Related Applications or Terminology

- Preset positions
- Indexing
- Move to position

Overview

Flex+Drive II provides the powerful feature of preset positions. This can be used for indexing applications such as tool changers.

Flex+Drive II also introduces the powerful capability of Mint programming. This application note describes how a tool changer can be implemented on the Flex+Drive II using a simple Mint program. If used with a controller supporting multi-tasking such as any of the NextMove family or MintDrive II, the tool changer program can be implemented as a separate task.

Why use Mint?

Mint can be used in place of preset positions to provide shortest path. Consider a tool carousel with 12 locations. If the PLC or host computer requests a tool change from 2 to 12, the preset positions will move anti-clockwise. We can however program the tool changer to move from 2 to 12 in a clockwise direction, passing only 1 tool and not 9.
Moving to position

The application assumes that the tools are spaced equidistant from each other. In this example, this is defined by the following constants:

```
Const _nNoOfTools = 12  'Number of tools in the machine
Const _nDistanceBetweenTools = 1000  'Distance between each tool (counts)
```

Each tool is defined to be 1000 motor counts apart. The total revolution of the carousel will therefore be 12,000 motor counts (12 tools spaced 1000 counts apart).

The application waits for the trigger input (input 4 in this example), and calls a subroutine to perform the move. The subroutine is passed the tool position to move to (the tool index) as a parameter. In this example, this is taken from the digital inputs IN0 to IN3. The trigger input for the move is taken from IN4.

```
Const _mkToolMask = 01111  'Used to mask off inputs 4-7
Define ipTrigger = INx:4
  'Wait here in a loop to allow the program to run
Loop
  Pause ipTrigger  'Wait for a trigger to activate
  subPickTool IN & _mkToolMask  'Pick tool based on digital inputs 0-3
  Pause !ipTrigger  'Wait for trigger to deactivate
End Loop
```

This code could easily be altered to cope with different inputs or even values being passed via the Mint comms array.

The ENCODER keyword is used to read back the position of the motor (regardless of whether it is encoder, resolver or absolute). ENCODER is used since its value can be wrapped at a known position. The wrap position is set using ENCODERWRAP and in this example has been set to:

```
'Number of encoder counts per rev of the tool changer
ENCODERWRAP(0) = _nNoOfTools * _nDistanceBetweenTools
```

This ensures that reading ENCODER at any time will return an absolute position within 1 revolution of the tool changes.

The first thing the subroutine needs to do is to calculate the distance the tool changer needs to rotate to move to the new tool:

```
nDistanceToMove = (nToolIndex * _nDistanceBetweenTools) – ENCODER(0)
```
Next we calculate what is the shortest path and perform the move:

'The required move distance need never be greater than half a 'revolution, so add or subtract 1/2 a tool revolution

If nDistanceToMove > (_nDistanceBetweenTools * _nNoOfTools) / 2 Then
    nDistanceToMove = nDistanceToMove - (_nDistanceBetweenTools * _nNoOfTools)
Else If nDistanceToMove < -( _nDistanceBetweenTools * _nNoOfTools) / 2 Then
    nDistanceToMove = nDistanceToMove + (_nDistanceBetweenTools * _nNoOfTools)
End If

'Perform a relative move to the tool location and wait for completion
MOVER(0) = PositionToMove
GO(0)
PAUSE IDLE(0)

The complete program example

The full program is shown on the following page. In this example an index of zero is defined as the home position and calls the Mint Home keyword. This would be changed to suit the application.

For interfacing to a PLC, an output is set during motion. This is defined as:

    Define opInMotion = OUTX(1)    'Output for in motion flag to PLC

In order for this program to operate on Flex+Drive II and MintDrive IID, the PLC lines operating the outputs need to be switched off.
'Constant declarations
Const _nNoOfTools = 12  'Number of tools in the machine
Const _nDistanceBetweenTools = 1000  'Distance between each tool (counts)
Const _mkToolMask = 01111  'Mask for digital inputs 0-3

'Macro declarations
Define ipTrigger = INX(4)  'Define the trigger input
Define opInMotion = OUTX(1)  'Output for in motion flag to PLC

'Clear any outstanding errors and enable the drive
CANCEL(0)
DRIVEENABLE(0) = 1

'Number of encoder counts per rev of the tool changer
ENCODERWRAP(0) = _nNoOfTools * _nDistanceBetweenTools

'Wait here in a loop to allow the program to run
Loop
  Pause ipTrigger  'Wait for a trigger to activate
  subPickTool IN(0) & _mkToolMask  'Pick tool based on digital inputs 0-3
  Pause !ipTrigger  'Wait for trigger to deactivate
End Loop

End

'Place the drive configuration parameters here
'This will be called on program start-up
Startup
  SCALEFACTOR(0) = 1  'Assume scaling is in counts
  INPUTMODE(0) = 0  'Setup the I/O – all level triggered
  HOMEINPUT(0) = 5  'The digital input used to mark the home position
  HOMESPEED(0) = 500
  HOMEBACKOFF(0) = 2  'Back-off at half the home speed
End Startup

'This subroutine serves two purposes, firstly to home the tool picker
'then a tool index of zero is supplied, and secondly to pick tools
'then supplied with a non-zero tool index.
Sub subPickTool(ByVal nToolIndex As Integer)
  Dim nDistanceToMove

  'Check that the tool index is within range
  If nToolIndex > _nNoOfTools Then Exit Sub

  opInMotion = 1  'Turn on the "in motion" signal

  If nToolIndex = 0 Then
    HOME(0) = _hmNEGATIVE_SWITCH
  End If
End Sub
Pause IDLE(0)
POS(0) = 0
ENCODER(0) = 0

Else

nDistanceToMove = (nToolIndex * _nDistanceBetweenTools) – ENCODER(0)

'The required move distance need never be greater than half a
'revolution, so add or subtract 1/2 a tool revolution
If nDistanceToMove > (_nDistanceBetweenTools * _nNoOfTools) / 2 Then
    nDistanceToMove = nDistanceToMove – (_nDistanceBetweenTools * _nNoOfTools)
Else If nDistanceToMove < -( _nDistanceBetweenTools* _nNoOfTools) / 2 Then
    nDistanceToMove = nDistanceToMove + (_nDistanceBetweenTools * _nNoOfTools)
End If

'Perform a relative move to the tool location and wait for completion
MOVER(0) = nDistanceToMove
GO(0) : PAUSE IDLE(0)
End If

oInMotion = 0  'Turn off the "in motion" signal
End Sub