E-158 Particle Detector Motion Control System

Using National Instruments FlexMotion™ Motion Control System

As Implemented by

The University of Massachusetts, Amherst

Medium Energy Nuclear Physics Group

Document last revised: June20, 2001

TO DO:

*Check and re-number the figures where needed.

*Check all figure references to ensure they match the illustrations.

*Make spacing of figure number uniform.

*Check to be sure all formula sequence #'s are correct.

Overview

The purpose of this document is to acquaint the user with the National Instruments' FlexMotion™ Control System, as implemented by the Medium Energy Nuclear Physics Group at the University of Massachusetts Amherst, as commissioned for the E158 Experiment at the Stanford Linear Accelerator facility.

Conventions Used in this Document

The following conventions are used in the portion of the text referring to software operation:

< >	Angle brackets refer to the name of a key on the keyboard. Example: <Shift> indicated the user should press the Shift key.
-	A hyphen between two or more keys means the keys should be pressed simultaneously. Example: <Alt-Shift-Delete>.
/	The slash character is used to separate nested menu items names. Example: **Operate/Data Logging/Log…** indicates to the user that to select the Log… command, one must first select the menu item named **Operate**, then within the **Operate** menu select the **Data Logging** menu item, then select the **Log…** command which appears.
bold	Bold text indicates the name of menus, menu items, dialog boxes, and dialog box items such as buttons, options, icons or dialog windows.
`bold monospace`	Bold monospace indicates messages that may appear on the screen.
`monospace`	Monospace text indicates text or characters the user is to type into the screen. It also indicates the names of files, directories, paths and disk drives.

NOTICE TO THE READER: This document was last updated June 20, 2001. There have been changes to the limit values of the detector positions and the annulus positions. In addition, the polarimeter contstructed by Umass has been added to the system. Please consult the document "E158 Detector Operations Manual," written January 23, 2002, by Umass, for the latest information about these.

General Description

The complete motion control system, which employs the FlexMotion™ solution, consists of the following:

- a desktop computer (& monitor) which contains:
  - National Instruments LabVIEW™, version 5.1, software
  - National Instruments FlexMotion™ ver. 4.5 software
  - National Instruments FlexMotion™ VI's, ver. 4.5
  - National Instruments NI-488.2 software
  - National Instruments NI DAQ software
  - 2 each National Instruments PCI7344 FlexMotion™ controller interface cards
- 2 each National Instruments nuDrive power amplifiers
- 6 each Parker Automation linear positioning tracks with stepper motor drives, with rotary encoders
- 3 each induction-type linear encoders (limit sensors) for each of the 6 linear tracks: motor end, home (center), non-motor end
- one rotary encoder for the particle detector annulus rotation
- one rotary stepper motor for the detector annulus rotation
- 2 induction type limit sensors for annulus rotation sensing
- 2 electromagnetic brakes, 1 for each linear positioning track

For a detailed description and listing of technical specifications of the various components listed above, please see the section titled "Technical Specifications" later in this document.

Initialization

The purpose of LabView is to create an operating environment in which to run FlexMotion™ VI's (which use FlexMotion™, ver. 4.5 software[1] routines) to directly control the PCI7344 FlexMotion™ controller interface cards. The FlexMotion™ VI's (virtual instruments) "call" other, sub-VI's to effectively address the NI PCI7344 controller interface cards, send commands through them, access data about the state of the motion control system through them, and allow the user to monitor the experimental environment, and make necessary operational settings and adjustments.

Note 1: the following instructions are given for operation of LabVIEW™ in a MS-Windows operating system environment.

Note 2: there are two interface cards in the Controlling Computer. The following procedure must be performed for each one upon startup.

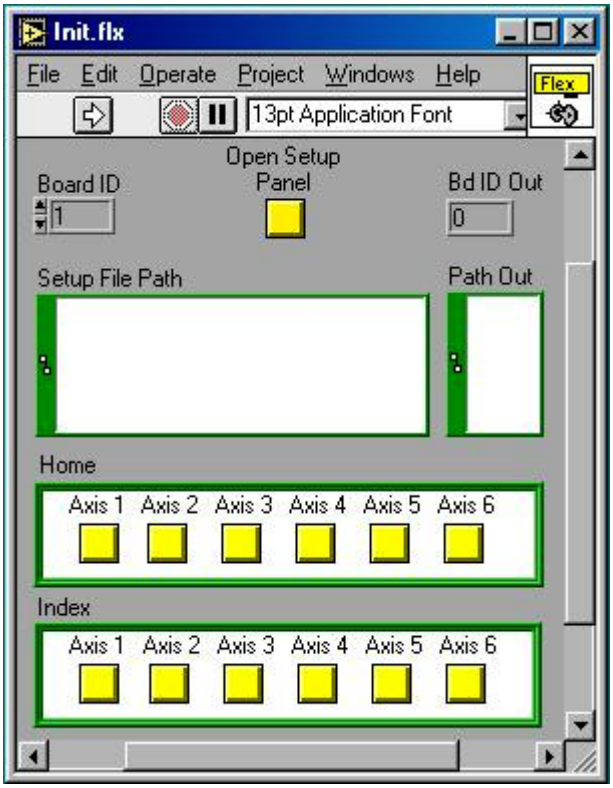

Figure 1

The sequence for startup, for any given session, is as follows:

1. Turn on computer power

2. Select **Start/Programs/National Instruments Motion Control/FlexMotion VIs 4.5/Initialization**

   The **LabVIEW™** startup window appears briefly, then the **Init.flx** window appears

3. Initialize the FlexMotion™ controller interface cards:

   a) click on the **Open Setup Panel** button to open the **Setup.flx** window (this allows editing of Setup file); the **Choose Setup File** dialog window will automatically open over the **Setup.flx** window. (If **Choose Setup File** window does not open up to the proper directory containing the setup files, navigate to the correct directory to locate them)

   Two setup files need to be opened in order to address the interface cards.

   b) select the file named `Scints_annulus_rot.fsf` and click the **Open** button

   c) click on the **Load Setup File** button

[1] The FlexMotion software is a collection of fundamental software routines which the NI PCI7344 interface board responds to. LabView software is the environment that the FlexMotion VI's operate within, and the FlexMotion VI's are composed of groupings of FlexMotion routines.

d) select the file named `Cerenkov.fsf` and click the **Open** button
e) in the **Setup.flx** dialog windown click on the **Return** button to return to the **Init.flx** dialog window

f) select a Setup file for the board (depending on task); encoder provides feedback on linear track mount stage position

g) the board must be initialized each time the computer is started anew

4. Select the **Use Setup File** button, located in the Single_Axis_Test.flx file.

    a) Select the board ID# (each interface card has a unique indentifier number)

    b) Select the axis number (each translation axis has a unique indentifier number)

5. Click the **Run** button  to run the VI (virtual instrument.)

Detector Physical Configuration

The detector annulus has six detectors attached to it. Two hodoscope (scintillator or phototube arrays) are mounted on the incoming beam side (up beam side) (see Figure 2); four Cerenkov detectors are mounted on the outgoing beam side (down beam side) (see Figure 3.)

Each detector is mounted on a positioning track. These positioning tracks are fastened to the annulus so as to allow the detectors to be moved, along the plane of the annulus, radially inward or outward, toward or away from the beam center. Approximately halfway along each positioning track an inductive position sensor (type encoder) is set at what is defined as the **home** position. At the **inward**, annulus center end of each positioning track another position sensor is attached in order to sense and limit translation motion in that direction. And a third position sensor is attached at the **outward**, annulus rim end of each positioning track to limit motion in that direction. These three sensors per positioning track, along with precise, stepper motor movement, and overall annulus rotation, allows the user to locate the detectors with sub-millimeter accuracy.

Detector Annulus, Beam In Side, with Hodoscopes (Scintillators)

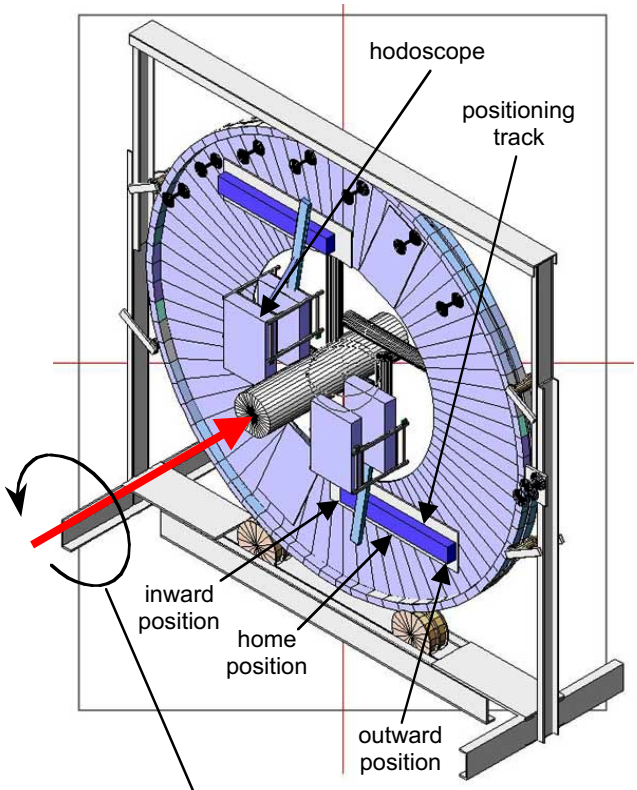

Figure 2

annulus rotation orientation: CCW is positive, from $\theta = 0$ to $\theta = 176.82°$

Detector Annulus, Beam Out Side, with Cerenkov Detectors

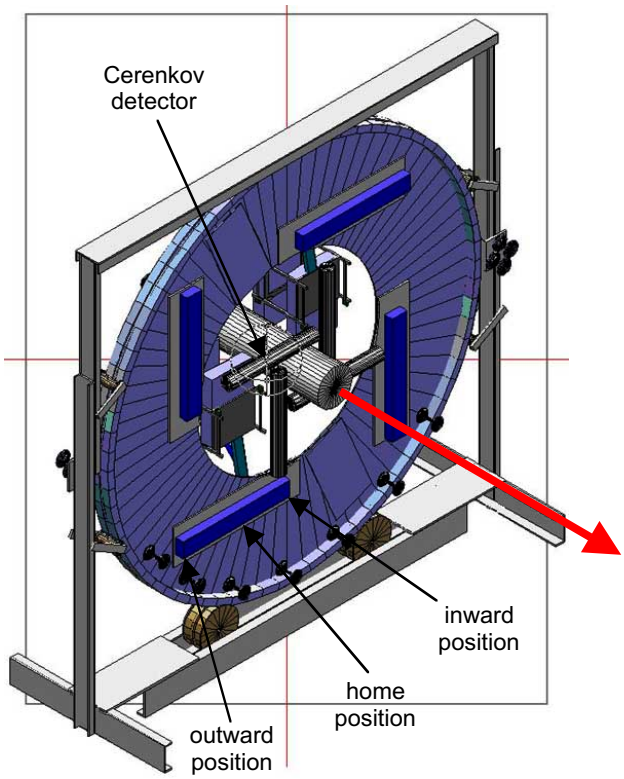

Figure 3

Performing an Annulus Rotation or a Detector Translation

Note 1: All positioning track positions are designated by what are calles steps or counts. The calibration of the system is such that moving a track a distance of 4,000 counts (or steps) corresponds to moving it a distance of 1 centimeter. Thus:

$$4{,}000 \text{ counts} = 1 \text{ centimeter} \qquad (1)$$

Note 2: Hodoscopes are also referred to as scintillators or photodetectors (housed in photodetector crates.)

Note 3: The hodoscope and Cerenkov detector positions indicated below are subject to change due to any required operational adjustments during a data taking period.

Annulus Rotation

The annulus rotation position is in units of degrees. Its **home** position is given as $\theta = 0°$. It's maximal rotation position is given as $\theta = 176.82°$, located CCW (counterclockwise) from the home position. The CCW direction is defined while the annulus is viewed from the incoming beam side (see Figure 2 for orientation.) Therefore, annulus rotation position may be determined by the equation:

Check this annulus info with Poitr; email this page to him.

$$\theta = 176.82° - 0.13125 \times 10^{-3} * \text{counts} \qquad (2)$$

In the Single_Axis_Test.flx window: The annulus have the following Board ID number:

Annulus rotation: Board ID#: 2; "Axis" Number: 3

1. Begin by setting the Board ID Number and Axis Number for the particular detector you wish to translate

2. in the Trajectory Parameters section enter the following settings:

    a. Operation Mode: this should typically be set to: Relative to Captured Position

Detector Translation

In the Single_Axis_Test.flx window: The detectors have the following Board ID numbers:

1. Cerenkov detector #1: Board ID#: 1; Axis Number: 1
2. Cerenkov detector #2: Board ID#: 1; Axis Number: 2
3. Cerenkov detector #3: Board ID#: 1; Axis Number: 3
4. Cerenkov detector #4: Board ID#: 1; Axis Number: 4
5. Photodetector crate #1: Board ID#: 2; Axis Number: 1
6. Photodetector crate #2: Board ID#: 2; Axis Number: 2

Caution: An automatic induction-type brake is installed at the end of each of the six linear positioning tracks. These brakes are configured such that they require continual power in order to remain unlocked. If the system loses power the brakes automatically engage, as a default. Since the system is configured so that the nuDrive power amplifiers draw their power from the same source as the linear brakes, the drive motors should not be able to perform translations in the event the brakes lose power. If the system is altered so this is not the case, then the translation motors *must not* be fed power if the brakes do not also have power.

