Startup by Stopping Other Axis

Configurable number of axes	Four axes can be configured independently.

Description		When the axis specified as start-up condition stops, the motor of other axis start automatically.
Comments	1	When the motor start-up condition is "two or more axes stop"; If all the other axes are in stop status when any one axis specified as stop condition starts and stops, the axes to which the condition is configured start. Refer to "Configuration Example 1" for the configuration.
	2	Operating the interpolation motion continuously without changing the interpolation axis; Configure the next motion to pre-register. Refer to "Configuration Example 2" for the configuration.
	3	Operating the interpolation motion continuously by changing the interpolation axis in the middle of the motion Refer to "Configuration Example 3" for the configuration.
Configuration		<Configuration Example 1> **The motor start-up condition is "two or more axes stop"** If Y-axis (or X-axis) is in stop status while X-axis (or Y-axis) starts and stops, the motor of U-axis starts. The motor start-up condition for U-axis is "start when the motors of X and Y axes stop". Write the command to start the motor. The motor of U-axis does not start while the start command is written. X-axis : The first axis Y-axis : The second axis U-axis : The forth axis
	1	**Munti-function DLL**

```
HANDLE hDeviceHandle;
int nRet;
WORD wSync[4];

hDeviceHandle = MtnOpen("FBIMTN1", MTR_FLAG_NORMAL );
:

wSync[3] = MTR_X | MTR_Y;
nRet = MtnSetSync( hDeviceHandle, 0x08, MTR_START_MODE, wSync );
if(nRet != MTR_ERROR_SUCCESS) return -1;

nRet = MtnStartMotion( hDeviceHandle, 0x03, MTR_ACC, MTR_PTP );
nRet = MtnStartMotion( hDeviceHandle, 0x08, MTR_ACC, MTR_JOG );
```

<Configuration Example 2>

Operating the interpolation motion continuously without changing interpolation axis using the pre-register

Perform circular interpolation of X and Y axes, and as continuing the motion, perform linear interpolation 1 motion of X and Y axes.
Use simultaneous start signal as the condition to start the first circular interpolation motion.

1	Set the motor start-up conditions of X and Y axes to "start at the synchronous start signal input".
2	Configure the circular interpolation motion as follows.

	X-axis	Y-axis
Moving direction	CW direction	
Constant control of combining velocity	OFF	
Moving velocity	1000 pps	
Ending coordinate	1000	1000
Central coordinate	1000	0
Motion result	90 degree arc	

3	Write command to start the motor.
4	Configure the linear interpolation 1 motion in the table below for the next motion.

	X-axis	Y-axis
Mode	Linear interpolation 1	Linear interpolation 1
Constant control of combining velocity	OFF	OFF
Moving step count	1000	5000
Moving velocity	1000 pps	
Startup motion	Constant velocity start	Constant velocity start
Interpolation control axis	O	
Master/slave axis	Slave axis	Master axis

5	Write command again to start the motor.
6	The configuration for a series of motor motions is completed. To actually start the motor, output the simultaneous start signal.

* When the pre-register is full, the next motion cannot be written.

2	**Multi-function DLL**

```
HANDLE hDeviceHandle;
int nRet;
WORD wSync[4];
```

```
MTNARC Arc;
MTNLINE Line;
hDeviceHandle = MtnOpen("FBIMTN1", MTR_FLAG_NORMAL );
:

wSync[0] = MTR_SYNC_EXT;
wSync[1] = MTR_SYNC_EXT;
nRet = MtnSetSync( hDeviceHandle, 0x03, MTR_START_MODE, wSync );

Arc.wAxis = 0x03;
Arc.wClock = 299;
Arc.wMode = MTR_ARC_CW;
Arc.fSpeed = 1000;
Arc.lCenterX = 1000;
Arc.lCenterY = 0;
Arc.lEndX = 1000;
Arc.lEndY = 1000;

nRet = MtnSetMotionArc( hDeviceHandle, MTR_ARC_NORMAL, &Arc );
if(nRet != MTR_ERROR_SUCCESS) return -1;

nRet = MtnStartMotion( hDeviceHandle, 0x03, MTR_CONST, MTR_ARC );
if(nRet != MTR_ERROR_SUCCESS) return -1;

Line.wAxis = 0x03;
Line.wClock = 299;
Line.wMode = MTR_LINE;
Line.wAccMode = MTR_ACC_NORMAL;
Line.fLowSpeed = 100;
Line.fSpeed = 1000;
Line.dwAcc = 100;
Line.dwDec = 100;
Line.fSAccSpeed = 0;
Line.fSDecSpeed = 0;
Line.lStep[0] = 1000;
Line.lStep[1] = 5000;

nRet = MtnSetMotionLine( hDeviceHandle, MTR_LINE_NORMAL, &Line );
if(nRet != MTR_ERROR_SUCCESS) return -1;

nRet = MtnStartMotion( hDeviceHandle, 0x07, MTR_CONST, MTR_LINE );
if(nRet != MTR_ERROR_SUCCESS) return -1;

/* Outputting simultaneous start signal */
nRet = MtnOutputSync(hDeviceHandle, MTR_EXT_START );
```

