Motion Control Products

Application note
Product labelling

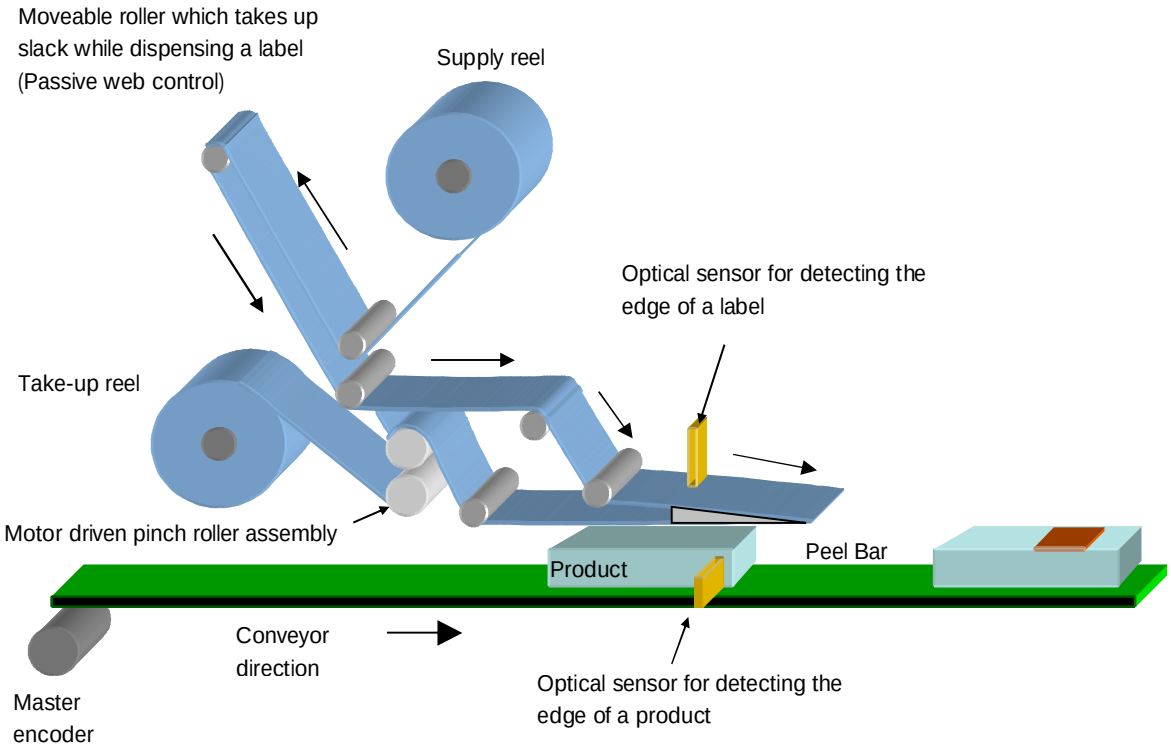
AN00103

Rev E (EN)

This document demonstrates how the Mint FLY command could be utilised as the principle motion control technique for an application such as product labelling. For more detailed information about using ABB motion products in applications like this, please contact your local ABB office.

Overview

The illustration below shows the layout of a typical single head labelling machine.



Products pass along a conveyor belt which may be independently controlled by an ABB ACS inverter. An encoder is fitted to the conveyor and is used to provide a master reference for product speed and position. The input encoder resolution will be selected for the accuracy required by the labeller. Increasing the number of counts per revolution can improve the final accuracy attainable by the system.

The MicroFlex e150 connects to a resolver or digital encoder based motor thereby allowing the master encoder (conveyor) signal to be fed into connector X8 on the MicroFlex e150 as an auxiliary encoder input (via the MicroFlex encoder breakout option card, part number OPT-MF- 200).

The products are detected by a fast-acting photoelectric sensor. The signal from this sensor is connected to a MicroFlex e150 input assigned for use as a fast (latch) input event. This can be used to call a designated event which will store the latched position of the conveyor. The Mint program will use a delayed version of these stored positions as “trigger values” to start the label application cycle for each product that is detected on the conveyor.
A servomotor is used to drive the pinch rolls, which in turn pull the label stock off of the supply reel. A peel bar (or beak) ensures that the labels are removed from their backing at a fixed point within the process.

The label stock must ramp up to a speed that matches the speed of the product passing by. Once at speed it must remain locked to the same speed as the product whilst the label is being applied. Once applied the label stock must then ramp down to a stop, whilst at the same time positioning the next label on the peel bar ready for the next labelling cycle.

