

## Flying Cutoff with the PTi210 (Unregistered Material)

### Objective

Demonstrate setting up the PTi210 to perform a flying cutoff on unregistered material.

### Overview

Many applications involve cutting a given material to a desired length by moving the material, stopping the material, performing the cut, and repeating the process. These applications are often referred to as “Cut to Length” applications. It was found that the throughput of this process could be improved if the material did not need to start and stop for each cut. To do this, the cutting mechanism must be able to match speed with the material while it performs the cutting action. With the use of a master encoder and a servo to drive the cutting mechanism, the user can accurately cut products to a desired length, without stopping the material, and achieve very high production rates. This type of application is called a “Flying Cutoff” or “Flying Shear”. Figure 1 below shows a simple flying cutoff system.

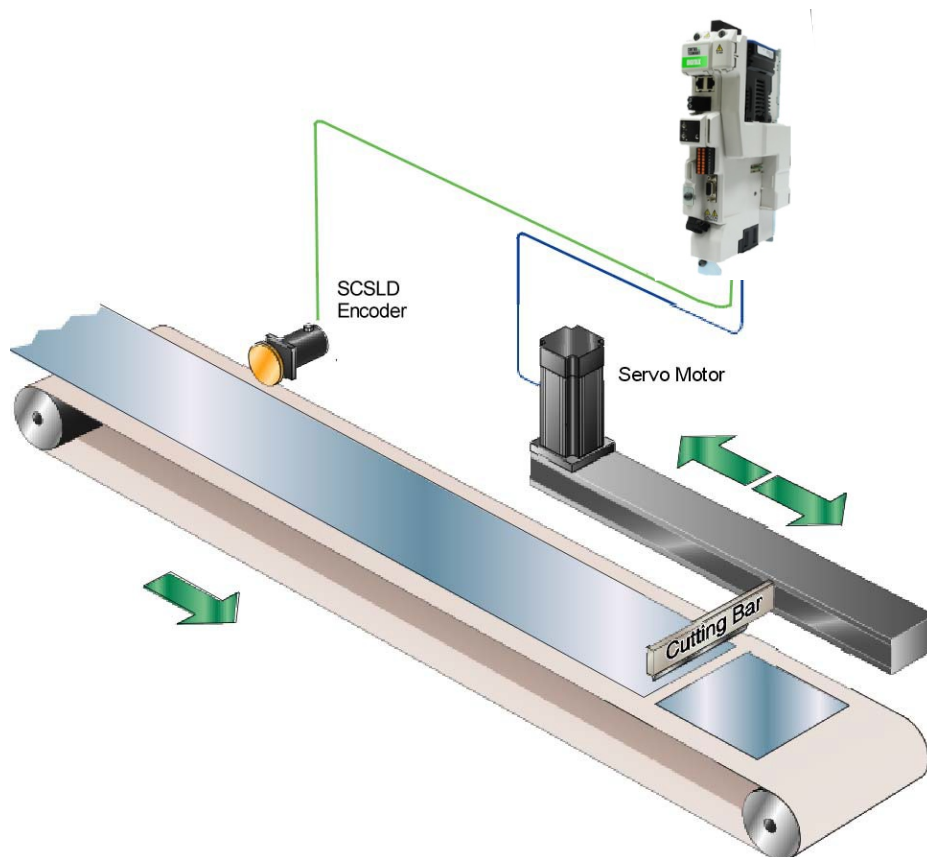
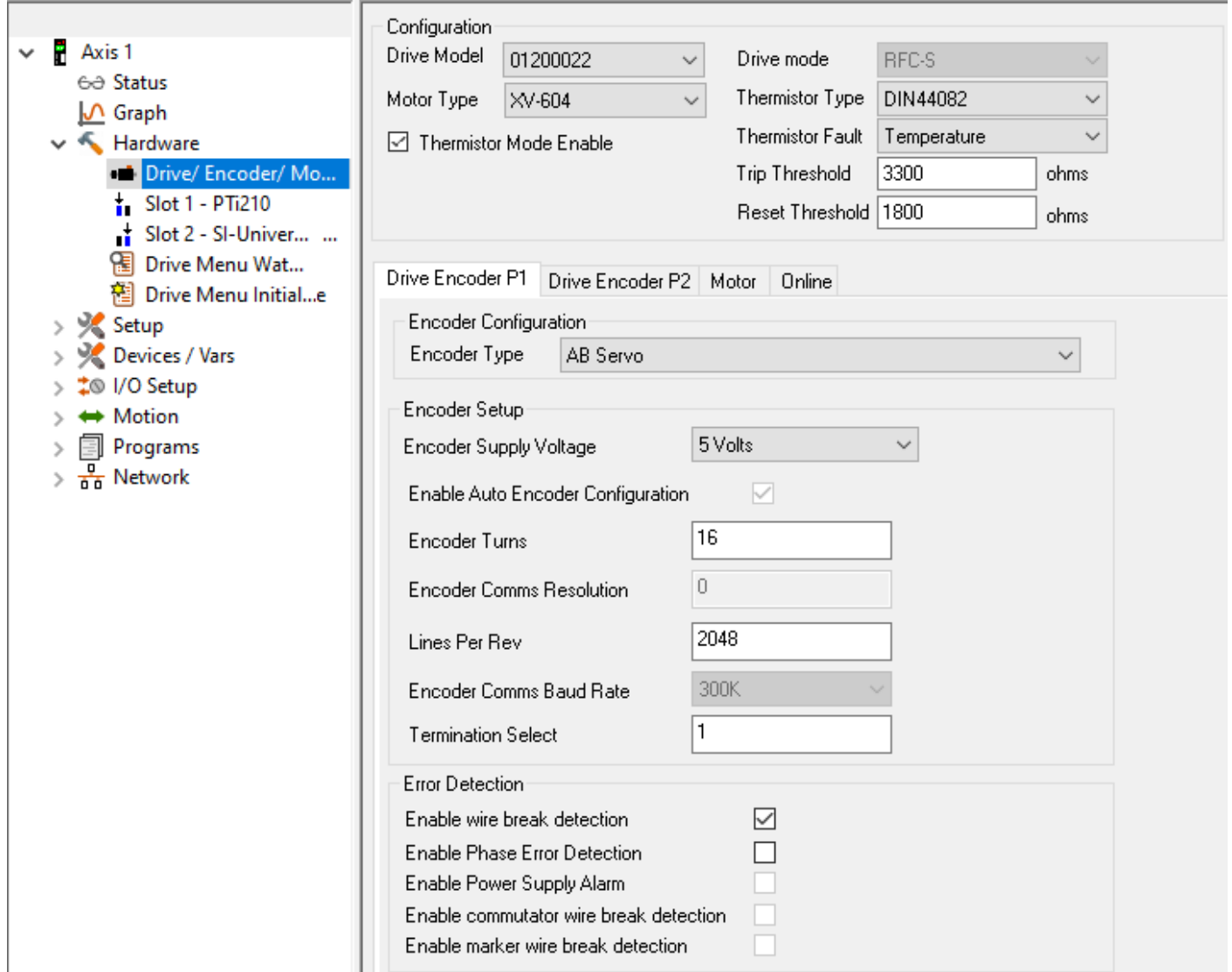