<Configuration Example 3>

Operating interpolation motion of X, Y, Z axes continuously as follows

		X	Y	Z	Description

1	Center point	1000	0		X and Y axes: redius 1000, 90 degree circular interpolation motion
	End point	1000	1000	0	X axis: moving pulse cout 0, PTP motion
	Startup condition	Synchronous start	Synchronous start	Waiting for X and Y-axes stop	
2	Moving pulse count	1000		-5000	X and Z axes: linear interpolation 1
	Startup condition	Waiting for Z-axis stop		Continuous startup (No conditions)	

Use simultaneous start signal as the condition to start the first circular interpolation motion motion.

1	Configure as follows; Y and Z-axes motor start-up condition: "start at the synchronous start signal input", Z-axis motor start-up condition: "start when X and Y axes stop".
2	Configure circular interpolation for X and Y-axes. Configure PTP motion of moving pulse count 0 for Z axis.
3	Write command to start the motor.
4	Configure as follows; X-axis motor start-up condition: "start when Z axis stops", Y and Z-axes motor start-up condition: "No start condition".
5	Configure linear interpolation 1 motion for X and Z axes.
6	Write command again to start the motor.
7	The configuration for a series of motor motions is completed. To actually start the motor, output the simultaneous start signal.

* When the pre-register is full, the next motion cannot be written.

3	**Multi-function DLL**

```
HANDLE hDeviceHandle;
int nRet;
WORD wSync[4];
MTNARC Arc;
MTNLINE Line;
MTNMOTION Motion[3];
hDeviceHandle = MtnOpen("FBIMTN1", MTR_FLAG_NORMAL );
:
```

```
// ====== step 1 ==========================

wSync[0] = MTR_SYNC_EXT;
wSync[1] = MTR_SYNC_EXT;
wSync[2] = MTR_X | MTR_Y;
nRet = MtnSetSync( hDeviceHandle, 0x07, MTR_START_MODE, wSync );

Arc.wAxis = 0x03;
Arc.wClock = 299;
Arc.wMode = MTR_ARC_CW;
Arc.fSpeed = 1000;
Arc.lCenterX = 1000;
Arc.lCenterY = 0;
Arc.lEndX = 1000;
Arc.lEndY = 1000;

nRet = MtnSetMotionArc( hDeviceHandle, MTR_ARC_NORMAL, &Arc );
if(nRet != MTR_ERROR_SUCCESS) return -1;

nRet = MtnStartMotion( hDeviceHandle, 0x03, MTR_CONST, MTR_ARC );
if(nRet != MTR_ERROR_SUCCESS) return -1;

Motion[2].wClock = 299;
//Motion[2].wMode = MTR_PTP;
Motion[2].wAccMode = MTR_ACC_NORMAL;
Motion[2].fLowSpeed = 10;
Motion[2].fSpeed = 500;
Motion[2].dwAcc = 100;
Motion[2].dwDec = 100;
Motion[2].fSAccSpeed = 0;
Motion[2].fSDecSpeed = 0;
Motion[2].lStep = 0;

nRet = MtnSetMotion( hDeviceHandle, 0x04, MTR_PTP, Motion );

nRet = MtnStartMotion( hDeviceHandle, 0x04, MTR_CONST, MTR_PTP );

// ====== step 2 ==========================

wSync[0] = MTR_Z;
wSync[1] = MTR_NO;
wSync[2] = MTR_NO;
nRet = MtnSetSync(hDeviceHandle, 0x07, MTR_START_MODE, wSync);

Line.wAxis = 0x05;
Line.wClock = 299;
Line.wMode = MTR_LINE;
Line.wAccMode = MTR_ACC_NORMAL;
Line.fLowSpeed = 100;
Line.fSpeed = 700;
```

```
Line.dwAcc = 100;
Line.dwDec = 100;
Line.fSAccSpeed = 0;
Line.fSDecSpeed = 0;
Line.lStep[0] = 1000;
Line.lStep[2] = -5000;

nRet = MtnSetMotionLine( hDeviceHandle, MTR_LINE_NORMAL, &Line );
if(nRet != MTR_ERROR_SUCCESS) return -1;

/* Synchronous start signal output */
nRet = MtnOutputSync(hDeviceHandle, MTR_EXT_START );
```


[Top]