ZMotion PC Library Programming Manual

Version 2.0

Copyright statement

Debug the machine pay attention to safety! Be sure to design effective safety devices in the machine, and adding an error handler in software, or loss caused by being transportedAction Technology has no obligation or responsibility responsible.

table of Contents

Chapter OnePC Programming Overview

1.1 Motion Controller Features

ZMC PC motion controller supports direct online control, provided DLL libraries and VC, VB, C #, LABVIEW and other routines. Library also provides support of LINUX and WINCE. Library for all types of controllers.

ZMC-line control of the relative motion controller PCI has the following advantages:

1, the slot is not used, the stability is better.

2, reduce the requirements for the PC, no PCI slots.

3, can use a computer or MINI ARM industrial computers, reducing overall cost.

4, the controller directly to do wiring board, saving space.

5, the program can be run in parallel on the controller, and PC requires only simple interactions, reduce the complexity of PC software. In summary it can be seen, the choice of the motion controller Ethernet interface in place of PCI motion control card, can be

Save space, reduce costs, optimize procedures, wiring more convenient, which is more and more applications using Ethernet The reason network.

1.2 Development Framework

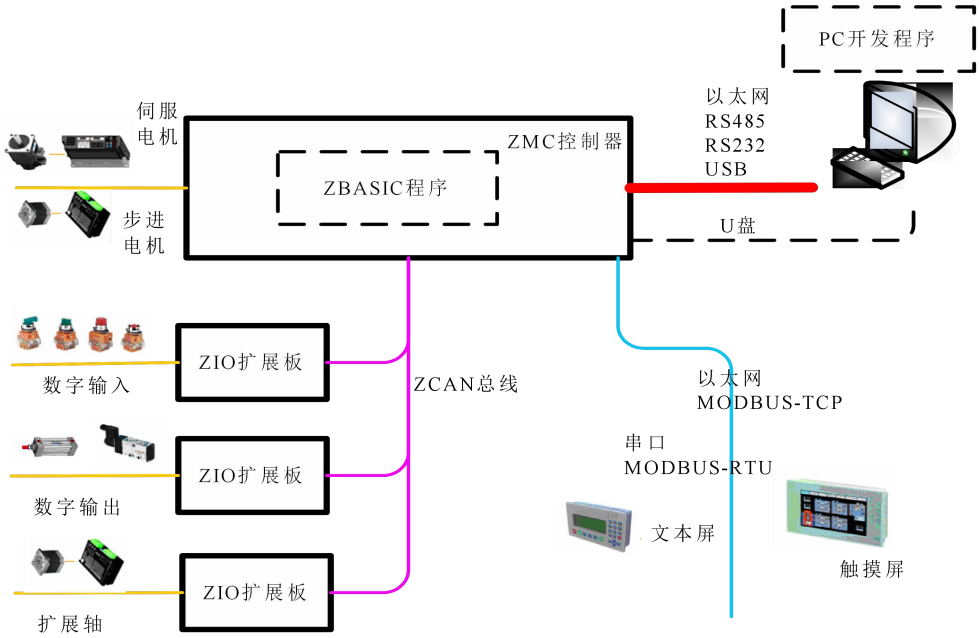


Figure 1.2 ZMC development Chart

1.2 part red bold line in FIG line PC program is interactive with the controller, the controller ZMC Ethernet, USB, serial / 485 is connected to the PC, and the PC program ZBASIC program can run simultaneously on the controller, improving the processing s efficiency.

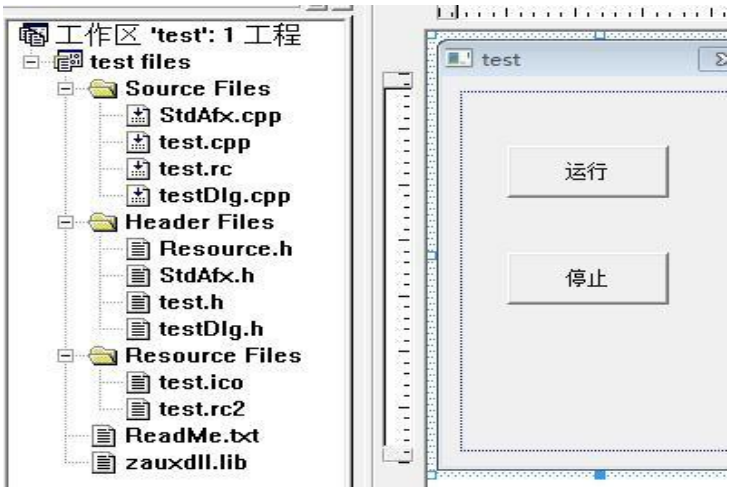
1.3 Development steps

1.3.1 Develop applications using VC

1) Open the VC ++ 6.0.
2) Create a new project
3) Select MFC APPWizard (exe)
4) Select the path to save the project
5) Set project name, select OK

Figure 1.3-1 New Project

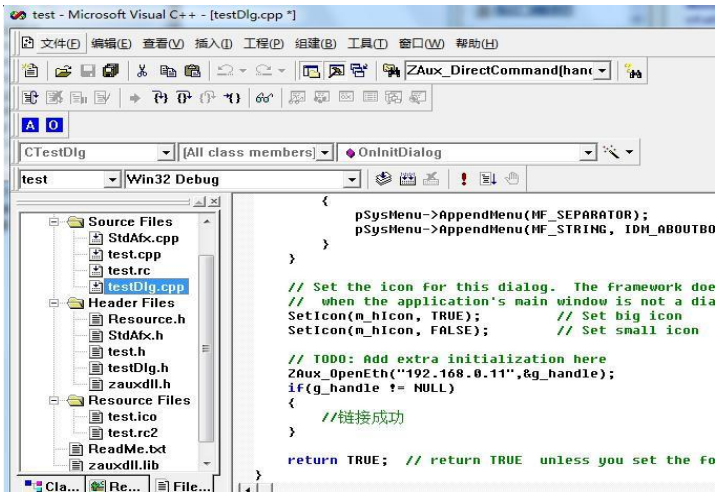
6) In the dialog box select the basic type of application. Completing the New
7) Zauxdll.h library functions to be provided, under zauxdll.lib, zauxdll.dll, and zmotion.dll into the project path (directly



or through zauxdll.cpp source code to join the project)

Figure 1.3-2 Adding Controls

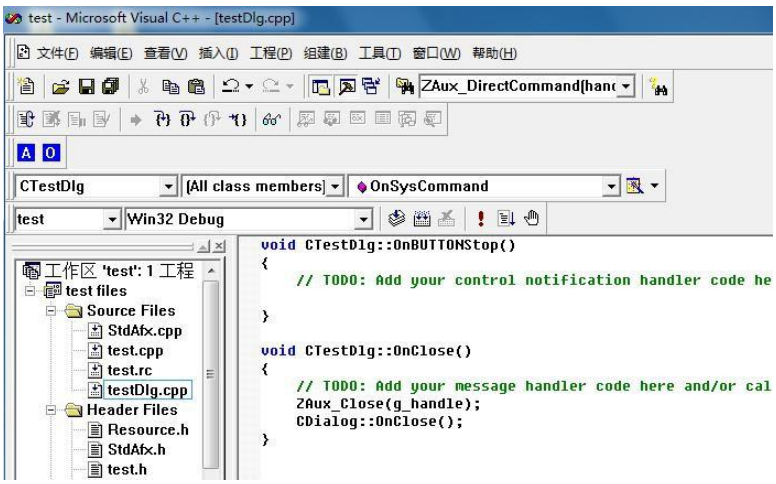
8) Adding header files testDlg.cpp#include "zauxdll.h".
9) Add a link controller code ZAux_OpenEth ("192.168.0.11", & g_handle) in CTestDlg ::



OnInitDialog () function

Figure 1.3-3 Link Controller

10) Add a member function CTestDlg :: OnClose () in CTestDlg; window for closing off controller link. Add ZAux_Close



(g_handle) in function; break the link.

Figure 1.3-4 off controller link

11) Double-click to start, stop adding the corresponding code in the corresponding event function: ZAux_Direct_Singl_Vmove (g_handle, 0,1); ZAux_Direct_Singl_Cancel (g_handle, 0,2);

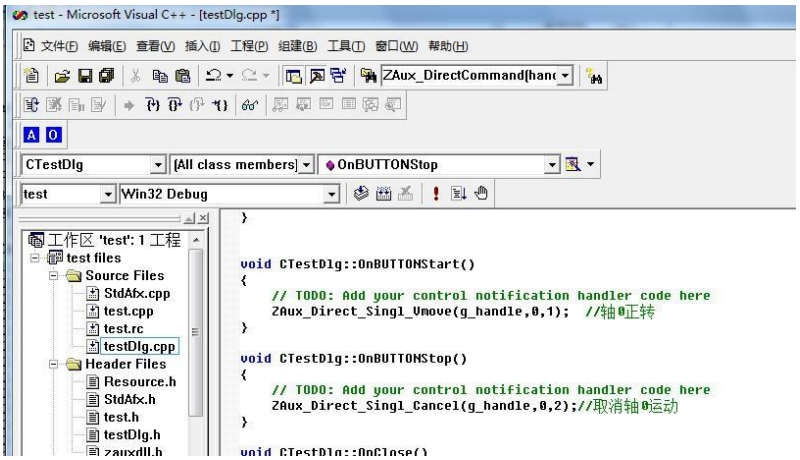


Figure 1.3-5 call the function

After 12) compiler, run the program, press running at the interface, the axis O will be initialized by Super

Of forward rotation, stop the press, the axis 0 stops.

1.4 Packaging method library function

ZAux library package and set the basic parameters of the motion, so as to provide open source, details, see source.

ZAux library by sending commands directly ZBASIC ZAux_DirectCommand ZAux_Execute manner or mode to the controller, the corresponding function can ZBASIC manual commands corresponding description. If the command does not want the package or packages own function, or may be transmitted by ZAux_Execute ZAux_DirectCommand, or modify existing code increase reference to the corresponding function.

Send command string in two ways, direct mode and buffer mode.

Direct method: direct execution individual variables / array / parameters related commands, this time must pass all parameters Specific numerical values shall be not an expression; see ZAux_DirectCommand function. ZAux_DirectCommand (handle controller, the command string, the string returns, returns the character length) buffer system can execute all the commands and supports expressions as parameters, but the speed is slower; see See ZAux_Execute function.

ZAux_Execute (handle controller, the command string, the string returns, returns the character length)

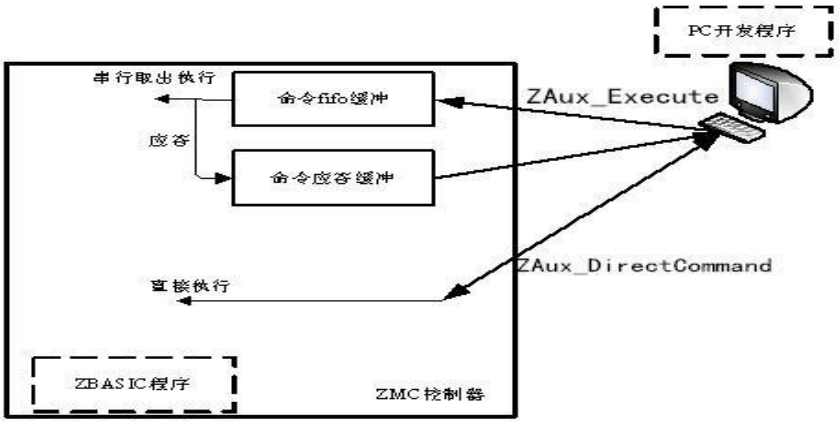


FIG channel command string 1.4-1

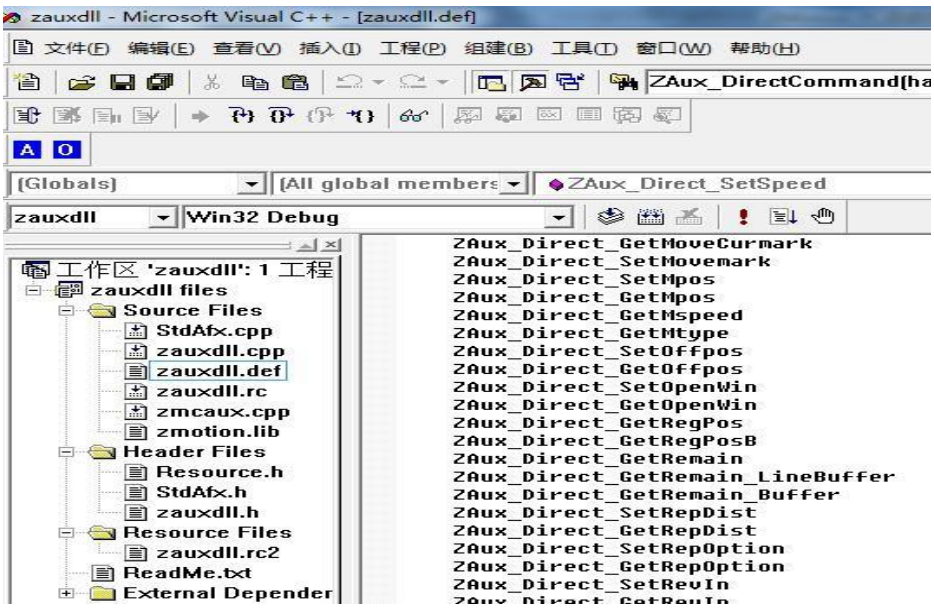
Functions such as setting shaft speed: ZAux_Direct_SetSpeed (ZMC_HANDLE handle, int iaxis, float fValue) equivalent to "SPEED (axis number) = velocity value" BASIC command. Command string is "speed (% d) =% f" when the wrapper function.

Encapsulating step:

a) Preparation of source code: Open the auxiliary source code, adding the corresponding function zmcaux.cpp

```
int32 stdcall ZAux_Direct_SetSpeed (ZMC_HANDLE handle, int iaxis,
float fValue)
{
charcmdbuff [2048]; char
        cmdbuffAck [2048];
if (iaxis> MAX_AXIS_AUX)
{
    returnERR_AUX_PARAERR;
}
sprintf (cmdbuff, "SPEED (% d) =% f", iaxis, fValue); // generates a
corresponding command character
string
return ZAux_DirectCommand (handle, cmdbuff, cmdbuffAck, 2048);
}
```

b) Definition of the newly added function is defined in the zauxdll.def



Adding 1.4-2 FIG function definition

c) increasing the corresponding function in the header file

Figure 1.4-3 Adding header files

d) generating a new compiler DLL

1.5 Multi-Link Controller

Positive motion controller can support one PC control a plurality of controllers, commonly used in the multi-axis, multi-line equipment, workshop or monitoring equipment. Usually a switch, provided by the controller to access each different IP address to the controller. Note: there are multiple devices on the Ethernet switch when, in addition to ensure a different IP address outside, to the MAC

Address may conflict (with n be moved by the auxiliary tool to modify IP and MAC address) Controller Link Code: char * ip_list [4] = { "192.168.0.11", "192.168.0.12", "192.168.0.13", "192.168.0.14 "};

for (int i = 0; i <4; i ++)

{

ZAux_OpenEth (ip_list [i], & g_card [i]);

```
}
```

Chapter twoBasic Features

This chapter describes the basic motor function using the controller, and provides related routines, reference details PC Corresponding library routines.

2.1 Controller Link

Table 2.1-correlation function controller is connected

Functi	Featur
ZAux_OpenCom	Serial Link Controller
ZAux_SetComDefaultBaud	Serial communication parameters
ZAux_OpenEth	Ethernet Link Controller
ZAux_SearchEthlist	Under the current search
ZAux_Close	Turn off the controller link

You must call the link function links the controller when the controller to operate successfully only when the link corresponding to the operation by the handle controller returned. To call when you close the application unlink function to releaselink.

Routine 2.1.1: Serial Link Controller

```
ZMC_HANDLEg_handle = NULL; //
        Link returns a handle uint32dwBaudRate =
38400; //        38400 baud rate
uint32dwByteSize = 8; // 8   Data bits
uint32dwParity =        0; //        No parity
uint32dwStopBits = 1; // 1   Stop bits
uint32comid = 1;//  Links COM port
ZAux_SetComDefaultBaud (dwBaudRate, dwByteSize, dwParity, dwStopBits); // set the serial communication parameters
ZAux_OpenCom (comid, & g_handle); // COM1   Interface link controller
```

Routine 2.1.2: network interface controller link

```
ZMC_HANDLEg_handle = NULL; //   Link
returns a handle char * ipaddr = "192.168.0.11";
                // controller IP ZAux_OpenEth (ipaddr, &
g_handle);      // Ethernet Link Controller
```

⚠ Detection function return value if the function succeeds, the

return 0 if successful, non-zero failed. Ethernet port link to return unsuccessful, check the transfer of IP is correct, and make sure your computer's IP address and the IP address of the controller on the same network segment.

2.2 The basic parameter initialization shaft

The basic parameters of the function shaft
Table 2.2

Functi	Featur
ZAux_Direct_SetAtype	Set axis type
ZAux_Direct_SetUnits	Setting stem pulse equivalent
ZAux_Direct_SetInvertStep	Set pulse output mode
ZAux_Direct_SetSpeed	Set shaft speed
ZAux_Direct_SetAccel	Set-axis acceleration
ZAux_Direct_SetDecel	Setting the deceleration shaft
ZAux_Direct_SetSramp	S-curve time setting stem

2.2 routines: initialization parameters

ZAux_Direct_SetAtype (g_handle, 0,1); // Set pulse output shaft axis 0
ZAux_Direct_ SetUnits (G_handle, 0,100); // Set parameters to the shaft axis 0 100 units of pulses ZAux_Direct_ SetInvertStep (g_handle, 0,1);
// set the direction of the axis 0 + pulse mode ZAux_Direct_ SetSpeed (G_handle, 0,200); // 0 Set speed shaft 200units / s ZAux_Direct_ SetAccel (g_handle, 0,2000); // set acceleration ZAux_Direct_ SetDecel axis 0 of 2000units / s / s (g_handle, 0,2000);
// set acceleration axis 0 of 2000units / s / s
ZAux_Direct_ SetSramp (G_handle, 0,0); // 0S time axis setting the curve (0-trapezoidal acceleration and deceleration)

Highlights 2.2:

Shaft Type: 1-type common pulse output type (stepper / servo), 3- encoder input type (external encoder / encoder shaft itself), the shaft is configured corresponding to the type of the actual hardware. Virtual axis is an axis type internal analog program run, typically with the robot shaft for superimposing the case.