Figure 1 – Example Flying Cutoff System

## Solution Summary

Use a PLS (Programmable Limit Switch) and Synchronized Indexes in the PTi210 to create a flying cutoff program.

## Hardware – Drive/Encoder/Motor Setup



Configuration

Drive Model: 01200022 | Drive mode: RFC-S

Motor Type: XV-604 | Thermistor Type: DIN44082

Thermistor Mode Enable | Thermistor Fault: Temperature

Trip Threshold: 3300 ohms

Reset Threshold: 1800 ohms

Drive Encoder P1 | Drive Encoder P2 | Motor | Online

Encoder Configuration

Encoder Type: AB Servo

Encoder Setup

Encoder Supply Voltage: 5 Volts

Enable Auto Encoder Configuration:

Encoder Turns: 16

Encoder Comms Resolution: 0

Lines Per Rev: 2048

Encoder Comms Baud Rate: 300K

Termination Select: 1

Error Detection

Enable wire break detection:

Enable Phase Error Detection:

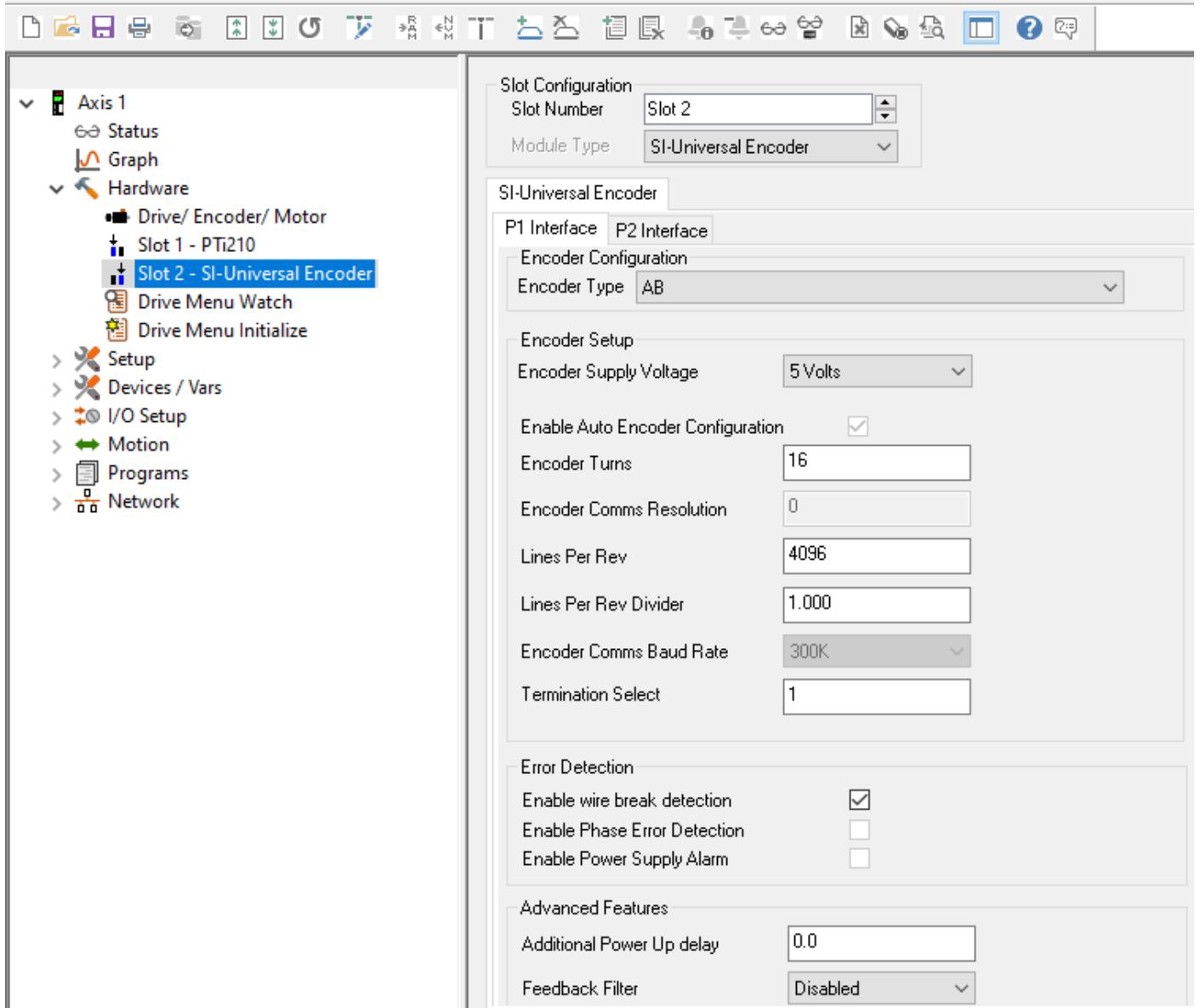
Enable Power Supply Alarm:

Enable commutator wire break detection:

Enable marker wire break detection:

Figure 2 – Drive/Encoder/Motor View

## Hardware – Slot 2 Setup



The screenshot displays the 'Slot 2 Configuration View' in PowerTools Studio. The left sidebar shows a tree view under 'Axis 1' with 'Hardware' expanded to 'Slot 2 - SI-Universal Encoder'. The main panel is titled 'Slot Configuration' and shows 'Slot Number' as 'Slot 2' and 'Module Type' as 'SI-Universal Encoder'. Below this, the 'SI-Universal Encoder' configuration is shown with two tabs: 'P1 Interface' and 'P2 Interface'. The 'Encoder Configuration' section shows 'Encoder Type' as 'AB'. The 'Encoder Setup' section includes 'Encoder Supply Voltage' (5 Volts), 'Enable Auto Encoder Configuration' (checked), 'Encoder Turns' (16), 'Encoder Comms Resolution' (0), 'Lines Per Rev' (4096), 'Lines Per Rev Divider' (1.000), 'Encoder Comms Baud Rate' (300K), and 'Termination Select' (1). The 'Error Detection' section includes 'Enable wire break detection' (checked), 'Enable Phase Error Detection' (unchecked), and 'Enable Power Supply Alarm' (unchecked). The 'Advanced Features' section includes 'Additional Power Up delay' (0.0) and 'Feedback Filter' (Disabled).

Figure 3 –Slot 2 Configuration View

Since a Master Encoder is required in the Flying Cutoff application, we need to add a SI-Universal Encoder module to the system to wire the encoder to. We need to define which slot in the drive that the encoder module will be plugged into as well as the PTi210 module. For this example, we have installed the Pti210 module in Slot 1 and the SI-Universal Encoder Plus module into Slot 2. Select these two modules in PowerTools Studio under the Hardware tree. (Note: We are using a Universal Encoder Module in this case since for this example our main motor has a quadrature incremental encoder with commutation outputs. This feedback device signals takes up most of the available pins on the 15-pin main feedback port. Otherwise, with other feedback types, it may be possible to wire in the master encoder into the main drive feedback port instead. In that case, the Universal Encoder Module is not required). Please refer to the feedback ports wiring diagram in the associated drive manual for more information.

**Encoder Type** – This parameter defines what type of encoder is being used as the feedback device. This encoder could either be an external stand-alone encoder, or it could be an encoder from an upstream motor. If using a standard stand-alone encoder from Control Techniques, this should be set to Quadrature Incremental. If the feedback device is an upstream motor, the Encoder Type should be set to Quadrature Incremental w/ Commutation Outputs. All encoder types supported by the drive can be found in this listbox.

**Encoder Supply Voltage** – Select the correct supply voltage for the encoder from this list. Valid selections are 5V, 8V, and 15V. For standard quadrature encoders, 5V should be selected.

**Enable Auto Encoder Configuration** – This checkbox is available only when using an Absolute encoder with a communications protocol. The encoder can be interrogated on power up for Encoder Turn Bits, Lines Per Rev, and Encoder Comms Resolution automatically.

**Encoder Turns** – This is an Absolute encoder type term. This is how many revolutions will be recorded before the Revolution Counter rolls over. The parameter is the number of bits of resolution and should be provided by the encoder manufacturer.

**Encoder Comms Resolution** – This is an Absolute encoder type with a communications protocol term. This parameter defines the maximum resolution of the absolute position of the encoder being transmitted.

**Lines Per Rev (pre quadrature)** – The user should enter the pre-quadrature resolution of the feedback device here. For an encoder such as a SCSLD-4, enter 3000. For a Control Techniques MG/MH/NT/XV motor enter 2048. For most Control Techniques Unimotors, enter 4096. Check the Unimotor nameplate, since it's possible it may have a 2048 device installed.

**Lines Per Rev Divider** – This is used to scale the equivalent lines per revolution of incremental and SINCOS encoders, without comms, on rotary motors, and all but comms only encoders on linear motors (servo encoders must have the same number and pitch of poles as the motor) The equivalent line per revolution parameter is divided by the Lines Per Rev Divider. This is most often used when an encoder is used with a linear motor where the number of counts or sine waves per pole is not an integer.

**Encoder Comms Baud Rate** – This is an Absolute encoder type with a communications protocol term. This is the user-defined baud rate of the Absolute encoder.

**Termination Select** – P1 Termination Select is used to enable or disable the terminations on the position feedback interface inputs. This function depends on the position feedback device type selected. Refer to the drive manual or Online Help in PowerTools Studio for more information.