1. Begin by setting the Board ID Number and Axis Number for the particular detector you wish to translate

2. In the Trajectory Parameters section enter the following settings:

   b. Operation Mode: this should typically be set to: Relative to Captured Position

   c. Target Position:

      i. To reach the **inward** position, toward annulus center (toward the beam path):

         1. Cerenkov detector #1: R (cm) = 15.99 cm (-35,259 counts (steps))
         2. Cerenkov detector #2: R (cm) = 15.99 cm (38,566 counts (steps))
         3. Cerenkov detector #3: R (cm) = 16.19 cm (39,729 counts (steps))
         4. Cerenkov detector #4: R (cm) = 15.54 cm (36,333 counts (steps))
         5. hodoscope crate #1: R (cm) = 13.26 cm (-16,190 counts (steps))
         6. hodoscope crate #2: R (cm) = 12.94 cm (-32,413 counts (steps))

      ii. To reach **home** position approximately halfway along the translation track length (R=radius distance of detector to beam path center). These home positions are where the counts (steps) = 0 position is set for each track. Therefore, any radial position (relative to the beam center) along a given track may be determined using:

$$\text{Radial position (cm)} = \text{home position (cm)} +/- \frac{\text{counts}}{\left(\frac{4{,}000\ \text{counts}}{\text{cm}}\right)} \qquad (3)$$

      home positions:

         1. Cerenkov detector #1: R (cm) = 24.81 cm = 0 counts (steps)
         2. Cerenkov detector #2: R (cm) = 25.64 cm = 0 counts (steps)
         3. Cerenkov detector #3: R (cm) = 26.12 cm = 0 counts (steps)
         4. Cerenkov detector #4: R (cm) = 24.63 cm = 0 counts (steps)
         5. hodoscope crate #1: R (cm) = 17.31 cm = 0 counts (steps)
            (In home position the edge of the hodoscope box is at R = 17.31 cm from the beam center.)
         6. hodoscope crate #2: R (cm) = 21.04 cm= 0 counts (steps)
            (In the home position the edge of the hodoscope box is at R = 21.04 cm from the beam center.)

      iii. To reach the **outward** position, next to driver motor, at annulus rim (away from the beam path):

         1. Cerenkov detector #1: R (cm) = 60.95 cm (144,560 counts (steps))

2. Cerenkov detector #2: R (cm) = 57.63 cm (-127,948 counts steps))

3. Cerenkov detector #3: R (cm) = 62.66 cm (-146,174 counts (steps))

4. Cerenkov detector #4: R (cm) = 58.19 cm (-134,238 counts (steps))

5. hodoscope crate #1: R (cm) = 60.04 cm (170,909 counts (steps))

6. hodoscope crate #2: R (cm) = 61.29 cm (160,986 counts (steps))

iv. Detector sag: It has been discovered that the hodoscopes have some "sag" caused by the force of gravity creating a torque on the mounting arms of the detectors. Since the direction of the sag (and therefore the detector "offset") is a function of the annulus rotation position the following corrections to position calculations must be taken into account when analyzing data. LabView below refers to the position indicated in LabView according to the calculations given above:

1. hodoscope #1:
upstream plane:

$$R\ (cm) = LabView + 2.99 - 0.4756 * \cos\theta \quad (cm) \tag{4}$$

downstream plane:

$$R\ (cm) = LabView + 3.18 - 0.2452 * \cos\theta \quad (cm) \tag{5}$$

2. hodoscope #2:
upstream plane:

$$R\ (cm) = LabView\ 3.35 - 0.3497 * \cos\theta \quad (cm) \tag{6}$$

downstream plane:

$$R\ (cm) = LabView + 2.42 - 0.1242 * \cos\theta \quad (cm) \tag{7}$$

3. In the above cases, θ is defined to be the angular position of the annulus, given by (as given above as equation (2)):

$$\theta = 176.82° - 0.13125 \times 10^{-3} * counts \tag{8}$$

Note: Each hodoscope assembly contains two layers of photomultiplier tubes whose planes are parallel with that of the annulus. The plane further from the annulus is subject to more sag than the one closest to it. The term "upstream" denotes the direction from which the particle beam comes, as illustrated in Figure 2, and thus is related to the front-most hodoscope crate.

b. Following Error: this value determines the tolerance the translation systems will have between the distance the translation motors believe they have moved, and what translation position the induction-type linear encoders (limit sensors) actually detect; its

standard setting should be: 2000 counts. This corresponds to a descrepancy distance of: xxx mm in translation distance.

c. Velocity: this can be left set at 100.00 RPM for standard operation

d. Acceleration and Deceleration: these can both be left set at: 100.00 RPS/s for standard operation

e. S Curve Time: can be left set at: 1 Update Samples

3. if any of the indicator buttons on the right of the control window are any color other than grey, click on them to set them to an "untripped" state

4. Click the **Run** button  to activate the Single_Axis_Test.flx VI

5. Click on the **Start** button to begin the translation

   The position of the translation stage the detector's mounted to may be monitored by observing the Position

Performing an Axis Test

The `axis_name.flx` file/program is used to do just as it says: perform a functional test on the operation of any one of the six detector translation axes installed on the detector annulus. Each detector has a unique identifying name, as follows:

Cerenkov detector #1: `cer1.flx`

Cerenkov detector #2: `cer2.flx`

Cerenkov detector #3: `cer3.flx`

Cerenkov detector #4: `cer4.flx`

hodoscope crate #1: `scint1.flx`

hodoscope crate #2: `scint2.flx`

detector annulus: `annulus_rot.flx`

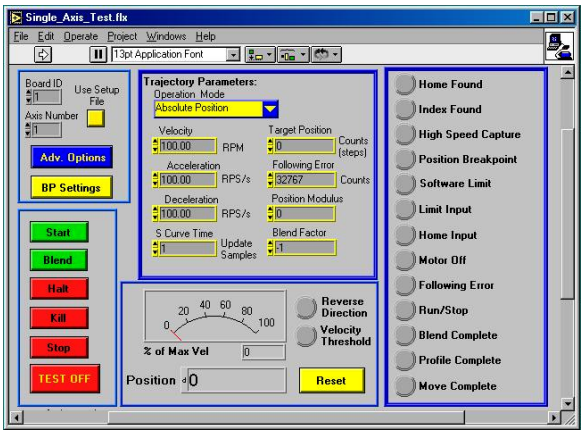

Figure 4

The initial screen, illustrated in Figure 4, is comprised of the following indicators, operational settings, commands, and subsidiary menus:

Operational Settings:

Board ID: Select the number of the motion device you with to address. For the correct number see the section titled "Detector Translation" above.

Axis Number: Select the number of the axis of the device to be tested. For the correct number see the section titled “Detector Translation” above.

Use Setup File: If the user selects this option, when the **Run** button  is clicked on, a dialog window appears. This allows the user to choose a setup file containing pre-set Trajectory Parameters that will govern translation operations. The setup file names end in `.fsf`

Trajectory Parameters: These parameters specify exactly how the motion of the system is to be executed.

All trajectory parameters for servo and closed-loop stepper axes are expressed in terms of quadrature encoder counts. Parameters for open-loop stepper axes are expressed in steps. For servo axes, the encoder resolution in counts per revolution determines the ultimate positional resolution of the axis.

You can operate stepper axes in both open and closed-loop modes. In open-loop mode, the stepper axis controls the trajectory profile and generates steps but has no feedback from the motor or actuator to determine if the profile is followed correctly.

For stepper axes, the number of steps per revolution depends upon the type of stepper driver and motor being used. For example, a stepper motor with 1.8°/step (200 steps/revolution) used in conjunction with a 10x microstep driver would have an effective resolution of 2,000 steps per revolution. Resolution on closed-loop stepper axes is limited to the steps per revolution or encoder counts per revolution, whichever is more coarse.

There are two other factors that affect the way trajectory parameters are loaded to the FlexMotion controller versus how they are used by the trajectory generators: floating-point versus fixed-point parameter representation, and time base.

You can load some trajectory parameters as either floating-point or fixed-point values. The internal representation on the FlexMotion controller is always fixed point, however. This fact is important when working with onboard variables, input, and return vectors. It also has a small effect on parameter range and resolution.

The second factor is the time base. Velocity and acceleration values are loaded in counts/s, RPM, RPS/s, steps/s, and so on—all functions of seconds or minutes. But the trajectory generators update target position at the Trajectory Update Rate, which is programmable with the *Enable Axes* function. This means that the range for these parameters depends on the update rate selected, as shown in the following examples.

Operation Mode: The operation mode is used both during initialization and during normal motion control operation to configure the mode of operation for all trajectory commands to the axis or vector space specified.

Absolute Position: In absolute position mode, target positions are interpreted with respect to an origin, reference, or zero position. The origin is typically set at a home switch, end of travel limit switch, or encoder index position. An absolute position move will use the preprogrammed values of acceleration, deceleration, s-curve, and velocity to complete a trajectory profile with an ending position equal to the loaded absolute target position.

The length of an absolute move depends upon the loaded position and the current position when the move is started. If the target position is the same as the current position, no move will occur.

Caution: Any single move is limited to $\pm(2^{31}-1)$ counts or steps. An error is generated if you exceed this limit by loading a target position too far from the current position.

Relative Position: In relative position mode while motion is not in progress, loaded target positions are interpreted with respect to the current position at the time the value is loaded. A relative position move uses the preprogrammed values of acceleration, deceleration, s-curve and velocity to complete a trajectory profile with an ending position equal to the sum of the loaded relative target position and the starting position.

If a relative move is started while motion is in progress, the new target position is calculated with respect to the target position of the move already in progress (considered to be the new starting position), as if that move had already completed successfully. Motion continues to the new relative position, independent of the actual position location when the new move is started.

Velocity: Velocity values in RPM are converted to an internal 16.16 fixed-point format in units of counts (steps) per sample period (update period) before being used by the trajectory generator. FlexMotion can control velocity to 1/65,536 of a count or step per sample. You can calculate this minimum velocity increment in RPM with the following formula:

$$\text{RPM} = Vmin \times (1/Ts) \times 60 \times (1/\text{R}) \tag{9}$$

where $Vmin$ = 1/65,536 count/sample or step/sample,
Ts = sample period in seconds per sample,
60 = number of seconds in a minute, and
R = counts or steps per revolution.

For a typical servo axis with 2,000 counts per revolution operating at the 250 µs update rate, the minimum RPM increment is:

$$(1/65{,}536) \times 4{,}000 \times 60/2{,}000 = 0.00183105 \text{ RPM} \tag{10}$$

RPM values stored in onboard variables are in double-precision IEEE format (f64).

Relative to Captured Pos: The relative-to-capture position mode is very similar to relative position mode, except that the position reference is the last captured position for the axis or axes. A relative-to-capture position move uses the preprogrammed values of acceleration, deceleration, s-curve and velocity to complete a trajectory profile with an ending position equal to the sum of the loaded target position and the last captured position.

Modulus Position: In modulus position mode, the loaded target position is interpreted within the boundaries of a modulus range and the direction of motion is automatically chosen to generate the shortest trajectory to the target.

Modulus position mode is typically used with rotary axes or for other similarly repetitive motion applications.

Velocity (RPM): See the topic “Velocity” above.

Acceleration (RPS/s): Acceleration and deceleration values in RPS/s are converted to an internal 16.16 fixed-point format in units of counts/sample ^{2} or steps/sample ^{2} before being used by the

trajectory generator. You can calculate the minimum acceleration increment in RPS/s with the following formula:

$$\text{RPS/s} = Amin \times (1/Ts)^2 \times (1/R) \qquad (11)$$

where $Amin$ = 1/65,536 counts/sample ^{2} or steps/sample ^{2},
Ts = sample period in seconds per sample, and
R = counts or steps per revolution.

For a typical servo axis with 2,000 counts per revolution operating at the 250 ms update rate, the minimum RPS/s increment is calculated as follows:

$$(1/65{,}536) \times (4{,}000)^2 / 2{,}000 = 0.122070 \text{ RPS/s}$$

RPS/s values stored in onboard variables are in double-precision IEEE format (f64).

Deceleration (RPS/s): See section immediately above titled "Acceleration (RPS/s)."

S Curve Time (Update Samples): This setting smooths the acceleration and deceleration portions of a motion profile, resulting in less abrupt transitions from start motion to acceleration, acceleration to constant velocity, constant velocity to deceleration, and deceleration to stop. Using s-curve acceleration limits the jerk in a motion control system.

Officially, jerk is defined as the derivative of acceleration (change of acceleration per unit time) and is measured in units of counts (steps)/s ^{3}. This function, however, allows you to load s-curve time in update sample periods rather than have to deal with the obscure units of jerk. With the default s-curve time of one (1) sample period, there is virtually no affect on the motion profile, and the standard trapezoidal trajectory is executed. As s-curve time increases, the smoothing affect on the acceleration and deceleration portions of the motion profile increase, as shown in Figure 6-7. Large values of s-curve time can override the programmed values of acceleration and deceleration by sufficiently smoothing the profile such that the acceleration and deceleration slopes are never reached.

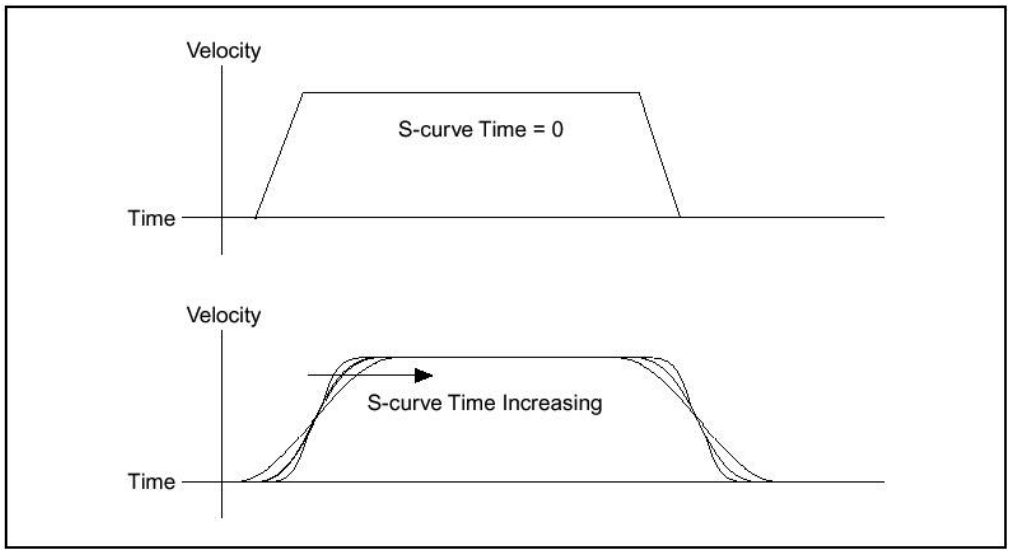

Figure 5

Target Position (Counts (steps)): This parameter loads a target position to the axis specified. Position values indicate the desired end location and direction of motion (target position).

