Allen-Bradley

Servo Positioning Assembly

(Cat. No. 1771–QC Series B)

User Manual

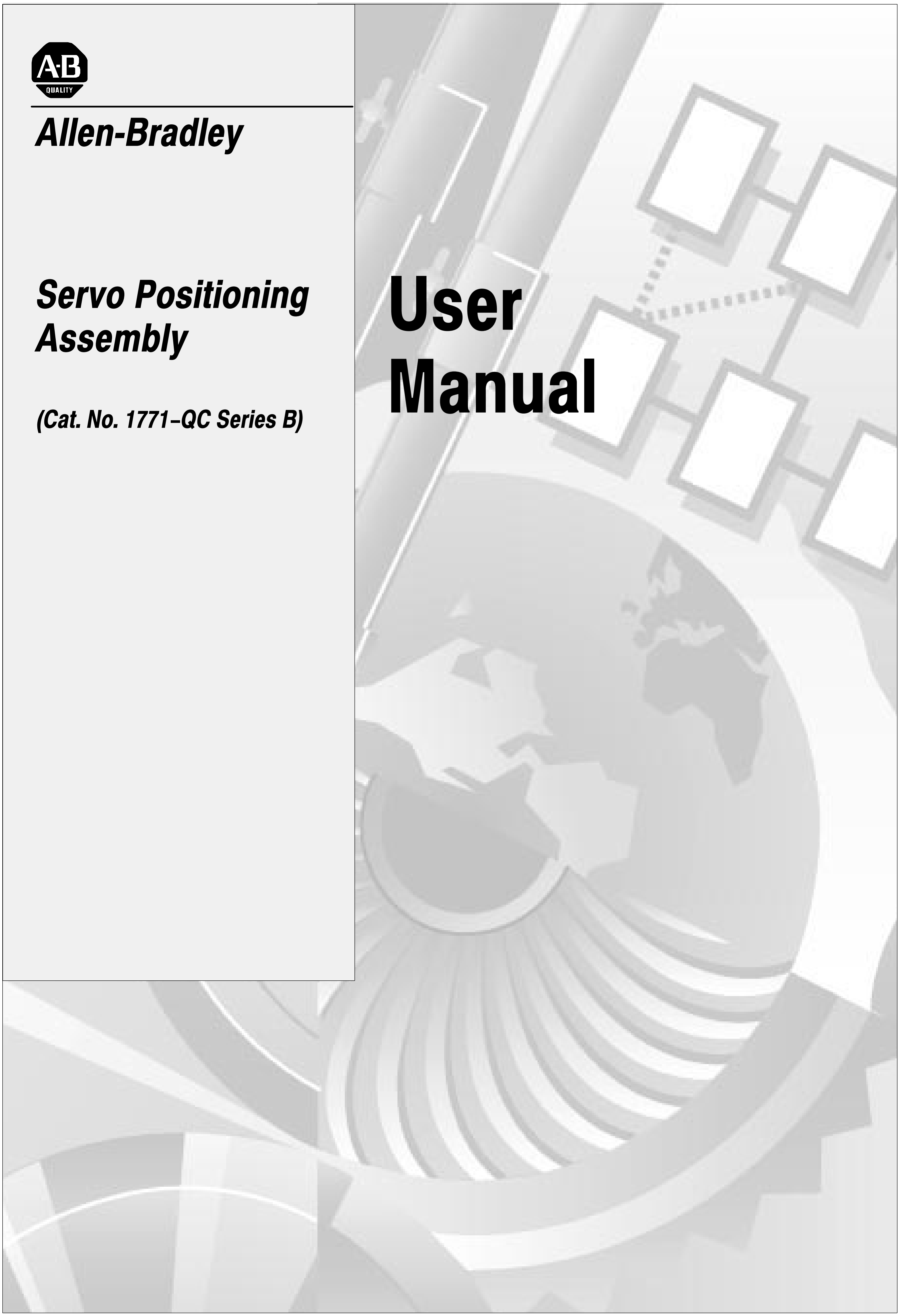


Table of Contents

Chapter 1

Using This Manual

Manual's Purpose

This manual shows you how to use the series B Servo Positioning Assembly (cat. no. 1771-QC). If you have a series A Servo Positioning Assembly, refer to publication 1771-817.

Audience

To use the servo positioning assembly, you must be able to program and operate an Allen-Bradley PC processor. In particular, you must be able to program block transfer instructions.

In this manual, we assume that you know how to do this. If you don't, refer to the appropriate manual for the PC processor you will be using. Consult our Publication Index (publication SD499) for a list of our publications.

Vocabulary

Some inconsistency exists throughout industry in the nomenclature used for components of closed-loop servo positioning systems. Therefore, as you read this manual, you should be aware of the names we use for these components.

- We refer to the Servo Controller Module (cat. no. 1771-M3) as the 1771-M3 controller.
- We refer to the Servo Expander Module (cat. no. 1771-ES) as the 1771-ES expander.
- We refer to the device that receives the velocity command signal from the 1771-ES expander as the servo drive. The servo drive converts ac power to dc power for the servo motor in proportion to the velocity command signal. What we refer to here as the servo drive, others may refer to as a servo controller. So, if you refer to this device as a servo controller, be aware of our nomenclature as you read this manual.
- PC refers to programmable controller.

For an extensive list of terms we use this publication, refer to the glossary in appendix A.

Manual Organization

This manual is organized into the following chapters:

Chapter	Title	What's Covered
2	Introducing the Servo Positioning Assembly	an overview of the servo positioning assembly, its applications, functions, and features
3	Positioning Concepts	concepts of closed-loop positioning, including velocity loop, positioning loop, and feed forward
4	Positioning with Allen-Bradley PC's	the servo positioning assembly's position in a servo system, and the servo positioning assembly's communication with the PC processor
5	Describing Hardware	describing the servo positioning assembly, its specifications, and its compatibility with other hardware components you will need for a closed-loop positioning system
6	Installing the Assembly	installing the servo positioning assembly and interconnecting hardware
7	Formatting and Interpreting Data Blocks	formatting parameter, move description, and control data for block transfer to the servo positioning assembly-interpreting status and diagnostic data received in block transfer from the servo positioning assembly
8	Programming	generating a ladder-diagram program to transfer data blocks between the PC data table and the servo positioning assembly
9	Integrating Axes	adjusting the servo positioning assembly for optimum operation with the machine axis it is to control
10	Troubleshooting	using indicator status and status-block information to diagnose and correct problems

Chapter 2

Introducing the Servo Positioning Assembly

Chapter Objectives

This chapter gives you an overview of the servo positioning assembly, its applications, functions and features.

What is the Servo Positioning Assembly?

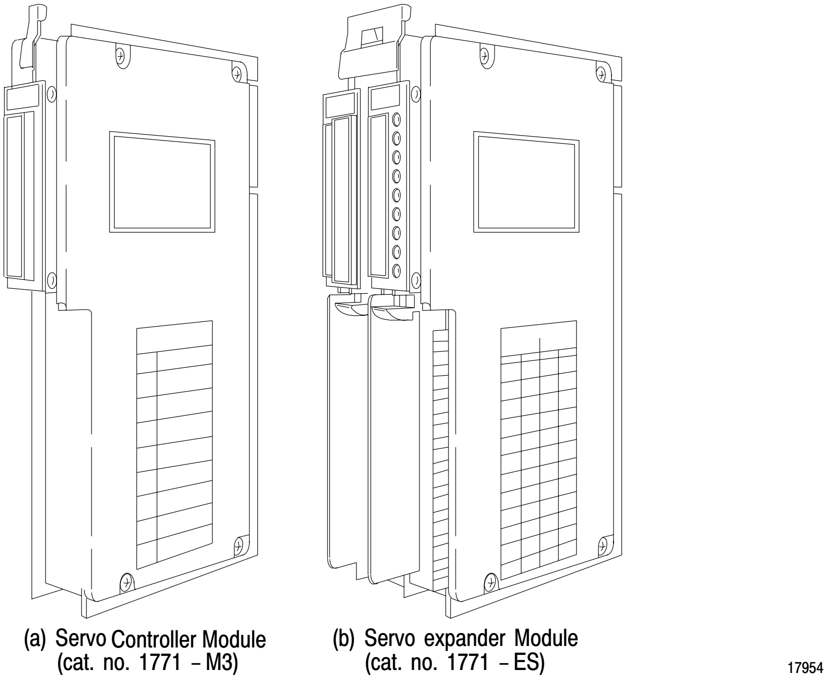
A servo positioning assembly controls the motion of one of your axes. It consists of:

- one Servo Controller Module (cat. no.1771-M3)
- one Servo Expander Module (cat. no. 1771-ES) that includes two Field Wiring Arms (cat. no. 1771-WB)

With a basic servo positioning assembly (plus a servo drive, motor, tachometer, and encoder) you can control the motion of one user-supplied machine axis. You can add a second 1771-ES expander to control a second axis and a third 1771-ES expander to control a third axis. A 1771 I/O chassis can accommodate one 1771-M3 controller and a maximum of three 1771-ES expanders.

The 1771-M3 controller requires one I/O chassis slot; it requires no wiring (figure 2.1a). You can install it at any I/O slot in the I/O chassis. The 1771-ES expander requires a pair of slots that make up an I/O module group (Figure 2.1b). You make all wiring connections to the 1771-ES expander.

Figure 2.1
Servo Positioning Assembly



Its Applications

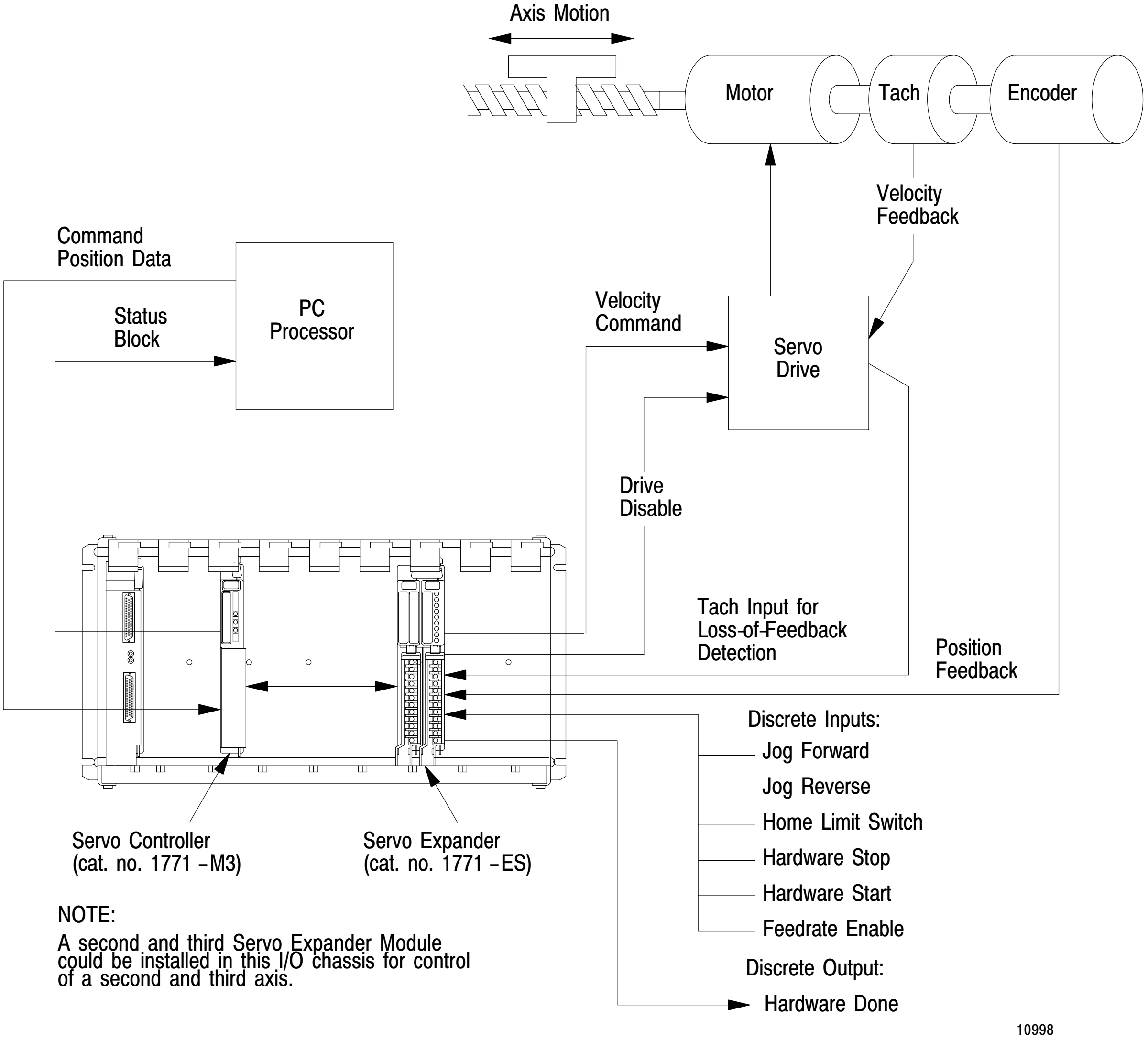
Typical applications for a servo positioning assembly include positioning for:

- grinding
- transfer lines
- material handling
- drilling
- riveting
- rotary indexing
- v-belt cutting
- glass cutting

Its Function

Figure 2.2 shows a servo system for closed-loop axis control. The 1771-M3 controller communicates with the 1771-ES expander through I/O chassis backplane connections.

Figure 2.2
Closed-loop Axis Servo System



The PC processor sends commands and user-programmed data from the data table to the 1771-M3 controller as directed by a block-transfer write instruction. The 1771-M3 controller coordinates the block transfer automatically, keeping ladder diagram programming to a minimum.

Based on information it receives from the processor, the 1771-M3 controller sends axis motion commands to the 1771-ES expander.

The 1771-ES expander closes the servo positioning loop. It commands axis motion by generating an analog voltage for your servo drive. Every 2.4 milliseconds (ms) it updates this analog output voltage according to motion commands from the 1771-M3 controller, discrete inputs, and feedback from your encoder. The 1771-ES expander is able to provide this fast servo sample rate because the update is independent of the I/O scan.

A drive-disable output provides a signal to disable the servo drive in conditions such as loss-of-feedback or a hardware-stop signal. A hardware-done output signals the completion of each single-step move. Discrete hardware inputs include:

- hardware stop
- jog forward
- jog reverse
- home limit
- feedrate enable
- hardware start

The 1771-M3 controller sends axis status and diagnostic data to the data table as directed by a block-transfer read instruction. Because axis-command and status data is stored in the data table, axis motion control can interact with other axes, discrete I/O, and report generation.

Its Features

See the following table for a list of the many useful benefits you’ss derive from an A-B servo positioning assembly.

Feature	Benefit
incremental digital encoder feedback	precise closed-loop positioning
absolute or incremental positioning commands	programming flexibility
programmable gain break	precise positioning at low speed with stability at high speed
programmable acceleration/deceleration	optimize the machine cycle time over varying loads
programmable in-position band	flexible positioning accuracy
programmable jog rates	flexible manual positioning
programmable dwell	precise dwell times
excess-following-detection	automatic drive shutdown if the axis following error becomes too large
loss-of-feedback detection	allow automatic drive shutdown during a move if tachometer or encoder feedback is lost
software travel limits	guards against axis overtravel
backlash takeup	compensates for mechanical backlash
offset	compensates for a variation in tool length or fixture dimension
preset	easy redefinition of axis coordinates

Feature	Benefit
optically isolated analog output[1]	guards against noise entering the backplane circuits and limits the potential for damage due to improper connection
external hardware start[1]	synchronizes moves with other axes
encoder input selectable for high-true or low-true[1]	compatibility with a wider range of encoders
synchronized start of feedrate override[1]	activates a pre-loaded feedrate override value to change speed on several axes simultaneously
sensing of customer power supply loss[1]	an orderly shutdown of the servo system and to provide you with this diagnostic information
feed forwarding[1]	to allow you to reduce following error by up to 99.9% without increasing instability
constant-velocity command[1]	runs an axis continuously at a selected velocity (could apply to controlling a conveyor with no programmed end point)
moveset override[1]	Modifies a moveset while it is being executed
diagnostic words in the status block[1]	provide your ladder-diagram program with access to diagnostic information
for	hardware and program troubleshooting

[1] These features are only available on the series B servo positioning assembly.

Summary

This chapter was intended to be very general. Upcoming chapters cover these topics in greater detail. To prepare for those details, read about positioning concepts in chapter 3.

Chapter 3

Positioning Concepts

Chapter Objectives

This chapter presents positioning concepts and terminology. If you are thoroughly familiar with the concepts of closed-loop servo positioning, you can skip ahead to chapter 4.

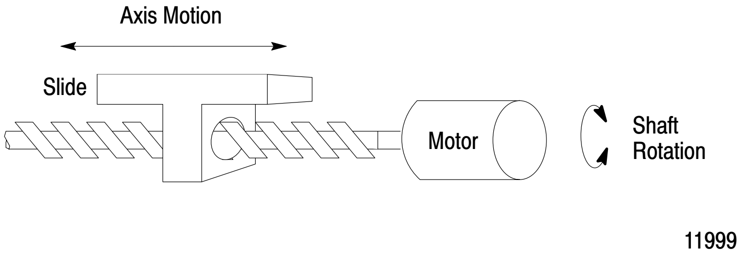
Closed-Loop Positioning

Closed-loop positioning is a precise means of moving an object from one position to another. Typically, an electric motor supplies the mechanical power, and the needed motion is linear. Therefore, we must convert the rotary motion of the motor's shaft to linear motion.

Axis Motion

One common method of converting rotary motion to linear motion is with a leadscrew (Figure 3.1)

Figure 3.1
Leadscrew Converting Rotory Motor Motion Into Linear Axis Motion



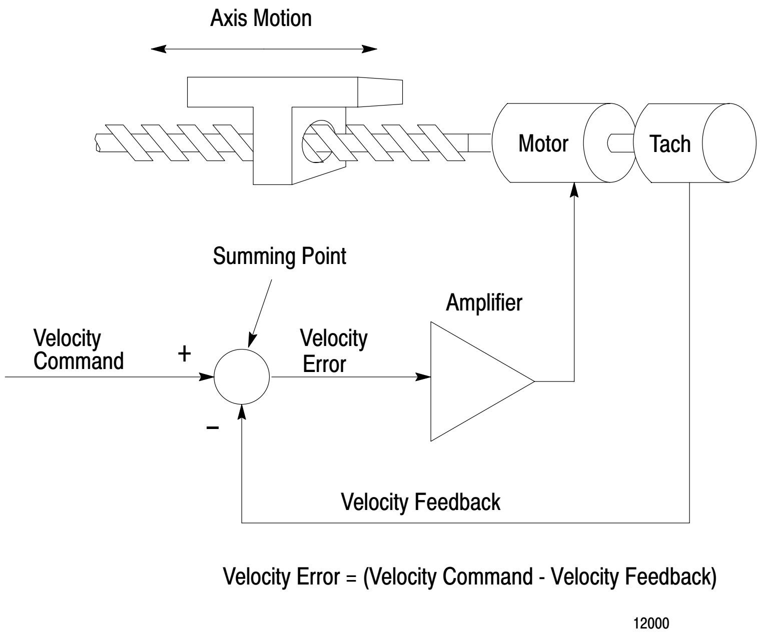
The leadscrew assembly is referred to as the axis. A leadscrew assembly consists of a long threaded shaft (the leadscrew) and slide having an internal thread that matches the leadscrew. When the motor rotates the leadscrew clockwise, the slide moves forward. When the motor rotates the leadscrew counterclockwise, the slide moves backward.

Velocity Loop

Most closed-loop servo positioning installations use a dc motor to power the leadscrew. To accurately control the velocity of the dc motor, we need a velocity loop (Figure 3.2).

The velocity loop contains a summing point, an amplifier, and a tachometer. A tachometer is a precision generator that produces a voltage signal directly proportional to the angular velocity of the motor shaft. The output of the tachometer is the velocity feedback signal which is subtracted from the velocity command signal. The difference is the velocity error signal that is amplified to provide power for the motor to run at the commanded velocity.

Figure 3.2
Velocity Loop

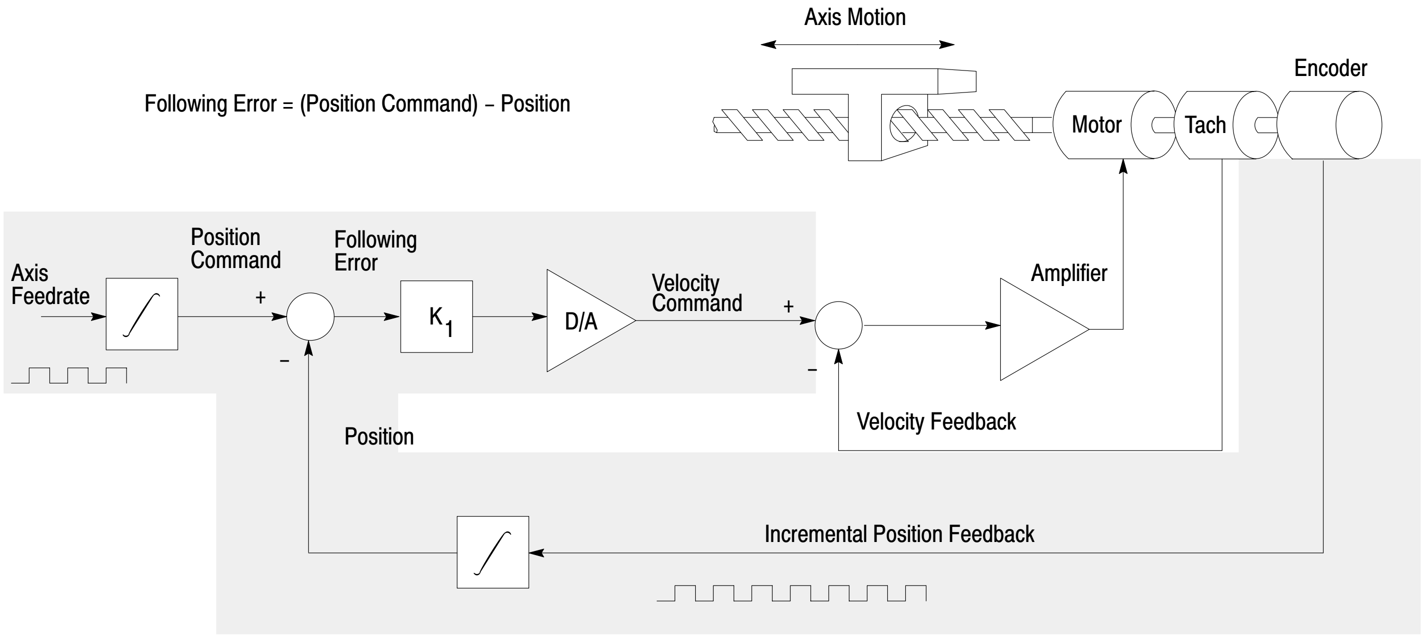


Whenever the velocity deviates from the commanded velocity, the velocity feedback signal adjusts the velocity error signal until the velocity matches the velocity command signal.

Positioning Loop

When we want to move the slide a specific distance, we can turn the motor on at a specific velocity for a specific length of time. However, this could produce imprecise positioning. To accurately control the position of the slide, we need a positioning loop (Figure 3.3).

Figure 3.3
Velocity Loop and Positioning Loop



12001

The positioning loop includes a summing point, an amplifier, a D/A converter, and an incremental digital encoder to produce a position feedback signal. The axis feedrate is integrated in a register to produce the position command value. Incremental position feedback is integrated in a register to produce the actual position value. The position value is subtracted from the position command value. The difference is the following error, which is amplified and converted to an analog velocity command signal. This signal directs the axis to move in the right direction; the position value moves closer to the position command value.

The following error is a function of the axis velocity divided by the positioning-loop gain (K1). The following error is multiplied by the gain to generate the velocity command. Gain is expressed in ipm/mil (where 1 mil - 0.001 in) or mmpm/mil (where 1 mil = 0.001 mm).

For example, with a velocity of 100 ipm and a gain of 1 ipm/mil, the following error is:

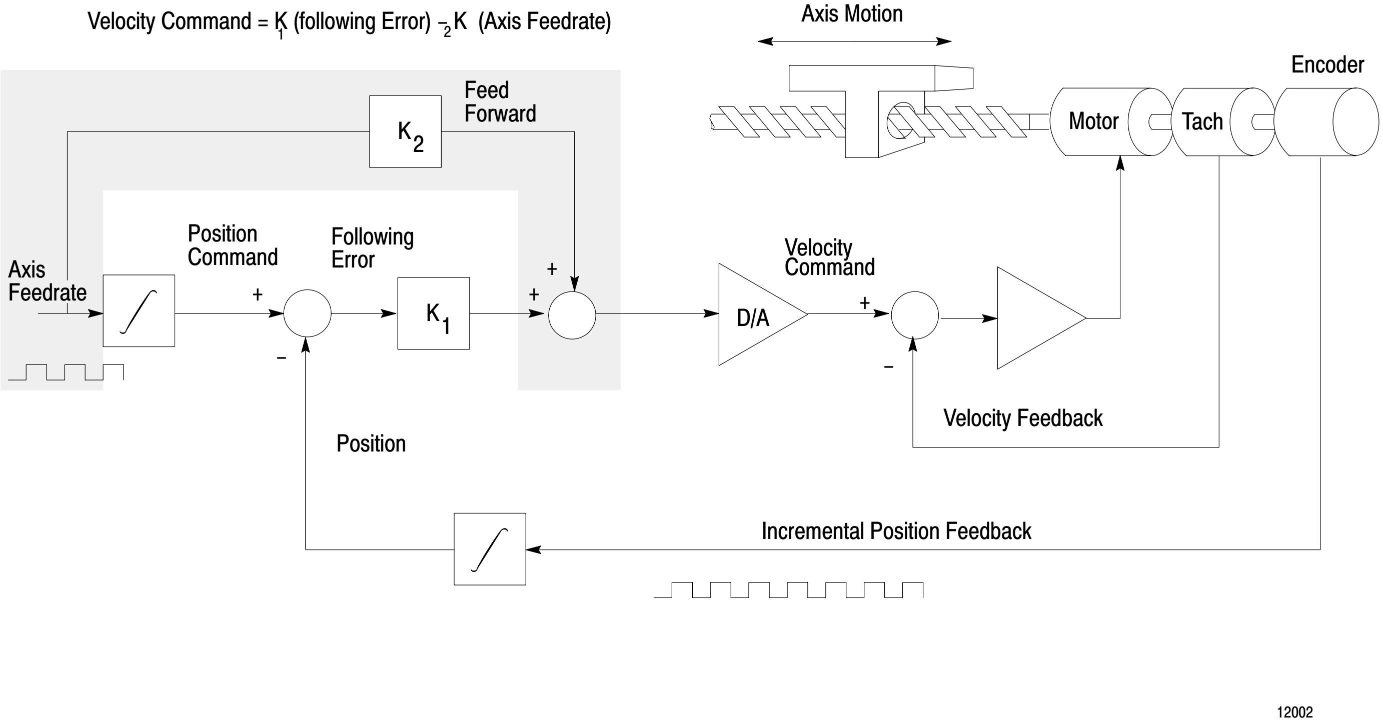
$$2\text{following error} = \frac{\text{velocity}}{\text{gain}} = \frac{100\text{ ipm}}{1\text{ ipm/mil}} = 100\text{ mil}$$

When you increase the gain, you decrease the following error and decrease the cycle time of the system. However, the gain that you can use is limited by the drive, the motor, and the machine; a gain that is too large causes instability.