## Setup – User Units

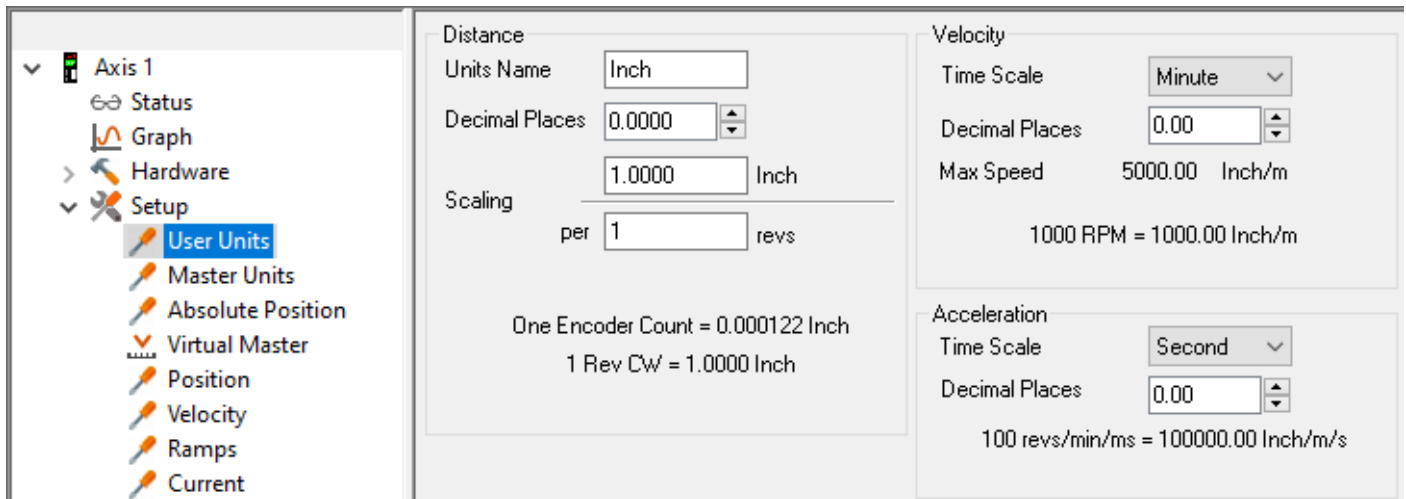


Figure 4 – User Units View

To perform the flying cutoff application, the PTi210 must be synchronized to the motion (or velocity) of a master signal. Typically, the master signal is either an external encoder riding on the material that is being cut (see Figure 1), or an upstream servo system that is conveying the material. In both cases, the User Units and Master Units must be setup accordingly. Figure 4 above shows the User Units setup view from PowerTools Studio software.

The parameters that require setup are as follows:

**Distance Units Name** – This is a character string, up to seven characters in length, that is used as the name for the user selected units of distance. For this example, the PTi210 program tool uses the Units Name of Inches.

**Scaling** - The scaling parameters are used to scale the users desired units back into real life units of revolutions of the motor. There are two parameters associated with the scaling factor: Characteristic Distance, and Characteristic Length. The Characteristic Distance is defined as the number of user units traveled per Characteristic Length. Characteristic Length is the number of whole motor revs it takes to travel the Characteristic Distance. The Characteristic Length must ALWAYS be specified in a whole number of motor revs.

Example:

- 1) The motor drives a cutting mechanism that travels 1.0 Inches per motor revolution.
- 2) The desired User Distance Units are Inches.

The scaling parameters would be as follows:

$$\text{Scaling} = \frac{\text{Characteristic Distance}}{\text{Characteristic Length}} = \frac{1.000 \text{ Inches}}{1 \text{ Revs}}$$

## Setup – Master Units

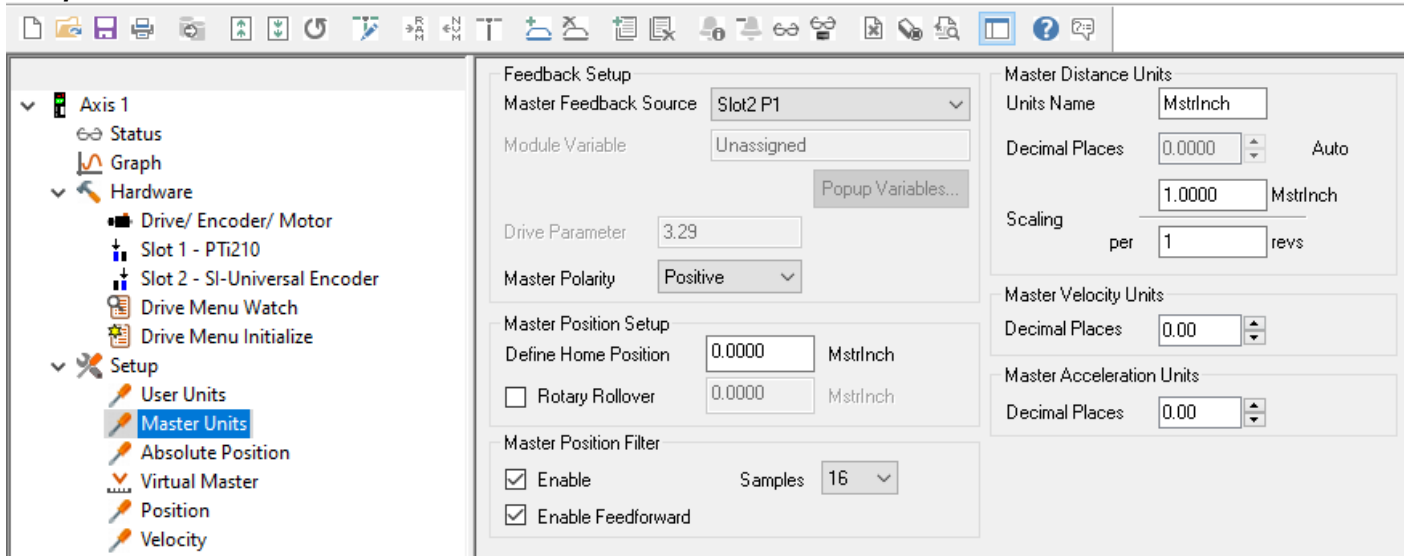


Figure 5 – Master Units View

The Master Units view is used to configure the necessary parameters for the master encoder (or the encoder that measures the material to be cut). Figure 5 above shows the Master Units setup view from PowerTools Studio. The parameters that require setup are as follows:

**Master Feedback Source** – This parameter is used to define where the master encoder is connected to the system. In this example, the master encoder will be connected to a SI-Universal Encoder module in Slot2 and connected to the P1 terminals. DO NOT set this parameter to be the same value as the Motor Feedback Source parameter found on the Setup view in PowerTools Studio.

**Encoder Polarity** – This parameter defines whether the position feedback increases when the encoder counts up, or whether the position feedback decreases when the encoder counts up. This effectively allows the user to change the polarity of the encoder so that the encoder can be physically mounted in any location. This can be set to Positive or Negative. If set to Positive, then increasing counts from the master correspond to positive movement of the master. If set to Negative, then increasing counts from the master correspond to negative movement of the master. This example program has the Polarity set to Positive.

**Units Name** – This is simply a text string that will be used as the name for Master Distance Units. The string can be up to 7 characters long. For this example, the PTi210 program tool uses the Units Name of Mstrlnch.

**Decimal Places** – This parameter determines the number of digits used after the decimal place in all master distance or sync distance parameters. This is automatically calculated based on the resolution of the master encoder.

**Scaling** - The scaling parameters are used to scale the users desired units back into real life units of encoder revs. There are two parameters associated with the scaling factor: Master Characteristic Distance, and

Master Revs. The Master Characteristic Distance is defined as the number of user units traveled per Master Revs.