Atype type	desc
0	Virtual axis;
1	Stepper or servo mode pulse direction
2	Servo control of the analog signal
3	Quadrature encoder
4	Stepper + Encoder
6	Direction encoder pulse mode, can be used
7	Pulse direction stepper or servo mode
8	ZCAN extended pulse direction with stepper
9	ZCAN extended orthogonal encoder.
10	ZCAN direction of extension encoder pulse
65	ECAT type of pulse

66	ECAT speed loop
67	ECAT closed-loop torque
70	ECAT custom action, only read the encoder.

FIG 2.2-1 axis type

When the pulsethe amount: Pulse equivalent to the basic unit vector axis parameters, such as speed, acceleration, position .... When the set units. 1, unit speed pulse / S, when the mechanism is set to mobile units 1mm desired number of pulses, the speed unit is mm./s. ⚠ Pulse equivalent must be modified, and then modify the speed, displacement parameters. Otherwise, when units change will lead to changes in speed, acceleration, location.

Pulse mode:Pulse plus direction mode 0-3, 4-7 for the double-pulse mode.

mol d	Forward		Negative	
	Pulse	Direction	Pulse	Direction
0		High		Low
1		High		Low
2		Low		High
3		Low		
4				
5	High			High
6		Low	Low	
7	Low			Low

2.2-2 pulse output mode of FIG.

2.3 Special IO settings

Table 2.3 Special IO functions

Functi	Featur
ZAux_Direct_SetDatumIn	Axis origin signal is provided
ZAux_Direct_SetFwdIn	Axis positive limit signal
ZAux_Direct_SetRevIn	Axis negative limit signal
ZAux_Direct_SetAlmIn	Spindle servo alarm signal is
ZAux_Direct_SetInvertIn	Set input signal inverted state

N dedicated motion controller is not particular IO port, a special IO are designated assigned to the respective input and output by the general-purpose command, the special state when the IO configuration -1 is a state corresponding to no.

Routine 2.3: Special IO Configuration

ZAux_Direct_SetDatumIn (g_handle, 0,1); // Set IN (1) of the axis origin signal 0
ZAux_Direct_SetFwdIn (G_handle, 0,2); // Set IN (2) of axis positive limit 0
ZAux_Direct_ SetRevIn (G_handle, 0,1); // setPut IN (1) axis 0 The negative limit, a channel with the same originnumber

ZAux_Direct_SetAlmIn (g_handle, 0, -1); // Cancel 0 axis servo configuration alert signal
ZAux_Direct_SetInvertIn (g_handle, 1,1); // Reverse IN (1) active high

Highlights 2.3:

Special IO ZMC series are considered valid when the OFF state, (i.e., corresponding to the positive input limit OFF state, corresponding to that encountered axis positive limit). Therefore, when the IO port for a particular type of sensor is normally open, corresponding to the input port needs to be inverted to the active high, normally closed by default it is active low.

ECI series of controllers with special signal ZMC series controller status is the opposite!

2.4 Single-axis motion

2.4.1 Uniaxial back to zero

Table 2.4-1 homing motion functions

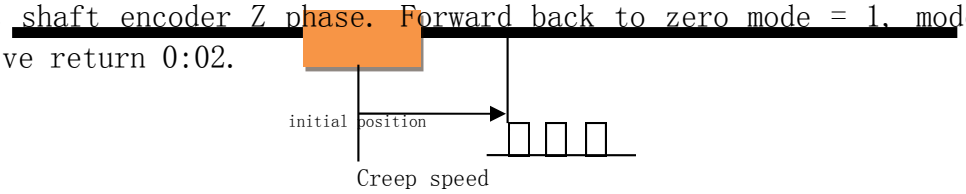
Functi	Featur
ZAux_Direct_Singl_Datum	Single-axis homing movement
ZAux_Direct_SetCreep	Set secondary creep speed back

2.4.1 Routine: Single back to zero

ZAux_Direct_SetDatumIn (g_handle, 0,1); // Set IN (0) axis origin of a signal ZAux_Direct_ SetSpeed 0 (G_handle, 0,100); // 0 Set speed shaft 200units / s ZAux_Direct_ SetCreep (g_handle, 0,10); // 0 creep speed of the shaft is provided 10units / s ZAux_Direct_Singl_Datum (g_handle, 0,4); // 4-axis mode 0 according to the reset to zero, to zero to the secondary negative feedback.

The motion controller provides several positive zero return, pass through zero return value selecting different modes direction corresponding manner.

Mode 1: Z-phase mode, the axis Z CREEP speed operation until the signal appears. DPOS correction value is automatically reset to 0 MPOS simultaneously. Only ATYPE to 7, and the corresponding valid access shaft encoder Z phase. Forward back to zero mode = 1, mode = negative return 0:02.



F

igure 2.4-1 Z-phase The first Z-phase signal mode back to zero

Option 2:+ Trans origin find mode, the origin axis SPEED speed running, until it hits the home switch. Then CREEP shaft speed reverse movement until the switch from the origin. DPOS value is automatically reset to 0 while correcting MPOS, mode = 3 in the forward direction back to zero, when the 4 mode = negative back to zero.

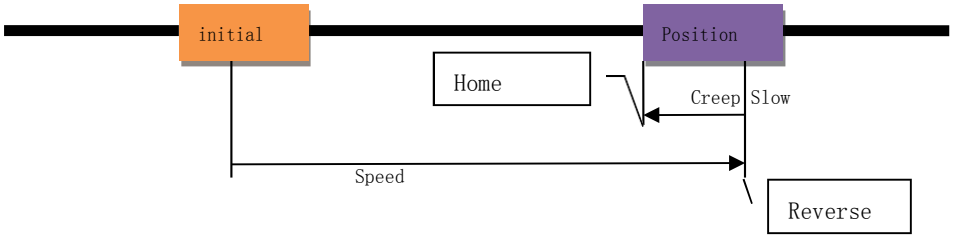


Figure 2.4-2 + Anti-origin back to zero

Mode 3:+ Z + trans find the origin signal pattern, the origin axis SPEED speed running, until it hits the home switch. Then CREEP shaft speed reverse movement until the switch from the origin, and then continue to creep speed reverse signal Z until it hits. DPOS correction value is automatically reset to 0 MPOS simultaneously. Only ATYPE to 7, and the corresponding valid access shaft encoder Z phase. mode = 5 forward back to zero, mode = 6 negative to zero.

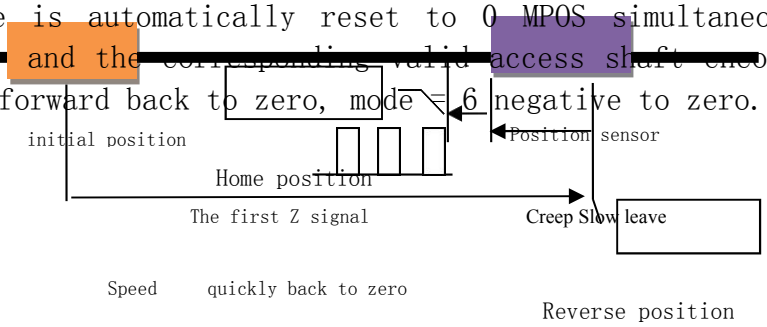


Figure 2.4-3 to find the origin of the anti-+ + Z phase back to zero

Mode 4: The origin of a return to zero mode, the axis SPEED speed to run until it hits the home switch. DPOS correction value is automatically reset to 0 MPOS simultaneously. mode = 8 forward back to zero, mode = 9 negative to zero.

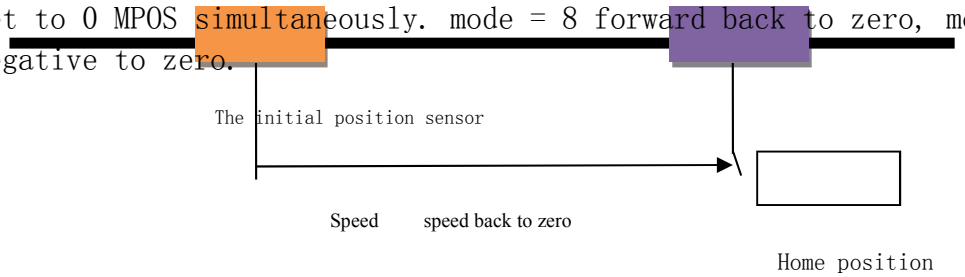


Figure 2.4-4 origin back to zero again

Mode 5: For the case where the origin of the middle positive and negative limit, plus 10 in the respective modes, shown in the process of homing We encountered a limit not cancel exercise, but continue to look for the origin of the reverse.

2.4.2 Uniaxial jog

Table 2.4-2 uniaxial

jog function

Functi	Featur
ZAux_Direct_Singl_Vmove	Single-axis continuous motion
ZAux_Direct_Singl_Move	Single-axis motion relative
ZAux_Direct_Singl _MoveAbs	Single-axis movement absolute
ZAux_ Direct_Singl_Cancel	Stop Motion

Each controller has its own separate shaft axis parameters, and the value of global motion parameters, i.e. speed, Acceleration of a change is effective immediately, the shift function can be achieved online.

Routine 2.4.2 **single-axis motion**

```
intrun_state = 0; //  Axis motion
ZAux_Direct_ SetSpeed (G_handle, 0,200); //  0 Set speed shaft 200units / s
ZAux_Direct_ SetAccel (G_handle, 0,2000); //0 axis is set acceleration
2000units / s / s ZAux_Direct_ SetDecel (G_handle, 0,2000); //  0 axis is
set acceleration 2000units / s / s ZAux_Direct_ SetSramp (G_handle, 0,0);
//                                  Set time curve 0 0S shaft
(trapezoidal acceleration and deceleration) ZAux_Direct_Singl_Move
(g_handle, 0,100); //                0 do shaft relative to the
current position 100 units
{
    ZAux_Direct_GetIfIdle (g_handle, 0, & run_state); // read the motion axis 0, 0 motion, stop -1
}
While (run_state == 0) //  Stop Waiting for axis 0
```

Highlights 2.4.2

When switching the direction of continuous motion can be called directly again **VMOVE direct replacement foregoing continuous motion, does not need to cancel the call.**

When the stop motion transfer cancel, deceleration is used ZAux_Direct_SetFastDec (DEFAULT value equal to the decel) value of the deceleration, fastdec may be set to a larger value when the EMERGENCY STOP.

2.4.3 Uniaxial state

Table 3.4-3 uniaxial state function

Functi	Featur
ZAux_Direct_GetDpos	Axis coordinate reading
ZAux_Direct_GetIfIdle	Axis motion read
ZAux_Direct_GetAxisStatus	Status Read axis

Highlights 2.4.3

Analyzing the axis of motion

Function call ZAux_Direct_GetIfIdle axis motion determines the current state, IDLE state only returns 0, two kinds of values -1, 0 indicates the current axis is in motion, the current axis -1 indicates no motion, motion may be determined by ZAux_Direct_GetMtype whether the current state of the shaft IDLE state to achieve the same effect.

Analyzing the state of the shaft

ZAux_Direct_GetAxisStatus by calling function returns a 32-bit status value, a state value of each state represented by a different axis.

Table 2.4.3-1 axis status

Place	value	significance
1	2	Overrun warning
2	4	Communication with the
4	16	Forward hardware limit
5	32	Negative alarm to the
6	64	To find the origin of
7	128	Velocity hold signal HOLD
8	256	Overrun Error Tracking
9	512	Forward soft limit
10	1024	Than negative soft
11	2048	CANCEL execution
12	4096	Pulse frequency
18	262144	Power anomalies
tw	4194304	Servo alarm signal
tw	8388608	Axis into the

2.5 Multi-axis interpolation motion

Positive motion controller can supportLinear, circular, spiral, space arcs, ellipses and elliptical other helical interpolation movement. And the shaft can be freely combined to make interpolated motion, the controller controls so as to achieve a multi-station.

2.5.1 Common interpolation motion

Table 2.5-1 Common interpolation motion function

Functi	Featur
ZAux_Direct_Base	Motion axis settings list
ZAux_Direct_Move	Multi-axis relative linear
ZAux_Direct_MoveAbs	Absolute multi-axis linear
ZAux_Direct_MoveCirc2	Three two-axis relative to a given
ZAux_Direct_MoveCirc2Abs	Two-axis circular interpolation
ZAux_Direct_MSpherical	Space circular interpolation

Motion interpolation motion parameters are provided on the spindle, that is, ZAux_Direct_Base (ZMC_HANDLE handle, int imaxaxises, int * piAxislist) instruction

list piAxislist axis of the first spindle axis. Spindle speed, acceleration, deceleration, ... synthesis parameters are interpolated motion vector direction of the shaft of participating parameters. Motion state is determined, the state of the spindle can be determined directly.

Routine 2.5.1 three-axis linear interpolation

int axislist [3] = {1,0,2}; // BASE axis motion list, wherein the spindle shaft 1
float poslist [3] = {200, 300}; // Sports list, shaft 100, shaft 0-200, 2-300 shaft

ZAux_Direct_SetSpeed (g_handle, axislist [0], 100); // Interpolation speed setting 100 provided on the spindle ZAux_Direct_SetAccel (g_handle, axislist [0], 1000); // Set interpolation acceleration 1000

ZAux_Direct_Base (g_handle, 3, axislist); // Select the axis list ZAux_Direct_MoveAbs (g_handle, 3, poslist); // call the movement, the axis absolute position corresponding 0,1,2 go

Highlights 2.5.1

Space arc command only relative motion commands, generally corresponding to the axis of movement through the read command ZAux_Direct_GetEndMoveBuffer end position to the absolute coordinate into the relative spatial fairly be called an offset arc command.

2.5.2 Continuous interpolation motion

Table 2.5-2 continuous interpolation function associated

Functi	Feat
ZAux_Direct_SetMerge	Continuous interpolation motion
ZAux_Direct_GetRemain_Buffer	The remaining number of read
ZAux_Direct_SetForceSpeed	SP movement speed
ZAux_Direct_SetStartMoveSpeed	Movement start speed SP
ZAux_Direct_GetEndMoveSpeed	SP movement end speed

ZMotion each axis the motion controller has a multistage buffering motion when the motion instruction currently being executed, the instruction calls back motion will fill the buffer, the buffer will empty automatically the current plane movement has been completed. Avoid blocking competition program. MOVE_DELAY delay can also be used to fill the instruction buffer, so that automatic delay between motion commands, or by inserting MOVE_OP IO operations in motion, when the continuous interpolation MERGE open switch, will automatically same spindle interpolation Games It is continuously up.

The next instruction NTYPE Read type Running command MOVE_CURMARK Read identification MTYPE Read type

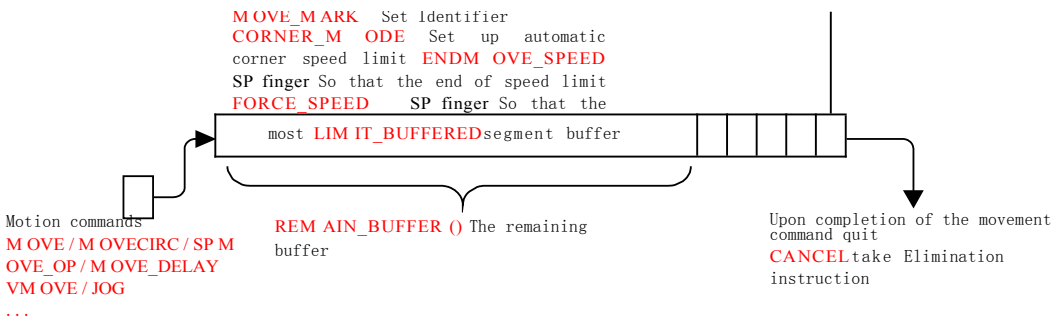


FIG 2.5-2
motion buffer

All the interpolated motion has a corresponding custom speed SP sports, such as movesp, moveabssp ... and so on. Moving speed SP is no longer controlled by the speed of the global velocity, but set the start, run, the speed of the end of the corresponding segment by special speed command, such as movement into the movement speed is buffered together.

2.5.2 Continuous interpolation routine movement

```
int axislist [3] = {0,1,2}; //            BASE axis motion list
float poslist [3] = {0,0,0}; //           Sports list,
int iremain = 0; //        The remaining
buffer

ZAux_Direct_Base (g_handle, 3, axislist); // Select the axis list ZAux_Direct_SetMerge
(g_handle, axislist [0], 1); //                Continuous interpolation switch
ZAux_Direct_SetStartMoveSpeed      (g_handle, axislist [0], 1000); // set a
larger value than the maximum FORCE_SPEED, represents the starting velocity is
not enabled
ZAux_Direct_GetEndMoveSpeed(G_handle, axislist [0], 1000);
for (int i = 0; i <100; i ++)
{
    poslist [0] = i *
    2; poslist [1] = i
    * 3; poslist [2]
    = i * 5; do
    {
    ZAux_Direct_GetRemain_Buffer (g_handle, 0, & iremain); // read the remaining
    buffer
    }
    While (iremain <1)        Wait for the remaining buffer

    ZAux_Direct_SetForceSpeed        (g_handle, axislist [0], i + 10); // Set
    speed ZAux_Direct_MoveAbsSp (g_handle, 3, poslist ); //  Call SP
    Sport
}
```

Highlights 2.5.2

Continuously transmitting motion before interpolation (including **move_op, move_delay instructions) need to first determine whether there is movement of the spindle buffer vacant, then the motion transmission, the transmission buffer is full when the motion to the controller, the commands sent**

returns an error, the motion can not be filled into the buffer. Of course, different motion commands occupy buffer size is inconsistent. ZAux_Direct_GetRemain_Buffer is accounted for by the maximum buffer space arc return instruction. It indicates the number of remaining space can fill the discharge arc command.