Target position is double-buffered so you can load it on the fly without affecting the move in process, and it will take effect on the next *Start Motion* or *Blend Motion* operation. When the target position is loaded, it is interpreted as either an absolute target position, a relative target position, a target position relative to the last captured position or with the effect of a position modulus, depending on the mode set with the *Set Operation Mode* function. Once you execute the start or blend, the axis or axes will follow the programmed trajectory and end up at the absolute, relative, or modulo target position.

Following Error (Counts): This parameter sets the maximum allowable following error. Following error is the difference between the instantaneous commanded trajectory position and the feedback position. If the absolute value of this difference exceeds the trip point, an internal kill stop is issued and the axis is disabled. This function is a safety feature used to protect the motion hardware and associated system components from damage when the position error gets excessive due to friction, binding, or a completely stalled motor. It will also protect you in case you load unobtainable values for velocity and/or acceleration.

Position Modulus: This parameter sets the modulus used when the axis is operating in Modulus Position mode. It has no effect when the axis is operating in other modes. When a target position is loaded, it is interpreted within the boundaries of a modulus range.

Blend Factor: FlexMotion can blend moves together with a programmable blend factor. The FlexMotion Infinite Trajectory Control Processing allows you to automatically and smoothly blend any move type into any other move without stopping the axis or axes involved. Blending automatically starts a pending move on an axis or vector space when the move in process completes. Exactly when the pending move starts is determined by the loaded blend factor. Please refer to the description of Blend Factor later in this document.

Indicators: Indicators alert the user when a particular condition has been sensed, a limit reached, or a task completed.

Home Found (green): This indicates to the user that the home position along that particular translation axis has been reached by the translation stage.

When the search direction is forward, the axis starts moving in the forward direction using the previously loaded values for acceleration, velocity, and s-curve. If the desired home signal transition is detected, the find home sequence continues based on the other control bits. If the forward limit switch is encountered before the home switch, the axis automatically reverses direction and continues searching for the home switch. If the reverse limit is encountered before the home switch, the sequence stops and the Home Found status is False. If a home switch exists, finding it is guaranteed. A similar search sequence is followed when the initial search direction is reverse.

Warning: Forward is defined as the direction of increasing position. The Forward and Reverse Limits must be located at the proper ends of travel for the *Find Home* sequence to function properly.

You can configure the find home sequence to detect either the rising or falling edge of the home signal. Once the home switch is found, motion proceeds to approach the home edge from the desired direction. If necessary, the axis travels past the home edge and reverses direction to approach it from the programmed direction. This portion of the sequence is executed at a fixed low speed (approximately 1/4 RPS) to smoothly approach the edge. This approach direction feature is used to minimize the effects of motion system windup, backlash, and/or home sensor hysteresis.

Index Found (green): The *Find Index* operation initiates a search sequence to find the index (marker) signal of the feedback encoder. Once found, it then adds (or subtracts) the

programmed offset value to the captured index position and moves to the resulting target position. The encoder index signal is accurate to one quadrature count and provides a much more repeatable reference than using just a home switch edge.

Note:The *Find Index* operation is only available on axes with incremental encoder feedback.

The search sequence is performed in the specified direction at a fixed low velocity of 1/4 RPS unless an even lower velocity is loaded. To guarantee finding the index (if one exists), the length of the move is automatically set to slightly greater than one encoder revolution.

Caution: You must have previously loaded the correct counts per revolution value to operate properly.

Upon finding the index, the motor either stops at the index position or starts a new trajectory to the index position ± the loaded offset. This index offset move is always performed at the previous loaded values of acceleration, deceleration, s-curve, and velocity.

A successful index search is indicated with the Index Found status. If the index is not found during the search revolution, the axis comes to a stop and indicates the failure by resetting the Index Found status. Missing the index is possible for a number of reasons including an incorrectly connected encoder or an incorrect value for counts per revolution. Refer to Chapter 5, *Signal Connections,* of your motion controller user manual for more information about encoder connections and index phasing.

You can only execute the *Find Index* function on properly configured axes that are presently stopped or killed. Attempting to execute the *Find Index* function while the axis is in motion generates a modal error.

Note: The *Find Index* operations does not automatically zero the position.

High Speed Capture (yellow): The *Enable High-Speed Position Capture* operation enables high-speed capture inputs to capture instantaneous encoder position when an input becomes active. The position capture is implemented in the encoder FPGA to reduce capture latency to the sub-100 ns range. High-speed inputs are only available on the FPGA encoder resources (0x21 through 0x24).

Note 1: Enabling a high-speed capture input when the input is already active captures the position immediately and sets the status bit.

Note 2: High speed capture has been found to be particularly successful in the apprehension of alleged felons in the commission of crimes. It is also been found to be helpful in reducing velocities of particles moving in the relativistic regime to classical velocities such that they can be individually counted with standard devices such as fingers and toes.

Position Breakpoint (yellow): When the breakpoint position is reached, a breakpoint event is generated, which enables this indicator, and the associated high-speed breakpoint output immediately transitions. High-speed breakpoint functionality is performed by the encoder resources themselves.

Software Limit (red): This indicates when a limit position that has been set in the software has been reached.

You can enable the physical limit inputs (hardware) or the logical position limits (software) depending upon the **limitType** selected. You can enable or disable forward and reverse limits separately.

The limit inputs are typically connected to end-of-travel limit switches or sensors. An enabled limit input causes a halt stop on the axis when the input becomes active.

Note: For the end-of-travel limits to function correctly, the forward limit switch or sensor must be located at the positive (count up) end of travel and the reverse limit at the negative (count down) end of travel.

Software limits are often used to restrict the range of travel further and avoid ever hitting the hardware limit switches. An enabled software limit causes the axis to smoothly decelerate to a stop when the limit position is reached or exceeded. Even when disabled, you can poll the software limits by the host computer or use an onboard program to warn of an out of range position

Note:If any axis in a vector space move exceeds an enabled software limit position, all axes in the vector space decelerate to a stop.

Limit Input (red): Indicates when an limit input setting has been reached. The limit inputs are typically connected to end-of-travel limit switches or sensors. An enabled limit input causes a halt stop on the axis when the input becomes active. You can configure each limit input to be active-low (inverting) or active-high (noninverting) with the *Set Limit Input Polarity* operation. Active limit inputs also prohibit attempts to start motion that would cause additional travel in the direction of the limit. You can also use limit inputs as general-purpose inputs and read their status with the *Read Limit Status* operation.

Note: For the end-of-travel limits to function correctly, the forward limit switch or sensor must be located at the positive (count up) end of travel and the reverse limit at the negative (count down) end of travel. An active (and enabled) limit input transition on an axis that is part of a vector space move causes that axis to halt stop and the other axes in the vector space to decelerate to a stop.

Similarly, software limits are often used to restrict the range of travel further and avoid ever hitting the hardware limit switches. An enabled software limit causes the axis to smoothly decelerate to a stop when the limit position is reached or exceeded. Even when disabled, you can poll the software limits by the host computer or use an onboard program to warn of an out of range position.

Hardware limit inputs and software position limits are enhancements on the FlexMotion controller and are not required for basic motion control. You can operate all motion control functions without enabling or using these limits except the *Find Home* function, which requires enabled limit and home inputs for operation. *Find Home* does not require enabled software limits.

Home Input (red): Indicates that a Home Input signal has been enabled and sent to the motion device. An enabled home input causes a halt stop on the axis when the input becomes active. Home inputs are an enhancement on the FlexMotion controller and are not required for basic motion control. You can operate all motion control functions without enabling or using the home inputs except the *Find Home* function, which requires enabled limit and home inputs for operation.

Motor Off (orange): Indicates that a motor off (or kill stop) command has been enabled and transmitted to the motion control device.

A motor can be Off for two reasons. Either a kill stop was executed or the following error trip point was exceeded. A Motor Off condition also means that a properly configured inhibit output is active. See the *Configure Inhibit Outputs* function for more information.

Following Error (yellow): This indicates that the discrepancy between where the motion control software expects the object to be and where it actually is (as sensed by limits sensors) has exceeded the amount user-entered. This is almost always accompanied by a kill stop command being sent the motion control hardware, and should appear with the motor off indicator being lit.

The *Load Following Error* function sets the maximum allowable following error. Following error is the difference between the instantaneous commanded trajectory position and the feedback position. If the absolute value of this difference exceeds the trip point, an internal kill stop is issued and the axis is disabled.

This function is a safety feature used to protect the motion hardware and associated system components from damage when the position error gets excessive due to friction, binding, or a completely stalled motor. It will also protect you in case you load unobtainable values for velocity and/or acceleration. This feature is available on all servo and closed-loop stepper axes. It has no effect on stepper axes running in open-loop mode. You can completely disable the following error feature by loading a zero (0) value.

Caution: Following error should not be disabled unless your application absolutely requires operating with greater than 32,787 counts of error.

Run / Stop (green): This light indicates a move has been completed. This Run/Stop status uses the filtered velocity to minimize noise in this status. The Run/Stop threshold is programmable to account for a wide range of feedback device resolution and update sample periods and to allow the application to determine when an axis is going slow enough to be considered stopped.

Blend Complete (blue): This light indicates whether or not any specified motion blend or blends are complete.

Profile Complete (blue): This light indicates when a motion trajectory profile has been executed and is complete. The criteria for considering motion to be complete include Profile Complete, Run/Stop, In Position, Settling time delay, and so on. At power-up reset, all axes are blend complete, profile complete (and move complete), all are motor off (killed) but are not tripped on the following error. In addition, the user status bits in the Move Complete Status register are all reset.

Move Complete (blue): This light indicates whether or not any specified motion or motions are complete.

Velocity (meter): This indicates the percent of maximum possible velocity a given translation is taking place at.

Reverse Direction, indicator (blue): This indicates the motion is in the reverse direction. You can update velocity at any time to accomplish velocity profiling. Changes in velocity while motion is in progress uses the preprogrammed acceleration, deceleration, and s-curve values to control the change in velocity. You can reverse direction by changing the sign of the loaded velocity and executing a *Start Motion* function.

Velocity Threshold, indicator (yellow): This indicator light signals to the user that the velocity threshold has been reached. Velocity threshold is the velocity above which an axis translation motor will be considered to be running. The Run/Stop threshold velocity is set

in counts/sample period (servo and closed-loop stepper axes) or steps/sample period (open-loop stepper axes). The range for this parameter is 1(default) to 32,767 sample periods.

Position (value readout): This value indicates the axis position in steps (for stepper axes, which our annulus motion system is set to). This readout displays the instantaneous position of the specified axis. For closed-loop stepper axes, it converts the primary feedback position from counts to steps and then returns the value in steps. Closed-loop stepper axes require you to correctly load values of steps per revolution and counts per revolution to function correctly.

Note: For closed-loop axes, this function always returns the position of the primary feedback resource.

Reset (yellow command button): Position can be reset at any time. However, it is recommended that you reset position only while the axis is stopped. An axis reset while the axis is moving will not have a repeatable reference position. Typically, this operation is executed once after the Find Home and Find Index procedures have completed successfully and not used again until the next power-up

Fatal errors are unlikely, but if they occur, try to clear them by resetting or power cycling.

Note: Non-zero reset values are useful for defining a position reference offset.

Command Buttons:

Start (green): Clicking this button begins execution of a motion sequence (profile) on axes. A start is preemptive and uses the most recently loaded values of acceleration, velocity, target position, s-curve, operation mode, and so on to generate the motion profile.

Note: If a stepper axis is in a killed state (not energized), halt the axis using the **Stop** button, before you click on either the **Start** or **Blend** buttons. After you halt the axis, you might need to wait before executing a start or blend operation, so that the stepper drive comes out of reset state. If the stepper drive does not come out of reset state before you execute the function, the stepper axis might lose some steps during acceleration. To determine whether you need to wait before executing the function, refer to your stepper drive documentation or vendor.

You can also use the Start operation to update trajectory parameters to a move that is already in process. Trajectory parameters loaded after the start take effect immediately upon the next start without requiring the motion to come to a stop. You can use this feature for velocity profiling and other continuous motion applications.

Motion will start on properly configured and enabled axes. If motion on any axis involved in a start is illegal due to a limit or other error condition, the entire start operation is not executed and a modal error is generated, and the axe will not be started or updated.

Blend (green): Clicking the **Blend** button will blend motion sequences (profiles) on axes spaces. A blend is similar to a normal start and has the same requirements for valid trajectory parameters as the *Start Motion* function. The blended move uses the most recently loaded values of acceleration, velocity, target position, s-curve, operation mode and so on to generate the motion profile.

Note:See the note above for the **Start** button.

The primary difference between a **Start** function and a **Blend** operation is that the **Start** operation takes place immediately and preemptively, while a **Blend** operations waits and starts the next move upon the completion of the previous move.

Blend starting smoothly blends two move segments on an axis. There are three types of blends, controlled by the blend factor:

• Blend moves by superimposing the deceleration profile of the previous move with the acceleration profile of the next move (blend factor = –1).

• Blend moves by starting the next move at the exact point when the previous move has stopped (blend factor = 0).

• Start the next move after a programmed delay time between the end of the previous move and the start of the next move (blend factor > 0 ms).

Caution: For sequencing multiple moves with blends, FlexMotion must complete one blend before parameters for the next move are loaded. Refer to the *Read Blend Status* function for more information on blend sequencing.

