HansRobot

Communication Protocol Interface

Author: LiJuan Liang Date: 2017 NOV

ShenzhenHHan'sHRobot Co.,Ltd

1 Introduction:

Product Name	HansRobot Communication Protocol Interface
Product Manager	Chongchong Liang
Author	LiJuan Liang
Document Submission Date	2017-11-30

2 Edit History:

Version N0.	Date	Author	Modified Content
V2.5.1	2017-11-30	LiJuan Liang	1.Match the 2.5.101.80 version DCS

3 Member:

Name	Responsible Content
Lijuan Liang	Test and maintain HansRobot communication protocol interface

4 Purpose of this document is written:

- ✧ Provide reference for interface preparation and maintenance of R&D personnel
- ✧ Provide interface usage instructions for other users who use the Hansrobot protocol interface

Interface Preview

Interface	Description
Electrify	Power the robot
BlackOut	Robot blackout
StartMaster	Start Master
CloseMaster	Close Master
GrpPowerOn	Robot servo on
GrpPowerOff	Robot servo off
GrpStop	Stop robot
GrpReset	Reset, Used to clear errors
MoveHoming	The robot returns to the origin
MoveJ	Robot moves to the specified angular coordinate position
MoveL	Robot moves straight to the specified space coordinate position
MoveP	Robot moves to the specified space coordinate position
MoveB	Immediately change the end point of the robot's current movement to the specified space coordinate position
MoveBJ	Immediately change the end point of the robot's current movement to the specified angle coordinate position
MoveC	Circular motion
MoveRelL	The robot moves a certain distance from the specified spatial coordinate axis
MoveRelJ	The robot moves a certain distance from the specified angular coordinate axis

ShortJogJ	Angular motion, Fixed distance
ShortJogL	Spatial motion, Fixed distance
LongJogJ	Angular motion, Unfixed distance
LongJogL	Spatial motion, Unfixed distance
SetKinematicCoordinate	Setting tool coordinates
SetUserCoordinate	Setting user coordinates
SetOverride	Set speed ratio
SetToolCoordinateMotion	Set tool coordinate motion
SetSpeedUp	Speed up
SetSpeedDown	Speed down
SetOutIOState	Set output IO state
SetAcsSafeSpaceLimit	Set angle safety range
SetPcsSafeSpaceLimit	Set space safety range
SetSerialDO	Set the serial port output IO state
SetConveyorScale	Set percentage of conveyor belts
SetTrackingSwitch	Set control tracking switch
SetRunningMode	Switch debug mode
SetSimulation	Seting simulative mode
SetPayload	Seting load
SetBaseMountingAngle	Seting up the installation Angle
StartAssistiveMode	Opening zero force teaching mode
CloseAssistiveMode	Close the zero force teaching mode
SetStartTimer	Open the timer
SetCloseTimer	Off timer
ReadAcsActualPos	Get the actual position of the angle coordinate
ReadPcsActualPos	Get the actual position of the space coordinate
ReadRobotPosInfo	Get the robot position information

ReadConveyorValue	Get the value of the conveyor encoder
ReadOverride	Get speed
ReadInIOState	Get the state of input IO
ReadOutIOState	Get the state of output IO
ReadMoveState	Get the motion state of robot
ReadMachineOrigin	Get the mechanical origin
ReadRobotState	Get the state of robot
ReadSerialDI	Get the digital input bit state specified by the serial port
ReadSerialDO	Get the digital output bit state specified by the serial port
ReadSerialAnalog	Get the analog amount of the serial port
HoldScriptFunc	Pause running script
ContinusScriptFunc	Continue running the script

1 Introduction

1.1 Overview

HansRobot communication protocol interface mainly provides HansRobot communication protocol standard. In accordance with content of this agreement, the user sends the designated messaged to the robot control system through tcp/ip (Hereinafter referred to as the control system).The control system is processed separately according to different message contents, and the processing result is returned to the sender server.

Messages for movement types, the control system returns the result of the command immediately(Typically, the return is 0,indicating that the command is successful and the others are error codes).Then the movement message sender needs to send the "ReadMoveState" message constantly to get the movement Whether done or not. In general, the next movement can be performed only when the robot has completes the current work.Except foer “MoveB/MoveBJ” instructions。

For other types of messages, the control system returns the request result. For specific results, please see the

communication protocol.

The communication system only processes one full message at a time. For example, When sending multiple messages at once, the control system processes only the first message that satisfies the protocol format, other discard. If the message sent is incomplete, the control system returns the invalid result of the command.

1.2 Message Format

(1) Communication protocol

Message name, Param1,Param2,Param3……Paramn,;

The message format, as shown above， is composed of a message name and parameters. A complete message ends in English comma and semicolon, where each field is separated by an English comma.

Message commands and message replies are all in the ASCII format(String form).

The specific message format is shown in reference 3.

The parameter that is not used is assigned 0.

(2) Return format

1) Successful return:

Message name,OK,Param1,Param2,Param3……Paramn,;

The message format, as shown above，consists of a message name, a success flag OK and parameters. A complete message ends in English comma and semicolon , where each field is separated by an English comma.

2) Fail to return:

Message name, Fail, ErrorCode,;

The message format, as shown above，consists of a message name, a failure flag Fail and error message code. A complete message ends in English comma and semicolon, where each field is separated by an English comma.

The error code is in the form of 'int'.

The specific error code is shown in reference 4.

2 System Architecture

The protocol adopts the standard c/s architecture. Han's Motor provides the client and server to receive and process the robot control message(can be configured through a configuration file change), and the whole communication process is carried out in tcp/ip mode.