Feed Forward

To decrease the following error without increasing the gain, we can add a feed forward component (Figure 3.4).

Figure 3.4
Velocity Loop, Positioning Loop, and Feed Forwarding



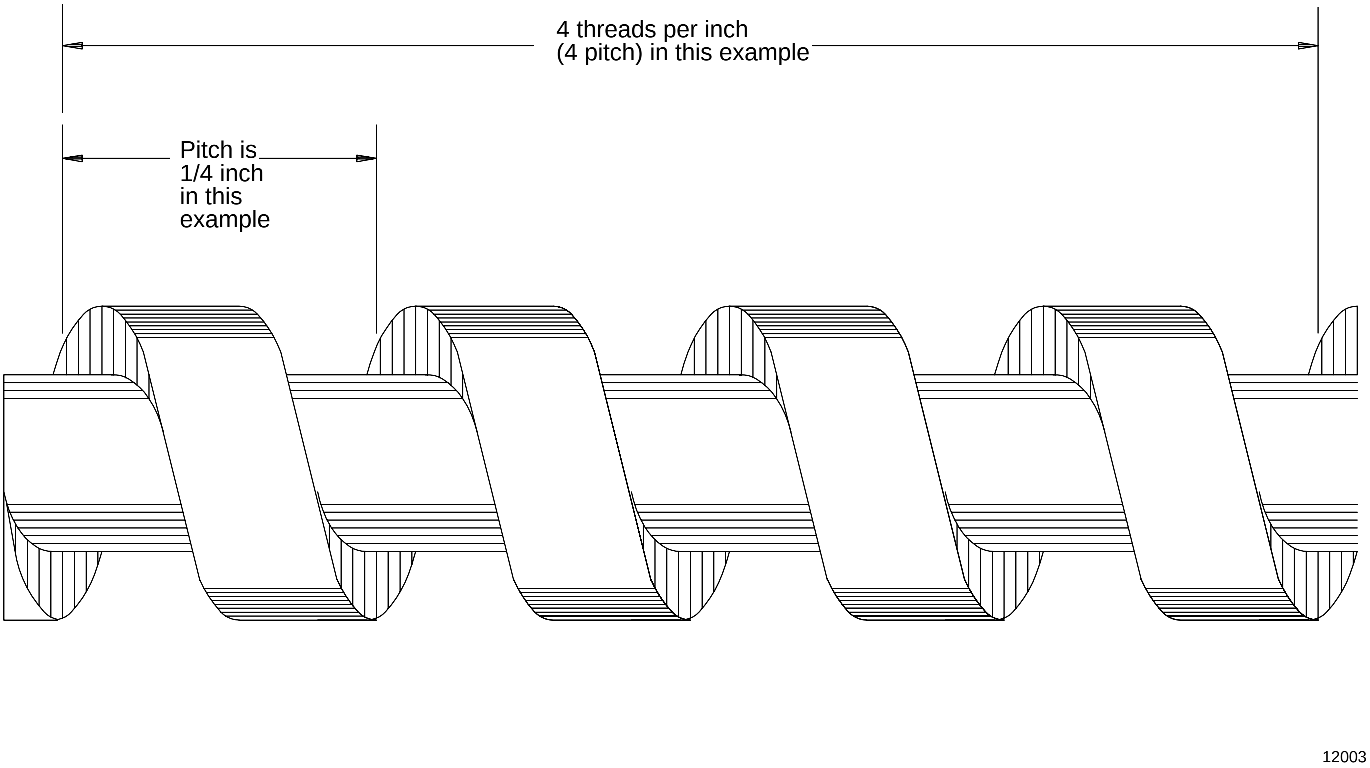
Feed forwarding requires an additional summing point and an amplifier. The axis feedrate is multiplied by the feed-forward gain (K2) to produce the feed-forward value. The feed-forward value is added to the following error multiplied by the gain to generate the velocity command.

Without feed forward, the axis will not begin to move until the axis feedrate builds up enough following error to generate a sufficiently large velocity command to overcome friction and inertia to move the axis. However, the feed-forward value could generate a velocity command to move the axis almost immediately. This immediate response keeps the actual position closer to the position command, thereby reducing the following error.

Leadscrew Pitch

Leadscrew pitch is the linear distance from one peak of the screw thread to the next. A leadscrew with a pitch of 1/4 inch is shown in Figure 3.5.

**Figure 3.5
Leadscrew Example Showing Pitch**



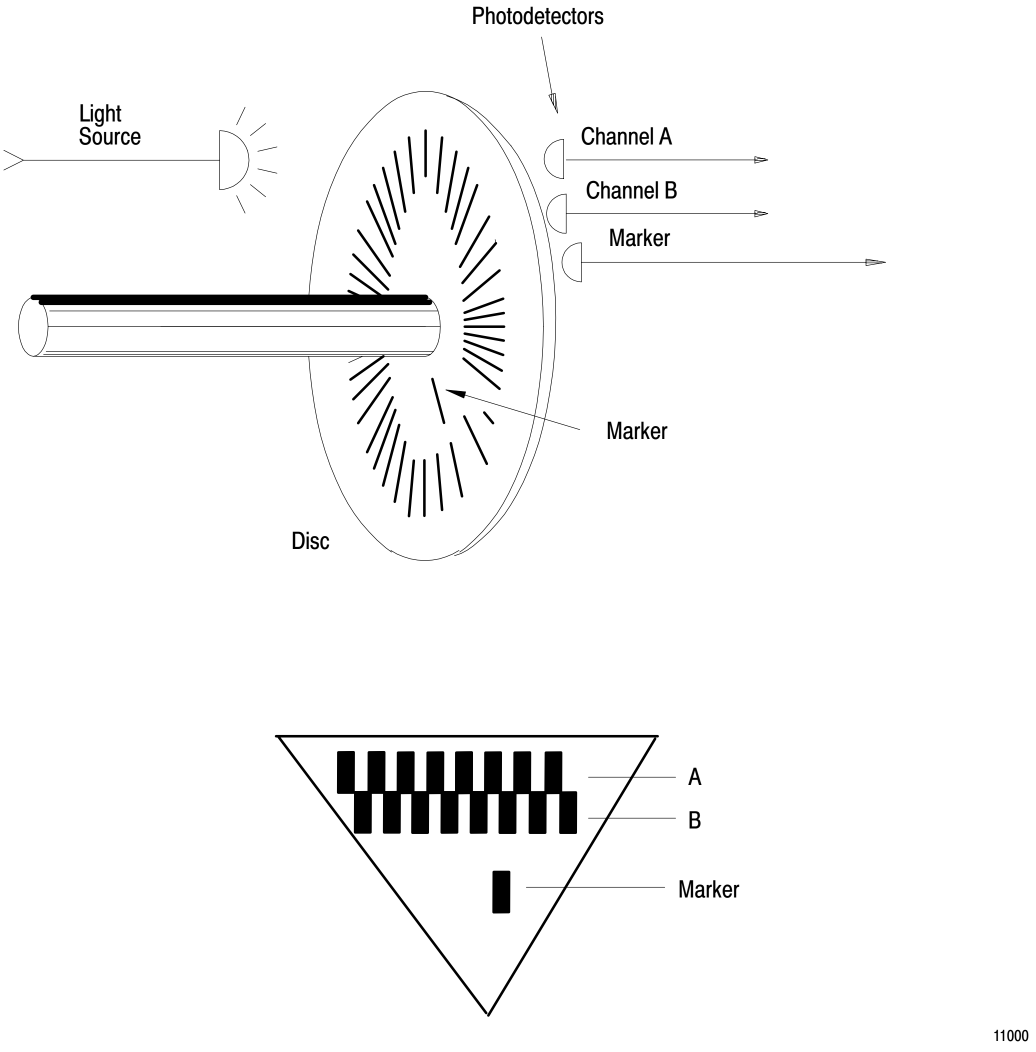
If the leadscrew has only one thread, the pitch is also equal to the lead, which is the distance the axis travels each revolution of the leadscrew. You can see from Figure 3.5 that the axis will travel 1/4 inch per revolution if the pitch is 1/4 inch. Since leadscrews normally have only one thread, and pitch is a more common term than lead, in this publication we use the term pitch to refer to the distance the axis travels for each revolution of the leadscrew.

Do not confuse leadscrew pitch with its inverse, which is the number of pitch (threads) per inch. In the example of Figure 3.5, the leadscrew has 4 pitch (threads) per inch. A leadscrew with a pitch of 1/4 inch is often described as being a 4-pitch (per inch) leadscrew.

Encoder Feedback

An incremental digital encoder provides feedback that indicates the magnitude and direction of any change of axis position. As shown in Figure 3.6, the encoder shaft is attached to a transparent disc marked with uniformly spaced lines. Strategically located photodiodes detect light. As the disc rotates, the lines break up the light reaching the photodiodes. As a result, the output (channel A, channel B, and marker) from each photodiode is a series of electrical pulses.

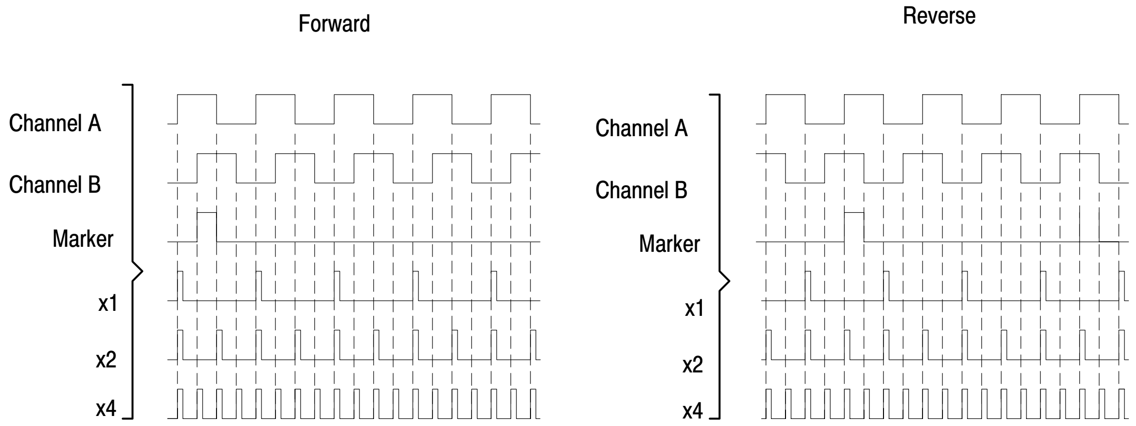
Figure 3.6
Incremental Encoder - Showing How Signals Are Generated



Channel Phase Relationship

The photodetectors are placed so that the channel A and channel B output signals are out of phase by 90° (Figure 3.7). The lead/lag relationship of these signals indicates the direction of axis motion. Also, the phase relationship of these signals allow the decoding circuit to count either 1, 2, or 4 feedback pulses for each line of the encoder (Figure 3.7). This provides flexibility in establishing feedback resolution.

Figure 3.7
Encoder Signals - Showing Phase Relationship



Note: For the servo positioning assembly, the encoder marker must be high when both channel A and channel B are high, or the marker is not recognized unless you set the marker logic jumper to the not-gated position.

11001

Feedback Resolution

The following discussion of feedback resolution assumes that you are using a leadscrew, and that the encoder is coupled directly to the leadscrew with no intermediate gearing. These assumptions apply to many applications. If your application differs, be sure to account for the differences.

Feedback resolution is the smallest axis movement the servo positioning system can detect. It is determined by:

- leadscrew pitch - axis displacement per revolution
- encoder lines - number of lines per revolution
- feedback multiplier - selected as x 1, x2, or x4

The following equation shows how these factors determine feedback resolution:

$$\text{feedback resolution} = \frac{\text{leadscrew pitch}}{\text{(encoder lines) (feedback multiplier)}}$$

You must select the leadscrew pitch, encoder lines, and feedback multiplier to provide desired feedback resolution and meet other requirements of your application.

The programming resolution of the servo positioning system is 0.0001 inch or 0.001 millimeter. If you select a feedback resolution coarser than that, round off your position commands so that the effective programming resolution is no finer than the feedback resolution you chose.

If you select a feedback resolution finer than the programming resolution, positioning can be smoother. However, the maximum axis speed is directly proportional to the feedback resolution. There is always a trade-off between feedback resolution and maximum axis speed. The maximum encoder input frequency for the servo positioning assembly is 250kHz. Therefore, to avoid a programming error, you must limit the axis speed to conform to this formula:

$$\text{programmed axis speed} < \frac{1.5 \text{ x } 10^{7}}{1.28} \text{ x feedback res x feedback mult}$$

The 1.28 factor allows for a 127% feedrate override value.

Each encoder line represents a fraction of a revolution of the leadscrew. For example, consider a 250 line encoder. Each line represents 1/250 of a revolution of the leadscrew.

Also, consider a 4-pitch (per inch) leadscrew for this example. The slide moves 1/4 inch for each revolution. With an x1 multiplier, each feedback increment represents 1/250 of 1/4 inch or 0.001 inch slide movement. This is the feedback resolution.

$$\begin{aligned}\text{feedback resolution} &= \frac{\text{0.25 in/rev}}{\text{250 lines/rev x 1 increment/line}}\\ &= \text{0.001 in/increment}\end{aligned}$$

Therefore, if we cause the leadscrew to move the slide 2 inches, we will get 2,000 feedback pulses.

Now, consider replacing the 250-line encoder with a 500-line encoder. By doubling the number of feedback pulses per revolution of the leadscrew, we improve the feedback resolution from 0.001 inch to 0.0005 inch.

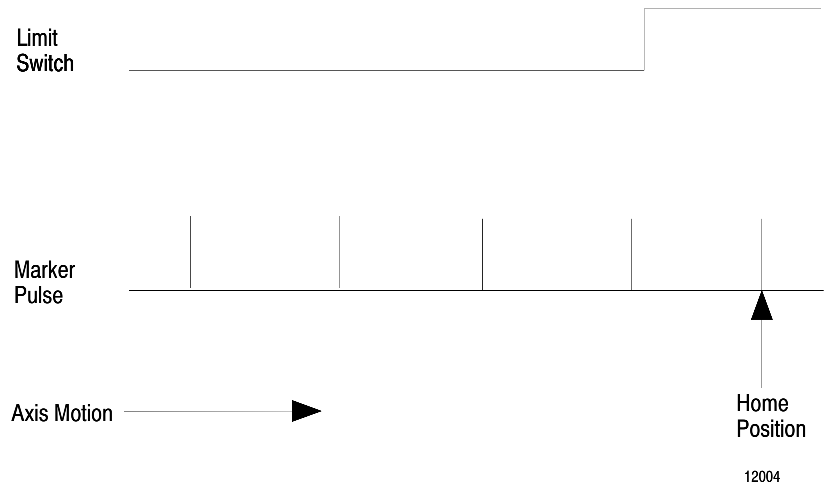
Another way to improve feedback resolution is to use a higher feedback multiplier. You can select a multiplier of x1, x2, or x4. For example, with the 4-pitch (per inch) leadscrew and the 250-line encoder, if you select an x2 multiplier you get the same feedback resolution improvement of from 0.001 inch to 0.0005 inch. With an x4 multiplier, you improve the feedback resolution to 0.00025 inch.

Marker

Besides the channel A and B output, an incremental encoder has a marker output (Figure 3.6 and Figure 3.7). The marker pulse occurs once every revolution. With a 4-pitch leadscrew, the marker pulse occurs at each 1/4 inch interval of slide travel.

We can use a market pulse to establish a home position somewhere along the slide travel. For example, we can place a limit switch near the end of the slide travel. The first market pulse after the limit switch is activated could then designate the home position (Figure 3.8).

Figure 3.8
Marker Pulse - Establishing a Home Position



Once we establish a home position, we can use it as an absolute reference point for all moves.

Summary

In this chapter we described concepts of closed-loop positioning. Now you are ready for concepts of position with an Allen-Bradley PC. This material is covered in chapter 4.

Chapter 4

Positioning With an Allen-Bradley Programmable Controller

Chapter Objectives

The previous chapter described concepts of closed-loop positioning. This chapter describes where the servo positioning assembly fits into a positioning system, and how the servo positioning assembly communicates with the PC processor.

Where the Servo Positioning Assembly Fits In

Figure 4.1 shows where the servo positioning assembly and a servo drive fit in the positioning system we described in the previous chapter. The servo drive contains the velocity loop summing point and amplifier. The servo positioning assembly contains the positioning loop summing point and the feed forward summing point. The servo positioning assembly sends the analog velocity command signal to the servo drive.

Figure 4.1
Where the Servo Positioning Assembly Fits in a Positioning System

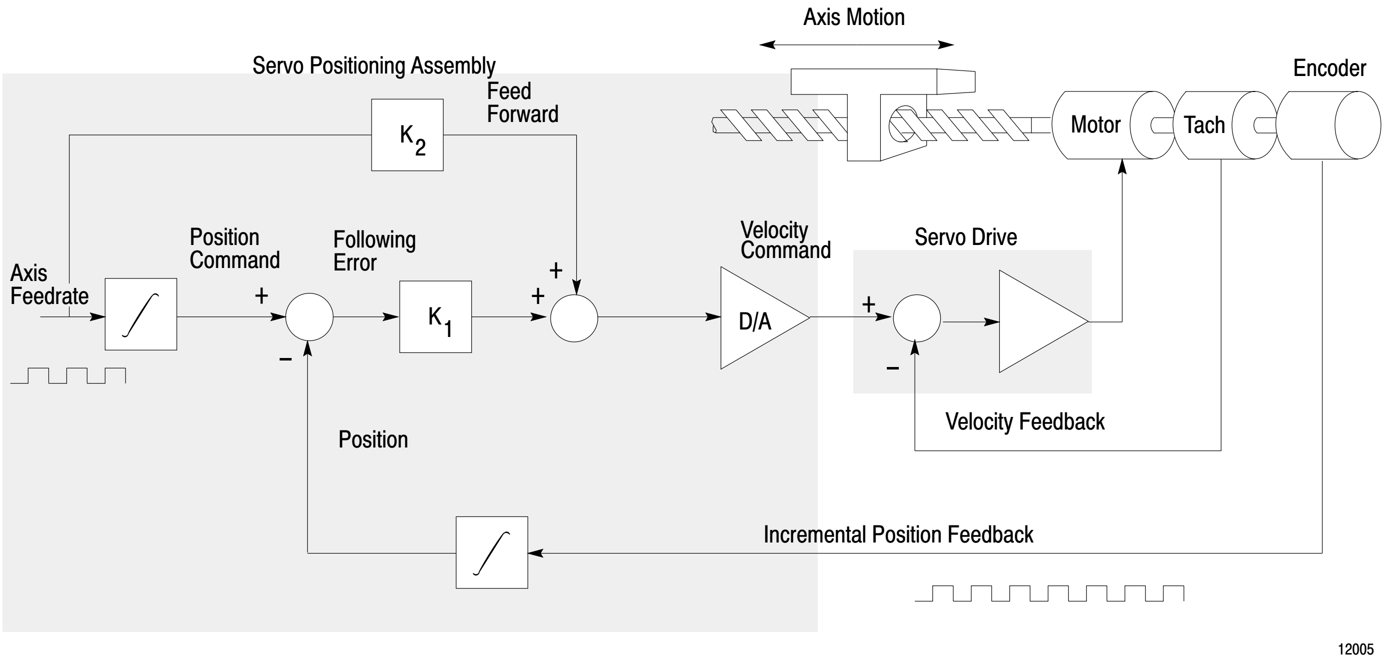
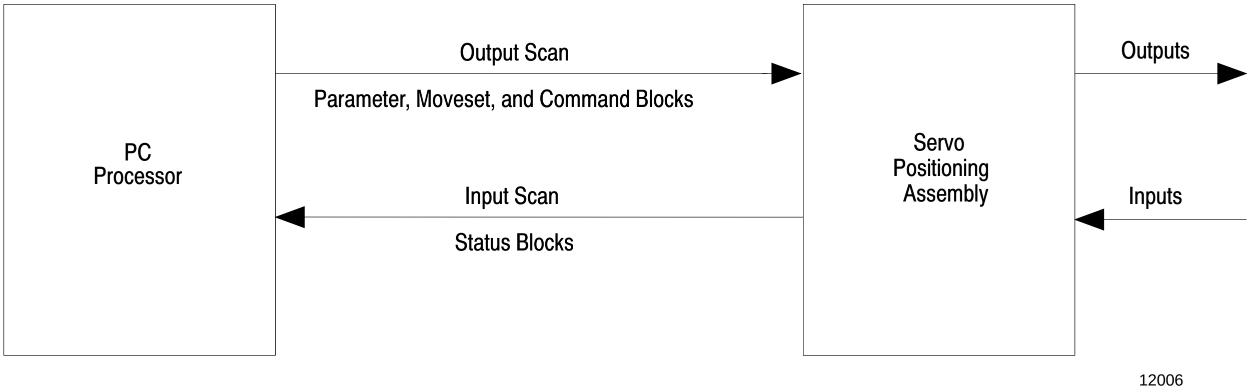


Figure 4.2 shows where the servo positioning assembly fits in a PC system. The PC processor constantly communicates with the servo

positioning assembly through the I/O scan. The PC processor acts on a block transfer read instruction to receive status blocks. Based on the status information received, the PC processor acts on a block transfer write instruction to send either parameter blocks, move blocks, or control blocks.

Figure 4.2
Where the Servo Positioning Assembly Fits in a PC System


PC Processor
Output Scan
Parameter, Moveset, and Command Blocks
Input Scan
Status Blocks
Servo Positioning Assembly
Outputs
Inputs
12006

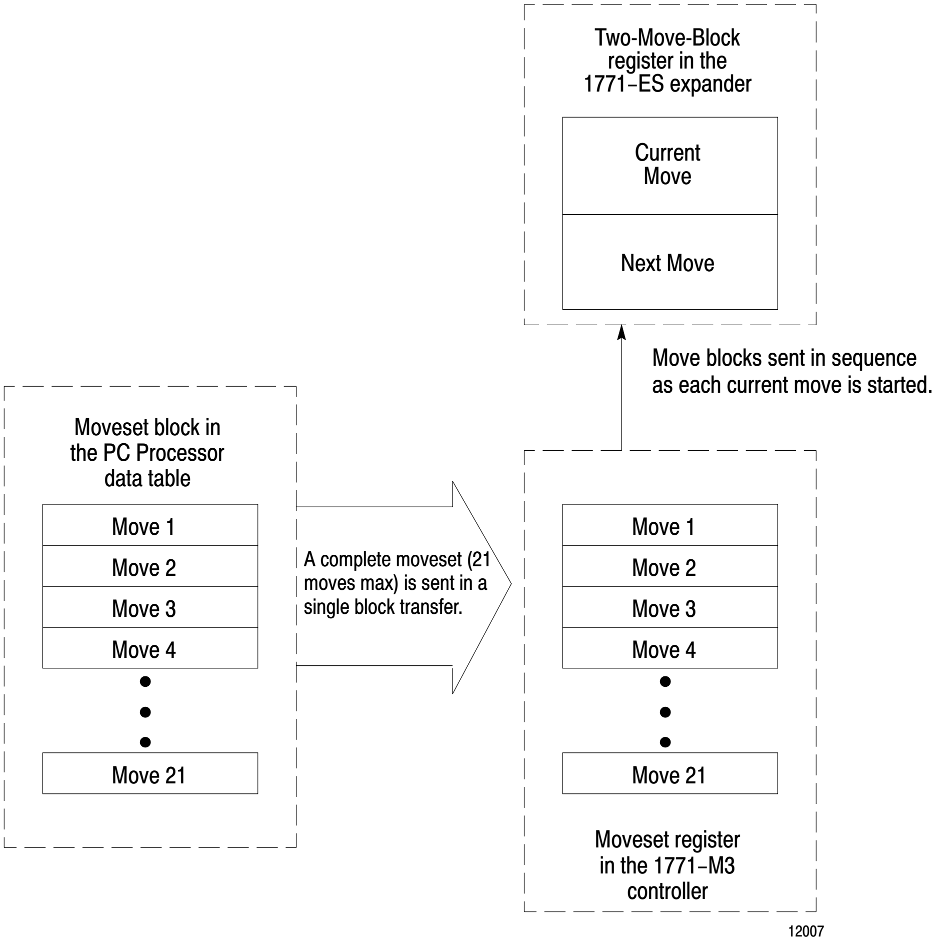
Independent of I/O Scan

Although the servo positioning assembly sends data to and receives data from the data table through the I/O scan, the positioning loop is closed on the 1771-ES expander (at the positioning loop summing point). This allows the 1771-ES expander to provide a servo sample period of 2.4ms, independent of I/O scan.

Move/Moveset

You must describe the axis motion you want in moveset blocks in the data table. You can enter a maximum of 21 separate move blocks in a moveset block (Figure 4.3).

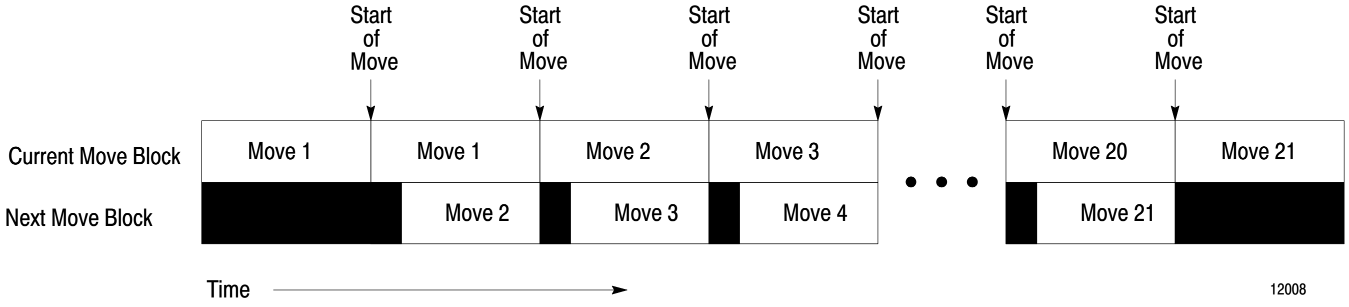
Figure 4.3
A Moveset Block is Sent to the 1771-M3 Controller That Sends the Move Blocks Sequentially to the 1771-ES Expander


Two-Move-Block register in the 1771–ES expander
Current Move
Next Move
Move blocks sent in sequence as each current move is started.
Moveset block in the PC Processor data table
Move 1
Move 2
Move 3
Move 4
Move 21
A complete moveset (21 moves max) is sent in a single block transfer.
Move 1
Move 2
Move 3
Move 4
Move 21
Moveset register in the 1771–M3 controller
12007

The PC processor sends a complete moveset block to the 1771-M3 controller in a single block transfer. The 1771-M3 controller can hold a moveset block for each of the three possible axes.