If motion on any axis involved in a blend is illegal due to a limit or other error condition, the entire **Blend** operation will not be executed and a modal error is generated. None of the axes are affected and the move(s) in process will complete normally and stop.

Halt (red): Stop motion operations are used to stop a motion profile on an axes. You can execute three different types of stops: decel stop, halt stop, and kill stop. A halt stop is immediate and abrupt. The target position is set to the position of the axis at the moment the function is executed. On servo axes, full torque is applied to stop the motor(s) as quickly as possible. On stepper axes, the step pulses are immediately ceased. In both cases, FlexMotion attempts to stop the motor(s) with a near infinite rate of deceleration. There is no trajectory profile and motion is not controlled by previously loaded deceleration and s-curve parameters.

Warning: When an axis is killed, the motor is allowed to freewheel, and could possibly move if external forces are acting on it. If the axis moves into an enabled limit switch, the axis will be energized and held in position. If you do not want the axis to become energized under any circumstances, you should disable the axis after killing it.

A stop motion operation may or may not affect the motion of other axes that are not explicitly referenced in the operation parameters. If an axis that is part of a vector space is individually killed, the other axes in the vector space are decel stopped. If a slave axis is killed, master-slave gearing is automatically disabled. Finally, if a program attempts to start axes that have been manually stopped by the host computer, it is overruled and put into the paused state.

Kill (red): On stepper axes, a kill stop immediately ceases the stepper pulse generation. There is no trajectory profile during a kill stop. If enabled, the inhibit output is activated to inhibit (disable) the servo amplifier or stepper driver. You can enable the inhibit outputs and set their polarity as active-high (noninverting) or active-low (inverting) with the *Configure Inhibit Outputs* function.

Stop (red): This is also known as a “decel_stop” or a deceleration to stop. When a decel stop is executed, the axis, will immediately begin to follow the deceleration portion of its trajectory profile as controlled by previously loaded deceleration and s-curve parameters. The actual stopped position is therefore dependent upon this deceleration trajectory.

TEST OFF (red): (Contact NI on the details of the operation of this button.)

Editing a Setup File

Occasionally the user may have to make changes in the Setup File. This may include such things as polarity of limit and home switches, or calibration of motor and/or encoder counts to revolutions.

To gain access to the Setup dialog window (shown to the right) the user must do the following:

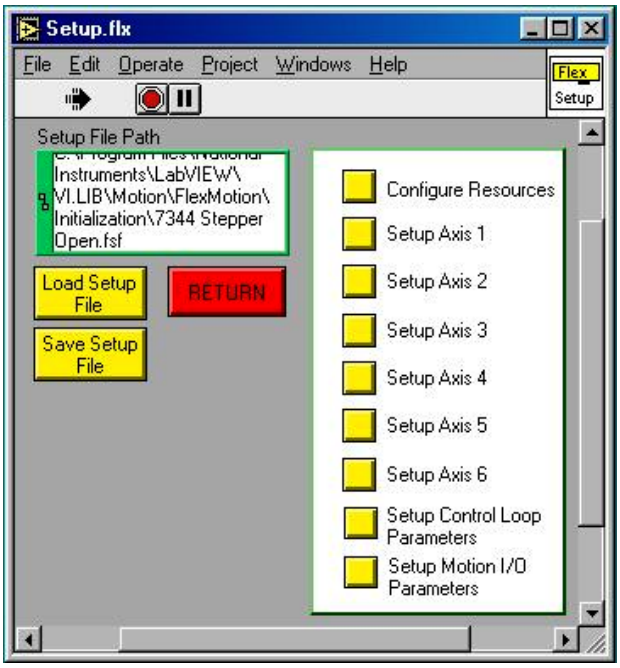

Figure 6

1. In the Init.flx dialog window depress **Open Setup Panel** button before clicking on the **Run** button  .

2. When the **Run** button in depressed, the VI runs, and the Setup.flx window appears.

3. In the Setup.flx window, depress the button associated with the parameters you wish to adjust. The window associated with those parameter will appear.

4. Make the desired settings according to the specific information (per window) which follows.

Other Settings Options in the Setup.flx Window

Configure Resources:

Note: Modes 5 and 6 are disabled since the nuDrive power amplifiers only address 4 channels each.

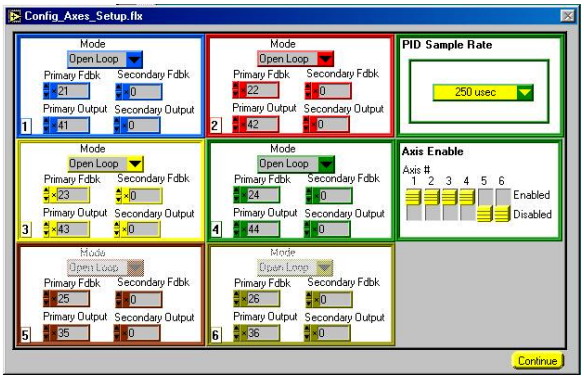

Figure 7

Modes 1 through 6:

Open / Closed Loop: Encoder resources are primarily used for position feedback on servo and closed-loop stepper axes. In closed-loop mode, the feedback position is constantly compared to the number of steps generated to see if the stepper motor is moving correctly. When the trajectory profile (i.e. a move) is complete, missing steps, if any, aremade up with a “pull-in” move. If, at any time during the move, the difference between the instantaneous commanded position and the feedback position exceeds the programmed following error threshold, the axis is killed and motion stops. Since our application requires constant monitoring during a data run, we will be operating in Closed Loop mode.

Primary Feedback: Encoder
Secondary Feedback: not used
Primariy Output: not used

Perhaps Piotr can confirm or advise on the items to the left here.

Secondary Output: not used

PID Sample Rate: This determines the frequency at which the interface card sample the channel in order to detect any change of state of the system. The default value is set to 250 μsec. PID stands for "Proportional Integral Derivative Control." A PID is defined as: "A combination of proportional, integral and derivative control actions. This refers to a control method in which the controller output is proportional to the error, its time history, and the rate at which it is changing. The error is the difference between the observed and desired values of a variable that is under control action." In other words, the PID sample rate is the rate at which the control system queries the motion system in order to see if there is any discrepancy between the position the controller is set to place the detector system at, and where the detector is actually, physically located at. Any sensed position that is farther off than the tolerance set in the Following Error input box causes the translation system to automatically shut down in order for the user to make corrections and avoid damaging equipment.

Axis Enable: These controls merely enable (turn on) or disable (turn off) a connection between the interface card and the nuDrive channel for a particular translation axis.

Setup Axis 1 through 6:

Note: Axes 5 and 6 are not used since the nuDrive power amplifiers only address 4 channels each.

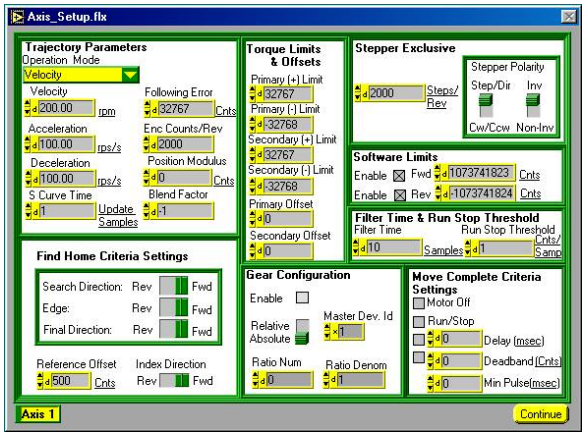

Figure 8

Trajectory Parameters: (Cnts. Below indicates counts.)

Operation Mode:

Absolute Position: In absolute position mode, target positions are interpreted with respect to an origin, reference, or zero position. The origin is typically set at a home switch, end of travel limit switch, or encoder index position. An absolute position move will use the preprogrammed values of acceleration, deceleration, s-curve, and velocity to complete a trajectory profile with an ending position equal to the loaded absolute target position. The length of an absolute move depends upon the loaded position and the current position when the move is started. If the target position is the same as the current position, no move will occur.

Caution: Any single move is limited to $\pm(2^{31}-1)$ counts or steps. An error is generated if you exceed this limit by loading a target position too far from the current position.

Relative Position: In relative position mode while motion is not in progress, loaded target positions are interpreted with respect to the current position at the time the value is loaded. A relative position move uses the preprogrammed values of acceleration, deceleration, s-curve and velocity to complete a trajectory profile with an ending position equal to the sum of the loaded relative target position and the starting position. If a relative move is started while motion is in progress, the new target position is calculated with respect to the target position of the move already in progress (considered to be the new starting position), as if that move had already completed successfully. Motion continues to the new relative position, independent of the actual position location when the new move is started.

Velocity: In velocity mode, the axis moves at the loaded (user entered) velocity until you execute a Stop Motion function, a limit is encountered, or a new velocity is loaded and you execute a Start Motion function. Load target positions have no effect in velocity mode. The direction of motion is determined by the sign of the loaded velocity. You can update velocity at any time to accomplish velocity profiling. Changes in velocity while motion is in progress uses the preprogrammed acceleration, deceleration, and s-curve values to control the change in velocity. You can reverse direction by changing the sign of the loaded velocity and executing a Start Motion function.

Relative To Capture: The relative-to-capture position mode is very similar to relative position mode, except that the position reference is the last captured position for the axis or axes. A relative-to-capture position move uses the preprogrammed values of acceleration, deceleration, s-curve and velocity to complete a trajectory profile with an ending position equal to the sum of the loaded target position and the last captured position.

Modulus Position: In modulus position mode, the loaded target position is interpreted within the boundaries of a modulus range and the direction of motion is automatically chosen to generate the shortest trajectory to the target. To load the modulus range execute the Load

Position Modulus function. Modulus position mode is typically used with rotary axes or for other similarly repetitive motion applications.

Velocity (rpm): Velocity values in RPM are converted to an internal 16.16 fixed-point format in units of counts (steps) per sample period (update period) before being used by the trajectory generator. FlexMotion can control velocity to 1/65,536 of a count or step per sample. You can calculate this minimum velocity increment in RPM with the following formula:

$$\text{RPM} = Vmin \times (1/Ts) \times 60 \times (1/R) \tag{12}$$

where $Vmin$ = 1/65,536 count/sample or step/sample,
Ts = sample period in seconds per sample,
60 = number of seconds in a minute, and
R = counts or steps per revolution.

For instance, a typical servo axis with 2,000 counts per revolution operating at the 250 µs update rate, the minimum RPM increment is:

$$(1/65{,}536) \times 4{,}000 \times 60/2{,}000 = 0.\mathit{00183105}\ \text{RPM} \tag{13}$$

RPM values stored in onboard variables are in double-precision IEEE format (f64).

Acceleration (rps/s): Acceleration and deceleration values in RPS/s are converted to an internal 16.16 fixed-point format in units of counts/sample ^{2} or steps/sample ^{2} before being used by the trajectory generator. You can calculate the minimum acceleration increment in RPS/s with the following formula:

$$\text{RPS/s} = Amin \times (1/Ts)^2 \times (1/R) \tag{14}$$

where Amin = 1/65,536 counts/sample ^{2} or steps/sample ^{2},
Ts = sample period in seconds per sample, and
R = counts or steps per revolution.

For a typical servo axis with 2,000 counts per revolution operating at the 250 ms update rate, the minimum RPS/s increment is calculated as follows:

$$(1/65{,}536) \times (4{,}000)^2/2{,}000 = 0.122070\ \text{RPS/s} \tag{15}$$

RPS/s values stored in onboard variables are in double-precision IEEE format (f64).

Deceleration (rps/s): See information above for Acceleration.

S Curve Time (Update Samples): S-Curve is defined as the profile (or plot) of the ramping up or down of velocity, as a function of time. This is designed to prevent abrupt impulses, and therefore jarring, of the object being moved by the translation system.

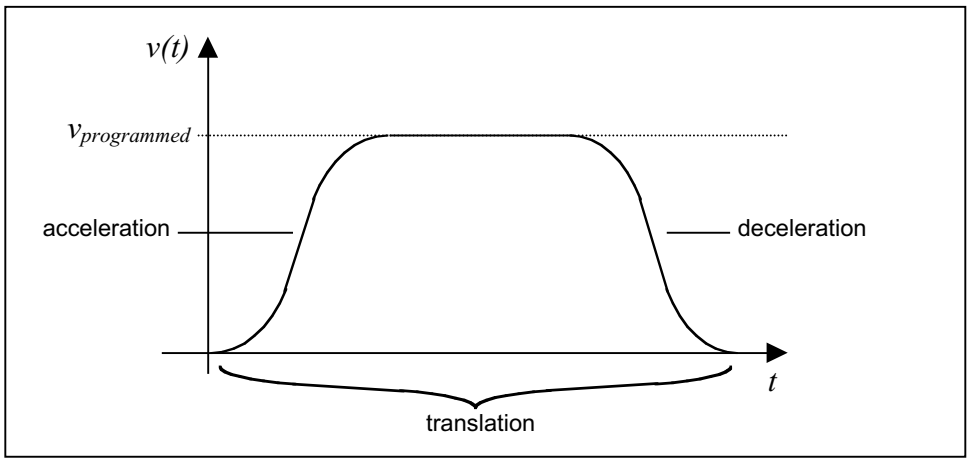

Figure 9

Following Error (Cnts): The following error option sets the maximum allowable following error. Following error is the difference between the instantaneous commanded trajectory position and the feedback position. If the absolute value of this difference exceeds the trip point, an internal kill stop is issued and the axis is disabled. This function is a safety feature used to protect the motion hardware and associated system components from damage when the position error gets excessive due to friction, binding, or a completely stalled motor. It will also protect you in case you load unobtainable values for velocity and/or acceleration.

This feature is available on all servo and closed-loop stepper axes. It has no effect on stepper axes running in open-loop mode. You can completely disable the following error feature by loading a zero (0) value.

Caution: Following error should not be disabled unless your application absolutely requires operating with greater than 32,787 counts of error.

