

---

## **FMC4030 Controller secondary development library description**

This document describes the detailed instructions for the secondary development library of FMC4030 controller developed by Chengdu Fuyu Technology. The secondary development library provides two operating system library files, Windows (32-bit, 64-bit) and Ubuntu (1604 64-bit). If you need other Linux version library files, you can contact our company's after-sales technology to compile the corresponding version library files.

FMC4030-Dll.h, FMC4030-Dll.dll, FMC4030-Dll.lib are provided under Windows.

FMC4030-Lib.h, libFMC4030-Lib.a, libFMC4030-Lib.so are provided under Ubuntu.

The function interfaces provided by the above two forms of library files are consistent in terms of naming and the number and order of parameters.

The communication interface adopts the Ethernet interface. In order to ensure the stability of the controller, the communication protocol is not provided to the outside world, and the use of the controller can be completed by calling the library function.

### **Windows Detailed explanation of secondary development library**

Dynamic link library and static link library are provided under Windows, which can be called by C++, C#, Python, Matlab, Labview and other programming languages. This library file can be used normally only if the computer has a network card and can communicate with the network normally. The library file is written and compiled with VS2012, and the VC++2012 runtime library needs to be installed before running.

Return value definition:

return	meaning	Solution
0	execution succeed	Null

-1	Connection failed	Check the network cable connection, check the IP address and port number, restart the controller, etc
-2	Null	null
-3	Null	null
-4	Data creation failed	Check memory
-5	Failed to send data	Check the network cable connection, check the IP address and port number, restart the controller
-6	Data reception control	Check the network cable connection, check the IP address and port number, restart the controller
-7	Receive data error	Check network connection
-8	Null pointer error	Check whether the incoming parameter is a null pointer

**int FMC4030\_API FMC4030\_Open\_Device(int id, char\* ip, int port);**

Function: Connect the controller, multiple controllers can be connected

---

Parameter id: the id number assigned to the controller, this id number is unique and cannot be repeated

Parameter ip: the ip address of the controller to be connected, the factory default controller ip is "192.168.0.30", which can be modified, if you forget it, you can modify it through the 232 debugging serial port

Parameter port: the port number of the controller to be connected, the default is 8088, which can be modified, if you forget it, you can modify it through 232 debugging

Return value: see return value definition list

**int FMC4030\_API FMC4030\_Close\_Device(int id);**

Function: Disconnect the controller connection, this function must be called to release resources before exiting the program, otherwise the connection will fail next time

Parameter id: the id number previously assigned to the controller, unique and one-to-one correspondence

Return value: see return value list

**int FMC4030\_API FMC4030\_Jog\_Single\_Axis(int id, int axis, float pos, float speed, float acc, float dec, int mode);**

Function: execute single-axis motion of the controller, which can start different axes multiple times. If the previous motion is not completed when the

---

same axis is started, the command will not respond to this command

Parameter id: the id number previously assigned to the controller

Parameter axis: the axis numbers 0, 1, 2 to be run correspond to the X, Y, and Z axes respectively

Parameter pos: running distance, difference between positive and negative, unit mm

Parameter speed: running speed, can only be a positive number, unit mm/s

Parameter acc: the acceleration of operation, can only be a positive number, the unit is mm/s<sup>2</sup>

Parameter dec: deceleration of operation, can only be a positive number, unit mm/s<sup>2</sup>

Parameter mode: operating mode. 1: Relative motion, 2: Absolute motion

Return value: see return value list

**int FMC4030\_API FMC4030\_Check\_Axis\_Is\_Stop(int id, int axis);**

Function: Check whether a certain axis is in a stopped state, used to judge the running state of a certain axis

Parameter id: the id number previously assigned to the controller

Parameter axis: 0, 1, 2 correspond to X, Y, Z axis respectively

Return value: 1: mark the axis as stopped state, 0 mark the axis as running state

**int FMC4030\_API FMC4030\_Home\_Single\_Axis(int id, int axis, float homeSpeed, float homeAccDec, float homeFallStep, int homeDir);**

Function: Control a certain axis to execute zero return

Parameter id: the id number assigned to the controller

Parameter axis: 0, 1, 2 correspond to X, Y, Z axis respectively

Parameter homeSpeed: return to zero speed, positive number, unit mm/s

Parameter homeAccDec: homing acceleration and deceleration, positive

---

number, unit mm/s<sup>2</sup>

Parameter homeFallStep: return to zero falling distance, positive number, unit mm. This parameter is the distance away from the limit switch after the zero return is completed.