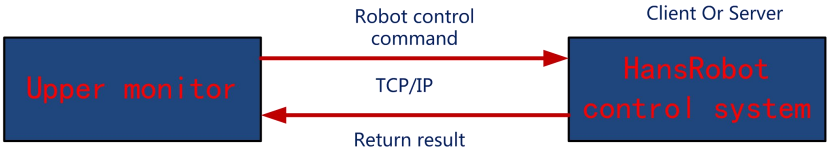

Fig 2-1 system architecture

2. 1 Connection Mode

2.1.1 As Client

As a communication client, the HansRobot control system will monitor the connection server signals which is sent by the server of the upper monitor(Triggered in the form of IO). When the IO signal is received, the client will configure the IP address and port number according to the configuration file, and use the tcp/ip protocol to automatically connect the upper monitor server. When the server sends a connection request, if the client already connects to the server, the client closes the current connection first and then connects to the server.

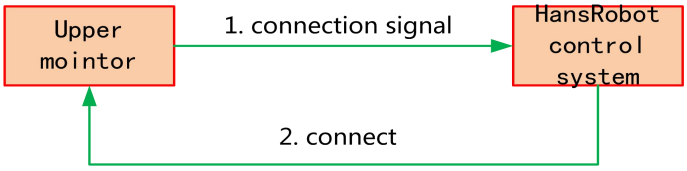

Fig 2-2 connection mode

2.1.2 As Server

As a communication server, the HansRobot control system will

monitor the designed port and waiting for the connection of the client of the upper monitor. In the above process, the port number of the server monitor can be set by configuration file.

2.1.3 Configuration File

By modifying the parameters through the communication protocol,it will not be saved to the configuration file,that is,the single cannon is valid. Restart the controller and restore the default parameters.In HansRobot,the settings of the instruction set will be saved to the configuration file.

3 Communication Protocol

3. 1 Electrify

Function: Power the robot

Format: Electrify,;

Parameter quantity: None

Successful return:

”Electrify,OK,;”

Notes: successful completion of power up before returning, power up time is about 44s.

Fail to return:

” Electrify,Fail,ErrorCode,;”, error types see the error code list

Example:

Electrify,;

3. 2 BlackOut

Function: Robot blackout

Format： BlackOut,;

Parameter quantity: None

Successful return:

”BlackOut,OK,;”

Notes: successful power outage will only return, power failure time is 3s.

Fail to return:

” BlackOut,Fail,ErrorCode,;”, error types see the error code list

Example:

BlackOut,;

3. 3 StartMaster

Function: Start master station

Format: StartMaster,;

Parameter quantity: None

Successful return:

”StartMaster,OK,;”

Notes: the master station will not be returned until successfully started, startup master time is about 4s.

Fail to return:

” StartMaster,Fail,ErrorCode,;”, error types see the error code list

Example:

StartMaster,;

3. 4 CloseMaster

Function: Close master station

Format: CloseMaster,;

Parameter quantity: None

Successful return:

”CloseMaster,OK,;”

Notes: the master station will not be returned until successfully closed, shut down the master station time is about 2s.

Fail to return:

” CloseMaster,Fail,ErrorCode,;” , error types see the error code list

Example:

CloseMaster,;

3. 5 GrpPowerOn

Function: Robot servo on

Format： GrpPowerOn,rbtID,;

Parameter quantity: 1

Successful return:

”GrpPowerOn,OK,;”

Fail to return:

” GrpPowerOn,Fail,ErrorCode,;” , error types see the error code list

Parameter detail:1

Parameter name	type	Meaning
rbtID	int	Robot index, count from zero

Example:

GrpPowerOn,0,;

3.6 GrpPowerOff

Function: Robot servo off

Format: GrpPowerOff,rbtID,;

Parameter quantity:1

Successful return:

”GrpPowerOff,OK,;”

Fail to return:

” GrpPowerOff,Fail,ErrorCode,;” , error types see the error code list

Parameter detail：

Parameter name	type	Meaning
rbtID	int	Robot index, count from zero

Example:

GrpPowerOff,0,;

3. 7 GrpStop

Function: Stop robot

Format: GrpStop,rbtID,;

Parameter quantity: 1

Successful return:

”GrpStop,OK,;”

Fail to return:

” GrpStop,Fail,ErrorCode,;” , error types see the error code list

Parameter detail：

Parameter name	type	Meaning
rbtID	int	Robot index, count from zero

Example:

GrpStop,0,;

3. 8 GrpReset

Function: Reset, used to clear errors

Format: GrpReset,rbtID,;

Parameter quantity: 1

Successful return:

”GrpReset,OK,;”

Fail to return:

” GrpReset,Fail,ErrorCode,;”, error types see the error code list

Parameter detail:

Parameter name	type	Meaning
rbtID	int	Robot index, count from zero

Example:

GrpReset,0,;

3. 9 MoveHoming

Function: Robot returns to the origin
Format: MoveHoming,rbtID,;
Parameter quantity: 1
Successful return:
”MoveHoming,OK,;”
Fail to return:
” MoveHoming,Fail,ErrorCode,;” , error types see the error code list
Parameter detail:

Parameter name	**type**	**Meaning**
rbtID	int	Robot index, count from zero

Example:
MoveHoming,0,;

3. 10 MoveJ

Function: Robot moves to the specified angular coordinate position
Format: MoveJ,rbtID,J1,J2,J3,J4,J5,J6,;
Parameter quantity: 7
Successful return:
”MoveJ,OK,;”
Fail to return:
” MoveJ,Fail,ErrorCode,;” , error types see the error code list
Parameter detail:

Parameter name	**type**	**Meaning**
rbtID	int	Robot index, count from zero
J1	double	Axis 1 command position, unit degree
J2	double	Axis 2 command position, unit degree