2.5.3 Automatic corner parameter settings

Table 2.5-3 corner setters

Functi	Feat
ZAux_Direct_SetCornerMode	Set corner deceleration mode
ZAux_Direct_SetDecelAngle	Setting the deceleration start
ZAux_Direct_SetStopAngle	Set stop angle corner
ZAux_Direct_SetFullSpRadius	Set small round speed limit
ZAux_Direct_SetZsmooth	Set automatic fillet radius

The device produces Qiang Lie vibrations around the corner running at high speed, then generally needs to be done at the turn of the deceleration processing. It can be done by the automatic deceleration processing corresponding to the corner on the controller mode. CornerMode different bit patterns corresponding to different deceleration, the deceleration setting means is not 0 corners.

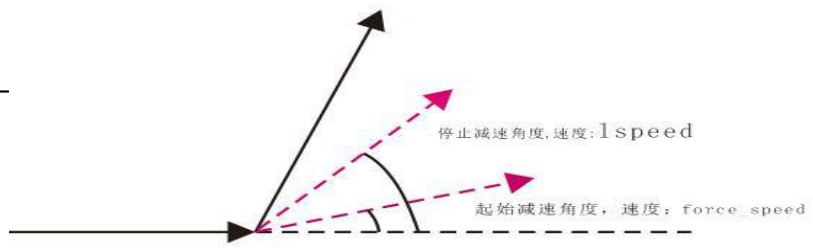
2.5.3 corner deceleration setting routine

```
int axislist [4] = {0,1,2,3}; //            BASE axis motion list
int corner_mode = 2 + 4 + 32;               // corner mode automatically chamfered
corner + small round speed limit + float m_startang = 3.14 * 15/180; //
                                            Corner deceleration start angle
float m_stopang = 90 * 3.14 / 180;          // corner deceleration
stop angle float m_fullradius = 5; //        Small round
radius
floatm_zsmooth                              = // 5   Fillet radius

ZAux_Direct_SetLspeed (g_handle, axislist [0], 0); //        Set the start speed
ZAux_Direct_SetCornerMode (g_handle, axislist [0], corner_mode);  // Set the corner
mode ZAux_Direct_SetDecelAngle (g_handle, axislist [0], m_startang); // set the
deceleration start angle ZAux_Direct_SetStopAngle (g_handle, axislist [0],
m_stopang);                                                  // set the deceleration start
angle ZAux_Direct_SetFullSpRadius (g_handle, axislist [0], m_fullradius); // set
small circle radius ZAux_Direct_SetZsmooth (g_handle, axislist [0], m_zsmooth); //
set chamfer radius
```

Highlights 2.5.3

When automatic corner deceleration, velocity vector direction is decelerated by the size of the corner, the angle from the start of deceleration to stop decelerating the angle between the linearly



decelerating, the deceleration to a stop when the corner angle, speed reduced lspeed set value. Relationship as shown in FIG.

2.5-3 corner

deceleration FIG.

2.6 Hand wheel movement

Table 2.6-1 handwheel motion functions

Functi	Feat
ZAux_Direct_Connect	Synchronization follow
ZAux_Direct_SetClutchRate	Connection rate setting

A handwheel external to the encoder shaft, will occupy an encoder interface controller. Handwheel pulse movementRed scale disposed synchronization follow handwheel movement, when connected by the movement speed is normally set to 0, it indicates / acceleration parameter to track connection according to the speed of the shaft.

2.6 Routine hand wheel movement

```
ZAux_Direct_SetAtype (g_handle, 4,3); //          Hand axle shaft 4 is arranged
ZAux_Direct_SetUnits (g_handle, 4,100); //        Pulse equivalent

int m_naxis = 0; //                               The current pulse axis
float m_radio = 1;//                             Magnification
ZAux_Direct_SetClutchRate (g_handle, m_naxis, 0); //  Set connection rate
ZAux_Direct_Connect (g_handle, m_radio, 4, m_naxis); // handwheel
synchronized movement, the shaft 4 following axis 0 motion
```

Highlights 2.6

Postamble connecting two shafts, with the need to cancel cancel instruction to cancel the current synchronized movement of the shaft, then call other sports, when the synchronization ratio can call handover to replace the preceding CONNECT CONNECT command.

2.7 IO and AD / DA setting the read

Table 2.7-1 input and output functions

Functi	Features
ZAux_Direct_GetIn	Reading the state
ZAux_Direct_SetOp	Providing a single
ZAux_Direct_GetOp	A single output
ZAux_Direct_GetAD	Read a single
ZAux_Direct_SetDA	Providing a single
ZAux_Direct_GetDA	Read a single

Above it is for a single input and output ports operate during

actual operation to improve efficiency may be simultaneously with a plurality of IO read or set, the operation may be directly continuous operation corresponding to a plurality of IO registers MODBUS, or AD, DA.

Routine 2.7

```
uint32 iostatus = 0; ZAux_Direct_SetOp (g_handle,
0,1); //                                Open
output 0
ZAux_Direct_GetIn (g_handle, 1, & iostatus); // Read Input Status 1
ZAux_Direct_SetDA (g_handle, 0, 2048); // Output Set da da (0) =
2047, 5V uint8 iostate1 [2] = {0};
ZAux_Modbus_Get0x (g_handle, 10000,16, & iostate1 [0]); //      Read in (0-15) state,
wherein
```

iostate1 [0] represents 0-7, iostate1 [1] indicates the status of 8-15.

Highlights 2.7

Input and output, there is a corresponding analog Operation modbus registers for input and output may operate directly modbus register.

Modbus_bit (10000) above the mapping in (0) above, namely 0x (10000) - in (0) ... Modbus_bit (20000) above mapping out (0) above, namely 0x (20000) - out (0) ... Modbus_reg (13000) above mapping DA (0) above, i.e., 4x (13000) - above ad (0) Modbus_reg (14000) mapping AD (0) above, i.e., 4x (14000) - da (0)

2.8PC Interact with the data controller

Table 2.8-1 data exchange related functions

Functi	Featur
ZAux_Modbus_Set0x	Modbus register bit set
ZAux_Modbus_Set4x	Word register set modbus
ZAux_Direct_SetTable	System register set table
ZAux_Direct_SetVrf	Set VR register
ZAux_FlashWritef	Write controller FLASH space
ZAux_Direct_GetVariablef	Reading on the controller global
ZAux_Direct_GetVariableInt	Reading the global variable
For more details refer to	

Routine 2.8

```
float array [2];
array [0] = 10.5;

array [1] = 12.14;          ZAux_Modbus_Set4x_Float
(g_handle, 20,2, & array [0]);
//Send data modification        modbus_ieee (20), modbus_ieee (22) value
CString strdata1;
strdata1.Format ( "data1 =% .2f", array [0]);
```

// string command mode, press the command BASIC commands sent down format. May operate a global function or global variable or parameter, modify the value of the global variable control of data1

ZAux_DirectCommand (g_handle, strdata1, NULL, 0);

Highlights 2.8

When operating modbus_ieee modbus_long and will occupy two registers 4X, attention spaced address.

by **ZAux_DirectCommand or ZAux_Execute access variables on the controller or modification**

Basic program can only access controller inside the SUB SUB when global variables or global function.

third chapterDLL function list

3.1 Controller Operation Function Descriptions

3.1.1.1 ZAux_OpenCom

Function: Serial link controller.
prototype: int32 stdcallZAux_OpenCom (uint32 comid, ZMC_HANDLE * phandle)
Input parameters:
Comid Links COM port number Output parameters:
phandle Link To handle
the return value: error code

3.1.1.2 ZAux_SearchAndOpenCom

Function: Automatic Search and serial link controller.
prototype:int32 stdcall ZAux_SearchAndOpenCom (uint32 uimincomidfind, uint32 uimaxcomidfind, uint * pcomid, uint32 uims, ZMC_HANDLE * phandle)
Input parameters:
uimincomidfind Search minimum COM port number
uimaxcomidfind The maximum number of search COM port
uims MS response time link
Output parameters:
Pcomid COM port number to search
phandle Link returns a handle
return value:error code

3.1.1.3 ZAux_SetComDefaultBaud

Function: Set serial communication parameters
prototype: int32 stdcall ZAux_SetComDefaultBaud (uint32 dwBaudRate, uint32 dwByteSize, uint32 dwParity, uint32dwStopBits)
Input parameters:
dwBaudRate The default baud rate: 38400 dwByteSize The

default size of the data bits: 8 dwParity
Check digit Default:
0 no parity dwStopBits Stop
bits default :1 Place

loseThe
parameters of:
Return
Value:error
code

3.1.1.4 ZAux_SetIp

Function: Set the controller IP address
prototype:int32 stdcall ZAux_SetIp (ZMC_HANDLE handle, char * ipaddress)
Input
parameters:
handle Link Handle
ipaddress IP address
values of the output parameters:
return value:error code

3.1.1.5 ZAux_OpenEth

Function: Ethernet Link Controller
prototype:int32 stdcall ZAux_OpenEth (char * ipaddr, ZMC_HANDLE * phandle)
Input
parameters:
ipaddr The IP
address of the link output parameters:
Phandle Links returned
handle the return value: error code

3.1.1.6 ZAux_SearchEthlist

IP addresses under the current search segment: Function Function
prototype:int32 stdcall ZAux_SearchEthlist (char * ipaddrlist, uint32 addrbufflength, uint32 uims)
Input
parameters:
addrbufflength The search returns the IP address of the total length
uims Search timeout
Output
parameters:
ipaddrlist Search the IP

address to return value: error code

3.1.1.7 ZAux_SearchEth

Function: to search the current IP address is the network segment

prototype: int32 stdcall ZAux_SearchEth (const char * Ipaddress, uint32 uims)

Input parameters:

ipaddress IP address search judgments

uims Search timeout

Output Parameters:

Return Value:error code

3.1.1.8 ZAux_Close

Function: Turn off the controller link

prototype:int32 stdcall ZAux_Close (ZMC_HANDLEhandle)

Input parameters:

handle Link Handle

Output Parameters:

Return Value:error code

3.1.1.9 ZAux_Pause

Function: Pause internal operation of the controller of the BASIC program prototype:int32 stdcall ZAux_Pause (ZMC_HANDLE handle) Input parameters:

handle Link Handle

Output Parameters:

Return Value:error code

3.1.1.10 ZAux_Resume

Function Function: BASIC prototype program continues to run internal controller:int32 stdcall ZAux_Resume (ZMC_HANDLE handle) Input parameters:

handle Link Handle

Output Parameters:

Return Value:error code

3.1.1.11 ZAux_BasDown

Function: single BAS file generated and downloaded to the controller running ZAR

prototype: int32 stdcall ZAux_BasDown (ZMC_HANDLE handle, const char * Filename, uint32 run_mode)

Input parameters:

handle	Link Handle
FilenameBAS	File name with path
run_mode	Download Mode 0-RAM 1-ROM

Output parameters:

returnReturnvalue:error code

3.1.1.12 ZAux_Execute

Function: string command sent to the controller
prototype: int32 stdcall ZAux_Execute (ZMC_HANDLE handle, const char * pszCommand, char * psResponse, uint32 uiResponseLength)
Input parameters:
handle Link Handle
pszCommand Send the command string
uiResponseLength Returns the character length of the output parameters:
psResponse String
Return Return Value: Error Code

3.1.1.13 ZAux_DirectCommand

Function: string command sent to the controller
prototype: int32 stdcall ZAux_DirectCommand (ZMC_HANDLE handle, const char * pszCommand, char * psResponse, uint32 uiResponseLength)
Input parameters:
handle Link Handle
pszCommand Send the command string
uiResponseLength Returns the character length of the output parameters:
psResponse String
Return Return Value: Error Code

3.2 Movement command function introduced

3.2.1. Assisted instruction

3.2.1.1 ZAux_Direct_Base

Function: Select the BASE axis list, see Software Manual inside

the "BASE" command. Prototype: int32 stdcall ZAux_Direct_Base (ZMC_HANDLE handle, int imaxaxises, int * piAxislist)

Input parameters:

handle Markup
imaxaxises Number of axes
piAxislist Axis
number list Return Value: Error
Code

3.2.1.2 ZAux_Direct _Defpos

Function Function: BASE axis list defines the current position as a new absolute position value, see Software Manual inside the "DEFPOS" command.

prototype:int32 stdcall ZAux_Direct_Defpos (ZMC_HANDLE handle, int imaxaxises, float * pfDposlist)

Input parameters:

handle Markup
imaxaxises The total number of axes modified
pfDposlist
Coordina
te list Return value: error
code

3.2.1.3 ZAux_Direct_MovePause

Functions Function: BASE axis motion suspend, valid only in single- or multi-interpolated motion, when the multi-axis suspended together.

prototype: int32 stdcall ZAux_Direct_MovePause (ZMC_HANDLE handle, int imode)

Input parameters:

handle Markup
imode Pause mode:
0 Pause the current motion.
Pause 1 after the completion of the current movement is ready to execute the next motion command.
2 is completed after the current exercise is ready to execute the next motion command, pause and MARK identified two instructions are not the same. This mode can be used when a plurality of instructions implemented by the operation, after a pause in the entire operation is

completed.

return value:error code

3.2.1.4 ZAux_Direct_MoveResume

Function: When the BASE axis Pause, continues to move.

prototype:int32 stdcall ZAux_Direct_MoveResume (ZMC_HANDLE handle)

loseThe parameters:
handle Markup
return value: error code

3.2.1.5 ZAux_Direct_Rapidstop

Function: list of all axes stop immediately, if the axes involved in the interpolation, the interpolation stopped movement.
prototype: int32 stdcall ZAux_Direct_Rapidstop (ZMC_HANDLE handle, int imode)
Input parameters:
handle Markup
imode mode
0 Cancel the current movement
1 Cancel buffer sports
2 Cancel the current buffer sports and sports

return value:error code

3.2.1.6 ZAux_Direct_CancelAxisList

Function: list of all axes stop immediately, if the axes involved in the interpolation, the interpolation stopped movement.
prototype:int32 stdcall ZAux_Direct_CancelAxisList (ZMC_HANDLE handle, int imaxaxises, int * piAxislist, int imode)
Input parameters:
handle Markup
imaxaxises The total number of axes of movement
piAxislist Axis list imode mode
0 cancel the current movement
1 Cancel buffer sports
2 Cancels the current buffer sports and sports
return value:error code

3.2.1.7 ZAux _Direct_Regist

Function: latching position, see Software Manual inside the "REGIST" command. prototype:int32 stdcall ZAux _Direct_Regist (ZMC_HANDLE handle, int imode) Input parameters:

handle	Markup
imode	Latch mode

1 When the absolute position of the rising edge of the pulse to the Z REG_POS
2 When the absolute position of the falling edge of the pulse to the Z REG_POS
3 When the absolute position of the rising edge of the input signal R0 to REG_POS
4 When the absolute position of the falling edge of the input signal to the R0 REG_POS
6 input signal Rising edge of the absolute position R0 to REG_POS, absolute position signal to the rising edge Z REG_POSB
7 input signal Rising edge of the absolute position R0 to REG_POS, the absolute position of the falling edge to the Z REG_POSB
8 input signal R0 to the absolute position of the falling edge REG_POS, absolute position signal to the rising edge Z REG_POSB
9 input signal R0 to the absolute position of the falling edge REG_POS, the absolute position of the falling edge to the Z REG_POSB
10 input signal Rising edge of the absolute position R0 to REG_POS, the absolute position of the rising edge of the input signal R1 to REG_POSB
11 R0 absolute position input signal to the rising edge of REG_POS, the absolute position of the edge of the input signal R1 to REG_POSB
12 input signal Absolute position R0 to the falling edge of the REG_POS, the absolute position of the rising edge of the input signal R1 to REG_POSB
13 input signal R0 to the absolute position of the falling edge REG_POS, the absolute position of the falling edge of the input signal to R1 REG_POSB

return value:error code

3.2.1.8 ZAux_Direct_EncoderRatio

Function: Electronic gear ratio setting, refer to the software manual ″ENCODER_RATIO "Instruction Prototype:. Int32 stdcall ZAux_Direct_EncoderRatio (ZMC_HANDLE handle ,, int imposcount, int inputcount)

Input parameters:

handle Markup
imposcount molecular
inputcount
Deno
minator Return Value:

Error code

3.2.1.9 ZAux_Direct_StepRatio

Function: Set a stepping output gear ratio, the reference software manual ″STEP_RATIO "Instruction Prototype:. Int32 stdcall ZAux_Direct_StepRatio (ZMC_HANDLE handle, int ioutcount, int idposcount)

Input parameters:

handle Markup

ioutcount molecular
idposcount
Deno
minator Return Value:
Error code

3.2.2. Multi-axis linear interpolation

The multi-axis interpolation is a motion in front of the current motion ″ZAux_Direct_Base″ function **Axis list to select the corresponding base of the axes involved, the spindle base parameters of a spindle axis interpolated motion, the motion parameters are used for interpolation. Movement divided fairly and absolute motions (ABS suffix), and can specify the current movement of the SP movement. SP Sport can refer to the software manual "* SP" instruction.**