When engineering or altering existing FlexMotion software, you can monitor following error status with the *Read Trajectory Status* or *Read per Axis Status* functions. A following error trip always sets the Motor Off status. You can further diagnose the cause of the trip by checking the torque limit status with the *Read DAC Limit Status* function.

In general, a following error trip is considered normal operation and does not generate an error. There are a few cases where an unexpected following error trip will generate a modal error: during Find Home or Find Index and while executing a stored program.

Enc Counts/Rev: (Encoder Counts per Revolution) All trajectory parameters for servo and closed-loop stepper axes are expressed in terms of quadrature encoder counts. Parameters for open-loop stepper axes are expressed in steps. For servo axes, the encoder resolution in counts per revolution determines the ultimate positional resolution of the axis.

For stepper axes, the number of steps per revolution depends upon the type of stepper driver and motor being used. For example, a stepper motor with 1.8°/step (200 steps/revolution) used in conjunction with a 10x microstep driver would have an effective

resolution of 2,000 steps per revolution. Resolution on closed-loop stepper axes is limited to the steps per revolution or encoder counts per revolution, whichever is more coarse.

Note 1: For stepper closed-loop configurations, where the encoder counts per revolution is greater than the steps per revolution, the range of the position parameters is reduced to $-(2^{31})$/counts or steps to $(2^{31}-1)$/counts or steps.

Note 2: Setting the maximum allowable count frequency for an encoder is useful for reducing the effect of noise on the encoder lines. Noise on the encoder lines can be interpreted as extra encoder counts. By setting the frequency to the lowest possible setting required for your motion application, you can ensure the highest degree of accuracy in positioning. In choosing the appropriate value, you should take into account the counts per revolution of your encoder and the maximum velocity for the axis in question.

Position Modulus (Cnts): Position modulus sets the position modulus value in counts (servo axes) or steps (stepper axes). The modulus range is from 0 (default) to $2^{31}-1$. In modulus position mode, the loaded target position is interpreted within the boundaries of a modulus range and the direction of motion is automatically chosen to generate the shortest trajectory to the target. To load the modulus range execute the Load Position Modulus function. Modulus position mode is typically used with rotary axes or for other similarly repetitive motion applications.

Blend Factor: The FlexMotion software can blend moves together with a programmable blend factor. The FlexMotion Infinite Trajectory Control Processing allows you to automatically and smoothly blend any move type into any other move without stopping the axis or axes involved.

The *Load Blend Factor* function controls how the *Blend Motion* function operates. Blending automatically starts a pending move on an axis or vector space when the move in process completes. Exactly when the pending move starts is determined by the loaded blend factor.

A blend factor of –1 causes the pending move to start when the existing move finishes its constant velocity segment and starts to decelerate, as shown in Figure 7. This blends the two moves together at the optimum blend point.

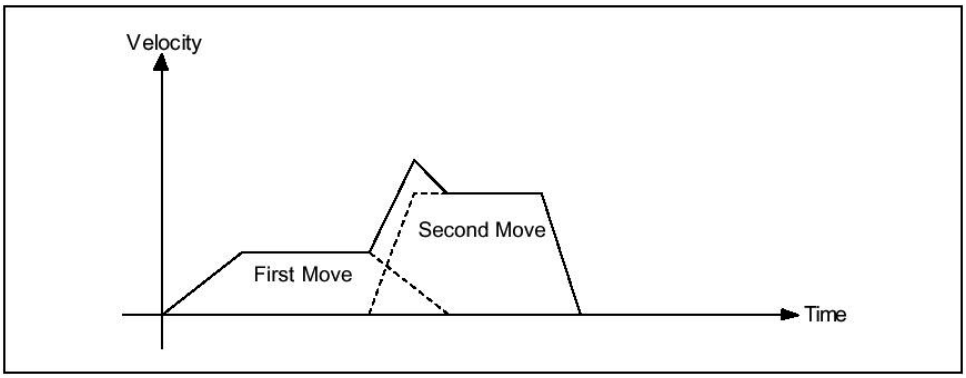

Figure 10: Blending with a Blend Factor of -1

If the two moves are at the same velocity, in the same direction, and have matching acceleration and deceleration, they will superimpose perfectly without a dip or increase in velocity.

For a vector move, if all of the axes are continuing in the same direction, the vector velocity remains constant. But, if one of the axes changes direction, the vector velocity does not remain constant during the transition phase.

A blend factor of zero (0) causes the pending move to start when the existing move fully completes its profile, as shown in Figure 10.

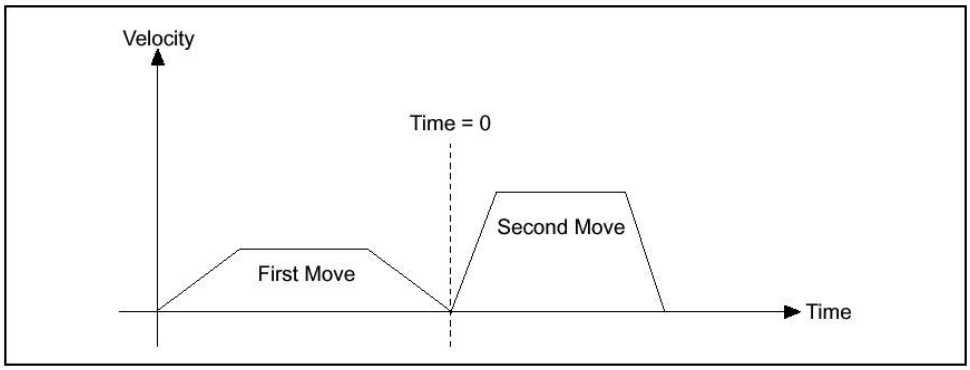

Figure 11: Blending with a Blend Factor of 0

Positive blend factors allow for a dwell at the end of the first move before the automatic start of the pending move, as shown in Figure 11. The blend factor dwell is programmed in milliseconds.

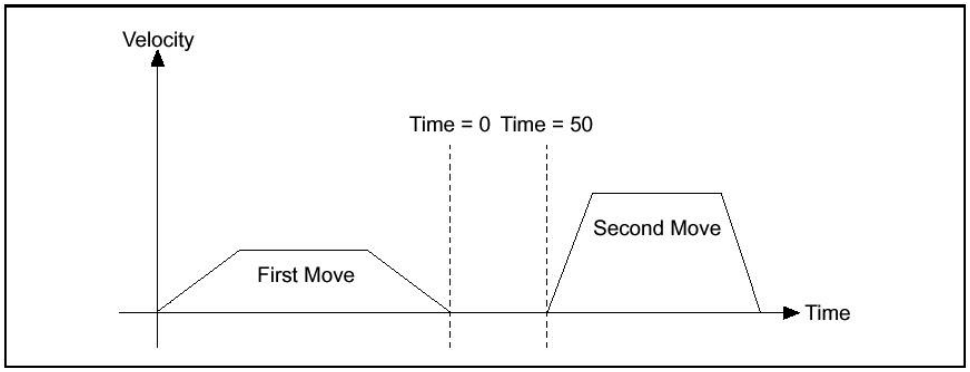

Figure 12: Blending with a Blend Factor of 50 ms

The maximum value of the positive blend factor depends upon the PIDrate that you set in the Enable Axes function, because the DSP delays the trajectory generators based on PID sample periods. The formula used to determine the maximum positive blend factor is as follows:

$$s = (time \_ 1000)/PIDrate \quad (16)$$

where s is the time in sample periods,
time is the positive blend factor value in milliseconds, and
PIDrate is in microseconds (62.5, 125, 188, 250, 312, 375, 438, or 500).

If s > 32,767, it is coerced to 32,767 sample periods. At a PIDrate of 500 ms, the maximum value of the positive blend factor is 16,383 ms and at a PIDrate of 250 ms, the maximum value is 8,192 ms.

If the first move has already completed when the Blend Motion function is executed, the second move will still wait the dwell time before starting. You can load blend factors to individual axes or to a vector space for coordinated blending of all axes in the vector space. When sent to a vector space, the blend factor is broadcast to all axes in the vector space to change the per-axis blend factors. If you later want to operate an axis independently with a different blend factor, you must execute the Load Blend Factor function again for that axis.

Note: All axes in a vector space must have the same blend factor. If the blend factors are different on each axis when you execute a *Blend Motion* function, an error is generated.

Torque Limits & Offsets: To operate in closed-loop mode, a stepper axis must have a primary feedback resource (encoder or ADC channel) mapped to it prior to enabling the axis. Refer to the Configure Axis Resources functions for more information on feedback resources. You can operate an axis with a primary feedback resource in either open or closed-loop mode and you can switch the mode at any time. You can still read the position of the mapped feedback resource even when the axis is in open-loop mode.

The *Read Position* function returns the instantaneous position of the specified axis. For servo axes, it returns the primary feedback position in counts. For open-loop stepper axes, it returns the number of steps generated. For closed-loop stepper axes, it converts the primary feedback position from counts to steps and then returns the value in steps. Closed-loop stepper axes require you to correctly load values of steps per revolution and counts per revolution to function correctly.

Primary (+) Limit: The Primary Positive Limit is the primary DAC positive torque (or velocity) limit. The range is –32,768 to + 32,767 (–10 V to +10 V) with a default value of 32,767 (+10 V).

Note: For closed-loop axes, this function always returns the position of the primary feedback resource.

Note: The positive limit cannot be less than the negative limit.

Primary (-) Limit: The Primary Negative Limit is the primary DAC negative torque (or velocity) limit. The range is –32,768 to + 32,767 (–10 V to +10 V) with a default value of –32,767 (–10 V).

Secondary (+) Limit: The Secondary Positive Limit is the optional secondary DAC positive torque (or velocity) limit. The range is –32,768 to + 32,767 (–10 V to +10 V) with a default value of 32,767 (+10 V).

Secondary (-) Limit: The Secondary Negative Limit is the optional secondary DAC negative torque (or velocity) limit. The range is –32,768 to + 32,767 (–10 V to +10 V) with a default value of –32,767 (–10 V).

Primary Offset: The Primary Offset is the primary DAC torque (or velocity) offset. The offset range is –32,768 to +32,767 (–10 V to +10 V) with a default value of 0 (0 V).

Secondary Offset: The Secondary Offset is the secondary DAC torque (or velocity) offset. The offset range is –32,768 to +32,767.

Requested information c this via emai to NI on May 2, 2001

Stepper Exclusive: Could find nothing describing this category in the User's Manual; contact NI with question re how to generically describe.

Steps/Rev: For stepper axes, the number of steps per revolution depends upon the type of stepper driver and motor being used. For example, a stepper motor with 1.8°/step (200 steps/revolution) used in conjunction with a 10x microstep driver would have an effective resolution of 2,000 steps per revolution. Resolution on closed-loop stepper axes is limited to the steps per revolution or encoder counts per revolution, whichever is more coarse.

Warning: For proper closed-loop operation, the correct values for steps/rev and counts/rev must be loaded with the *Load Counts/Steps per Revolution* function. Incorrect counts to steps ratio can result in failure to reach the desired target position and erroneous closed-loop stepper operation.

Stepper Polarity:

Step/Dir vs. Cw/Ccw: You use this setting to configure a stepper output to correctly interface with a stepper driver. FlexMotion supports the two industry standards for stepper control outputs. The most popular mode is Step and Direction where one output produces the step pulses and the other output produces a direction signal.

In clockwise (CW) and counter-clockwise (CCW) mode, the first output produces pulses when moving forward or CW while the second output produces pulses when moving reverse or CCW.

Inv vs. Non-Inv: In either Step/Dir or Cw/Ccw mode, you can set the active polarity with the polarity bit to be active low (inverting) or active high (noninverting). For example, in Step and Direction mode, the polarity bit determines whether a high direction output is forward or reverse. It also determines the resting states of outputs when they are not pulsing. The Configure Step Mode & Polarity setting is typically set for each stepper axis prior to using the axis for position control. Once the mode and polarity are set, they remain in effect until changed.

Software Limits:

Enable; Fwd (Cnts) (Forward): The limit inputs are typically connected to end-of-travel limit switches or sensors. An enabled limit input causes a halt stop on the axis when the input becomes active. You can configure each limit input to be active-low (inverting) or active-high (noninverting) with the Set Limit Input Polarity function. Active limit inputs also prohibit attempts to start motion that would cause additional travel in the direction of the limit. You can also use limit inputs as general-purpose inputs and read their status with the Read Limit Status function.

Note: For the end-of-travel limits to function correctly, the forward limit switch or sensor must be located at the positive (count up) end of travel and the reverse limit at the negative (count down) end of travel.

An active (and enabled) limit input transition on an axis that is part of a vector space move causes that axis to halt stop and the other axes in the vector space to decelerate to a stop. Similarly, software limits are often used to restrict the range of travel further and avoid ever hitting the hardware limit switches. An enabled software limit causes the axis to smoothly decelerate to a stop when the limit position is reached or exceeded. Even when disabled, you can poll the software limits by the host computer or use an onboard program to warn of an out of range position. For information about loading and reading the forward and reverse software limits, see the Load Software Limit Positions and the Read Limit Status functions.

Hardware limit inputs and software position limits are enhancements on the FlexMotion controller and are not required for basic motion control. You can operate all motion control functions without enabling or using these limits except the Find Home function, which requires enabled limit and home inputs for operation. Find Home does not require enabled software limits.

Note: If any axis in a vector space move exceeds an enabled software limit position, all axes in the vector space decelerate to a stop.

Enable: Rev (Cnts) (Reverse): See the information above for Enable; Forward limit.

Filter Time & Run Stop Threshold

Filter Time (Samples): Filter Time is the velocity filter time constant in update sample periods. The range for this parameter is 0 to 255 with a default value of 10 sample periods.

Run Stop Threshold (Cnts/Sample): Run Stop Threshold is the Run/Stop threshold velocity in counts/sample period (servo and closed-loop stepper axes) or steps/sample period (open-loop stepper axes). The range for this parameter is 1(default) to 32,767 sample periods.