J3	double	Axis 3 command position, unit degree
J4	double	Axis 4 command position, unit degree
J5	double	Axis 5 command position, unit degree
J6	double	Axis 6 command position, unit degree

Example:

MoveJ,0,0,0,90,0,90,0,;

3. 11 MoveL

function: Robot moves straight to the specified space coordinates

format: MoveL,rbtID,X,Y,Z,RX,RY,RZ,;

parameter quantity: 7

successful return:

”MoveL,OK,;”

fail to return:

” MoveL,Fail,ErrorCode,;” , error types see the error code list

Parameter detail:

Parameter name	**type**	**Meaning**
rbtID	int	Robot index, count from zero
X	double	X axis command position, unit mm
Y	double	Y axis command position, unit mm
Z	double	Z axis command position, unit mm
RX	double	RX axis command position, unit degree
RY	double	RY axis command position, unit

		degree
RZ	double	RZ axis command position, unit degree

Example:

MoveL,0,450,0,450,180,0,-180,;

Note：there is a singular point in space motion.

3.12 MoveP

Function: Robot moves to the specified space coordinate position

Format: MoveP,rbtID,X,Y,Z,A,B,C,;

Parameter quantity: 7

Successful return:

”MoveP,OK,;”

Fail to return:

” MoveP,Fail,ErrorCode,;” , error types see the error code list

Parameter detail:

Parameter name	type	Meaning
rbtID	int	Robot index, count from zero
X	double	X axis command position, unit mm
Y	double	Y axis command position, unit mm
Z	double	Z axis command position, unit mm
RX	double	RX axis command position, unit degree
RY	double	RY axis command position, unit degree
RZ	double	RZ axis command position, unit degree

Example:

MoveP,0,450,0,450,180,0,-180,;

Note：there is a singular point in space motion.

3. 13 MoveB

Function: Immediately change the end point of the robot's current movement to the specified space coordinate position

Format: MoveB,rbtID,X,Y,Z,A,B,C,;

Parameter quantity: 7

Successful return:

”MoveB,OK,;”

Fail to return:

” MoveB,Fail,ErrorCode,;” , error types see the error code list

Parameter detail:

Parameter name	type	Meaning
rbtID	int	Robot index, count from zero
X	double	X axis command position, unit mm
Y	double	Y axis command position, unit mm
Z	double	Z axis command position, unit mm
RX	double	RX axis command position, unit degree
RY	double	RY axis command position, unit degree
RZ	double	RZ axis command position, unit degree

Example:

MoveB,0,450,0,450,180,0,-180,;

;Sleep

MoveB,0,440,0,450,180,0,-180,;

;Sleep

MoveB,0,430,0,450,180,0,-180,;

Note：there is a singular point in space motion.

3.14 MoveBJ

Function: Immediately change the end point of the robot's current movement to the specified angular coordinate position

Format: MoveBJ,rbtID,X,Y,Z,A,B,C,;

Parameter quantity: 7

Successful return:

”MoveBJ,OK,;”

Fail to return:

” MoveBJ,Fail,ErrorCode,;” , error types see the error code list

Parameter detail:

Parameter name	type	Meaning
rbtID	int	Robot index, count from zero
J1	double	Axis 1 command position, unit degree
J2	double	Axis 2 command position, unit degree
J3	double	Axis 3 command position, unit degree
J4	double	Axis 4 command position, unit degree
J5	double	Axis 5 command position, unit degree
J6	double	Axis 6 command position, unit degree

Example:

MoveBJ,0,0,0,90,0,90,0,;

;Sleep

MoveBJ,0,0,0,90,0,93,0,;

;Sleep

MoveBJ,0,0,0,90,0,96,0,;

Note：there is a singular point in space motion.

3.15 MoveC

Function: Immediately change the end point of the robot's current movement to the specified angular coordinate position

Format: MoveC,rbtID,ViaCoord[3],GoalCoord[6],;

Parameter quantity: 10

Successful return:

”MoveC,OK,;”

Fail to return:

” MoveC,Fail,ErrorCode,;” , error types see the error code list

Parameter detail:

Parameter name	type	Meaning
rbtID	int	Robot index, count from zero
ViaCoord[3]	double	Through position，X、Y、Z axis command position, unit mm
GoalCoord[6]	double	Target position，X、Y、Z、RX、RY、RZaxis command position, unit mm/degree

Example:

;Motion to the initial position

MoveJ,0,0,0,90,0,90,0,;

;Using the “ReadMoveState” instruction to judge the movement place

MoveC,0,410,-50,420,350,-100,420,180,0,180,;

Note：there is a singular point in space motion.

3. 16 MoveRelL

Function: Robot moves a certain distance from the specified spatial coordinate axis

Format: MoveRelL,rbtID,axisID,direction,distance,;

Parameter quantity: 4

Successful return:

”MoveRelL,OK,;”

Fail to return:

” MoveRelL,Fail,ErrorCode,;” , error types see the error code list

Parameter detail:

Parameter name	type	Meaning
rbtID	int	Robot index, count from zero
axisID	int	Spatial coordinate axis ID, count from zero
direction	int	Movement direction: 0=negative;1=positive;
distance	double	Relative motion distance, X, Y, Z unit mm, A, B,C unit degree

Example:

The Z axis moves in the direction of 10 millimetres in the direction of positive direction

MoveRelL,0,2,1,10,;

Note：there is a singular point in space motion.