3.2.2.1 ZAux_Direct_Move

:: relative function-linear interpolation, the movement shaft is a listing of previously defined BASE. Prototype: int32 stdcall ZAux_Direct_Move (ZMC_HANDLE handle, int imaxaxises, float * pfDisancelist)
Input parameters:
handle Markup
imaxaxises Axes of Motion
pfDisancelist The return
value from the list of sports: error
code

3.2.2.2 ZAux_Direct_MoveSp

Function: SP relative movement corresponding to linear interpolation instructions, which army Software Manual ″* SP″ instruction.
prototype: int32 stdcall ZAux_Direct_MoveSp (ZMC_HANDLE handle, int imaxaxises, float * pfDisancelist)
Input parameters:
handle Markup
imaxaxises Axes of Motion
pfDisancelist The return
value from the list of sports: error

code

3.2.2.3 ZAux_Direct_MoveAbs

Function: Absolute linear interpolation, a list BASE axis motion axis is defined in advance.
prototype: int32 stdcall ZAux_Direct_MoveAbs (ZMC_HANDLE handle, int imaxaxises, float * pfDisancelist)
Input parameters:

handle Markup
imaxaxises Axes of Motion
pfDisancelist The return value from the list of sports: error code

3.2.2.4 ZAux_Direct_MoveAbsSp

Function: Absolute linear interpolation motion command corresponding to SP, which the army Software Manual "* SP" instruction.
prototype: int32 stdcall ZAux_Direct_MoveAbsSp (ZMC_HANDLE handle, int imaxaxises, float * pfDisancelist)
Input parameters:
handle Markup
imaxaxises Axes of Motion
pfDisancelist The return value from the list of sports: error code

3.2.2.5 ZAux_Direct_MoveModify

Function: to modify the target position on a movement when there is no movement in front of the MOVEABS the same effect, refer to the software manual in the "MOVEMODIFY" command. Use MOVEMODIFY speed will destroy the continuity of the continuous interpolation; MOVEMODIFY while not necessarily a linear interpolation of multiaxial movement.
prototype: int32 stdcall ZAux_Direct_MoveModify (ZMC_HANDLE handle, int imaxaxises, float * pfDisancelist)
Input parameters:
handle Markup
imaxaxises Axes of Motion
pfDisancelist The return value from the list of sports: error code

3.2.3 Arc, ellipse, spiral interpolation

The multi-axis interpolation is a motion in front of the current motion "ZAux_Direct_Base" function **Axis list to select the corresponding base of the axes involved, the spindle base parameters of a spindle axis**

interpolated motion, the motion parameters are used for interpolation. Movement divided fairly and absolute motions (ABS suffix), and can specify the current movement of the SP movement. SP Sport can refer to the software manual "* SP" instruction. And a complete circle arc instruction has three points define two ways and given center circle. 3:00 specifies a point on the circle and intermediate to the end point timing round. Videos may be a full circle or a space by calling the center circle arc corresponding to the selected mode is implemented.

3.2.3.1 ZAux_Direct_MoveCirc

Function Function: BASE The first and second axes circular interpolation, relatively movable manner. See Software Manual inside the "MOVECIRC" command.
prototype: int32 stdcall ZAux_Direct_MoveCirc (ZMC_HANDLE handle, float fend1, float fend2, float fcenter1, float fcenter2, int idirection);
Input parameters:

handle	Markup
fend1	A first coordinate axis motion
fend2	The second motion axis coordinate
fcenter1	The first center axis motion, relative to the starting point
fcenter2	The second center axis motion, relative to the starting point
idirection0-	Counterclockwise,

clockwise 1- Return Value: Error Code

3.2.3.2 ZAux_Direct_MoveCircSp

Function: SP motion instructions corresponding to the relative movement, which army Software Manual "* SP" instruction.
prototype:int32 stdcall ZAux_Direct_MoveCircSp (ZMC_HANDLE handle, float fend1, float fend2, float fcenter1, float fcenter2, int idirection);
Input parameters:

handle	Markup
fend1	A first coordinate axis motion
fend2	The second motion axis coordinate
fcenter1	The first center axis motion, relative to the starting point
fcenter2	The second center axis motion, relative to the starting point
idirection0-	Counterclockwise,

clockwise 1- Return Value: Error Code

3.2.3.3 ZAux_Direct_MoveCircAbs

Function Function: BASE The first and second axes circular interpolation absolute move mode, see inside the software manual "MOVECIRCABS" instruction.
prototype:int32 stdcall ZAux_Direct_MoveCircAbs (ZMC_HANDLE handle, float fend1, float fend2, float fcenter1, float fcenter2, int idirection);

Input parameters:

handle Markup

fend1 The first movement coordinate axis, absolute position

fend2 The second motion coordinate axes, the absolute position

fcentre1 The first movement of the center axis, absolute position

fcentre2 The second movement of the center axis, absolute position

idirection0- Counterclockwise, clockwise 1-

returnReturn value:error code

3.2.3.4 ZAux_Direct_MoveCircAbsSp

Function: Absolute arc motion correspondence **SP motion commands, which join the army Software Manual "* SP" instruction.**
prototype: int32 stdcall ZAux_Direct_MoveCircAbsSp (ZMC_HANDLE handle, float fend1, float fend2, float fcenter1, float fcenter2, int idirection);
Input parameters:

handle Markup
fend1 The first movement coordinate axis, absolute position
fend2 The second motion coordinate axes, the absolute position
fcentre1 The first movement of the center axis, absolute position
fcentre2 The second movement of the center axis, absolute position
idirection0- Counterclockwise, clockwise 1- Return Value: Error Code

3.2.3.5 ZAux_Direct_MoveCirc2

Function: circular interpolation, the first and second shafts BASE circular interpolation, relatively movable manner, see inside the software manual "MOVECIRC2" instruction.
prototype: int32 stdcall ZAux_Direct_MoveCirc2 (ZMC_HANDLE handle, float fmid1, float fmid2, float fend1, float fend2);
Input parameters:

handle Markup
fmid1 A point intermediate the first axis, starting from the opposite
fmid2 The second axis intermediate point, starting from the opposite
fend1 The first shaft end point, starting from the opposite
fend2 The second shaft end points, starting from the relative return value: error code

3.2.3.6 ZAux_Direct_MoveCirc2Sp

Function: SP motion instructions corresponding to the relative

movement, which army Software Manual "* SP" instruction.

prototype:int32 stdcall ZAux_Direct_MoveCirc2Sp (ZMC_HANDLE handle, float fmid1, float fmid2, float fend1, float fend2);

Input parameters:

handle	Markup
fmid1	A point intermediate the first axis, starting from the opposite
fmid2	The second axis intermediate point, starting from the opposite
fend1	The first shaft end point, starting from the opposite
fend2	The second shaft end points, starting from the opposite

returnReturn value:error code

3.2.3.7 ZAux_Direct_MoveCirc2Abs

Function: circular interpolation, the first and second shafts BASE circular interpolation, absolute moves, see inside the software manual "ZAux_MoveCirc2Abs" instruction.
prototype: int32 stdcall ZAux_Direct_MoveCirc2Abs (ZMC_HANDLE handle, float fmid1, float fmid2, float fend1, float fend2);
Input parameters:
handle Markup
fmid1 A first axis intermediate point, the absolute position
fmid2 The second axis intermediate point, the absolute position fend1 The first axis end point, the absolute position fend2 The second axis end point, the absolute position
return value:error code

3.2.3.8 ZAux_Direct_MoveCirc2AbsSp

Function: Absolute arc motion correspondence **SP motion commands, which join the army Software Manual "* SP" instruction.**
prototype:int32 stdcall ZAux_Direct_MoveCirc2AbsSp (ZMC_HANDLE handle, float fmid1, float fmid2, float fend1, float fend2);
Input parameters:
handle Markup
fmid1 A first axis intermediate point, the absolute position
fmid2 The second axis intermediate point, the absolute position fend1 The first axis end point, the absolute position fend2 The second axis end point, the absolute position
return value:error code

3.2.3.9 ZAux_Direct_MHelical

Functions Function: BASE of the first and second axes circular interpolation axis of the third coil, relative to the starting point.

prototype: int32 stdcall ZAux_Direct_MHelical (ZMC_HANDLE handle, float fend1, float fend2, float fcenter1, float fcenter2, int idirection, float fDistance3, int imode);

Input parameters:

handle	Markup
fend1	A first coordinate axis motion
fend2	The second motion axis coordinate
fcentre1	The first center axis motion, relative to the starting point

fcentre2 The second center axis motion, relative to the starting point
idirection0- Counterclockwise, clockwise 1-
fdistance3 The third axis movement distance.
imode Calculating speed of the third shaft:
0 (default) Participation third shaft speed calculation.
1 Third shaft does not participate in the velocity calculation.

return value:error code

3.2.3.10 ZAux_Direct_MHelicalSp

Function: SP motion instructions corresponding to the relative movement, which army Software Manual "* SP" instruction.
prototype:int32 stdcall ZAux_Direct_MHelicalSp (ZMC_HANDLE handle, float fend1, float fend2, float fcenter1, float fcenter2, int idirection, float fDistance3, int imode);
parameter:
handle Markup
fend1 A first coordinate axis motion
fend2 The second motion axis coordinate
fcentre1 The first center axis motion, relative to the starting point
fcentre2 The second center axis motion, relative to the starting point
idirection0- Counterclockwise, clockwise 1-
fdistance3 The third axis movement distance.
imode Calculating speed of the third shaft:
0 (default) Participation third shaft speed calculation.
1 Third shaft does not participate in the velocity calculation.

return value:error code

3.2.3.11 ZAux_Direct_MHelicalAbs

Function Function: BASE The first and second axes circular interpolation, a third helical axis, absolute moves.
prototype:int32 stdcall ZAux_Direct_MHelicalAbs (ZMC_HANDLE handle, float fend1, float fend2, float fcenter1, float fcenter2, int idirection, float fDistance3, int imode);
Input parameters:

handle	Markup
fend1	The first axis motion coordinate, absolute coordinate
fend2	The second axis motion coordinate absolute coordinates
fcentre1	The first center axis motion, absolute coordinates
fcentre2	The second center axis motion, absolute coordinates
idirection0-	Counterclockwise, clockwise 1-
fdistance3	The third axis movement distance. Absolute coordinates

imode Calculating speed of the third shaft:
0 (default) Participation third shaft speed calculation.
1 Third shaft does not participate in the velocity calculation.

return value:error code

3.2.3.12 ZAux_Direct_MHelicalAbsSp

Function: SP motion instructions corresponding to the relative movement, which army Software Manual "* SP" instruction.
prototype:int32 stdcall ZAux_Direct_MHelicalAbsSp (ZMC_HANDLE handle, float fend1, float fend2, float fcenter1, float fcenter2, int idirection, float fDistance3, int imode);
Input parameters:

handle Markup
fend1 The first axis motion coordinate, absolute coordinate
fend2 The second axis motion coordinate absolute coordinates
fcentre1 The first center axis motion, absolute coordinates
fcentre2 The second center axis motion, absolute coordinates
idirection0- Counterclockwise, clockwise 1-
fdistance3 The third axis movement distance. Absolute coordinates
imode Calculating speed of the third shaft:
0 (default) Participation third shaft speed calculation.
1 Third shaft does not participate in the velocity calculation.

return value:error code

3.2.3.13 ZAux_Direct_MHelical2

Functions Function: BASE of the first and second axes circular interpolation axis of the third coil, the relative movement mode.
prototype: int32 stdcall ZAux_Direct_MHelical2 (ZMC_HANDLE handle, float fmid1, float fmid2, float fend1, float fend2, float fDistance3, int imode);
Input parameters:

handle Markup
fmid1 A first intermediate point coordinate axis,

relative to the starting point

fmid2 The second intermediate point coordinate axes, as opposed to the starting point

fend1 The first axis end point coordinates, relative to the starting point

fend2 Second axis end point coordinates, relative to the starting point

fdistance3 The third axis movement distance.

imode Calculating speed of the third shaft:

0 (default) Participation third shaft speed calculation.

1 Third shaft does not participate in the velocity calculation.

return value:error code

3.2.3.14 ZAux_Direct_MHelical2Sp

Function: SP motion instructions corresponding to the relative movement, which army Software Manual "* SP" instruction.
prototype: int32 stdcall ZAux_Direct_MHelical2Sp (ZMC_HANDLE handle, float fmid1, float fmid2, float fend1, float fend2, float fDistance3, int imode);
Input parameters:

handle Markup
fmid1 A first intermediate point coordinate axis, relative to the starting point
fmid2 The second intermediate point coordinate axes, as opposed to the starting point
fend1 The first axis end point coordinates, relative to the starting point
fend2 Second axis end point coordinates, relative to the starting point
fdistance3 The third axis movement distance.
imode Calculating speed of the third shaft:
0 (default) Participation third shaft speed calculation.
1 Third shaft does not participate in the velocity calculation.

return value: error code

3.2.3.15 ZAux_Direct_MHelical2Abs

Functions Function: BASE of the first and second axes circular interpolation axis of the third coil, the absolute moves.
prototype: int32 stdcall ZAux_Direct_MHelical2Abs (ZMC_HANDLE handle, float fmid1, float fmid2, float fend1, float fend2, float fDistance3, int imode);
Input parameters:

handle Markup
fmid1 A first intermediate shaft coordinates, absolute coordinates
fmid2 The second intermediate shaft coordinates, absolute coordinates
fend1 The first axis end point coordinates, absolute coordinates
fend2 The second axis end point coordinates, absolute coordinates
fdistance3 The third axis movement distance.
imode Calculating speed of the third shaft:
0 (default) Participation third shaft speed

calculation.

1 Third shaft does not participate in the velocity calculation.

return value:error code

3.2.3.16 ZAux_Direct_MHelical2AbsSp

Function: Absolute movement corresponding to the SP sports instruction, which the army Software Manual "* SP" instruction.

prototype:int32 stdcall ZAux_Direct_MHelical2AbsSp (ZMC_HANDLE handle, float fmid1, float fmid2, float fend1, float fend2, float fDistance3, int imode); Input parameters:

handle Markup

fmid1 A first intermediate shaft coordinates, absolute coordinates
fmid2 The second intermediate shaft coordinates, absolute coordinates
fend1 The first axis end point coordinates, absolute coordinates
fend2 The second axis end point coordinates, absolute coordinates
fdistance3 The third axis movement distance.
imode Calculating speed of the third shaft:
0 (default) Participation third shaft speed calculation.
1 Third shaft does not participate in the velocity calculation.

return value:error code

3.2.3.17 ZAux_Direct_MEclipse

Function Function: BASE The first and second shafts elliptical interpolation, relatively movable manner, see inside the software manual "MECLIPSE" instruction.

prototype: int32 stdcall ZAux_Direct_MEclipse (ZMC_HANDLE handle, float fend1, float fend2, float fcenter1, float fcenter2, int idirection, float fADis, float fBDis);

Input parameters:

handle Markup
fend1 A first coordinate axis motion
fend2 The second motion axis coordinate
fcentre1 The first center axis motion, relative to the starting point
fcentre2 The second center axis motion, relative to the starting point
idirection0- Counterclockwise, clockwise 1-
fADis Elliptical radius of the first shaft, can be semi-major axis or minor axis half
fBDis Elliptical radius of the second shaft can be semi-major axis or minor axis half, AB Automatically equal arc or spiral.

return value:error code

3.2.3.18 ZAux_Direct_MEclipseSp

Function: SP motion instructions corresponding to the relative

movement, which army Software Manual "* SP" instruction.
prototype:int32 stdcall ZAux_Direct_MEclipseSp (ZMC_HANDLE handle, float fend1, float fend2, float fcenter1, float fcenter2, int idirection, float fADis, float fBDis);

Input parameters:

handle Markup

fend1 A first coordinate axis motion

fend2 The second motion axis coordinate

fcentre1 The first center axis motion, relative to the starting point

fcentre2 The second center axis motion, relative to the starting point

idirection0- Counterclockwise, clockwise 1-

fADis Elliptical radius of the first shaft, can be semi-major axis or minor axis half
fBDis Elliptical radius of the second shaft can be semi-major axis or minor axis half, AB Automatically equal arc or spiral.
return value:error code

3.2.3.19 ZAux_Direct_MEclipseAbs

Function Function: BASE The first and second shafts elliptical interpolation absolute move mode, see inside the software manual "MEclipseAbs" instruction.
prototype:int32 stdcall ZAux_Direct_MEclipseAbs (ZMC_HANDLE handle, float fend1, float fend2, float fcenter1, float fcenter2, int idirection, float fADis, float fBDis);
Input parameters:
handle Markup
fend1 The first coordinate axis movement absolute coordinates
fend2 The second axis motion coordinate absolute coordinates
fcentre1 The first center axis motion, absolute coordinates
fcentre2 The second center axis motion, absolute coordinates
idirection0- Counterclockwise, clockwise 1-
fADis Elliptical radius of the first shaft, can be semi-major axis or minor axis half
fBDis Elliptical radius of the second shaft can be semi-major axis or minor axis half, AB Automatically equal arc or spiral.
return value:error code

3.2.3.20 ZAux_Direct_MEclipseAbsSp

Function: Absolute movement corresponding to the SP sports instruction, which the army Software Manual "* SP" instruction.
prototype:int32 stdcall ZAux_Direct_MEclipseAbsSp (ZMC_HANDLE handle, float fend1, float fend2, float fcenter1, float fcenter2, int idirection, float fADis, float fBDis);
Input parameters:
handle Markup
fend1 The first coordinate axis movement

absolute coordinates

fend2 The second axis motion coordinate absolute coordinates

fcentre1 The first center axis motion, absolute coordinates

fcentre2 The second center axis motion, absolute coordinates

idirection0- Counterclockwise, clockwise 1-

fADis Elliptical radius of the first shaft, can be semi-major axis or minor axis half

fBDis Elliptical radius of the second shaft can be semi-major axis or minor axis half, AB Automatically equal arc or spiral.