Parameter homeDir: return direction, 1: positive limit return to zero, 2: negative limit return to zero

Return value: see return value list

**int FMC4030\_API FMC4030\_Stop\_Single\_Axis(int id, int axis, int mode);**

Function: Stop a certain axis, this function can only be used to stop after starting a single axis, and cannot be used to stop during interpolation motion

Parameter id: the id number assigned to the controller

Parameter axis: the axis numbers 0, 1, 2 to be stopped correspond to the X, Y, and Z axes respectively

Parameter mode: 1: decelerate to stop, 2: stop immediately

Return value: see return value list

**int FMC4030\_API FMC4030\_Get\_Axis\_Current\_Pos(int id, int axis, float\* pos);**

Function: Get the current actual position of an axis. This position is generated by the internal count of the control card. If the motor is blocked or stuck, this position is not accurate

Parameter id: the id number assigned to the controller

Parameter axis: 0, 1, 2 correspond to X, Y, Z axis respectively

Parameter pos: Pass in a pointer to a floating point number or a pointer to a four-byte array, the data will be updated to the variable after calling this function, the unit is mm

---

Return value: see return value list

---

**int FMC4030\_API FMC4030\_Get\_Axis\_Current\_Speed(int id, int axis, float\* speed);**

Function: Get the current running speed of an axis

Parameter id: the id number assigned to the controller

Parameter axis: 0, 1, 2 correspond to X, Y, Z axis respectively

Parameter speed: Pass in a pointer to a floating point number or a pointer to a four-byte array, the data will be updated to the variable after calling this function, the unit is mm/s

Return value: see return value list

**int FMC4030\_API FMC4030\_Set\_Output(int id, int io, int status);**

Function: Set the state of the output port of the controller. This output port is an open-drain output and can be connected to high-power relays and other equipment.

Parameter id: the id number assigned to the controller

Parameter io: 0, 1, 2, and 3 correspond to OUT0, OUT1, OUT2, and OUT3 respectively

Parameter status: set the status of the output port, 0: output high level, 1: output low level

Return value: see return value list

---

**int FMC4030\_API FMC4030\_Get\_Input(int id, int io, int\* status);**

Function: Get input port status

Parameter id: the id number assigned to the controller

Parameter io: 0, 1, 2 and 3 correspond to IN0, IN1, IN2, and IN3 respectively

Parameter status: Pass in this variable pointer, the value in the variable will be updated after the call

Return value: see return value list

**int FMC4030\_API FMC4030\_Write\_Data\_To\_485(int id, char\* data, int length);**

Function: Send data through the 485 bus function of the controller, this function can be used to control external 485 devices

Parameter id: the id number assigned to the controller

Parameter data: pointer to the data array to be sent

Parameter length: the length of the data to be sent

Return value: see return value list

**int FMC4030\_API FMC4030\_Read\_Data\_From\_485(int id, char\* data, int\* length);**

Function: read data from 485 bus

Parameter id: the id number assigned to the controller



---

Parameter data: pointer to the array to receive the data, after the call, the data in the array will be updated

Parameter length: the length of the received data

Return value: see return value list

**int FMC4030\_API FMC4030\_Set\_FSC\_Speed(int id, int slaveId, float speed);**

Function: Set the speed of the company's single-axis controller through the 485 bus

Parameter id: the id number assigned to the controller

Parameter slaveid: the slave address of the single-axis controller to be controlled

Parameter speed: the speed to be set, in mm/s

Return value: see return value list

**int FMC4030\_API FMC4030\_Line\_2Axis(int id, unsigned int axis, float endX, float endY, float speed, float acc, float dec);**

Function: Two-axis linear interpolation with the current point as the starting point, the current point is controlled by the controller's internal count

Parameter id: the id number assigned to the controller

---

Parameter axis: the two axes to be controlled. Since this controller has three axes, the low three digits of a 32-bit unsigned number are used to represent the selected axis, 0x03 represents the X and Y axes, 0x05 represents the X and Z axes, and 0x06 Represents Y, Z axis