The 1771-ES expander can hold two move blocks, the current move block available for execution and the next move block. After the current move is completed and the next move is to be executed, the next move block becomes the current move block (Figure 4.4).

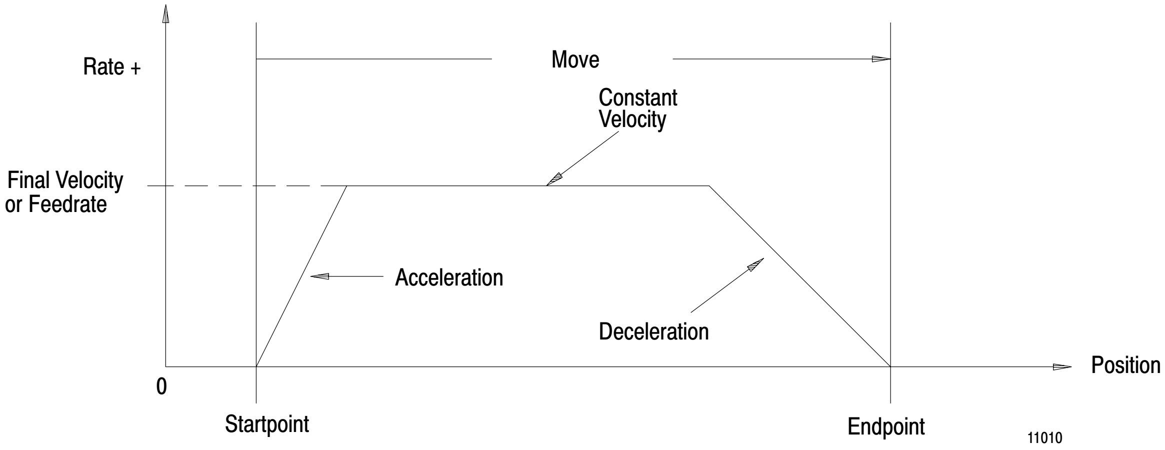
Figure 4.4
In the 1771-ES Expander, as Each Current Move is Completed, the Next Move Block is Ready to Take its Place



Initially, the 1771-M3 controller sends the first move block to the 1771-ES expander. Then, as each move is started the 1771-M3 controller sequentially sends each of the remaining move blocks to the 1771-ES expander.

A move block for a move to position defines motion of the axis from one position to another. Figure 4.5 shows the profile of an axis move. The horizontal axis in the figure represents axis position. The vertical axis represents axis velocity. Moves plotted above the position axis are in the positive direction (from left to right), moves plotted below the position axis are in the negative direction (right to left).

Figure 4.5
One-move Profile for an Axis



In the move shown in Figure 4.5, the axis:

- starts from a resting position
- accelerates to a final velocity

- moves at the final velocity some distance
- decelerates to zero velocity (at which time it has reached the programmed endpoint)

Move Values

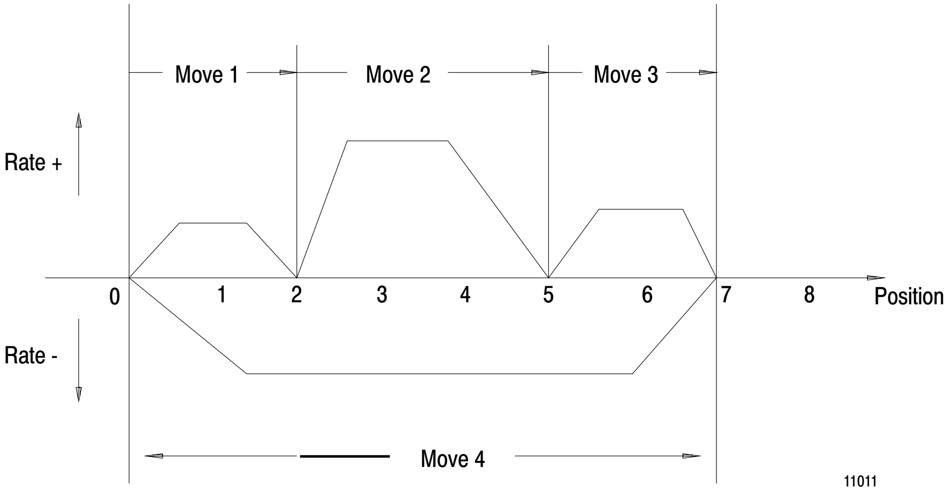
Each move block can specify several values. The servo positioning assembly executes the move based on these items you enter:

- endpoint
- acceleration
- final feedrate
- deceleration

When you select a deceleration value, the 1771-ES expander automatically calculates the point at which the deceleration must begin.

You can combine several single moves like that of Figure 4.5 to form a moveset. Figure 4.6 shows an example that consists of four moves. Move 1 starts at position coordinate 0 and ends at position coordinate 2. Move 2 continues axis motion to position coordinate 5. Move 3 continues to position coordinate 7. Move 4 then causes the axis to reverse direction and move back to position 0. The axis stops after it returns to its initial starting position. A drawing like that of Figure 4.6 is a moveset profile. You can use such profiles as an aid in programming axis motion.

Figure 4.6
Moveset Profile with All Single-step Moves



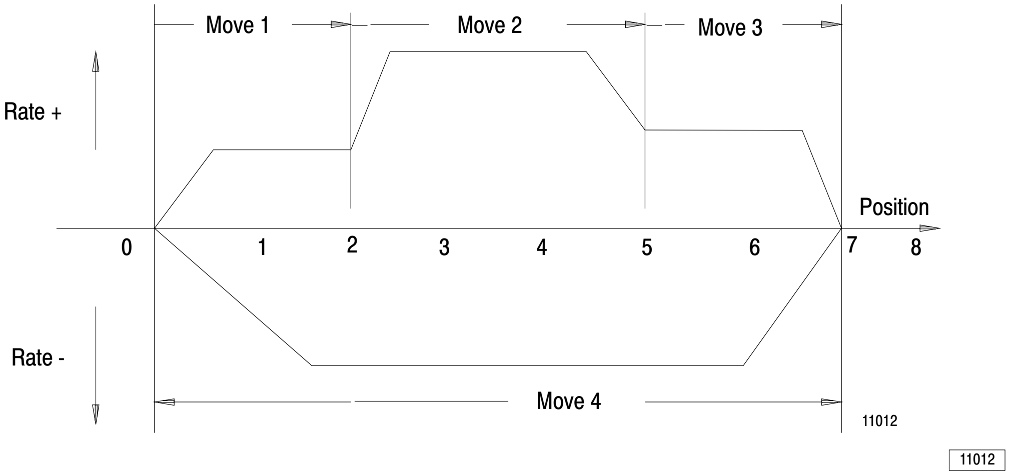
You can program multiple movesets for a given axis.

Move Selection

For each move, you have each of the following selections:

- **Absolute or incremental positioning** - In an absolute move, the endpoint value specifies a position coordinate relative to the current axis zero position. In an incremental move, the endpoint value specifies a position coordinate relative to the last programmed endpoint achieved by the axis.
- **Global or local values** - You enter a global final feedrate value and a global accel/decel rate value. These global rates apply to all moves except those for which you select to specify local rates. A local rate applies only to a single move.
- **Halt or run** - After completing a move for which you have selected halt, the 1771-ES expander will not execute the next move until it receives a begin or start command. After completing a move for which you have selected run, the 1771-ES expander will immediately execute the next move without waiting for a start command. With halt selected, the module executes a single-step move. With run selected, you can select moves to be either single-step moves or continuous moves.
- **Single-step or continuous** - When the 1771-ES expander executes a single-step move, it decelerates the axis to zero velocity at the programmed endpoint. When it executes a continuous move, it attempts to blend the move smoothly with the final feedrate of the next move (if the next move is in the same direction). The moves in Figure 4.6 are all programmed as single-step moves. Figure 4.7 shows the same moveset with all moves programmed as continuous. A moveset can contain a mix of single-step and continuous moves.

Figure 4.7
Moveset Profile with all Continuous Moves

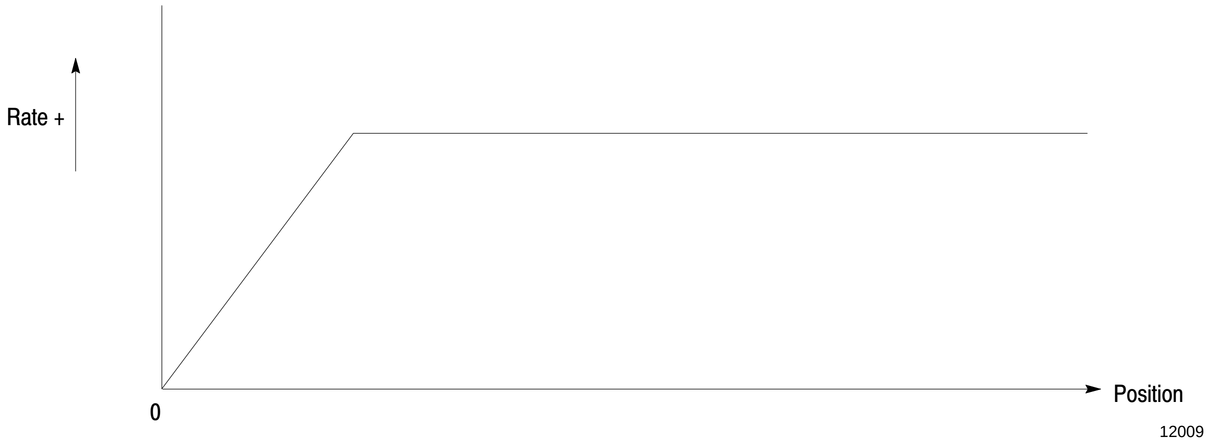


Move Alternatives

In place of a move to position, in any move block you can select one of the following:

- **Dwell** - Instead of an endpoint and rates, you can program a time in seconds in the move block. When the 1771-ES expander executes a dwell move block, it stops axis motion for the programmed amount of time.
- **Preset to Position** - You can program an axis position preset value in the command block. When the 1771-ES expander executes a preset to position, it sets its axis position register to the programmed preset value. No axis motion occurs.
- **Move to Position with Offset** - The parameter block contains an offset value. When the 1771-ES expander executes a move to position with offset, it adds this offset value to an offset accumulator. For every move, it adds the value stored in the accumulator to the programmed endpoint then executes the move.
- **Constant Velocity** - This command clears the position register to zero before moving the axis to the position you specify. By repeatedly generating continuous constant velocity moves, you can cause uninterrupted motion, which could, for example, be applied to a conveyor.(Figure 4.8).

Figure 4.8
Moveset Profile for Constant Velocity Moves



In Position

For a continuous move with the next move in the same direction, the move is complete when the axis feed is done. The 1771-ES expander immediately begins the feedrate for the next move without waiting for the following error to close.

For any halt move, single-step move, or a continuous move with the next move in the opposite direction, the move is not complete until the axis is in position. The axis is in position when the following conditions are met:

- the axis feed is done
- following error has closed to within the in-position band

You establish the in-position band in the parameter block. The in-position band is the largest distance from the endpoint at which you will allow the axis to be considered in position.

Synchronizing Axes

In many applications it is important to synchronize the motion of two or more axes. In the following sections, we will tell you how to do this.

Halt Moves

For halt moves, axis synchronization is straightforward. When an axis is in position after a move, the next axis move will not begin until you send a start command.

You can monitor the in-position signal of each axis through the status block. When all axes are in position, you can send a start command to each axis through the command block.

Alternatively, you can monitor the in-position signal of each axis through the hardware done output terminal of the 1771-ES expander. When all axes are in position, you can send a start command to each axis through the hardware start input terminal of the 1771-ES expander.

Using the hardware start and done signals is faster than using block transfer for the status and command blocks. Furthermore, if the axis synchronization includes multiple servo positioning assemblies, precise synchronization cannot occur through block transfer because two block transfers cannot occur simultaneously.

Continuous Moves

For continuous moves with the next move in the same direction, axis synchronization requires precise programming of feedrates, acceleration rates, and deceleration rates. You must program the move blocks so that each axis takes the same amount of time for corresponding moves. Furthermore, you must plan the moves to be long enough to adhere to the following constraints:

- Each move must take longer than the time it takes to transfer a move block from the 1771-M3 controller to the 1771-ES expander. This time is a function of the number of axes as follows:

No. of Axes	Time
1	20ms
2	25ms
3	30ms

- If the number of moves requires additional moveset blocks, the last two moves of each preceding moveset block must not be too short. They must take a long enough time for the following moveset block to be transferred from the data table. (Refer to chapter 8 for details about block transfer timing.)

Run-Single-Step Moves

For run-single-step moves, axis synchronization is dependent upon the axis response on each move. The same is true for continuous moves with the next move in the opposite direction.

In both cases, the 1771-ES expander executes the next move automatically as soon as the current move is done, without waiting for a start signal. However, the time it takes for each move cannot be precisely calculated because the following error has to close before the move is done.

Auto Position Correction

The auto position correction feature may prevent an accumulation of position error caused by occasional noise on the channel A and B inputs. However, if the environment is excessively noisy, or if the cabling and shielding is not proper, this feature causes the axis to jump or jerk. This jump or jerk should indicate to you that a problem exists.

You enter the number of lines on the encoder and the feedback multiplier into the parameter block. From this, the 1771-ES expander knows how many feedback pulses it should receive each encoder revolution. The module also receives a marker pulse each revolution.

Each time the 1771-ES expander receives a marker pulse, it checks the value in the position register to see if it is an even multiple of the number of feedback pulses per revolution. If the value is off, the 1771-ES expander will automatically adjust it.

This feature corrects position errors caused by noise on the channel A and B encoder feedback signals. However, the function of this feature assumes a noise-free marker signal.

The marker signal does have some noise protection because the 1771-ES expander only accepts a marker signal when the channel A and B signals are high (unless you set the marker logic jumper to the not-gated position).

Specifying Axis Position

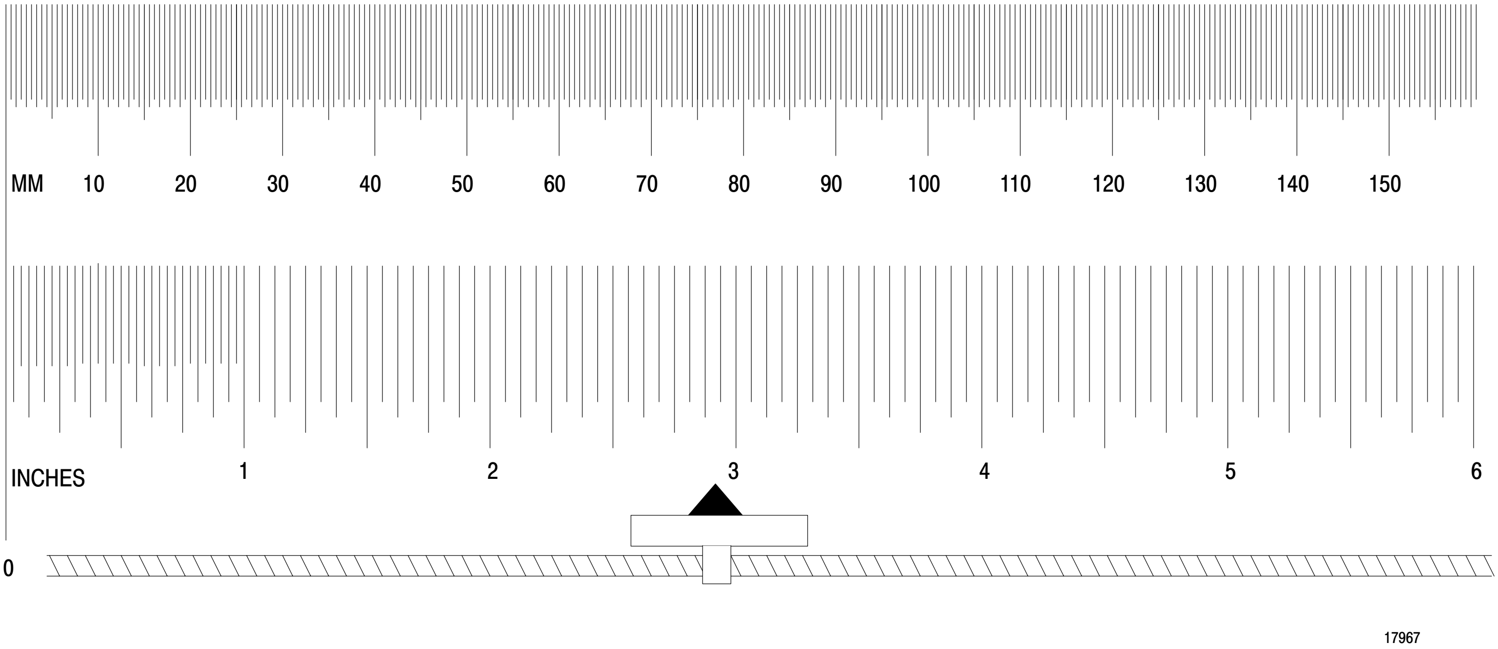
To command axis motion, you must be able to specify axis position by establishing an axis position scale, or coordinate system, for each axis.

Figure 4.9 shows an example of an axis and its position scale. Any axis position within the range of travel can be identified by a number. For the servo positioning assembly, the axis position scale can be either in inches or millimeters.

The position scale is an internal scale used by the servo positioning assembly to identify axis position. It is not printed on the axis slide. You can shift the axis position scale by entering (through the command block) any of the following commands:

- search home
- preset
- initialize home

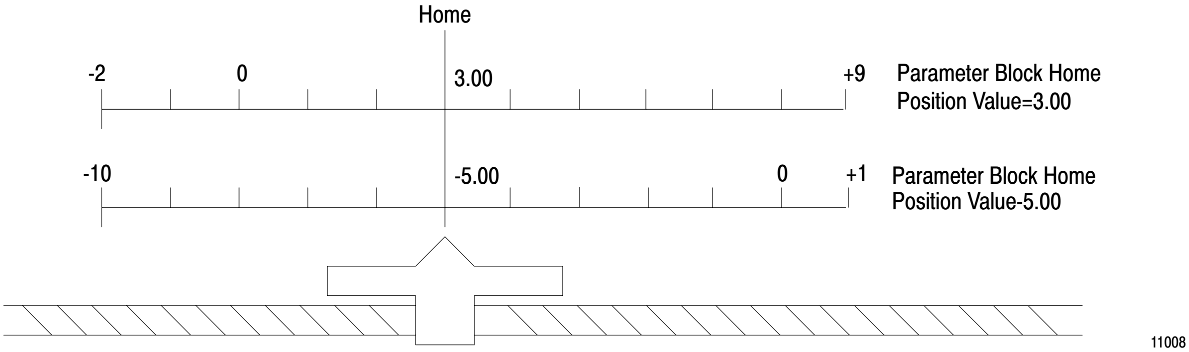
Figure 4.9
Axis Position Scale



Search Home

Because the position feedback is incremental rather than absolute, the servo positioning assembly does not know the axis position when it first receives power. You must command a search home (through the command block) each time after powering up. In the search home operation, the axis moves until the servo positioning assembly detects the first encoder marker beyond the user-installed home limit switch. The axis stops on the marker. The servo positioning assembly then sets it position register to the home position value you specify in the parameter block. This initializes the axis position scale. Figure 4.10 shows how the home position value you specify in the parameter block can affect the axis position scale. This figure compares the scales for an axis after search home operations with different home position values form the parameter block representing the same physical position.

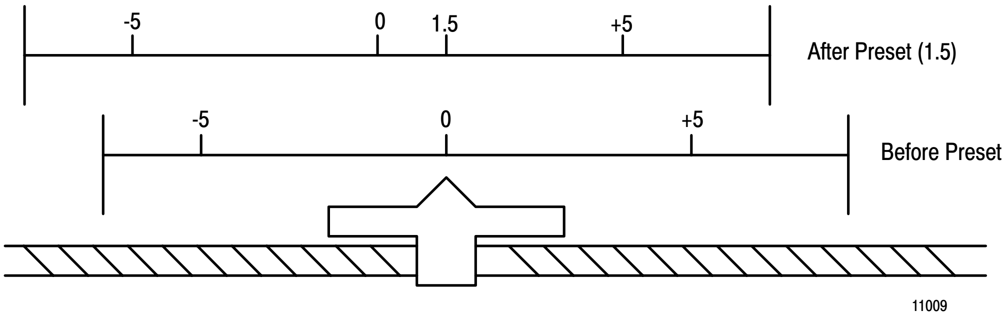
Figure 4.10
Axis Position Scales for 2 Home Position Values


Home
-2
0
3.00
+9
Parameter Block Home Position Value=3.00
-10
-5.00
0
+1
Parameter Block Home Position Value-5.00
11008

Preset

Through a command block, you can command the servo positioning assembly to preset a specified value into its position register. When the servo positioning assembly executes a preset command, it sets its position register to the specified value without causing axis motion. This action effectively shifts the axis position scale. Figure 4.11 shows an axis position scale before and after a preset operation.

Figure 4.11
Axis Position Scale before and after Preset



Initialize Home

Through a command block you can generate an initialize home command. The initialize home operation assigns the home position value (which you specify in the parameter block) to the current axis position. Its effect is the same as that of the preset operation, except that the new position value is the home position value.

Summary

Now that you have been familiarized with the general concepts of how the servo positioning assembly functions in a closed-loop positioning system and in a PC system, you are ready for specific details of the servo positioning assembly in chapter 5.

Chapter 5

Hardware Description

Chapter Objectives

The previous chapter described how the servo positioning assembly fits into a positioning system as part of a programmable controller. This chapter describes specific hardware of the servo positioning assembly and lists its specifications. This chapter also describes other hardware items you need for a positioning system.

Indicators

There are three indicators on the 1771-M3 controller. With the PC processor operating in the run mode, the indicators have the following functions:

- Processor Communication Fault - This red indicator turns on when the module detects a fault in the communication between it and the PC processor. The I/O adapter module or PC processor will not detect this as a fault.
- Expander Communication Fault - This red indicator turns on when the module detects a fault in the communication between it and a 1771-ES expander.
- Active - This green indicator is normally on. It turns off when a hardware fault is detected on a 1771-ES expander. it blinks if you have not properly configured the modules.

There are six indicators on the 1771-ES expander. With the PC processor operating in the run mode, the indicators have the following functions:

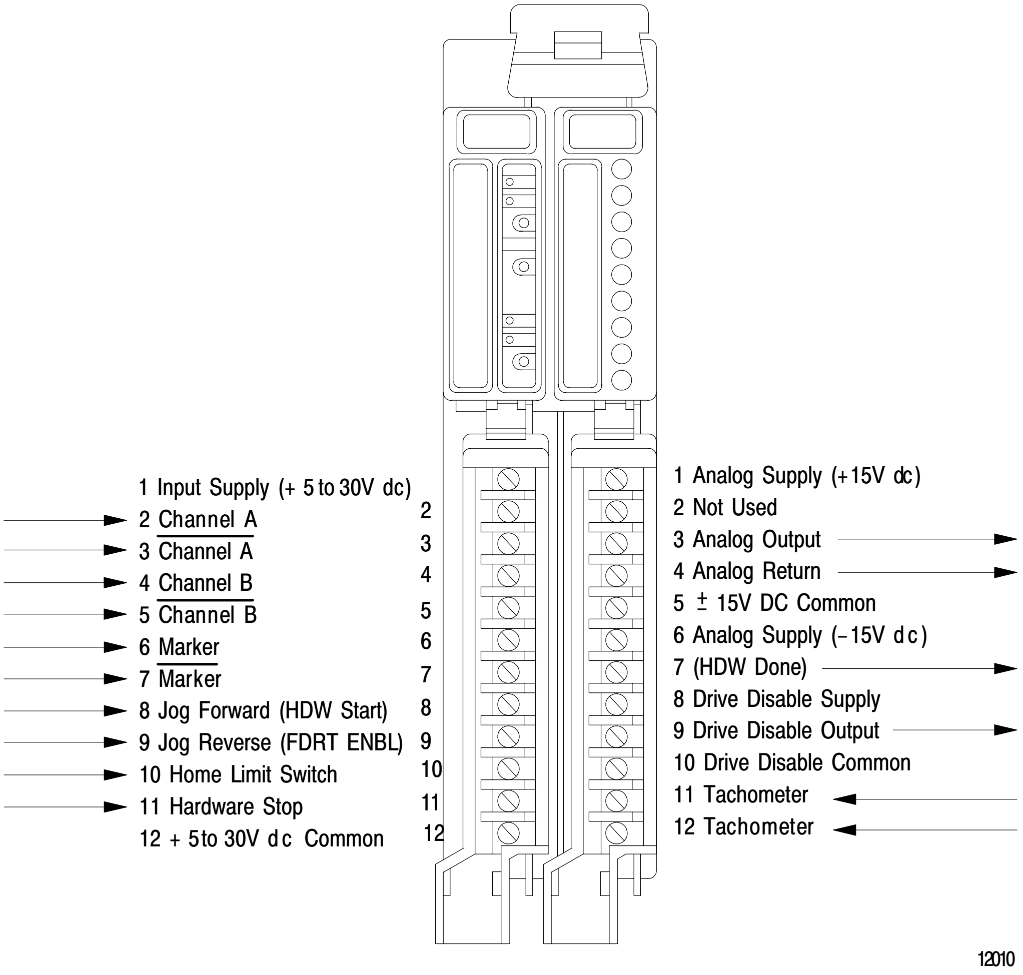
- Module Active - This green indicator is on when the module is operating normally.
- Marker - This green indicator is on when the channel A, channel B, and marker signals are true simultaneously.
- Home - This green indicator is on when the axis is in the home position.
- Tach Calibrate - This green indicator is used in setting the adjustments for loss of feedback detection.
- Hardware Stop - This red indicator goes on when the hardware stop input opens. It stays on until the input closes and the servo expander module is reset.
- Diagnostic - This red indicator goes on when a fault is detected at the servo expander module.