return value:error code

3.2.3.21 ZAux_Direct_MEclipseHelical

Function:The first and second shafts BASE elliptical interpolation moves relative to the third axis Synchronization spiral

prototype: int32 stdcall ZAux_Direct_MEclipseHelical (ZMC_HANDLE handle, float fend1, float fend2, float fcenter1, float fcenter2, int idirection, float fADis, float fBDis, float fDistance3);

Input parameters:

handle	Markup
fend1	A first coordinate axis motion
fend2	The second motion axis coordinate
fcentre1	The first center axis motion, relative to the starting point
fcentre2	第二个轴运动圆心，相对与起始点
idirection0-	逆时针，1-顺时针
fADis	第一轴的椭圆半径，半长轴或者半短轴都可
fBDis	第二轴的椭圆半径，半长轴或者半短轴都可，AB 相等时自动为圆弧或螺旋。
fDistance3	第三个轴的运动坐标

返回值：错误码

3.2.3.22 ZAux_Direct_MEclipseHelicalSp

函数功能：相对运动对应 SP 运动指令，参军软件手册里面“*SP”指令。

原 型： int32 stdcall ZAux_Direct_MEclipseHelicalSp(ZMC_HANDLE handle, float fend1, float fend2, float fcenter1, float fcenter2, int idirection, float fADis,
float fBDis, float fDistance3);

输入参数：

handle	链接标识
fend1	第一个轴运动坐标
fend2	第二个轴运动坐标
fcentre1	第一个轴运动圆心，相对与起始点
fcentre2	第二个轴运动圆心，相对与起始点
idirection0-	逆时针，1-顺时针
fADis	第一轴的椭圆半径，半长轴或者半短轴都可
fBDis	第二轴的椭圆半径，半长轴或者半短轴都可，AB 相等时自动为圆弧或螺旋。
fDistance3	第三个轴的运动坐标

返回值：错误码

3.2.3.23 ZAux_Direct_MEclipseHelicalAbs

函数功能：BASE 第一轴和第二轴进行椭圆插补，绝对移动方式，第三个轴

同步螺旋

原 型 ： int32 stdcall ZAux_Direct_MEclipseHelicalAbs(ZMC_HANDLE handle, float fend1, float fend2, float fcenter1, float fcenter2, int idirection, float fADis, float fBDis,float fDistance3);

输入参数：

handle	链接标识
fend1	第一个轴运动坐标，绝对坐标
fend2	第二个轴运动坐标，绝对坐标
fcentre1	第一个轴运动圆心，绝对坐标
fcentre2	第二个轴运动圆心，绝对坐标
idirection0-	逆时针，1-顺时针
fADis	第一轴的椭圆半径，半长轴或者半短轴都可
fBDis	第二轴的椭圆半径，半长轴或者半短轴都可，AB 相等时自动为圆弧或螺旋。
fDistance3	第三个轴的运动坐标

返回值：错误码

3.2.3.24 ZAux_Direct_MEclipseHelicalAbsSp

函数功能：绝对运动对应 SP 运动指令，参军软件手册里面“*SP”指令。

原 型 ： int32 stdcall ZAux_Direct_MEclipseHelicalAbsSp (ZMC_HANDLE handle, float fend1, float fend2, float fcenter1, float fcenter2, int idirection, float fADis, float fBDis, float fDistance3);

Input parameters:

handle	Markup
fend1	The first axis motion coordinate, absolute coordinate
fend2	The second axis motion coordinates, absolute coordinates
fcentre1	The first center axis motion, absolute coordinates
fcentre2	The second center axis motion, absolute coordinates
idirection0-	Counterclockwise, clockwise 1-
fADis	Elliptical radius of the first shaft, can be semi-major axis or minor axis half
fBDis	Elliptical radius of the second shaft can be semi-major axis or minor axis half, AB Automatically equal arc or spiral.
fDistance3	Movement of the third axis coordinate value returns:

Error Code

3.2.3.25 ZAux_Direct_MSpherical

Function: space inside circular interpolation motion relative movement mode, see Software Manual "MSPHERICAL".
prototype: int32 stdcall ZAux_Direct_MSpherical (ZMC_HANDLE handle, float

fend1, float fend2, float fend3, float fcenter1, float fcenter2, float fcenter3, int imode, float fcenter4, float fcenter5)

Input parameters:

handle Markup

fend1 A first motion axis distance parameter 1

fend2 The second motion axis distance parameter 1

fend3 The third axis motion distance parameter 1

fcentre1 A first motion axis distance parameter 2

fcentre2 The second axis motion distance parameter 2

fcentre3 The third axis motion distance parameter 2 imode Front meaning specified parameters

0 current point, intermediate point, the end point of the arc is three, Parameter 1 is the distance from the end point, the distance parameter 2 Distance of the intermediate point.

1 take the minimum arc, distance from the end of the parameter 1 Away from the center of the distance parameter 2.

2 the current point, intermediate point, end three fixed circle, from From parameters 1 as the end point distance, a distance parameter 2 Distance of the intermediate point.

3 go first smallest arc, and then continue to go full circle, from End at a distance parameter 1, parameter 2 as the center distance distance

Fcentre4 Fourth helical axis distance

Fcentre5 Fifth shaft

helical distance Return Value: Error

Code

3.2.3.26 ZAux_Direct_MSphericalSp

Function: SP motion instructions corresponding to the relative movement, which army Software Manual "* SP" instruction.
prototype:int32 stdcall ZAux_Direct_MSphericalSp (ZMC_HANDLE handle, float fend1, float fend2, float fend3, float fcenter1, float fcenter2, float fcenter3, int imode, float fcenter4, float fcenter5)
Input parameters:

handle Markup
fend1 A first motion axis distance parameter 1
fend2 The second motion axis distance parameter 1
fend3 The third axis motion distance parameter 1
fcentre1 A first motion axis distance parameter 2
fcentre2 The second axis motion distance parameter 2
fcentre3 The third axis motion distance parameter 2 imode Front meaning specified parameters
0 current point, intermediate point, the end point of the arc is three,

Parameter 1 is the distance from the end point, the distance parameter 2
Distance of the intermediate point.

1 take the minimum arc, distance from the end of the parameter 1
Away from the center of the distance parameter 2.

2 the current point, intermediate point, end three fixed circle, from
From parameters 1 as the end point distance, a distance parameter 2
Distance of the intermediate point.

3 go first smallest arc, and then continue to go full circle,
Parameter 1 is the distance from the end point, the distance parameter 2
As the center distance

Fcentre4 Fourth helical axis distance
Fcentre5 Fifth shaft

helical distance Return Value: Error

Code

3.2.3.27 ZAux _Direct_MoveSpiral

Function: involute circular interpolation, relatively movable manner, when starting radius **0 direct diffusion**
0 point of view from the beginning, the army inside the software manual ″MOVESPIRAL "Instruction Prototype: int32 stdcall ZAux_Direct_MoveSpiral (ZMC_HANDLE handle, float centre1, float centre2, float circles, float pitch, float distance3, float distance4); Input parameters:

handle Markup
centre1 The center axis of the first relative distance
centre2 The center of the second shaft relative distance
circles Number of turns to rotate, can be a decimal circle, a negative number indicates clockwise

pitch The diffusion distance per lap, can be negative.

distance3 Third shaft spiral function, the relative distance specified third axis, this axis does not participate in the velocity calculation.

distance4 Spiral shaft 4 functions specify the relative distance between the first axis 4, this axis does not participate in the velocity calculation.

return value:error code

3.2.3.28 ZAux _Direct_MoveSpiralSp

Function: SP motion instructions corresponding to the relative movement, which army Software Manual "* SP" instruction.

prototype:int32 stdcall ZAux_Direct_MoveSpiralSSp (ZMC_HANDLE handle, float centre1, float centre2, float circles, float pitch, float distance3, float distance4);

Input parameters:

handle Markup

centre1 The center axis of the first relative distance

centre2 The center of the second shaft relative distance

circles Number of turns to rotate, can be a decimal circle, a negative number indicates clockwise

pitch The diffusion distance per lap, can be negative.

distance3 Third shaft spiral function, the relative distance specified third axis, this axis does not participate in the velocity calculation.

distance4 Spiral shaft 4 functions specify the relative distance between the first axis 4, this axis does not participate in the velocity calculation.

return value:error code

3.2.4 Special motion commands

3.2.4.1 ZAux_Direct_ MoveLimit

Function: the increase in the current movement position of the end of the speed limit, for forcibly corner deceleration, see inside the software manual "MOVELIMIT" instruction.

prototype: int32 stdcall ZAux_Direct_MoveLimit (ZMC_HANDLE handle, float limitspeed);

Input parameters:

handle Markup

limitspeed To limit the speed Return Value: Error Code

3.2.4.2 ZAux_Direct_MoveOp

Function: The command output is written to buffer movement inside, see Software Manual inside the "MOVE_OP" command.

prototype: int32 stdcall ZAux_Direct_MoveOp (ZMC_HANDLE handle, int ioutnum, int ivalue);

Input parameters:

handle Markup

ioutnumIO Numbering

ivalueIO status

return value:error code

3.2.4.3 ZAux_Direct_MoveOpMulti

Function: The command output is written to buffer movement inside, see Software Manual inside the "MOVE_OP" command.
prototype:int32 stdcall ZAux_Direct_MoveOpMulti (ZMC_HANDLE handle, int ioutnumfirst, int ioutnumend, int ivalue);
Input parameters:
handle Markup
ioutnumfirst A first output channel
to be operated ioutnumend The last output
channel to operate ivalueIO status
return value:error code

3.2.4.4 ZAux_Direct_MoveOp2

Function: The command output is written to buffer movement inside, see Software Manual inside the "MOVE_OP2" command.
prototype: int32 stdcall ZAux_Direct_MoveOp2 (ZMC_HANDLE handle, int ioutnum, int ivalue, int iofftimems);
Input parameters:
handle Markup
ioutnumIO Numbering
ivalueIO status
iofftimemsms After inversion time, to generate
a pulse output results Return value: Error Code

3.2.4.5 ZAux_Direct_MoveAout

Function: the analog output motion command written to the buffer inside, see Software Manual inside "MOVE_AOUT" command.
prototype: int32 stdcall ZAux_Direct_MoveAout (ZMC_HANDLE handle, int ioutnum, float fvalue);
Input parameters:
handle Markup
ioutnumAOUT Numbering
fvalue Output analog
value Return value: Error Code

3.2.4.6 ZAux_Direct_MoveDelay

Function: the delay instruction buffer is written to the movement inside, see Software Manual inside the "MOVE_DELAY"
prototype: int32 stdcall ZAux_Direct_MoveDelay (ZMC_HANDLE handle, int itimems);
Input parameters:
handle Markup
itimems Time
delay return value: error code

3.2.5 Synchronous motion commands

3.2.5.1 ZAux_ Direct_Cam

Function: electronic cam, CAM command according to the data stored in the TABLE to determine the shaft Sports, see Software Manual inside the "CAM" command.
prototype:int32 stdcall ZAux_Direct_Cam (ZMC_HANDLE handle, int istartpoint, int iendpoint, float ftablemulti, float fDistance);
Input parameters:
handle Markup
istartpoint TABLE numbered starting position, the first point is stored
iendpoint TABLE end point number
ftablemulti This ratio is multiplied by a position, a pulse is generally equivalent.
fDistance Reference movement distance, to calculate the total exercise time. Return value: error code

3.2.5.2 ZAux_Direct_Cambox

Function: electronic cam, CAMBOX instructions to determine the movement of the shaft, see the inside of the software manual "CAMBOX" instruction data stored in the TABLE.
prototype: int32 stdcall ZAux_Direct_Cambox (ZMC_HANDLE handle, int istartpoint, int iendpoint, float ftablemulti, float fDistance, int ilinkaxis, int ioption, float flinkstartpos);
Input parameters:

handle Markup

istartpoint TABLE numbered starting position, the first point is stored

iendpoint TABLE end point number

ftablemulti This ratio is multiplied by a position, a pulse is generally equivalent.

fDistance Reference movement distance, to calculate the total exercise time.

Ilinkaxis Reference axis.

Ioption Connection with the reference axis, different bits represent different significance

When Bit 0 latch signal when the reference axis (the Regist) event is triggered, the current axis motion reference axis make the connection.

When Bit 1 is set to the reference axis motion absolute position, the reference axis current axis connecting motion starts.

Bit 2 auto-repeat continuous two-way operation. (By setting REP_OPTION = 1, can be canceled repeat)

Bit 5 only forward motion of the reference axis is connected only

Flinkstartpos When ioption parameter is set to 2, the parameter indicates the absolute position of the connection started

return value:error code

3.2.5.3 ZAux_Direct_Connpath

Function: electronic gear, the position of the interpolation vector of the target current idringaxis axis length of the shaft are connected by an electronic gear. See Software Manual inside the "CONNPATH" command. prototype:int32 stdcall ZAux_Direct_Connpath (ZMC_HANDLE handle, float ratio, int link_axis, int move_axis);

Input parameters:

handle Markup

fratio Rate, either positive or negative, note the number of pulses proportional

idringaxis Number shaft (spindle) of the connecting shaft

moveaxis No movement

of the shaft axis Return Value:

Error Code

3.2.5.4 ZAux_Direct_Connect

Function: electronic gear, the position of the interpolation vector of the target current idringaxis axis length of the shaft are connected by an electronic gear. See Software Manual inside the "CONNECT" command. prototype:int32 stdcall ZAux_Direct_Connect (ZMC_HANDLE handle, float ratio, int link_axis, int move_axis);

Input parameters:

handle	Markup
fratio	Rate, either positive or negative, note the number of pulses proportional
idringaxis	Number shaft (spindle) of the connecting shaft
moveaxis	No movement of the shaft axis

returnReturn value:error code

3.2.5.5 ZAux_Direct_Movelink

Function: Self camming defined, with acceleration and deceleration of the stage movement can be provided, see Software Manual inside the "MOVELINK" command.

prototype: int32 stdcall ZAux_Direct_Movelink (ZMC_HANDLE handle, float fDistance, float fLinkDis, float fLinkAcc, float fLinkDec, int iLinkaxis, int ioption, float flinkstartpos);

Input parameters:

handle Markup

fDistance Starting from the end connected to the current distance of the axis, by using Family units

fLinkDis Moving throughout the process connection from the forward reference axis, with units units.

fLinkAcc In the acceleration phase of current axis, a forward movement distance of the reference axis, with units units.

fLinkDec In the current deceleration phase axis, a forward movement distance of the reference axis, with units units.

iLinkaxis Reference numbers shaft axis.

Ioption Connection mode options, different representatives of different binary significance.

1 Bit 0 is connected to a precise time reference axis regist start event is triggered.

2 Bit 1, the reference axis of a connection start position absolute arrival.

4 Bit 2, when this is set, MOVELINK automatically and repeatedly performed may be reversed.

Flinkstartpos When the link options parameter is set to 2, the reference axis parameter indicates the position at which the absolute value of the connection start

return value:error code

3.2.5.6 ZAux_Direct_Moveslink

Function: Self camming defined, with acceleration and deceleration of the stage movement can be provided, see Software Manual inside the "MOVESLINK" command.

prototype: int32 stdcall ZAux_Direct_Moveslink (ZMC_HANDLE handle, float fDistance, float fLinkDis, float startsp, float endsp, int iLinkaxis, int ioption, float flinkstartpos)

Input parameters:

handle	Markup
fDistance	Starting from the end connected to the current distance of the axis, by using

Householdunit

fLinkDis Moving throughout the process connection from the forward reference axis, with units units.

startsp When starting the main motion of the shaft speed ratio.

endsp At the end of the movement of the main shaft speed ratio.

iLinkaxis Reference numbers shaft axis.

Ioption Connection mode options, different representatives of different binary significance.

3 Bit 0 is connected to a precise time reference axis regist start event is triggered.

4 Bit 1, the reference axis of a connection start position absolute arrival.

5 Bit 2, when this is set, MOVELINK automatically and repeatedly performed may be reversed.

Flinkstartpos When the link options parameter is set to 2, the reference axis parameter indicates the position at which the absolute value of the connection start

return value:error code

3.2.5.7 ZAux_ Direct_Connframe

Function: robot instructions, see Software Manual inside the "CONNFRAME" command. prototype:int32 stdcall ZAux_Direct_Connframe (ZMC_HANDLE handle, int frame, int tablenum, int imaxaxises, int * piAxislist);

Input parameters:

handle Markup

frame Robot Type 1-scara3-Stacking....

Tablenum TABLE robot parameters stored starting point imaxaxises The number of associated virtual axis

piAxislist Virtual Axis list associated with the return value:

error code

3.2.5.8 ZAux_ Direct_ Connreframe

Function: robot instructions, see Software Manual inside the "CONNREFRAME" command. prototype: int32 stdcall ZAux_Direct_Connreframe (ZMC_HANDLE handle, int frame, int tablenum, int imaxaxises, int * piAxislist);

Input parameters:

handle Markup

handle Markup

frame Robot Type 1-scara3-Stacking....

