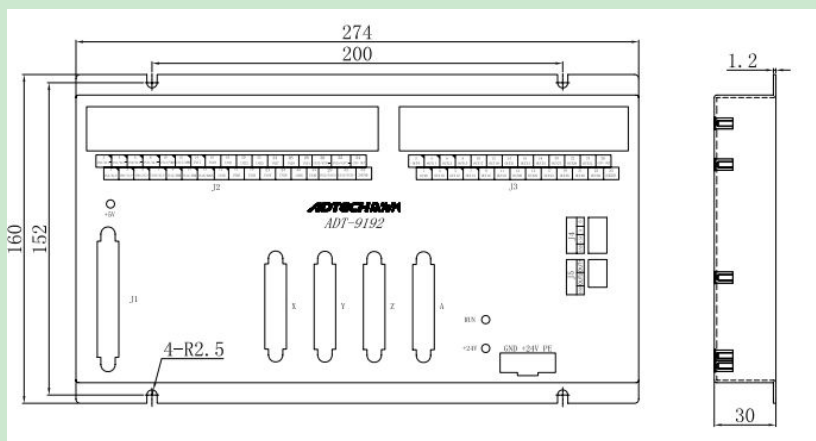
1.1 Packing list:

No.	Name	Description	Qty.
1	User's Manual	Instructions	1
2	User CD	SDK, examples and other information package	1

3	ADT-8949	Four-axis motion card	1
4	ADT-9192	Terminal box	1
5	ADT-D62GG	Data cable	1

1.2 Installation size:

The appearance and installation dimensions of ADT-9192 are as follows:

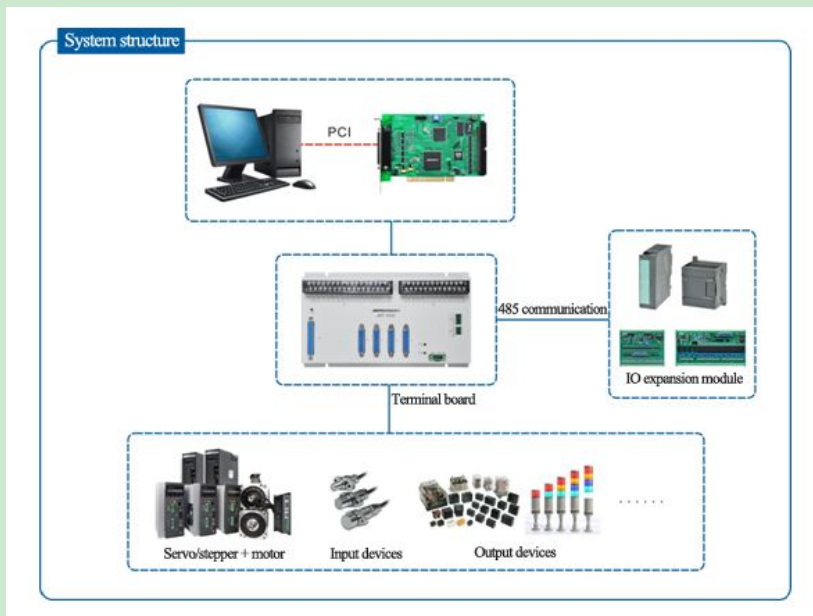


1.3 Installation steps:

1. Turn off the PC (Note: Turn off the master switch for ATX power supply)
2. Open the rear cover of PC case
3. Select an unoccupied PCI slot, and insert ADT-8949.
4. Ensure that the gold finger of ADT-8949 is completely inserted into the PCI slot, and tighten the screws.
5. Connect one end of the D62GG cable to J4 interface of the control card and connect the other end to J1 of ADT-9192 board.

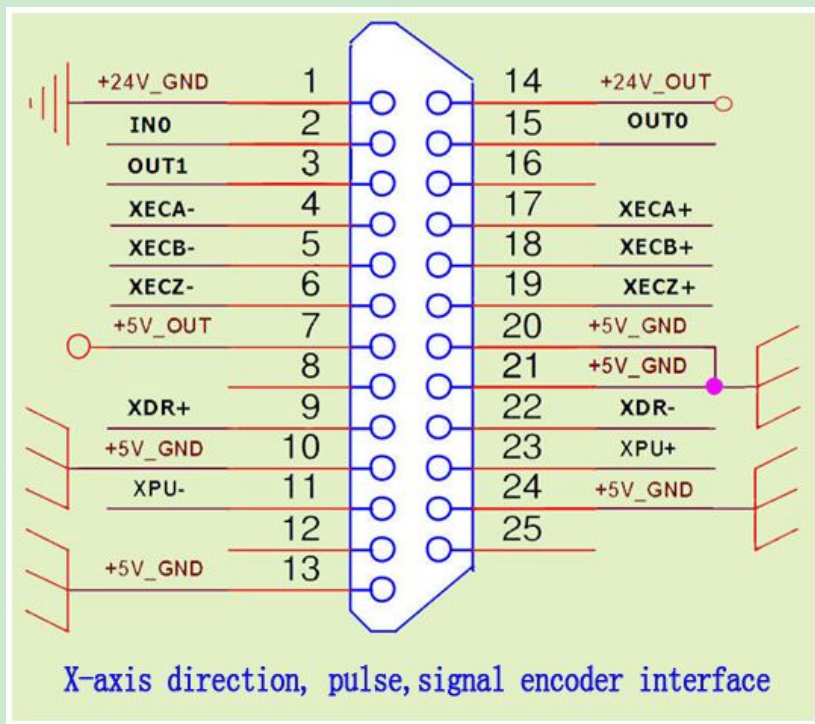
Chapter 2 Electrical Connection

2.1 Wiring diagram:



2.2 25-pin port I/O signal definition:

Four DB terminals correspond to four axes (XYZA). Below is the example of X axis. Other axes are similar.

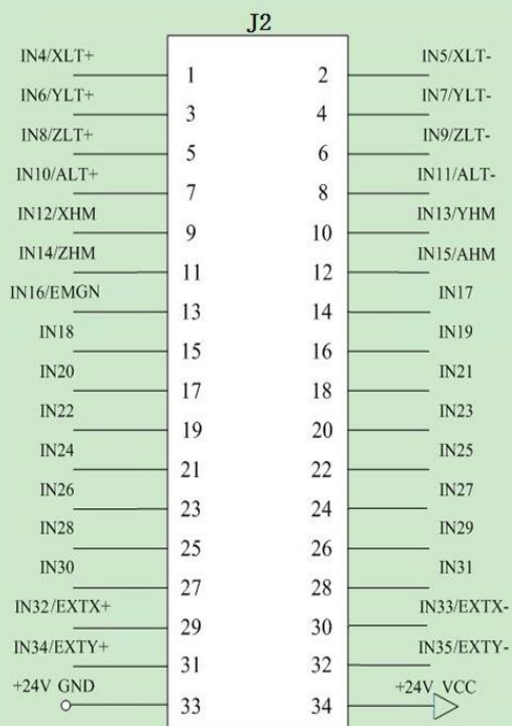


No.	Signal name	Definition
1		External 24V power grounding
2	EXT_IN0	External input, low level active, does not support two-wire sensor connection.
3	EXT_OUT1	Output control signal, opto-isolated output, low voltage level active
4	XECA-	X-axis encoder A phase input negative, can be used as an general input point, corresponding to sample program input point IN36, Y-axis, Z-axis and A-axis correspond to IN37, IN38, and IN39 respectively. When used as general input point,

		refer to 3.9 digital input connection - encoder signal as general input for wiring.
5	XECB-	X-axis encoder B phase input negative, can be used as an general input point, corresponding to sample program input point IN40 , Y-axis, Z-axis and A-axis correspond to IN41, IN42, and IN43 respectively. When used as general input point, refer to 3.9 digital input connection - encoder signal as general input for wiring.
6	XECZ-	X-axis encoder Z phase input negative, can be used as an general input point, X-axis corresponds to XSTOP1 (IN44) , Y-axis corresponds to YSTOP1 (IN45) , Z-axis corresponds to ZSTOP1 (IN46) and A-axis corresponds to ASTOP1 (IN47) . When used as general input point, refer to 3.9 digital input connection - encoder signal as general input for wiring.
7	VCC	+5V power output (can't be connected to external power supply)
8	NC	
9	XDR+	X axis direction positive signal
10	GND	5V power grounding
11	XPU-	X axis pulse negative signal
12	NC	
13	GND	5V power grounding
14	OVCC	+24V power output (can't be connected to external 24V+)
15	EXT_OUT0	Output control signal, opto-isolated output, low voltage level active

16	NC	
17	XECA+	X axis encoder phase A input positive
18	XECB+	X axis encoder phase B input positive
19	XECZ+	X axis encoder phase Z input positive
20	GND	5V power grounding
21	GND	5V power grounding
22	XDR-	X axis direction negative signal
23	XPU+	X axis pulse positive signal
24	GND	5V power grounding
25	NC	

2.3 Signal definition of J2 interface:

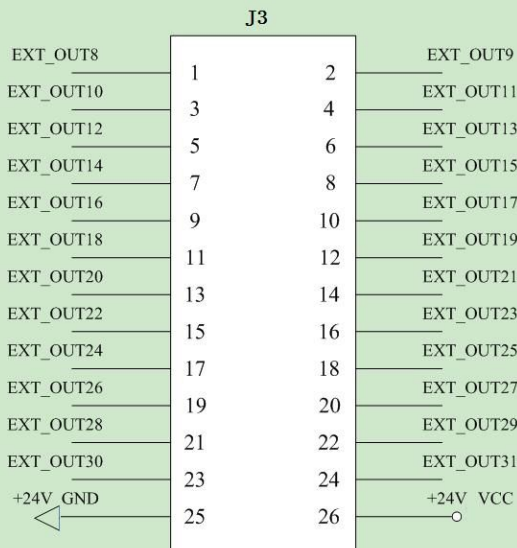


34-bit double dislocation terminal wiring is defined as below

1	IN4/XLT+	X positive limit signal, can be used as general input
2	IN5/XLT-	X negative limit signal, can be used as general input
3	IN6/YLT+	Y positive limit signal, can be used as general input
4	IN7/YLT-	Y negative limit signal, can be used as general input
5	IN8/ZLT+	Z positive limit signal, can be used as general input
6	IN9/ZLT-	Z negative limit signal, can be used as general input
7	IN10/ALT+	A positive limit signal, can be used as general input
8	IN11/ALT-	A negative limit signal, can be used as general input
9	IN12/XHM	X home signal (STOP0), can be used as general input
10	IN13/YHM	Y home signal (STOP0), can be used as general input
11	IN14/ZHM	Z home signal (STOP0), can be used as general input
12	IN15/AHM	A home signal (STOP0), can be used as general input
13	IN16/EMGN	Emergency stop signal, can be used as general input
14	IN17	General input
15	IN18	General input
16	IN19	General input

17	IN20	General input
18	IN21	General input
19	IN22	General input
20	IN23	General input
21	IN24	General input
22	IN25	General input
23	IN26	General input
24	IN27	General input
25	IN28	General input
26	IN29	General input
27	IN30	General input
28	IN31	General input
29	IN32/EXTX+	X manual forward rotation signal, can be used as general input
30	IN33/EXTX-	X manual reverse rotation signal, can be used as general input
31	IN34/EXTY+	Y manual forward rotation signal, can be used as general input
32	IN35/EXTY-	Y manual reverse rotation signal, can be used as general input
33	EXT_+24V GND	24V power grounding
34	EXT_+24V VCC	+24V power output (can't be connected to external 24V+)

2.4 Signal definition of J3 interface



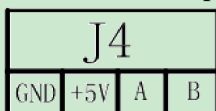
26-bit double dislocation terminal wiring is defined as below

Wire No.	Name	Function
1	EXT_OUT8	Output control signal, opto-isolated output, low voltage level active
2	EXT_OUT9	Output control signal, opto-isolated output, low voltage level active
3	EXT_OUT10	Output control signal, opto-isolated output, low voltage level active
4	EXT_OUT11	Output control signal, opto-isolated output, low voltage level active
5	EXT_OUT12	Output control signal, opto-isolated output, low voltage level active
6	EXT_OUT13	Output control signal, opto-isolated output, low voltage level active

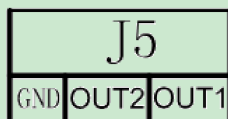
7	EXT_OUT14	Output control signal, opto-isolated output, low voltage level active
8	EXT_OUT15	Output control signal, opto-isolated output, low voltage level active
9	EXT_OUT16	Output control signal, opto-isolated output, low voltage level active
10	EXT_OUT17	Output control signal, opto-isolated output, low voltage level active
11	EXT_OUT18	Output control signal, opto-isolated output, low voltage level active
12	EXT_OUT19	Output control signal, opto-isolated output, low voltage level active
13	EXT_OUT20	Output control signal, opto-isolated output, low voltage level active
14	EXT_OUT21	Output control signal, opto-isolated output, low voltage level active
15	EXT_OUT22	Output control signal, opto-isolated output, low voltage level active
16	EXT_OUT23	Output control signal, opto-isolated output, low voltage level active
17	EXT_OUT24	Output control signal, opto-isolated output, low voltage level active
18	EXT_OUT25	Output control signal, opto-isolated output, low voltage level active
19	EXT_OUT26	Output control signal, opto-isolated output, low voltage level active
20	EXT_OUT27	Output control signal, opto-isolated output, low voltage level active
21	EXT_OUT28	Output control signal, opto-isolated output, low voltage level active

22	EXT_OUT29	Output control signal, opto-isolated output, low voltage level active
23	EXT_OUT30	Output control signal, opto-isolated output, low voltage level active
24	EXT_OUT31	Output control signal, opto-isolated output, low voltage level active
25	EXT_+24V GND	24V power grounding
26	EXT_+24V VCC	+24V power output (can't be connected to external 24V+)

2.5 J4 signal definition, reserved, no specific function



2.6 J5 signal definition



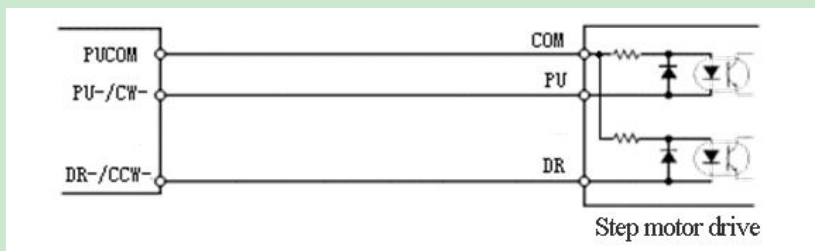
Wire No.	Name	Function
1	OUT1	DA1 output, 0~10V output
2	OUT2	DA2 output, 0~10V output
3	GND	Reference grounding

2.7 Connecting pulse/direction output signal:

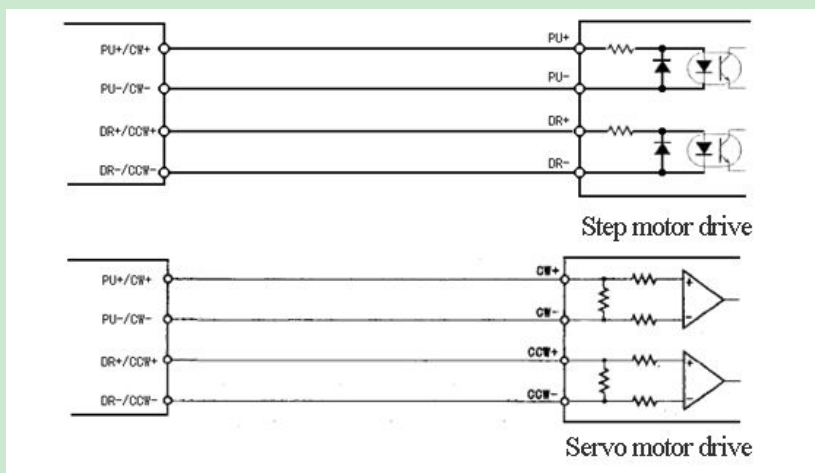
Pulse output is differential output

Can be easily connected to the stepper / servo drives

Below is the connection that pulse anode and direction anode have been connected.

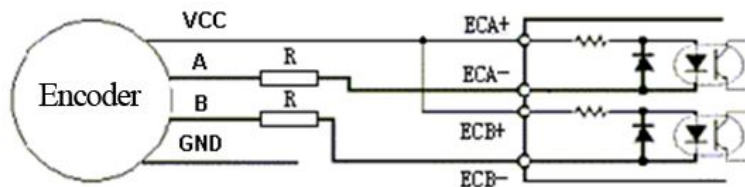


Below is independent connection of pulse and direction signal. Differential connection is recommended due to strong anti-interference.



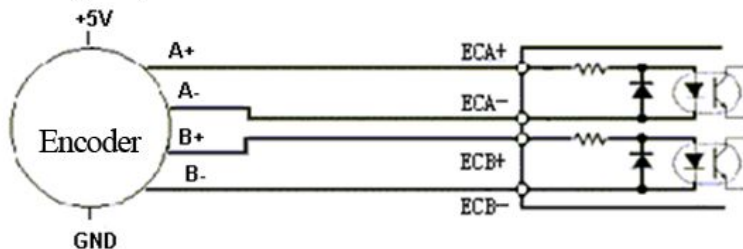
Note: Refer to Chapter 5 for the wiring diagram of common servo motor drive and terminal board.

2.8 Connecting encoder input signal:

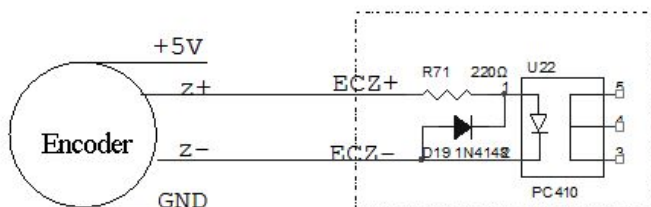


Open Collect Output Encoder Wiring Diagram

For +5V power, R can be omitted; for +12V power, R=1k Ω ; for +24V power, R=2k Ω



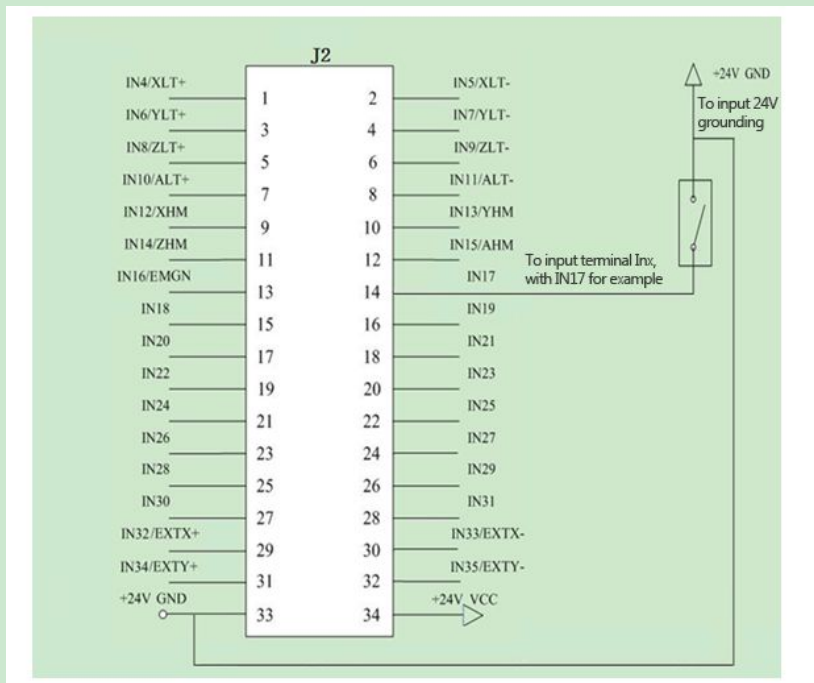
Line Driver Output Encoder Wiring Diagram



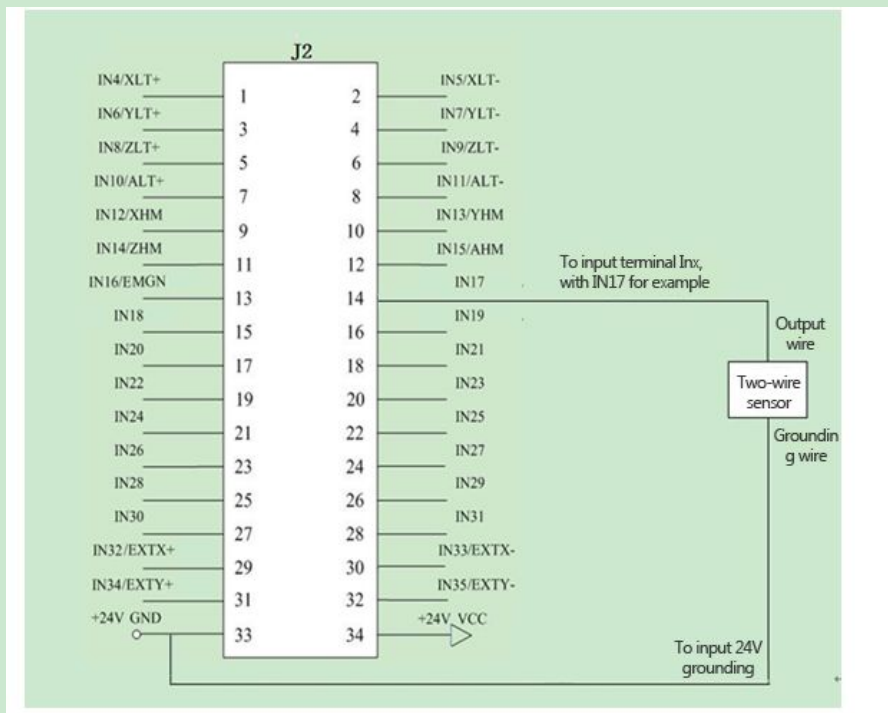
Encoder Z phase connection

2.9 Connecting digital input:

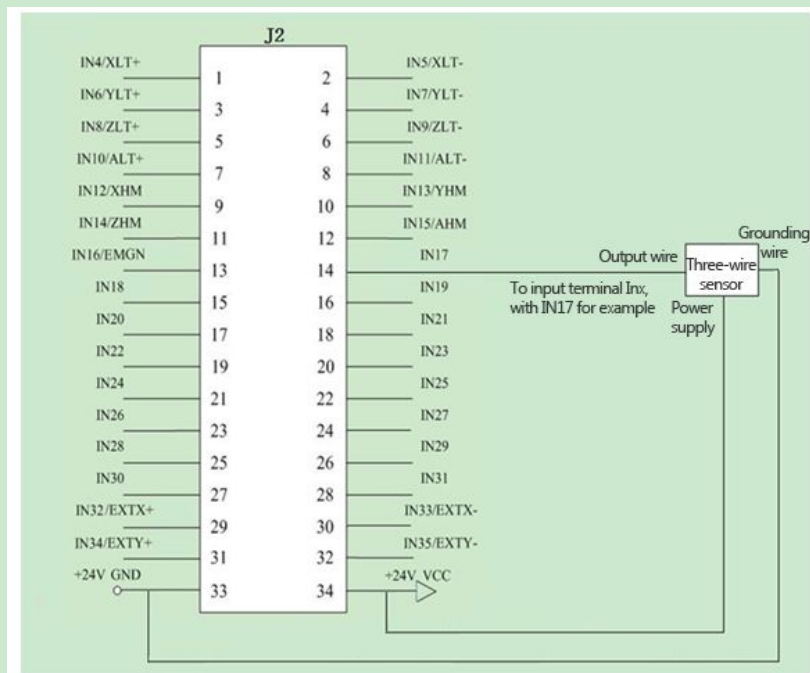
The following figure shows the mechanical switch connection with IN17 for example (regardless of switch polarity):