Gear Configuration

Gearing is an advanced feature of FlexMotion and is used in applications where either the master axis is not under control of the FlexMotion controller or whenever extremely tight synchronization between multiple axes is required.

Enable: Selecting this check box enables and disables master-slave gearing functionality of slave axes. When gearing is enabled, the positions of the slave axes and their corresponding masters are recorded as their absolute gearing reference. From then on, as long as the gear ratio remains absolute, every incremental change of a master position is multiplied by the corresponding absolute gear ratio and applied to the slave axis.

Relative/Absolute: Enabled gear configuration settings load the gear ratio of the slave axis relative to its master and selects whether this ratio is absolute or relative.

When you enable the gear configuration settings, the positions of the slave and its master are recorded as their absolute gearing reference. From then on, as long as the gear ratio remains absolute, every incremental change of the master position is multiplied by the absolute gear ratio and applied to the slave axis or axis.

If a relative gear ratio is selected and loaded after gearing is enabled, the position of the master is recorded as its relative reference point and every incremental change from this reference point is multiplied by the relative gear ratio and applied to the slave axis or axis.

Caution: While changing an absolute gear ratio on the fly is allowed, you should be careful because the slave axis will jump with full torque to the position defined by the new ratio even when the master position has not changed.

Master Dev. Id: Master Device ID allows the user to indicate the ID number of the device to be used as the master gearing information source which the slave device is to be moved in relation to.

Ratio Num (-erator): Ratio numerator is the value you assign to the numerator of the electronic gearing ratio. The electronic gearing ratio is loaded as a numerator and denominator because it is a natural format for a ratio (numerator: denominator) and it allows a broad range of ratios, from 1:32,767 to 32,767:1. The ratio is always specified as slave relative to master (slave:master).

Ratio Denom (-inator): Ratio denominator is the value you assign to the denominator of the electronic gearing ratio.

Move Complete Criteria Settings: The Configure Move Complete Criteria setting is typically called by the software for each axis prior to using the axis for position control. Once the criteria are set, they remain in effect until changed. You can execute this function at any time. When an axis starts, its corresponding bit in the Move Complete Status register is reset to zero. When the move completes, the bit is set to one. You can check the status of an axis or axes at any time by polling the MCS register.

Motor Off: If the Motor Off bit is set, any condition that causes the axis to turn its motor off (a kill or following error trip) will satisfy this basic requirement for Move Complete. In other words, either Profile Complete OR Motor Off must be True for Move Complete to be True.

Run/Stop, Delay (msec), Deadband (Cnts): Run/Stop, Delay, and In Position, are optional conditions that are logically AND'd to further qualify the Move Complete status. If the Run/Stop bit is set, the axis must also be logically stopped for the move to be considered complete. For information about the Run/Stop status, refer to the *Configure Velocity Filter* function in the National Instrument's FlexMotion™ Software Reference Manual.

If the Delay bit is set, the axis must wait a user-defined delay after the other criteria are met before the move is considered complete. The user-defined delay parameter is typically used to wait the mechanical settling time so that a move is not considered complete until vibrations in the mechanical system have damped out. It can also be used to compensate for PID pull-in time due to the integral term. This pull-in is typically at velocities below the Run/Stop threshold.

Note: You can use the Delay parameter to guarantee a minimum time for the False state. The status will transition from Complete to Not Complete at the start of a move and stay in the Not Complete state for at least this delay time even in the case of a zero distance move.

Min Pulse (msec): Entering a value here sets the minimum time that the Move Complete status must stay True. A non-zero value for minPulse guarantees that the status stays in the True state for at least this minimum time even if another move starts immediately. You can use this feature to make sure that the host does not miss a Move Complete status when it polls the Move Complete Status register.

Find Home Criteria Settings: Hardware limit inputs and software position limits are enhancements on the FlexMotion controller and are not required for basic motion control. You can operate all motion control functions without enabling or using these limits except the Find Home function, which requires enabled limit and home inputs for operation. Find Home does not require enabled software limits.

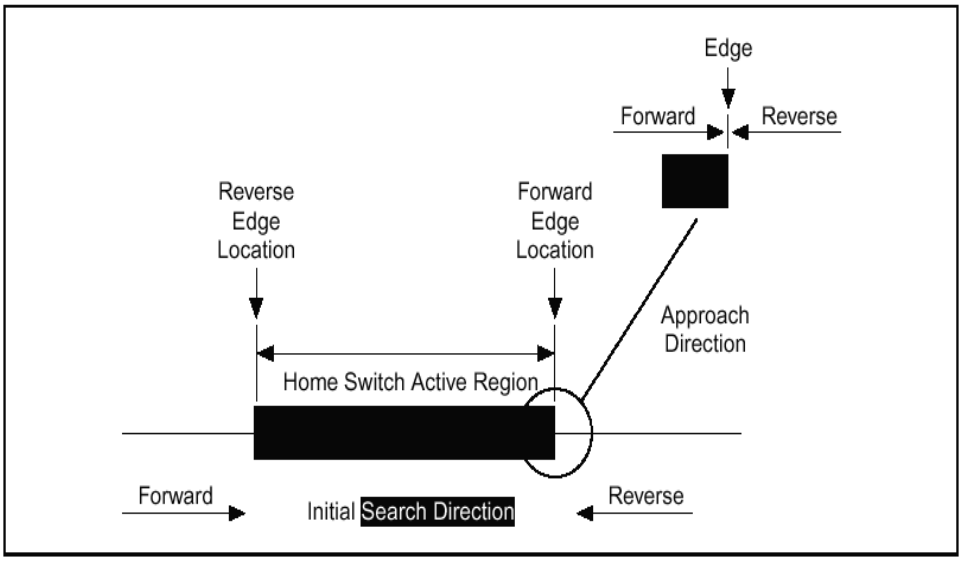

Figure 13: Find Home Definitions

Note: If any axis in a vector space move exceeds an enabled software limit position, all axes in the vector space decelerate to a stop.

Caution: You must enable both Limits and Home inputs prior to executing the Find Home function. If any of the axes limit or home inputs are disabled, the Find Home function does not start and a modal error is generated.

The Find Home function configures and initiates a search sequence for a home switch or other sensor. You can specify the initial search direction, the edge (rising or falling) of the home signal to stop on, and the direction you want to be travelling when you approach the specified home edge.

You can configure the find home sequence to detect either the rising or falling edge of the home signal, as shown in Figure 13. You can also set the polarity of the limit and home inputs with the Set Limit Input Polarity and Set Home Input Polarity functions, respectively. Once the home switch is found, motion proceeds to approach the home edge from the desired direction. If necessary, the axis travels past the home edge and reverses direction to approach it from the programmed direction. This portion of the sequence is executed at a fixed low speed (approximately 1/4 RPS) to smoothly approach the edge. This approach direction feature is used to minimize the effects of motion system windup, backlash, and/or home sensor hysteresis.

Make indents consistent.

Search Direction, Rev/Fwd: When the search direction is forward, the axis starts moving in the forward direction using the previously loaded values for acceleration, velocity, and s-curve. If the desired home signal transition is detected, the find home sequence continues based on the other control bits. If the forward limit switch is encountered before the home switch, the axis automatically reverses direction and continues searching for the home switch. If the reverse limit is encountered before the home switch, the sequence stops and the Home Found status is False. If a home switch exists, finding it is guaranteed. A similar search sequence is followed when the initial search direction is reverse.

Warning: Forward is defined as the direction of increasing position. The Forward and Reverse Limits must be located at the proper ends of travel for the Find Home sequence to function properly.

Edge, Rev/Fwd: This determines whether a given translation will stop on the home edge while moving in the reverse or forwad direction.

Final Direction, Rev/Fwd: This determines the final direction of the translation after either a leading or trailing edge of the home switch active region is encountered.

Reference Offset (Cnts): This allows the user to indicate, in units of positive or negative counts, a reference position some distance from the zero position.

Index Direction, Rev/Fwd: This allows the user to set the expected direction of an encoder index (marker) position signal. The encoder index signal is accurate to one quadrature count and provides a much more repeatable reference than using just a home switch edge.

Author question: what is the technical definition of an encoder index?

Setup Control Loop Parameters:

Note: Axes 5 and 6 are not used since the nuDrive power amplifiers only address 4 channels each.

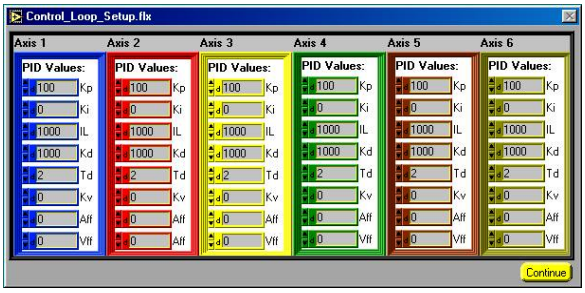

Figure 14

PID Values:

Kp: The proportional gain (Kp) determines the contribution of restoring force that is directly proportional to the position error. This restoring force functions in much the same way as a spring in a mechanical system.

Each sample period, the PID loop calculates the position error (the difference between the instantaneous trajectory position and the primary feedback position) and multiplies it by Kp to produce the proportional component of the 16-bit DAC command output.

The formula for calculating this proportional contribution is as follows:

$$\text{Vout(proportional)} = (20\ \text{V} / 2^{16}) \times \text{Kp} \times \text{Position Error} \qquad (17)$$

An axis with zero or too small a value of Kp will not be able to hold the axis in position and will be very soft. Increasing Kp stiffens the axis and improves its disturbance torque rejection. However, too large a value of Kp will often result in instability.

Ki: The integral gain (Ki) determines the contribution of restoring force that increases with time and thus ensures that the static position error in the servo loop is forced to zero. This restoring force works against constant torque loads to help achieve zero position error when the axis is stopped.

Each sample period, the position error is added to the accumulation of previous position errors to form an integration sum. This integration sum is scaled by dividing by 256 prior to being multiplied by Ki. Therefore, the formula for calculating the integral contribution in the 16-bit DAC command output is as follows:

$$\text{Vout(integral)} = (20\ \text{V} / 2^{16}) \times \text{Ki} \times \text{LIMIT(Integration Sum/256)} \qquad (18)$$

where LIMIT = shorthand for the effects of the integration limit described in the following sections.

Note: The scaling by 1/256 allows the use of integer values for the integral gain even when only a small amount of integral contribution is required.

In applications with small static torque loads, this value can be left at its default value of zero (0). For systems having high static torque loads, this value should be tuned to minimize position error when the axis is stopped.

Non-zero values of Ki, while reducing static position error, tend to increase position error while accelerating and decelerating. This effect can be mitigated through the use of the Integration Limit parameter. Too high a value of Ki will often result in servo loop instability. For these reasons, it is recommended that Ki be left at its default value of zero until the servo system operation is stable and then you can add a small amount of Ki to minimize static position errors.

Note: Ki has no effect when Ilim is equal to zero.

IL: The integration limit (Ilim) is used to clamp the contribution of the integral term in the PID loop. The scaled integration sum is compared to the integration limit and the lesser of the two values is multiplied by Ki to produce the integral term of the control output. This limiting function is indicated with LIMIT() in the following integral term output equation:

$$\text{Vout(integral)} = (20\ \text{V} / 2^{16}) \times \text{Ki} \times \text{LIMIT}((\text{Integration Sum}/256), \text{Ilim}) \quad (19)$$

You can use Ilim to limit excessive restoring forces and to minimize the adverse effects that integral compensation has during acceleration and deceleration.

Kd: The derivative gain (Kd) determines the contribution of restoring force proportional to the rate of change (derivative) of position error. This force acts much like viscous damping in a damped spring and mass mechanical system (for example, shock absorber).

The PID loop computes the derivative of position error every derivative sample period (a multiple of PID sample period; see the following section, *Derivative Sample Period*). This derivative term is multiplied by Kd every PID sample period to produce the derivative component of 16-bit DAC command output.

The formula for calculating the derivative contribution is as follows:

$$\text{Vout(derivative)} = (20\ \text{V} / 2^{16}) \times \text{Kd} \times (\text{pos\_err}(t1) - \text{pos\_err}(t0)) \quad (20)$$

where the time between t1 and t0 is the derivative sample period.

A non-zero value of Kd is required for all systems that use torque block amplifiers (where the command output is proportional to motor torque) for the servo loop operation to be stable. Too small a Kd value will result in servo loop instability.

With velocity block amplifiers (where the command output is proportional to motor velocity) you typically set Kd to zero or to a very small value.

Td: The derivative sample period parameter (Td) is used as a multiplier of the PID sample period (PID update rate). For information about setting the PID update rate, refer to the *Enable Axes* function. Td determines how often (in update samples) the derivative of position error is calculated.

The formula for calculating the derivative sample period from Td is as follows:

$$\text{Derivative Sample Period} = (\text{Td} + 1) \times \text{PID Sample Period} \quad (21)$$

Because the range for Td is 0 to 63, the shortest derivative sample period is as follows:

$$\text{Derivative Sample Period} = 1 \times 62.5\ \mu\text{s} = 62.5\ \mu\text{s} \quad (22)$$

The longest derivative sample period is as follows:

$$\text{Derivative Sample Period} = 64 \times 500\ \mu\text{s} = 32\ \text{ms} \quad (23)$$

Adjusting Td provides greater flexibility in tuning the PID loop derivative term. As Td is increased, you can use a proportionally lower value of Kd for similar results. You should start the Td parameter at its default value of 2 and make small adjustments as required by your motion system configuration.

For low inertia systems, Td should be set to a small value (0 or 1) so that the derivative is calculated often enough to provide adequate damping for servo loop stability. Systems with higher inertia can benefit from larger values of Td. Because the higher inertia means that the position error can not change quickly, it is acceptable to calculate the derivative less often. This means you can use a lower value of Kd, have the same effective amount of damping and the system will be smoother with less torque noise from the derivative term. In higher inertia systems, using a Td of zero and therefore a larger value for Kd results in increased torque noise and motor heating without any improvement in system stability.