Tablenum TABLE robot parameters stored starting point imaxaxises The number of associated joint axis

piAxislist A list of associated joint axis Return Value: Error Code

3.2.6 Single-axis motion command

3.2.6.1 ZAux_ Direct_Singl_Addax

Function: sports overlay.
prototype:int32 stdcall ZAux_ Direct_Singl_Addax (ZMC_HANDLE handle, int iaxis, int iaddaxis);
Input parameters:

handle Markup
iaxis Axis number
iaddaxis Axis number superimposed Return Value: Error Code

3.2.6.2 ZAux Direct_Singl_Cancel

Function: uniaxial decelerated to a stop.
prototype: int32 stdcall ZAux Direct_Singl_Cancel (ZMC_HANDLE handle, int iaxis, int imode);
Input parameters:

handle Markup
iaxis Axis number
imode mode
0 Cancel the current movement
1 Cancel buffer sports
2 Cancel the current buffer sports and sports

return value:error code

3.2.6.3 ZAux_Direct_Singl_Vmove

functionFunction: Single Instruction, continuous movement in one direction, see Software Manual inside the "VMOVE" instruction.

prototype: int32 stdcall ZAux_Direct_Singl_Vmove (ZMC_HANDLE handle, int iaxis, int idir);

loseThe parameters:

handle Markup
iaxis Axis number
idir -1- negative direction,
1- Forward Return Value: Error Code

3.2.6.4 ZAux_Direct_Singl_Move

Function: single-axis relative motion.
prototype: int32 stdcall ZAux_Direct_Singl_Move (ZMC_HANDLE handle, int iaxis, float fdistance)
Input parameters:

handle Markup
iaxis Axis number
fdistance distance
return value:error code

3.2.6.5 ZAux_Direct_Singl_MoveAbs

Function: single-axis absolute motion
prototype: int32 stdcall ZAux_Direct_Singl _MoveAbs (ZMC_HANDLE handle, int iaxis, float fdistance);
Input parameters:

handle Markup
iaxis Axis number
fdistance Absolute
distance Return Value: Error
code

3.2.6.6 ZAux_Direct_Singl_Datum

Function: single-axis homing command.
prototype: int32 stdcall ZAux_Direct_Singl_Datum (ZMC_HANDLE handle, int iaxis, int imode);
Input parameters:

handle Markup
iaxis Axis number
imode mode
0 Clear the error status of all axes.
1 CREEP shaft speed forward run
until the DPOS signal value Z is

reset to 0 while correcting MPOS.

2 CREEP axis Z to run until the reverse speed signal appears. DPOS value is reset to 0 while correcting MPOS.

3 runs forward at the shaft speed SPEED, until it hits the home switch. Then CREEP shaft speed reverse movement until the switch from the origin. DPOS value is reset to 0 while correcting MPOS

In reverse operation shaft 4 SPEED speed until hitting the home switch. Then CREEP axis from the origin until the speed of forward motion switch. DPOS value is reset to 0 while correcting MPOS.

5 SPEED axis runs forward at speed, until it hits the home switch. Then CREEP shaft speed reverse movement until the switch from the origin, and then continue to creep speed reverse signal Z until it hits. DPOS value is reset to 0 while correcting MPOS

6 SPEED-axis inverted run at speed until hitting the home switch. Then CREEP axis from the origin until the speed of forward motion switch, and then continuing until the forward creep speed signal Z encountered. DPOS value is reset to 0 while correcting MPOS.

return value:error code

3.3 Introduction state function parameters shaft axis

3.3.1.1 ZAux_Direct_SetParam

Function: Review axis parameter, the parameter corresponding axis by sending the read command basic

prototype: int32 stdcall ZAux_Direct_SetParam (ZMC_HANDLE handle, const char * sParam, int iaxis, float fset);

Input parameters:

handle	Markup
sParam	String name of the parameter
iaxis	Axis number

fset Parameter Value

return value:error code

3.3.1.2 ZAux_Direct_GetParam

Function: read axis parameter

prototype: int32 stdcall ZAux_Direct_GetParam (ZMC_HANDLE handle, const char * sParam, int iaxis, float * pfValue);

loseThe parameters:

handle Markup

sParam String name of the parameter

iaxis Axis number

pfValue Return

value Return value: error code

3.3.2 The basic parameters of the function shaft

3.3.2.1 ZAux_Direct_SetAtype

Function: Set axis type, see inside the software manual "ATYPE" instruction. prototype:int32 stdcall ZAux_Direct_SetAtype (ZMC_HANDLE handle, int iaxis, int iValue);

Input parameters:

handle Markup

iaxis Axis number

fset Types of

Atype type	desc
0	Virtual axis;
1	Stepper or servo mode pulse
2	Servo control of the analog signal
3	Quadrature encoder
4	Stepper + Encoder
6	Direction encoder pulse mode, can be used to input hand wheel
7	Pulse direction stepper or servo
8	ZCAN extended pulse direction with
9	ZCAN extended orthogonal encoder.
10	ZCAN direction of extension encoder

returnReturn value:error code

3.3.2.2 ZAux_Direct_GetAtype

Function: read axis parameter
prototype:int32 stdcall ZAux_Direct_GetAtype (ZMC_HANDLE handle, int iaxis, int * piValue);
Input parameters:
handle Markup
iaxis Axis number
Output parameters:
piValue Return
Value Returns axis type: Error
Code

3.3.2.3 ZAux_Direct_SetInvertStep

Function: Set pulse output mode, see inside the software manual "INVERT_STEP" instruction.
prototype: int32 stdcall ZAux_Direct_SetInvertStep (ZMC_HANDLE handle, int iaxis, int iValue);
parameter:
handle Markup
iaxis Axis number
iValue Default mode type 0

低8位（位0-位7）表示的模式如下。
0-3 脉冲方向模式。
4-7 双脉冲方式（部分控制器版本不支持，详细咨询厂商）
各个模式对应的电平如下：

模式	正向运动		负向运动	
	脉冲线	方向线	脉冲线	方向线
0		High		Low
1		High		Low
2		Low		High
3		Low		High
4		High	High	
5	High			High
6		Low	Low	
7	Low			Low

高8位（位8-位15）表示方向变化保护时间，单位微秒：0-255
如果模式设定不正确，步进马达可能会在换向时丢失一步的位置，当不确定步进马达的设置时，可以设置100微秒左右的保护时间。

return value:error code

3.3.2.4 ZAux_Direct_GetInvertStep

Function: reading pulse output mode
prototype: int32 stdcall ZAux_Direct_GetInvertStep (ZMC_HANDLE handle, int iaxis, int * piValue);
Input parameters:
handle Markup
iaxis Axis number
Input parameters:
piValue Returns the current mode Return value: error code

3.3.2.5 ZAux_Direct_SetUnits

Function: Set pulse equivalent, see inside the software manual "UNITS" instruction. When set 1 is expressed in units of one pulse.
prototype: int32 stdcall ZAux_Direct_SetUnits (ZMC_HANDLE handle, int iaxis, float fValue); Input parameters:
handle Markup
iaxis Axis number
fValue Pulse equivalent set return value: error code

3.3.2.6 ZAux_Direct_GetUnits

Function: read pulse equivalent
prototype: int32 stdcall ZAux_Direct_GetUnits (ZMC_HANDLE handle, int iaxis, float * pfValue);
Input parameters:
handle Markup
iaxis Axis number
Output parameters:
pfValue Axis pulse equivalent return Return Value: Error Code

3.3.2.7 ZAux_Direct_SetAccel

Function: an acceleration unit units / s / s, see Software Manual inside the "ACCEL" instruction.

prototype:int32 stdcall ZAux_Direct_SetAccel (ZMC_HANDLE handle, int iaxis,

float fValue);
Input parameters:
handle Markup
iaxis Axis number
pfValue Acceleration
settings return value: error code

3.3.2.8 ZAux_Direct_GetAccel

Function: read acceleration
prototype:int32 stdcall ZAux_Direct_GetAccel (ZMC_HANDLE handle, int iaxis, float * pfValue);
Input parameters:
handle Markup
iaxis Axis number
Output parameters:
pfValue Acceleration
of the return value is returned:
error code

3.3.2.9 ZAux_Direct_SetDecel

Function: Set the deceleration units units / s / s, see inside the software manual "DECEL" instruction.
prototype:int32 stdcall ZAux_Direct_SetDecel (ZMC_HANDLE handle, int iaxis, float fValue);
Input parameters:
handle Markup
iaxis Axis number
pfValue Deceleration
set return value: error code

3.3.2.10 ZAux_Direct_GetDecel

Function: reading deceleration
prototype: int32 stdcall ZAux_Direct_SetFastDec (ZMC_HANDLE handle, int iaxis, float fValue);
Output parameters:
handle Markup
iaxis Axis number
pfValue Return Value

Returns deceleration: error code

3.3.2.11 ZAux_Direct_SetFastDec

Function: rapid deceleration, the unit is **units / s / s, is automatically adopted in the CANCEL, or abnormally stops, when the value is set to 0 automatically DECEL, see inside the software manual "FASTDEC" instruction.**

prototype: int32 stdcall ZAux_Direct_SetFastDec (ZMC_HANDLE handle, int iaxis, float fValue);

Input parameters:

handle	Markup
iaxis	Axis number
pfValue	Rapid

deceleration set return value: error code

3.3.2.12 ZAux_Direct_GetFastDec

Function: rapid deceleration reading

prototype: int32 stdcall ZAux_Direct_GetFastDec (ZMC_HANDLE handle, int iaxis, float * pfValue);

Input parameters:

handle	Markup
iaxis	Axis number

Output parameters:

pfValue	Rapid

deceleration return Return value: Error Code

3.3.2.13 ZAux_Direct_SetSpeed

Function: an acceleration unit units / s, see Software Manual inside the "SPEED" instruction.

prototype:int32 stdcall ZAux_Direct_SetSpeed (ZMC_HANDLE handle, int iaxis, float fValue);

Input parameters:

handle	Markup
iaxis	Axis number
pfValue	Shaft speed

set return value: error code

3.3.2.14 ZAux_Direct_GetSpeed

Function: read acceleration

prototype:int32 stdcall ZAux_Direct_GetSpeed (ZMC_HANDLE handle, int iaxis,

float * PfValue);
Input parameters:
handle Markup
iaxis Axis number
Output parameters:
pfValue Shaft speed
returns the return value: error code

3.3.2.15 ZAux_Direct_SetCreep

Function: Set 2 crawling back to zero speed in units / s, see Software Manual inside the "CREEP" command.
prototype: int32 stdcall ZAux_Direct_SetCreep (ZMC_HANDLE handle, int iaxis, float fValue);
Input parameters:
handle Markup
iaxis Axis number
pfValue Crawling speed
setting return value: error code

3.3.2.16 ZAux_Direct_GetCreep

Function: read 2 times the speed of crawling back to zero
prototype: int32 stdcall ZAux_Direct_GetCreep (ZMC_HANDLE handle, int iaxis, float * pfValue);
Input parameters:
handle Markup
iaxis Axis number
Output parameters:
pfValue Return creep
speed the return value: error code

3.3.2.17 ZAux_Direct_SetLspeed

Function: Set the start speed in units / s, see inside the software manual "LSPEED" instruction.
prototype: int32 stdcall ZAux_Direct_SetLspeed (ZMC_HANDLE handle, int iaxis, float fValue);
Input parameters:
handle Markup
iaxis Axis number

pfValue	Initial speed setting

returnvalue:error code

3.3.2.18 ZAux_Direct_GetLspeed

Function: read start speed
originaltype:int32 stdcall ZAux_Direct_GetLspeed (ZMC_HANDLE handle, int iaxis, float * pfValue);
Input parameters:
handle Markup
iaxis Axis number
Output parameters:
pfValue Return Value
Returns the initial velocity:
error code

3.3.2.19 ZAux_Direct_SetMerge

Function: Set continuous interpolation switches, software manual see inside "MERGE" command. prototype:int32 stdcall ZAux_Direct_SetMerge (ZMC_HANDLE handle, int iaxis, int iValue);
Input parameters:
handle Markup
iaxis Axis number
iValue Close 0-, 1- open
Return Value: Error Code

3.3.2.20 ZAux_Direct_GetMerge

Function: Continuous interpolation read switch status
prototype:int32 stdcall ZAux_Direct_GetMerge (ZMC_HANDLE handle, int iaxis, int * piValue);
Input parameters:
handle Markup
iaxis Axis number
Output parameters:
piValue Returns status 0 - 1-
Open Close Return Value: Error Code

3.3.2.21 ZAux_Direct_SetSramp

Function: S-curve setting unit ms, 0 indicates when the trapezoidal acceleration and deceleration. See Software Manual inside the "SRAMP" command.

prototype:i__stdcall ZAux_Direct_SetSramp (ZMC_HANDLE handle, int iaxis, nt32 float
fValue);
Input
parameters:
handle Markup
iaxis Axis number
pfValue S curve set
return value: error code

3.3.2.22 ZAux_Direct_GetSramp

Function: read the S-curve provided
prototype:int32 stdcall ZAux_Direct_GetSramp (ZMC_HANDLE handle, int iaxis, float * pfValue);
parameter:
handle Markup
iaxis Axis number
Output parameters:
pfValue S curve value
returned Return Value: Error Code

3.3.2.23 ZAux_Direct_SetDpos

Function: Set the current position of the axis, units units. See Software Manual inside the "DPOS" instruction.
prototype:int32 stdcall ZAux_Direct_SetDpos (ZMC_HANDLE handle, int iaxis, float fValue);
Input parameters:
handle Markup
iaxis Axis number
pfValue Current
axis position value return:
error code

3.3.2.24 ZAux_Direct_GetDpos

Function: Read the current position
prototype:int32 stdcall ZAux_Direct_GetDpos (ZMC_HANDLE handle, int iaxis, float * pfValue);
Input parameters:
handle Markup
iaxis Axis number

Output parameters:

pfValue Returns the current position

returnvalue:error code

3.3.2.25 ZAux_Direct_SetMpos

Function: Set axis measuring position feedback units units. See Software Manual inside the "MPOS"instruction.
prototype:int32 stdcall ZAux_Direct_SetMpos (ZMC_HANDLE handle, int iaxis, float fValue);
Input parameters:
handle Markup
iaxis Axis number
pfValue Current axis position value return:
error code

3.3.2.26 ZAux_Direct_GetMpos

Function: reading the position feedback
prototype:int32 stdcall ZAux_Direct_GetMpos (ZMC_HANDLE handle, int iaxis, float * pfValue);
Input parameters:
handle Markup
iaxis Axis number
Output parameters:
pfValue Returns the current position of the return value: error code

3.3.2.27 ZAux_Direct_GetEndMove

Function: read the current movement of the final position, the unit units, see Software Manual inside the "ENDMOVE" command.
prototype: int32 stdcall ZAux_Direct_GetEndMove (ZMC_HANDLE handle, int iaxis, float * pfValue);
Input parameters:
handle Markup
iaxis Axis number
Output parameters:
pfValue Return coordinate value Return Value:
Error code

3.3.2.28 ZAux_Direct_GetEndMoveBuffer

Function: the final position of the current read buffer and the movement can be used to convert the relative absolute, Units units, see the software manual inside. "ENDMOVE_BUFFER "Instruction Prototype:. Int32 stdcall ZAux_Direct_GetEndMoveBuffer (ZMC_HANDLE handle, int iaxis, float * pfValue)

Input parameters:

handle	Markup
iaxis	Axis number

Output parameters:

pfValue	Return coordinate value

Return Value: Error code

3.3.2.29 ZAux_Direct_SetFsLimit

Function: Set axis positive soft limit, set to a larger value when the cancel soft limit. singleBit units. See Software Manual inside the "FS_LIMIT" command.

prototype: int32 stdcall ZAux_Direct_SetFsLimit (ZMC_HANDLE handle, int iaxis, float fValue);

Input parameters:

handle	Markup
iaxis	Axis number
pfValue	Positive limit value

Return value: error code

3.3.2.30 ZAux_Direct_GetFsLimit

Function: read the positive soft limit value

prototype: int32 stdcall ZAux_Direct_GetFsLimit (ZMC_HANDLE handle, int iaxis, float * pfValue);

Input parameters:

handle	Markup
iaxis	Axis number

Output

parameters:

pfValue Returns the current position of the return value: error code

3.3.2.31 ZAux_Direct_SetRsLimit

Function: Set axis negative soft limit, set to a larger value when the cancel soft limit. single

Bit units. See Software Manual inside the "RS_LIMIT" command.
prototype: int32 stdcall ZAux_Direct_SetRsLimit (ZMC_HANDLE handle, int iaxis, float fValue);
Input parameters:
handle Markup
iaxis Axis number
pfValue Setting the limit value to the negative return value: Error Code

3.3.2.32 ZAux_Direct_GetRsLimit

Function: read the positive soft limit value
prototype: int32 stdcall ZAux_Direct_GetRsLimit (ZMC_HANDLE handle, int iaxis, float * pfValue);
Input parameters:
handle Markup
iaxis Axis number
Output parameters:
pfValue Returns the current position of the return value: error code

3.3.2.33 ZAux_Direct_GetIfIdle

Function: Read current axis motion has ended, see Software Manual inside the "IDLE" command. prototype:int32 stdcall ZAux_Direct_GetIfIdle (ZMC_HANDLE handle, int iaxis, int * piValue);
Input parameters:
handle Markup
iaxis Axis number
Output parameters:
piValue Returns status 0 - not completed, -1-- motion completion Return Value: Error Code

3.3.2.34 ZAux_Direct_GetLoaded

Function: Are there buffer after reading the current movement, which refer to the software manual "LOADED" instruction.
prototype:int32 stdcall ZAux_Direct_GetIfIdle (ZMC_HANDLE handle, int iaxis, int * piValue);
Input parameters:
handle Markup

iaxis	Axis number

Exportparameter:

piValue Returns status 0 - there is a buffer -1 - None Return Value: Error Code

3.3.2.35 ZAux_Direct_GetMspeed

Function powerCan: read the current speed feedback unit **unitsSee Software Manual inside the "MSPEED" command.**

prototype: int32 stdcall ZAux_Direct_GetMspeed (ZMC_HANDLE handle, int iaxis, float * pfValue);