If the product conveyor is moving at a fixed speed the Mint incremental move commands INCA/INCR could be used. However, if the conveyor speed is variable then the requirement to match speeds over known distances makes this application ideal for the Mint FLY command.

The cycle starts as soon as the product sensor is activated by a product passing by. Because the sensor is located ahead of the peel bar the program will command that the product travels a further specified distance before the label stock is accelerated up to synchronous speed, this is often called the ‘product delay’.
It is common to make this delay adjustable by the operator as this provides a method of setting the label position on the product. An ABB CP series HMI panel can be connected to provide the required user interface.

Worked example

A labelling machine similar to that illustrated earlier has the following characteristics:

- Master encoder with 10000 counts per revolution (i.e. a 2500 line encoder) is mounted on the back of the induction motor driving the conveyor. One revolution of this motor moves the conveyor 600mm. This encoder is wired to the MicroFlex e150 as encoder channel 2
- An ABB BSM R-series servo motor with a Smartabs digital encoder (resolution of 131072 counts per motor revolution) driving a 3:1 speed reduction gearbox, is mounted to drive a set of pinch rollers (nips) of 60mm diameter.
- Nominal label pitch to be applied is 75mm (the label pitch is the measure from the leading edge of a label to the leading edge of the following label)
- The product detection sensor is located 30mm away from the edge of the peel bar

A suitable scale for both the conveyor encoder (master) and nip roller axis (slave) should be chosen, in this case we are going to scale our values to be linear mm.

Conveyor scaling:
Conveyor scale factor is found from the number of encoder counts for a known distance…there are 10000 counts in 600mm of linear travel so…

```
ENCODERSCALE(2) = 10000 / 600
```

Nip roll scaling:
One motor revolution will move the nip rolls 1/3 of a revolution (where one revolution is Pi times the roll diameter of linear movement), therefore the scale for this axis is calculated from…

```
SCALEFACTOR(0) = 131072 / ((1/3) * _Pi * 60)
```

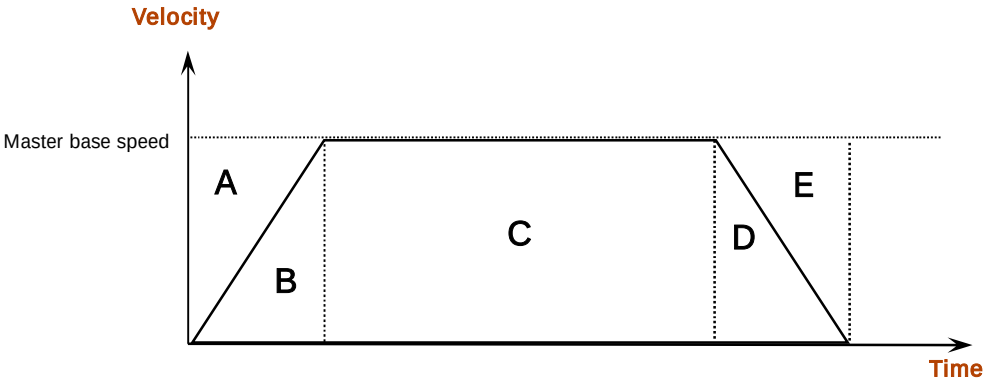
We can set the nip roll axis to use the conveyor encoder as both its master reference and the source for triggering motion using the following code:

```
MASTERSOURCE(0) = _msENCODER
MASTERCHANNEL(0) = 2
TRIGGERSOURCE(0) = _tsENCODER
TRIGGERCHANNEL(0) = 2
```

By setting a TRIGGERVALUE for the nip roll axis that is equal to the latched conveyor encoder value plus an offset (the product delay) we can ensure any motion loaded on the servo axis does not start until the correct point in time (i.e. so that the label is applied to the product in the correct position).

To apply a label the BSM servo motor will ramp up to match the label stock speed to the conveyor speed over a certain (often fixed) displacement of the label itself. This distance is dependent on the mechanical design of the labeller (e.g. properties such as the gap between the label beak and the product and the design of the label wipes) and discussion of this is beyond the scope of this document. For this example, we will assume this label acceleration distance is 6mm. This is our acceleration segment of the move and will be our first known FLY segment value.