Kv: When an axis is configured with a secondary feedback encoder, you can use that encoder for velocity feedback. The velocity feedback gain (Kv) is used to scale this velocity feedback before it is added to the other components in the 16-bit DAC command output.

Note: Velocity feedback is only available from encoders. It is not available from ADC channels.

It is possible to use Kv with only one feedback encoder. Map the encoder resource as both the primary and secondary resource for the axis. Velocity feedback gain (Kv) is similar to derivative gain (Kd) except that it scales the velocity estimated from the secondary feedback resource only. The derivative gain scales the derivative of the position error, which is the difference between the instantaneous trajectory position and the primary feedback position. Like the Kd term, the velocity feedback derivative is calculated every derivative sample period and the contribution is updated every PID Sample Period.

The formula for calculating the velocity feedback contribution is as follows:

$$\text{Vout} = (20\ \text{V} / 2^{16})\ \text{Kv}\ (\text{position}(t1) - \text{position}(t0)) \qquad (24)$$

where the time between t1 and t0 is the derivative sample period.

Velocity feedback is estimated through a combination of speed dependent algorithms. At high speeds, velocity is simply the change in position per sample. At low speeds, the estimator seamlessly transitions to a 1/T method that measures the time between encoder counts and then calculates the inverse. This method is used for smoother performance when estimating velocities less than one encoder count per sample derivative sample period.

Using Kv and a secondary feedback encoder creates a minor velocity feedback loop. This is very similar to the traditional analog servo control method using a tachometer and a velocity block amplifier and is commonly referred to as dual-loop feedback. Dual-loop feedback is most useful when the primary position sensor (encoder or analog transducer) is located on the end-effector for improved accuracy, and is separated from the motor by gears, ballscrews, belt drives, and/or other mechanical apparatus with potentially poor dynamics. In this case, it can be difficult to achieve a high performance, stable control system without using the minor loop velocity feedback from an encoder mounted directly on the back of the motor. Typically, Kd is set to zero when Kv is used. However, FlexMotion allows you to use both Kv and Kd terms simultaneously for improved performance.

Note: Operating with zero derivative gain (Kd) and either velocity feedback or a velocity block amplifier is often referred to as PIVff mode.

You can operate FlexMotion in PID mode, PIVff mode, or in a combination of both modes, by using Kd, Kv, or both.

Aff: The acceleration feedforward gain (Aff) determines the contribution in the 16-bit DAC command output that is directly proportional to the instantaneous trajectory acceleration. Aff is used to minimize following error (position error) during acceleration and deceleration and can be changed at any time to tune the PID loop.

Using acceleration feedforward is an open-loop compensation technique and cannot affect the stability of the system. However, if you use too large a value of Aff, following error during acceleration and deceleration can reverse, thus degrading rather than improving performance.

Vff: The velocity feedforward gain (Vff) determines the contribution in the 16-bit DAC command output that is directly proportional to the instantaneous trajectory velocity. This value is used to minimize following error during the constant velocity portion of a move and can be changed at any time to tune the PID loop.

Using velocity feedforward is an open-loop compensation technique and cannot affect the stability of the system. However, if you use too large a value of Vff, following error during the constant velocity portion can reverse, thus degrading rather than improving performance. Velocity feedforward is typically used when operating in PIVff mode with either a velocity block amplifier or substantial amount of velocity feedback (Kv). In these cases, the uncompensated following error is directly proportional to the desired velocity. You can reduce this following error by applying velocity feedforward. Increasing the integral gain (Ki) will also reduce the following error during constant velocity but only at the expense of increased following error during acceleration and deceleration and reduced system stability. For these reasons, increased Ki is not the recommended solution.

Velocity feedforward is rarely used when operating in PID mode with torque block amplifiers. In this case, because the following error is proportional to the torque required (not to the velocity), it is typically much smaller and velocity feedforward is not required.

Setup Motion I/O Parameters:

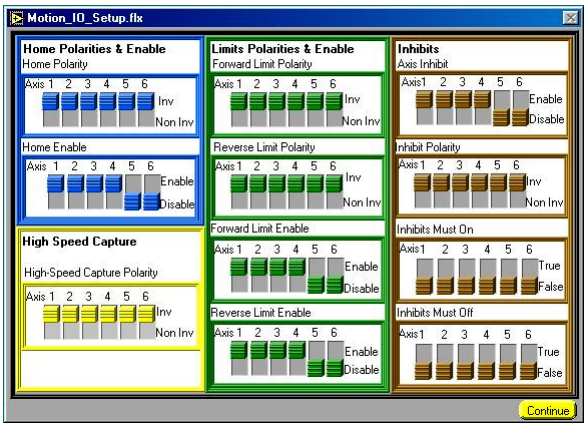

Figure 15

Home Polarities & Enable:

Home Polarity, Inv/Non Inv: The *Set Home Input Polarity* function defines the active polarity for each home input as either inverting or noninverting. Inverting polarity means that an active-low input corresponds to a logical True or On state. Conversely, noninverting polarity means that an active-high input corresponds to a logical True (On) state. You can enable home inputs to cause halt stops when the input becomes active with the *Enable Home Inputs* function. You can also use a home input as a general-purpose input and read its status with the *Read Home Input Status* function.

Home Enable, Enable/Disable: The *Enable Home Inputs* function enables/disables any combination of axis home inputs. An enabled home input causes a halt stop on the axis when the input becomes active. You can configure each home input to be active-low (inverting) or active-high (noninverting) with the *Set Home Input Polarity* function. You can also use a home input as a general-purpose input and read its status with the *Read Home Input Status* function. Home inputs are an enhancement on the FlexMotion controller and are not required for basic motion control. You can operate all motion control functions without enabling or using the home inputs except the *Find Home* function, which requires enabled limit and home inputs for operation.

High Speed Capture Polarity, Inv/Non Inv: The *Set High-Speed Capture Polarity* function defines the active polarity for each high-speed capture input as either inverting or noninverting. Inverting polarity means that an active-low input corresponds to a logical True or On state. Conversely, noninverting polarity means that an active-high input corresponds to a logical True or On state.

High-speed capture inputs are an integral part of the encoder resources. You can execute this function indirectly on axes or directly on encoder resources. When sent to multiple axes, this function sets the polarity of the high-speed capture inputs of the encoders mapped to the corresponding axes.

You can enable high-speed capture inputs to capture instantaneous encoder position when the input becomes active with the *Enable High-Speed Position Capture* function. You can also use a high-speed input as a general-purpose input and read its status with the *Read High-Speed Capture Status* function.

Limits Polarities & Enable:

Forward Limit Polarity, Inv/Non Inv; Reverse Limit Polarity, Inv/Non Inv: The *Set Limit Input Polarity* function defines the active polarity for each forward and reverse limit input as either inverting or noninverting. Inverting polarity means that an active-low input corresponds to a logical True or On state. Conversely, noninverting polarity means that an active-high input corresponds to a logical True (On) state. You can enable limit inputs to cause halt stops when the input becomes active with the *Enable Limits* function. You can also use a limit input as a general-purpose input and read its status with the *Read Limit Status* function.

Forward Limit Enable, Enable/Disable; Reverse Limit Enable, Enable/Disable: The *Enable Limits* function enables/disables any combination of axis limits.You can enable the physical limit inputs (hardware) or the logical position limits (software) depending upon the **limitType** selected. You can enable or disable forward and reverse limits separately. You can enable both software and hardware limits on an axis or axes by calling this function twice. The limit inputs are typically connected to end-of-travel limit switches or sensors. An enabled limit input causes a halt stop on the axis when the input becomes active. You can configure each limit input to be active-low (inverting) or active-high (noninverting) with the *Set Limit Input Polarity* function. Active limit inputs also prohibit attempts to start motion that would cause additional travel in the direction of the limit. You can also use limit inputs as general-purpose inputs and read their status with the *Read Limit Status* function.

Note: For the end-of-travel limits to function correctly, the forward limit switch or sensor must be located at the positive (count up) end of travel and the reverse limit at the negative (count down) end of travel.

An active (and enabled) limit input transition on an axis that is part of a vector space move causes that axis to halt stop and the other axes in the vector space to decelerate to a stop.

Similarly, software limits are often used to restrict the range of travel further and avoid ever hitting the hardware limit switches. An enabled software limit causes the axis to smoothly decelerate to a stop when the limit position is reached or exceeded. Even when disabled, you can poll the software limits by the host computer or use an onboard program to warn of an out of range position. For information about loading and reading the forward and reverse software limits, see the *Load Software Limit Positions* and the *Read Limit Status* functions.

Hardware limit inputs and software position limits are enhancements on the FlexMotion controller and are not required for basic motion control. You can operate all motion control functions without enabling or using these limits except the *Find Home* function, which requires enabled limit and home inputs for operation. *Find Home* does not require enabled software limits.

Note: If any axis in a vector space move exceeds an enabled software limit position, all axes in the vector space decelerate to a stop.

Inhibits: The *Configure Inhibit Outputs* function enables/disables and sets the polarity (inverting or noninverting) of the axis inhibit outputs. When enabled, a per-axis inhibit output is linked to the motor off state of the corresponding axis. A killed axis (motor off) forces the corresponding inhibit output On. When the axis is active, the inhibit output is Off. Inhibit outputs are typically used to disable the servo amplifier or stepper driver for power savings, safety, or specific application reasons.

Note: Killing a servo axis also zeros its DAC output. With torque block amplifiers this means that the motor freewheels whether or not the amplifier is disabled. With velocity block servo amplifiers or stepper drivers, the motor does not freewheel unless the amplifier/driver is disabled with the inhibit output.

You can also use inhibit outputs as general-purpose outputs. Disabled inhibit outputs ignore the state of their corresponding axis and can be directly controlled through the *Set Inhibit MOMO* function.

You can configure the active polarity of each inhibit output as inverting or noninverting. Inverting polarity means that a logical True or On state corresponds to an active-low output. Conversely,

noninverting polarity means that a logical True (On) corresponds to an active-high output. The inhibit polarity is always in effect, whether the inhibit is linked to its axis (enabled) or directly controlled thought the *Set Inhibit MOMO* function.

Axis Inhibit, Enable/Disable: (see above)

Inhibit Polarity, Inv/Non Inv: (see above)

Inhibits Must On, True/False: (see above)

Inhibits Must Off, True/False: (see above)

Notice to the reader: Due to time constraints in the construction of this document, the reader is referred to the document Motion Control; FlexMotion Software Reference Manual (august 1999 Edition; part number 3219438-01) for explanations of the features below.

Other Settings Options in the `Single_Axis_Test.flx` Window

Adv. Options (Advanced Options) in the `Advanced_Options.flx` window:

Run Stop & Filter Time:

Filter ?? (Samples):

Run Stop Threshold (Cnts / Sample):

Move Complete Criteria Settings:

Motor Off

Run / Stop

Delay (msec)

Deadband ?? (Cnts)

Min Pulse (msec)

Torque Limits & Offsets

Primary [+] Limit

Primary [-] ?? Limit

Secondary [+] Limit

Secondary [-] ?? Limit

Primary Offset

Secondary Offset

BP Settings, in the `Position_BP_Settings.flx` window:

Position BreakPoint Settings

Position Break point (Counts (steps))

Break point Settings:

Breakpoint Off

Breakpoint Options

Don’t Affect

Breakpoint Modulus

Velocity Threshold (RPM):

Technical Specifications of System Components

The following is a detailed description and listing of technical specifications of the various components listed in the section titled “General Description” earlier in this document.

Desktop Computer

The desktop computer directly containing the interface cards and directly controlling the NuDrive power amplifiers is itself controlled remotely from the Counting House via a local area network. The computer system has (approximately) the following specifications:

- a Pentium II, 350Mhz microprocessor
- operating system: Microsoft Windows

National Instruments Software

LabView™

This is a standard implementation of LabVIEW™ (ver. 5.1.) It has been installed in order to provide the environment in which to operate the FlexMotion™ Motion Control software.

FlexMotion™

The purpose of FlexMotion™, ver. 4.5 is to directly control the PCI7344 FlexMotion™ controller interface cards. This vendor-developed software is a grouping of VI’s (virtual instruments) which “call” other, sub-VI’s to effectively address the NI PCI7344 controller interface cards, send commands through them, access data about the state of the motion control system through them, and allow the user to monitor the experimental environment, and make necessary operational settings and adjustments.

NI-488.2

NI-488.2 creates the internal, software-to-hardware connections necessary for the controller software to correctly talk to the 2 NI PCI7344 FlexMotion™ controller interface cards.

Vendor contact information

National Instruments Corporate Headquarters, 6504 Bridge Point Parkway, Austin, TX 78730-5039; Phone: (512)794-0100; Support: (512)795-8248; Support Fax: (512)794-5678; Fax-on-Demand Support: (512)418-1111; Email: support@natinst.com; FTP site: ftp.natinst.com; Website: http://www.natinst.com

NI PCI7344 FlexMotion™ Controller Interface Cards

This appendix lists the hardware and software performance specifications for the 7344 controller.

