Motion Control Products

Application note
Tool changer

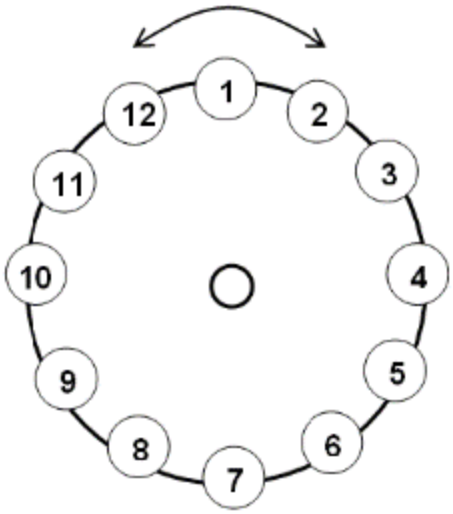
AN00109

Rev E (EN)

MicroFlex e190 can be ordered as an intelligent drive, which offers a high level Mint multitasking language, as well as powerful motion functions. This application note describes how a tool changer can be implemented on a MicroFlex e190 using a simple Mint task.

Introduction

Consider a tool carousel with 12 locations. If the PLC or host computer requests a tool change from 1 to 12, the carousel may move anti-clockwise. We can however program the tool changer to move from 1 to 12 in a clockwise direction, passing only 1 tool instead of 11. This would typically be known as ”taking the shortest path”.



Application example

Imagine we have an application based on a PLC which is controlling a milling machine as well as a tool changer. At some point the milling machine will need to change the tool, the PLC will send the number of the needed tool and a trigger signal to the tool changer drive via one of the supported fieldbus connections (i.e. ModbusTCP or EtherNet/IP). This data will be presented to the Mint program via the drive's netdata array.

First of all, the distance between tools must be calculated. In order to achieve that, the application note assumes that the tools are spaced equidistant from each other. In this example, assuming the motor is coupled directly to the tool carousel, the distance between tools is defined by the following equation:

fDistanceBetweenTools = (ENCODERRESOLUTION(0) * 4) / nNoOfTools

Where the encoder resolution multiplied by four is used to define the number of counts per revolution and then divided by the number of tools.

The program waits in a loop for the trigger status to become active, and calls a subroutine to perform the move. The tool position to move to (the tool index) is passed as a parameter to the subroutine. Once the subroutine is executed the application waits for the trigger to be deactivated.

```
Define niToolIndex = NETINTEGER (0)     'Define the index variable
Define niTrigger = NETINTEGER (1)        'Define the trigger variable
Define niInMotion = NETINTEGER (2)           'Define the motion in progress flag

Loop
  Pause niTrigger                  'Wait for the trigger to activate
  subPickTool (niToolIndex)              'Pick the tool based on tool index netdata from PLC
  Pause (Not(niTrigger))           'Wait for the trigger to deactivate
End Loop
```

The first thing the subroutine needs to do is to calculate the new target position for the new tool based on the tool index parameter, which is received from the main task; this number is going to be the absolute position in counts. The equation is as follows:

fTargetPos = nToolIndex * fDistanceBetweenTools

Next we calculate what the shortest path is and perform the move. In order to achieve this the WrapOffset command can be used. This command takes the original and final position, as well as upper and lower limits and works out the shortest path for the move. Before using the WrapOffset keyword the encoder channel must be wrapped to the total amount of counts in one revolution - this makes sure that the encoder value is always going to be in the correct range. Note that this technique can only be used with rotational moves:

nMoveDistance = WrapOffset (ENCODER(0), fTargetPos, 0, ENCODERRESOLUTION(0) * 4)

The nMoveDistance variable will be a relative distance from the actual position to the target position. With the move distance calculated we can perform the move and wait for completion:

```
'Set the motion in progress flag to active
niInMotion = _on

'Perform the relative move and wait until it is completed
MOVER(0) = nMoveDistance
GO(0)
Pause IDLE(0)

'Reset the motion in progress flag when the axis complete the move
niInMotion = _off
```

Another possible control method is to use digital inputs and outputs instead of netdata for the exchange of parameters between the PLC and MicroFlex e190.

For example, we can use inputs 0 to 3 to define a binary value for the tool index and input 4 for the trigger value. We can also define an output channel to be on when the axis is in motion. For the tool index we should create a mask to consider only the first four bits from the input bank and ignore the rest of them.

```
Const  _mkToolMask = 2#1111          'Mask for digital inputs 0-3
Define ipTrigger = INX (4)            'Define the trigger input
Define opInMotion = OUTX (1)          'Output for in motion flag to PLC
```

In the main loop the parameter passed to the subroutine (tool index) should be calculated through the logic equation as follows:

```
Loop
  Pause (ipTrigger)                          'Wait for a trigger to activate
  subPickTool (IN(0) And _mkToolMask)          'Pick tool based on digital inputs 0-3 using the mask to take only the first 4
inputs
  Pause (Not(ipTrigger))                     'Wait for trigger to deactivate
End Loop
```

Two Mint programs are included with this application note, one for using netdata, another for I/O. Both programs are ready to run on a MicroFlex e190.

Contact us

For more information please contact your
local ABB representative or one of the following:

new.abb.com/motion
new.abb.com/drives
new.abb.com/drives/drivespartners
new.abb.com/PLC