Parameter endX: X coordinate of the end point of linear interpolation, this X is not the actual X axis, it is the X of the virtual coordinate system, the unit is mm

Parameter endY: the Y coordinate of the end point of linear interpolation, this Y is not the actual Y axis, it is the Y of the virtual coordinate system, the unit is mm

Parameter speed: the combined speed of two-axis linear interpolation, which does not represent the actual speed of each axis. Unit mm/s

Parameter acc: two-axis linear interpolation synthesis acceleration, not the actual acceleration of each axis, unit mm/s<sup>2</sup>

Parameter dec: two-axis linear interpolation synthetic deceleration, not the actual deceleration of each axis, unit mm/s<sup>2</sup>

Return value: see return value list

**int FMC4030\_API FMC4030\_Line\_3Axis(int id, unsigned int axis, float endX, float endY, float endZ, float speed, float acc, float dec);**

Function: Three-axis linear interpolation with the current point as the starting point

---

Parameter id: the id number assigned to the controller

Parameter axis: the selection of the three axes to be run. Since it is a three-axis interpolation motion, the arbitrary filling of this function does not affect the actual effect

Parameter endX: X coordinate of the end point of linear interpolation, unit mm

Parameter endY: Y coordinate of the end point of linear interpolation, unit mm

Parameter endZ: Z coordinate of the end point of linear interpolation, unit mm

Parameter speed: linear interpolation synthesis speed, unit mm/s

Parameter acc: linear interpolation synthesis acceleration, unit mm/s<sup>2</sup>

Parameter dec: linear interpolation synthetic deceleration, unit mm/s<sup>2</sup>

Return value: see return value list

**int FMC4030\_API FMC4030\_Arc\_2Axis(int id, unsigned int axis, float endX, float endY, float centerX, float centerY, float radius, float speed, float acc, float dec, int dir);**

Function: Two-axis circular interpolation based on the starting point of the current point

Parameter id: the id number assigned to the controller

Parameter axis: similar to two-axis linear interpolation, 0x03 represents X, Y axis, 0x05 represents X, Z axis, 0x06 represents Y, Z axis

---

Parameter endX: X coordinate of the arc interpolation end point, in mm

Parameter endY: Y coordinate of the arc interpolation end point, in mm

Parameter centerX: X coordinate of the center of arc interpolation, unit mm

Parameter centerY: Y coordinate of the center of arc interpolation, unit mm

Parameter radius: arc interpolation radius, unit mm

Parameter speed: circular interpolation synthesis speed, unit mm/s

Parameter acc: arc interpolation synthetic acceleration, unit mm/s<sup>2</sup>

Parameter dec: circular interpolation synthetic deceleration, unit mm/s<sup>2</sup>

Parameter dir: 1: clockwise circular interpolation, 2: counterclockwise circular interpolation

Return value: see return value list

**int FMC4030\_API FMC4030\_Stop\_Run(int id);**

Function: Stop interpolation movement, including linear interpolation and circular interpolation

Parameter id: the id number assigned to the controller

Return value: see return value list

**int FMC4030\_API FMC4030\_Get\_Machine\_Status(int id, unsigned char\* machineData);**

---

Function: Get equipment status and operating parameters, including three-axis position, three-axis speed, zero return status, input status, equipment serial number, etc.

Parameter id: the id number assigned to the controller

Parameter machineData: pointer to the device state structure, the value in the structure will be updated after the call

Return value: see return value list

**int FMC4030\_API FMC4030\_Get\_Device\_Para(int id, unsigned char\* devicePara);**

Function: Get equipment setting parameters and each axis setting parameters, including ip, port number, lead, subdivision and other parameters

Parameter id: the id number assigned to the controller

Parameter devicePara: device parameter structure pointer, pass in the structure pointer, the value will be updated

Return value: see return value list

**int FMC4030\_API FMC4030\_Set\_Device\_Para(int id, unsigned char\* devicePara);**

Function: Set the equipment parameters and the parameters of each axis, please do not modify at will, to avoid equipment operation errors and damage to the equipment

---

Parameter id: the id number assigned to the controller

Parameter devicePara: pointer to device parameter structure, avoid being a null pointer

Return value: see return value list

## Ubuntu and other versions of Linux system function description

Similar to Windows, all function names and parameter order and number are the same。