These indicators are useful troubleshooting aids, described fully in chapter 9.

Inputs/Outputs

The 1771-M3 controller requires no connections. You will make all wiring connections to the 1771-ES expander. Figure 5.1 shows the terminals on the 1771-ES expander. These terminals provide the connection points for all the inputs and outputs of the servo positioning assembly. Limit the cable length to 50 feet for all connections.

Figure 5.1
Terminals On the 1771-ES Expander Showing Input and Output Signals



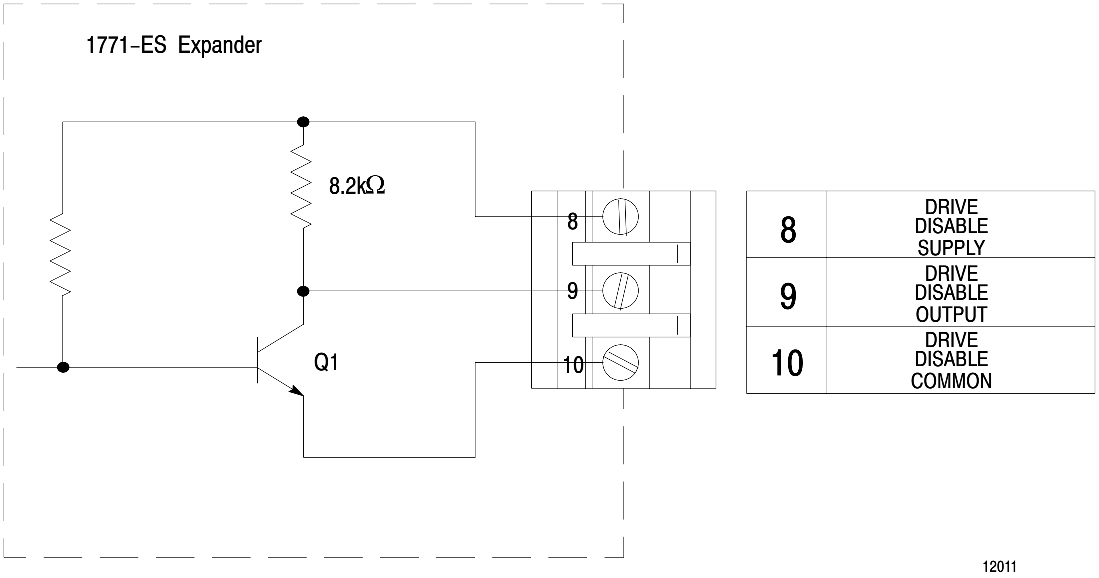
Outputs to Servo Drive

Terminals 3 and 4 on the right wiring arm provide connection points for the velocity command signal to the serve drive. This analog output is a +10V dc differential signal.

Terminals 8, 9, and 10 on the right wiring arm provide connection points for a drive disable signal (Figure 5.2). In chapter 6 we will show you how to connect this output to either source or sink 100mA maximum to enable the drive. The module normally provides current thru this transistor to enable the drive. However, the module will turn off the current to disable the drive if:

- the hardware stop input goes high
- a command block commands an immediate stop
- a firm ware or hardware watchdog timers times out
- the 1771-ES expander detects excess following error, a loss of feedback, or a power supply loss

Figure 5.2
Schematic Diagram of the Drive-disable Output Circuit



The 1772-ES expander is compatible with a wide variety of servo drives, including Allen-Bradley Bulletin dc Servo Controllers (refer to publication 1388 -5.0). Allen Bradley also offers Bulletin 1326 dc servo Motors to match the Bulletin 1388 dc Servo Controllers.

Tachometer Input

Terminals 11 and 12 on the right wiring arm provide connection points for the velocity feedback signal from the tachometer. Although the velocity loop is closed on the servo drive, the 1771-ES expander uses the velocity feedback signal to compare to the position feedback signal from the encoder. If the module detects an imbalance between these signals, it disables the servo drive and sends a loss of feedback signal through the status block.

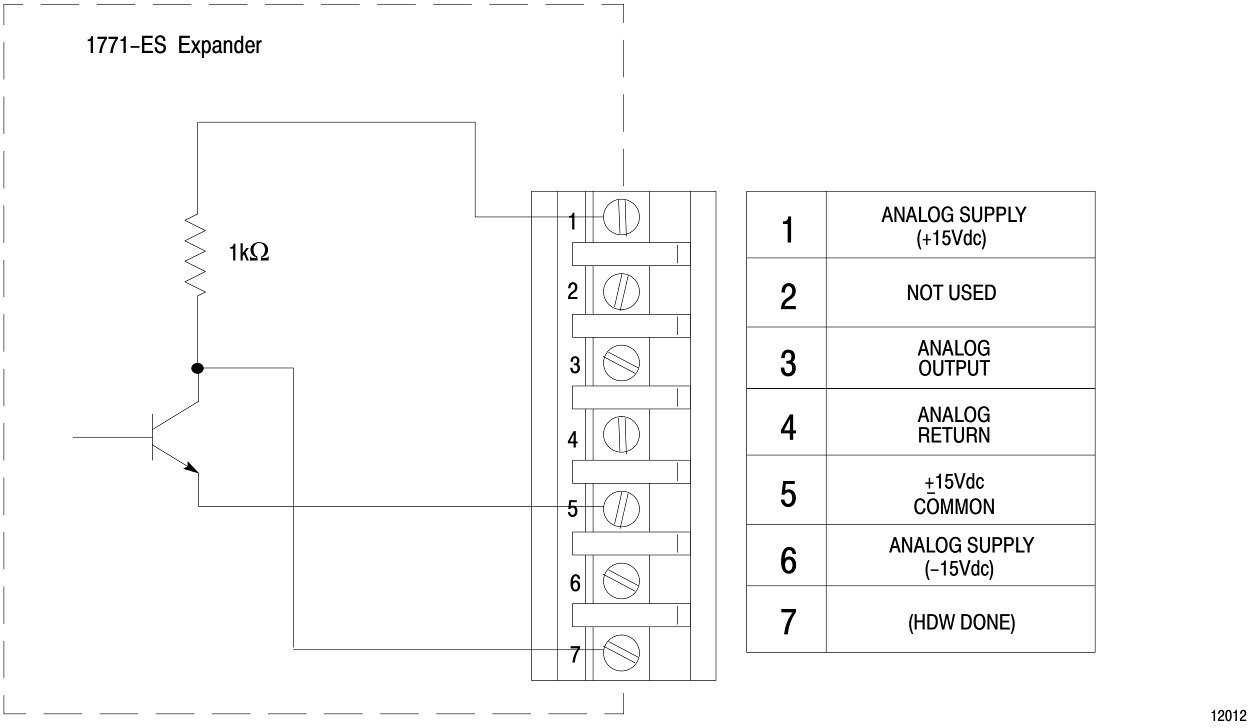
The 1771-ES expander accepts a full scale tachometer signal of 3V to 50V dc. If the full scale tachometer signal is greater than 50V dc, you must reduce it through a voltage divider on the servo drive before connecting it to the module.

⚠ **CAUTION:** Do not connect a signal greater than 50V dc across these terminals. A signal greater than 50V dc could damage the 1771-ES expander.

Hardware Done Output

Terminal 7 on the right wiring arm provides a connection point for a hardware done output signal (Figure 5.3).

Figure 5.3
Schematic Diagram of the Hardware-done Output Cirucit



1	ANALOG SUPPLY (+15Vdc)
2	NOT USED
3	ANALOG OUTPUT
4	ANALOG RETURN
5	±15Vdc COMMON
6	ANALOG SUPPLY (–15Vdc)
7	(HDW DONE)

The output transistor, normally on, provides a 15mA (maximum) sink. When the axis feed is done and the axis is in position, the transistor is off and the circuit provides +15V dc through a 1k resistor. This provides you with a hardware done signal that is high-true.

In chapter 6, we will show you how to connect the hardware done signal to a dc (12-24V) Input Module (cat. no. 1771-IB) for axis synchronization of halt moves.

Discrete Inputs

Terminals 8, 9, 10, and 11 on the left wiring arm provide connection points for discrete input signals. The module accepts a discrete input signal as being high when it reaches 40% of the input power supply voltage. The module accepts a discrete input signal as being low when it reaches 20% of the input power supply voltage.

Each discrete input has an internal pull-up resistor. In chapter 6, we will show you how to select an internal pull-up resistor of 1.2k or 11.2k. You select each input individually through a switch setting.

For a high signal, the input device you connect to a discrete input does not have to source current. For a low signal, the input device you connect to a discrete input has to sink current through the pull-up resistor.

Hardware Start

In the auto mode, the module accepts a high-to-low transition at terminal 8 of the left wiring arm as a low-true hardware start input signal.

After completing a halt move, the 1771-ES expander will not execute the next move until it receives a start command. The start command could come through block transfer of a control block or through the hardware start signal.

Feedrate Override Enable

In the auto mode, the module accepts a high-to-low transition at terminal 9 of the left wiring arm as a **low-true** feedrate override enable signal.

After setting a feedrate override value for the axis through the command block and enabling external synchronization of feedrate override through the parameter block, you can enable the feedrate override through this input. Do this by setting bit 16 of word 17 in the parameter block ON (Axis 1). (Set bit 16 of words 36 and 55 for axis 2 and 3, respectively.). This allows you to activate a preloaded feedrate override value to change speed on several axes at the same instant.

Jog Forward

In the manual mode, the module accepts the signal at terminal 8 of the left wiring arm as a **low-true** jog forward signal. When the module receives this signal, it moves the axis in the positive direction at the rate established through block transfer.

Jog Reverse

In the manual mode, the module accepts the signal at terminal 9 of the left wiring arm as a **low-true** jog reverse signal. When the module receives this signal, it moves the axis in the negative direction at the rate established through block transfer.

Home

The module accepts the signal at terminal 10 of the left wiring arm as a **low-true** home signal. The module considers the first marker pulse after the home signal as the home position.

Hardware Stop

The module accepts the signal at terminal 11 of the left wiring arm as a **high-true** hardware stop signal. Unless this input is pulled low, the module holds the velocity command output signal at zero and disables the servo drive by turning off the drive disable circuit.

Encoder Inputs

Terminals 2, 3, 4, 5, 6, and 7 on the left wiring arm provide connection points for input signals from the encoder. Through jumpers on the module, you can select each channel individually for either single-ended or differential, and for either high-true of low-true input signals.

If you use a single-ended encoder, limit the input pulse rate to 20k Hz. If you use a differential encoder, limit the input pulse rate to 250k Hz.

The 1771-ES expander is compatible with Allen-Bradley Incremental Differential Line Driver Encoders (cat. no. 845N-SJDN4-C) and with other encoders having current-sinking (5-30V dc) line-driver outputs, totem-pole (TTL) outputs, or open-collector outputs.

External Power Supplies

You must provide at least two external dc power supplies to provide power for the input and output circuits.

Input Supply

You must connect a 5-30V dc power supply between terminals 1 and 12 of the left wiring arm. This provides power for the input circuits. The input circuits require 500mA (maximum) at 30V. You can use the same power supply to power the encoder if the power supply has enough additional current capacity for the encoder.

Drive Disable Supply

Unless the servo drive provides its own dc voltage source for this circuit, you'll need a 5 - 30V dc power supply to provide 100mA (maximum) for the drive disable circuit. How you connect this power supply depends on whether the servo drive requires a current source or a current sink to enable it.

Analog Supply

A separate +15V dc supply is needed to provide 200mA (maximum) for the digital/analog converter (DAC) to generate the analog output signal and for the hardware done output circuit.

Compatible Processors

The servo positioning assembly can be used with PC processors that have block transfer capability and adequate data table size to contain the data blocks you need for your application. Compatible PC processors include:

- Mini-PLC-2/05 (cat. no. 1772-LS,-LSP)
- Mini-PLC-2/15 (cat. no. 1772-LV)
- PLC-2/20 (cat. no. 1772-LP2)
- PLC-2/30 (cat. no. 1772-LP3)
- PLC-3 (cat. no 1775-L1,-L2)

Fault Responses

The servo positioning assembly provides a means for detecting and responding to faults in your servo positioning system.

Since the servo positioning assembly is part of a PC system, diagnostic information about fault conditions detected by the servo positioning assembly can be block transferred to the PC processor.

At the PC processor, you can use the ladder diagram program to respond to diagnostic information about fault conditions in any way you feel is appropriate for your application. This may include turning off machinery, turning on alarms, or generating report printouts. Furthermore, with an Allen-Bradley Data Highway network, you can send this diagnostic information to a computer or other Allen-Bradley PC processors.

The servo positioning assembly provides specific fault responses if certain critical connections are broken.

Loss of Feedback

The 1771-ES expander continuously monitors the tachometer and encoder feedback. If it senses an imbalance between these signals, it holds the velocity command output signal at zero and disables the servo drive through the drive disable circuit. Therefore, if the cable from either the encoder or the tachometer breaks, the 1771-ES expander will disable the servo drive.

Hardware Stop

You must connect a set of normally open contacts of your master control relay between the hardware stop input terminal and the input power supply common terminal. Normally, the master control relay would be energized, pulling the hardware stop input low. This allows the module to enable the servo drive.

However, if the master control relay de-energizes for any reason (such as extreme overtravel limit or emergency stop), the hardware stop input goes high. This forces the module to hold the velocity command output signal at zero and disable the servo drive by turning off the drive disable circuit. Therefore, if a connection in the hardware stop circuit breaks, the 1771-ES expander will disable the servo drive.

Loss of Power

The 1771-ES expander holds the velocity command output signal at zero and disables the servo drive by turning off the drive disable circuit if it is unable to sense the specified voltage as the following power-supply terminals:

- positive (+) terminal for the input power supply
- common (-) terminal for the input power supply
- positive (+) terminal for the analog power supply
- negative (-) terminal for the analog power supply

Therefore, if one of these power supplies connected to the 1771-ES expander terminal fails or if one of these connections from these power supply breaks, the 1771-ES expander will disable the servo drive.

The drive disable circuit normally provides current to a sensing circuit on the servo drive to enable it. However, if the 1771-ES expander detects a fault, it cuts off the current in the drive disable circuit, thereby disabling the servo drive. Therefore, if a connection in the drive disable circuit breaks, this disconnection will disable the servo drive.

Auto Position Correction

Each time the 1771-ES expander receives a marker pulse, it checks the value in the position register to see if it is an even multiple of the number of feedback increments per revolution. If the value is off, the 1771-ES expander will automatically adjust it to the closest even multiple.

This auto position correction feature corrects position errors caused by noise on the channel A and B encoder feedback signals. However, the function of this feature assumes a noise-free marker signal. Although this feature may be able to prevent an accumulation of position error caused by occasional noise on the channel A and B inputs, it cannot maintain position accuracy if the environment is excessively noisy or if the cabling and shielding is not proper.

If the environment is excessively noisy or if the cabling and shielding is not proper, this feature will cause the axis to jump or jerk. This jump or jerk indicates a problem.

Note that when the module detects a position error, it does not necessarily disable the servo drive.

Because this feature adjusts the position register to the closest even multiple of the number of feedback increments per revolution, it is essential that the axis move less than half an encoder revolution per servo sample period (2.4ms). Therefore, to avoid a programming error, you must limit the axis speed to conform to this formula:

$$\text{programmed axis speed} < \frac{12{,}500}{1.28} \text{ x FR x FM x EL}$$

Where:

FR = feedback resolution
FM = feedback multiplier (1, 2, or 4)
EL = encoder lines per revolution

Specifications

Here is a list of specifications for the servo positioning assembly.

Servo Output Voltage

- ±10V dc maximum (isolated)

D/A Converter (DAC)

- Signed 12 bit resolution

Encoder Input

- High: 1.6V
- Low: 1.0V sinking lmA

Encoder Input Rate

- Differential: 250k Hz maximum
- Single-ended: 20k Hz maximum
- Jumper selection of differential or single ended input

Encoder Multiplier

- x1, x2, or x 4,programmable

Tachometer Input (For loss-of-feedback detection)

- Full scale voltage: 3V dc minimum, 50V dc maximum
- Input impedance: 20k ohmss

Discrete Inputs

- Resistance to high side of supply 11.2k ohms or 1.2k ohms, switch selectable for each input
- For a low, required sink current with 1.2k ohms resistance: 4mA @ 5V, 24mA @ 30V
- For a low, required sink current with 11.2k ohms resistance: 0.4mA @ 5V, 2.7mA @ 30V
- High: 40% of + dc supply voltage
- low: 20% of + dc supply voltage

Hardware Done Output

- On: +15V source thru 1k ohms resistance
- Off: 15mA sink

Drive Disable Output

- Current: 100mA maximum, source or sink
- Voltage: 30V dc maximum to 5V dc minimum

Backplane Current

- 1771-M3 controller: 1.75A
- 1771-ES expander: 1.70A

External Power Supply Requirements

- External supply for inputs, +4.75 dc minimum, +30V dc maximum, 500mA maximum
- External supply for DAC and hardware done output, +15V dc, 200mA maximum
- External supply for drive disable output, +4.75V dc minimum, +30V dc maximum, 100mA maximum

Maximum Programmable Position

- ±999.9999 inches (resolution 0.0001 inch)
- ±19999.999 millimeters (resolution 0.001 mm)

Programmable Speed

- 0.0001-9990.0000 ipm (resolution 0.0001 ipm)
- 0.001-199900.000 mmpm (resolution 0.001 mmpm)

Accel/Decel

- 9999 ipm/s maximum (resolution 1 ipm/s)
- 99.99 mpm/s maximum (resolution 0.01 mpm/s)

Initial Servo Gain (Programmable)

- 0.01-9.99 ipm/mil following error (1 mil = .001 inch)
- 0.01-9.99 mmpm/mil following error (1 mil x .001 mm)

Servo Sample Period

- 2.4ms

Environmental Conditions

- Operational Temperature: 0° to 60°C (32° to 140°F)
- Storage Temperature: -40° to 85°C (-40° to 185°F)
- Relative Humidity: 5% to 95% (without condensation)

Keying

- Servo controller slot: between 2 and 4, 8 and 10
- Left servo expander slot: between 2 and 4, 14 and 16
- Right servo expander slot: between 4 and 6, 32 and 34

Summary

Now that you have read about the function of each input and each output, you are ready to install the servo positioning assembly. Chapter 6 gives you this information.

Chapter 6

Installing the Assembly

Chapter Objectives

The previous chapter described the hardware of the servo positioning assembly. This chapter tells you how to install the servo positioning assembly. As you install it, you will make hardware selections to direct its operation to fit your application requirements.

Configuring the Modules

The first step of installing a servo positioning assembly is to plan how to configure modules in the I/O chassis.

Planning Module Combinations

You can install one 1771-M3 controller in an I/O chassis together with either one, two, or three 1771-ES expanders. However, the I/O chassis must not contain any other module combination of a master (such as an analog module) and its slave (expander).

A master must communicate with its slaves through the backplane. Two masters trying to communicate through the backplane interferes with each other.

If you have an illegal combination of 1771-ES expanders or a second master/slave combination in the I/O chassis, the **active** indicator on the 1771-M3 controller blinks. An illegal combination of 1771-ES expanders would be:

- the number of 1771-ES expanders not matching the number of axes in the parameter block
- an axis 2 with no axis 1
- an axis 3 with no axis 2
- two axes with the same number

Always use the same series level of 1771-M3 controller and 1771-ES expander. You cannot use a series A 1771-M3 controller with a series B 1771-ES expander. Likewise, you cannot use a series B 1771-M3 controller with a series A 1771-ES expander.

Avoiding Backplane Power Supply Overload

For each module you plan to install in the I/O chassis, add up it current load on the backplane power supply. Be sure that this total current is not so large as to overload the backplane power supply.

The backplane power supply current load of the servo positioning assembly is:

1771-M3 controller	1771-ES expanders	Total Current
1	1	3.45A
1	2	5.15A
1	3	6.85A

Note that if you add the total current draw of one 1771-M3 controller, three 1771-ES expanders, and either an I/O adapter or mini-processor module, the total would exceed 8A. In that case you could not use a 1771-P1 or 1771-P2 power supply because they are rated at 6.5A.

If the total current exceeds 6.5A, you can use Power-Supply Modules (cat. no. 1771-P3, -P4, -P5) to provide 8A, 11A or 16A. The following table lists the number of axes you can control with a servo positioning system in a 1771-A4 I/O chassis, based on power requirements and compatibility of other components used with the 1771-A4 I/O chassis.

Power Supply Cat. No.	I/O Adapter or Mini-Processor Module Cat. No.				
	1771-AL	1771-AS	1772-LS	1772-LSP	1771-LV
1771-P1					1 Axis
1771-P2	2 Axes	2 Axes			
1771-P3	2 Axes			1 Axis	
1771-P4	3 Axes		3 Axes	3 Axes	
1771-P4 plus 1771-P3 or a second 1771-P4	3 Axes		3 Axes		

Planning Module Location

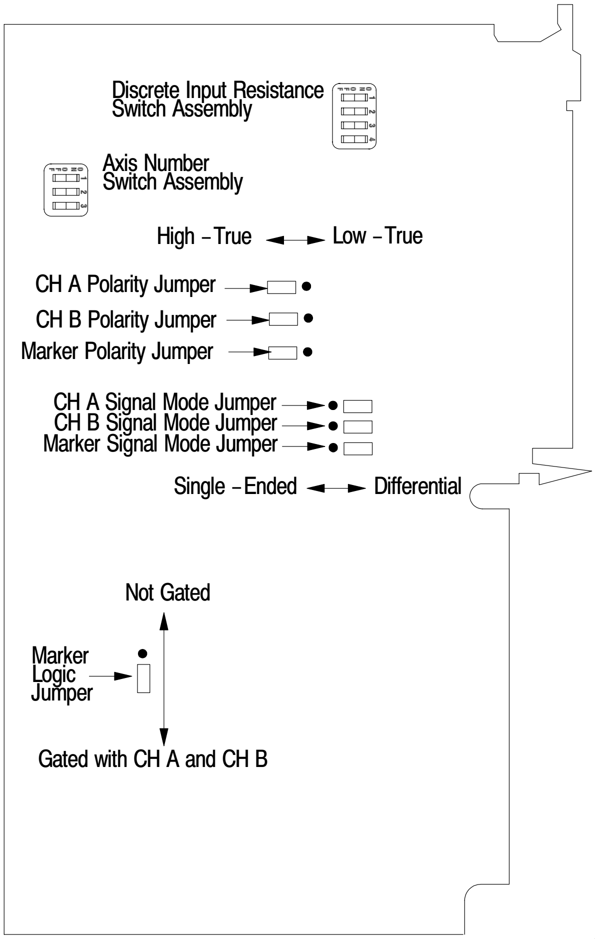
The 1771-M3 controller requires one I/O chassis slot. You can install it in any I/O in the I/O chassis. The 1771-M3 controller uses both the output image table byte and the input image table byte that correspond to its location address.

The 1771-ES expander requires two slots. Install it in a pair of slots that make up an I/O module group.

Setting Switches and Jumpers

Through switches and jumpers on the 1771-ES expander, you can select various aspects of the module's operation. To access these switches and jumpers, lay the 1771-ES expander on its right side and remove the left cover. Locate the switches and jumpers through Figure 6.1.

Figure 6.1
1771-ES Expander Switches and Jumpers



12013

This publication shows and describes switches as being on or off. Printed on the actual switch assemblies are the words ON and OFF or the word OPEN. OPEN corresponds to OFF.

Use a blunt-pointed instrument such as a ball-point pen to set these switches. **Never** use a pencil; graphite could jam the switch.

Figure 6.2 shows details of a jumper connecting two pins. Each jumper connects two of a set of three pins. To change a jumper setting, follow these steps:

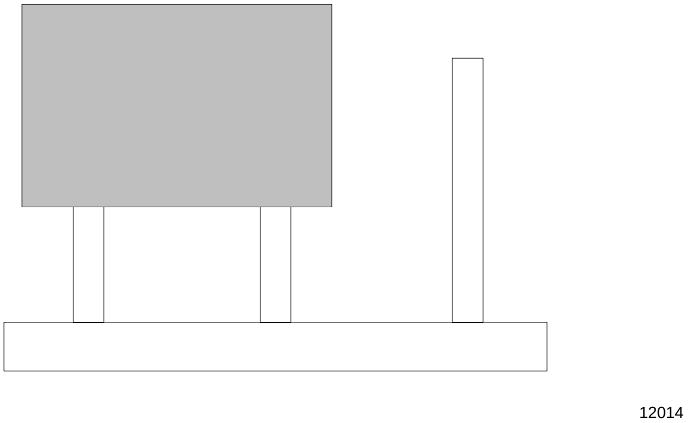
1. Pull the jumper straight up.

2. Position the jumper over the pins you want to connect.

3. Push the jumper straight down.

If you position the jumper correctly, it slides down over the pins easily.

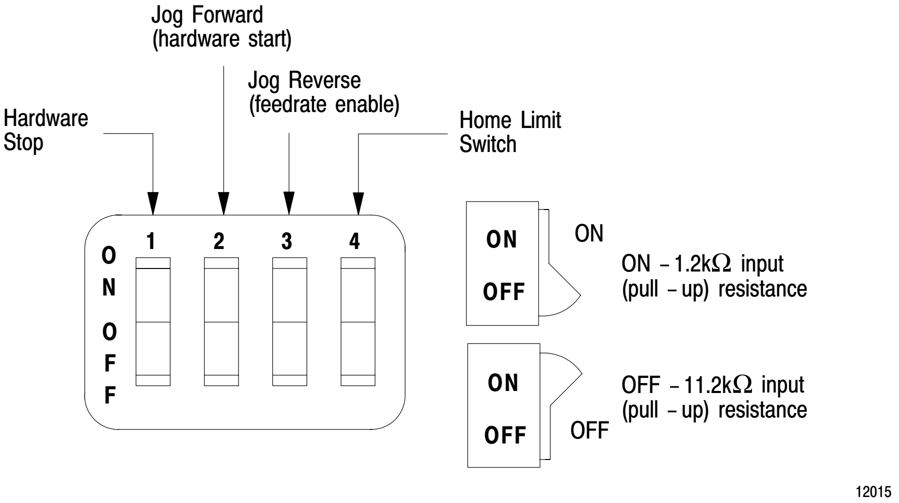
Figure 6.2
Jumper in the Left Position



Selecting Discrete Input Resistance

Select the resistance between each discrete input terminal and the high side of the input power supply. To select 1.2k ohms, set the switch on. To select 11.2k ohms, set the switch off. (Figure 6.3)

Figure 6.3
Discrete-input-resistance Switch Assembly



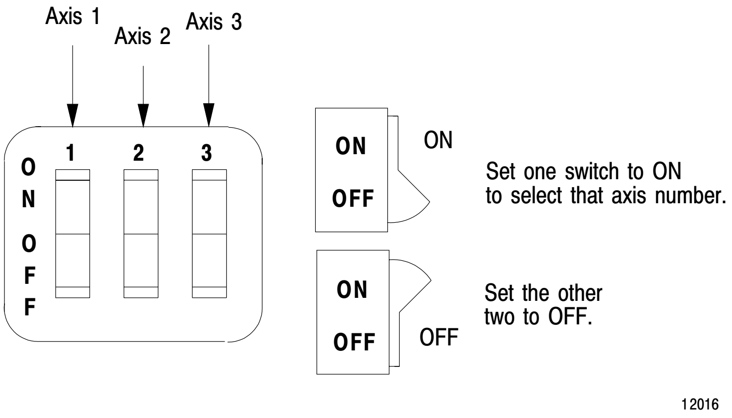
With 1.2k ohms, your input device must sink 4mA for a 5V power supply to 25mA for a 30V power supply. With 11.2k ohms, your input device must sink 0.5mA for a 5V power supply to 2.7mA for a 30V power supply.