3. 17 MoveRelJ

Function: Robot moves a certain distance from the specified angular coordinate axis

Format: MoveRelJ,rbtID,axisID,direction,distance,;

Parameter quantity: 4

Successful return:

”MoveRelJ,OK,;”

Fail to return:

” MoveRelJ,Fail,ErrorCode,;” , error types see the error code list

Parameter detail:

Parameter name	type	Meaning
rbtID	int	Robot index, count from zero
axisID	int	Angular coordinate axis ID, count from zero
direction	int	Movement direction: 0=negative;1=positive;
distance	double	Relative motion distance, unit degree

Example:

The J3 axis moves 10 degrees in the direction of positive direction

MoveRelJ,0,2,1,10,;

3. 18 ShortJogJ

Function: Angular motion, fixed distance exercise(the movement distance is 2 degrees)

Format: ShortJogJ,rbtID,AxisID,Derection,;

Parameter quantity:3

Successful return:

”ShortJogJ,OK,;”

Fail to return:

” ShortJogJ,Fail,ErrorCode,;” , error types see the error code list

Parameter detail:

Parameter name	type	Meaning
rbtID	int	Robot index, count from zero
AxisID	int	Point moving axle ID(0-5)

		(0:J1,1:J2,2:J3,3:J4,4:J5,5:J6)
Derection	int	Movement direction: 0=negative;1=positive;

Example:

ShortJogJ,0,3,1,;

3. 19 ShortJogL

Function: Space movement, fixed distance exercise(movement distance: X, Y, Z axis is 2 mm, A, B, C axis is 2 degrees)

Format: ShortJogL,rbtID,AxisID,Derection,;

Parameter quantity: 3

Successful return:

”ShortJogL,OK,;”

Fail to return:

” ShortJogL,Fail,ErrorCode,;” , error types see the error code list

Parameter detail:

Parameter name	type	Meaning
rbtID	int	Robot index, count from zero
AxisID	int	Point moving axle ID(0-5) (0:X,1:Y,2:Z,3:A,4:B,5:C)
Derection	int	Movement direction: 0=negative;1=positive;

Example:

ShortJogL,0,3,1,;

3. 20 LongJogJ

Function: Angular, unfixed distance movement

Format: LongJogJ,rbtID,AxisID,Derection,;

Parameter quantity:3

Notes: When the order is issued, the other stop command must be issued to stop the movement

Successful return:

”LongJogJ,OK,;”

Fail to return:

” LongJogJ,Fail,ErrorCode,;” , error types see the error code list

Parameter detail:

Parameter name	type	Meaning
rbtID	int	Robot index, count from zero
AxisID	int	Point moving axle ID(0-5) (0:J1,1:J2,2:J3,3:J4,4:J5,5:J6)
Derection	int	Movement direction: 0=negative;1=positive;

Example:

LongJogJ,0,3,1,;

3. 21 LongJogL

Function: Spatial motion, unfixed distance motion

Format: LongJogL,rbtID,AxisID,Drection,;

Parameter quantity:3

Notes: When the order is issued, the other stop command must be issued to stop the movement

Successful return:

”LongJogL,OK,;”

Fail to return:

” LongJogL,Fail,ErrorCode,;” , error types see the error code list

Parameter detail:

Parameter name	type	Meaning
rbtID	int	Robot index, count from zero

AxisID	int	Point moving axle ID(0-5) (0:X,1:Y,2:Z,3:A,4:B,5:C)
Derection	int	Movement direction: 0=negative;1=positive;

Example:

LongJogL,0,3,1,;

3. 22 SetKinematicCoordinate

Function: Setting tool coordinates

Format: SetKinematicCoordinate,rbtID,X,Y,Z,A,B,C,;

Parameter quantity:7

Successful return:

"SetKinematicCoordinate,OK,;"

Fail to return:

" SetKinematicCoordinate,Fail,ErrorCode,;" , error types see the error code list

Parameter detail:

Parameter name	type	Meaning
rbtID	int	Robot index, count from zero
X	double	X axis command position, unit mm
Y	double	Y axis command position, unit mm
Z	double	Z axis command position, unit mm
RX	double	RX axis command position, unit degree
RY	double	RY axis command position, unit degree
RZ	double	RZ axis command position, unit degree

Example:

SetKinematicCoordinate,0,0,0,100,0,0,0,;

3. 23 SetUserCoordinate

Function: Setting user coordinates

Format: SetUserCoordinate,rbtID,X,Y,Z,A,B,C,;

Parameter quantity:7

Successful return:

SetUserCoordinate,OK,;"

Fail to return:

" SetUserCoordinate,Fail,ErrorCode,;" , error types see the error code list

Parameter detail:

Parameter name	type	Meaning
rbtID	int	Robot index, count from zero
X	double	X axis command position, unit mm
Y	double	Y axis command position, unit mm
Z	double	Z axis command position, unit mm
RX	double	RX axis command position, unit degree
RY	double	RY axis command position, unit degree
RZ	double	RZ axis command position, unit degree

Example:

SetUserCoordinate,0,0,0,0,180,0,180,;

3. 24 SetToolCoordinateMotion

Function: Set tool coordinate motion

Format: SetToolCoordinateMotion,rbtID,Switch,;

Parameter quantity:2

Successful return:

SetToolCoordinateMotion,OK,;”

Fail to return:

” SetToolCoordinateMotion,Fail,ErrorCode,;” , error types see the error code list

Parameter detail:

Parameter name	type	Meaning
rbtID	int	Robot index, count from zero
Switch	int	Switch：0=close； 1=open；

Example:

SetToolCoordinateMotion,0,1,;

3. 25 SetOverride

function: Set speed ratio

format: SetOverride,rbtID,override,;

parameter quantity:2

successful return:

”SetOverride,OK,;”

fail to return:

” SetOverride,Fail,ErrorCode,;” , error types see the error code list

Parameter detail:

Parameter name	type	Meaning

rbtID	int	Robot index, count from zero
override	double	set speed ratio, range of 0.01~1 (1%~100%)

Example:
SetOverride,0,0.05,;

3. 26 SetSpeedUp

function: Speed up
format: SetSpeedUp,rbtID,nType,nSwitch,;
parameter quantity: 3
successful return:
"SetSpeedUp,OK,;"
fail to return:
" SetSpeedUp,Fail,ErrorCode,;" , error types see the error code list
Parameter detail:

Parameter name	type	Meaning
rbtID	int	Robot index, count from zero
nType	int	0=manual accelerate;1=auto accelerate
nSwitch	int	0=stop acceleration;1=sustained acceleration

3. 27 SetSpeedDown

function: Speed down
format: SetSpeedDown,rbtID,nType,nSwitch,;
parameter quantity:3
successful return:

”SetSpeedDown,OK,;”

fail to return:

” SetSpeedDown,Fail,ErrorCode,;” , error types see the error code list

Parameter detail:

Parameter name	type	Meaning
rbtID	int	Robot index, count from zero
nType	int	0=manual deceleration; 1=automatic deceleration
nSwitch	int	0=stop deceleration;1=sustained deceleration

3. 28 SetOutIOState

function: Set output IO state

format: SetOutIOState,rbtID,ioIndex,;

parameter quantity:3

successful return:

”SetOutIOState,OK,ioState,;”

fail to return:

” SetOutIOState,Fail,ErrorCode,;” , error types see the error code list

Parameter detail:

Parameter name	type	Meaning
rbtID	int	Robot index, count from zero
ioIndex	int	The output IO index to set, different robots can be use different index ranges
ioState	int	State required to set （0:low level;1:high level;）

Example:

SetOutIOState,0,1,;

3. 29 SetSerialDO

function: Set the serial port output IO state

format: SetSerialDO,bit,;

parameter quantity:2

successful return:

”SetSerialDO,OK,state,;”

fail to return:

” SetSerialDO,Fail,ErrorCode,;” , error types see the error code list

Parameter detail:

Parameter name	type	Meaning
bit	int	Serial port to set
state	int	Set state: 0=low level;1=high level;

Example:

SetSerialDO,1,;

3. 30 SetConveyorScale

Function: Set percentage of conveyor belts

Format: SetConveyorScale,rbtID,dCount,nDirection,;

Parameter quantity: 3

Successful return:

” SetConveyorScale,OK,;”

Fail to return:

” SetConveyorScale,Fail,ErrorCode,;” , error types see the error code list

Parameter detail:

Parameter name	type	Meaning
rbtID	int	Robot index, count from zero
scale	double	conveyor belt scale (range of 0~1)
dCount	double	count per meter,greater than 0
nDirection	int	Following direction: 0=negative direction of Y axis 1=positive direction of Y axis

Example:

SetConveyorScale,0,38956,1,;

3. 31 SetTrackingSwitch

Function: Set control tracking switch

Format: SetTrackingSwitch,rbtID,TrackSwitch,;

Parameter quantity:2

Successful return:

"SetTrackingSwitch,OK,;"

Fail to return:

" SetTrackingSwitch,Fail,ErrorCode,;" , error types see the error code list

Parameter detail:

Parameter name	type	Meaning
rbtID	int	Robot index, count from zero
TrackSwitch	Int	Follwing switch:0=close;1=open

Example:

SetTrackingSwitch,0,1,;

3. 32 SetRunningMode

Function: Switch debug mode

Format: SetRunningMode,nType,;

Parameter quantity:1

Successful return:

"SetRunningMode,OK,;"

Fail to return:

"SetRunningMode,Fail,ErrorCode,;" , error types see the error code list

Parameter detail:

Parameter name	type	Meaning
nType	int	0=normal mode; 1=debug mode

Example:

SetRunningMode,1,;

3. 33 SetSimulation

Function:Seting simulative mode

Format:SetSimulation,nSimulation,;

Parameter quantity:1

Successful return:

"SetSimulation,OK,;"

Fail to return:

" SetSimulation,Fail,ErrorCode,;", error types see the error code list

Parameter detail:

Parameter name	type	Meaning
nSimulation	int	0=close simulative mode 1=open simulative mode

Example:

SetSimulation,1,;

3. 34 SetPayload

Function:Seting load

Format:

SetPayload,nRbtID,mass,masscenterX,masscenterY,masscenterZ,;

Parameter quantity:4

Successful return:

"SetPayload,OK,;"

Fail to return:

" SetPayload,Fail,ErrorCode,;", error types see the error code list

Parameter detail:

Parameter name	type	Meaning
nRbtID	int	Robot index, count from zero
mass	double	Load mass,unit kilogram
masscenterX	double	Load centroid x coordinates,unit millimeter
masscenterY	double	Load centroid y coordinates,unit millimeter
masscenterZ	double	Load centroid z coordinates,unit millimeter

Example:

SetPayload,0,3,0,0,0,;

3. 35 SetBaseMountingAngle

Function:Seting up the installation Angle

Format:SetBaseMountingAngle,nRbtID,Rotation,Tilt,;

Parameter quantity:3

Successful return:

"SetBaseMountionAngle,OK,;"

Fail to return:

" SetBaseMountionAngle,Fail,ErrorCode,;", error types see the error code list

Parameter detail:

Parameter name	type	Meaning
nRbtID	int	Robot index, count from zero
Rotation	double	Rotation,unit degree
Tilt	double	Tilt,unit degree

Example:

SetBaseMountingAngle,0,90,90,;