This example uses Master Distance Units of MstrInch, and since the master encoder is 2048 lines/rev, the Scaling is set to 1MstrInch per 1 Rev.

$$\text{MstrInch Scaling} = \frac{\text{Characteristic Distance}}{\text{Master Counts}} = \frac{1.0000}{1 \text{ Revs}}$$

**Master Velocity Decimal Places** – This parameter determines the number or digits used after the decimal place in all synchronized motion velocity parameters. In this example application, the decimal places are set to two, so the velocity ratio has a resolution of 0.01 Inches / MstrInch.

### Setting the Feedback Source Parameters

The user must correctly define where the different feedback devices are connected to the system. To define where the Motor feedback is connected, the user must set the Motor Feedback Source parameter, which can be found on the Setup view (shown below). To define where the Master Encoder feedback is connected, the user must set the Master Feedback Source parameter, which is found on the Master Units view.

The Motor Feedback Source and the Master Feedback Source should NOT be set to the same setting. Doing so can result in incorrect resolution of the feedback and command values.

### Setup – PLS

#	Source	ON Point	OFF Point	Direction	Rotary Enable	Rollover Point
0	MasterPosnFeedba...	1.0000 MstrInch	2.5000 MstrInch	Both	<input checked="" type="checkbox"/>	0.0000 MstrInch
1	MotorPosnFeedba...	0.0000 Inch	0.0000 Inch	Both	<input type="checkbox"/>	0.0000 Inch
2	MotorPosnFeedba...	0.0000 Inch	0.0000 Inch	Both	<input type="checkbox"/>	0.0000 Inch
3	MotorPosnFeedba...	0.0000 Inch	0.0000 Inch	Both	<input type="checkbox"/>	0.0000 Inch

Figure 6 – PLS Setup View

The PLS is a Programmable Limit Switch that is used to activate a signal when the position of a selected source parameter is between two values called the ON Point and OFF Point. Since this application tool is developed to work with unregistered material (no registration mark printed on the material), we will simulate a registration mark by activating a PLS each time the desired cut length passes.

Figure 6 above shows the PLS setup view from Power Tools Pro. The parameters that require setup are as follows:

PLS.#.Status will be active if: PLS.#.OnPosn ≤ Source Value < PLS.#.OffPosn

**Source** – The source of a PLS can be assigned to the motor axis (MotorPosnFeedback or MotorPosnCommand) or a master synchronization signal (MasterPosnFeedback). In this example the MasterPosnFeedback (from the master encoder as seen in Figure 1) was selected.

**ON Point** – PLS.#.Status will be active when the selected source position is between the PLS.#.OnPosn and the PLS.#.OffPosn. All ON/OFF positions are defined in user units. In this example the ON Point was arbitrarily chosen to be 1.0000 MstrInch and the OFF Point is set to 2.5000 MstrInch. This means that the PLS will be on for a duration of 1.5 Master Inches.

**OFF Point** – PLS.#.Status will be OFF when the selected source position is less than the PLS.#.OffPosn and greater than or equal to the PLS.#.OnPosn. All ON/OFF positions are defined in user units. In this example the OFF Point was arbitrarily chosen to be 2.5000 MstrInch.

**Direction** – This parameter specifies the direction of motion that will activate a PLS.#.Status change. If set to Both, as in this example, the PLS will activate regardless of whether master is moving in the positive or negative direction. If set to Plus, the PLS will activate only when the master is moving in the positive direction. If set to minus, the opposite is true.

**Rotary Enable** – This parameter is used to enable the RotaryRolloverPosn for the PLS. What this means is that when Rotary Enable is active, the PLS will repeat itself every rotary length defined by the Rollover Point (or PLS.#.RotaryRolloverPosn)

**Rollover Point** – This parameter is the absolute position of the first repeat position for this PLS. When enabled, as in this example, it causes the PLS to reset to zero every time the rollover point is reached. The repeating range begins at an absolute position of zero and ends at the RotaryRolloverPosn. In this example the Rollover Point is very crucial in the fact that it is the actual desired cut length. The PLS Rollover point could be modified from an MMI to change the physical part length. For real life application situations, this number may have limits based on desired throughput rates and the physical load seen at the motor. In this example, the PLS Rollover Point is set within the user program, so the actual value set on the PLS Setup View is irrelevant.