Input parameters:

handle Markup

iaxis Axis number

Output parameters:

pfValue Return Value

Returns speed feedback: error code

3.3.2.36 ZAux_Direct_GetMtype

Function: read the current movement instruction type, see Software Manual inside the "MTYPE" command. prototype:int32 stdcall ZAux_Direct_GetMtype (ZMC_HANDLE handle, int iaxis, int * piValue);

Input parameters:

handle Markup

iaxis Axis number

Output parameters:

piValue Return type Return value: error code

3.3.2.37 ZAux_Direct_GetNtype

Function: read the current movement of the next movement instruction type, see Software Manual inside the "NTYPE" command.

prototype:int32 stdcall ZAux_Direct_GetNtype (ZMC_HANDLE handle, int iaxis, int * piValue);

Input parameters:

handle Markup

iaxis Axis number

Output parameters:

piValue Return type Return value: error code

3.3.2.38 ZAux_Direct_GetRemain

Function: Reads the current remaining movement distance in **units, see Software Manual inside the "REMAIN" command.**
prototype: int32 stdcall ZAux_Direct_GetRemain (ZMC_HANDLE handle, int iaxis, float * pfValue);
Input parameters:
handle Markup
iaxis Axis number
Output parameters:
pfValue Return Value
Returns the remaining distance:
error code

3.3.2.39 ZAux_Direct_VectorBuffered

Function: read the current buffer and the remaining movement distance in Units, see inside the software manual "VECTOR_BUFFERED" instruction.
prototype: int32 stdcall ZAux_Direct_ VectorBuffered (ZMC_HANDLE handle, int iaxis, float * pfValue);
Input parameters:
handle Markup
iaxis Axis number
Output parameters:
pfValue Return Value
Returns the remaining distance:
error code

ZAux_Direct_GetVpSpeed letterNumber Function: read velocity axis, units **units / s,**

See Software Manual inside the "VP_SPEED" command.
prototype: int32 stdcall ZAux_Direct_GetVpSpeed (ZMC_HANDLE handle, int iaxis, float * pfValue);
Input parameters:
handle Markup
iaxis Axis number
Output

parameters:

pfValue Returns the current speed of the return value: error code

3.3.2.40 ZAux_Direct_AxisStatus

Function: read the status of the current axis, "AXISSTATUS" instruction.

prototype: int32 stdcall ZAux_Direct_GetAxisStatus (ZMC_HANDLE handle, int iaxis, int * piValue);

loseThe parameters:
handle Markup
iaxis Axis number
Output parameters:
piValue Return status value, the corresponding bit indicates different states of the return value: error code

3.3.3 Other function parameters

3.3.3.1 ZAux_Direct_GetAddax

Function: Reads the current **No. ADDAX axis command superimposed axis and -1 is not superimposed. See Software Manual inside the "ADDAX_AXIS" command.**
prototype:int32 stdcall ZAux_Direct_GetRevIn (ZMC_HANDLE handle, int iaxis, int * piValue);
Input parameters:
handle Markup
iaxis Axis number
Output parameters:
piValue Returns the number corresponding to the superposition of the shaft, -1-- axis overlay
no return value: error code

3.3.3.2 ZAux_Direct_SetAxisAddress

Function: the shaft extension axis address configuration, see inside the software manual "AXIS_ADDRESS" instruction.
prototype: int32 stdcall ZAux_Direct_SetAxisAddress (ZMC_HANDLE handle, int iaxis, int iValue);
Input parameters:
handle Markup
iaxis Axis number
iValue Address configuration return value:
error code

3.3.3.3 ZAux_Direct_GetAxisAddress

Function: the read address of the current axis of the shaft

prototype:int32 stdcall ZAux_Direct_GetAxisAddress (ZMC_HANDLE handle, int iaxis, int * piValue);

Input parameters:

handle Markup
iaxis Axis number
Output parameters:
piValue Return
Value Returns axis address:
Error Code

3.3.3.4 ZAux_Direct_SetClutchRate

Function: Speed connect connection, ratio / sec; When set to 0, to track connection according shaft speed / acceleration parameter, more suitable handwheel movement (when the speed is high enough may cause to continue moving for some time to end). See Software Manual inside the "CLUTCH_RATE" command.
prototype:int32 stdcall ZAux_Direct_SetClutchRate (ZMC_HANDLE handle, int iaxis, float fValue);
Input parameters:
handle Markup
iaxis Axis number
pfValue Current
axis position value return:
error code

3.3.3.5 ZAux_Direct_GetClutchRate

Function: reading the position feedback
prototype:int32 stdcall ZAux_Direct_GetClutchRate (ZMC_HANDLE handle, int iaxis, float * pfValue);
Input parameters:
handle Markup
iaxis Axis number
Output parameters:
pfValue Returns the
current position of the return
value: error code

3.3.3.6 ZAux_Direct_SetCornerMode

Function: corner deceleration mode settings, see Software Manual inside the "CORNER_MODE" instruction.
prototype:int32 stdcall ZAux_Direct_SetCornerMode (ZMC_HANDLE handle, int iaxis, int iValue);

Input parameters:

handle	Markup
iaxis	Axis number
iValue	Mode setting

mode：不同的位代表不同的意义。

位	值	描述
0	1	预留
1	2	自动拐角减速
2	4	预留
3	8	自动小圆限速

returnReturn value:error code

3.3.3.7 ZAux_Direct_GetCornerMode

Function: Reads the current corner mode.
prototype: int32 stdcall ZAux_Direct_GetCornerMode (ZMC_HANDLE handle, int iaxis, int * piValue);
Input parameters:
handle Markup
iaxis Axis number
Output parameters:
piValue Returns the current corner mode Return Value: Error code

3.3.3.8 ZAux_Direct_SetDecelAngle

Function: Minimum corner deceleration start, see the software manual inside "DECEL_ANGLE" instruction.
prototype:int32 stdcall ZAux_Direct_SetDecelAngle (ZMC_HANDLE handle, int iaxis, float fValue);
Input parameters:
handle Markup
iaxis Axis number
pfValue Deceleration start angle Return Value: Error Code

3.3.3.9 ZAux_Direct_GetDecelAngle

Function: reading the minimum deceleration start corner,
prototype:int32 stdcall ZAux_Direct_GetDecelAngle (ZMC_HANDLE handle, int iaxis, float * pfValue);
Input parameters:

handle Markup

iaxis Axis number
Output parameters:
pfValue Return Value
Returns corner deceleration angle:
error code

3.3.3.10 ZAux_Direct_SetStopAngle

Function: slow down to the lowest minimum corner in radians. See Software Manual inside"STOP_ANGLE" command.
prototype:int32 stdcall ZAux_Direct_SetStopAngle (ZMC_HANDLE handle, int iaxis, float pfValue);
Input parameters:
handle Markup
iaxis Axis number
pfValue Angular stop
deceleration Return Value: Error
Code

3.3.3.11 ZAux_Direct_GetStopAngle

Function: Stop reading angle corner deceleration
prototype:int32 stdcall ZAux_Direct_GetStopAngle (ZMC_HANDLE handle, int iaxis, float * pfValue);
Input parameters:
handle Markup
iaxis Axis number
Output parameters:
pfValue Back stop angle
corner deceleration Return Value: Error
Code

3.3.3.12 ZAux_Direct_SetZsmooth

Function: fillet radius. See Software Manual inside the "ZSMOOTH" command.
prototype: int32 stdcall ZAux_Direct_SetZsmooth (ZMC_HANDLE handle, int iaxis, float pfValue);
Input parameters:
handle Markup
iaxis Axis number

pfValue Fillet radius Return Value: Error Code

3.3.3.13 ZAux_Direct_GetZsmooth

Function: Stop reading angle corner deceleration
prototype: int32 stdcall ZAux_Direct_GetZsmooth (ZMC_HANDLE handle, int iaxis, float * pfValue);
Input parameters:
handle Markup
iaxis Axis number
Output parameters:
pfValue Fillet
radius Return Value: Error
Code

3.3.3.14 ZAux_Direct_SetFullSpRadius

Function: the maximum speed of the small circular arc radius, the unit is **units. See Software Manual inside the "**FULL_SP_RADIUS "command.
prototype: int32 stdcall ZAux_Direct_SetFullSpRadius (ZMC_HANDLE handle, int iaxis, float fValue);
Input parameters:
handle Markup
iaxis Axis number
fValue radius
return value:error code

3.3.3.15 ZAux_Direct_GetFullSpRadius

Function: the maximum reading speed small circular arc radius
prototype:int32 stdcall ZAux_Direct_GetFullSpRadius (ZMC_HANDLE handle, int iaxis, float * fValue);
Input parameters:
handle Markup
iaxis Axis number
Output parameters:
pfValue Return
Value Returns radius value:
Error Code

3.3.3.16 ZAux_Direct_SetStartMoveSpeed

Function: the movement from the starting speed of the defined speed SP, the movement parameter is brought into the buffer units units / s. See Software Manual inside the "STARTMOVE_SPEED" command.

prototype:int32 stdcall ZAux_Direct_SetStartMoveSpeed (ZMC_HANDLE handle, int iaxis, float fValue);

Input parameters:

handle Markup

iaxis Axis number

fValue speed

return value:error code

3.3.3.17 ZAux_Direct_GetStartMoveSpeed

Function: starting reading speed custom moving speed SP

prototype: int32 stdcall ZAux_Direct_GetStartMoveSpeed (ZMC_HANDLE handle, int iaxis, float fValue);

Input parameters:

handle Markup

iaxis Axis number

Output parameters:

pfValue The rate

of return return value: error code

3.3.3.18 ZAux_Direct_SetForceSpeed

Function: SP moving speed from the defined speed, the movement parameter is brought into the buffer, in units units / s. See Software Manual inside the ″FORCE_SPEED "Instruction Prototype: int32 stdcall ZAux_Direct_SetForceSpeed (ZMC_HANDLE handle, int iaxis, float fValue);.

Input parameters:

handle Markup

iaxis Axis number

fValue speed

return value:error code

3.3.3.19 ZAux_Direct_GetForceSpeed

Function: starting reading speed custom moving speed SP

prototype:int32 stdcall ZAux_Direct_GetForceSpeed (ZMC_HANDLE

handle, int iaxis, float fValue);

Input parameters:

handle	Markup
iaxis	Axis number

Output parameters:

pfValue	Return speed

returnvalue:error code

3.3.3.20 ZAux_Direct_SetEndMoveSpeed

Function: SP moving speed of since the end of the defined velocity, the movement parameter is brought into the buffer, in units units / s. See Software Manual inside the ″ENDMOVE_SPEED "Instruction Prototype: int32 stdcall ZAux_Direct_SetEndMoveSpeed (ZMC_HANDLE handle, int iaxis, float fValue);.

Input parameters:

handle	Markup
iaxis	Axis number
fValue	speed

return value:error code

3.3.3.21 ZAux_Direct_GetEndMoveSpeed

Function: the end of the reading speed custom moving speed SP

prototype:int32 stdcall ZAux_Direct_GetEndMoveSpeed (ZMC_HANDLE handle, int iaxis, float fValue);

Input parameters:

handle	Markup
iaxis	Axis number

Output parameters:

pfValue	Return rate of return value: error code

3.3.3.22 ZAux_Direct_SetInterpFactor

Function: When the axis is involved in interpolation rate, the default involved in (1), see the software manual inside. ″INTERP_FACTOR "command.

prototype:int32 stdcall ZAux_Direct_SetInterpFactor (ZMC_HANDLE handle, int iaxis, int iValue);

Input parameters:

handle	Markup
iaxis	Axis number

iValue Mode setting 0 - 1-
participation is not involved in the return
value: error code

3.3.3.23 ZAux_Direct_GetInterpFactor

Function: when the reading speed calculating interpolation axis is involved.

prototype:int32 stdcall ZAux_Direct_GetInterpFactor (ZMC_HANDLE handle, int __
iaxis, int iValue);
Input parameters:
handle Markup
iaxis Axis number
Output parameters:
piValue Mode setting 0 - 1- participation is not involved in the return value: error code

3.3.3.24 ZAux_Direct_GetMark

Function: Returns the latch event is generated, see Software Manual inside the "MARK" command. prototype:int32 stdcall ZAux_Direct_GetMark (ZMC_HANDLE handle, int iaxis, int * piValue);
Input parameters:
handle Markup
iaxis Axis number
Output parameters:
piValue Mode setting 0 - no occurrence -1- occur Return Value: Error Code

3.3.3.25 ZAux_Direct_GetMarkB

Function: Returns the latch event is generated, see Software Manual inside the "MARKB" command.
prototype:int32 stdcall ZAux_Direct_GetMarkB (ZMC_HANDLE handle, int iaxis, int * piValue);
Input parameters:
handle Markup
iaxis Axis number
Output parameters:
piValue Mode setting 0 - no occurrence -1- occur Return Value: Error Code

3.3.3.26 ZAux_Direct_GetRegPos

Function: Returns the latched position measurement feedback (MPOS) units units, software manual see inside "REG_POS" instruction.
prototype: int32 stdcall ZAux_Direct_GetRegPos (ZMC_HANDLE handle, int iaxis, float * pfValue);
Input parameters:

handle Markup

iaxis Axis number
Output parameters:
pfValue Feedback position coordinates Return Value: Error Code

3.3.3.27 ZAux_Direct_GetRegPosB

Function: Returns the latched position measurement feedback (MPOS) units units, software manual see inside "REG_POSB" instruction.
prototype: int32 stdcall ZAux_Direct_GetRegPosB (ZMC_HANDLE handle, int iaxis, float * pfValue);
Input parameters:
handle Markup
iaxis Axis number
Output parameters:
pfValue Feedback position coordinates Return Value: Error Code

3.3.3.28 ZAux_Direct_SetOffpos

Function: All modifications offset coordinates, which will not have an impact movement. When you modifyAfter into, OFFPOS reduced to 0. See Software Manual inside the "OFFPOS" command.
prototype:int32 stdcall ZAux_Direct_SetOffpos (ZMC_HANDLE handle, int iaxis, float fValue);
Input parameters:
handle Markup
iaxis Axis number
pfValue Offset distance Return Value: Error Code

3.3.3.29 ZAux_Direct_GetOffpos

Function: the offset distance read
prototype:int32 stdcall ZAux_Direct_GetOffpos (ZMC_HANDLE handle, int iaxis, float fValue);

Input parameters:

handle Markup

iaxis Axis number

Output parameters:

pfValue Offset distance Return Value: Error Code

3.3.3.30 ZAux_Direct_SetMaxSpeed

Function: the maximum output pulse frequency limit, if it is found over this setting forces, and is provided AXISSTATUS, see inside the software manual "MAX_SPEED" instruction.
prototype: int32 stdcall ZAux_Direct_SetMaxSpeed (ZMC_HANDLE handle, int iaxis, int iValue);
Input parameters:
handle Markup
iaxis Axis number
iValue Settings
return value: error code

3.3.3.31 ZAux_Direct_GetMaxSpeed

Function: Returns the current maximum limit frequency of the pulse output shaft.
prototype: int32 stdcall ZAux_Direct_GetMaxSpeed (ZMC_HANDLE handle, int iaxis, int * piValue);
Input parameters:
handle Markup
iaxis Axis number
Output parameters:
piValue The highest frequency return return value:
error code

3.3.3.32 ZAux_Direct_SetMovemark

Function Function: MARK label movement of the next instruction to be called, will write the label instructions and sports movement with buffer, see Software Manual inside the "MOVE_MARK" command.
prototype: int32 stdcall ZAux_Direct_SetMovemark (ZMC_HANDLE handle, int iaxis, int iValue);
Input parameters:
handle Markup
iaxis Axis number
iValue Settings
return value: error code

3.3.3.33 ZAux_Direct_GetMoveCurmark

Function: Returns the current instruction is moving axis **MOVE_MARK label. See Software Manual inside the "MOVE_CURMARK" command.**

prototype:int32 stdcall ZAux_Direct_GetMoveCurmark (ZMC_HANDLE handle,

int iaxis, int * piValue);
Input parameters:
handle Markup
iaxis Axis number
Output parameters:
piValue Returns the current MARK
return value:error code

3.3.3.34 ZAux_Direct_GetRemain_LineBuffer

Function: the remaining buffer shaft, calculated by straight line segments. See Software Manual inside the "REMAIN_BUFFER" command.
prototype: int32 stdcall ZAux_Direct_GetRemain_LineBuffer (ZMC_HANDLE handle, int iaxis, int * piValue);
Input parameters:
handle Markup
iaxis Axis number
Output parameters:
piValue Returns the number of remaining linear motion put Return Value:
Error Code

3.3.3.35 ZAux_Direct_GetRemain_Buffer

Function: the remaining buffer shaft, according to the most complex spatial arc calculated.
prototype:int32 stdcall ZAux_Direct_GetRemain_Buffer (ZMC_HANDLE handle, int iaxis, int * piValue);
Input parameters:
handle Markup
iaxis Axis number
Output parameters:
piValue Returns the number of remaining arc discharge space Return Value:
Error Code

3.3.4 Special function signal parameter (origin, limit ...)

DATUM_IN, JOG, FWD_IN, REV_IN, FHOLD_IN other special input signals, when input is OFF, the input signal is considered to be the opposite effect can be

reversed by INVERT_IN (except ECI series controller).