3. 36 StartAssistiveMode

Function:Opening zero force teaching mode。

Format:StartAssistiveMode,rbtID,;

Parameter quantity:1

Successful return:

"StartAssistiveMode,OK,;"

Fail to return:

" StartAssistiveMode,Fail,ErrorCode,;", error types see the error code list

Parameter detail:

Parameter name	type	Meaning
rbtID	int	Robot index, count from zero

Example:

StartAssistiveMode,0,;

3. 37 CloseAssistiveMode

Function:Close the zero force teaching mode

Format:CloseAssistiveMode,rbtID,;

Parameter quantity:1

Successful return:

"CloseAssistiveMode,OK,;"

Fail to return:

" CloseAssistiveMode,Fail,ErrorCode,;", error types see the error code list

Parameter detail:

Parameter name	type	Meaning
rbtID	int	Robot index, count from zero

Example:

CloseAssistiveMode,0,;

3. 38 SetStartTimer

Function:Open the timer

Format:SetStartTimer,;

Parameter quantity:0

Successful return:

"SetStartTimer,OK,;"

Fail to return:

" SetStartTimer,,Fail,ErrorCode,;", error types see the error code list

Example:

SetStartTimer,;

3. 39 SetCloseTimer

Function:Off timer

Format:SetCloseTimer,;

Parameter quantity:0

Successful return:

"SetCloseTimer,OK,;"

Fail to return:

" SetCloseTimer,Fail,ErrorCode,;" error types see the error code list

Example:

SetCloseTimer,;

3. 40 ReadAcsActualPos

Function: Get the actual position of the angle coordinate

Format: ReadAcsActualPos,rbtID,;

Parameter quantity: 1

Successful return:

" ReadAcsActualPos,OK,J1,J2,J3,J4,J5,J6,;"

Fail to return:

" ReadAcsActualPos,Fail,ErrorCode,;" , error types see the error code list

Parameter detail:

Parameter name	type	Meaning
rbtID	int	Robot index, count from zero
J1	double	The position of J1 axis, unit: degree
J2	double	The position of J2 axis, unit: degree
J3	double	The position of J3 axis, unit: degree

J4	double	The position of J4 axis, unit: degree
J5	double	The position of J5 axis, unit: degree
J6	double	The position of J6 axis, unit: degree

Example:

ReadAcsActualPos,0,;

3. 41 ReadPcsActualPos

Function: Get the actual position of the space coordinate

Format: ReadPcsActualPos,rbtID,;

Parameter quantity:1

Successful return:

" ReadPcsActualPos,OK,X,Y,Z,A,B,C,;"

Fail to return:

" ReadPcsActualPos,Fail,ErrorCode,;" , error types see the error code list

Parameter detail:

Parameter name	type	Meaning
rbtID	int	Robot index, count from zero
X	double	The position of X axis, unit: mm
Y	double	The position of Y axis, unit: mm
Z	double	The position of Z axis, unit: mm
RX	double	The position of RX axis, unit: mm
RY	double	The position of RY axis, unit: mm

RZ	double	The position of RZ axis, unit: mm

Example:

ReadPcsActualPos,0,;

3. 42 ReadRobotPosInfo

Function:Get the robot position information

Format: ReadRobotPosInfo,nRbtID,;

Parameter quantity:1

Successful return:

"ReadRobotPosInfo,OK,AcsActualPos[6],PcsActualPos1[6],ActualCurr[6],AcsCmdPos[6],PcsCmdPos[6],CmdCurr[6],;"

Fail to return:

" ReadPcsActualPos,Fail,ErrorCode,;", error types see the error code list

Parameter detail:

Parameter name	type	Meaning
rbtID	int	Robot index, count from zero
AcsActualPos[6]	double	Actual angle position,unit: degree
PcsActualPos[6]	double	Actual space position,unit: millimetre/degree
ActualCurr[6]	double	Actual current,unit: ampere
AcsCmdPos[6]	double	Command angle position,unit: degree
PcsCmdPos[6]	double	Command space position,unit: millimetre/degree
CmdCurr[6]	double	Command current,unit: ampere

Example:

ReadRobotPosInfo,0,;

3. 43 ReadConveyorValue

Function: Get the value of the conveyor encoder

Format: ReadConveyorValue,rbtID,;

Parameter quantity:1

Successful return:

" ReadConveyorValue,OK,conveyorVal,;"

Fail to return:

" ReadConveyorValue,Fail,ErrorCode,;" , error types see the error code list

Parameter detail:

Parameter name	type	Meaning
rbtID	Int	Robot index, count from zero
conveyorVal	double	The value of the conveyor encoder

Example:

ReadConveyorValue,0,;

3. 44 ReadOverride

Function: Get speed

Format: ReadOverride,rbtID,;

Parameter quantity:1

Successful return:

" ReadOverride,OK,;"

Fail to return:

" ReadOverride,Fail,ErrorCode,;" , error types see the error code list

Parameter detail:

Parameter name	type	Meaning
rbtID	Int	Robot index, count from zero

Override	Double	Speed ratio，0.01~1 (1%~100%)

Example:

ReadOverride,0,;

3. 45 ReadInIOState

Function: Get the state of the input IO

Format: ReadInIOState,rbtID,ioIndex,;

Parameter quantity:2

Successful return:

" ReadInIOState,OK,state,;"

Fail to return:

" ReadInIOState,Fail,ErrorCode,;" , error types see the error code list

Parameter detail:

Parameter name	type	Meaning
rbtID	Int	Robot index, count from zero
ioIndex	int	The IO index to read
State	int	The state of input IO: 0=low level; 1=high level;

Example:

ReadInIOState,0,1,;

3. 46 ReadOutIOState

Function: Get the state of the output IO

Format: ReadOutIOState,rbtID,ioIndex,;

Parameter quantity:2

Successful return:

” ReadOutIOState,OK,state,;”

Fail to return:

” ReadOutIOState,Fail,ErrorCode,;” , error types see the error code list

Parameter detail:

Parameter name	type	Meaning
rbtID	int	Robot index, count from zero
ioIndex	int	The IO index to read, count from one
state	int	The state of input IO: 0=low level; 1=high level;

Example:

ReadOutIOState,0,1,;

3. 47 ReadMoveState

Function: Get the motion state of the robot

Format: ReadMoveState,rbtID,;

Parameter quantity:1

Successful return:

” ReadMoveState,OK,MoveState,;”

Fail to return:

” ReadMoveState,Fail,ErrorCode,;” , error types see the error code list

Parameter detail:

Parameter name	type	Meaning
rbtID	int	Robot index, count from zero

MoveState	int	Current state of motion of robot: 0=motion completion; 1009=in motion; 1013=waiting for execution; 1025=Error reporting state;

Example:

ReadMoveState,0,;

3. 48 ReadMachineOrigin

Function:Get the mechanical origin

Format: ReadMachineOrigin,rbtID,;

Parameter quantity:1

Successful return:

" ReadMachineOrigin,OK,HomePos[6],;"

Fail to return:

" ReadMachineOrigin,Fail,ErrorCode,;" , error types see the error code list

Parameter detail:

Parameter name	type	Meaning
rbtID	int	Robot index, count from zero
HomePos[6]	int	The mechanical origin of axis 1~6

Example:

ReadMachineOrigin,0,;

3. 49 ReadRobotState

Function: Get the state of the robot

Format: ReadRobotState,rbtID,;

Parameter quantity:1

Successful return:

” ReadRobotState,OK,movingState,powerState,errorState, errorCode,errAxisID,BrakingState,reserval1State,reserval2State,;”

Fail to return:

” ReadRobotState,Fail,ErrorCode,;” , error types see the error code list

Parameter detail:

Parameter name	type	Meaning
rbtID	int	Robot index, count from zero
movingState	int	0=no movement 1=in motion
powerState	int	0=de enable state 1=enabling state
errorState	int	0=Error free 1=Report errors
errorCode	int	Robot error code
errAxisID	Int	The wrong axis ID of the manipulator
BrakingState	int	0=no brake operation 1=brake work
Reserval1State	int	Spare
Reserval2State	int	Spare

Example:

ReadRobotState,0,;

3. 50 ReadSerialDI

Function: Get the input bit state specified by the serial port

Format: ReadSerialDI,bit,;

Parameter quantity:1

Successful return:

”ReadSerialDI,OK,state,;”

Fail to return:

” ReadSerialDI,Fail,ErrorCode,;” , error types see the error code list

Parameter detail:

Parameter name	type	Meaning
bit	Int	Input bits needed to obtain
State	Int	The state of being returned, 0=low level;1=high level;

Example:

ReadSerialDI,1,;

3. 51 ReadSerialDO

Function: Get the output bit state specified by the serial port

Format:ReadSerialDO,bit,;

Parameter quantity: 1

Successful return:

”ReadSerialDO,OK,state,;”

Fail to return:

” ReadSerialDO,Fail,ErrorCode,;” , error types see the error code list

Parameter detail:

Parameter name	type	Meaning
Bit	Int	Output bits needed to obtain
State	Int	The state of being returned, 0=low level;1=high level;

Example:

ReadSerialDO,1,;

3. 52 ReadSerialAnalog

Function:Get the analog amount of the serial port

Format:ReadSerialAnalog,nIndex,;

Parameter quantity: 1

Successful return:

”ReadSerialAnalog,OK,nVal,;”

Fail to return:

” ReadSerialAnalog,Fail,ErrorCode,;”, error types see the error code list

Parameter detail:

Parameter name	type	Meaning
nIndex	Int	Analog bits that need to be obtained
nVal	Int	The amount of simulation returned

Example:

ReadSerialAnalog,1,;

3. 53 HoldScriptFunc

Function：Pause running script

Format:HoldScriptFunc,;

Parameter quantity:0

Successful return:

”HoldScriptFunc,OK,;”

Fail to return:

” HoldScriptFunc,Fail,ErrorCode,;”, error types see the error code list

Example:

HoldScriptFunc,;

3. 54 ContinusScriptFunc

Function：Continue running the script

Format:ContinusScriptFunc,;

Parameter quantity:0

Successful return:

”ContinusScriptFunc,OK,;”

Fail to return:

” ContinusScriptFunc,Fail,ErrorCode,;”, error types see the error code list

Example:

ContinusScriptFunc,;

4 ErrorCode

ErrorCode	Meaning
10000	Short circuit error
10001	Over voltage limit error
10002	Under voltage limit error
10003	Over velocity limit error
10004	Execute error
10005	Over current error
10006	Encoder error
10007	Following position error
10008	Following velocity error
10009	Negative limit error
10010	Positive limit error
10011	Server over heating error
10012	Max current error
10013	Emergency stop error
10014	UDM error
10015	Server parameter error
20000	Controller is not started
20001	Master is not started
20002	some slave is dropped
20003	Robot on safe stop state
20004	Robot on physical stop state
20005	Robot out safe space
20006	Robot enable time out
20007	Robot not electrify
30000	Collision shutdown
30001	Robot Collide with body
30002	Over joint limit error
30003	Singularity error