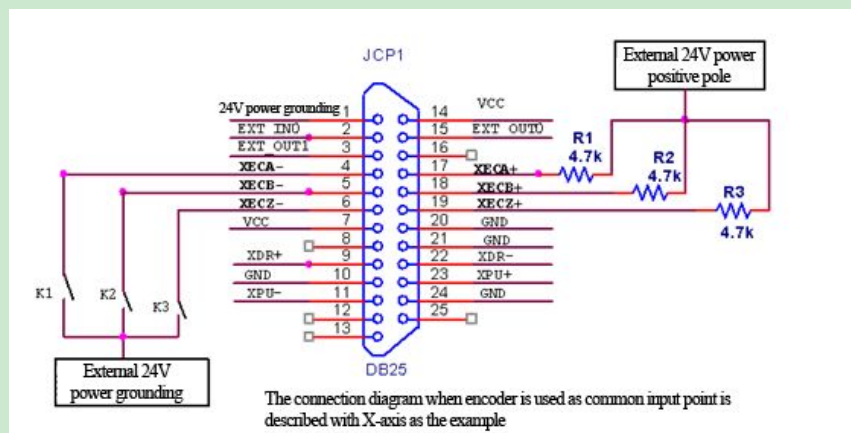
The following figure shows the two-wire sensor connection (note the polarity of two-wire sensor; blue is ground wire, and brown is output wire). **Note: The input points IN0~IN3 of four-axis 25-pin DB connector do not support two-wire sensor connection.**



Below is three-wire sensor connection (pay attention to the polarity of three-wire sensor; refer to corresponding manual):



The connection diagram of encoder AB phase signal as normal input point (K1, K2, K3 in mechanical switch connection):



Note: After ADT-9192 is connected to the power, the power of wiring terminal J2 is synchronized to 24V, XECA+, XECB+ and XECC+ are 4.7K resistance respectively, and then connect to the positive pole of the 24V power supply of J2 terminal, XECA-, XECB- and XECC- correspond the corresponding input point. Other axes are similar.

The relationship between encoder as common input point and sample program input point

XECA- corresponds to IN36, XECB- corresponds to IN40, and XECZ- corresponds to IN44

YECA- corresponds to IN37, XECB- corresponds to IN41, and XECZ- corresponds to IN45

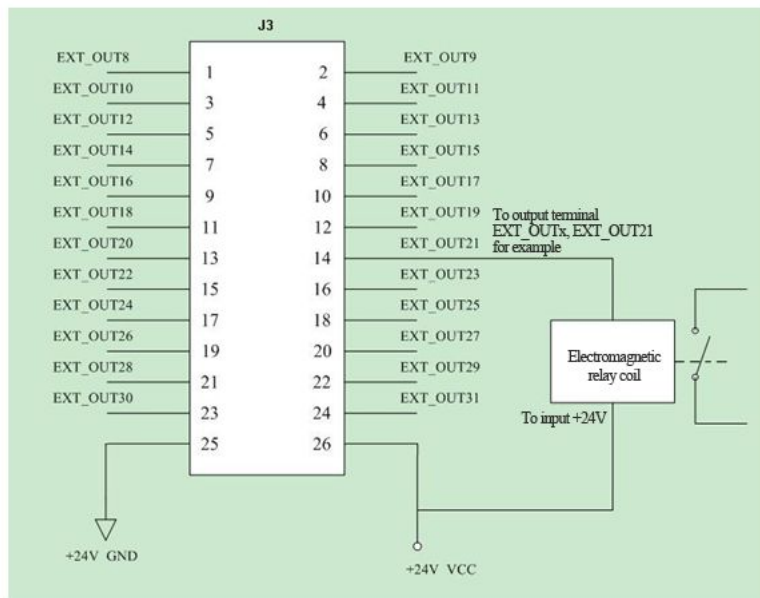
ZECA- corresponds to IN38, XECB- corresponds to IN42, and XECZ- corresponds to IN46

AECA- corresponds to IN39, XECB- corresponds to IN43, and XECZ- corresponds to IN47

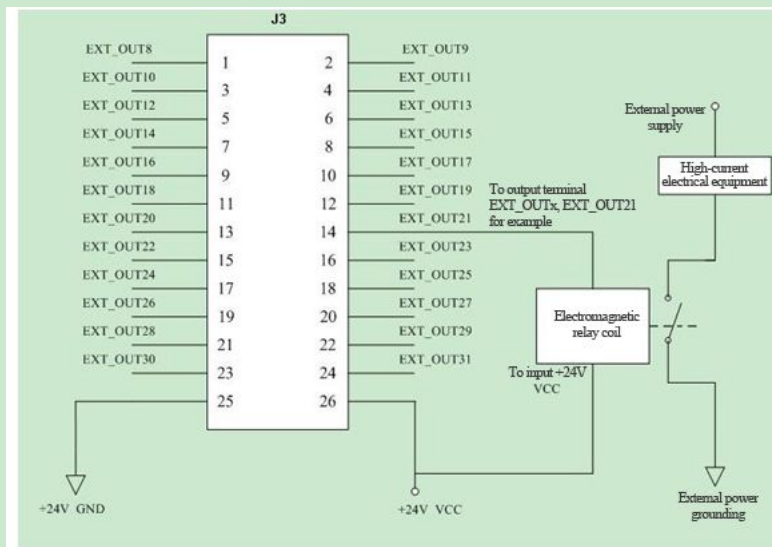
2.10 Connecting digital output:

All common output points of the board are open drain output, and the drive current at each point is within 1A. Before use, consider if the drive current of the output point is adequate; if not, expand with external relay and connect freewheeling protection diode.

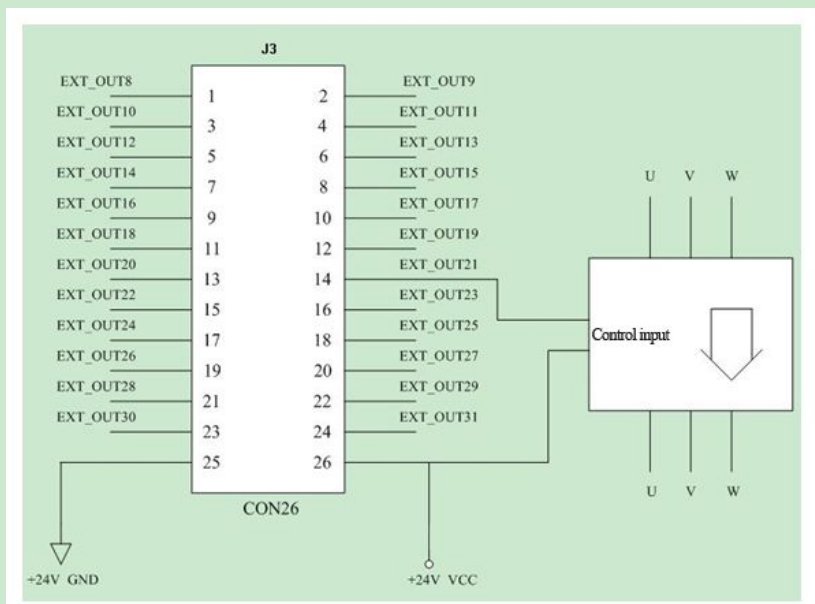
Below is the connection of output point driven general electromagnetic relay (regardless of relay coil polarity unless otherwise specified):



The connection of electromagnetic relay expansion flow output point (only normally open relay can be used for expansion flow)



Solid state relay connection (pay attention to the polarity of solid state relay control terminal):



The figure shows the connection of three-phase solid state relay. Two-phase and single-phase solid state relays have the similar connection.

Chapter 3 Software Installation

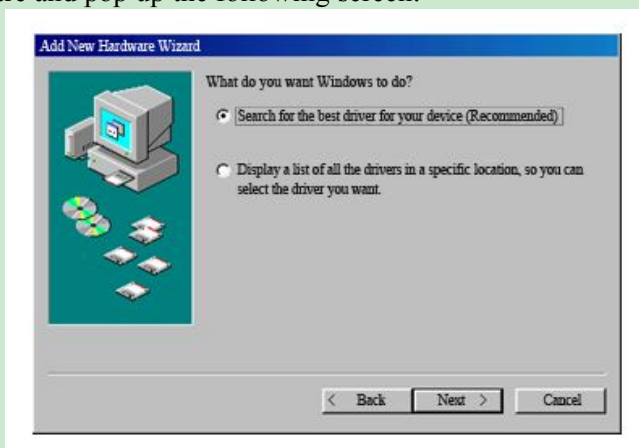
To use ADT-8949 card properly on Win95 / Win98 / NT / Win2000 / WinXP, you need to install the drivers. No drivers are needed in DOS. Below is the example on Win2000 and WinXP. Refer to this section for other systems.

The drivers of the control card are located on the CD "Development kits \ Drivers \ Control Card Drivers", and the file name is 8949.INF.

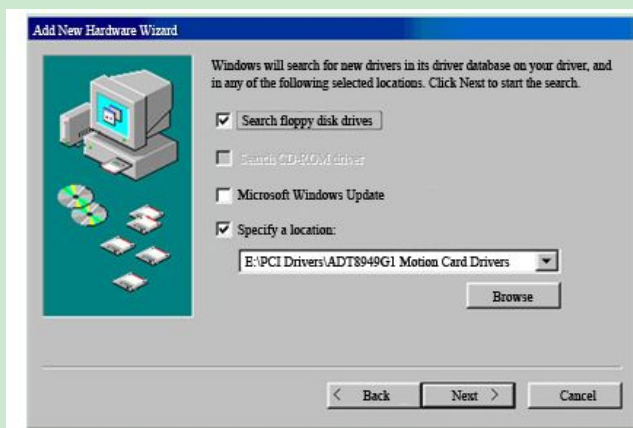
3.1 Installing drivers on Win98:

Below is an example of driver installation on Win98 Professional Chinese version. Other versions are similar to Win98.

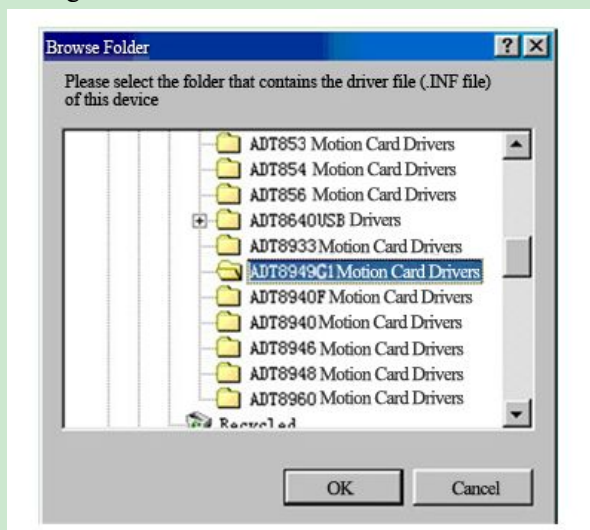
After installing the ADT-8949 card into a PCI slot on the computer, log in as administrator, and the computer should find the new hardware and pop up the following screen:



Click “Next” to pop up the following screen



Click the "Browse" button, select CD "Development Kits \ Drivers \ Motion Card Driver" and find the file 8949.inf, and click "OK" to pop up the following interface.



Click "OK" to pop up the following screen:

Click "Next" to pop up the following screen



Click “Next” to pop up the following screen



Click “Finish” to finish the installation of ADT-8949 card

3.2 Installing drivers on WinXP:

Below is the example of installing drivers on WinXP. Other systems are similar.



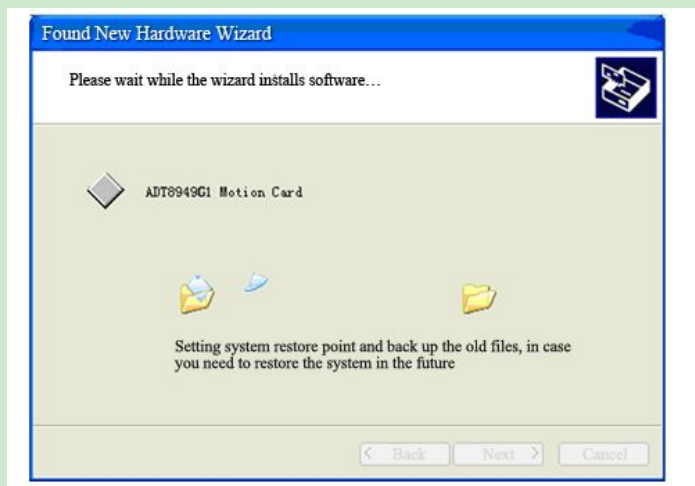
Select in above figure to pop up the following screen



Select in above figure and click Next to pop up the following screen



Click “Browse”, select CD “Development Kits\Drivers\Motion Card Drivers” to locate the file 8949.INF, and click Next to pop up the following screen

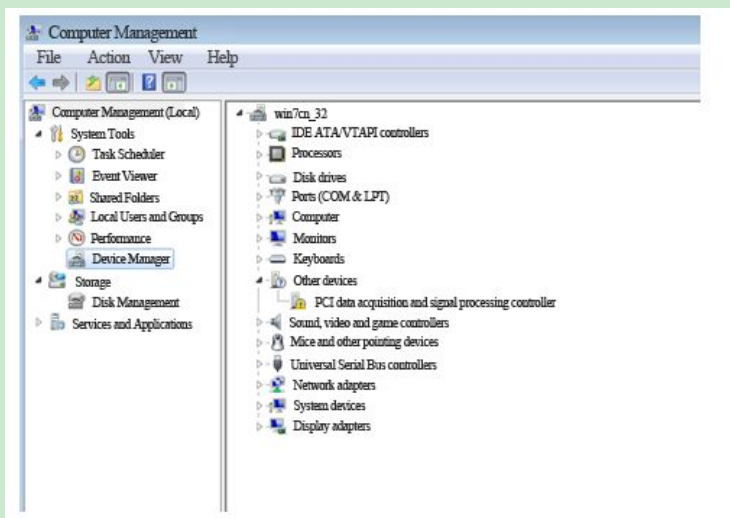


Click “Finish” to finish the installation of ADT-8949 card

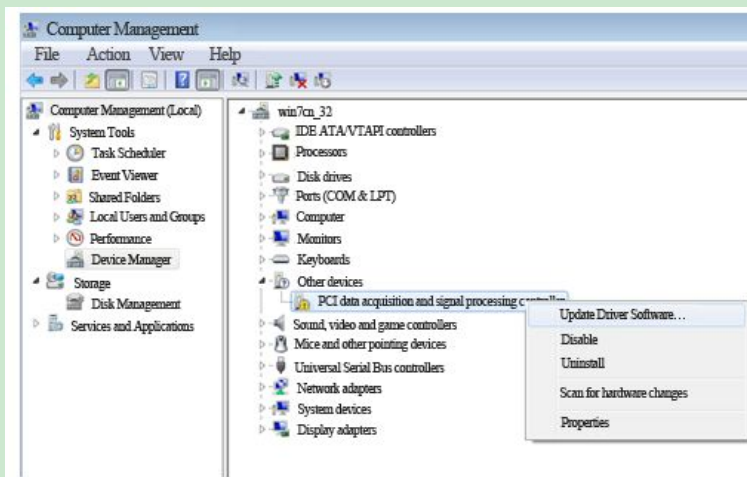
3.3 Installing drivers on Win7

Install the drivers on Win7 system as follows:

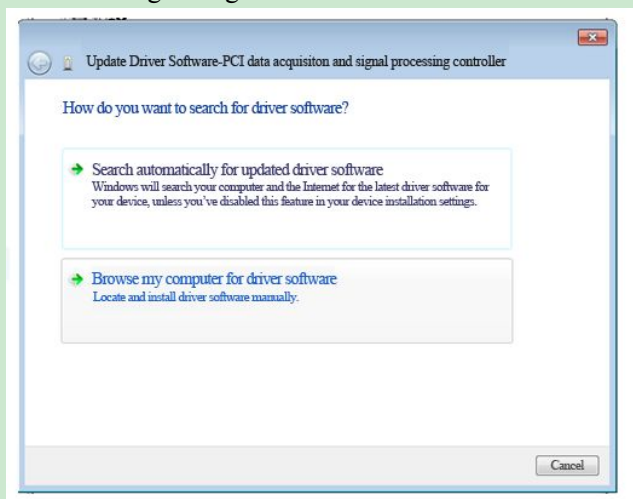
1. Insert the control card into PCI slot, right click "My Computer" and select "Properties" to enter Device Manager, as shown below:



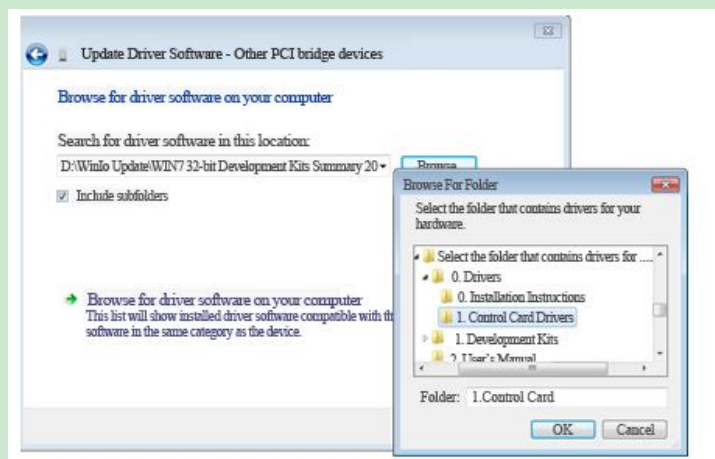
Expand "Other devices", select "PCI data acquisition and signal processing controller", and right click, as shown below:



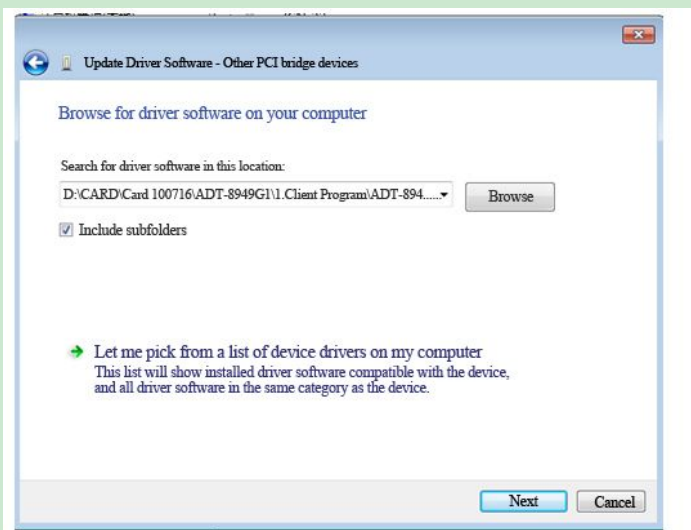
2. In the popup dialog box, click "Update Driver Software" to show the following dialog box:



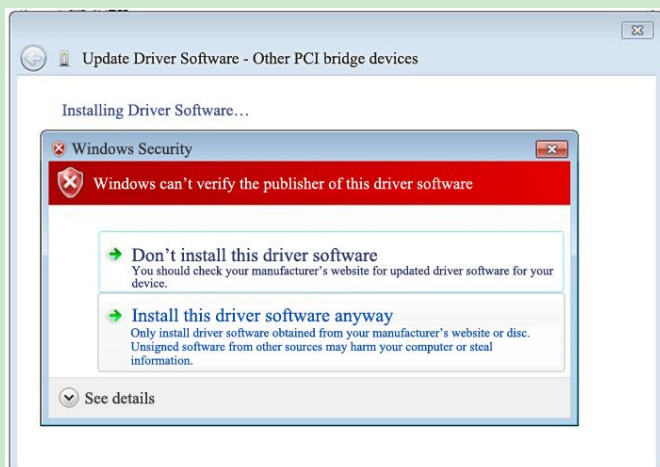
- Select the option "Browse my computer for driver software", and then click the "Browse" button to specify the path for the driver, as shown below:



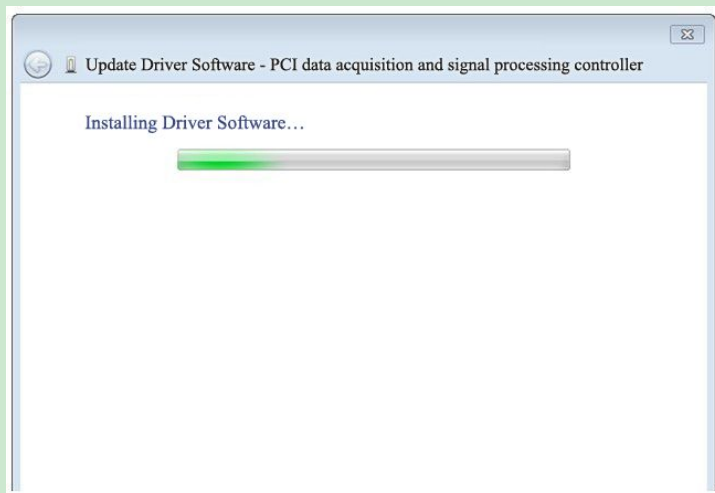
3. Click "OK", and the following dialog box appears:



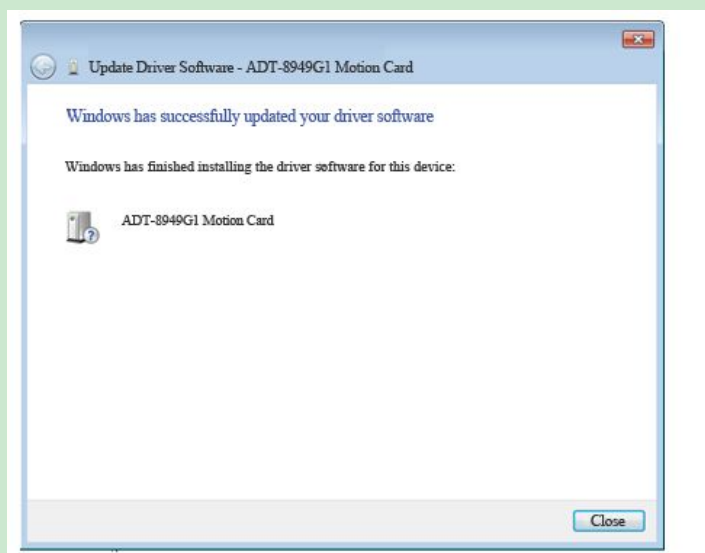
4. Click "Next" to install the drivers, and the following interface appears:



5. Select "Install this driver software", and the following interface appears

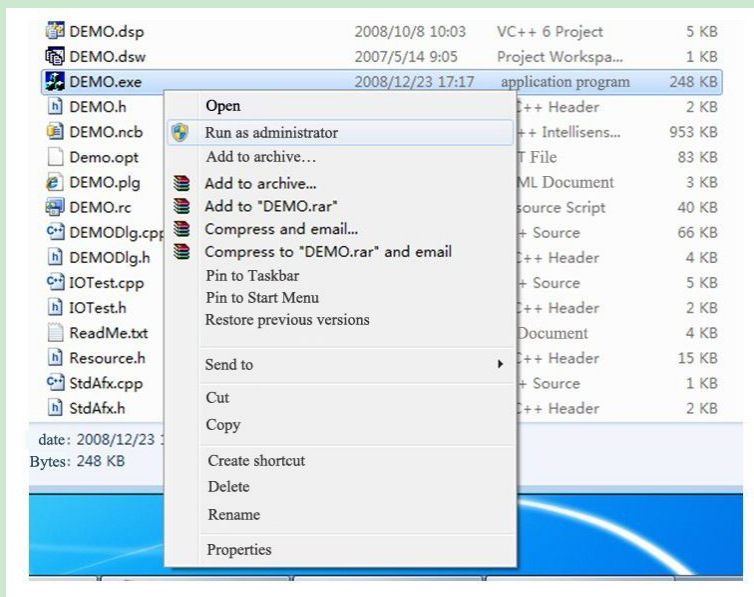


Wait to complete, and the following dialog box appears



The ADT-8949 card is installed. **Note: WIN7 system requires administrator privileges to load the PCI drivers. If the control card application is run for the first time, double-clicking will**

lead to the control card initialization failed. Therefore, when the drivers are installed, right click the control card application (e.g. VC demonstration program "DEMO.EXE") and select "Run as Administrator" (see below). You can double-click the application to run it normally later.



Chapter 4 Electrical Specifications

4.1 Switch input:

Channel: 36 channels, all optically isolated.

Input voltage: 12-24V

High voltage level > 4.5V

Low voltage level <1.0V

Isolation voltage: 2500V DC

4.2 Count input:

Channel: 4 channels AB phase encoder input, all optically isolated.

Maximum count frequency: 4MHz

Input voltage: 5-24V

High voltage level >4.5V

Low voltage level <1.0V

Isolation voltage: 2500V DC

4.3 Pulse output:

Channel: 4 pulses, four directions, all optically isolated.

Maximum pulse frequency: 5MHz

Output type: 5V differential output

Output: Pulse + direction or pulse + pulse

4.4 Switching output:

Output channels: 32 channels, all optically isolated.

Output type: NPN open collector 5-24VDC, maximum current of single output of common output port: 1A; maximum current of single output of DB terminal: 50mA.

4.5 Power output:

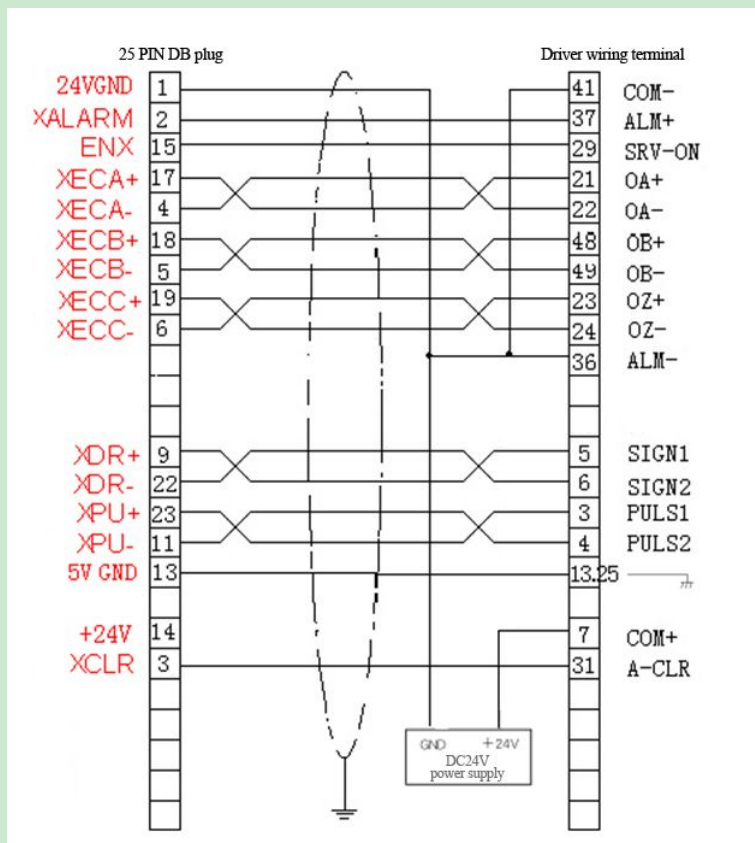
Output voltage: + 5V.

Output type: DC source, maximum current 500mA.

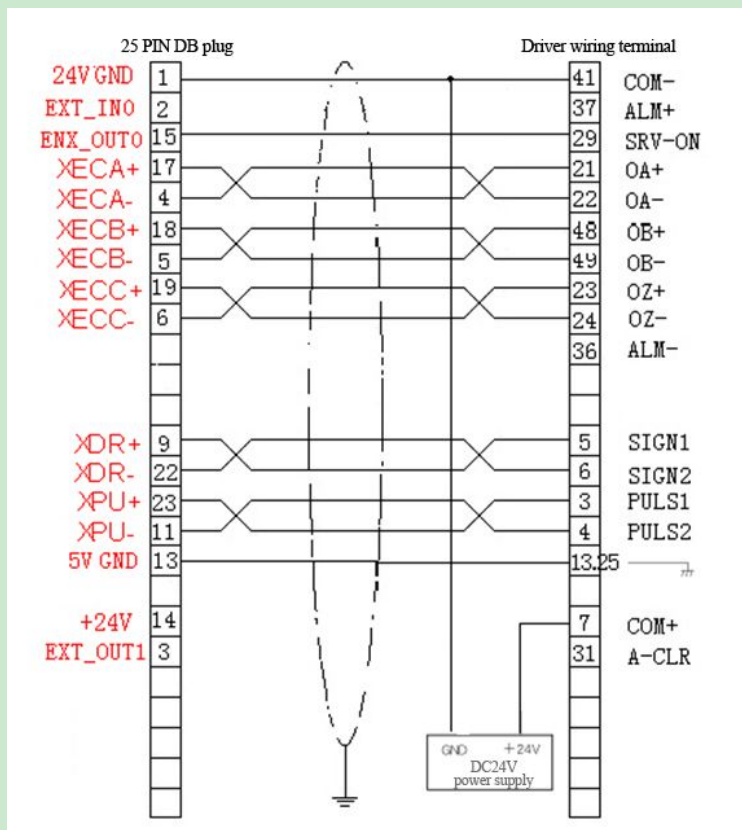
Chapter 5 Common Servo Wiring Diagrams

5.1 Panasonic A5 servo wiring diagram:

Below is the wiring of motion card connected to Panasonic A5 servo drive: use external power supply, external enable, including alarm signal.



Below is the wiring of motion card connected to Panasonic A5 servo drive: use external power supply, external enable, alarm information not specified.



Chapter 6 Working Environment

6.1 Operating temperature:

Operating temperature: 0℃~60℃

6.2 Storage temperature:

Storage temperature: -20℃~80℃

6.3 Operating humidity:

Operating humidity: 20%~95%

6.4 Storage humidity:

Storage humidity: 0%~95%





Software Programming

Chapter 1 Function Description

1.1 Pulse output mode:

Driving output pulse has the following two pulse output modes below. When independent two pulses mode is used in positive driving, PU/CW outputs driving pulse; in negative driving, DR/CCW outputs driving pulse; when one pulse mode is used, PU/CW outputs driving pulse and DR/CCW outputs direction signal.

Both pulse and direction are positive logic setting

Pulse output mode	Driving direction	Output signal wave	
		PU/CW signal	DR/CCW signal
Independent two pulses mode	+ direction driving output		Low level
	- direction driving output	Low level	
One pulse mode	+ direction driving output		Low level
	- direction driving output		High level

1.2 Hardware limit signal:

Hardware limit signal (LMT+, LMT-) is the input signal that limits positive and negative driving pulse. It can be set to valid, invalid, high level or low level, and the positive and negative limits can be set to valid / invalid independently. When set to invalid, it can be used as normal input point.

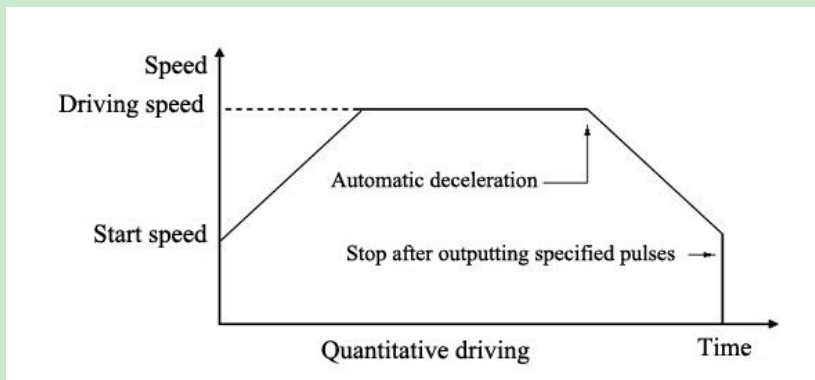
Hardware limit signal (STOP0, STOP1) can achieve input signal to stop driving of each axis. It can be set to valid, invalid, high level or low level. When set to invalid, it can be used as normal input point.

1.3 Quantitative driving:

Quantitative driving is to output specified quantity of pulse in constant speed or acceleration/deceleration. Use this function to move to specified position or perform specific action. Quantitative driving of acceleration/deceleration is shown below. When the remaining of output pulse is less than acceleration accumulated pulses, it starts accelerating, and the driving also stops after outputting specified pulses.

The following parameters should be set for the quantitative driving for acceleration/ deceleration:

- a) Acceleration/deceleration A/D
- b) Start speed SV
- c) Driving speed V
- d) Output pulses P



Acceleration/deceleration quantitative driving usually starts automatic acceleration from the calculated deceleration point shown in the picture above; besides, manual deceleration is also possible. In the following cases, it is not possible to or can't calculate the automatic deceleration point, and thus manual calculation is required:

- Change speed frequently during linear acceleration/ deceleration quantitative driving

1.4 Continuous driving:

During continuous driving, output driving pulse continuously until high level stop command or external stop signal is valid. Use this function for home search, scanning operation and motor rotation control.

Two stop commands are available, one is deceleration stop and the other is immediate stop. Each axis has two external signals (STOPO, and STOP1) used for deceleration/ immediate stop. Each signal can set valid/invalid level. STOPO and STOP1 signals are deceleration stop in

acceleration/deceleration driving and immediate stop in constant speed driving.

Home search action of continuous driving.

Arrange the home signal and encoder Z-phase signal to STOP0 and STOP1. Set the valid/invalid and logical level of every signal in every axis. During high speed search, use acceleration/deceleration continuous driving, and decelerate to stop when the set valid signal is in activated level. During low speed search, use constant speed continuous driving, and immediately stop when the set valid signal is in activated level. For acceleration/deceleration continuous driving, all the parameters except output pulses must be same as quantitative driving.

1.5 Speed curve:

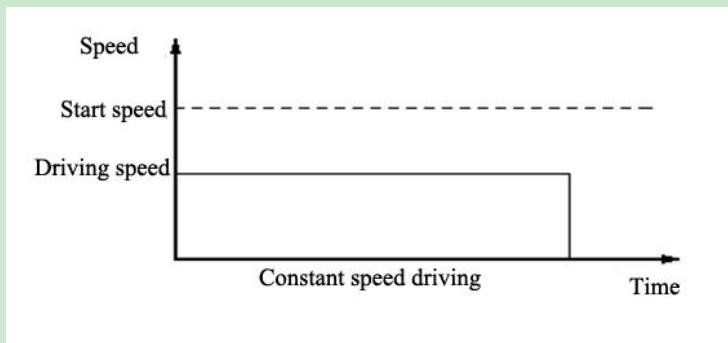
You can select multiple acceleration and deceleration modes for speed setting: T-shaped, S-shaped, E-shaped, C-shaped acceleration and deceleration, and supports asymmetric acceleration, featuring smooth running and quiet motor.

1.5.1 Constant speed driving

Constant speed driving will output driving pulse in constant speed. If the driving speed is lower than the start speed, there will be constant speed driving only instead of acceleration/deceleration driving. When use home search, encoder Z phase and similar signals, acceleration/deceleration driving isn't required if stop immediately after signal is searched; instead, the system runs low speed constant driving.

The following parameters should be set for constant speed driving:

- Start speed SV
- Driving speed V



1.5.2 T-shaped linear acceleration/deceleration driving

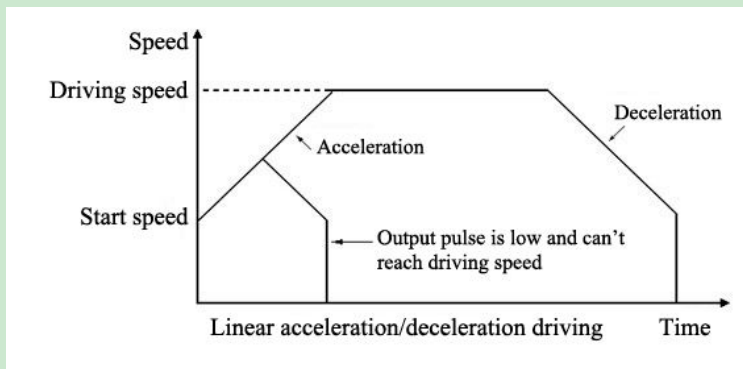
Linear acceleration/deceleration is to accelerate from the start speed to specified driving speed linearly.

1) Symmetric linear acceleration/deceleration drive

During quantitative driving, the acceleration counter records the accumulated pulses. When the remaining output pulses are less than acceleration pulse, it starts decelerating (automatic deceleration), and decelerates to start speed linearly in specified deceleration.

The following parameters should be set for linear acceleration/deceleration driving:

- Acceleration A (if only A is set, it is symmetric acceleration/deceleration by default, that is, A and D are equal)
- Start speed SV
- Driving speed V



2) Asymmetric linear acceleration / deceleration drive

When moving object in the vertical direction, the object has the burden of gravitational acceleration, so the up and down acceleration and deceleration should be changed for the quantitative driving of asymmetric linear acceleration/deceleration with different acceleration and deceleration. At this moment, automatic deceleration is allowed, and the manual deceleration point doesn't need to be set in advance. Fig. 1 is an example that the acceleration is smaller than the deceleration, and Fig. 2 is an example that the deceleration is smaller than the acceleration.

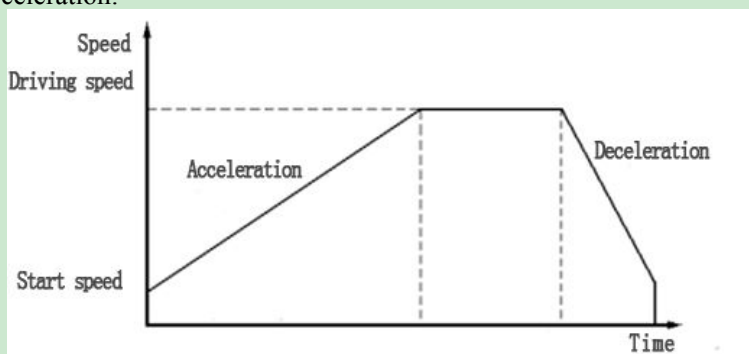


Fig. 1 Asymmetric Linear Acceleration / Deceleration Drive (Acceleration < Deceleration)

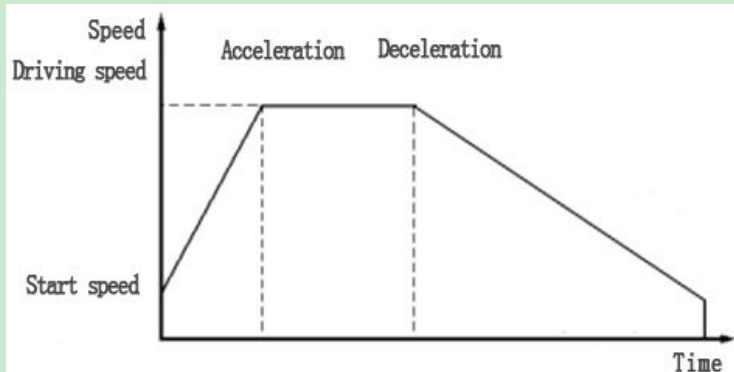


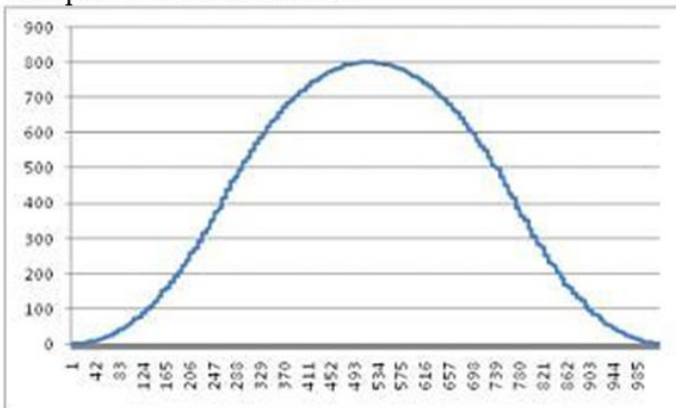
Fig. 2 Asymmetric Linear Acceleration / Deceleration Drive (Acceleration > Deceleration)

In addition, same as the usual linear acceleration and deceleration drive, the following parameters should be set:

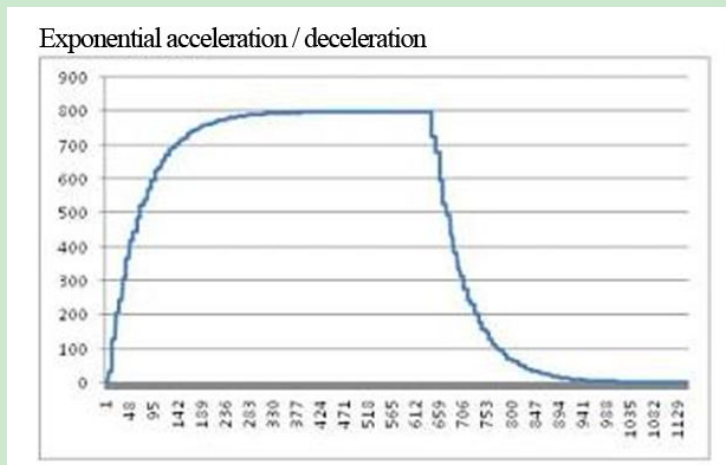
- Acceleration A
- Deceleration D
- Start velocity SV
- Diving velocity V

1.5.3 S-shaped curve acceleration/deceleration driving

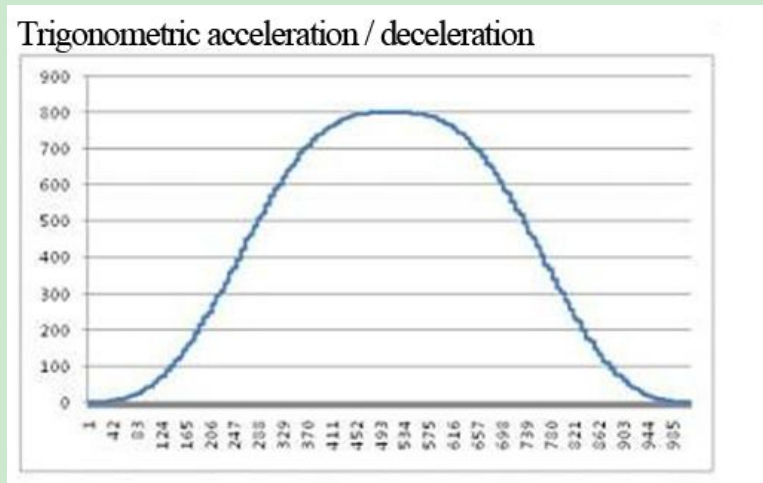
S-shaped acceleration/deceleration



1.5.4 Exponential acceleration / deceleration driving



1.5.5 Trigonometric acceleration / deceleration driving



1.6 Position latch:

Use IN signal of each axis to achieve hardware position latch function. Use one latch signal to lock current position of all axes, and the locked position can be logic location or actual position. The latch signals corresponding to axis 1~4 are IN12, IN13, IN14 and IN15 respectively. Position latch function has important applications in the measurement system.