In any flying shear application, the relationship between the distance travelled by the master axis (MASTERDISTANCE) and the distance travelled by the slave axis (FLY) is as shown in the figure below



This figure illustrates that for the slave axis to accelerate up to synchronous speed with the master axis it must travel the distance denoted by area **B**, whilst at the same time the master axis travels the distance denoted by **A+B.** It can also be clearly seen that **B = A** (or put another way, the slave travels exactly half the distance travelled by the master).
Whilst at synchronous speed the master and slave axes travel the same distance **C.**
To decelerate to a stop the slave again travels half the distance **D** travelled by the master axis **D + E.**
We have already stated that our slave axis will accelerate over a label travel of 6mm, hence we can now see that the master distance travelled over this same period must be 12mm (i.e. twice the label travel).

This can also be worked out using the generic formula for Mint flying shears segments:

$$FLY = \frac{(\text{initial ratio} + \text{final ratio})}{2} \times MASTERDISTANCE$$

…which re-arrange results in…

$$MASTERDISTANCE = 2 \times FLY / (\text{initial ratio} + \text{final ratio})$$

When we start the label moving our initial speed ratio is 0. As we are aiming to match the product speed our final speed ratio is 1), hence why the MASTERDISTANCE can be found from 2 x FLY.

It is always good practice to write a program using declared Mint constants and variables rather than hard coded numbers as this allows for a more flexible program.

```
Const _axLabelFeed As Integer = 0
Const _fAccelDistance As Float = 6

MASTERDISTANCE(_axLabelFeed) = _fAccelDistance * 2      ' Accelerate up to product speed
FLY(_axLabelFeed) = _fAccelDistance                     ' Do this over 6mm of label travel
```

The remainder of the profile is made up from two further FLY segments.

The first segment is at synchronous speed (where the distance travelled by the slave equals the distance travelled by the master) and the final segment brings the slave axis to a halt (where the distance travelled by the slave is half the distance travelled by the master again).
We will bring the slave axis to a halt over 6mm of travel of the label stock (i.e. an identical segment to that used to start the move initially). This means we use the same calculation as we used for the acceleration segment.

The synchronous distance travelled by the slave must be the total label pitch minus the distance travelled to accelerate up to speed and the distance travelled to decelerate down to a stop. We know that our label pitch is 75mm and that the slave will travel 6mm for the acceleration segment and a further 6mm for the deceleration segment. Therefore, the synchronous segment label travel can be found from:

75mm - 6mm – 6mm = 63mm. Again, it is best practise to use variables in the code to dynamically calculate this distance rather than hard code a value.

If we assume the label pitch is stored in a variable called “fLabelPitch” we can now construct our final two flying shear segments….

```
MASTERDISTANCE(_axLabelFeed) = fLabelPitch – (2 * _fAccelDistance)
FLY(_axLabelFeed) = fLabelPitch – (2 * _fAccelDistance)

MASTERDISTANCE(_axLabelFeed) = _fAccelDistance * 2
FLY(_axLabelFeed) = _fAccelDistance
```

Once the FLY segments are loaded and the TRIGGERVALUE set. The program then needs to check that the move is possible before the product passes the trigger value. If the program fails to trigger the move in time the label feed will not occur until the conveyor encoder value comes back around to the trigger value once more.

```
Repeat
    fTriggerDist = Wrap(TRIGGERVALUE(_AxLabelfeed) – ENCODER(_Conveyor) / 2, -ENCODERWRAP(_Conveyor)  / 2)
Until fTriggerDist < 0
```

Once the distance to trigger value is less than 0 the program can check MOVEBUFFERSTATUS to make sure that the move is in progress. If the move has been correctly triggered the move buffer will then be emptied ready for the next set of move segments to be loaded. If however the moves were not triggered the program will issue a CANCEL command to clear the moves in the move buffer ready for the next product.

That concludes our motion profile for this example. In an actual product labelling application, labels would not be uniformly pitched on the backing material and therefore some form of label pitch detection (via a second fast input used to latch the label edges) would have to be used in order to measure the actual pitch, and further Mint code produced to utilise the actual pitch for each label feed rather than a nominal pitch. This information may need to be “queued” if there is more than one label pitch between the label sensor and the beak end. Similarly, it may be necessary to “queue” the product position data if the product sensor is more than one product pitch away from the label application point.

Other factors may complicate the program further, for example the product’s shape may dictate that a ratio other than 1:1 is used.

If you require further help or support with a labelling or similar application please contact your local ABB technical support office.

This application note is supplied with a sample Mint program which will work with a MicroFlex e150 that has the auxiliary encoder input (encoder channel 2) connected to a RS422 differential line driver encoder. The ABB MicroFlex e150 demonstration unit (part number DEMO-E150-001) is ideal for this purpose, please contact your local ABB sales office for pricing information.

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