30004	General stopping criterion
30005	calculate failed
30006	UDM Status Error
30007	Init slave Error
30008	Home Step2 Error
30009	Out Of Axis Limit Error
1001	The robot has not been initialized
1002	Master station has not been started
1003	Slave station drop off
1004	The robot is safely locked
1005	The physical stop
1006	Robot has not been servo on
1007	Error reporting from slave station
1008	Robot beyond safe space
1009	In robot motion
1010	Invalid command

1011	Parameter error
1012	Function call format error
1013	Waiting for command execution
1014	IO does not exist
1015	Robots do not exist
1016	No connection server
1017	Network timeout
1018	Connection failed
1019	Serial connection failed
1020	No zero position is set
1021	The last same command has not been completed
1022	Serial port Di is empty
1023	Serial port DO is empty
1024	Wait timeout
1025	Error status
1026	Stop robot
1027	Robot has been servo off
1028	Robot has been servo on
1029	Function has not been enabled
1030	Start master timeout
1031	The robot has not been powered on
1032	Serial port has not been started
1033	The simulation state command is invalid
1034	RTOS Library not exsit
1035	DCS Handle Command thread crash
1039	Script running
1040	Xml Param Error
1041	System Board Not Connect
1042	Controller Not Start
1043	Controller Statu Error
1044	Robot in TeachMode
1045	Robot Already Electrify

1046	Connect to Modbus Failed
1047	Master is Started
1048	Parameter over specified payload
1049	DCS Status Error
1050	Target position invalid
2000	Load library failed
2001	The script is empty
2002	Compile error
2003	Reload script error
2004	Function does not exist
2005	Function return type error
2006	MissSignal1
2007	MissSignal2
2008	Parameter type error
2009	There is no header file included
2010	No return value
2012	UDM Stack Err
2013	Script been lock,maybe compiling
2014	Not In RunScript Statu
2015	Serial Close
2016	Serial Close
2017	Controller not started
2018	Socket Not Connected
2020	Function Name have Space.
2021	Socket Error
2020	RS:Function broken stop.

5 Terminology

Terminology	Explain
ACS	Joint coordinates, unit degree
PCS	World coordinate, unit mm、unit degree

6 Control robot motion process—Sample

6. 1 Flow chart

When the robot has been servo on, the robot can move to the target position by sending a series of move instructions and a short and long moving instructions. The following points need to be noted in the process of sending motion instructions.

1. Only when the current motor command of the robot is successfully completed can the next movement command be issued(The robot motion state is obtained by sending the robot motion state command " ReadMoveState ").
2. When the return of the command that calls the robot's motion state is in motion, The upper monitor should be called periodically to obtain instructions about the robot's motion state until the robot's motion state is identified. such as, the motion is completed or faulty.
3. When the control system returns the robot motion error, the upper monitor should call GrpReset command to clean the control system. Only when the error is successful can the call to the next movement be continued.
4. If the control system returns to the robot movement is complete, you can continue with the next movement instruction, then repeat 2-4.

The specific process is shown below:

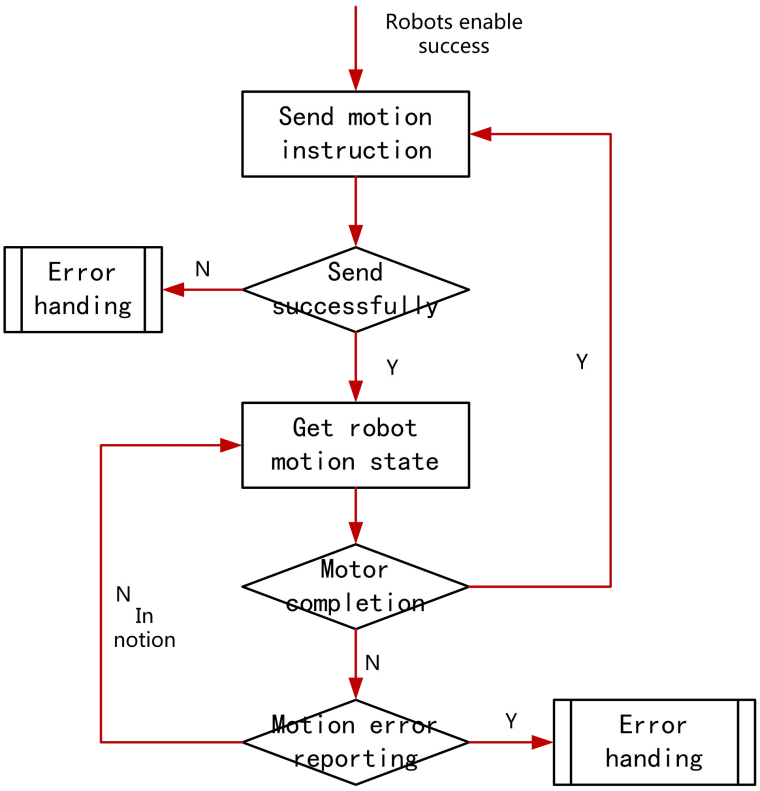

Fig Flow chart of robot motion

6.2 Sample code

Pseudo code is as follows, for reference only:

```
// Waiting for the movement to complete
int WaitMotionFinish()
{
   while(1)
   {
      // 1. Send get robot motion state message: ReadMoveState,;

      // 2. Parsing the returned motion state: MoveState
      if(MoveState == 1009)
      {
         // In motion, wait for 10 milliseconds, query again
         Sleep(10);
      }
      else if(MoveState == 0)
```

```
            {
                // The movement is completed and is returned directly to
the zero.
                return 0;
            }
            else
            {
                // Other error conditions are dealt with separately
                return nErrCode;
            }
        }
        return 0;
    }
```