1.7 External signal driving:

External signal driving is the movement controlled by external signal (hand wheel or switch), and is mainly used for manual debugging, particularly convenient in teaching system.

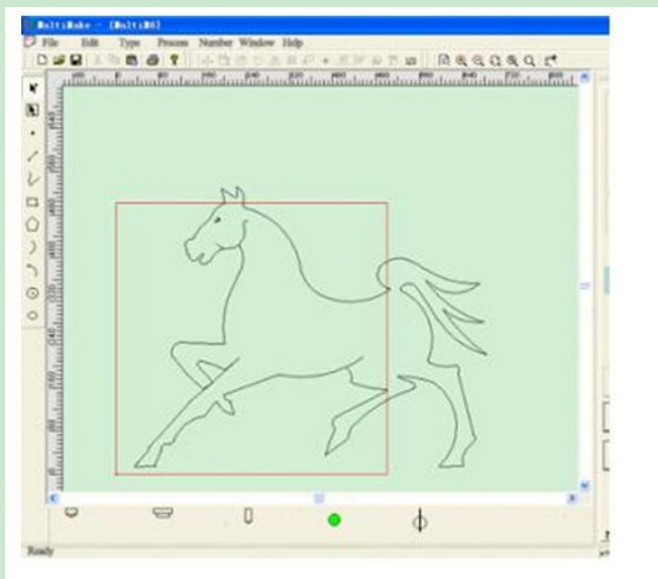
1.8 Large cache small segment:

Large cache small segment: large capacity multi-axis cache interpolation, store 10K interpolation instructions; small segments and large cache are used for engraving or cutting applications to make discrete CAM data can be restored to processing model .

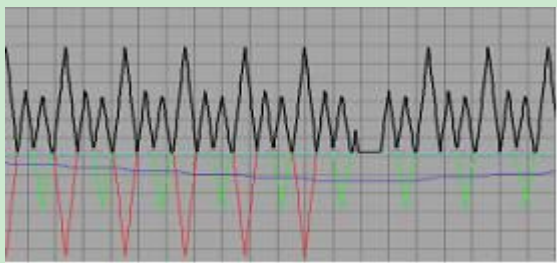
1.9 Speed adaptive model:

Speed adaptive model: to ensure precision under high speed, automatic speed optimization; used for milling machine, tooling and other applications requiring high precision control, makes the motor work in reasonable error range from speed planning.

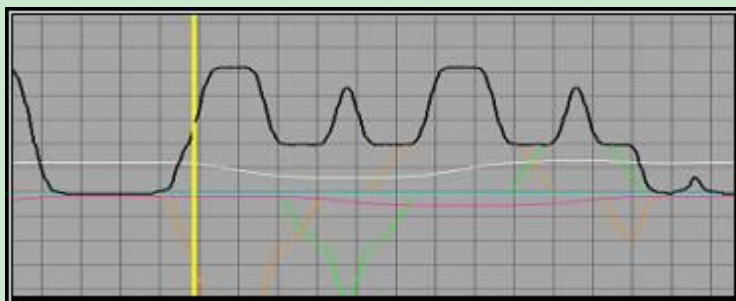
For example, to run the following track, monitor the speed curve:



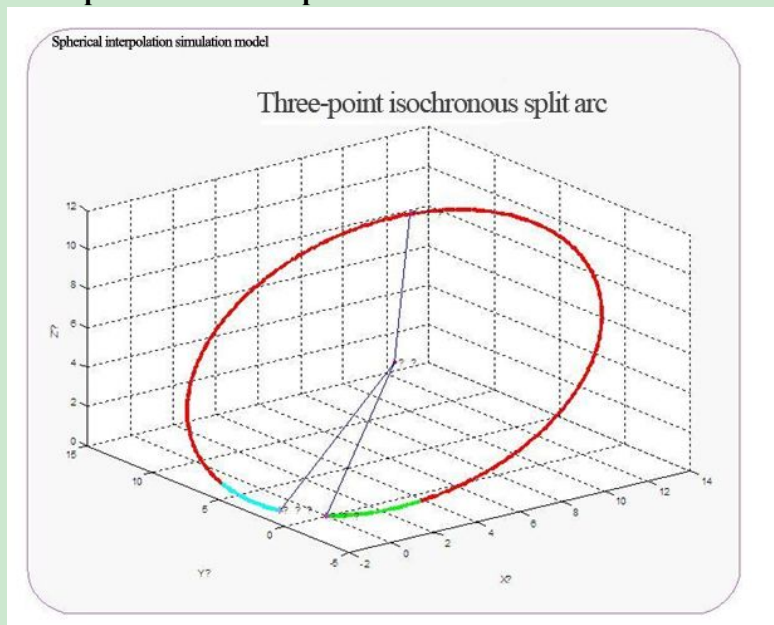
General speed curve:



Speed adaptive model curve:



1.10 Spherical arc interpolation:



3D arc interpolation (spherical interpolation): achieve arc in any spatial plane and spherical arc, suitable for teaching operation of simplified complex graphics. Comprehensive 3D arc interpolation, hardware-level support, support cache interpolation, only occupies four data segments, arc approximation accuracy is determined by interpolation speed dynamically to avoid discrete error of small segment approximation and contradictions of difficult speed trade-off. Helical interpolation and plane arc interpolation can be easily achieved based on spherical interpolation technology.

1.11 NURBS interpolation:

General cards provide only linear and arc interpolation. For non-linear and arc curve, linear and arc piecewise fitting method is used for interpolation. This method may cause large data size, poor accuracy,

uneven feed rate, complicated programming and other issues in processing complex curves, which will inevitably lead to a greater impact on the processing quality and costs. NURBS (spline) interpolation is a method capable of direct interpolation of complex free-shaped curve and surface.

1.12 Simultaneous control of multiple processes:

Simultaneous control of multiple processes: You can open two programs to control one card. (A single monitoring program run simultaneously with the execution program; the execution program doesn't need to switch a lot of time for display, which makes the display more real-time.)

1.13 Gantry dual-drive, changing drive speed and target position in motion:

Gantry dual-drive, changing drive speed and target position in motion in real time.

1.14 Specify the time for four-axis linear interpolation:

Specify the time for four-axis linear interpolation, facilitate speed planning customization.

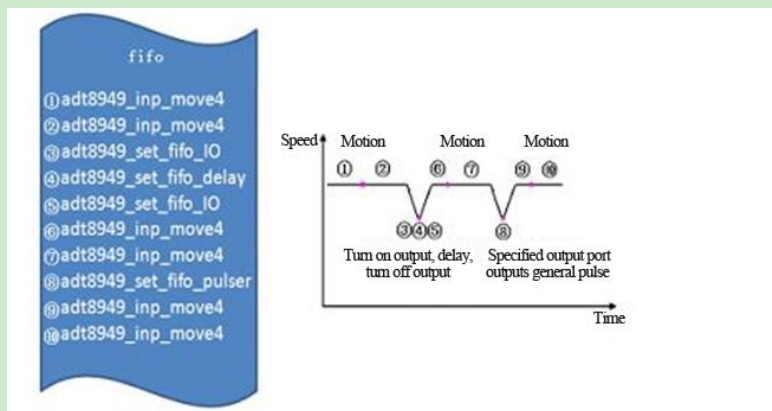
1.15 Pulse generator:

Up to 15-channel pulse generator function can be inserted for interpolation. You can specify the number of turns of output level, and retention time of high and low level in millisecond. It can be used in dispensing, cutting, assembly line and other industries.

For example, in dispensing industry, the capacity of glue gun is fixed and it is required to feed the glue gun from time to time during motion. For conventional practice, the user calculates the glue amount and feed the glue gun by operating the output point on host computer. This

approach is inaccurate, the operation is not flexible and result in untimely feeding easily.

The pulse generator function only needs to specify the glue position of the glue gun track and number of operations of glue feeding piston to achieve the glue feeding action easily. When the machine tool moves to the position specified by the user, the motion card will automatically activate the pulse generator to drive the piston, which can ensure easier operation and more accurate control.

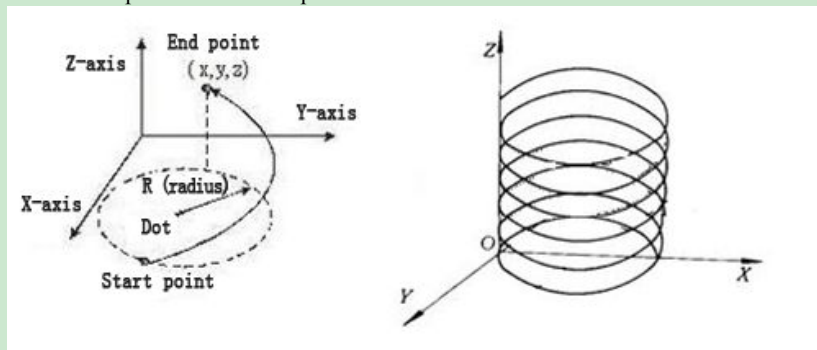


1.16 Get arc length, set 15 filter levels for input point:

Get the arc length, and set 15 filter levels for input point.

1.17 Helical interpolation:

Helical interpolation based on plane arc.



Chapter 2 Motion Control Library Function Guide

2.1 Introduction of ADT-8949 function library

ADT-8949 function library is the interface for users operating motion control card. The users can control the motion card to complete corresponding function by transferring interface functions.

The motion control card provides the motion function library in DOS and DLL in Windows. The transferring methods of function library in Windows are introduced as below.

2.2 Calling DLL on Windows

The DLL “adt8949.dll” in Windows is written with VC. It is in “Development Kit \Drivers\DLL” in the CD, and is suitable for the programming language tools in Window: VC++, VB6.0, .NET, C#, C++Builder, Delphi and configuration software LabVIEW, etc.

2.3 Calling in VC

- (1) Create a new project;
- (2) Copy the files “adt8949.lib” and “adt8949.h” from “Development Kits \VC” in the CD to the path of the new project;
- (3) In the “File View” of the “Work Area” of the new project, right click the mouse and select “Add Files to Project”, select “Library Files(.lib)” in the Add Files dialog box, search and select “adt8949.lib” and click “OK” to load the static library;
- (4) Add #include “adt8949.h” to the statement of source file or header file or global header file “StdAfx.h”;

After above four steps, the user can call the functions in the DLL.

Note: The calling in VC.NET is similar as VC.

2.4 Calling in VB

- (1) Create a new project;

- (2) Copy the file “adt8949lib.bas” from “Development Kits \VB” in the CD to the path of the new project;
- (3) Select “Project\Add module” menu, and select the “Save” tab in the dialog box, search the “adt8949lib.bas” module file, and click the Open button;

After above three steps, the user can call the functions in the DLL.

Note: The calling in VB.NET and C# is similar as VB.

2.5 Calling in C++Builder

- (1) Create a new project;
- (2) Copy the files “adt8949.lib” and “adt8949.h” from “Development Kits \ C++Builder” in the CD to the path of the new project;
- (3) Select the “Project\Add to Project” menu, select “Library Files(.lib)” in the dialog box, search and select “adt8949.lib” and click the “OK” button;
- (4) Add #include “adt8949.h” to the statement of the program file;

After above four steps, the user can call the functions in the DLL.

2.6 Return value and meaning of library function

To ensure that the user can control the execution when using library function, every function in the library will return the result after execution. The user can check whether the function transfer is successfully according to the return value.

Except “int adt8949_initial(void)” and “int read_bit(int cardno, int number)” in the function library, other functions only return “0” and “Non-0”, where “0” indicates successful transfer and “Non-0” indicates failed.

The meanings of the return values are described in the table below.

Function name	Return value	Meaning
adt8949_initial	-1	Service is not installed
	-2	PCI bridge fault
	-3	DSP program download error
	-4	Hardware exception or DLL version does not match
	-5	Failed to create mutex
	-6	Failed to open mutex
	-7	Other causes
	-8	DIP switch settings repeats, DIP switch pointing should be manually adjusted
	0	Control card is not installed
	>0	Quantity of control cards
adt8949_read_bit	0	Low level
	1	High level
	-1	Card number or input point overrun error
All other functions	0	Correct
	Non-0	Wrong

Note: Non-zero return value (usually -1) represents a function call error, which is usually caused by errors in transferred parameter cardno (card number) or axis (axis number) when calling library functions. The card number is the value of the DIP switch. When using multiple 8949, please turn the DIP switch to different values.

Chapter 3 Key Points for Motion Control Development

There will be some problems in the programming of the card. In fact, most of the problems are caused by misunderstanding the principal of the control card. Below is the description of some familiar instances that are easily understood.

3.1 Card initialization

3.1.1 Description

At the beginning of the program, call function `adt8949_initial()` first, check whether ADT8949 card is installed properly, and then set the pulse output mode and limit switch work mode. Above parameters should be set according to specific machine, and set only once when the program is initialized.

Note: library function “`adt8949_initial`” is the “door” to ADT8949 card.

Calling other functions has meaning only after transferring this function and initializing the motion control card successfully.

3.1.2 Pulse equivalent setting

The displacement of moving part driven by a single pulse signal is called the pulse equivalent, also known as the minimum unit. For example: if the pulse equivalent is 0.0001mm, it corresponds to 10,000 pulses run 1mm, so `adt8949_set_gear` is set to the reciprocal of pulse equivalent, i.e. the number of pulses per millimeter, $1/0.0001 = 10,000$. Prior to linkage and interpolation motion instructions, the axis number parameter of `adt8949_set_gear` starts from 1, 2, 3, and 4. Please note that the value corresponding to the number of pulses per millimeter can be floating point numbers.

3.2 Speed setting

3.2.1 Constant speed motion

The parameter setting is simple. It is only required to set the driving speed to be lower than or equal to the start speed, and other parameters do not need setting.

Related functions:

set_startv

set_speed

3.2.2 Trapezoidal acceleration/deceleration

This is a most commonly used mode. It requires setting start speed, driving speed and acceleration, and uses automatic deceleration.

Related functions:

set_startv

set_speed

set_acc

3.2.4 STOP0, STOP1 signal

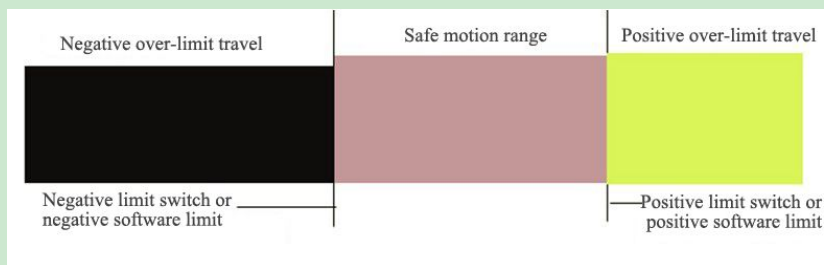
STOP0 and STOP1 (encoder Z-phase signal) are the signals of each axis, so that there are eight STOP signals in total. The signals are mainly used for machine homing, and the homing mode may use one or several signals depending on the circumstances. Please note the stop mode of STOP signals. If the motion speed is constant, stop immediately; for acceleration/deceleration motion, decelerate and stop. For high-speed homing, add a deceleration switch in front of the home switch, that is, use two STOP signals, one as the home switch and the other as the deceleration switch. It may use only one signal, decelerate and stop when encounter STOP signal, move reversely at constant speed, and stop when encounter STOP signal again.

Chapter 4 System Security Mechanism

4.1 Monitoring error message:

Using `get_stopdata ()` function to get the error message can get the stop information of the axis, including the stop caused by hardware limit, stop caused by origin signal, normal stop, and other stops.

4.2 Limit:



The motion control card can use limit switch or software limit to control motion range of axes. If negative limit switch or negative software limit is triggered, it only moves towards positive direction; if positive limit switch or positive software limit is triggered, it only moves towards negative direction; note that the software limit is available only after successful homing.

➤ Use of software limit

The positive and negative limit of software limit is a concept of absolute position rather than an incremental value: the positive limit travel and negative limit travel are relative to the origin of the axis. Therefore, in a specific project, be sure to enable the software limit only when each axis has been reset (successful homing), and the mechanical origin at this time is the origin of the axis. The use of

software limit needs to call `adt8949_set_softlimit_mode`. See Chapter 14 for detailed explanation of functions.

Chapter 5 High Speed Capture of External Signal Homing

5.1 Homing motion:

5.1.1 List of required functions:

Set homing mode	<code>adt8949_SetHomeMode_Ex</code>
Set homing speed	<code>adt8949_SetHomeSpeed_Ex</code>
Start homing	<code>adt8949_HomeProcess_Ex</code>
Check status	<code>adt8949_GetHomeStatus_Ex</code>
Note	See Important Note

Below is an example of single axis homing; multi-axis is similar.

Use STOP0 as home signal

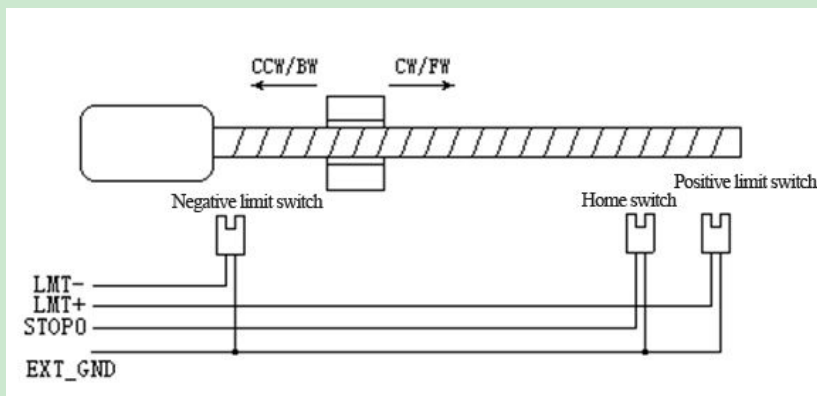
(1) Homing is divided into three steps:

Step 1: approach stop0 quickly (logical0 home setting), and find stop0;

Step 2: record the locking position, and move to the recorded position.

Step 3: approach stop1 slowly (logical1 encoder Z-phase).

(2) You can choose whether to perform in the third step through `logical1`.



5.1.2 Important Note

Note the parameters setting for homing. See detailed explanation of functions for the specific meaning of the parameters. The start speed of homing (STOP0) can't be greater than the homing speed. If it is not necessary to search STOP1 signal, set this parameter to -1. When necessary, follow the function instructions to set. Abovementioned is the example of X-axis. For multi-axis homing, just add function calling. Please note that the example doesn't search Z phase signal, i.e. STOP1 signal. When necessary, set and call it.

Chapter 6 Linkage Control

Linkage control includes single axis point motion and multi-axis point motion. The specific implementation in the project depends on the requirement.

All the routines in this chapter are independent of each other. In the process of implementing the project, card initialization and pulse equivalent configuration only need calling once, that is, no setting is required after card initialization and pulse equivalent setting. For the

settings of pulse equivalent, see Chapter 3 Section 3.1.2 Pulse Equivalent Settings.

6.1 Single axis quantitative uniform motion:

6.1.1 List of required functions

Set start speed	adt8949_set_startv
Set driving speed	adt8949_set_speed
Driving instruction	adt8949_pmove
Read driving status	adt8949_get_status
Note	See Important Note

6.1.2 Important Note

The control card initialization function involved in this routine only need to be set once before setting the pulse mode and pulse equivalent function.

6.2 Single axis quantitative symmetry trapezoidal acceleration / deceleration motion:

6.2.1 List of required functions:

Set to trapezoidal acceleration / deceleration	adt8949_set_admode
Set start speed	adt8949_set_startv
Set driving speed	adt8949_set_speed
Set acceleration	adt8949_set_acc
Driving instruction	adt8949_pmove
Read driving status	adt8949_get_status
Note	See Important Note

6.2.2 Important Note:

The control card initialization function involved in this routine only need to be set once before setting the pulse mode and pulse equivalent function.

6.3 Single axis quantitative asymmetry trapezoidal acceleration / deceleration motion:

6.3.1 List of required functions:

Set to trapezoidal acceleration / deceleration	adt8949_set_admode
Set start speed	adt8949_set_startv
Set driving speed	adt8949_set_speed
Set acceleration	adt8949_set_acc
Set deceleration	adt8949_set_dec
Driving instruction	adt8949_pmove
Read driving status	adt8949_get_status
Note	See Important Note

6.3.2 Important Note:

The control card initialization function involved in this routine only need to be set once before setting the pulse mode and pulse equivalent function.