Unless your input device cannot sink enough current, select 1.2k ohms because it provides better noise immunity than an 11.2k ohms input resistance.

Selecting Axis Number

Select the axis number as shown in Figure 6.4.

Figure 6.4
Axis-number Switch Assembly



Set to <u>on</u> the switch corresponding to the number for the axis. Set to <u>off</u> the other two switches in the assembly. Set each 1771-ES expander in an I/O chassis to a unique axis number, starting with 1.

Selecting Encoder Input Polarity

Select the polarity of each encoder input to allow your encoder to function properly with the 1771-ES expander (Figure 6.1).

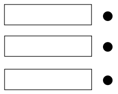
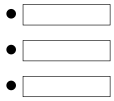
Encoder Polarity	Jumper Position
High–true	Left ▭ ●
Low–True	Right ● ▭

With a differential encoder, the connections and the polarity jumper positions determine the polarity of the feedback signals. With a single-ended encoder, the polarity jumper positions alone determine the polarity of the feedback signals.

The polarity selections are important to the marker logic. Set the polarity so that the marker is true at the same time that channels A and B are true (refer to Figure 3.7)

Selecting Encoder Input Signal Mode

Select the signal mode of each encoder input to match the encoder (Figure 6.1).

Encoder Signal Mode	Jumper Position
Single-ended	Left 
Differential	Right 

Selecting Marker Logic

For almost all encoders, set the marker logic jumper to the bottom position to gate the marker with channel A and channel B. This gives the marker signal a level of noise immunity.

However, if you cannot select the polarity so that the marker on your encoder is always true at the same time as the channel A and B signals, set the market logic jumper to the top position.

Keying

A package of plastic Keys (cat. no. 1771-RK) is provided as standard with each I/O chassis. When properly installed, these keys can guard against the seating of all but a selected type of module in a particular I/O chassis module slot. Keys also help align the module with the backplane connector.

Each module is slotted at the rear edge. Position the keys on the chassis backplane connector to correspond to these slots to allow the seating of the module.

Insert keys into the upper backplane connectors. Position the keys between the numbers at the right of the connectors. Refer to Figure 6.5 for the 1771-M3 controller keying position. Refer to Figure 6.6 for the 1771-ES expander keying positions.

Figure 6.5
Keying Diagram for the 1771-M3 Controller

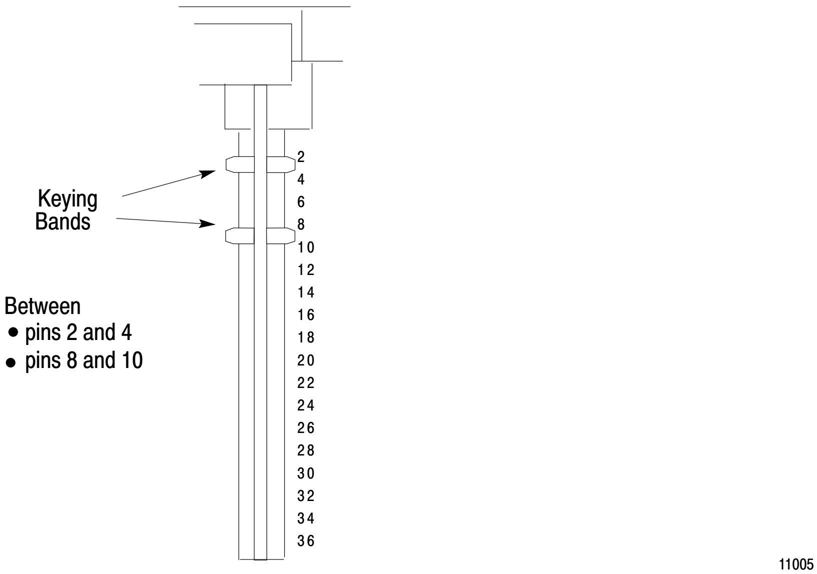
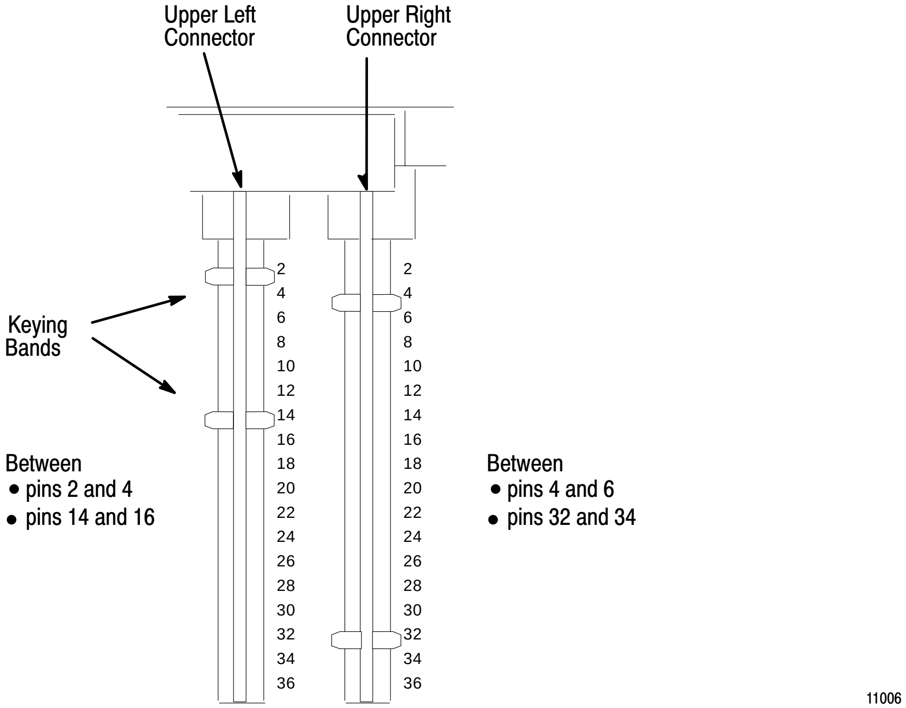


Figure 6.6
Keying Diagram for the 1771-ES Expander



Inserting the Module

To insert a module into an I/O chassis, follow these steps:

1. Remove power from the I/O chassis before inserting or removing a module.

2. Open the module locking latch on the I/O chassis and insert the module into the slot keyed for it.

3. Press the module firmly to seat it into its backplane connector.

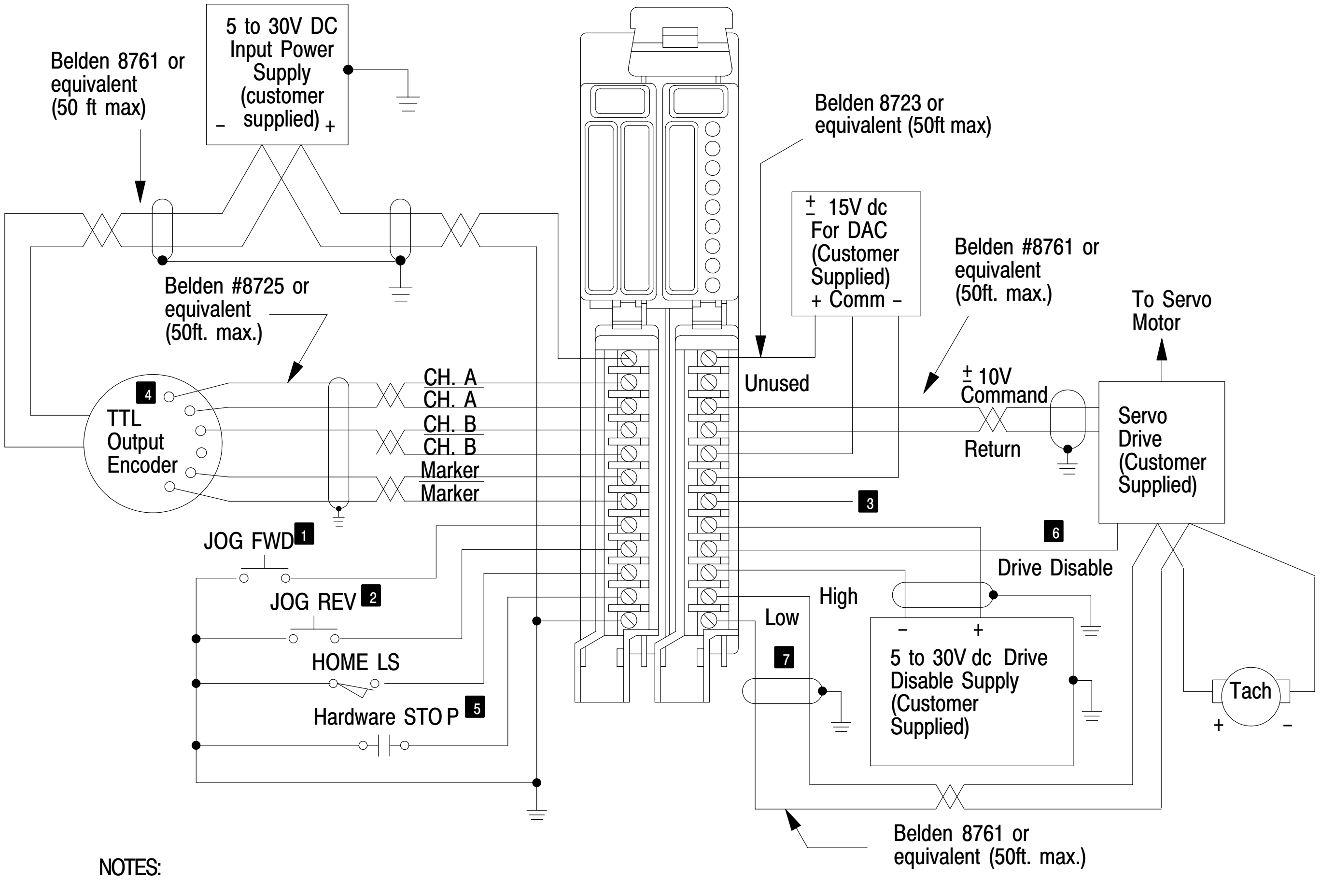
4. Secure the module in place with the module locking latch.

CAUTION: Do not force a module into a backplane connector; if you cannot seat a module with firm pressure, check the alignment and keying. Forcing a module can damage the backplane connector or the module.

Connecting to Terminals

Make connections to the 1771-ES expander as shown in Figure 6.7.

Figure 6.7
Simplified I/O Terminal Connection Diagram



NOTES:

If equipment permits, one supply can be used for encoder and input circuits. Current requirements depend on hardware configuration.

1 In the auto mode, the module accepts this input as the hardware start signal (figure 6.9).

2 In the auto mode, the module accepts this input as the feedrate enable signal (figure 6.9).

3 The module generates a hardware done signal at this +15V dc driver output terminal (figure 6.12).

4 Refer to figures 6.10 and 6.11.

5 Refer to figure 6.8.

6 Refer to figures 6.13 and 6.14.

7 Refer to figure 6.15.

12017

This is a simplified diagram to give you an overall view of how you are to connect these terminals. We give you further details in the following sections and their associated figures. For all connections to the terminals, limit the cable length to 50 feet.

Keep low-level conductors separate from high-level conductors. This is particularly important for cable connections to the encoder. Follow the practices outlined in the PC Grounding and Wiring Guidelines (publication 1770-980).

Power Supplies

Use shielded cable for connecting the input power supply and the analog power supply. Route these cables only with low-level conductors. Keep these power supply cables as short as possible. Ground the common terminal for each of these power supplies.

Encoder and Tachometer

For an encoder or tachometer connection, use only a single, continuous, shielded cable segment. Do not break the cable for connection in a junction box. Connect the cable directly from the encoder to the 1771-ES expander.

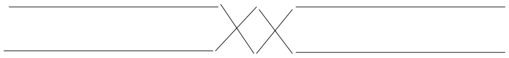
Important: Ensure that the power supply for the encoder provides the voltage recommended by the encoder manufacturers.

Shielded Cables

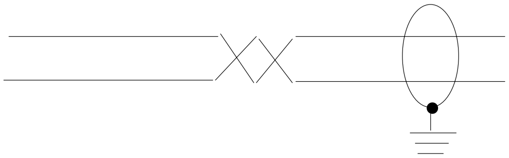
For many connections, we tell you to use shielded cable. Using shielded cables and properly connecting their shields to ground protects against electromagnetic noise interfering with the signals transmitted through the cables.

WARNING: Use shielded cable where we tell you to use it and how we tell you to use it. If you do not, the axis motion in your positioning system could be unpredictable; this could result in damage to equipment and/or injury to personnel.

Within a shielded cable, pairs of wires are twisted together. Using a twisted pair for a signal and its return path provides further protection against noise. We show a twisted pair like this:



We show a shielded twisted pair like this:



Connect each shield to ground at one end only. At the other end, cut the shield foil and drain wire short and cover them with tape to protect against their accidentally touching ground. Keep the length of leads extending beyond the shield as short as possible.

Use cables with the proper number of individually shielded twisted pairs as follows:

To connect to:	Number of Individually Shielded Twisted Pairs:	Use:
Encoder	4	Belden 8725 or equivalent
Analog power supply	2	Belden 8723 or equivalent
All other shielded cable connections	1	Belden 8761 or equivalent

Connecting the Input Supply

To connect the input power supply, follow these steps:

1. Connect the plus side of the input power supply to terminal 1 of the left wiring arm.

2. Connect the minus side to terminal 12 and to ground at the I/O chassis.

3. Connect the shields of the two cable segments if you use the same supply to power the encoder.

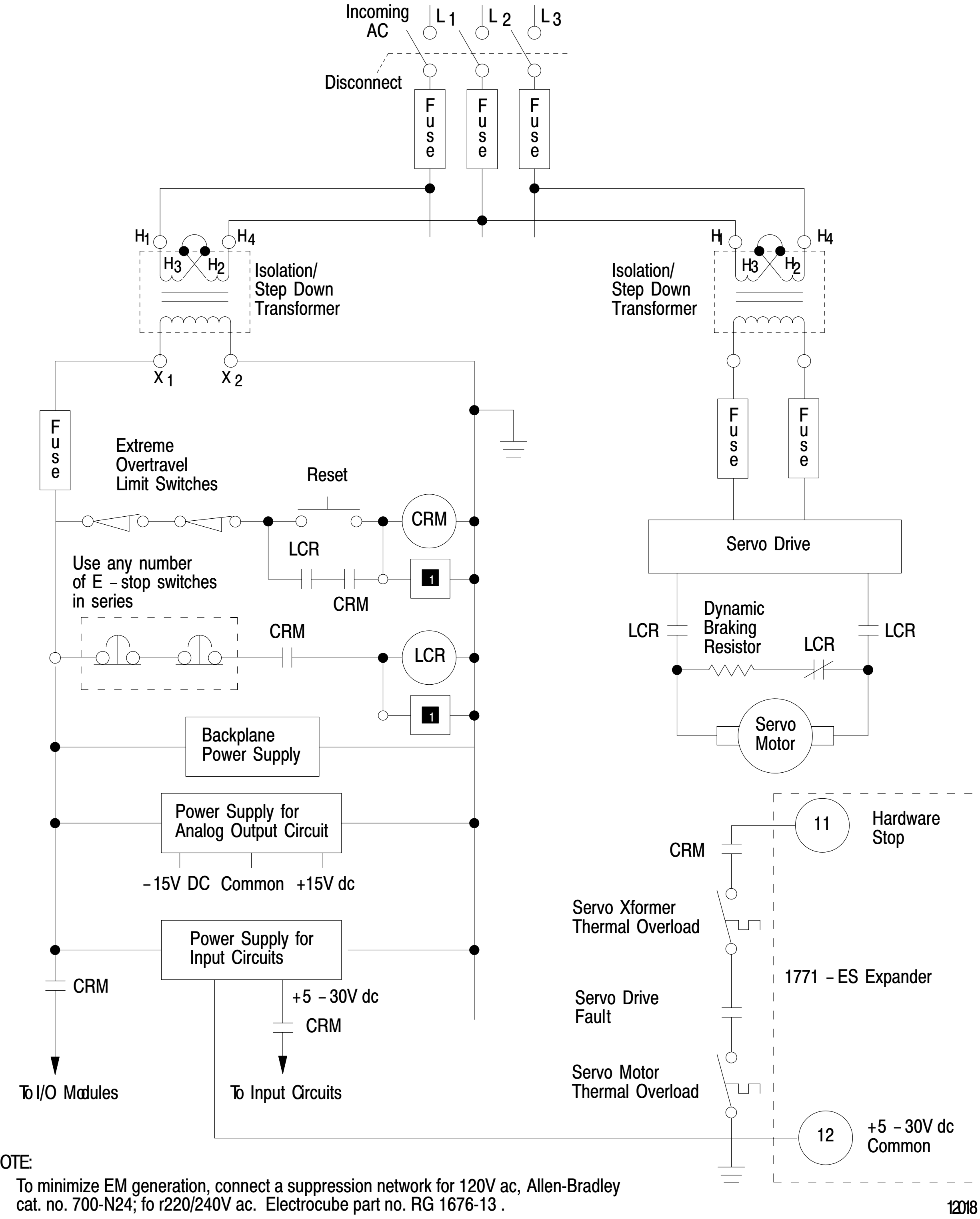
4. Connect the shield to ground at the I/O chassis end.

5. Connect the power-supply chassis to ground.

Connecting Hardware Stop

Before you connect to the hardware stop input, you should first consider overall power distribution, including the master-control relay and loop-contactor relay (Figure 6.8). Connect a suppression network across each relay coil.

Figure 6.8
Simplified Power Distribution with the Master-Control Relay, Loop-contactor Relay, and Hardware Stop



NOTE:

1 To minimize EM generation, connect a suppression network for 120V ac, Allen-Bradley cat. no. 700-N24; fo r220/240V ac. Electrocube part no. RG 1676-13 .

Provide one transformer for the master-control relay (CRM) circuit, the loop-contactor relay (LCR) circuit, the dc power supplies, and any ac I/O chassis. Provide a separate transformer for the servo drives to provide noise immunity.

Use normally-open LCR contacts to switch power from the servo drive to the servo motor. Also, use normally closed LCR contacts to switch in the dynamic braking resistor across the servo motor whenever power is removed from the servo motor. Check with the servo drive and servo motor manufacturer for the resistance and power rating for the dynamic braking resistor.

⚠ **WARNING:**Without a dynamic braking resistor, removing servo motor power while the axis is in motion allows momentum to keep the axis in motion. In an emergency situation, this could be dangerous. A dynamic braking resistor can help stop the servo motor by quickly dissipating the energy of momentum. Even with dynamic braking, a vertical axis may also require an electric brake or counter balance.

An extreme overtravel limit switch or an emergency stop switch can de-energize the LCR, thereby turning off servo motor power. However, abruptly stopping an axis in this way stresses the servo motor and the mechanical linkage. Therefore, use the LCR to stop a moving axis only in an emergency. To stop an axis in a non-emergency situation, use the slide-stop bit in the command block thru the ladder diagram program. A slide stop decelerates the axis feedrate before stopping it. After a slide stop you can use an emergency stop switch if you want to remove power.

Connect a set of normally-open CRM contacts in series with servo transformer overload, servo drive fault, and servo motor overload contacts. Connect this series of contacts between the hardware stop input terminal and common. The opening of any of these contacts indicates that power to the servo motor is interrupted. When any of these sets of contacts open the hardware stop circuit, the following occur:

1. When this circuit opens, the 1771-ES expander immediately sets the velocity command output to zero and disables the serve drive by turning off the drive disable circuit.

2. The 1771-M3 controller sends the hardware stop signal to the PC data table thru the status block transfer.

3. After this circuit closes again, the 1771-ES expander still holds the velocity command at zero and holds the servo drive disabled until you either:

  - send a reset signal through a command block transfer (This allows the 1771-ES controller to maintain the accumulated axis position.)
  - cycle I/O chassis backplane power off, then back on (This clears the accumulated axis position.)

When you restart the axis after a hardware stop, the axis feedrate accelerates before reaching the final velocity rate. This allows a smooth start-up after a hardware stop.

Do not provide switch contacts in the hardware stop circuit for an operator to turn off the axis motion. Opening the hardware stop circuit stops the axis abruptly, stressing the servo drive, the servo motor, and the mechanical linkage, just as the CRM would. Use the hardware stop input only for backup to inform the 1771-ES expander of a condition that has already stopped the axis so that the expander can provide a controlled start-up.

Connecting Home Limit Switch

To connect a home limit switch, follow these steps:

1. Connect a normally open limit switch between the home limit switch terminal and common.

2. Place the limit switch so that it closes as the axis reaches a point approximately one half of an encoder revolution from the point you want to establish as home position.

3. Adjust the angular position of the encoder to have the marker pulse occur precisely at the point you want to establish as home position.

Through the command block transfer you can command a search home function (sections titled “Axis Control Word” and “Axis Control Word 2”). The 1771-ES expander:

- moves the axis to the limit switch
- decelerates the axis

- establishes the point of the next marker pulse following the limit switch as the home position
- stops the axis at the home position

You must re-establish the home position after each time power to the I/O chassis backplane goes off, because the encoder feedback is incremental, .

Connecting Jog Reverse (Feedrate Override Enable)

Figure 6.9 shows details of how to connect jog reverse and feedrate override enable. Follow these steps:

1. Provide a 3-pole selector switch to select between auto and manual mode.

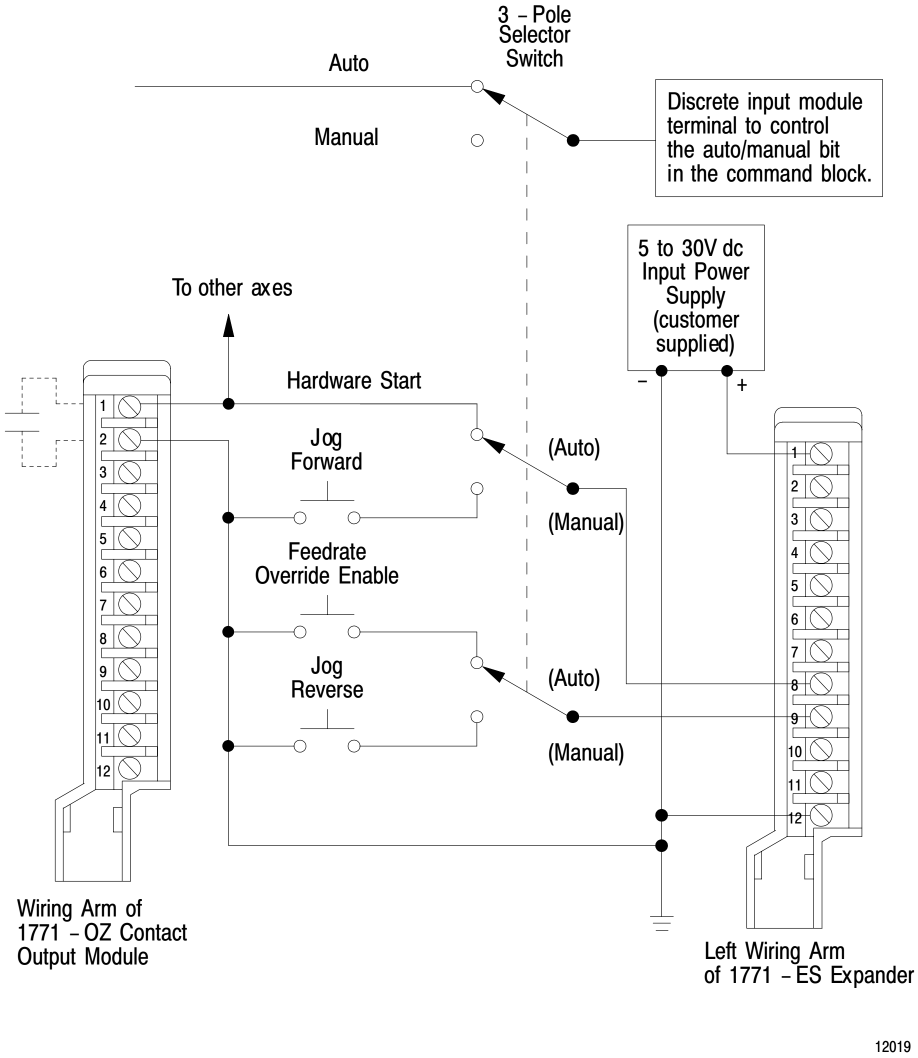
2. Connect one pole of the selector switch to a discrete input module terminal. Use this input to control the auto/manual bit in the control block. This bit controls whether the 1771-ES expander is in the auto or manual mode.

3. Connect a second pole of the selector switch to the jog reverse (feedrate override enable) terminal of the 1771-ES expander.

4. Connect a momentary-contact jog reverse switch to the selector switch contact corresponding to manual on the second pole.

5. Connect a momentary-contact feedrate override switch to the selector switch contact corresponding to auto on the second pole.

Figure 6.9
Connection Details for Jog Forward (Hardware Start) and Jog Reverse (Feedrate Override Enable)


3 – Pole
Selector
Switch
Auto
Manual
Discrete input module terminal to control the auto/manual bit in the command block.
5 to 30V dc Input Power Supply (customer supplied)
–
+
To other axes
Hardware Start
Jog Forward
(Auto)
(Manual)
Feedrate Override Enable
Jog Reverse
(Auto)
(Manual)
1
2
3
4
5
6
7
8
9
10
11
12
Wiring Arm of 1771 – OZ Contact Output Module
Left Wiring Arm of 1771 – ES Expander
12019

In the manual mode, the jog reverse switch controls whether the input is high or low. In the auto mode, the feedrate override enable switch controls whether the input is high or low. You can connect the same feedrate override enable signal to several 1771-ES expanders to coordinate the start of feedrate override for those axes.