3.3.4.1 ZAux_Direct_SetInvertIn

Function: Set the inverting input status, see the software manual inside "INVERT_IN" command. prototype: int32 stdcall ZAux_Direct_SetInvertIn (ZMC_HANDLE handle, int ionum, int bifInvert);
Input parameters:
handle Markup
iaxis Axis number
bifInvert State 0-Normally open 1- NC Return Value: Error Code

3.3.4.2 ZAux_Direct_GetInvertIn

Function: read the status inverting input
prototype: int32 stdcall ZAux_Direct_GetInvertIn (ZMC_HANDLE handle, int iaxis, int iValue);
Input parameters:
handle Markup
iaxis Axis number
Output parameters:
iValue Return Value
Returns status values: Error Code

3.3.4.3 ZAux_Direct_SetAlmIn

Function: Set axis alarm signal -1 is canceled, see inside the software manual "ALM_IN" instruction.
prototype:int32 stdcall ZAux_Direct_SetAlmIn (ZMC_HANDLE handle, int iaxis, int iValue);
Input parameters:
handle Markup
iaxis Axis number
iValue IO port number, -1 is unset
Return Value: Error Code

3.3.4.4 ZAux_Direct_GetAlmIn

Function: the alarm signal corresponding to the read input shaft

prototype:int32 stdcall ZAux_Direct_GetMerge (ZMC_HANDLE handle, int iaxis,

int * PiValue);
Input parameters:
handle Markup
iaxis Axis number
Output parameters:
piValue Returns the number corresponding to the input port, -1-- not set return value: error code

3.3.4.5 ZAux_Direct_SetDatumIn

Function: Set-axis origin signal -1 is canceled, see inside the software manual "DATUM_IN" instruction.
prototype: int32 stdcall ZAux_Direct_SetDatumIn (ZMC_HANDLE handle, int iaxis, int iValue);
Input parameters:
handle Markup
iaxis Axis number
iValue IO port number, -1 is unset
Return Value: Error Code

3.3.4.6 ZAux_Direct_GetDatumIn

Function: the read signal corresponding to the input axis origin
prototype: int32 stdcall ZAux_Direct_GetDatumIn (ZMC_HANDLE handle, int iaxis, int * piValue);
Input parameters:
handle Markup
iaxis Axis number
Output parameters:
piValue Returns the number corresponding to the input port, -1-- not set return value: error code

3.3.4.7 ZAux_Direct_SetFwdIn

functionFunction: Set the forward limit signal -1 is canceled, see Software Manual inside the "**FWD_IN "command.**
prototype:int32 stdcall ZAux_Direct_SetFwdIn (ZMC_HANDLE handle, int iaxis, int iValue);
Input parameters:
handle Markup
iaxis Axis number
iValue IO port number, -1 to unset

returnReturn value:error code

3.3.4.8 ZAux_Direct_GetFwdIn

Function: Limit the read signal corresponding to the positive input port
prototype:int32 stdcall ZAux_Direct_GetFwdIn (ZMC_HANDLE handle, int iaxis, int * piValue);
Input parameters:
handle Markup
iaxis Axis number
Output parameters:
piValue Returns the number corresponding to the input port, -1-- not set return value: error code

3.3.4.9 ZAux_Direct_SetRevIn

functionFunction: Set the forward limit signal -1 is canceled, see Software Manual inside the ″REV_IN "**command.**
prototype:int32 stdcall ZAux_Direct_SetRevIn (ZMC_HANDLE handle, int iaxis, int iValue);
Input parameters:
handle Markup
iaxis Axis number
iValue IO port number, -1 is unset
Return Value: Error Code

3.3.4.10 ZAux_Direct_GetRevIn

Function: Limit the read signal corresponding to the positive input port
prototype:int32 stdcall ZAux_Direct_GetRevIn (ZMC_HANDLE handle, int iaxis, int * piValue);
Input parameters:
handle Markup
iaxis Axis number
Output parameters:
piValue Returns the number corresponding to the input port, -1-- not set return value: error code

3.4 Input and output functions Introduction

3.4.1.1 ZAux_Direct_SetInvertIn

Function: Set the inverting input status, see the software manual inside "INVERT_IN" command. prototype: int32 stdcall ZAux_Direct_SetInvertIn (ZMC_HANDLE handle, int ionum, int bifInvert);
Input parameters:
handle Markup
ionumIO Port number
bifInvert State 0-Normally
open 1- NC Return Value:
Error Code

3.4.1.2 ZAux_Direct_GetInvertIn

Function: read the status inverting input
prototype: int32 stdcall ZAux_Direct_GetInvertIn (ZMC_HANDLE handle, int ionum, int * piValue);
Input parameters:
handle Markup
ionumIO Port number
Output parameters:
piValue Return Value
Returns status values: Error
Code

3.4.1.3 ZAux_Direct_GetIn

Function: reading input status, see Software Manual inside the "IN" instruction. prototype:int32 stdcall ZAux_Direct_GetIn (ZMC_HANDLE handle, int ionum, uint32 * piValue);
Input parameters:
handle Markup
ionumIO Port
number Output parameters:
piValue Return
Value Returns status values:
Error Code

3.4.1.4 ZAux_Direct_SetOp

Function: Open the output port, refer to the software manual inside the "OP" instruction.
prototype:int32 stdcall ZAux_Direct_SetOp (ZMC_HANDLE handle, int ionum, uint32 iValue);
Input parameters:
handle Markup
ionumIO Port number
iValue Set the value 0-shut down 1- Open Return Value:
Error Code

3.4.1.5 ZAux_Direct_GetOp

Function: read the status of the output port, see Software Manual inside the "OP" instructions. prototype:int32 stdcall ZAux_Direct_GetOp (ZMC_HANDLE handle, int ionum, uint32 * piValue);
Input parameters:
handle Markup
ionumIO Port number Output parameters:
piValue Return Value Returns status values:
Error Code

3.4.1.6 ZAux_Direct_GetAD

Function: read the analog input values, which refer to the software manual "AIN" instruction. prototype:int32 stdcall ZAux_Direct_GetAD (ZMC_HANDLE handle, int ionum, float * pfValue);
Input parameters:
handle Markup
ionumIO Port number Output parameters:
pfValue Back analog value Return value: Error Code

3.4.1.7 ZAux_Direct_SetDA

Function: turn on the analog output, see inside the software manual "AOUT" instruction. prototype:int32 stdcall ZAux_Direct_SetDA

(ZMC_HANDLE handle, int ionum, float fValue);

Input parameters:

handle Markup
ionumIO Port number
fValue Settings
return value:error code

3.4.1.8 ZAux_Direct_GetDA

Function: read the analog input values, which refer to the software manual "AOUT" instruction. prototype:int32 stdcall ZAux_Direct_GetDA (ZMC_HANDLE handle, int ionum, float * pfValue);
Input parameters:
handle Markup
ionumIO Port
number Output parameters:
pfValue Returns the
output value provided Return Value:
Error Code

3.4.1.9 ZAux_Direct_SetPwmFreq

Function Function: pwm frequency settings, see Software Manual inside the "PWM_FREQ" command. prototype:int32 stdcall ZAux_Direct_SetPwmFreq (ZMC_HANDLE handle, int ionum, float fValue);
Input parameters:
handle Markup
ionumPWM Port number
fValue Settings
return value:error code

3.4.1.10 ZAux_Direct_GetPwmFreq

Functions Function: pwm frequency read
prototype: int32 stdcall ZAux_Direct_GetPwmFreq (ZMC_HANDLE handle, int ionum, float * pfValue);
Input parameters:
handle Markup
ionumPWM Port
number Output parameters:
pfValue Returns the
output value provided Return Value:
Error Code

3.4.1.11 ZAux_Direct_SetPwmDuty

Function Function: pwm duty cycle setting, see Software Manual inside the "PWM_DUTY" command. prototype: int32 stdcall ZAux_Direct_SetPwmDuty (ZMC_HANDLE handle, int ionum, float fValue);
Input parameters:
handle Markup
ionumPWM Port number
fValue Settings
return value:error code

3.4.1.12 ZAux_Direct_GetPwmDuty

Function Function: pwm duty cycle of reading
prototype: int32 stdcall ZAux_Direct_GetPwmDuty (ZMC_HANDLE handle, int ionum, float * pfValue);
Input parameters:
handle Markup
ionumPWM Port
number Output parameters:
pfValue Returns the
output value provided Return Value:
Error Code

3.4.1.13 ZAux_GetModbusIn

Function: a plurality of input quickly read, returns the value of each bit indicates the state of the input port.
prototype: int32 stdcall ZAux_GetModbusIn (ZMC_HANDLE handle, int ionumfirst, int ionumend, uint8 * pValueList);
Input parameters:
handle Markup
ionumfirst The read start input
ionumend Output end of
the input port of the read parameters:
pValueList Returns the
corresponding input port status return
values: Error Code

3.4.1.14 ZAux_GetModbusOut

Function: a plurality of Fast Read the current output, the return value of each bit indicates the output port state.

original type: int32stdcall ZAux_GetModbusOut (ZMC_HANDLE handle, int
ionumfirst, int ionumend, uint8 * pValueList);
Input parameters:
handle Markup
ionumfirst Outputs the read start
ionumend The output end
of the output port parameters read:
pValueList Returns the
corresponding output port status return
values: Error Code

3.4.1.15 ZAux_GetModbusDpos

Function: Quick read more current DPOS.
prototype: int32 stdcall ZAux_GetModbusDpos (ZMC_HANDLE handle, int imaxaxises, float * pValueList);
Input parameters:
handle Markup
imaxaxises
Outp
ut axes parameters:
pValueList Returns the
corresponding list of coordinate axis
Return Value: Error Code

3.4.1.16 ZAux_GetModbusMpos

Function: Quick read more current MPOS
prototype: int32 stdcall ZAux_GetModbusMpos (ZMC_HANDLE handle, int imaxaxises, float * pValueList);
Input parameters:
handle Markup
imaxaxises
Outp
ut axes parameters:
pValueList Returns the
corresponding list of coordinate axis
Return Value: Error Code

3.4.1.17 ZAux_GetModbusCurSpeed

Function: read more current rapid rate.

prototype:int32 stdcall ZAux_GetModbusCurSpeed (ZMC_HANDLE handle, int imaxaxises, float * pValueList);

Input parameters:

handle	Markup
imaxaxises	Number of axes

loseThe parameters:
pValueList Shaft speed
returns the corresponding list Return
Value:

3.5 Data communication function

3.5.1.1 ZAux_Direct_GetVariablef

Function: float global variables to read, it can be parameters, etc.
prototype:int32 stdcall ZAux_Direct_GetVariablef (ZMC_HANDLE handle, const char * pname, float * pfValue);
Input parameters:
handle Markup
pname Variable or
parameter name output parameters:
pfValue Return
The return value: error
code

3.5.1.2 ZAux_Direct_GetVariableInt

Function: shaping global variable read, etc. can also be a parameter.
prototype: int32 stdcall ZAux_Direct_GetVariableInt (ZMC_HANDLE handle, const char * Pname, int * PiValue);
Input parameters:
handle Markup
pname Variable or
parameter name output parameters:
piValue Return
The return value: error
code

3.5.1.3 ZAux_Direct_SetVrf

Function: write VR
prototype: int32 stdcallZAux_Direct_SetVrf (ZMC_HANDLEhandle, int vrstartnum, int numes, float *
PfValue);
parameter:

handle	Markup
vrstartnum	VR started operation starting number
numes	Write the number
pfValue	Data

List Return value: error code

3.5.1.4 ZAux_Direct_GetVrf

Function: Reading floating point format VR
prototype: int32 stdcall ZAux_Direct_GetVrf (ZMC_HANDLE handle, int vrstartnum, int numes, float * pfValue)
parameter:

handle	Markup
vrstartnum	VR started operation starting number
numes	Read number
pfValue	Data

List Return value: error code

3.5.1.5 ZAux_Direct_GetVrInt

Function: Shaping read VR format
prototype: int32 stdcall ZAux_Direct_GetVrInt (ZMC_HANDLE handle, int vrstartnum, int numes, int * piValue)
Input parameters:

handle	Markup
vrstartnum	VR started operation starting number
numes	Read the number of output parameters:
piValue	Data

List Return value: error code

3.5.1.6 ZAux_Direct_SetTable

Function: Set TABLE array
originaltype:int32 stdcall ZAux_Direct_SetTable (ZMC_HANDLE handle, int tabstart, int numes, float * pfValue)
Input parameters:

handle	Markup
tabstart	TABLE beginning of the operation starting number
numes	Write the number
pfValue	Data

List Return value: error code

3.5.1.7 ZAux_Direct_GetTable

Function: Array read TABLE

prototype: int32 stdcall ZAux_Direct_GetTable (ZMC_HANDLE handle, int

tabstart, int numes, float * pfValue)
Input parameters:
handle Markup
tabstart TABLE beginning of the operation starting number
numes Read the
number of output parameters:
pfValue Data
List Return value: error
code

3.5.1.8 ZAux_Modbus_Set0x

Function: bit register setting modbus
prototype: int32 stdcall ZAux_Modbus_Set0x (ZMC_HANDLE handle, uint16 start, uint16 inum, uint8 * pdata)
Input parameters:
handle Markup
start Operating modbus_bit starting number
inum Write the number
pdata Data List
return value:error code

3.5.1.9 ZAux_Modbus_Get0x

Function: bit register read modbus
prototype: int32 stdcall ZAux_Modbus_Get0x (ZMC_HANDLE handle, uint16 start, uint16 inum, uint8 * pdata)
Input parameters:
handle Markup
start Operating modbus_bit starting number
inum Read number
Output parameters:
pdata Data List
return value:error code

3.5.1.10 ZAux_Modbus_Set4x

Function: Set modbus word register
prototype: int32 stdcall ZAux_Modbus_Set4x (ZMC_HANDLE handle, uint16 start, uint16 inum, uint16 * pdata)
Input parameters:
handle Markup

start	Operating modbus_reg starting number

inum The number of settings
pdata Data List
return value:error code

3.5.1.11 ZAux_Modbus_Get4x

Function: read word register modbus
prototype: int32 stdcall ZAux_Modbus_Get4x (ZMC_HANDLE handle, uint16 start, uint16 inum, uint16 * pdata)
Input parameters:
handle Markup
start Operating modbus_reg starting number
inum The number
of output parameters read:
pdata Data List
return value:error code

3.5.1.12 ZAux_Modbus_Set4x_Float

Function: Set floating point data word to modbus register
prototype: int32 stdcall ZAux_Modbus_Set4x_Float (ZMC_HANDLE handle, uint16 start, uint16 inum, float * pfdata)
Input parameters:
handle Markup
start Operating modbus_ieee starting number
inum Write the number
pfdata Data
List Return value: error
code

3.5.1.13 ZAux_Modbus_Get4x_Float

Function: read the floating point data word register modbus
prototype: int32 stdcall ZAux_Modbus_Get4x_Float (ZMC_HANDLE handle, uint16 start, uint16 inum, float * pfdata)
Input parameters:
handle Markup
start Operating modbus_ieee starting number
inum Read number
Output parameters:

pfdata	Data List

returnvalue:error code

3.5.1.14 ZAux_Modbus_Set4x_Long

Function: Set shaping to modbus data word register
prototype: int32 stdcall ZAux_Modbus_Set4x_Long (ZMC_HANDLE handle, uint16 start, uint16 inum, int32 * pidata)
Input parameters:

handle	Markup
start	Operating modbus_ieee starting number
inum	Write the number
pidata	Data

List Return value: error code

3.5.1.15 ZAux_Modbus_Get4x_Long

Function: integer data read word register modbus
prototype: int32 stdcall ZAux_Modbus_Get4x_Long (ZMC_HANDLE handle, uint16 start, uint16 inum, int32 * pidata)
Input parameters:

handle	Markup
start	Operating modbus_ieee starting number
inum	Read number

Output parameters:

pidata	Data

List Return value: error code

3.5.1.16 ZAux_FlashWritef

Function: write data to floating point FLASH
prototype: int32 stdcall ZAux_FlashWritef (ZMC_HANDLE handle, uint16 uiflashid, uint32 uinumes, float * pfvlue)
Input parameters:

handle	Markup
uiflashidFLASH	Block number
uinumes	Write the number
pfvlue	Data List

return value:error code

3.5.1.17 ZAux_FlashReadf

Function: floating-point data read from the FLASH
prototype: int32 stdcall ZAux_FlashReadf (ZMC_HANDLE handle, uint16 uiflashid, uint32 uibuffnum, float * pfvlue, uint32 * puinumesread)
Input parameters:
handle Markup
uiflashidFLASH Block number
uibuffnum The number
of buffer variable output parameters:
pfvlue Data List
puinumesread The number of
variables to read Return Value: Error Code

3.5.1.18 ZAux_WriteUFile

Function Function: **float variable into a file format list, the controller and the disk file format consistent U**
prototype:int32 stdcall ZAux_WriteUFile (const char * sFilename, float * pVarlist, int inum)
Input parameters:
sFilename file name
pVarlist Data List
inum The number of
return value:error code

3.5.1.19 ZAux_ReadUFile

Function: read float format variable list, and U disk file controller prototype formats consistent:int32 stdcall ZAux_ReadUFile (const char * sFilename, float * pVarlist, int * pinum)
Input parameters:
sFilename file name
Output parameters:
pVarlist Data List
inum The number of
return value:error code

Chapter FourDirect serial control

4.1 Serial Port Command Control Mode

MODBUS protocol controller from the serial default mode, 38400 baud, no parity. You can modify the serial mode SETCOM command: See "ZBASIC Software Manual." SETCOM (baudrate, databits, stopbits, parity, port [, mode] [, variable] [, timeout])
When the mode 15 is configured as a direct mode command, this time directly from the serial input command string (newline).
This model is particularly suitable for embedded serial board to the control of the motion controller.

Example:

The first step: first implementation setcom (38400, 8,1,0,0,15,2) serial port causes the controller to enter commands directly control mode, if the power controller will need to enter this mode, this command may be written to the program file, to download controller ROM.

Step two: through the serial port software and controller interaction:
Note 1: To set the ASCII character mode, do not use HEX mode. Note 2: the end of the line feed character, so that the controller can distinguish the command.