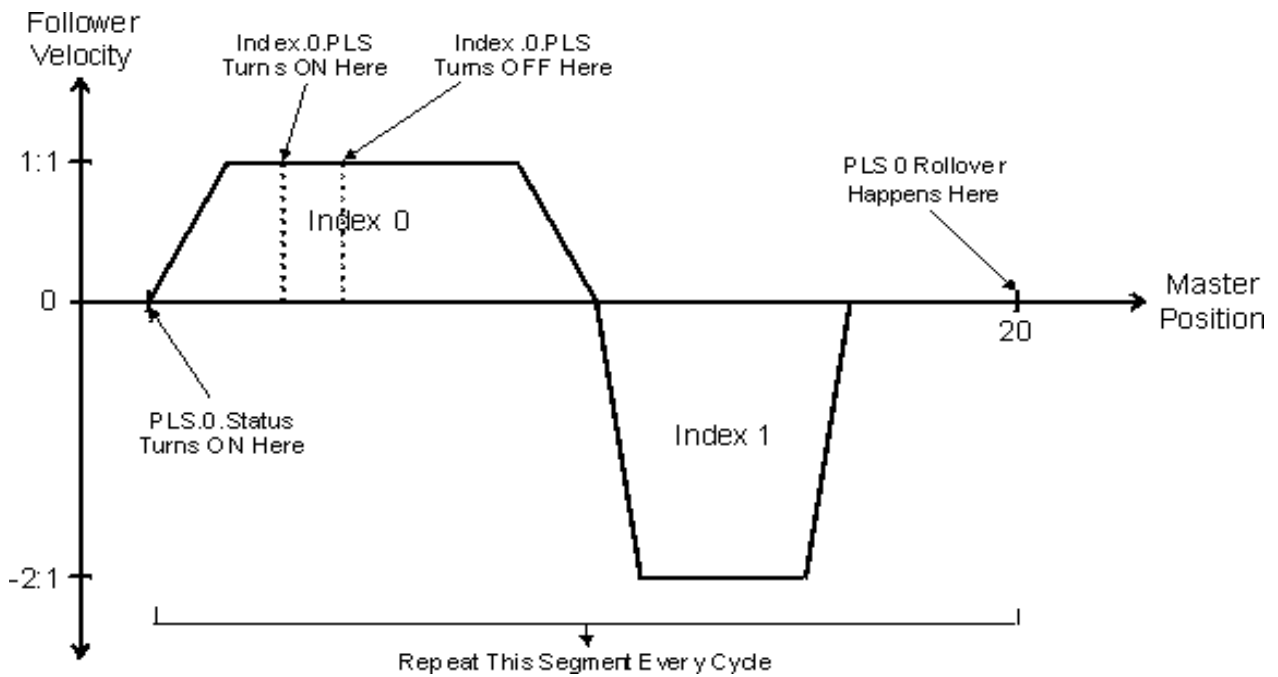


Figure 7 – Flying Cutoff Motion Profile

### I/O Setup – Assignments

Source	Assigned to	Polarity	Destination	Set From	Polarity
Index			Capture		
Cut_Move			Capture0		
Index.0.PLSStatus	DriveOutput.1	Active On	Capture0.CaptureActivate	← PLS.0.Status	Active On
Index.0.CommandInP...	ModuleOutput.1	Active On	Program		
Return			Program0		
Index.1.CommandInP...	ModuleOutput.2	Active On	Program0.Initiate	← ModuleInput.1	Active On
PLS			Outputs		
PLS.0.Status	Capture0.CaptureActivate	Active On	DriveOutput		
Inputs			DriveOutput.1	← Index.0.PLSStatus	Active On
ModuleInput			ModuleOutput		
ModuleInput.1	→ Program.0.Initiate	Active On	ModuleOutput.1	← Index.0.CommandInP...	Active On
ModuleInput.2	→ Home.0.SensorTrigger	Active On	ModuleOutput.2	← Index.1.CommandInP...	Active On
			Home		
			Home0		
			Home0.SensorTrigger	← ModuleInput.2	Active On

Figure 8 – Assignments View (Actual Assignments Used for Flying Cutoff Example)

Figure 8 above shows the Assignments view from PowerTools Studio. The assignments used are the key to bringing together all of the functions used in the PTi210 to perform the flying cutoff. Assignments can be made in one of two ways: by dragging and dropping across the screen the appropriate sources and destinations to their places shown above or by highlighting the desired source and destination and selecting ASSIGN. An explanation of each Source to Destination assignment follows to explain how it is used in the flying cutoff. Figure 8 above shows graphically how the Index and Global PLSs work together to create the flying cutoff.

1. ModuleInput.1 → Program.0.Initiate

This assignment is created so that the user can initiate the program that controls the Flying Cutoff using a digital input. Any input can be used for this purpose. This example uses ModuleInput.1 to start the program. This is an edge-sensitive assignment meaning once a rising edge is seen on ModuleInput.1, the program will begin running. ModuleInput.1 does NOT have to remain ON for the program to continue to run.

2. ModuleInput.2 → Home.0.SensorTrigger

This assignment is created so that the cutting device can align itself on start-up to a known position. This must be done once before any cuts are made. A digital input on the PTi210 must be used because they can be captured within 1 usec to give a very accurate home routine, whereas the Drive Inputs cannot be captured at the 1usec rate. “verify this” Ummmmmmmmmmmmmmmmmmmm

3. Index.0.PLStatus → DriveOutput.1

This assignment is created to control the digital output used to activate the shear (or other cutting mechanism). An index PLS is used to control the output because we can easily and accurately determine when the cut output activates based on the position and velocity of the knife.

4. Index.0.CommandInProgress → ModuleOutput.1

This assignment is purely optional and is used to turn on a digital output when the Cut index is in progress. There is a duplicate assignment from Index.1.CommandInProgress to SPIO.2.Out, which is used to indicate that the Return index is in progress.

5. PLS.0.Status → Capture.0.CaptureActivate

This assignment is used to capture the exact master position when the PLS turns on. This way we can use the captured master position as the starting point for the Cut index, which will result in a very accurate cut being made.

## Motion – Home Setup

Axis 1

- Status
- Graph
- Hardware
- Setup
- Devices / Vars
- I/O Setup
- Motion
  - Jogs
    - Jog0
    - Jog1
  - Homes
    - Home0
  - Indexes
    - Cut\_Move
    - Return
    - Index2
    - Index3
    - Index4
    - Index5
    - Index6
    - Index7

Home Number: 0

Name: Home0

Home Reference: Sensor

Time Base: Realtime

Velocity: 50.00 Inch/m

Acceleration: 1000.00 Inch/m/s

Deceleration: 1000.00 Inch/m/s

If on sensor...  
 Back off before homing  
 Go forward to next sensor

Home Offset  
 Calculated offset 0.0225 Inch  
 Specified offset 0.0000 Inch

Limit Distance 0.0000 Inch

End Of Home Position 0.0000 Inch

Online

Feedback

Position Feedback	0.0002 Inch
Velocity Feedback	0.00 Inch/m
Current Demand	0.0 %
Following Error	0.0000 Inch
Master Velocity	0.0000 MstrInch/s