Connecting Jog Forward (Hardware Start)

Figure 6.9 also shows details of how to connect jog forward and hardware start. Follow these steps:

1. Connect a third pole of the selector switch to the jog forward (hardware start) terminal of the 1771-ES expander.

2. Connect a momentary-contact jog forward switch to the selector switch contact corresponding to manual on the third pole.

3. Connect an output terminal of a Contact Output Module (cat. no. 1771-OZ) to the selector switch contact corresponding to auto on the third pole.

In the manual mode, the jog forward switch controls whether the input is high or low. In the auto mode, the hardware start output from the 1771-OZ module controls whether the input is high or low.

You can use the ladder diagram program to generate a hardware start signal (by closing the contacts of 1771-OZ module output) when each of several axes generates a hardware done signal. You can connect the same hardware start signal to several 1771-ES expanders to coordinate the start of motion following halt moves for these axes.

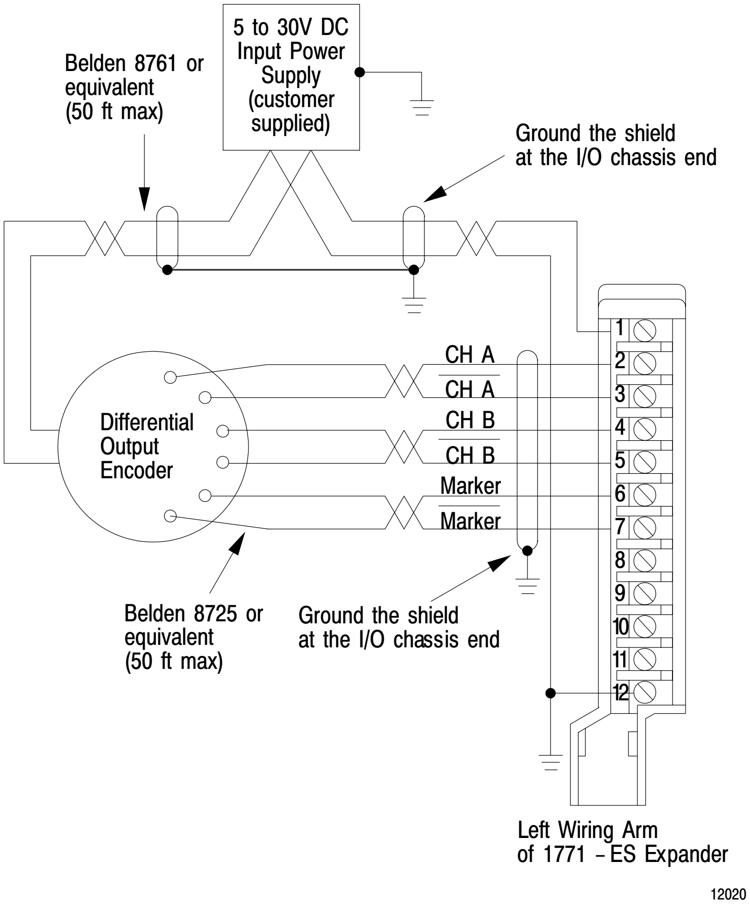
Connecting a Differential Encoder

Figure 6.10 shows details of how to connect a differential encoder. With a differential encoder, reversing the connections on a channel or changing the position of the polarity jumper for the channel reverses the polarity of the signal on that channel. Set the polarity so that the marker is true at the same time that channels A and B are true.

If you switch channel A with channel B, you reverse the direction of the feedback. If the direction of the feedback does not correspond to the axis motion direction, as you have defined it, switch channel A with channel B.

Ground the shield at the I/O chassis end.

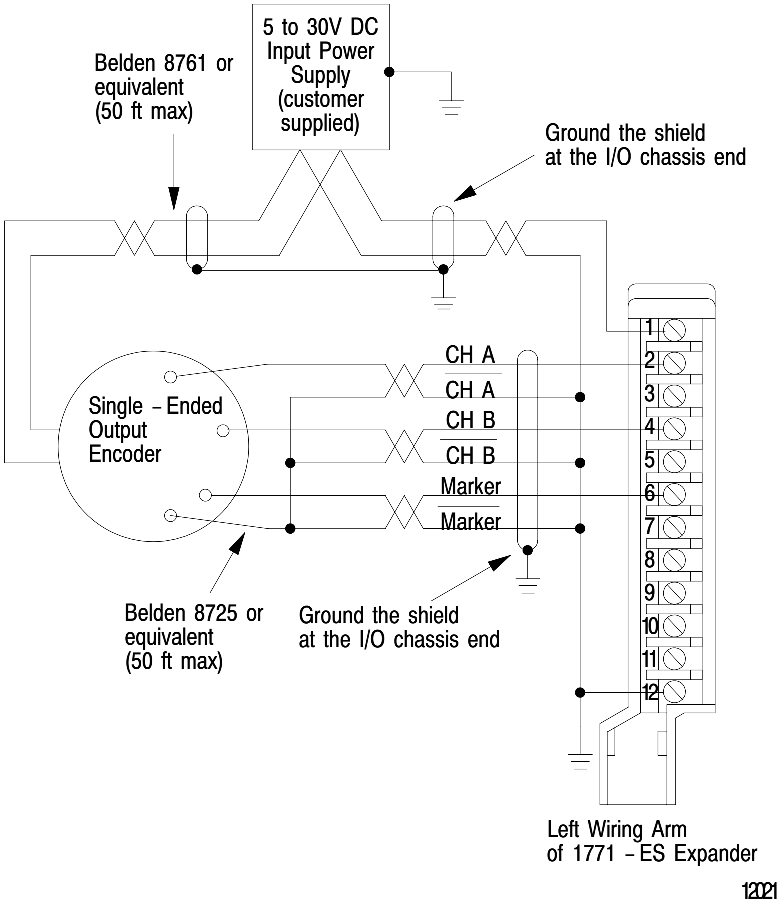
Figure 6.10
Connection Details for a Differential Encoder


5 to 30V DC Input Power Supply (customer supplied)
Belden 8761 or equivalent (50 ft max)
Ground the shield at the I/O chassis end
Differential Output Encoder
CH A
$\overline{\text{CH A}}$
CH B
$\overline{\text{CH B}}$
Marker
$\overline{\text{Marker}}$
1
2
3
4
5
6
7
8
9
10
11
12
Belden 8725 or equivalent (50 ft max)
Ground the shield at the I/O chassis end
Left Wiring Arm of 1771 – ES Expander
12020

Connecting a Single-Ended Encoder

Figure 6.11 shows details of how to connect a single-ended encoder. Connect each channel return line to common.

Figure 6.11
Connection Details for a Single-ended Encoder



If you switch channel A with channel B, you reverse the direction of the feedback. If the direction of the feedback does not correspond to the axis motion direction, as you have defined it, switch channel A with channel B.

Ground the shield at the I/O chassis end.

Connecting the Analog Output Supply

To connect the analog output supply, follow these steps:

1. Connect the plus (+) side of the analog and hardware one output power supply to terminal 1 of the right wiring arm.

2. Connect the minus (-) side to terminal 6.

3. Connect the common to terminal 5.

4. Connect the shield to ground at the I/O chassis.

Connecting Velocity Command

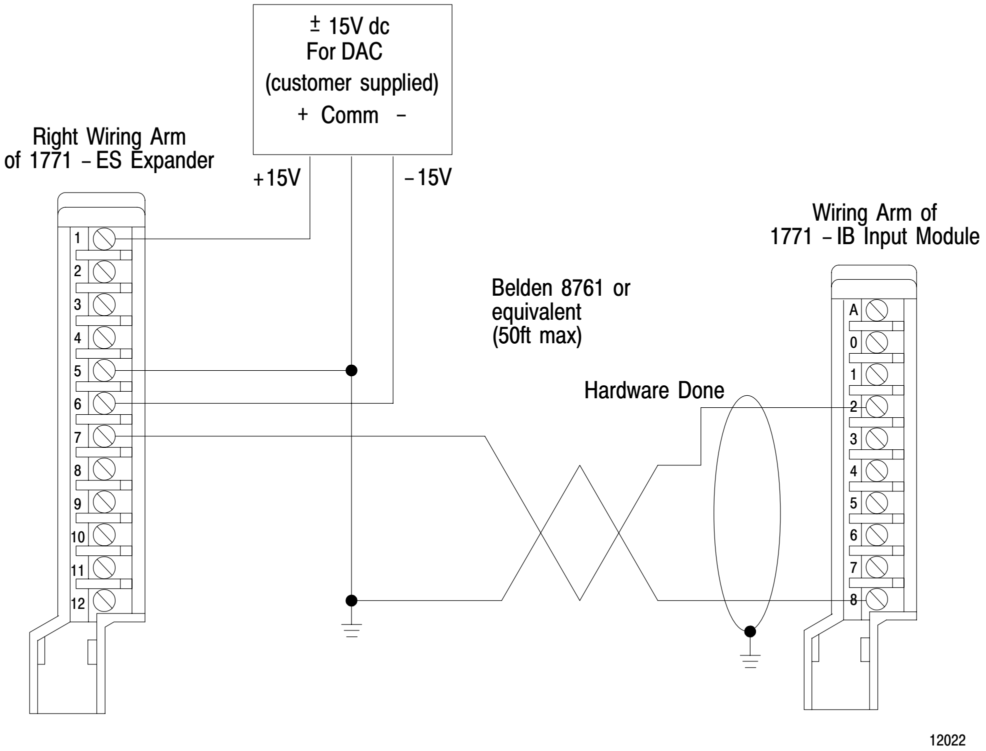
Connect the analog velocity command output signal from terminals 3 and 4 on the right wiring arm to the corresponding terminals of the servo drive. Reversing these connections reverses the direction the axis moves in response to the velocity command. Connect this signal so that the direction of motion that results from it matches the correct direction of motion as you have defined it.

Connect the shield to ground at the servo drive end.

Connecting Hardware Done

Figure 6.12 shows details of how to connect hardware done. Follow these steps:

Figure 6.12
Connection Details for Hardware Done Output



1. Connect the hardware done output from terminal 7 on the right wiring arm to an input terminal of a dc (12-24V) Input Module (cat. no. 1771-IB).

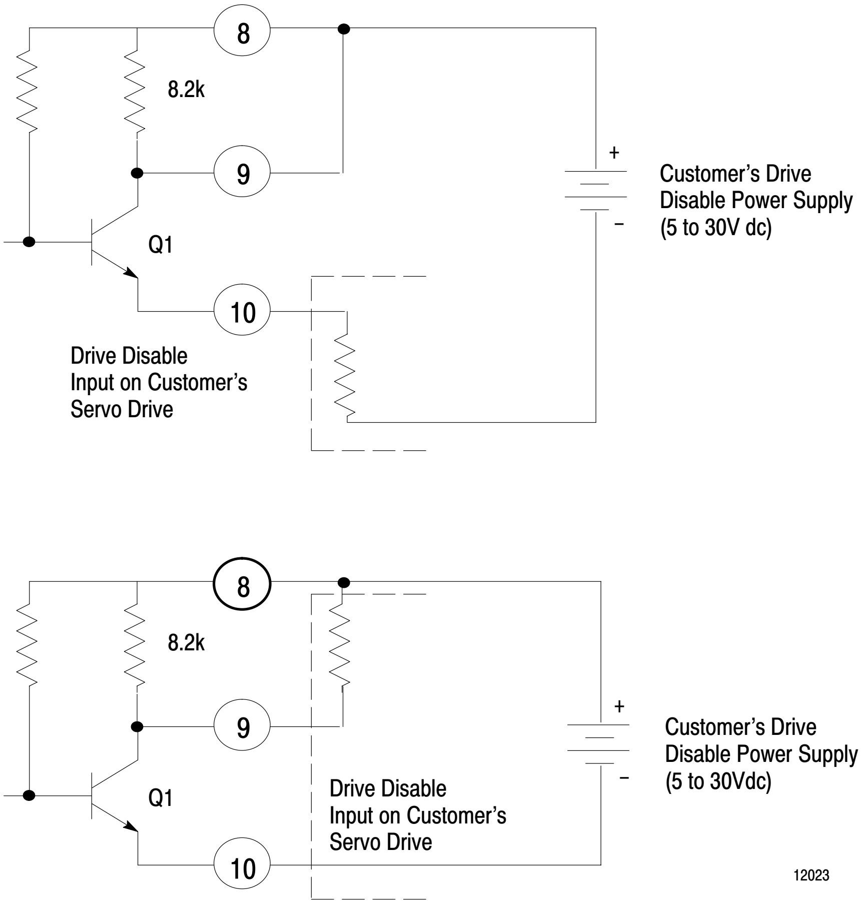
2. Connect the analog and hardware done output power supply common to the 1771-IB input module common terminal. This power supply provides the +15V dc source for the hardware done signal.

Examine the hardware done signal thru the ladder diagram program. You can synchronize the motion of several axes after each halt move: send a hardware start signal to all axes when you have received the hardware done signal from each axis.

Connecting Drive Disable

Figure 6.13 shows details of how to connect drive disable for two basic types of configurations. Some servo drives require a current source connected to an input to enable the drive. Some require a current sink connected to an input to enable the drive. We provide all three connection points (base, emitter, and collector) of the drive disable circuit to provide you with a flexibility of connecting it in a configuration that applies to your servo drive.

Figure 6.13
Connection Details for Two Basic Drive Configurations


8
8.2k
9
Q1
10
+
–
Customer's Drive Disable Power Supply (5 to 30V dc)
Drive Disable Input on Customer's Servo Drive
8
8.2k
9
Q1
10
+
–
Customer's Drive Disable Power Supply (5 to 30Vdc)
Drive Disable Input on Customer's Servo Drive
12023

a) Current Sourcing Configuration

Drive Enable Q1 on: Current is sourced from terminal 10 into the servo drive.

Drive Disabled Q1 off: Current into the servo drive is inhibited.

b) Current Sinking Configuration
Drive Enable Q1 on: Current is sunk thru terminal 9 and Q1.

Drive Disabled Q1 off: Current thru Q1 is inhibited. Terminal 9 is pulled up to the potential of terminal 8.

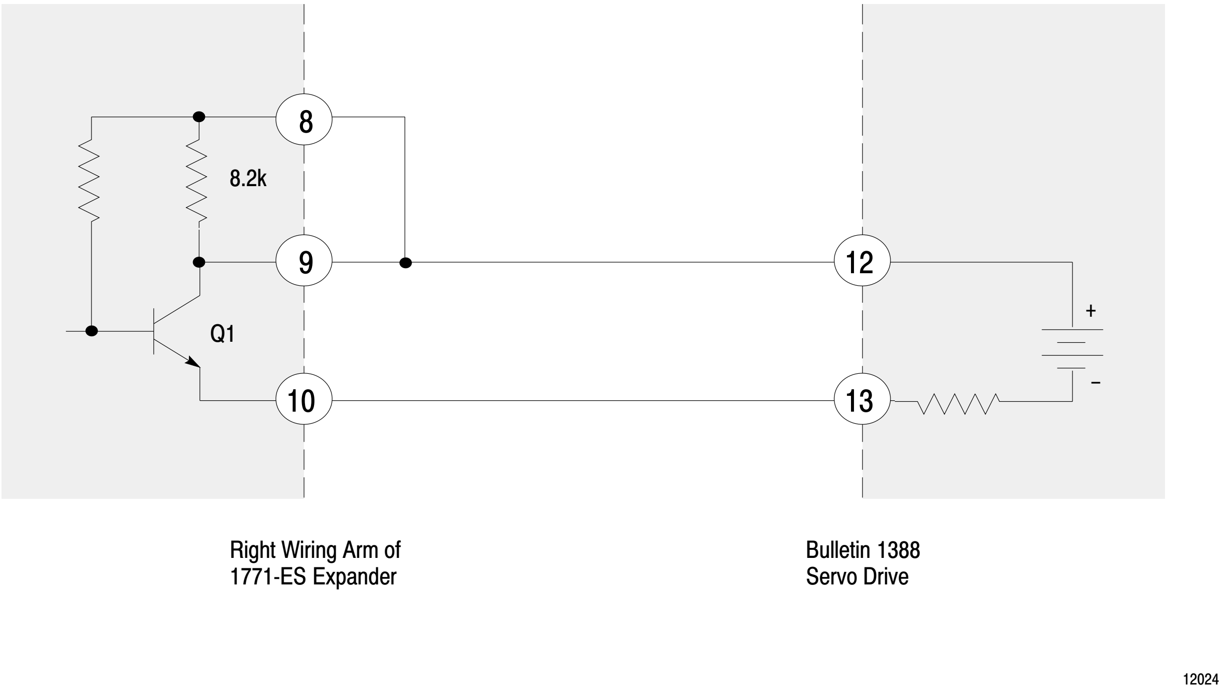
For the drive disable circuit, you must provide a 5-30V dc power supply which can provide 100mA maximum. The power supply can be separate or an integral part of the servo drive. Each of the configurations of figure 6.13 includes a separate power supply.

Figure 6.13a shows a current sourcing configuration. Normally the drive disable circuit is on, sourcing current into the drive thru terminal 10. When the drive disable circuit turns off, the drive is disabled.

Figure 6.13b shows a current sinking configuration. Normally the drive disable circuit is on, sinking current from the drive thru terminal 9. When the drive disable circuit turns off, the drive is disabled.

Figure 6.14 shows how to connect the drive disable circuit to the Bulletin 1388 servo drive which has an internal power supply and requires a current source to enable it.

Figure 6.14
Connection Details for Providing a Drive–disable Signal to the Bulletin 1388 Servo Drive

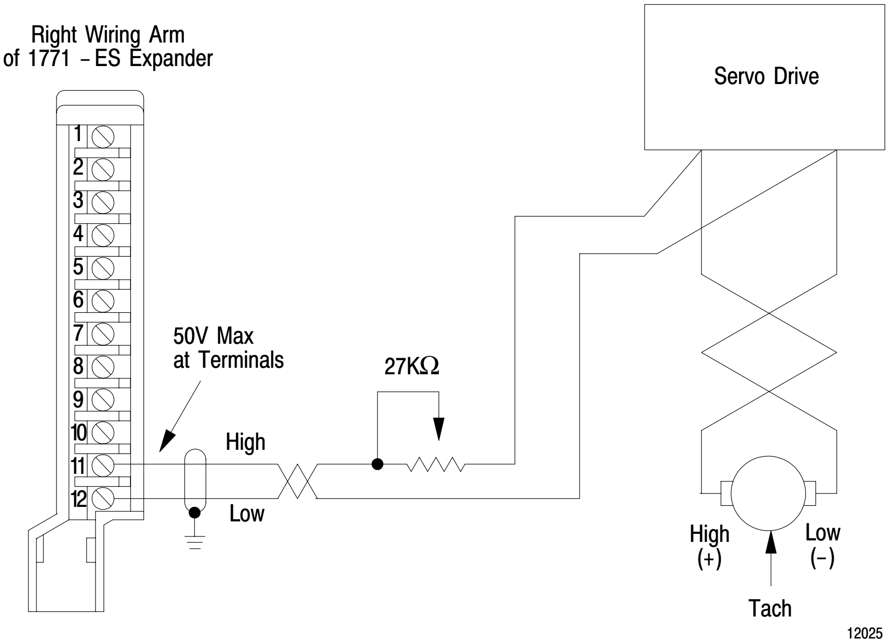


Note that whatever configuration your drive requires, you must connect the plus side of the power supply to terminal 8 on the right wiring arm of the 1771-ES expander. Without this connection, the drive disable circuit will not turn on; the 1771-ES expander will not enable the servo drive.

Connecting the Tachometer

Figure 6.15 shows details of how to connect the tachometer. Follow these steps:

Figure 6.15
Connection Details for Tachometer



1. Connect the tachometer directly to the servo drive.

2. Connect the tachometer signal at the servo drive to the right wiring arm of the 1771-ES expander. This allows the 1771-ES expander to detect loss of tachometer feedback at the servo drive. Limit the voltage at the terminals to 50V maximum. Tachometers typically generate much larger voltages than 50V at high speed. Therefore, you must drop the voltage thru a voltage divider.

3. Unless you have access to a voltage divider in the servo drive, place a 27k ohms 1/4 Watt potentiometer between the servo drive and terminal 11 of the 1771-ES expander.

4. Set the potentiometer for maximum resistance until you perform the integration procedures (chapter 9).

5. Connect the tachometer high signal to terminal 11.

6. Connect the tachometer low signal to terminal 12.

7. Connect the shields of the cable segments.

8. Connect the shield to ground at the I/O chassis end.

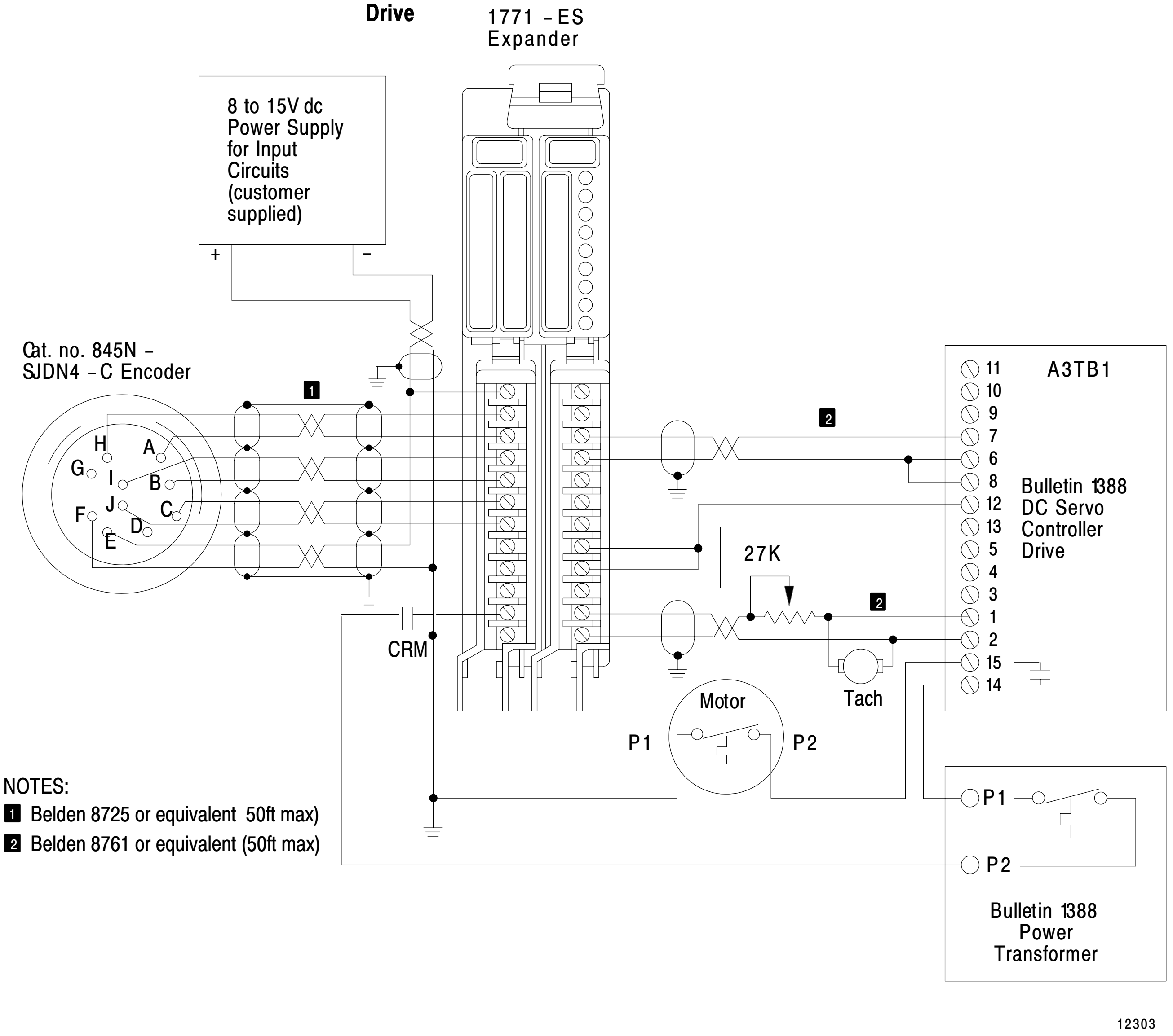
Connecting A-B Encoder and Drive

Figure 6.1 shows the jumpers in the position in which we place them for shipping the 1771-ES expander to you. These channel polarity jumper settings select high-true polarity. These channel signal mode jumper settings select differential mode. This marker logic jumper setting selects the marker to be gated with channel A and channel B. If you use the Allen-Bradley 845N-SJDN4-C encoder, leave the jumpers set to the position shown in Figure 6.1.

With the jumpers set as shown in figure 1, connect the 845N-SJDN 4-C encoder to the 1771-ES expander as shown in Figure 6.16. We show the **channel A** signal connection reversed with the **not channel A** connection and the **channel B** signal connection reversed with the **not channel B** connection. This inversion of the channel A and B polarity allows the marker to be high at a time when both channels A and B are high.

Use an 8 to 15V dc power supply for the input circuits. Connect the plus side of the supply voltage to pin E of the encoder. With this configuration, 5V dc power is generated at the encoder; the signals from the encoder are 5V dc.

Figure 6.16
Connections to a Cat. No. 845N-SJDN4-C Encoder and a Bulletin 1388 dc Servo Controller Drive



Connect the bulletin 1388 dc servo controller drive and its bulletin 1388 power transformer to the 1771-ES expander as shown in Figure 6.16. Tachometer input terminal 12 on the 1771-ES expander and terminal 2 on the drive each connect to a dc common at ground potential; therefore, you must connect these terminals directly as shown.

Connect the analog output signal from terminal 3 of the 1771-ES expander to terminal 7 of the drive. Connect the analog return signal from terminal 4 of the 1771-ES expander to terminals 6 and 8 of the drive. With this signal orientation, when you connect the tachometer to the drive with the proper polarity for negative feedback, the signal will also have the proper polarity for loss-of-feedback detection at the 1771-ES expander. If you use the opposite analog output signal orientation, you will not be able to utilize the loss-of-feedback detection feature.

WARNING: Always utilize the loss-of-feedback feature. Without loss-of-feedback detection, if encoder or tachometer feedback is lost, unexpected axis motion can occur, resulting in damage to equipment and/or injury to personnel.