6.4 Single axis quantitative S-curve acceleration / deceleration motion:

6.4.1 List of required functions:

Set to S-curve acceleration / deceleration	adt8949_set_admode
Set start speed	adt8949_set_startv
Set driving speed	adt8949_set_speed
Set acceleration	adt8949_set_acc
Set jerk	adt8949_set_jcc
Driving instruction	adt8949_pmove
Read driving status	adt8949_get_status
Note	See Important Note

Example 1: Full S-curve

Purpose:

Let X-axis moves 20000 steps of S-curve acceleration motion at the following speed

Start speed: 1000 pss

Driving speed: 40000 pss

Acceleration time: 0.4 sec

6.5 Single axis quantitative exponential acceleration / deceleration motion:

6.5.1 List of required functions:

Set to exponential acceleration / deceleration	adt8949_set_admode
--	--------------------

Set start speed	adt8949_set_startv
Set driving speed	adt8949_set_speed
Set acceleration	adt8949_set_acc
Driving instruction	adt8949_pmove
Read driving status	adt8949_get_status
Note	See Important Note

Example 1: Full S-curve

Purpose:

Let X-axis moves 20000 steps of exponential acceleration motion at the following speed

Start speed: 1000 pss

Driving speed: 40000 pss

Acceleration time: 0.4 sec

6.6 Single axis quantitative trigonometric acceleration and deceleration mode motion:

6.6.1 List of required functions:

Set to exponential acceleration / deceleration	adt8949_set_admode
Set start speed	adt8949_set_startv
Set driving speed	adt8949_set_speed
Set acceleration	adt8949_set_acc
Driving instruction	adt8949_pmove
Read driving status	adt8949_get_status
Note	See Important Note

Example 1: Full S-curve

Purpose:

Let X-axis moves 20000 steps of exponential acceleration motion at the following speed; if the derivative of pulse equivalent is 1000, i.e. 1mm

Start speed: 1000 pss

Driving speed: 40000 pss

Acceleration time: 0.4 sec

6.7 Multi-axis motion

6.7.1 List of required functions:

Set to trapezoidal acceleration /deceleration	adt8949_set_admode
Set start speed	adt8949_set_startv
Set driving speed	adt8949_set_speed
Set acceleration	adt8949_set_acc
Set jerk	adt8949_set_jcc
Driving instruction	adt8949_pmove
Continuous motion instructions	adt8949_continue_move
Read driving status	adt8949_get_status
Note	See Important Note

Although the above is single axis, you can set the data of additional axes at the same time in practice. They won't affect each other. In X-axis driving, set the parameters of Y-axis properly and then drive the Y-axis. It will not have any effect on the motion of X-axis, so that

four axes can be operated independently.

Below is a simple example. X-axis moves 1000 steps at a constant speed (1000 pps), Y-axis moves 300,000 steps in linear acceleration/deceleration (start speed: 10000pps, drive speed: 200,000 pps, acceleration time: 0.2 sec), Z-axis performs complete S-curve accelerated continuous motion (start speed: 1000 pps, drive speed: 4000 pps, acceleration time: 1.2 sec), and A-axis performs continuous motion at a constant speed (300,000 pps); press the “S” key to stop.

6.7.2 Important Note:

The control card initialization function involved in this routine only need to be set once before setting the pulse mode and pulse equivalent function.

Chapter 7 Interpolation Motion Control

All the routines in this chapter are independent of each other. In the process of implementing the project, card initialization and pulse equivalent configuration only need calling once, that is, no setting is required after card initialization and pulse equivalent setting. For the settings of pulse equivalent, see Chapter 3 Section 3.1.2 Pulse Equivalent Settings.

7.1 Two-axis linear interpolation (constant speed)

7.1.1 List of required functions:

Set start speed	adt8949_set_startv
Set driving speed	adt8949_set_speed
Driving instruction	adt8949_inp_move4
Read interpolation status	adt8949_get_inp_status
Number of segments of pretreatment cache	adt8949_set_precount
Note	See Important Note

Interpolation speed bases on resultant velocity. Below is a simple example of constant speed linear interpolation. The constant speed driving of arc interpolation is basically same as multi-axis linear interpolation.

7.1.2 Important Note:

The control card initialization function involved in this routine only need to be set once before setting the pulse mode and pulse equivalent function.

7.2 Two-axis linear interpolation (acceleration / deceleration)

7.2.1 List of required functions:

Set acceleration / deceleration mode	adt8949_set_admode
Set start speed	adt8949_set_startv
Set driving speed	adt8949_set_speed

Set acceleration	adt8949_set_acc
Set deceleration	adt8949_set_dec
Interpolation instruction	adt8949_inp_move4
Read interpolation status	adt8949_get_inp_status
Number of segments of pretreatment cache	adt8949_set_precount
Note	See Important Note

For the acceleration / deceleration driving of two-axis linear interpolation, set the interpolation axis to trapezoidal acceleration/ deceleration, S-curve acceleration/ deceleration, exponential acceleration/ deceleration, and trigonometric acceleration/ deceleration.

7.2.2 Important Note:

The control card initialization function involved in this routine only need to be set once before setting the pulse mode function.

Linear interpolation is suitable for trapezoidal, S-shaped, exponential, and trigonometric acceleration/deceleration.

7.3 2D arc interpolation (acceleration/deceleration)

7.3.1 List of required functions:

Set acceleration / deceleration mode	adt8949_set_admode
Set start speed	adt8949_set_startv
Set driving speed	adt8949_set_speed
Set acceleration	adt8949_set_acc

Interpolation instruction	adt8949_inp_arc2
Read interpolation status	adt8949_get_inp_status
Number of segments of pretreatment cache	adt8949_set_precount
Note	See Important Note

Two-axis arc interpolation is generally driven by constant speed or trapezoidal and trigonometric acceleration/deceleration, but can't be driven by S-curve or exponential acceleration/deceleration.

Constant speed drive is relatively simple. You just need to set the start speed of the first axis same to the driving speed.

Acceleration/deceleration driving requires setting acceleration / deceleration mode. The example below illustrates the driving method by driving a full circle of 10 mm radius.

7.3.2 Important Note:

The control card initialization function involved in this routine only need to be set once before setting the pulse mode function.

Arc interpolation is only suitable for trapezoidal and trigonometric acceleration/deceleration.

7.4 3D arc interpolation (acceleration/deceleration)

7.4.1 List of required functions:

Set acceleration / deceleration mode	adt8949_set_admode
Set start speed	adt8949_set_startv
Set driving speed	adt8949_set_speed

Set acceleration	adt8949_set_acc
Interpolation instruction	adt8949_inp_arc3
Read interpolation status	adt8949_get_inp_status
Number of segments of pretreatment cache	adt8949_set_precount
Note	See Important Note

Three-axis arc interpolation is generally driven by constant speed or trapezoidal and trigonometric acceleration/deceleration, but can't be driven by S-curve or exponential acceleration/deceleration.

Three-axis arc interpolation doesn't need entering the center of the circle. You can just enter the intermediate point and the end point of the arc to draw an arc segment or a circle.

Three-axis arc interpolation command can also be used for flat arc interpolation.

Constant speed drive is relatively simple, and just set the initial speed of the first axis to same as the driving speed.

Acceleration and deceleration drive to set the acceleration/deceleration mode. The driving method is described below with driving space arc segment as an example.

7.4.2 Important Note:

The control card initialization function involved in this routine only need to be set once before setting the pulse mode function.

Arc interpolation is only suitable for trapezoidal and trigonometric acceleration/deceleration.

Chapter 8 Track Motion Control

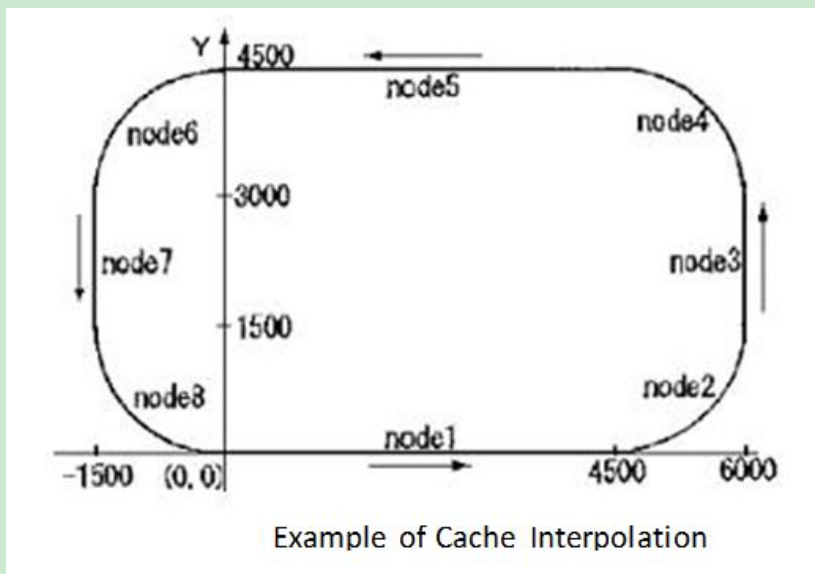
All the routines in this chapter are independent of each other. In the process of implementing the project, card initialization and pulse equivalent configuration only need calling once, that is, no setting is required after card initialization and pulse equivalent setting. For the settings of pulse equivalent, see Chapter 3 Section 3.1.2 Pulse Equivalent Settings.

8.1 Cache interpolation

8.1.1 List of required functions:

Set to acceleration/deceleration mode	adt8949_set_admode
Set start speed	adt8949_set_startv
Set driving speed	adt8949_set_speed
Set acceleration	adt8949_set_acc
Set deceleration	adt8949_set_dec
Linear interpolation instruction	adt8949_inp_move4
Arc interpolation instruction	adt8949_inp_arc2
Note	See Important Note

Constant speed interpolation cache is similar to single interpolation. Just set the start speed same as the driving speed and set the number of preprocessing cache segments to a predetermined value. Cache interpolation of acceleration/deceleration only requires setting the drive speed greater than the start speed.



In the above case, the start speed is set to 1000 pps, the driving speed is 2000 pps, the acceleration is 2000 pps/sec, the last segment in the figure is arc interpolation and the radius is 1500.

8.1.2 Important Note:

The control card initialization function involved in this routine only need to be set once before setting the pulse mode and pulse equivalent function.

The number of pretreatment segment should be set. Only interpolation cache over 10,000 segments needs to be queried with `adt8949_get_fifo_len`.

Chapter 9 Universal Digital I/O

The motion control card provides users with a universal digital input/output port. The host can operate the input/output port through instructions.

9.1 Input port definition: See Hardware CHAPTER 2 Electrical Connection for specific definition

9.2 Output port definition: See Hardware CHAPTER 2 Electrical Connection for specific definition

9.3 Output port:

9.3.1 List of required functions:

Output function	write_bit
Note	See Important Note

9.3.2 Important Note:

Output port defines port number as needed.

9.4 Input port:

9.4.1 List of required functions:

Read the function of input point	read_bit
Note	See Important Note

9.4.2 Important Note:

This function is called according to the return value of the port.

Chapter 10 Auxiliary Control

All the routines in this chapter are independent of each other. In the process of implementing the project, card initialization and ratio only need to be set once in the process of system initialization. For the setting of ratio, see Chapter 3 Section 3.1 Card initialization.

11.1 Position locking:

11.1.1 List of required functions:

Locking mode	adt8949_set_lock_position
Check if position lock is executed	adt8949_get_lock_status
Get locking position	adt8949_get_lock_position
Note	See Important Note

11.1.2 Important Note:

One locking signal is bound to one axis.

The control card initialization function involved in this routine only need to be set once in program initialization.

Chapter 11 List of ADT8949 Basic Library Functions

List of V100 library functions

Function Category	Function Name	Description	Page
Basic parameters	adt8949_initial	Initialize the card	12.1.1
	adt8949_close_card	Close source of motion card	12.1.2
	adt8949_get_lib_version	Get current library version	12.1.3
	adt8949_get_firmware_ver	Get current firmware version	12.1.4
	adt8949_get_card_index	Get DIP switch No.	12.1.5
	adt8949_set_pulse_mode	Set the pulse output mode	12.1.6
	adt8949_set_limit_mode	Set the limit mode	12.1.7
	adt8949_set_stop0_mode	Set stop0 signal	12.1.8
	adt8949_set_stop1_mode	Set stop1 signal	12.1.9
	adt8949_set_input_mode	Set input signal mode (including positive/negative limit, home)	12.1.10
	adt8949_set_gear	Set the pulse equivalent	12.1.11
	adt8949_set_input_filter	Set filter level of input signal	12.1.12
	adt8949_set_actual_count_mode	Set the working mode of actual counter (encoder input)	12.1.13
	adt8949_set_emergency_stop_mode	Set the emergency stop signal mode	12.1.14
	adt8949_set_softlimit_mode	Set software limit mode	12.1.15
Reset	adt8949_reset_card	Reset the motion card	12.2.1

Drive status checking	adt8949_get_status	Get axis drive state	12.3.1
	adt8949_get_status_all	Get the drive state of all axes	12.3.2
	adt8949_get_inp_status	Get interpolation drive status	12.3.3
	adt8949_get_communication_err	Get the number of communication errors	12.3.4
	adt8949_get_inp_index	Get the current interpolation instruction index	12.3.5
Motion parameter setting	adt8949_set_precount	Set the number of pretreatment cache segments	12.4.1
	adt8949_set_jcc	Set S-shaped jerk	12.4.2
	adt8949_set_acc	Set axis acceleration	12.4.3
	adt8949_set_dec	Set axis deceleration	12.4.3
	adt8949_set_admode	Set axis acceleration/ deceleration mode	12.4.4
	adt8949_set_startv	Set axis start speed	12.4.5
	adt8949_set_speed	Set axis running speed	12.4.5
	adt8949_set_endv	Set axis end speed	12.4.5
	adt8949_set_speed_constraint	Set speed constraint for motion path connection	12.4.6
	adt8949_set_acc_constraint	Set acceleration constraint for motion path connection	12.4.7
	adt8949_set_arc_speed_clamp	Set arc speed clamp	12.4.8
	adt8949_set_command_pos	Set axis pulse logic position	12.4.9
	adt8949_set_actual_pos	Set axis pulse actual position	12.4.10
	adt8949_set_syncpos	Synchronize axis cache position and actual position	12.4.11
	adt8949_set_follow_axis	Set follow axis	12.4.12
	adt8949_set_rate1	Set total speed rates	12.4.13
	adt8949_set_rate2	Set speed rate of single axis	12.4.14

Parameter checking	adt8949_get_command_pos	Get the logical location of each axis	12.5.1
	adt8949_get_actual_pos	Get the actual position of each axis	12.5.2
	adt8949_get_speed	Get current driving speed of each axis	12.5.3
	adt8949_get_fifo_len	Query remaining segments in 10000 segments interpolation cache area	12.5.4
	adt8949_get_arc2_length	Get the length of two-axis arc	12.5.5
	adt8949_get_arc3_length	Get the length of three-axis arc	12.5.6
	adt8949_get_syserr	Get the latest error number of the system	12.5.7
	adt8949_get_stopdata	Get the stop data of each axis	12.5.8
Driving	adt8949_pmove	Single axis quantitative motion	12.6.1
	adt8949_abs_pmove	Absolute coordinates quantitative driving	12.6.2
	adt8949_continue_move	Single axis continuous motion	12.6.3
	adt8949_inp_arc2	Two-axis relative arc interpolation	12.6.4
	adt8949_inp_abs_arc2	Two-axis absolute arc interpolation	12.6.5
	adt8949_inp_arc3	Three-axis relative arc interpolation	12.6.6
	adt8949_inp_abs_arc3	Three-axis absolute arc interpolation	12.6.7
	adt8949_inp_move4	Four-axis interpolation instruction (relative position)	12.6.8

	adt8949_inp_abs_move4	Set four-axis interpolation instruction (absolute position)	12.6.9
	adt8949_time_move4	Four-axis relative coordinates linear interpolation (specify motion time)	12.6.10
	adt8949_time_abs_move4	Four-axis absolute coordinates linear interpolation (specify motion time)	12.6.11
	adt8949_inp_NURBS	NURBS interpolation	12.6.12
	adt8949_dec_stop	Driving deceleration stop	12.6.13
	adt8949_sudden_stop	Driving immediately stop	12.6.14
Switch quantity	adt8949_get_out	Get output point status	12.7.1
	adt8949_write_bit	Set output IO status (write_bit)	12.7.2
	adt8949_read_bit	Get input IO status (read_bit)	12.7.3
	adt8949_get_gpio	Read IO status by group	12.7.4
	adt8949_set_gpio	Operate output by group	12.7.5
	adt8949_set_multi_io	Set voltage level for multiple output points	12.7.6
	adt8949_set_daout	Set DA output voltage	12.7.7
FIFO operation output	adt8949_set_fifo_io	Single point IO output in interpolation	12.8.1
	adt8949_set_fifo_multi_io	Set voltage level for multiple output points in interpolation	12.8.2
	adt8949_set_fifo_delay	Specific position delay motion in interpolation	12.8.3
	adt8949_set_fifo_pulser	Insert pulse generator in interpolation	12.8.4
	adt8949_get_fifo_event_len	Query remaining number of	12.8.5

		segments in 1000 events cache section	
Position locking	adt8949_set_lock_position	Set position locking function	12.9.1
	adt8949_get_lock_status	Get position locking status	12.9.2
	adt8949_get_lock_position	Get the position of position locking	12.9.3
	adt8949_clr_lock_status	Clear locking status	12.9.4
	adt8949_set_EXlock_position	Set extended position locking function	12.9.5
	adt8949_get_EXlock_status	Get extended position locking status	12.9.6
	adt8949_get_EXlock_position	Get the position of extended position locking	12.9.7
	adt8949_clr_EXlock_status	Clear extended locking status	12.9.8
Homing module	adt8949_SetHomeMode_Ex	Set home parameter	12.10.1
	adt8949_SetHomeSpeed_Ex	Home speed parameter	12.10.2
	adt8949_HomeProcess_Ex	Start homing	12.10.3
	adt8949_GetHomeStatus_Ex	Get home status	12.10.4
One-dimensional position comparator	adt8949_get_pos_compare_len	Get margin of one-dimensional position comparator	12.11.1
	adt8949_set_pos_compare_mode	Set the mode of one-dimensional position comparator	12.11.2
	adt8949_clear_pos_compare_point	Clear all comparison points of one-dimensional position comparator	12.11.3
	adt8949_set_pos_compare_io	Add output point control function of one-dimensional	12.11.4

		position comparator	
	adt8949_set_pos_compare_multi_io	Add multiple output point control function of one-dimensional position comparator	12.11.5
	adt8949_set_pos_compare_pulse	Add pulse output function of one-dimensional position comparator	12.11.6
	adt8949_set_pos_compare_stop_axis	Add axis stop function of one-dimensional position comparator	12.11.7
Helical interpolation	adt8949_inp_helix2	Relative coordinate helical interpolation based on plane arc	12.12.1
	adt8949_inp_abs_helix2	Absolute coordinate helical interpolation based on plane arc	12.12.2
Handwheel function	adt8949_set_hand_wheel_mode	Set handwheel function mode	12.13.1

Chapter 12 Detailed Description of ADT8949 Basic Library Functions

12.1 Basic parameter setting

12.1.1 Initialize the card

Function	int adt8949_initial(void)
Function description:	Initialize the card
Parameter settings:	None
Return value:	>0: it indicates the quantity of ADT8949 cards. If it is 3, the available card numbers will be 0, 1 and 2 respectively; =0: no ADT8949 card is installed. <0: -1 indicates port drivers are not installed. -2 indicates PCI bridge failure. -3 indicates DSP program download error -4 indicates hardware exception or DLL version does not match -5 indicates failed to create mutex -6 indicates failed to open mutex -7 other causes of exceptions -8 indicates repeat of DIP switch setting, and DIP switch pointing should be adjusted manually

12.1.2 Close source of motion card:

Function	int adt8949_close_card(void)
Function description:	Close source of motion card
Parameter settings	None
Return value:	0: Successful; -1: Failed

Remark	After calling this function, the control card will automatically reset, clear all motion instructions, and clear all output points; Using VB6.0 programming, if this function isn't called, the encoder may exit unexpectedly
--------	--

12.1.3 Get the current library version:

Function	int adt8949_get_lib_version(void)
Function description:	Get current library version
Parameter settings	None
Return value:	Library (dll) version number, where: bit0~bit15: program library version number, bit16 ~ bit31: program library category number.
Remark	<ul style="list-style-type: none"> ◆ Example: When the acquired library version number is 0 and category number is 1, the current version number of the library is displayed as: 1.0. ◆ ADT-8949 has driver library version number and firmware version number. The same firmware version may have different driver library version, so pay attention to the information of both version numbers for problem tracing.

12.1.4 Get current firmware version:

Function	int adt8949_get_firmware_ver(int cardno)
Function description	Get current firmware version
Parameter settings	cardno : Card number
Return value:	The return value includes firmware version and card ID, of which: bit0~bit15: DSP version; bit16~bit23: FPGA version; bit24~bit26:















	FPGA serial number (1 indicates the enhanced series ADT-8949G1, 2 indicates track series ADT-8949B1, and 3 indicates general series ADT-8949A1); bit27~bit31: control card DIP ID
Remark	<ul style="list-style-type: none"> ◆ Example: When acquired FPGA serial No. is 1, FPGA version is 7, and DSP version is 25, the firmware version is displayed as: 1.7.25. ◆ ADT-8949 has driver library version number and firmware version number. The same firmware version may have different driver library version, so pay attention to the information of both version numbers for problem tracing.