Control Panel

Start

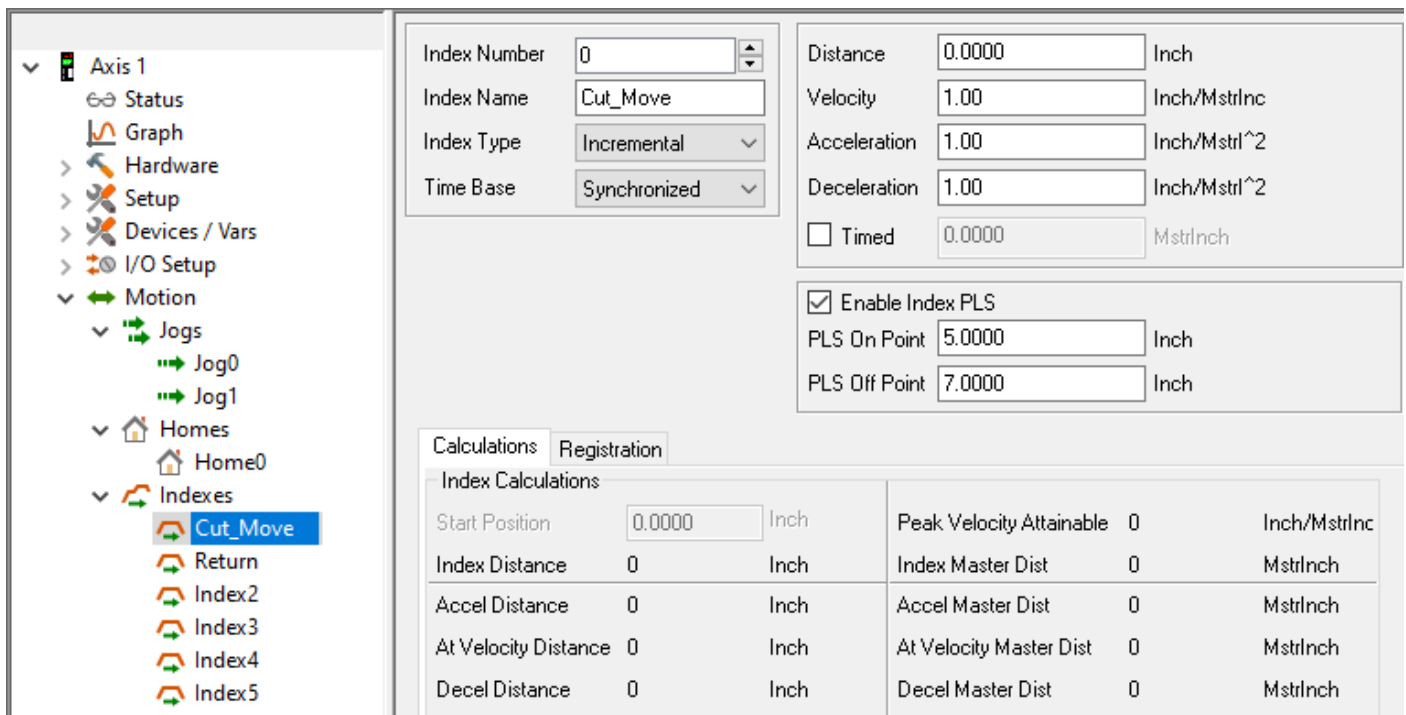
Press [PAUSE/BREAK] to toggle Stop All

**Figure 9 – Home Configuration**

## Motion – Indexes – Cutoff and Return Indexes Setup

To cut the material without stopping it, the cutting mechanism must move in unison with the moving material. If the cutting device is not moving at the same speed as the material, the material could be damaged. For instance, if the cutting blade moves slower than the material, the material will build up or wrinkle behind the blade. If the cutting blade moves faster than the material, the material could stretch or tear in the wrong place. Some applications may require that the cutting device move at a slightly different speed from the material, but most systems use an exact 1:1 ratio between master and follower speed.

To run the motor (or cutting mechanism) at the same speed as the material a synchronized index is used. This means that the index velocity will be referenced to the master encoder velocity. Figure 9 below shows the Index setup view.



**Figure 10 – Cut\_Move Configuration**

To run a synchronized index, modify the Time Base parameter in the index setup view. Figure 9 above shows the Time Base set to “Synchronized”. When Time Base is set to Synchronized, the units of Velocity, Acceleration, and Deceleration change to reflect how the index parameters are now referenced to the master encoder.

The units of Velocity in a synchronized index are as follows:

$$\frac{\text{Follower Distance Units}}{\text{Master Distance Units}}$$

The follower is the axis that is being controlled by the PTi210 module. If the desired velocity is 1:1, and the follower and the master distance units are the same (i.e. inches, mm, revs, etc.), then set the Velocity parameter to 1.0. If you want the follower to run twice as fast as the master, enter a Velocity of 2.0.

The units of Acceleration and Deceleration are as follows:

$$\frac{\text{Follower Distance Units}}{(\text{Master Distance Units})^2}$$

The flying shear application requires two indexes. The first (Index 0) is a synchronized index with a velocity of 1.0 to perform the cut. The second index (Index 1 seen in Figure 10) is also a synchronized index with a velocity of

2.0 used to return to the start position. Index 1 could be either synchronized or real-time, but this example uses a synchronized return index. The greater the velocity for Index 1, the faster the return stroke will be.

If the return stroke is a synchronized index, be sure not to set the velocity too high such that the index velocity would exceed the maximum motor velocity. This can be calculated as follow:

$$\text{Maximum Return Index Velocity} = \frac{\text{Motor Max Velocity (in User Units)}}{\text{Master Max Velocity (in User Units)}}$$

Motor Max Velocity (in User Units) Master Max Velocity (in User Units)

The maximum motor velocity in user units can be found on the User Units view in the velocity units setup box.

Index Calculations		Registration	
Start Position	0.0000 Inch	Peak Velocity Attainable	0 Inch/MstrInch
Index Distance	0 Inch	Index Master Dist	0 MstrInch
Accel Distance	0 Inch	Accel Master Dist	0 MstrInch
At Velocity Distance	0 Inch	At Velocity Master Dist	0 MstrInch
Decel Distance	0 Inch	Decel Master Dist	0 MstrInch

**Figure 11-Return Move Configuration**

Index 0 also uses an Index PLS to trigger the output to the cutting mechanism. The index PLS allows the user to turn on a bit (usually assigned to an output) a certain distance into a move. By looking at the Calculations tab on the index view, we can determine how much distance is traveled during the acceleration portion of the profile.

Make sure that the Index PLS On Point is greater than the follower distance it takes to accelerate to velocity. If not, the cutting mechanism will not have reached the same speed as the material and could damage the material.