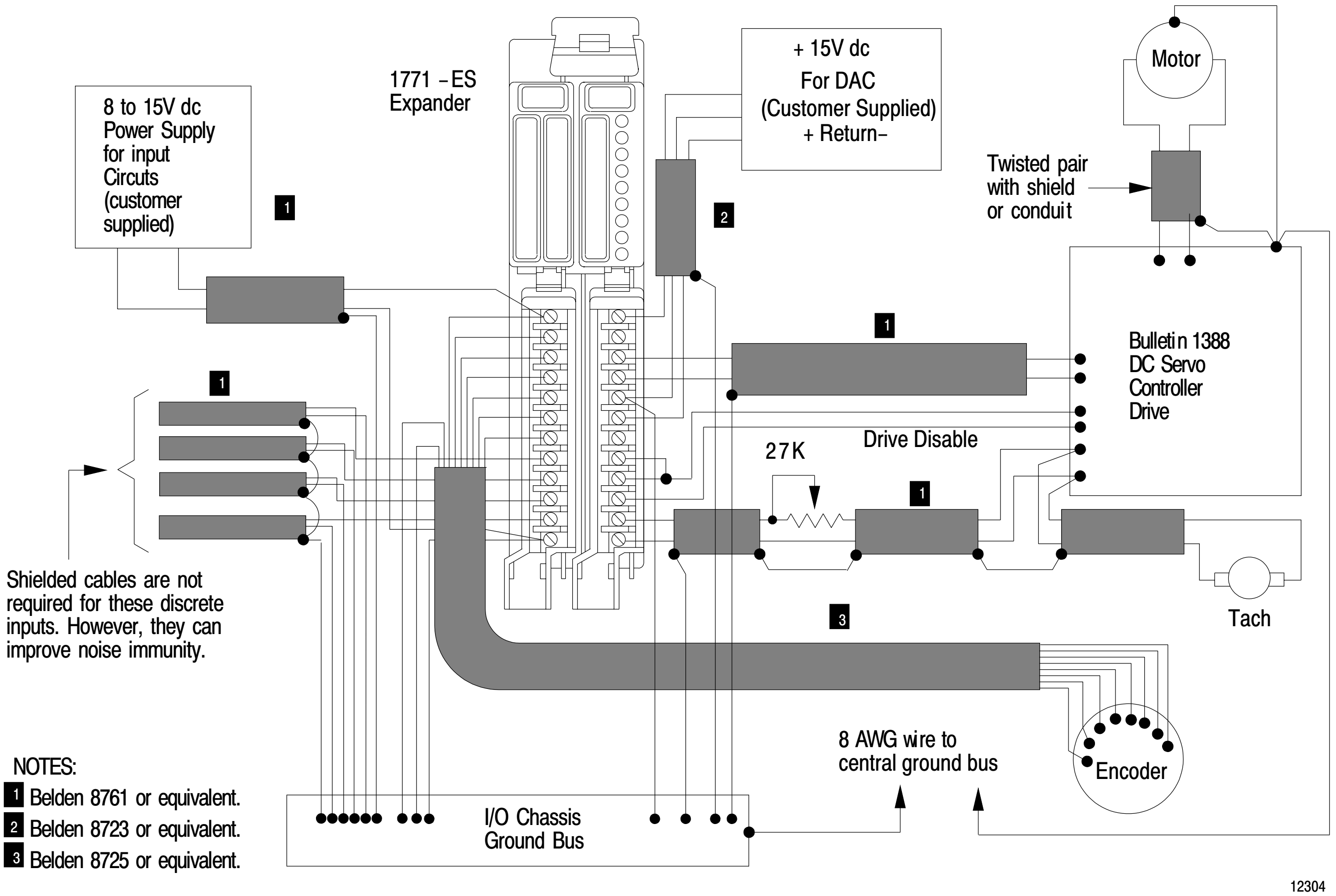
Limit the cable lengths to 50 feet. If your application requires a cable length greater than 50 feet, contact your local Allen-Bradley representative.

Grounding Cable Shields

Figure 6.17 is a pictorial representation of the shielded cable connections. Mount a ground bus directly below the I/O chassis to provide a connection point for cable shield drain wires and the common connections for the input circuits. Connect the I/O chassis ground bus through 8 AWG wire to the central ground bus to provide a continuous path to ground.

The tachometer cable is broken into three segments because of the connection to the drive and potentiometer in the middle of the cable. Connect these cable shield segments together as shown. Connect the shield to ground only at the I/O chassis end. Do not connect the shield to the drive.

Figure 6.17
Shielded Cable Grounding Connections



NOTES:
1 Belden 8761 or equivalent.
2 Belden 8723 or equivalent.
3 Belden 8725 or equivalent.

Connecting AC Power

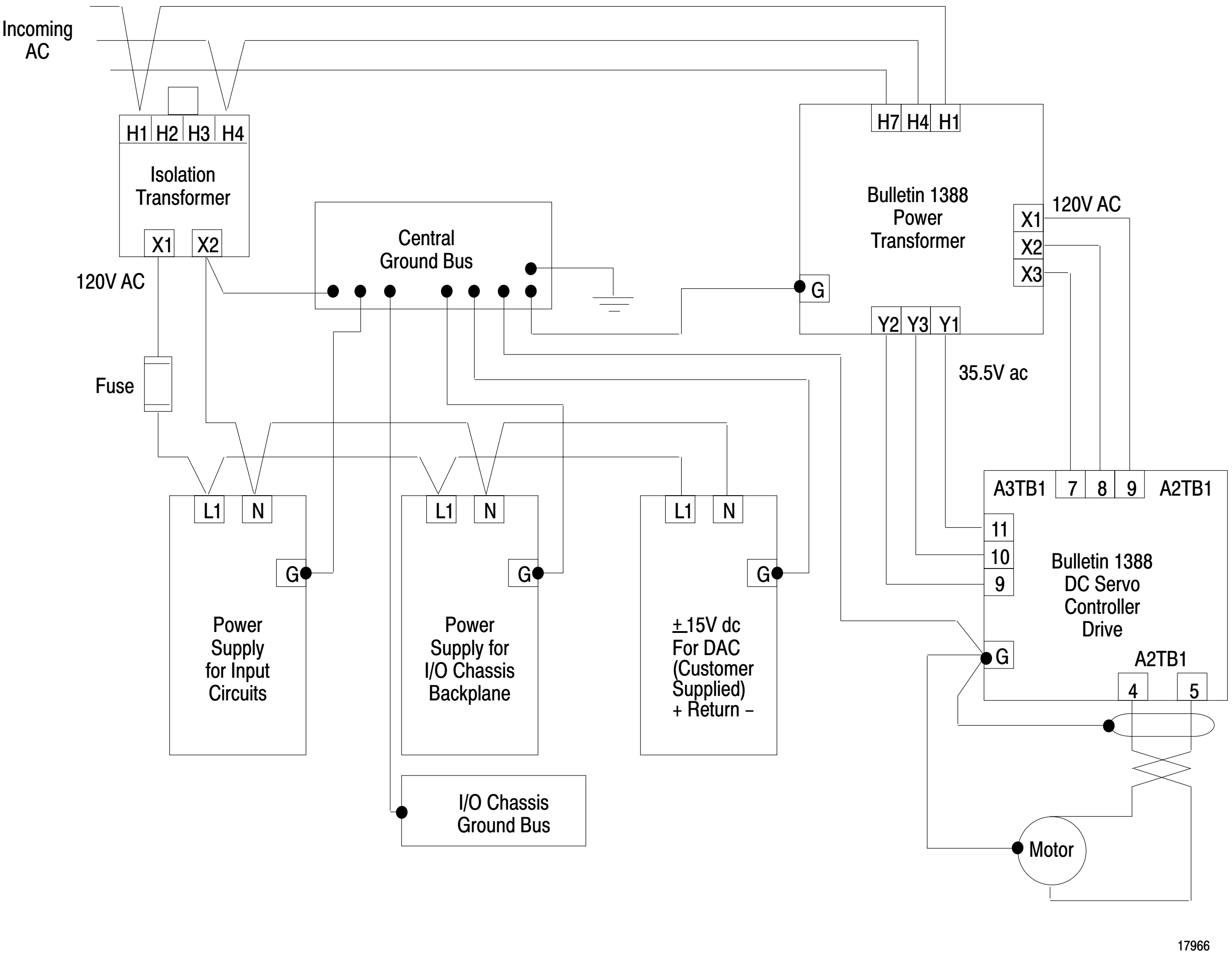
Figure 6.18 shows ac power connections. Incoming ac connects to the primary of the bulletin 1388 power transformer. Both the 120V secondary and the 35.5V secondary connect to the bulletin 1388 dc servo controller drive.

Incoming ac also connects to the primary of an isolation transformer. The secondary of the isolation transformer connects to:

- the power supply for the input circuits
- the power supply for the I/O chassis backplane
- the power supply for the analog output circuit

Figure 6.18 shows a grounded ac system; the low side of the isolation transformer is connected to the central ground bus. Figure 6.18 also shows connections from the central ground bus to each chassis and to the I/O chassis ground bus shown in Figure 6.17.

Figure 6.18
AC Power and Ground Connections


Incoming AC
H1
H2
H3
H4
Isolation Transformer
X1
X2
120V AC
Fuse
Central Ground Bus
H7
H4
H1
Bulletin 1388 Power Transformer
X1
X2
X3
120V AC
G
Y2
Y3
Y1
35.5V ac
L1
N
G
Power Supply for Input Circuits
L1
N
G
Power Supply for I/O Chassis Backplane
L1
N
G
±15V dc For DAC (Customer Supplied) + Return –
I/O Chassis Ground Bus
A3TB1
7
8
9
A2TB1
11
10
9
Bulletin 1388 DC Servo Controller Drive
G
A2TB1
4
5
Motor
17966

Start-up Sequence

After properly installing your servo positioning assembly, formatting the data blocks, entering the program, and integrating each axis, you start up the system in the following sequence.

1. De-energize the CRM relay.

2. Turn on the dc power connected to the wiring arms.

3. Turn on the power supply for the I/O chassis backplane.

4. Energize the CRM relay.

5. Generate a reset command through the command block.

Summary

Now that you have installed the servo positioning assembly, you are ready to enter data blocks into the data table of the PC processor. During installation you made hardware selections to direct module operation. In chapter 7, we tell you how to make software selections to direct other aspects of module operation.

Chapter 7

Formatting and Interpreting Data Blocks

Chapter Objectives

The previous chapter told you how to install the modules. During installation, you made hardware selections through switch and jumper settings. These hardware selections direct some aspects of module operation.

This chapter tells you how to make software selections through data blocks you set up in the data table. Through data blocks you direct module operation.

This chapter also tells you how to monitor module operation through a data block that the module sends to the data table.

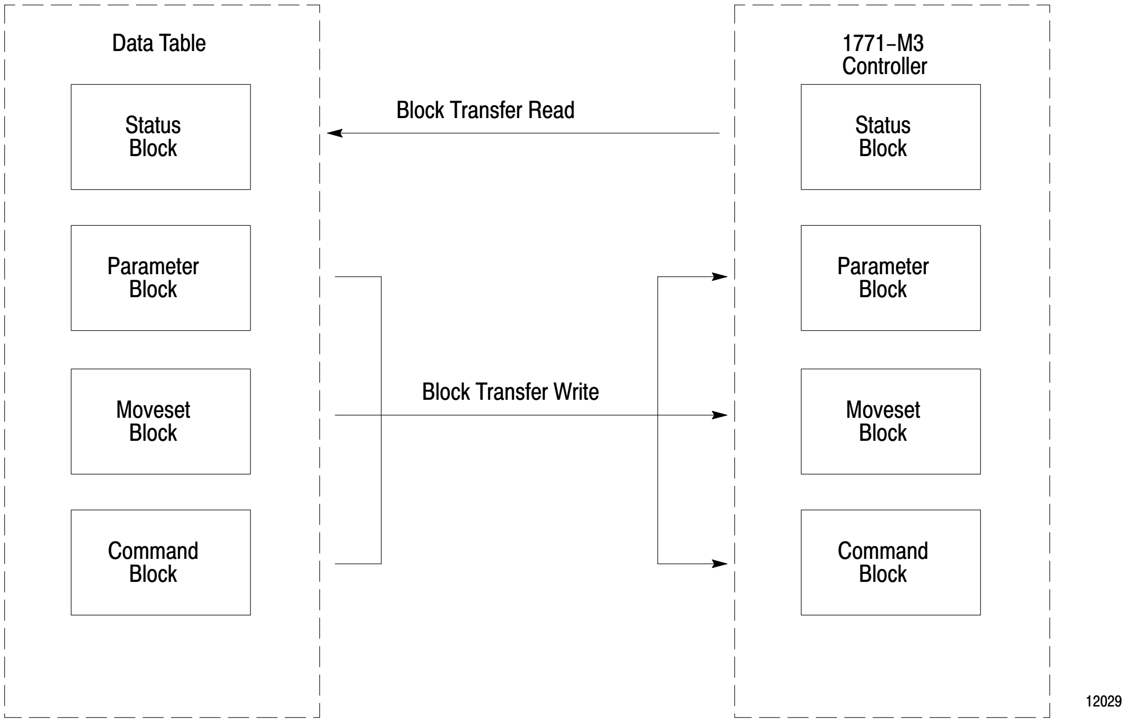
Relationship of Data Blocks

You must program the PC processor to communicate with the 1771-M3 controller through a block-transfer-read instruction and a block-transfer-write instruction. The data blocks are:

- status block
- parameter block
- moveset block
- command block

The block-transfer-read instruction transfers status block data from the 1771-M3 controller to the data table. The block-transfer-write instruction transfers the parameter block, the moveset block, and the command block data from the data table to the module. (Figure 7.1).

Figure 7.1
The Status Block Transfers to the Data Table - the Parameter, Moveset, and Command Blocks go to the 1771-M3 Controller



Status Block

The status block is regularly transferred to the data table to provide updated information about the current status of each axis. This status includes:

- actual axis position
- in position
- at home position
- slide stop
- emergency stop
- software travel limit exceeded
- feed reduction
- excess following error
- auto/manual mode
- address pointer to tell the program which block (parameter, moveset, or control) to write transfer to the 1771-M3 controller next
- diagnostic status that tells you where programming errors are in parameter, moveset, and command blocks

The first block transfer after power-up writes a 6-word status block into the data table. After that, the status block consists of 6 words for a 1-axis system, 10 words for a 2-axis system, or 14 words for a 3-axis system. You establish the address for the status block through the block transfer read instruction. Because axis command and status data is stored in the data table, axis motion control can interact with other axes, discrete I/O, and report generation.

Parameter Block

The parameter block for a 1-axis system has 25 words; a 2-axis system has 44 words; a 3-axis system has 63 words.

You specify parameters for each axis separately. You specify parameters such as:

- software travel limits
- home position
- servo gain
- global accel/decel rate
- rapid traverse rate

In the parameter block, you also specify the address of the parameter block, the command block, and the first moveset block for each axis. With these addresses, the 1771-M3 controller can ask (through the status block) for the block it needs at any particular time.

The processor transfers the parameter block to the 1771-M3 controller through a block transfer write. This provides axis parameter information after a power-up and after a command block commands a reset or new parameters.

Moveset Block

A moveset block describes a sequence of axis moves. You can program axis motion to provide either single-step moves or continuous moves.

Each move requires a minimum of three words (a single-move control word and two words to define position or dwell time) and can include three optional words (a rate word, an accel word, and a decel word) for a total of six. A moveset control word applies to the entire block. If additional moveset blocks are needed, you also need a next-moveset-point word. A moveset block can be 64 words long maximum and describe 21 moves maximum. To describe 21 moves in a single moveset block, all 21 moves would have to use the global accel/decel and final rate values from the parameter block.

Upon request from the status block, the PC processor sends a moveset block to the 1771-M3 controller, which transfers each move description to the 1771-ES expander one at a time. The 1771-ES expander generates the analog voltage to command axis motion as programmed.

Command Block

The command block for a 1-axis system has up to four words; a 2-axis system has up to eight words; a 3-axis system has up to 12 words. This block regularly transfers from the data table to provide commands (such as start, slide stop, search home, jog, reset and offset) for each axis unless the 1771-M3 controller needs a parameter or moveset block. You must include the command block address in the parameter block.

Data Table Allocation

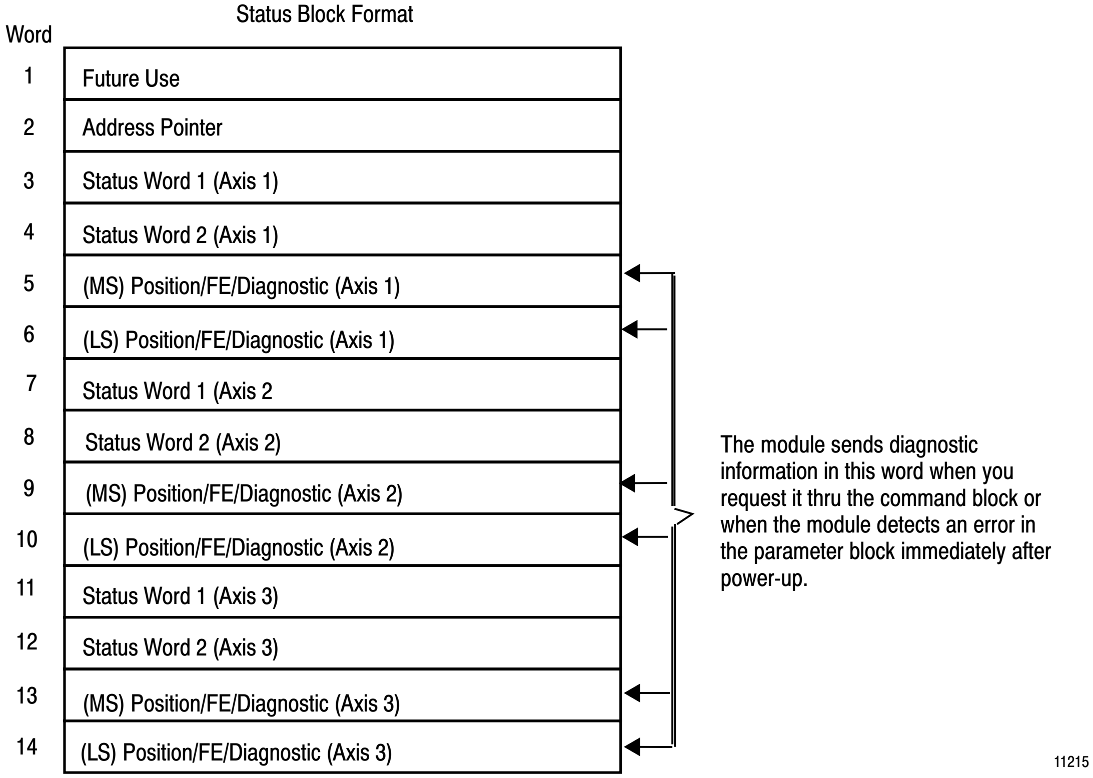
You must allocate a sufficiently large data table area for the data blocks needed in the block transfer communication. Furthermore, the parameter block must start at least 63 words before the end of a contiguous data table area. Also, each moveset block (regardless of size) must start at least 64 words before the end of a contiguous data table area. For a PLC-2 family processor, assign data block addresses of 200 or greater to avoid processor work areas.

Status Block

The status block, which is the only block transferred from the 1771-M3 controller to the processor, contains information about axis and servo positioning assembly status. The first block transfer after power-up writes a 6-word status block into the data table. After that, the status block consists of word assignments (Figure 7.2):

Number of Axes	Size of Status Block
1	6 words
2	10 words
3	14 words

Figure 7.2
Status Block - Showing Word Assignments



Status Block Format

Word	
1	Future Use
2	Address Pointer
3	Status Word 1 (Axis 1)
4	Status Word 2 (Axis 1)
5	(MS) Position/FE/Diagnostic (Axis 1)
6	(LS) Position/FE/Diagnostic (Axis 1)
7	Status Word 1 (Axis 2
8	Status Word 2 (Axis 2)
9	(MS) Position/FE/Diagnostic (Axis 2)
10	(LS) Position/FE/Diagnostic (Axis 2)
11	Status Word 1 (Axis 3)
12	Status Word 2 (Axis 3)
13	(MS) Position/FE/Diagnostic (Axis 3)
14	(LS) Position/FE/Diagnostic (Axis 3)

The module sends diagnostic information in this word when you request it thru the command block or when the module detects an error in the parameter block immediately after power-up.

We reserve the first word of the status block for future use. It contains all zeros when returned by the 1771-M3 controller. The second word is an address pointer that identifies the next block the processor is to transfer to the 1771-M3 controller. Words 3 thru 6 provide the status of axis 1. Words 7 thru 10 provide the status of axis 2. Words 11 thru 14 provide the status of axis 3.

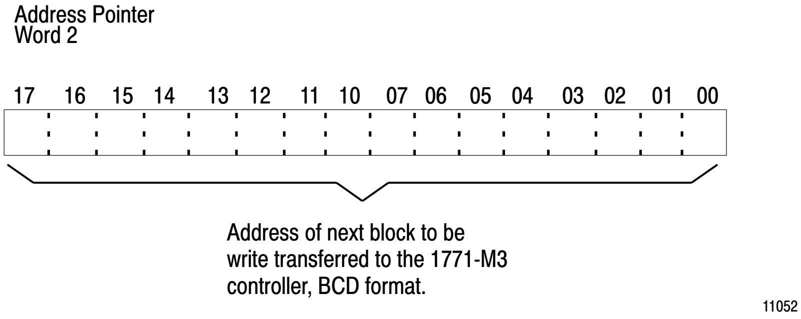
The following sections describe status block words.

The servo positioning assembly configures all words in the status block.

Address Pointer

The address pointer word (Figure 7.3) contains, in BCD format, the data table address of the next block to be transferred from the processor to the 1771-M3 controller. Your ladder diagram program reads this address and uses it to configure a write block transfer instruction. The 1771-M3 controller programs this word according to its requirements. When it does not need to request the parameter block or a moveset block, it requests the command block.

Figure 7.3
Address Pointer Word



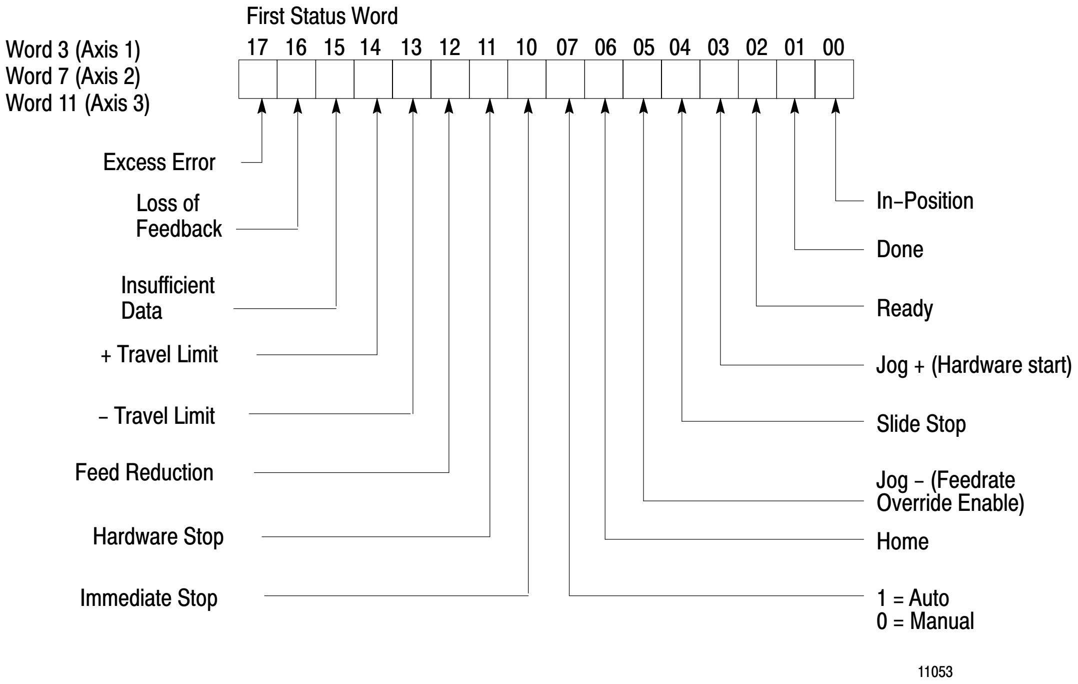
The value that appears in this word is one of the pointer addresses you put into:

- word 2 (parameter block) of the parameter block
- word 3 (command block) of the parameter block
- word 4 (initial moveset block, axis 1) of the parameter block
- word 5 (initial moveset block, axis 2) of the parameter block
- word 6 (initial moveset block, axis 3) of the parameter block
- the last word (next moveset block) of a moveset block

First Status Word

Each bit of the first status word (Figure 7.4) corresponds to a particular axis condition.

Figure 7.4
First Status Word



Bit 0 In-Position

The 1771-M3 controller turns on this bit when following error is less than twice the in-position band value programmed in the parameter block (word 11). When the in-position bit is on, it indicates that the axis has moved to within a specified distance of the programmed end point.

Bit 1 Done

The 1771-M3 controller turns on this bit when the 1771-ES expander has finished feeding the axis for a programmed move or finished a dwell.

Bit 2 Ready

The 1771-M3 controller turns off the ready bit after power-up or after you execute the reset command. The controller turns on this bit when it receives valid parameter-block values. When the ready bit is on, the 1771-M3 controller is ready to respond to commands you issue through the command block.

The processor must not transfer the command or moveset blocks to the servo controller until the ready bit is on.

Bit 3 Hardware Jog + (Hardware Start)

The 1771-M3 controller turns on this bit when the 1771-ES expander recognizes a jog plus or hardware start input signal.

Bit 4 Slide Stop

The 1771-M3 controller turns on this bit when it receives a slide-stop request from the command block (word 1, bit 5). The slide stop status bit stays on even after the slide-stop command is no longer present in the command block. This bit turns off when you command axis motion or reset. A reset command while the axis is in motion will also turn on this bit and cause a slide stop. When the axis stops, this bit turns off.

Bit 5 Hardware Jog - (Feedrate Override Enable)

The 1771-M3 controller turns on this bit when the 1771-ES expander recognizes a jog minus or feedrate override enable input signal.

Bit 6 Home

The 1771-M3 controller turns on this bit when the axis feed is done after any command to move to the home position, if you have established a home position. You establish a home position through an initialize home, or search home command. This bit turns off when the axis moves away from the home position.

Bit 7 Auto/Manual

This bit indicates the current mode of the axis, based on the status of the auto/manual bit (word 1, bit 7) in the command block (1=auto, 0=manual).