12.1.5 Get card number:

Function	int adt8949_get_card_index (int *num,int index[10]);
Function description	Get DIP switch No. of currently available card
Parameter settings	*num: number of available cards index: DIP switch No.
Return value:	0: Correct Non-0: Wrong
Remark	<ul style="list-style-type: none"> ◆ Call this function to obtain DIP switch number after a successful initialization; ◆ The DIP switch numbers are arranged in sequence close to the CPU, for example: index [0] indicates the DIP switch number closest to the CPU, followed by index [1]....

12.1.6 Set the work mode of output pulse:

Function	int adt8949_set_pulse_mode(int cardno, int axis, int value,int logic,int dir_logic);
Function	Set the work mode of output pulse

description	Default mode: pulse + direction, positive logic pulse, positive logic direction signal																													
Parameter settings	<div>cardno Card number</div> <div>axis Axis number (1-4)</div> <div>value 0: pulse + pulse 1: pulse + direction 2: 90° phase difference 2-phase pulse</div> <div>logic 0: positive logic pulse, 1: negative logic pulse</div> <div>dir-logic 0: positive logic direction signal, 1: negative logic direction signal</div>																													
Return value:	0: Correct Non-0: Wrong																													
Remark	<div>Fig. 1:</div> <div><div>Both pulse and direction are positive logic setting</div><table><tr><th rowspan="2">Pulse output mode</th><th rowspan="2">Driving direction</th><th colspan="2">Output signal wave</th></tr><tr><th>PUCW signal</th><th>DRACW signal</th></tr><tr><td rowspan="2">Independent two pulses mode</td><td>+ direction driving output</td><td></td><td>Low level</td></tr><tr><td>- direction driving output</td><td>Low level</td><td></td></tr><tr><td rowspan="2">One pulse mode</td><td>+ direction driving output</td><td></td><td>Low level</td></tr><tr><td>- direction driving output</td><td></td><td>High level</td></tr></table></div> <div>Fig. 2:</div> <div><div>Positive logic pulse : </div><div>Negative logic pulse: </div></div> <div>Fig. 3:</div> <table><tr><th>dir_logic</th><th>Positive direction pulse output</th><th>Negative direction pulse output</th></tr><tr><td>0</td><td>Low</td><td>Hi</td></tr><tr><td>1</td><td>Hi</td><td>Low</td></tr></table>	Pulse output mode	Driving direction	Output signal wave		PUCW signal	DRACW signal	Independent two pulses mode	+ direction driving output		Low level	- direction driving output	Low level		One pulse mode	+ direction driving output		Low level	- direction driving output		High level	dir_logic	Positive direction pulse output	Negative direction pulse output	0	Low	Hi	1	Hi	Low
Pulse output mode	Driving direction			Output signal wave																										
		PUCW signal	DRACW signal																											
Independent two pulses mode	+ direction driving output		Low level																											
	- direction driving output	Low level																												
One pulse mode	+ direction driving output		Low level																											
	- direction driving output		High level																											
dir_logic	Positive direction pulse output	Negative direction pulse output																												
0	Low	Hi																												
1	Hi	Low																												

12.1.7 Set positive/negative limit signal mode:

Function	int adt8949_set_limit_mode(int cardno,int axis, int v1,int v2,int logic);
Function description	Set positive/negative limit mode
Parameter	cardno Card number

settings	axis Axis number (1-4) v1 0: Positive limit active, 1: Positive limit inactive v2 0: Negative limit active, 1: Negative limit inactive logic 0: Low level active, 1: High level active
Return value:	0: Correct Non-0: Wrong
Remark	The default mode is: positive limit active, negative limit active, low level active. Immediate stop mode in positive and negative limit stop;

12.1.8 Set STOP0 (mechanical home) signal mode:

Function	int adt8949_set_stop0_mode(int cardno,int axis, int v,int logic,int admode);
Function description	Set STOP0 (mechanical home) signal mode
Parameter settings	cardno Card number axis Axis number (1-4) v 0: Inactive 1: Active logic 0: Low level active, 1: High level active admode 0: Deceleration stop, 1: Immediate stop
Return value:	0: Correct Non-0: Wrong
Remark	The default mode is: STOP0 inactive. stop0 signal is mechanical home, the four corresponding XYZA axes are XHM/IN12, YHM/IN13, ZHM/IN14, and AHM/IN15

12.1.9 Set STOP1 (servo home) signal mode:

Function	int adt8949_set_stop1_mode(int cardno,int axis, int v,int logic,int admode);
Function description	Set STOP1 (servo home) signal mode

Parameter settings	cardno	Card number	
	axis	Axis number (1-4)	
	v	0: Inactive	1: Active
	logic	0: Low level active,	1: High level active
	admode	0: Deceleration stop,	1: Immediate stop
Return value:	0: Correct Non-0: Wrong		
Remark	The default mode is: STOP1 inactive. stop1 signal is the servo home or Z-phase signal, the four corresponding XYZA axes are IN44, IN45, IN46, and IN47		

12.1.10 Set input signal mode (including positive/negative limit, home):

Function	int adt8949_set_input_mode(int cardno,int axis,int mode,int port,int logic,int admode);		
Function description	Set input signal mode (including positive/negative limit, home)		
Parameter settings	cardno	Card number	
	axis	Axis number (1-4)	
	mode	Mode 0: Positive limit, 1: Negative limit 2: Home (or encoder Z-phase zero)	
	port	Port number; specify the following input port (0-18, 36-47) as positive limit, negative limit, and home (or servo zero); set the port parameter to 255 or -1 if the corresponding mode is disabled.	
	logic	Active voltage level 0: Low level active, 1: High level active	
	admode	Whether use deceleration when limit is active, 0: deceleration stop; 1: immediate stop	
Return value:	0: Correct Non-0: Wrong		
Remark	For the convenience of the machine limit, home, and servo		

	<p>Z-phase signal setting, the following functions have been packaged: <code>adt8949_set_limit_mode (...)</code>, <code>adt8949_set_stop0_mode (...)</code>, <code>adt8949_set_stop1_mode (...)</code></p> <p>The default mode is: Positive limit low level active, negative limit low level active, home signal (or encoder Z-phase signal) inactive; Positive and negative limit temporarily support the immediate stop mode only;</p> <p>The home or encoder Z-phase zero mode used for homing can't be enabled at the same time; you can set homing first, and then reset encoder Z-phase.</p> <p>This function can only specify fast input port, which has particular function (0-18, 36-47) When the encoder signals of four driving axes (A phase, B phase, Z phase) are used as digital input points, the corresponding ports are:</p> <p>IN36~IN39: A1...A4 IN40~IN43: B1...B4 IN44~IN47: Z1...Z4</p> <p>When used as digital inputs, ABZ phases can only accept 5V input; to connect to 24V input, please connect 3K resistor in series.</p>
--	--

12.1.11 Setting derivative of axis pulse equivalent:

Function	<code>int adt8949_set_gear(int cardno, int axis, float gear)</code>
Function description	Set the pulse equivalent derivative of each axis, the number of pulses corresponding to 1mm is 1000 by default
Parameter settings	<p><code>cardno</code> Card number</p> <p><code>axis</code> Axis number (1-4)</p> <p><code>gear</code> Pulse equivalent derivative (1~10000); it may be floating-point numbers, the default is 1000, that is,</p>

	moves 1mm every 1000 pulses, and pulse equivalent is 0.001 (mm/p)
Return value:	0: Correct Non-0: Wrong
Remark	Gear is the derivative of pulse equivalent, known as the number of pulses per 1mm, or control 1/gear mm corresponding to each pulse.

12.1.12 Set the filter level of input signal:

Function	int adt8949_set_input_filter(int cardno,int gp,int grade);
Function description	Set the filter level of input signal The default mode is no filter for all input signals.
Parameter settings	cardno Card number gp Input signal types 0: limit, home, common IO 1: encoder signal (A, B, Z) grade Range: 0-15. 0 indicates no filter; if it is set to n, the filter time is $2^{(n-1)}$ us (microsecond)
Return value:	0: Correct Non-0: Wrong
Remark	The default mode is that no input signal is filtered.

12.1.13 Set the working mode of actual position counter (encoder input):

Function	int adt8949_set_actual_count_mode(int cardno, int axis, int value,int dir_logic);
Function description	Set the working mode of actual position counter (encoder input) Default mode: A/B phase pulse input, positive direction logic
Parameter settings	cardno Card number axis Axis number (1-6), X-axis number of handwheel is 5,

12.1.15 Setting software limit mode:

Function	int adt8949_set_softlimit_mode(int cardno,int axis,int EnableFlag,float Ppos,float Npos,int admode);		
Function description	Setting software limit mode		
Parameter settings	cardno	Card number	
	axis	Axis number (1-4)	
	EnableFlag	0: Invalid 1: Valid	
	Ppos	Positive trigger position. Unit: mm	
	Npos	Negative trigger position. Unit: mm	
	admode	0: deceleration stop; 1: immediate stop	
Return value:	0: Correct -1: Wrong		
Remark	The default state is invalid		

12.2 Reset motion card

12.2.1 Reset motion card

Function	int adt8949_reset_card(int cardno);		
Function description	Reset motion card		
Parameter settings	cardno	Card number	
Return value:	0: Correct -1: Wrong		
Remark	After calling this function, the control card will erase all cache events and motion instruction data. If synchronized axis has been set before, the synchronization relationship will also be cleared, but the limit, acceleration, pulse equivalent and other motion parameters that have been set won't be cleared and do not need re-setting;		

	When the machine has external emergency stop, positive/negative limit or abnormal stop, or before cache interpolation of large amount of data, it is recommended to call the function to reset the control card.
--	--

12.3 Driving status checking

12.3.1 Get the driving status of each axis

Function	int adt8949_get_status(int cardno,int axis,int *v)
Function description	Get the driving status of each axis
Parameter settings	cardno Card number axis Axis number (1-4) v Pointer of driving status 0: Driving ends Non-0: Driving
Return value:	0: Correct -1: Wrong
Remark	This function is used to query API. For consecutive query, insert a Sleep(1) sentence between two queries

12.3.2 Get driving status of all axes:

Function	int adt8949_get_status_all(int cardno,int *v)
Function description	Get driving status of all axes
Parameter settings	cardno Card number v Pointer of driving status. bit0~bit3 indicate axis 1~4; bit15 indicates interpolation axis 0: Driving ends Non-0: Driving
Return value	0: Correct 1: Wrong
Remark	This function is used to query API. For consecutive query, insert a Sleep(1) sentence between two queries

12.3.3 Get driving status of interpolation:

Function	int adt8949_get_inp_status(int cardno,int *v)
Function description	Get driving status of interpolation
Parameter settings	cardno Card number v Pointer of driving status 0: Driving ends Non-0: Driving
Return value	0: Correct 1: Wrong
Remark	This function is used to query API. For consecutive query, insert a Sleep(1) sentence between two queries

12.3.4 Reading the times of communication errors:

Function	int adt8949_get_communication_err(int cardno,int type,int *ErrCount);
Function description	Reading the times of communication errors
Parameter settings	cardno Card number type 0: Control card 1: Wiring board ErrCount Communication error count pointer
Return value	0: Correct 1: Wrong
Remark	The number of errors will be cleared when the power of control card and wiring board is cut off; Non-zero communication error does not indicate control card output error; if checkout fails, the communication data is automatically resent.

12.3.5 Getting the index information for the current interpolation motion

Function	int adt8949_get_inp_index(int cardno,unsigned short *index);
Function description	Getting the index information for the current interpolation motion
Parameter settings	cardno Card number index Pointer of index information (0-65535)
Return value	0: Correct 1: Wrong
Remark	Get the interpolation instruction of the control card running, instruction number index is sent to the card when calling the interpolation function. When using the default interpolation instruction index 0, the index value obtained doesn't have reference meaning.

12.4 Motion parameter setting

☛ **Note:** The following parameters are uncertain after initialized and should be set before using

12.4.1 Setting the segments of interpolation pretreatment:

Function	int adt8949_set_precount(int cardno,unsigned short prec)
Function description	Setting the segments of interpolation pretreatment, and start speed look-ahead
Parameter settings	cardno Card number prec Look-ahead segments (0~50); the function is turned off if it is 0
Return value	0: Correct 1: Wrong
Remark	Pretreatment segments do not represent the cache capacity. For small line segments, it is recommended to set more than 30 segments to ensure consistency of the interpolation. After starting the pretreatment, call the function adt8949_set_acc_constraint (...) and adjust the acceleration constraint value of each axis to enhance

	the processing efficiency and stability of the machine. Look-ahead segments do not need repeated setting.
--	---

12.4.2 Set S-shaped jerk:

Function	int adt8949_set_jcc(int cardno, int axis,unsigned short jcc);
Function description	Set S-shaped jerk; default: 0
Parameter settings	cardno Card number axis Axis number (1-4, number of interpolation axis: 0x3f) Jcc Jerk: (0~10)
Return value	0: Correct Non-0: Wrong
Remark	Jcc, S acceleration/deceleration mode sets jerk grade (default: 0), the smaller the value, the more obvious the acceleration/deceleration effect

12.4.3 Set axis acceleration and deceleration:

Function	int adt8949_set_acc(int cardno, int axis,float add) int adt8949_set_dec(int cardno, int axis,float add)
Function description	Set axis acceleration (set_acc) and deceleration (set_dec) Unit: mm/sec^2 Default: acceleration = deceleration =500mm/sec^2
Parameter settings	cardno Card number axis 1~4 single axis INPA_AXISREG interpolation axis add Acceleration/deceleration value (100~100,000)
Return value	0: Correct 1: Wrong
Remark	◆ ADT-8949 supports asymmetric acceleration/deceleration; when setting acceleration, deceleration will be equal to the acceleration by default; therefore, to set the deceleration,

	<p>place set_dec after set_acc, or else the deceleration value will be overwritten by acceleration.</p> <p>◆ INPA_AXISREG is the axis number of group A interpolator, interpolation acceleration/deceleration, which calculates the synthesized position, is also set independently; therefore, the acceleration/deceleration of interpolation axis is less than or equal to interpolation acceleration/deceleration.</p> <pre>#define INPA_AXISREG 0x3f</pre>
--	--

12.4.4 Set axis acceleration/deceleration mode:

Function	int adt8949_set_admode(int cardno, int axis, unsigned short mode)
Function description	Set axis acceleration/deceleration mode
Parameter settings	<p>cardno Card number</p> <p>axis Axis number (1-4, number of interpolation axis: INPA_AXISREG)</p> <p>mode Range (0-3)</p> <p>0 S-shaped acceleration/deceleration mode</p> <p>1 trapezoidal acceleration/deceleration mode</p> <p>2 exponential acceleration/deceleration mode</p> <p>3 trigonometric acceleration/deceleration mode</p>
Return value	0: Correct -1: Wrong
Remark	<p>Remark: The default option is trapezoidal acceleration/deceleration mode</p> <p>Point motion and single linear interpolation can use any mode, Single arc interpolation uses mode 1 and 3.</p> <p>Spline interpolation uses mode 1.</p> <p>To use non-trapezoidal acceleration/deceleration mode in interpolation, ensure that the segment of pretreatment cache is zero.</p>

	#define INPA_AXISREG	0x3f
--	----------------------	------

12.4.5 Set axis running speed, start speed and end speed:

Function	int adt8949_set_speed(int cardno, int axis,float speed); int adt8949_set_startv(int cardno, int axis,float speed); int adt8949_set_endv(int cardno, int axis,float speed);	
Function description	Set axis running speed, start speed and end speed in sequence	
Parameter settings	cardno	Card number
	axis	1~4 single axis
		INPA_AXISREG interpolation axis
	speed	Speed, unit: mm/sec, 0.001~100000
Return value	0: Correct	Non-0: Wrong
Remark	When setting the start velocity by default, the end velocity is equivalent to the start velocity, so the end velocity set_endv must be set after set_startv, or else the end velocity will be overwritten by start velocity. #define INPA_AXISREG 0x3f	

12.4.6 Set the maximum axis speed at the connection of two line segments in cache interpolation:

Function	int adt8949_set_speed_constraint (int cardno, int axis,float speed);	
Function description	Set speed constraint at motion path connection	
Parameter settings	cardno	Card number
	axis	1~4 single axis
		INPA_AXISREG interpolation axis
	speed	Speed, unit: mm/sec, 0.001~100000

Return value	0: Correct	Non-0: Wrong
Remark	<p>◆ This function doesn't need to be called generally.</p> <p>◆ In special process, it is used to set the maximum axis speed at the connection of two line segments in cache interpolation; it is active in pretreatment, and limits the maximum speed of each axis at the connection of line segments.</p>	
	#define INPA_AXISREG	0x3f

12.4.7 Set acceleration constraint at motion path connection:

Function	int adt8949_set_acc_constraint(int cardno, int axis, float add);	
Function description	Set acceleration constraint at motion path connection	
Parameter settings	cardno	Card number
	axis	Axis number (1-4)
	add	Range (100-100000), default: 8000mm/sec ²
Return value	0: Correct	Non-0: Wrong
Remark	<p>◆ In cache interpolation, it is used to set the acceleration constraint of axis; it is active in pretreatment, and limits the maximum speed change of each axis at the connection of line segments.</p> <p>◆ In the debugging process, constraint value can be gradually increased as long as the interpolation accuracy won't be affected, which will reduce machine shake in interpolation.</p>	

12.4.8 Set arc speed clamp:

Function	int adt8949_set_arc_speed_clamp(int cardno, float radius, float speed);
Function	When using cache interpolation, set axis acceleration constraint

description	active in pretreatment, and limit the maximum speed changes of each axis at the connection of line segments.	
Parameter settings	cardno	Card number
	radius	Radius coefficient
	speed	Speed coefficient, range (0.01-100000 mm/sec)
	Default	Radius coefficient = 10mm, speed coefficient = 100mm/sec
Return value	0: Correct	Non-0: Wrong
Remark	<p>◆ The function will be active in flat arc or spatial arc interpolation; if the function is called and the actual radius is smaller than the radius coefficient, the arc speed is less limited;</p> <p>◆ If the actual radius is larger than the radius coefficient, the arc speed limit is greater. If the actual radius is equal to the radius coefficient, the maximum speed of arc is equal to the speed coefficient.</p>	

12.4.9 Set logic position of axis pulse:

Function	int adt8949_set_command_pos(int cardno, int axis, long pos)	
Function description	Set logic position of axis pulse	
Parameter settings	cardno	Card number
	axis	1~4 single axis
	pos	Logic position
	Set range (-2147483648~+2147483647)	
Return value	0: Correct	Non-0: Wrong
Remark		

12.4.10 Set actual position of axis pulse:

Function	int adt8949_set_actual_pos(int cardno, int axis, long pos);
Function description	Set actual position
Parameter settings	cardno Card number axis Axis number (1-4) pos Range (-2147483648~+2147483647)
Return value	0: Correct -1: Wrong
Remark	

12.4.11 Axis cache position and logic position synchronization:

Function	int adt8949_set_syncpos(int cardno, int axisbit)
Function description	Axis cache position and logic position synchronization
Parameter settings	cardno Card number axisbit Axis mapping bit, bit0 indicates 1# axis, bit1 indicates 2# axis, bit2 indicates 3# axis, bit3 indicates 4# axis
Return value	0: Successful; Non-0: Failed
Remark	Set after zeroing and tool setting and before one group of interpolation motion to improve accuracy. Do not set when the machine is in motion.

12.4.12 Set follow axis:

Function	int adt8949_set_follow_axis(int cardno, int slaveaxis, int masteraxis);
Function description	Set follow axis
Parameter	cardno Card number

settings	slaveaxis Slave axis (following master axis), Axis number (1-4) masteraxis Master axis (followed by slave axis), Axis number (0-4), 0: cancel follow
Return value	0: Successful; Non-0: Failed
Remark	

12.4.13 Set total speed rate:

Function	int adt8949_set_rate1(int cardno,float rate);
Function description	Set total speed rate
Parameter settings	cardno Card number rate Total system rate (0~2.0)
Return value	0: Successful; -1: Failed
Remark	<ul style="list-style-type: none"> ◆ ADT-8949 does not support online speed change. The value is to change the speed dynamically in the concept of rate, and acceleration/deceleration also will be changed accordingly. The acceleration/deceleration mode will not be affected, that is, when S-shaped acceleration/deceleration is selected, the acceleration curve is still S shape even if the speed rate is set to the highest. ◆ Rate1 affects all axes and all motion modes, so rate1=0 has pause effect. ◆ Rate1 refreshes immediately; so if the rate of change is too large, it will cause speed step; the ideal method is to set timing gradually to produce a deceleration effect. ◆ The impact of rate1 on actual speed also depends on rate2; the actual axis speed =rate1*rate2[axis]*speed;

12.4.14 Set speed rate of single axis:

Function	int adt8949_set_rate2(int cardno,int axis ,float rate);
Function description	Set speed rate of each axis
Parameter settings	cardno Card number axis Axis number (1~4), interpolation axis INPA_AXISREG rate Rate (0~2.0)
Return value	0: Successful; -1: Failed
Remark	<ul style="list-style-type: none"> ◆ It has same effect as rate1, but the scope is only limited to the specified axis number. ◆ Rate2 also refreshes immediately, so pay attention to speed step; set update gradually according to the note of rate1 to produce an acceleration/deceleration effect. ◆ During single-axis motion, the impact of rate2 on the actual speed is $\text{rate1} * \text{rate2} [\text{axis}] * \text{speed}$. Through the multiple proportions effect of rate1 and rate2, the speed can be increased by up to four times of the set speed, and the minimum is zero. ◆ When speed ratio of interpolation axes INPA_AXISREG is rate2 during interpolation motion, it only take effect for the interpolation axis, and the effect is $\text{rate1} * \text{rate2} * \text{speed}$; <pre>#define INPA_AXISREG 0x3f</pre>

12.5 Parameter checking

The following functions can be called at any time

12.5.1 Get the logic position of each axis:

Function	int adt8949_get_command_pos(int cardno,int axis,long *pos)
Function	Get the pulse count value sent by the card

description	
Parameter settings	cardno Card number axis 1~4 single axis pos Pulse count
Return value	0: Correct Non-0: Wrong
Remark	This function can get the logic position of the axis at any time; in case that the motor isn't out of step, pos value is the current position of the axis.