Servo Performance

PID update rate range..... 62.5 to 500 µs/sample

Max PID update rate .....62.5 µs/axis

4-axis PID update rate .....250 µs total

Trajectory update rate .....Same as PID update rate

Multi-axis synchronization .....< 1 update sample

Position accuracy

Encoder feedback ..... ±1 quadrature count

Analog feedback ..... ±1 LSB

Long-term velocity accuracy ..... Oscillator based, ±100 ppm

Double-buffered trajectory parameters

Absolute position range ..... $\pm 2^{31}$ counts

Max relative move size ..... $\pm 2^{31}$ counts

Velocity range ..... 1 to ±20,000,000 counts/s

RPM range ..... 10^{-5} to 10^{6} revolutions/minute

Acceleration/deceleration ..... 4,000 to 128,000,000 counts/s ^{2}

RPS/s range ..... 10^{-1} to 10^{8} revolutions/s ^{2}

S-Curve time range ..... 1 to 32,767 samples

Following error range ..... 0 to 32,767 counts

Gear ratio ..... ±32,767:1 to 1:32,767

Servo control loop modes ..... PID, PIVff, S-Curve, Dual Loop

PID (Kp, Ki and Kd) gains ..... 0 to 32,767

Integration limit (Ilim) ..... 0 to 32,767

Derivative sample period (Td) ..... 1 to 63 samples

Feedforward (Aff, Vff) gains ..... 0 to 32,767

Velocity feedback (Kv) gain ..... 0 to 32,767

Servo command analog outputs

Voltage range ..... ±10 V

Resolution ..... 16 bits (0.000305 V/ LSB)

Programmable torque (velocity) limits

Positive limit ..... ±10 V (–32,768 to + 32,767)

Negative limit ..... ±10 V (–32,768 to + 32,767)

Programmable offset ..... ±10 V (–32,768 to + 32,767)

Stepper Performance

Trajectory update rate range ..... 62.5 to 500 µs/sample

Max update rate ..... 62.5 µs/axis

4-axis update rate ..... 250 µs total

Multi-axis synchronization ..... < 1 update sample

Position accuracy

Open-loop stepper ..... 1 full, half, or microstep

Encoder feedback ..... ±1 quadrature count

Analog feedback ..... ±1 LSB

- Long-term velocity accuracy ..... Oscillator based, ±100 ppm
- Double-buffered trajectory parameters
  - Position range ..... $\pm 2^{31}$ steps
  - Max relative move size ..... $\pm 2^{31}$ steps
  - Velocity range ..... 1 to 8,000,000 steps/s
  - RPM range ..... 10^{-5} to 10^{6} revolutions/minute
  - Acceleration/deceleration ..... 4,000 to 128,000,000 steps/s ^{2}
  - RPS/s range ..... 10^{-1} to 10^{8} revolutions/s ^{2}
  - S-curve time range ..... 1 to 32,767 samples
  - Following error range ..... 0 to 32,767 counts
  - Gear ratio ..... ±32,767:1 to 1:32,767
- Stepper outputs
  - Max pulse rate ..... 8 MHz (full, half, and microstep)
  - Min pulse width ..... 60 ns at 8 MHz
  - Step output mode ..... Step and direction or CW/CCW
  - Voltage range ..... 0 to 5 V
    - Output low voltage ..... < 0.6 V at 64 mA sink
    - Output high voltage ..... Open collector w/ built-in 3.3 kΩ pull-up to +5V
  - Polarity ..... Programmable, active-high or active-low

System Safety

- Watchdog timer function ..... Resets board to startup state
  - Watchdog timeout ..... 63 ms
- Shutdown input
  - Voltage range ..... 0 to 12 V
    - Input low voltage ..... 0.8 V
    - Input high voltage ..... 2 V
  - Control ..... Disable all axes and command outputs

Motion I/O

- Encoder inputs ..... Quadrature, incremental, single-ended
  - Max count rate ..... 20 MHz
  - Voltage range ..... 0 to 12 V
    - Input low voltage ..... 0.8 V
    - Input high voltage ..... 2 V
  - Min index pulse width ..... 60 ns
- Forward, reverse, and home inputs
  - Number of inputs ..... 12 (3 per axis)

- Voltage range ..... 0 to 12 V
  - Input low voltage ..... 0.8 V
  - Input high voltage ..... 2 V
- Polarity ..... Programmable, active-high or active-low
- Control ..... Individual enable/disable, stop on input, prevent motion, Find Home

Trigger inputs

- Number of inputs ..... 4 (Encoders 1 through 4)
- Voltage range ..... 0 to 12 V
  - Input low voltage ..... 0.8 V
  - Input high voltage ..... 2 V
- Polarity ..... Programmable, active-high or active-low
- Min pulse width ..... 83 ns
- Capture latency ..... <100 ns
- Capture accuracy ..... 1 count
- Max repetitive capture rate ..... 1 kHz

Breakpoint outputs

- Number of outputs ..... 4 (Encoders 1 through 4)
- Voltage range ..... 0 to 5 V
  - Output low voltage ..... < 0.6 V at 64 mA sink
  - Output high voltage ..... Open collector w/ built-in 3.3 kΩ pull-up to +5V
- Polarity ..... Programmable, active-high or active-low
- Inhibit/enable output
  - Number of outputs ..... 4 (1 per-axis)
  - Voltage range ..... 0 to 5 V
    - Output low voltage ..... < 0.6 V at 64 mA sink
    - Output high voltage ..... Open collector w/ built-in 3.3 kΩ pull-up to +5V
  - Polarity ..... Programmable, active-high or active-low
  - Control ..... MustOn/MustOff or automatic when axis off
- Analog inputs
  - Number of inputs ..... 8, multiplexed
    - Number for user signals 4
    - Number for calibration 4
  - Voltage range ..... ±10 V, ±5 V, 0–10 V, 0–5 V
  - Input resistance ..... 10 kΩ min
  - Resolution ..... 12 bits
  - Analog reference output ..... 7.5 V (nominal)

- Reference drift ..... ±30 ppm/°C typ.
- INL ..... ±1 LSB
- DNL ..... ±1 LSB
- Offset error
  - Unipolar ..... ±5 LSB
  - Bipolar ..... ±10 LSB
- Gain error
  - Unipolar ..... ±10 LSB
  - Bipolar ..... ±10 LSB
- Conversion time ..... 6 µs
- Multiplexor scan rate ..... 50 µs/enabled channel

Analog outputs

- Number of outputs 4
- Voltage range ..... ±10 V
- Output current ..... ±5 mA
- Resolution ..... 16 bits (0.000305 V/ LSB)
- Gain accuracy ..... ±3%
- Drift ..... 100 ppm/°C typ.

Digital I/O

Ports ..... 4, 8-bit ports

- Line direction ..... Individual bit programmable

Inputs

- Voltage range ..... 0 to 5 V
  - Input low voltage ..... 0.8 V
  - Input high voltage ..... 2.0 V
- Polarity ..... Programmable, active-high or active-low

Outputs

- Voltage range ..... 0 to 5 V
  - Output low voltage ..... < 0.45 V at 24 mA sink
  - Output high voltage ..... > 2.4 V at 24 mA source
- Polarity ..... Programmable, active-high or active-low
- PWM outputs
  - Number of PWM outputs 2
  - Max PWM frequency ..... 32 kHz
  - Resolution 8-bit
  - Duty cycle range ..... 0 to (255/256)%

Clock Sources..... Internal or external

Power Requirements (Max)

- PCI-7344 and PXI-7344

  +5 V (±3%) ..... 1 A

  +12 V (±3%) ..... 30 mA

  –12 V (±3%)..... 30 mA

  Power consumption..... 5.7 W

- FW-7344

  Voltage range ..... 9 to 25 VDC

  Power consumption..... 30 W

Physical

Dimensions (Not Including Connectors)

PCI-7344..... 17.5 by 9.9 cm (6.9 by 3.9 in.)

PXI-7344 ..... 16 by 10 cm (6.3 by 3.9 in.)

FW-7344..... 30.7 by 25.4 by 4.3 cm (12.1 by 10.0 by 1.7 in.)

Connectors

Motion I/O connector..... 68-pin female high-density VHDCI type

32-bit digital I/O connector..... 68-pin female high-density VHDCI type

Environment

Operating temperature..... 0 to 55 °C

Storage temperature ..... –20 to 70 °C

Relative humidity range..... 10 to 90% (noncondensing)

nuDrive™ Power Amplifiers

The two nuDrive™ amplifiers are capable of addressing 4 control channels each. The first amplifier drives the Parker Automation linear positioning tracks of the two photodetector crates on the “beam in” side of the detector annulus, as well the overall annulus rotation. The second amplifier drives the Parker Automation linear positioning tracks of the Cerenkov detectors on the “beam out” side of the detector annulus. For any given linear positioning track the NuDrive™ performs the following functions:

- sends stepper motor drive pulses to either advance or retreat the position of a detector mount platform along a linear track
- receives encoder information from the stepper motor to estimate the position of the mount platform along the linear track
- reads impulses from the induction-type limit sensors along the linear track (the FlexMotion™ control software running in LabVIEW™ then compares the motor’s position (encoded)

information with that detected by the limit sensors along the linear tracks. Whenever a discrepancy between the two is detected, beyond a certain user-set limit, the FlexMotion™ software halts the translation of the linear drive and alerts the user.)

The nuDrive™ technical specifications are as follows:

Servo amplifiers:

Type	Elmo Motion Control SSA 8/100
Peak current limit (2s)	0.8 to 10 A (default 10 A)
Continuous current limit	0.8 to 8 A (default 5 A)
DC-bus motor voltage	48 VDC
PWM frequency	20 kHz
Continuous power (all axes combined)	325 W

Stepper Drivers

Type	Intelligent Motion Systems (IMS) IM483, bipolar chopper
Current per phase	0.36-4.0 A peak (0.26-2.8 A RMS) (factory setting is 1.4 A peak)
Motor bus voltage	24 or 40 VDC
Microstepping selections	x2, 4, 8, 16, 32, 64, 128, 256 x5, 10, 25, 50, 125, 250 (default is 10 times)

Encoder Interface (each axis)

Inputs	Quadrature, incremental
Differential input threshold	±0.3 V (typical)
Single-ended input threshold	TTL/CMOS
Voltage range	0 to 5 VDC
Noise filter (RC time constant)	100 ns
Max quadrature frequency	1 MHz

Limit and Home Switch Inputs (each axis)

Noise filter (RC time constant)	10 µ

Configurable I/O

Compatibility	Signal pass-through

Connectors

Encoders	8-pin terminal blocks (1 per axis)
Limits	6-pin terminal blocks (1 per axis)
Motor	5-pin terminal blocks (1 per axis)
I/O	6-pin terminal blocks (2 total)
E-Stop	3-pin terminal block (1 total)
AC power	Detachable AC power cord (IEC standard type)

Safety

Installation Category II, Pollution Degree 2

Environment

Operating temperature	0 to 45 °C (32 to 112 °F)
Storage temperature	-20 to 70 °C (-4 to 158 °F)
Humidity	10 to 90% (noncondensing)

Power Supply

Input voltage	115 VAC ±15%, 50-60 Hz
Input Fuse	
24VDC units	F5 (5 A, 5 x 20 mm)
40 and 48 VDC units	F10 (10 A, 5 x 20 mm)

Host Bus Voltage Interlock

Undervoltage threshold	4 VDC

Dimensions

Width	42.9 cm (16.9 in.)
Height	13.2 cm (5.2 in.) (3 U type)
Depth	30.5 cm (12 in.)
Weight	20-35 lb (depending on model)

Vendor contact information

National Instruments Corporate Headquarters, 6504 Bridge Point Parkway, Austin, TX 78730-5039; Phone: (512)794-0100; Support: (512)795-8248; Support Fax: (512)794-5678; Fax-on-Demand Support: (512)418-1111; Email: support@natinst.com; FTP site: ftp.natinst.com; Website: http://www.natinst.com

Parker Automation Linear Positioning Tracks

The 6 linear positioning tracks, manufactured by Parker Hannifin Corp., are of the 406000XR Series. To quote the product manual: "The 406XR is a linear positioning table which includes a square rail guide/linear bearing system, precision ground or rolled ballscrew drive, protective covering with strip seal protection, and a clear extruded aluminum finish."

The track stepper motor, in addition to providing translation force, also supplies a "holding torque" so as to fix the detector mounting stage until further translation is required. If the stepping loses power, it is possible for the mounting stage to creep when the track is positioned vertically and under mechanical load. To prevent this, electromagnetic brakes are implemented.

The electromagnetic brake attached to the non-motor end of each track drive shaft are used to prevent the track from "backdriving" when it is positioned vertically. It provides a 5.6 N-m holding torque and requires 24VDC, 0.31mA to deactivate. The brake is engaged when *not* powered. It is intended that the brake is to be powered from the same electrical outlet box as the NuDrive amplifiers so that it will automatically engage in the event of a power failure. Otherwise it will remain disengaged when supplied with constant current.

The rotary encoder for the track stepper motor requires 5VDC, 135mA (which is supplied by the nuDrive™ power amplifier.) It provides A/B quadrature and reference mark, with a differential line output. It operates with a resolution of 1250 lines per revolution, 5000 counts post quadrature, with an accuracy of +/- 2 arc minutes.

The 3 induction-type linear encoders (limit sensors) attached to each linear track are digital output, tape scale type encoders. They require 5VDC, 150mA power. They provide an A/B quadrature and reference mark, differential line drive output. They operate at a resolution of 1.0, 0.5 or 0.1 microns (depending on configuration), with an accuracy of +/- 3 micron, after linear slope correction. The reference mark has a resolution of +/- 2 resolution bits, unidirectional. They are capable of sensing speeds of: 3 meters/second

(at 1 micron resolution scale), 1.5 m/s (at 0.5 micron resolution scales) or 0.3 m/s (at 0.1 micron resolution scale.) They also are capable of sensing 30 g's acceleration at all resolutions.

Caution: these encoders are destroyed if their power polarity is inadvertently reversed. *Exercise extreme caution* to prevent damage to them.

Vendor contact information

Parker Hannifin Corporation, Daedal Division, 1140 Sandy Hill Road, Irwin, PA 15642; Phone: (724)861-8200; (800)245-6903; Fax: (724)861-3330; Email: ddlcat@parker.com; Website: www.daedalpositioning.com

---

This document was assembled by Matthew Breuer, with assistance from Dr. Piotr Decowski of Smith College, and Dr. Ross Hicks and Jon Celli, of the Medium Energy Nuclear Physics Group, Physics Department, University of Massachusetts, Amherst, MA. Please call any errors to the attention of the author by sending email to: neomistic@physics.umass.edu

We owe our appreciation for a major portion of this document to National Instruments, through whose kind permission we are allowed to reproduce several sections of the of the *Motion Control, FlexMotion Software Reference Manual*.