## Program 0 – Actual Program Code

```
'-----  
'Description - Flying Cutoff (Unregistered Material) -  
'Revision 1 - 2/15/21, Created using PowerTools Studio Software V01.02.00.02 -  
'and Pti-210 Firmware V01.02.00.04 -  
'-----  
  
'Var.CutLength = 'Enter the desired cutlength here  
'Var.CutStrokeLen = 'Enter the stroke length of the cutting mechanism here  
'Var.CutVelRatio = 'Enter the velocity ratio at which to perform the cut  
'Var.ReturnRatio = 'Enter the velocity of the return index as a ratio to the master  
'Var.BatchCount = 'Enter number of cuts to make (if infinite enter 0)  
  
'Initialize Calculations  
PLS.0.RotaryRolloverPosn = Var.CutLength  
Index.0.Dist = Var.CutStrokeLen  
Index.0.Vel = Var.CutVelRatio  
Index.1.Vel = Var.ReturnRatio  
Var.Counter = 0  
  
'InitializeSettings  
Capture.0.CaptureClear  
Capture.0.CaptureEnable = ON  
PLS.0.PLSEnable = ON  
  
MasterAxis.UndefineHome = ON  
Wait For Time 0.05 'Seconds  
MasterAxis.UndefineHome = OFF  
'Optional output can be assigned and turned OFF here to reset the batch count status  
  
Home.0.Initiate 'Home0,Sensor,SpecifiedOffset=0.0000 Inch,Vel=50.00 Inch/m  
Wait For Home.AnyCommandComplete  
  
MasterAxis.DefineHome = ON  
Wait For MasterAxis.AbsolutePosnValid = ON  
MasterAxis.DefineHome = OFF  
  
DriveOutput.2 = ON 'Signal to the material feeder that system is ready  
  
Do While TRUE  
    Wait For Capture.0.CaptureTriggered  
    Index.0.Initiate Using Capture.0 'Cut_Move,Synchronized,Incremental,Dist=90.0000  
    Inch,Vel=1.00 Inch/MstrInc  
    Index.1.Initiate 'Return,Synchronized,Absolute,Dist=0.0000 Inch,Vel=2.00  
    Inch/MstrInc  
    Wait For Index.1.CommandComplete  
    Capture.0.CaptureClear  
  
    If Var.BatchCount <> 0 Then
```

```

Var.Counter += 1 'increment batch counter
  If Var.Counter < Var.BatchCount Then
    GoTo Repeat:
  Else
    DriveOutput.2 = OFF 'signal to the material feeder that the system is no
longer ready (batch count met)
    GoTo Stop:
  Endif
Else
  GoTo Repeat:
Endif

Repeat:
Loop

Stop:
'optional output turned ON here to signal the master controller that the batch count is
complete
End

```

### **Description of Program 0 Code**

The program begins by having the user enter the proper values for the five User Variables and remove the apostrophes from the beginning of lines to make the commands valid. This is based on the mechanics of the machine and the desired product output.

The next section of the program contains a series of formulas that utilize the constants previously entered by the user. The first calculation sets the PLS.0.RotaryRolloverPosn equal to the user variable called Cut Length.

Effectively, this sets the rollover value of the PLS to the desired cut length so that the PLS Status will activate when a cut needs to be made. The next calculation sets the Index Distance for the Cut move to be equal to the stroke length of the cutting mechanism. The next two lines set the velocities of the Cut and Return indexes to be equal to the ratios defined by the user. The last calculation sets a temporary counter (used to keep track of the running batch count) to zero so that it is always reset at the beginning of the program.

The next section of the program configures some internal settings so that the internal objects (i.e. PLS, Capture, etc.) are all ready to go. The first two lines clear the capture object in case it had already been set in a previous run of the program, and re-enable the capture object so that we are prepared to capture the first rising edge of the PLS. Next, the PLS is enabled so that we can generate PLS outputs for the next capture. For PLS's to operate, the PLS must be enabled AND the Absolute Position Valid signal must be active (the Absolute Position Valid signal automatically activates when a home has been completed or a Define Home function is activated).

Once the Define Home function has been used, it must first be undefined before it can be redefined again. Therefore, the program activates and deactivates the MasterAxis.UndefineHome function before trying to define the home. The last program line in this section turns OFF a digital output that is used to tell the master controller that the batch count has been reached. This function is entirely optional.

Next, we run a home routine to home the knife axis to its home sensor. This aligns the cutting mechanism so that it is in a known position and is ready to make the first cut. The program is then commanded to wait for the home sequence to be complete before moving on.

Once the home is complete, we need to define the master axis home so that the PLS will work properly. Therefore, we use the `MasterAxis.DefineHome` function to zero out the master axis feedback position. The program then waits for a short period of time before deactivating the `DefineHome` function.

Now that everything is homed and ready to run, we manually force a digital output ON which is a signal to the master controller to begin feeding the material.

Once the material is feeding, the program steps into a loop that will repeat forever because the argument for the `Do While` function is set to `TRUE`. Inside this loop we will check the current running batch count to determine whether to continue making cuts or not. Inside the loop, we first wait for the `Capture.0.CaptureTriggered` signal to activate. This signal will automatically come on when the PLS activates which will cause the capture object to capture the master position. Once the capture happens, the first cut index must be started. This is done using the `Index.0.Initiate` instruction. The `Using Capture.0` motion modifier is placed at the end of the index initiate instruction so that the accurate position captured from the PLS is used as the starting point for the index. This results in a very accurate cut being made. The next line of the program then initiates the return index. The program waits for the return index to be complete before moving on. Once the return index is complete, we must clear the capture object out so that it can capture on the next rising edge of the PLS signal.

The next group of instructions is used to determine whether we have reached our desired batch count or not. If the user has not selected to repeat the cutting sequence infinitely (by setting batch count equal to 0), then we check if the batch count is met. If the counter has not reached the desired batch count, the program increments the counter, and then repeats the process by jumping to the `Repeat` label in the program which loops us back to the top of the `Do While` loop. If the counter has reached the batch count, then the program jumps to a different label in the program called `Stop`. When the program gets to the `Stop` label, a digital output is forced on to indicate to the master controller that the batch count was reached. The next program instruction is the `END` instruction, which causes the program to stop.

This batch count routine is purely optional and is in no way necessary to make the `Flying Cutoff` application function.