12.5.2 Get the value of motor AB phase encoder feedback counter:

Function	int adt8949_get_actual_pos(int cardno,int axis,long *pos)
Function description	Get the actual position of each axis
Parameter settings	cardno Card number axis Axis number (1-4) pos Pointer of actual position
Return value	0: Correct -1: Wrong
Remark	

12.5.3 Get current running speed of the axis (instantaneous speed):

Function	int adt8949_get_speed(int cardno,int axis,float *speed);
Function description	Get current running speed of the axis (instantaneous speed)
Parameter settings	cardno Card number axis 1~4 single axis, INPA_AXISREG interpolation axis speed Instantaneous speed, unit: mm/sec

Return value	0: Correct Non-0: Wrong
Remark	If the number of feed axis is INPA_AXISREG, the feedback is resultant speed. #define INPA_AXISREG 0x3f

12.5.4 Query left interpolation segments in cache zone:

Function	int adt8949_get_fifo_len(int cardno,int *len);
Function description	Query left interpolation of 10,000 segments in cache zone
Parameter settings	cardno Card number len Cache length variable to be gotten
Return value	0: Correct Non-0: Wrong
Remark	The interpolation cache zone has 10000 segments in total. Interpolation data and cache events are stored in different regions of the control card. An arc occupies four segments of cache space, and a full circle occupies eight segments of cache space. It is recommended to issue motion data when the cache margin is greater than 8 segments. This function is used to query API. For consecutive query, insert a Sleep(1) sentence between two queries.

12.5.5 Get arc length of two axes:

Function	int adt8949_get_arc2_length(unsigned char arcmap,float pos[4],float Center[4],int dir,float *length);
Function description	Get arc length of two axes
Parameter settings	axismap Axis selection mapping flag, mark any two axes in the space for plane arc interpolation. bit0 indicates 1# axis, bit1 indicates 2# axis, may mark any two axes in the space to execute plane arc interpolation

	pos	Target point coordinates of the arc (relative to the current point)
	Center	Center coordinates of the arc (relative to the current point)
	dir	Arc direction (1: clockwise; 0: counterclockwise)
	length	Arc length of two axes
Function	0: Correct	Non-0: Wrong
Function description		

12.5.6 Get arc length of three axes:

Function	int adt8949_get_arc3_length(unsigned char arcmap,float pos2[4],float pos3[4],float *length,int arc_flag);	
Function description	Get arc length of three axes	
Parameter settings	arcmap	axis selection mapping flag, bit0 indicates 1# axis, bit1 indicates 2# axis, bit2 indicates 3# axis, up to three bits can be marked as 1, and support arc interpolation up to three axes
	pos2	Second point coordinates of the arc (relative to the current point)
	pos3	Third point coordinates of the arc (relative to the current point)
	length	Arc length of three axes
	arc_flag	0: Arc 1: Circle
Return value	0: Correct	Non-0: Wrong
Remark		

12.5.7 Get the latest error number of the system:

Function	int adt8949_get_syserr(int cardno,int *ErrNum);
Function description	Get the latest error number of the system
Parameter settings	cardno Card number ErrNum Pointer of the system error number
Return value	Remark: Get system error number regularly, and view the running of the motion card
Remark	

12.5.8 Get stop data of each axis:

Function	int adt8949_get_stopdata(int cardno,int axis,int *value);
Function description	Function: Get stop data of each axis
Parameter settings	cardno Card number axis Axis number (1-4) value Pointer of stop data (0: no error stop; non-0: has limit, home or encoder Z phase signal triggers stop): bit0==1: positive limit triggers stop bit1==1: negative limit triggers stop bit2==1: home signal triggers stop bit3==1: encoder Z phase signal triggers stop bit4==1: external emergency stop signal triggers stop bit5==1, software positive limit triggers stop bit6==1, software negative limit triggers stop
Return value	0: Correct -1: Wrong
Remark	Stop information may also appear in combination; for example: bit0 and bit2 = 1 indicates that both positive limit and home signal are triggered, causing axis stop. For consecutive query, insert a

	Sleep(1) sentence between two queries, or else it will affect the efficiency of the control card.
--	---

12.6 Driving

12.6.1 Single axis quantitative motion:

Function	int adt8949_pmove(int cardno,int axis,float pos)
Function description	Single axis quantitative relative motion (PTP) Set the relative position of axis motion, unit: mm
Parameter settings	cardno Card number axis Axis number (1~4) pos >0: positive motion <0: negative motion Range (+/- 9999999.999mm)
Return value	0: Correct Non-0: Wrong state
Remark	Before writing drive command, be sure to set the acceleration/deceleration parameters required by the speed curve properly

12.6.2 Absolute coordinates quantitative driving:

Function	int adt8949_abs_pmove(int cardno,int axis,float pos)
Function description	Absolute coordinates quantitative driving, unit: mm
Parameter settings	cardno Card number axis Axis number (1-4) pos Unit: mm, (+/- 9999999.999) >0: positive driving, <0: negative driving
Return value	0: Correct Non-0: Wrong state
Remark	Note: Before writing in the drive command, be sure to set the speed parameter correctly.

12.6.3 Single axis continuous driving:

Function	int adt8949_continue_move(int cardno,int axis,int dir);
Function description	Single axis continuous driving, Unit: mm
Parameter settings	ccardno Card number axis Axis number (1-4) dir 0: Positive movement 1: Negative movement
Return value	0: Correct Non-0: Wrong state
Remark	Note: Before writing drive command, be sure to set the speed parameters properly.

12.6.4 Two-axis relative coordinates arc interpolation:

Function	int adt8949_inp_arc2(int cardno,unsigned short index,unsigned char arcmap,float pos[4],float Center[4],int dir);
Function description	Two-axis relative coordinates arc interpolation, Unit: mm
Parameter settings	cardno Card number index Data index, used to identify the motion information, usually set to 0 axismap axis selection mapping flag, bit0 indicates 1# axis, bit1 indicates 2# axis, mark any two axes in the space for plane arc interpolation pos Target point coordinates of the arc (relative to the current point). Unit: mm Center Center coordinates of the arc (relative to the current point). Unit: mm dir Arc direction (1: clockwise; 0: counterclockwise)
Return value	0: Correct Non-0: Wrong state

Remark	<ul style="list-style-type: none"> ◆ axismap: axis mapping in binary. 0011: axis X and Y involve in two-axis arc interpolation, 0110: axis Y and Z involve in two-axis arc interpolation. ◆ An arc segment occupies 4 cache spaces, and a full circle occupies 8 cache spaces. If this function is called to draw a full plane circle, set a certain number of preprocessing cache segments, or else there will be an acceleration/deceleration process at semicircle.
--------	--

12.6.5 Two-axis absolute coordinates arc interpolation:

Function	int adt8949_inp_abs_arc2(int cardno,unsigned short index,unsigned char arcmap,float pos[4],float Center[4],int dir);	
Function description	Two-axis absolute coordinates arc interpolation, unit: mm	
Parameter settings	cardno	Card number
	index	Index info, used to identify the motion information, usually set to 0.
	axismap	axis selection mapping flag, bit0 indicates 1# axis, bit1 indicates 2# axis, mark any two axes in the space for plane arc interpolation.
	pos	Absolute coordinates of the target point on the arc, unit: mm
	Center	Absolute coordinates of center point on the arc, unit: mm
	dir	Arc direction (1: CW, 0: CCW)
Return value	0: Correct	Non-0: Wrong state
Remark	<ul style="list-style-type: none"> ◆ axismap: axis mapping in binary. 0011: axis X and Y involve in two-axis arc interpolation, 0110: axis Y and Z involve in two-axis arc interpolation. ◆ An arc segment occupies 4 cache spaces, and a full circle 	

	occupies 8 cache spaces. If this function is called to draw a full plane circle, set a certain number of preprocessing cache segments, or else there will be an acceleration/deceleration process at semicircle.
--	--

12.6.6 Three-axis relative coordinates arc interpolation

Function	int adt8949_inp_arc3(int cardno,unsigned short index,unsigned char arcmap,float pos2[4],float pos3[4],int arc_flag)
Function description	Three-axis relative coordinates arc interpolation, unit: mm
Parameter settings	<p>cardno Card number</p> <p>index Index info, usually set to 0</p> <p>arcmap axis selection mapping flag, up to three bits can be marked as 1, and support arc interpolation up to three axes</p> <p>pos2 Coordinates of second point on the arc (relative to the current point), unit: mm</p> <p>pos3 Coordinates of third point on the arc (relative to the current point), unit: mm</p> <p>arc_flag 0: Arc 1: Circle</p>
Return value	0: Correct Non-0: Wrong
Remark	<ul style="list-style-type: none"> ◆ axismap axis mapping in binary. 0111: axis X, Y and Z involve in three-axis arc interpolation, 1110: axis Y, Z and A involve in three-axis arc interpolation. ◆ pos2 and pos3 are the coordinates of the second point and the third point; if arc_flag=1, the machine will continue to move from the third point coordinates and reach the starting point of the arc to create a full circle. ◆ An arc segment occupies 4 cache spaces, and a full circle occupies 8 cache spaces. If this function is called to draw a full plane circle, set a certain number of preprocessing cache

	segments, or else there will be an acceleration/deceleration process at semicircle.
--	---

12.6.7 Three-axis absolute coordinates arc interpolation

Function	int adt8949_inp_abs_arc3(int cardno,unsigned short index,unsigned char arcmap,float pos2[4],float pos3[4],int arc_flag);
Function description	Three-axis absolute coordinates arc interpolation, unit: mm
Parameter settings	cardno Card number index Index info, usually set to 0. arcmap axis selection mapping flag, up to three bits can be marked as 1, and support arc interpolation up to three axes pos2 Absolute coordinates of second point on the arc, unit: mm pos3 Absolute coordinates of third point on the arc, unit: mm arc_flag 0: Arc 1: Circle
Return value	0: Correct Non-0: Wrong
Remark	<ul style="list-style-type: none"> ◆ axismap axis mapping in binary. 0111: axis X, Y and Z involve in three-axis arc interpolation, 1110: axis Y, Z and A involve in three-axis arc interpolation. ◆ pos2 and pos3 are the coordinates of the second point and the third point; if arc_flag=1, the machine will continue to move from the third point coordinates and reach the starting point of the arc to create a full circle. ◆ An arc segment occupies 4 cache spaces, and a full circle occupies 8 cache spaces. If this function is called to draw a full plane circle, set a certain number of preprocessing cache segments, or else there will be an acceleration/deceleration process at semicircle.

12.6.8 Four-axis interpolation instruction (relative position)

Function	int adt8949_inp_move4(int cardno,unsigned short index,float pos1,float pos2,float pos3,float pos4)
Function description	Set four-axis interpolation instruction (relative position)
Parameter settings	cardno Card number index Index info, usually set to 0. pos1, pos2, pos3, pos4 indicate the relative movement of axis XYZA, unit: mm
Return value	0: Correct Non-0: Wrong
Remark	<ul style="list-style-type: none"> ◆ The interpolation instruction will be cached. Since the control card has 10000 segments of cache, the instruction can be returned soon; if you want to determine whether the motion ends, combine the interpolation status to determine. ◆ If an axis shift is set to 0, the axis will not be occupied by interpolator, that is, the position can also be driven by PTP. ◆ Similarly, if the set axis number has been occupied by PTP, the interpolation command will fail, all cache instructions including previously fed instructions will be invalid and should be re-computed and fed into the interpolator. ◆ If the return value isn't zero, search the content of error through error status table; if the cache is full, it will also return an error. ◆ If you set up a synchronized axis, the pos parameter of the auxiliary axis should be set to 0, or else the function calling will fail.

12.6.9 Four-axis interpolation instruction (absolute position):

Function	int adt8949_inp_abs_move4(int cardno,unsigned short
----------	--

	index,unsigned char axismap,float pos1,float pos2,float pos3,float pos4)
Function description	Set four-axis interpolation instruction (absolute position)
Parameter settings	<p>cardno Card number</p> <p>index Index info, usually set to 0.</p> <p>axismap Axis mapping bit, bit0 indicates 1# axis, bit1 indicates 2# axis... for the axes not marked, the target position will not take effect.</p> <p>pos1, pos2, pos3, pos4 indicate the coordinates of axis XYZA, unit: mm</p>
Return value	0: Correct Non-0: Wrong
Remark	<ul style="list-style-type: none"> ◆ The interpolation instruction will be cached. Since the control card has 10000 segments of cache, the instruction can be returned soon; if you want to determine whether the motion ends, combine the interpolation status to determine. ◆ Axismap is used to identify the control axis; if the corresponding bit is marked as 0, it won't be processed regardless of pos value, and the card will automatically take the current position. ◆ If an axis shift is set to 0, the axis will not be occupied by interpolator, that is, the position can also be driven by PTP. ◆ Similarly, if the set axis number has been occupied by PTP, the interpolation command will fail, all cache instructions including previously fed instructions will be invalid and should be re-computed and fed into the interpolator. ◆ If the return value isn't zero, search the content of error through error status table; if the cache is full, it will also return an error.

12.6.10 Four-axis relative coordinates linear interpolation (specifying motion time)

Function	int adt8949_time_move4(int cardno,unsigned short index,float pos1,float pos2,float pos3,float pos4,float time);
Function description	Set four-axis interpolation instruction (relative position)
Parameter settings	<p>cardno Card number</p> <p>index Data index, used to identify the data of the motion, generally set to 0 to</p> <p>pulse1,pulse2,pulse3,pulse4 indicate the relative distance of axis XYZW</p> <p>time Time required by motion of the straight line, unit: ms</p>
Return value	0: Correct Non-0: Wrong
Remark	<ul style="list-style-type: none"> ◆ The straight line moves at constant speed within the specified time. Before calling this function, ensure that the number of preprocessing segments is zero, or else the call will fail; ◆ If synchronized axis is set, pos parameter of slave axis should be set to 0, or else the call will fail. ◆ This function is suitable for users to design speed plan; the user can split the track into small segments and specify the speed of each segment (each segment is uniform). Since the interpolation instruction is executed in the cache interpolation, each segment is interrelated and reflects the designed planning speed on the overall trend.

12.6.11 Four-axis absolute coordinates linear interpolation (specifying motion time)

Function	int adt8949_time_abs_move4(int cardno,unsigned short index,unsigned char axismap,float pos1,float pos2,float pos3,float
----------	---

	pos4,float time);
Function description	Four-axis absolute coordinates linear interpolation (specifying motion time)
Parameter settings	<p>cardno Card number</p> <p>index Data index, used to identify the data of the motion, generally set to 0</p> <p>axismap Axis mapping bit, bit0 indicates 1# axis, bit1 indicates 2# axis; if the axis is not marked, the target position will not take effect.</p> <p>pulse1, pulse2, pulse3, pulse4 indicate the coordinates that axis XYZW move to.</p> <p>time Time required by motion of the straight line (non-cumulative time), unit: ms (millisecond)</p>
Return value	0: Correct Non-0: Wrong
Remark	<p>◆ The straight line moves at constant speed within the specified time. Before calling this function, ensure that the number of preprocessing segments is zero, or else the call will fail.</p> <p>◆ If synchronized axis is set, pos parameter of slave axis should be set to 0, and axismap value doesn't need to consider the slave axis position.</p> <p>◆ This function is suitable for users to design speed plan; the user can split the track into small segments and specify the speed of each segment (each segment is uniform). Since the interpolation instruction is executed in the cache interpolation, each segment is interrelated and reflects the designed planning speed on the overall trend.</p>

12.6.12 NURBS interpolation

Function	int adt8949_inp_NURBS(int cardno,unsigned short
----------	--

	index,unsigned char axismap,float comp[][4],float weight[],float node[],int nodenum);
Function description	NURBS interpolation
Parameter settings	<p>cardno Card number</p> <p>index Data index, used to identify the data of the motion, generally set to 0.</p> <p>Axismap Axis selection mapping flag, only three positions can be marked to 1, that is, support NURBS interpolation up to three axes.</p> <p>comp Spline control point (relative to the current point, the coordinates of first control point should be 0, or else it will return error).</p> <p>weight The weight corresponding to the control point of the spline > 0. If it is set to -1, the default weight of the control points of AutoCAD is used.</p> <p>node Node value of spline</p> <p>nodenum Number of spline nodes</p>
Return value	0: Correct Non-0: Wrong
Remark	<p>◆ For NURBS interpolation, it is recommended to set a bigger number of pre-processing cache segments and use trapezoidal acceleration/deceleration mode; calling the function to set arc speed clamp will affect planning of spline interpolation speed.</p> <p>◆ Using spline interpolation will occupy a larger number of cache segments of control card. To continue to call interpolation instruction, check the remaining cache capacity in advance.</p>

12.6.13 Set deceleration stop for manual intervention of axis

Function	int adt8949_dec_stop(int cardno,int axis)
Function description	Set deceleration stop for manual intervention of axis Deceleration is the effective value in the last setting Manual intervention deceleration is allowed in linear mode only
Parameter settings	cardno Card number axis Axis number (1~4)
Return value	0: Correct Non-0: Wrong state
Remark	<ul style="list-style-type: none"> ◆ If the set axis participates in the interpolation, other corresponding axes in the interpolation will be decelerated to stop. ◆ After calling this function, the machine stops abnormally. It is recommended to call the function adt8949_reset_card (...) when the axis stops.

12.6.14 Set immediate stop for manual intervention of axis

Function	int adt8949_sudden_stop(int cardno,int axis)
Function description	Set immediate stop for manual intervention of axis
Parameter settings	cardno Card number axis Axis number (1~4)
Return value	0: Correct Non-0: Wrong state
Remark	<ul style="list-style-type: none"> ◆ If the set axis participates in the interpolation, other corresponding axes in the interpolation will be decelerated to stop. ◆ After calling this function, the machine stops abnormally. It is recommended to call the function adt8949_reset_card (...) when the axis stops.

12.7 Switch quantity

Read IO status and read/write IO

12.7.1 Getting status of output point

Function	int adt8949_get_out(int cardno, int number)
Function description	Getting status of single output point
Parameter settings	cardno Card number number Output point (0~31)
Return value	Current status of the output point: 0: Off, 1: On, -1: Parameter error
Remark	

12.7.2 Operating output point

Function	int adt8949_write_bit(int cardno, int number, int value)
Function description	Operate a single output point, turn on/off the output point
Parameter settings	cardno Card number number Output point (0-31) value 0: Off 1: On
Return value	0: Correct Non-0: Wrong state
Remark	

12.7.3 Getting status of input point

Function	int adt8949_read_bit(int cardno, int number)
Function description	Getting status of single input point
Parameter	cardno Card number

	point; if the bit value is 0, output point isn't affected levelmap Voltage level setting of this group of IO (determine the status of certain output point through bit value, e.g. OUT0 corresponds to bit0, OUT15 corresponds to bit15); only the output points with iomap bit value 1 are affected.
Return value	0: Low level Non-0: Wrong
Remark	◆ This function only take effect for the output port with iomap value is 1 in the group.

12.7.7 Set DA output voltage

Function	int adt8949_set_daout(int cardno,int port,int value);
Function description	Set DA output voltage
Parameter settings	cardno Card number port Set DA output port (1-2) value Set DA output size (0-10), unit: V
Return value	0: Correct Non-0: Wrong
Remark	DA output voltage is accurate to two decimal places

12.8 FIFO operation output

12.8.1 Single-point output in interpolation

Function	int adt8949_set_fifo_io(int cardno,int number,int value,float speed);
Function description	Single-point output in interpolation
Parameter settings	cardno Card number number Output point (0-14)

	value 0: Low level 1: High level speed speed constraint, -1 indicates no speed constraint before IO action, others: range (0.0-100000.0 mm/sec)
Return value	0: Low level Non-0: Wrong
Remark	<ul style="list-style-type: none"> ◆ Speed setting is same as the motion track speed, and motion track won't decelerate. If the speed setting is smaller, the motion track will decelerate to the set speed in advance before corresponding IO operation, and then accelerate to motion track speed, and form V-shaped process. ◆ FIFO event capacity of a single control card is 1000. If the function returns a value other than 0, the event cache is full, and should be retransmitted when the event cache has margin. To query remaining number of segments in cache event zone, call the function <code>adt8949_get_fifo_event_len(...)</code>. ◆ The range of output points must be 0~14.

12.8.2 Set voltage level for multiple output points at the same time in interpolation:

Function	<code>int adt8949_set_fifo_multi_io(int cardno,int gp,short int iomap,short int levelmap,float speed);</code>
Function description	Set voltage level for multiple output points at the same time in interpolation
Parameter settings	ccardno Card number gp Group number; it must be set to 0 (OUT0~OUT14); only the output point setting of group 0 is available temporarily iomap Specify the output point to be operated by bit (bit0~bit14), value 1 indicates operating corresponding output point, value 0 indicates no

	<p>operation</p> <p>levelmap The level setting of this group of IO (setting the status of a output point by bit value, e.g. OUT0 corresponds to bit0, and OUT15 corresponds to bit15) only affects the output points with iomap value is 1</p> <p>speed speed constraint, -1 indicates no speed constraint before IO action, others: range (0.0-100000.0 mm/sec)</p>
Return value	0: Low level Non-0: Wrong
Remark	<p>◆ The speed setting is consistent with the track velocity, which won't decelerate. If the speed setting is low, the track will decelerate to the set speed in advance, then perform the corresponding IO operation, and then accelerate to the track speed to form a V-process.</p> <p>◆ FIFO event capacity of a single control card is 1000. If the function returns a value other than 0, the event cache is full, and should be retransmitted when the event cache has margin. To query remaining number of segments in cache event zone, call the function <code>adt8949_get_fifo_event_len (...)</code>.</p>

12.8.3 Specific position delay motion in interpolation:

Function	<code>int adt8949_set_fifo_delay(int cardno,int millisecond);</code>
Function description	Specific position delay motion in interpolation
Parameter settings	<p>cardno Card number</p> <p>millisecond delay time, unit: ms</p>
Return value	0: Correct Non-0: Wrong
Remark	<p>◆ FIFO event capacity of a single control card is 1000. If the function returns a value other than 0, the event cache is full, and should be retransmitted when the event cache has</p>

	margin. To query remaining number of segments in cache event zone, call the function <code>adt8949_get_fifo_event_len (...)</code> .
--	--

12.8.4 Insert pulse generator in interpolation:

Function	<code>int adt8949_set_fifo_pulser(int cardno,int port,int NormalLevel,int NormalTime,int UnNormalTime,int ReverseNum,float speed);</code>	
Function description	Insert pulse generator in interpolation	
Parameter settings	cardno	Card number
	port	Output point port of pulse generator (0-14)
	NormalLevel	Normal state voltage level 0: Low, 1: High
	NormalTime	Holding time of normal level, unit: ms
	UnNormalTime	Holding time of non-normal level, unit: ms
	ReverseNum	Reverse times of output level
	speed	peed constraint, -1 indicates no speed constraint before IO action, others: range (0.0-100,000.0 mm/sec)
Return value	0: Correct	Non-0: Wrong
Remark	<ul style="list-style-type: none"> ◆ Speed setting is same as the motion track speed, and motion track won't decelerate. If the speed setting is smaller, the motion track will decelerate to the set speed in advance before corresponding IO operation, and then accelerate to motion track speed, and form V-shaped process. ◆ FIFO event capacity of a single control card is 1000. If the function returns a value other than 0, the event cache is full, and should be retransmitted when the event cache has margin. To query remaining number of segments in cache event zone, call the function <code>adt8949_get_fifo_event_len</code> 	

	<p>(...).</p> <ul style="list-style-type: none"> ◆ Normal level refers to the output level before the number of turns is counted. If the pulse generator is in normal level state before operation, it turns immediately, or else it turns to the normal level state first; it uses a normal level time, and the turn isn't included in the set number of turns.
--	---

12.8.5 Query remaining segments in cache event area:

Function	int adt8949_get_fifo_event_len(int cardno,int *len);
Function description	Query remaining number of segments in 1000 events cache section
Parameter settings	cardno Card number len Pointer of length variable of the cache event to be gotten
Return value	0: Correct Non-0: Wrong
Remark	<ul style="list-style-type: none"> ◆ Interpolation data and cache events are stored in different sections of the control card. ◆ Functions setting cache events include: adt8949_set_fifo_io (...), adt8949_set_fifo_multi_io (...), adt8949_set_fifo_delay (...), adt8949_set_fifo_pulser (...). ◆ This function is to query API. For continuous query, insert a Sleep (1) statement between two queries.

12.9 Position locking

12.9.1 Setting position locking function

Function	int adt8949_set_lock_position(int cardno,int axis,int logic,int type);
Function description	Setting position locking function
Parameter	cardno Card number

settings	axis	Axis number (1-4), the ports corresponding to each axis are: Axis 1: IN12 Axis 2: IN13 Axis 3: IN14 Axis 4: IN15
	logic	Effective level Single edge change: 0: Descend, 1: Ascend Dual edge change: 0: First edge change descends, second edge change ascends; 1: First edge change ascends, second edge change descends
	type	1: Single edge change locking, 2: Dual edge change locking
Return value	0: Correct -1: Wrong	
Remark	<ul style="list-style-type: none"> ◆ Default mode: Locking function is disabled. ◆ When position locking function is triggered, the logic position and actual position when edge change signal occurs will be recorded. When this function is enabled, it is used for one time. To capture the position that edge change occurs again, ensure that the last locking status has been cleared, and enable position locking function again. 	

12.9.2 Getting status of position locking

Function	int adt8949_get_lock_status(int cardno,int axis,int type,int *v);	
Function description	Getting status of position locking	
Parameter settings	cardno	Card number
	axis	Axis number (1-4), the ports corresponding to each axis are: Axis 1: IN12

Remark	◆
--------	---

12.9.7 Getting the position of extended position locking

Function	int adt8949_get_EXlock_position(int cardno,int port,int type,long *pos);	
Function description	Getting the position of extended position locking	
Parameter settings	cardno	Card number
	port	Port (17-18)
	type	0x10: indicates getting the logic position when the first edge change occurs 0x11: indicates getting the actual position when the first edge change occurs 0x20: indicates getting the logic position when the second edge change occurs (for dual-edge change position lock) 0x21: indicates getting the actual position when the second edge change occurs (for dual-edge change position lock)
	pos	Pointer of the position to be gotten
Return value	0: Correct	-1: Wrong
Remark	◆	

12.9.8 Clear extended locking status

Function	int adt8949_clr_EXlock_status(int cardno,int port);	
Function description	Clear extended locking status	
Parameter settings	cardno	Card number
	port	Port (17-18)
Return value	0: Correct	-1: Wrong

Remark	◆
--------	---

12.10 Homing module

12.10.1 Set home signal mode:

Function	int adt8949_SetHomeMode_Ex(int m_nCardNum,int m_nAxisNum,int m_nHomeDir, int m_nStop0Active,int m_nLimitActive,int m_nStop1Active,long m_nBackRange,long m_nEncoderZRange,long m_nOffset);	
Function description	Set home signal, step parameter	
Parameter settings	m_nCardNum	Card number
	m_nAxisNum	Axis number
	m_nHomeDir	Home direction, 0: negative direction, 1: positive direction
	m_nStop0Active	stop0 active level setting; 0: Low level stop; 1: High level stop
	m_nLimitActive	limit signal active level setting; 0: Low level stop; 1: High level stop
	m_nStop1Active	stop1 active level setting; 0: Low level stop; 1: High level stop
	m_fBackRange	Reverse distance >1, shouldn't exceed the distance between positive limit and stop0 (Unit: pulse)
	m_fEncoderZRange	Encoder Z phase range >1 (Unit: pulse)
	m_fOffset	Home offset; =0: No, >0: positive direction, <0: negative direction (Unit: pulse)
Return value	0: Correct	-1 ~ -8: Parameter error
	-x: x parameter error	
Remark		

Return value	0: Correct -1 ~ -3: Parameter error -x: x parameter error
Remark	Homing action is started when this function is called. After started, adt8949_GetHomeStatus_Ex (...) should be called regularly until homing successfully.

12.10.4 Get home status:

Function	adt8949_GetHomeStatus_Ex(int m_nCardNum,int m_nAxisNum);
Function description	Get home status, step parameter
Parameter settings	m_nCardNum Card number m_nAxisNum Axis number m_fGear Pulse equivalent: (1-10000)
Return value	0: Home successfully; -1: Parameter 1 error; -2: Parameter 2 error; -3: Homing isn't started; (1-10) steps 1: Approach home fast and search STOP0 2: Check if STOP0 is found 3: Exit home reversely 4: Check if reverse exit is complete 5: Approach home slowly and search STOP0 6: Check if STOP0 searching completes 7: Approach Z phase slowly, and search STOP1. If STOP1 is set to -1, skip step 7 & 8. 8: Check if STOP1 searching completes 9: Home offset 10: Check home offset -100x: The x homing step is abnormal, e.g.: -1001: the first homing step is abnormal -1020: Homing is terminated

Remark	It should be called regularly in the homing process until homing successfully
--------	---

12.11 One-dimensional position comparator

12.11.1 Query margin of one-dimensional position comparator:

Function	int adt8949_get_pos_compare_len(int cardno,int axis,int *len);
Function description	Query margin of one-dimensional position comparator
Parameter settings	cardno Card number axis Axis number (1-4) len Pointer of the comparison point margin of the axis to be gotten
Return value	0: Correct Non-0: Wrong
Remark	<p>The comparison point capacity of each axis is 500. The functions adding comparison point are:</p> <p>adt8949_set_pos_compare_io(...), adt8949_set_pos_compare_multi_io(...), adt8949_set_pos_compare_pulse(...), adt8949_set_pos_compare_stop_axis(...),</p> <p>This function is used to query API. For consecutive query, insert a Sleep(1) sentence between two queries.</p>

12.11.2 Setting mode of one-dimensional position comparator:

Function	int adt8949_set_pos_compare_mode(int cardno,int axis,int EnableFlag,int cmp_source);
Function description	Setting mode of one-dimensional position comparator
Parameter settings	cardno Card number axis Axis number (1-4)

	EnableFlag Enable flag, 0: Disable 1: Enable cmp_source Comparison source, 0: Logic position 1: Actual position (encoder position)
Return value	0: Correct Non-0: Wrong
Remark	Position comparison is disabled by default

12.11.3 Clear all comparison points of one-dimensional position comparator:

Function	int adt8949_clear_pos_compare_point(int cardno,int axis);
Function description	Clear all comparison points of one-dimensional position comparator
Parameter settings	cardno Card number axis Axis number (1-4)
Return value	0: Correct Non-0: Wrong
Remark	It is recommended to clear the data of comparison points of the axis before adding position comparison points to avoid comparison point data not triggered stored on the control card.

12.11.4 Adding output point control function of one-dimensional position comparator:

Function	int adt8949_set_pos_compare_io(int cardno,int axis,long pos,int dir,int type,int port);
Function description	Adding output point control function of one-dimensional position comparator
Parameter settings	cardno Card number axis Axis number (1-4) pos Comparison position, unit: pulse dir Comparison direction, 0: ≤, 1: ≥ type Control mode: 1: On 2: Off 3: Invert

	port Output point (0-14)
Return value	0: Correct Non-0: Wrong
Remark	Adding comparison point will fail in the following cases: <ol style="list-style-type: none"> 1. The current position has met the trigger condition; 2. Adding over 10 comparison points that should be triggered in the same position to the axis consecutively; 3. The margin of the comparison point of the axis is zero.

12.11.5 Adding multiple output point control function of one-dimensional position comparator:

Function	int adt8949_set_pos_compare_multi_io(int cardno,int axis,long pos,int dir,int iomap,int levelmap);
Function description	Adding multiple output point control function of one-dimensional position comparator
Parameter settings	cardno Card number axis Axis number (1-4) pos Comparison position, unit: pulse dir Comparison direction, 0: ≤, 1: ≥ iomap Specify the output point to be operated by bit (bit0~bit14), value 1 indicates operating corresponding output point, value 0 indicates no operation levelmap The level setting of this group of IO (setting the status of a output point by bit value, e.g. OUT0 corresponds to bit0, and OUT1 corresponds to bit1) only affects the output points with iomap value is 1
Return value	0: Correct Non-0: Wrong
Remark	Adding comparison point will fail in the following cases: <ol style="list-style-type: none"> 1. The current position has met the trigger condition; 2. Adding over 10 comparison points that should be

	<p>triggered in the same position to the axis consecutively;</p> <p>3. The margin of the comparison point of the axis is zero.</p>
--	--

12.11.6 Adding pulse output function of one-dimensional position comparator:

Function	int adt8949_set_pos_compare_pulse(int cardno,int axis,long pos,int dir,int port,int level,int millisecond);
Function description	Adding pulse output function of one-dimensional position comparator
Parameter settings	<p>cardno Card number</p> <p>axis Axis number (1-4)</p> <p>pos Comparison position, unit: pulse</p> <p>dir Comparison direction, 0: ≤, 1: ≥</p> <p>port Pulse output port (0-14)</p> <p>level Active level of output pulse</p> <p>millisecond Time length of output pulse, unit: ms</p>
Return value	0: Correct Non-0: Wrong
Remark	<p>If the function is triggered, the current output point level has been at the set active level, and the output level won't invert temporarily; it will invert once after the set delay in milliseconds.</p> <p>Adding comparison point will fail in the following cases:</p> <ol style="list-style-type: none"> 1. The current position has met the trigger condition; 2. Adding over 10 comparison points that should be triggered in the same position to the axis consecutively; 3. The margin of the comparison point of the axis is zero.

12.11.7 Adding axis stop function of one-dimensional position comparator:

Function	int adt8949_set_pos_compare_stop_axis(int cardno,int axis,long
----------	--

	pos,int dir,int DesAxis,int admode);
Function description	Adding axis stop function of one-dimensional position comparator
Parameter settings	cardno Card number axis Axis number (1-4) pos Comparison position, unit: pulse dir Comparison direction, 0: ≤, 1: ≥ DesAxis Specify the axis number to be stopped (1-4) admode Stop mode, 0: Decelerate 1: Immediate
Return value	0: Correct Non-0: Wrong
Remark	Adding comparison point will fail in the following cases: 1. The current position has met the trigger condition; 2. Adding over 10 comparison points that should be triggered in the same position to the axis consecutively; 3. The margin of the comparison point of the axis is zero.

12.12 Helical interpolation

12.12.1 Relative coordinate helical interpolation based on plane arc:

Function	int adt8949_inp_helix2(int cardno,unsigned short index,int AxisList[4],float pos[4],float center[2],int dir)
Function description	Relative coordinate helical interpolation based on plane arc
Parameter settings	cardno Card number index Index info AxisList Axis number list (1-4); the first two axes are axis numbers involved in plane arc interpolation, and the last two axes are slave axes. If the last two axes are 0, only plane arc interpolation is executed

	<p>pos pos [0] and pos [1] are coordinates of the target point on the arc, pos [2] and pos [3] are absolute coordinates of the target point on follower axis (relative to current point), unit: mm</p> <p>center Coordinates of the center point on the arc (relative to current point), unit: mm</p> <p>dir Arc direction (0: CW, 1: CCW)</p>
Return value	0: Correct Non-0: Wrong
Remark	An arc segment occupies 4 cache spaces, and a full circle occupies 8 cache spaces. If this function is called to draw a closed circular helix, set a certain number of preprocessing cache segments, or else there will be an acceleration/deceleration process at semicircle.

12.12.2 Absolute coordinate helical interpolation based on plane arc:

Function	int adt8949_inp_abs_helix2(int cardno,unsigned short index,int AxisList[4],float pos[4],float center[2],int dir)
Function description	Absolute coordinate helical interpolation based on plane arc
Parameter settings	<p>cardno Card number</p> <p>index Index info</p> <p>AxisList Axis number list (1-4); the first two axes are axis numbers involved in plane arc interpolation, and the last two axes are slave axes. If the last two axes are 0, only plane arc interpolation is executed</p> <p>pos pos [0] and pos [1] are absolute coordinates of the target point on the arc, pos [2] and pos [3] are absolute coordinates of the target point on follower axis, unit:</p>

	mm center Absolute coordinates of the center point on the arc, unit: mm dir Arc direction (0: CW, 1: CCW)
Return value	0: Correct Non-0: Wrong
Remark	Remark: An arc segment occupies 4 cache spaces, and a full circle occupies 8 cache spaces. If this function is called to draw a closed circular helix, set a certain number of preprocessing cache segments, or else there will be an acceleration/deceleration process at semicircle.

12.13 Handwheel function (not enabled)

12.13.1 Set handwheel mode

Function	int adt8949_set_hand_wheel_mode(int cardno,int axis,int fun_mode);
Function description	Set handwheel mode
Parameter settings	cardno Card number axis Axis number (5-6), X-axis number of handwheel is 5, and Y-axis number is 6 fun_mode Function mode, 0: General input port mode; 1: Handwheel encoder signal input mode
Return value	0: Correct -1: Wrong
Remark	Remark: After using handwheel function, set to slow input point mode in time, or else it will affect the efficiency of the wiring board. Slow input point mode is default. After setting to handwheel

	encoder signal input mode, the default mode is PPIN/PMIN pulse input and input signal direction positive logic; to modify encoder input signal type or direction logic, please call the function <code>adt8949_set_actual_count_mode(...)</code> .
--	--

Chapter 13 Troubleshooting

13.1 Motion control card detections fails

If the control card can be detected in the process of using the control card, please follow the method below to eliminate the problem.

- (1) Be sure to follow the installation instructions of the control card to install the drivers step by step, and make sure that the system directory (system32 or System) has the control card dll file;
- (2) Check if the motion control card and the socket contact properly. You can re-insert or replace the slot to test, or clean the gold finger of the control card with an eraser before test;
- (3) In System Device Manager, check if the motion control card conflicts with other hardware. When using PCI cards, remove other boards and cards, such as: sound card and network card; PC104 card allows adjusting DIP switch to reset the base address; the base address used for card initialization in the program must be same as the actual base address;
- (4) Check if the operating system has any problem by re-installing other operating systems;
- (5) If the motion control card still can't be detected after examination with the above steps, please replace the motion control card, and further test if the motion control card has been damaged;

13.2 Motor running abnormal

If the motion control card is normal, but the motor is abnormal, refer to the following troubleshooting method.

- (1) The motor doesn't run when the motion control card sends pulses
 - Check if the control card and the terminal board are connected properly;
 - Check if the pulse and direction signal cables of the motor drive are properly connected to the terminal board;
 - Check if the external power of the servo drive has been connected properly;
 - Check if the servo and stepper motor drive have alarm; if yes, check the reason according to the alarm code;
 - Check if servo SON is connected properly, and if the servo motor has excitation status;
 - For servo motor, please check the drive control; the control card supports "position control mode";
 - Motor or drive is damaged
- (2) The stepper motor screams in operation, motor is significantly out of step.
 - Controller speed is too high; please calculate the motor speed, 10~15 rotations per second is normal for stepper motor;
 - Mechanical part is stuck or too much resistance;
 - Motor selection is inappropriate; please replace a motor of high torque;
 - Check the drive current and voltage, set the current to 1.2 times the rated current of the motor, and set the supply voltage within the rated range of the drive;
 - Check the initial speed of the controller, which is generally 0.5~1; deceleration time is 0.1 second or more;
- (3) Servo or stepper motor has significant vibration or noise during processing
 - Position loop gain and speed loop gain of servo drive are

too large, reduce the position loop gain and speed loop gain of the servo drive in permitted positioning accuracy;

- Machine rigidity is poor; please adjust the structure of the machine;
- Stepper motor selection is inappropriate; please replace a motor of high torque;
- The speed of stepper motor is in the resonance region of the motor, please avoid the resonance region or increase the subdivision;

(4) Motor positioning inaccurate

- Check if the mechanical screw pitch and number of pulses per revolution of the motor and are consistent with the parameters set by the practical application system, i.e. pulse equivalent;
- For servo motor, increase the position loop gain and speed loop gain;
- Check the screw gap of the machine, test the backlash of lead screw with a dial gauge, and adjust the screw if there is gap;
- If the time and position of inaccurate positioning are indefinite, check the external interference signal;
- Motor selection is inappropriate, and shake or out of step occurs in motion;

(5) The motor has no direction

- Check if the DR+ and DR- wiring have error, and are connected securely;
- Check if the pulse mode of the control card is consistent with the actual drive mode; the control card supports “pulse + direction” and “pulse + pulse” mode
- For stepper motor, check if the motor lines have breakage or poor contact;

13.3 Switch input abnormal

In the process of system commissioning and operation, if some input signals are abnormal, please use the method described below to check.

(1) No signal input

- Check if the wiring is correct according to the wiring

diagram of common switch and proximity switch, and ensure that the "optocoupler common terminal" of input signal has been connected to the positive terminal of internal or external power supply (+ 12V or 24V);

- The I/O point input switch of the company is NPN type; if not available, please check the switch model and wiring;
- Check if the optical coupler has been damaged. If the line is normal, the input state does not change when the input point is opened and closed; please use a multimeter to test if the optical coupler has been broken down, and solve the breakdown problem by replacing the optical coupler;
- Check if 12V or 24V switching power supply is normal;
- The switch is damaged;

(2) Signal is intermittent

- Check if there is interference, and test the signal status in I/O test screen; if interference exists, add model 104 monolithic capacitor or use shielded lines;
- The machine has significant shake or unusual stop during normal operation; please check if the limit switch signal has interference or if the limit switch has reliable performance;
- Check if external wiring is in proper contact;

(3) Inaccurate homing

- Speed too fast; reduce homing speed;
- External signal interference; check the sources of interference;
- Homing direction error;
- Homing switch is installed in improper position or switch is loose;

(4) Invalid limit

- In I/O test, check if the limit switch is valid;
- Manual or automatic processing speed is too fast;
- External signal interference; check the sources of interference;
- Manual direction error;
- Limit switch is installed in improper position or switch is

loose;

13.3 Switch output abnormal

Switch output is abnormal; please troubleshoot in the method described below.

(1) Output abnormal

- Check if the lines are correct according to the wiring diagram of the output points described earlier, and ensure that the common output terminal (ground wire) is connected to the ground wire of the power supply;
- Check if the output device has been damaged;
- Check if the optical coupler has been damaged; please use a multimeter to test if the optical coupler has been broken down, and solve the breakdown problem by replacing the optical coupler;
- Security essentials; when the output uses inductive load, connect a freewheeling diode (IN4007 model or IN4001) in parallel;

(2) Method to determine output problem

Disconnect the external wiring of output point, connect a 10K pull-up resistor from the output point to the power supply terminal; connect the output ground wire to GND terminal of the power, put the red pen of the multimeter on the 12V positive pole, put the black pen on the output terminal, and tap the button on the test screen to check if there is voltage output; if yes, check external circuit; if not, check if the common terminal of the card is connected properly, and if the optical coupler has problem;

13.4 Encoder abnormal

If the encoder is abnormal while operating, please check in the method below.

- (1) Check encoder wiring. Make sure that the wiring of encoder complies with differential or collector open circuit mode previously introduced;
- (2) Check encoder voltage. The motion control card accepts +5V signal. If +12V or +24V encoder is used, please connect 1K (+12V) resistor in serial between A/B phase of the encoder and A/B phase of terminal board;
- (3) Encoder counting is inaccurate. The external wire of the encoder must be shielded twisted pair, and the encoder wire should be at least 30~50MM away from the strong electric wires that have strong interference.