Bit 10 Immediate Stop

When this bit is on it indicates that the 1771-ES expander is holding its analog output signal at zero and is disabling the servo drive through its drive disable output. You can clear this immediate stop condition through a reset command or by cycling I/O chassis backplane power off, then on. Commands and events that can cause the immediate stop condition are:

- software stop command
- hardware stop input open
- excess following error
- loss of feedback
- loss of power
- firmware or hardware watchdog timeout on the 1771-ES expander

Bit 11 Hardware Stop

The 1771-M3 controller turns on this bit only if the hardware stop input of the 1771-ES expander is open. Note that the immediate stop bit (bit 10) is also on if this bit is on. You can turn off this bit with a reset command or by cycling power to the I/O chassis backplane.

Bit 12 Feed Reduction

This bit goes on when axis following error reaches 106.25% of rapid traverse following error, resulting in 50% feedrate reduction, but has not necessarily reached the excess error point. When axis following error does reach the excess error point, the feed reduction bit stays on, and the immediate stop status bit goes on.

Important: If the excess error point is less than 106.25% of rapid traverse following error, immediate stop occurs before feed reduction. Consequently, the feedrate reduction bit in the status block does not turn on.

Bit 13, 14 + and - Travel Limits

These bits are on when the axis is at the corresponding software travel limit positions. You enter the travel limits in the parameter block.

Bit 15 Insufficient Data

When the servo positioning assembly receives a command to execute axis motion, such as start or begin, but does not have moveset data to execute a move, it turns on the insufficient data bit. It also turns on this bit when you issue an escape command, even though you had never stored an escape move on the 1771-ES expander.

This insufficient data bit stays on until the 1771-M3 controller receives a new moveset block and then a start or begin command.

Bit 16 Loss of Feedback

This bit is meaningful only if you enable the loss-of-feedback detection feature by setting bit 15 of the most significant home position word of the parameter block.

If loss-of-feedback is enabled, and the servo positioning assembly detects a loss of feedback, it turns on the loss-of-feedback bit in the status word. If this bit is on, then the immediate stop bit in the status block is on, indicating that the 1771-ES expander has executed an immediate stop after detecting the loss-of-feedback.

Bit 17 Excess Error

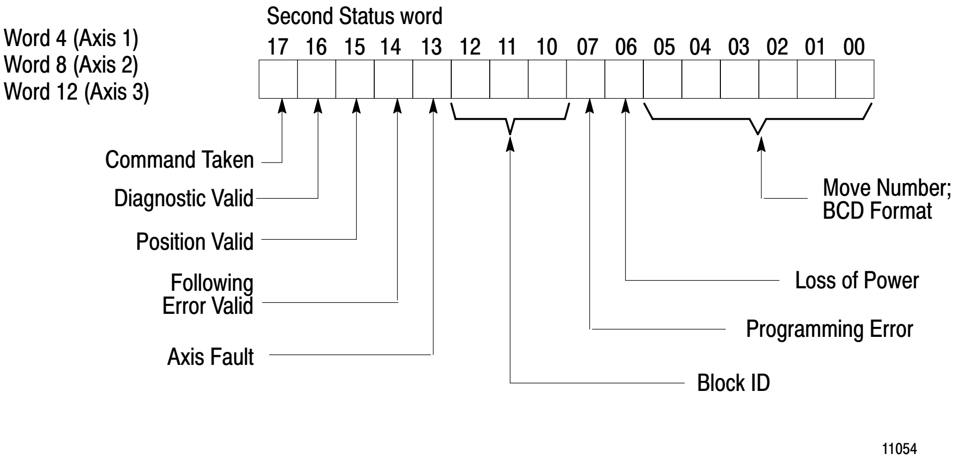
If following error equals or exceeds the excess following error value you enter in the parameter block, the 1771-M3 controller turns on this bit. Since excess following error turns on immediate stop, the immediate stop bit in the status block is also on. Additionally, if the 1771-ES expander applies feedrate reduction to an axis for which excess error is greater than the 106.25% built-in excess error value, then the feedrate reduction bit (bit 12) of the first status word for the axis is on. If, however, the excess error point you enter is less than 106.25%, then the feedrate reduction bit is not on.

Second Status Word

The second status word (Figure 7.5) identifies the active moveset and move as well as providing additional status bits.

⚠ **CAUTION:** The function of bits 06, 16, and 17 are different from the function of the corresponding bits for the series A servo positioning assembly. If you replace a series A assembly with a series B assembly without changing your program accordingly, you may cause unexpected results.

Figure 7.5
Second Status Word



Bit 0-5 Move Number

These bits indicate the active move within the moveset in BCD format.

Bit 6 Loss of Power

When set, this bit indicates a loss of power across one of the following sets of terminals:

- terminals 1 and 12 (input supply) of the left wiring arm
- terminals 1 and 6 (analog supply) of the right wiring arm

If this bit is on, then the immediate stop bit in the status block is on, indicating that immediate stop has been executed after detection of the loss of power.

Bit 7 Programming Error

If the 1771-M3 controller detects an illegal bit combination, such as a non-BCD value where one is expected, or an illegal bit combination in the command block, it turns on the programming error bit.

When this bit is on, bits 10 thru 12 of this status word provide a code to identify the block containing the programming error.

When you detect that this bit is on, you may want to turn on bit 11 of axis control word 2 in the command block to display diagnostic status in the 3rd and 4th status words for the axis.

Bits 12, 11, 10 Block ID

These bits are the block ID of the moveset block currently being executed, unless the diagnostic valid bit (bit 6) is on. When the programming error bit is on, bits 10 thru 12 indicate the block in which the error was detected:

ID (Bits 12, 11, 10)	Block
000	Parameter
001	Axis 1 Odd Moveset
010	Axis 2 Odd Moveset
011	Axis 3 Odd Moveset
100	Axis 1 Even Moveset
101	Axis 2 Even Moveset
110	Axis 3 Even Moveset
111	Command

Bit 13 Axis Fault

The 1771-M3 controller turns on this bit when communication between it and the 1771-ES expander is lost.

Bit 14 Following Error Valid

This bit is on if the next two status block words for this axis contain axis following error.

Bit 15 Position Valid

This bit is on if the next two status block words for this axis contain axis position.

If the axis position value exceeds the maximum allowable value (999.9999 in or 19999.999 mm), the servo positioning assembly turns off both the position valid and following error valid bits (bits 15 and 14), and sets the position value in the status block at the maximum value.

Bit 16 Diagnostic Valid

When you turn on the select diagnostic bit of axis control word 2 of the command block, this bit goes on to indicate that the position (or following error) words in the status block contain diagnostic information.

Bit 17 Command Taken

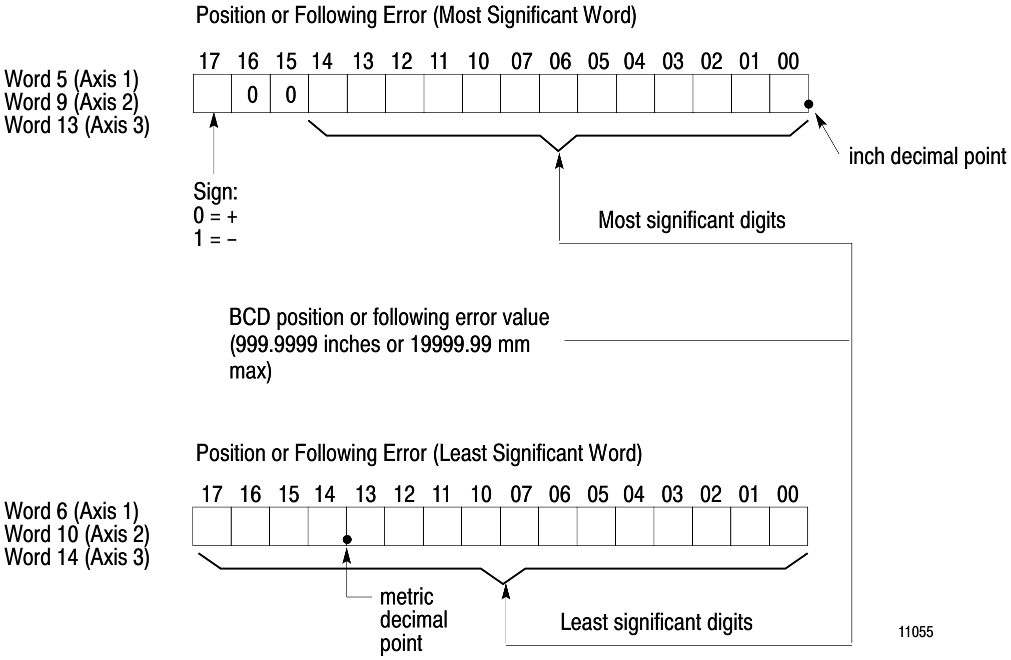
When you turn on the new-parameter, moveset override, offset preset, or get-new-preset-value bit in the command block, this bit goes on to indicate that the command has been taken. When you detect this bit to be on, you can turn off the command-block bit.

Position/Following-Error/Diagnostic Words

The 3rd and 4th status words for an axis provide either current axis position, following error, or diagnostic information. You can select which status to display by controlling the state of bits 11 and 15 of the axis control word 2 of the command block (refer to Figure 7.44 and its associated text for more information on Axis Control Word 2).

Turn off bits 11 and 15 to display the current axis position as shown in Figure 7.6. The maximum value is 999.9999 inch or 19999.999 mm. If the axis exceeds the maximum, it displays the maximum, and the position-valid bit goes off.

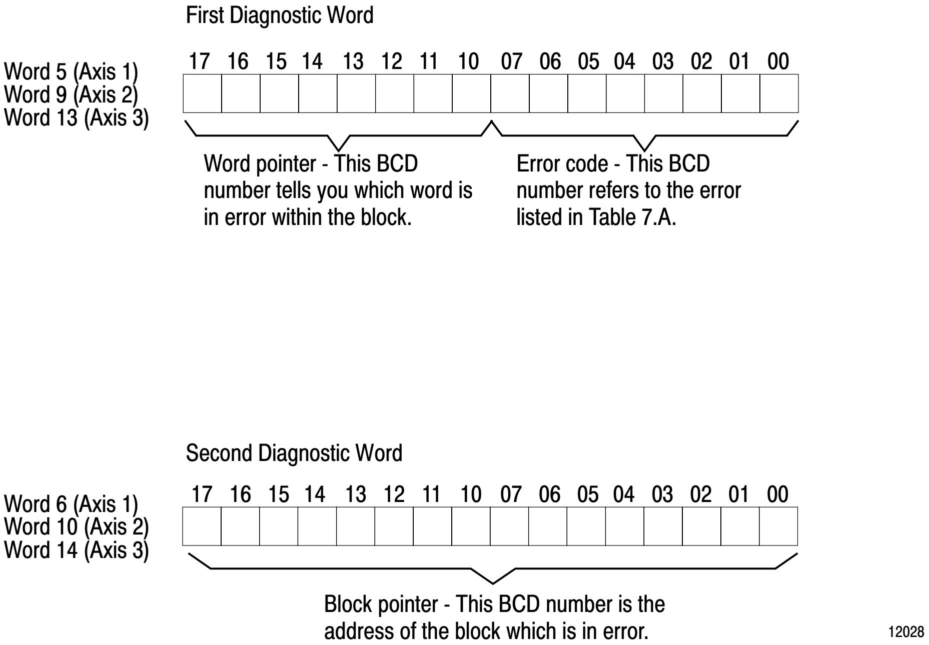
Figure 7.6
Position/ Following error/ Diagnostic Words - with Position or Following error Selected



Turn off bit 11 and turn on bit 15 to display the following error as shown in Figure 7.6. The maximum value is 999.9999 inch or 19999.999 mm. If the axis exceeds the maximum, it displays the maximum.

Turn on bit 11 to display the diagnostic status as shown in Figure 7.7.

Figure 7.7
Position/Following-Error/Diagnositc Words with Diagnostic Selected



Word pointer - This BCD number tells you which word is in error within the block.

Error code - This BCD number refers to the error listed in Table 7.A.

Block pointer - This BCD number is the address of the block which is in error.

Also, this diagnostic status displays automatically when the 1771-M3 controller detects an error in the parameter block immediately after power-up or an invalid ID in a command block. The diagnostic status displays automatically in that case because the error prevents your selecting it through the command block.

The second diagnostic word is the block pointer. The block pointer is a BCD number that indicates the starting address of the block in error. The 1771-M3 controller gets these block pointers you enter into the parameter block or the moveset block.

The high byte (bit 10 thru 17) of the first diagnostic word is the word pointer. The word pointer is a BCD number (1 thru 64) that indicates which word is in error within the block.

The low byte (bits 00 thru 10) of the first diagnostic word is the error code. The error code is a BCD number that references the errors listed in table 7.A.

Use the block pointer and word pointer to identify the location of the problem. Then use the error code to determine the nature of the problem.

Table 7.A
Diagnostic Code Definitions

Code	Definition
01	Invalid block identifier.
02	Non-BCD number entered.
03	Invalid bit setting unused bits must be zero.
04	MS “metric only” bit set in inch format.
05	Overflow: Converted data is too large for internal registers.
06	Can only change feedback multiplier from a power-up rest.
07	Invalid “axes used” programmed.
08	Invalid write block address points.
09	Invalid feedback resolution (<0.00001 in. or 0.0001 mm).
10	Invalid feedback multiplier bit setting.
11	(Counts per rev) x (feedback mult) x (encoder lines mult)>32767 decimal.
12	D/A voltage too small for selected rapid rate.
13	Initial gain too small for selected rapid rate.
14	Rapid rate entered exceeds 250 kHz maximum input frequency.
15	Rapid rate entered exceeds 1/2 revolution of encoder/2.4ms.
16	Programmed velocity >rapid rate.
17	Invalid velocity exponent programmed.
18	Entered speed is too small for selected feedback resolution.
19	Accel velocity or decel value is too small for selected feedback resolution
20	Not as many valid SMCWs as there were moves declared in the MCW.
21	Local parameters or run at velocity not allowed for a preset or dwell.
22	Invalid preset position (must be an absolute position).
23	Invalid dwell time (must be≥20ms).
24	Escape move block can only have 1 move declared.
25	Invalid escape move block; only moveset blocks identified in the parameter block can be escape move blocks.
26	Cannot program a preset or dwell as an escape move.
27	A valid next-moveset pointer could not be found.
28	Command results in overflow of offset accumulator.
29	Attempted context switch with dual meaning bits on.
30	Attempted context switch while axis is commanding motion.
31	Manual mode only bit(s) on while in auto mode.

Code	Definition
32	Invalid motion command bit combination or command not allowed.
33	Invalid command (cannot process new parameters, preset, or offset commands while axis is in motion).
34	Attempted switch to auto mode before first marker is found.

Parameter Block

Through the parameter block you specify axis parameters such as software travel limits, home position value, servo gain; and rapid traverse rate. You specify these parameters for each axis individually (Figure 7.8).

Figure 7.8
Parameter Block - Showing Word Assignments

1	Parameter Block Control Word	Fixed Overhead
2	Parameter Block Pointer	
3	Command Block Pointer	
4	Moveset Block Pointer - Axis 1	
5	Moveset Block Pointer - Axis 2	
6	Moveset Block Pointer - Axis 3	
7	Feedback Resolution	Parameters for Axis 1
8	Encoder Lines	
9	Feedback Mult., Encoder Lines Mult., Initial Gain	
10	Gain Break Speed	
11	In-Position Band/Gain Reduction Factor	
12	Rapid Traverse Rate	
13	High Jog Rate	
14	Low Jog Rate	
15	% Excess Following Error, +D/A Vlotage	
16	% Excess Following Error, -D/A Voltage	
17	Home Position (MS)	
18	Home Position (LS)	
19	Global Accel/Decel Rates	
20	Decel Step Rate	
21	+Software Travel Limit	
22	-Software Travel Limit	
23	Backlash Take-up	
24	Offset	
25	FE Reduction, Tach Conversion Factor	
26 . . . 44	Words 26-44 specify same parameters as words 7-25 but for Axis 2. (Values may be different).	Parameters for Axis 2
45 . . . 63	Words 45-63 specify same parameters as words 7-25, but for Axis 3. (Values may be different).	Parameters for Axis 3

The size of the parameter block you must provide depends on the number of axes:

Number of Axes	Size of Parameter Block
1	25 words
2	44 words
3	63 words

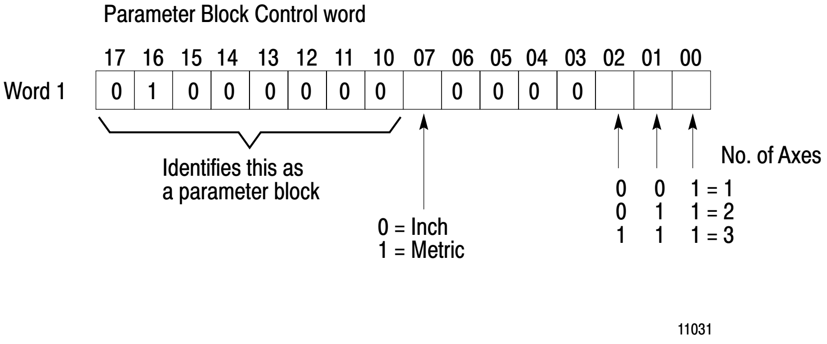
Your program must transfer the parameter block at power-up. After that, the 1771-M3 controller calls for your program to send the parameter block again only when you issue a new parameter or reset command.

Parameter Control Word

The parameter control word (Figure 7.9):

- identifies the block as the parameter lock (bits 10-17)
- specifies the units as either inch or metric (bit 7)
- identifies the number of axes in the system (bit 0,1, and 2)

Figure 7.9
Parameter Block Control Word



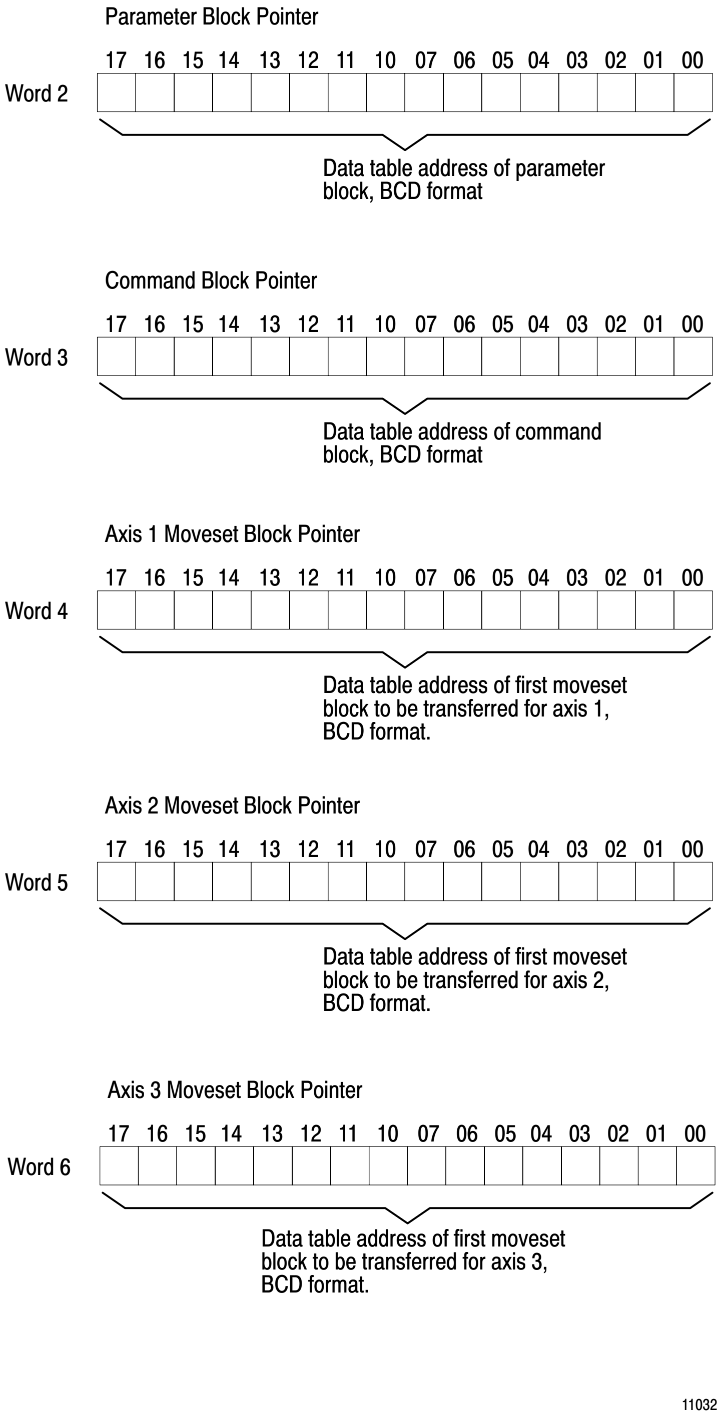
Parameter Block Control word

	17	16	15	14	13	12	11	10	07	06	05	04	03	02	01	00
Word 1	0	1	0	0	0	0	0	0		0	0	0	0			

Address Pointers

Words 2 through 5 specify the starting addresses of the parameter, command, and first moveset block for each axis (Figure 7.10).

Figure 7.10
Address Pointer Words



Data table address of parameter block, BCD format

Data table address of command block, BCD format

Data table address of first moveset block to be transferred for axis 1, BCD format.

Data table address of first moveset block to be transferred for axis 2, BCD format.

Data table address of first moveset block to be transferred for axis 3, BCD format.

Important: The address pointer value you enter in each of these words must be a **BCD** value other than **000**.

For each axis, include only the address of the first moveset block in the parameter block. Include address pointers to subsequent movesets in the blocks that precede them. If you program an escape move, you must enter its address in the parameter block moveset address pointer word. You must do this because the escape move must be the first moveset transferred to the 1771-M3 controller, even though it is not the first moveset normally executed.

Feedback Resolution

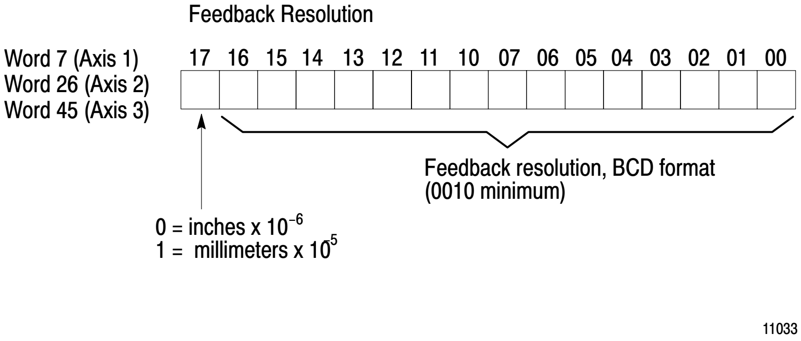
Feedback resolution is the smallest unit of axis motion that can be distinguished by the servo positioning assembly. That is, it is the distance the axis moves per feedback increment.

Enter the value of feedback resolution in the feedback resolution word of the parameter block (Figure 7.11). As described in chapter 3, feedback resolution is determined by the number of encoder lines, the feedback multiplier, and leadscrew pitch:

$$\text{Feedback Resolution} = \frac{\text{Axis Displacement per Encoder Rev.}}{\text{(Encoder Lines) (Feedback Multiplier)}}$$

If the system has no gearing, the axis displacement per revolution is the same as the leadscrew pitch or lead.

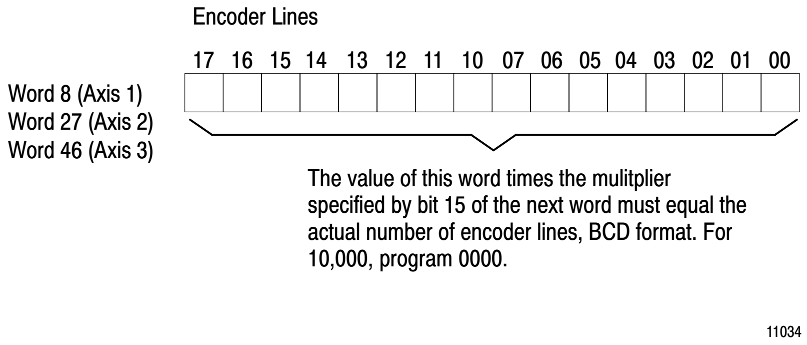
Figure 7.11
Feedback Resolution Word



Encoder Lines

This word specifies the number of encoder lines per encoder revolution (Figure 7.12).

Figure 7.12
Encoder Lines Word



The value of this word times the mulitplier specified by bit 15 of the next word must equal the actual number of encoder lines, BCD format. For 10,000, program 0000.

The value of this word times the encoder lines multiplier specified by bit 15 of the **next** higher word, must equal the actual number of lines on the encoder.

You can enter values up to 10,000 with the x1 multiplier. Entering zero (0000) specifies 10,000 lines. You can enter higher values by using the x4 multiplier.

For example, if your encoder has 12,000 lines, you can enter 3000 in the encoder-lines word, and turn on bit 15 of the next word to indicate x4: 3,000 x 4 = 12,000 lines.

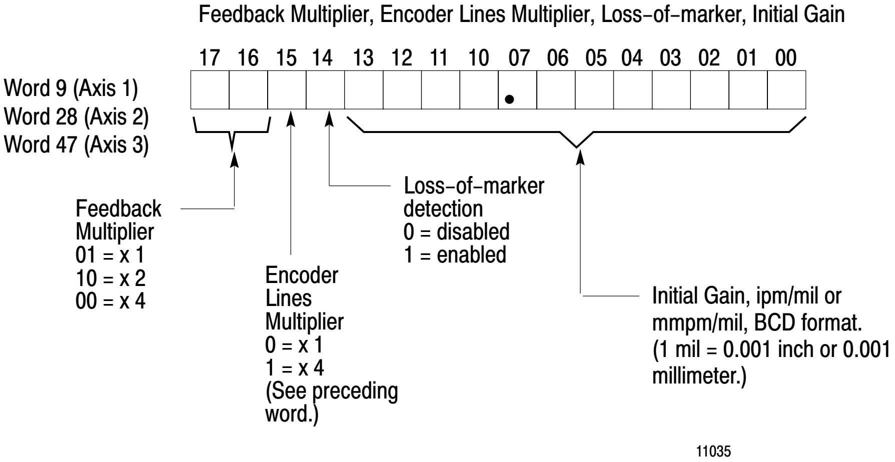
The status block indicates a programming error after transfer of the parameter block if:

(Encoder Lines) x (Feedback Multiplier) x (Encoder Lines Multiplier) > 32,767

Initial-Gain/Multipliers

Bits 0 thru 13 of this word (Figure 7.13) specify the servo gain for this axis at speeds below the gain break speed specified in word 10. You must enter gain values in BCD format, from 0.01 to 9.99 ipm/mil or from 0.01 to 9.99 mmpm/mil. (A mil is 0.001 inch or 0.001 mm.)

Figure 7.13
Feedrate Multiplier, Encoder Lines Multiplier, and Initial Gain Word



Servo gain is the ratio of axis speed to following error:

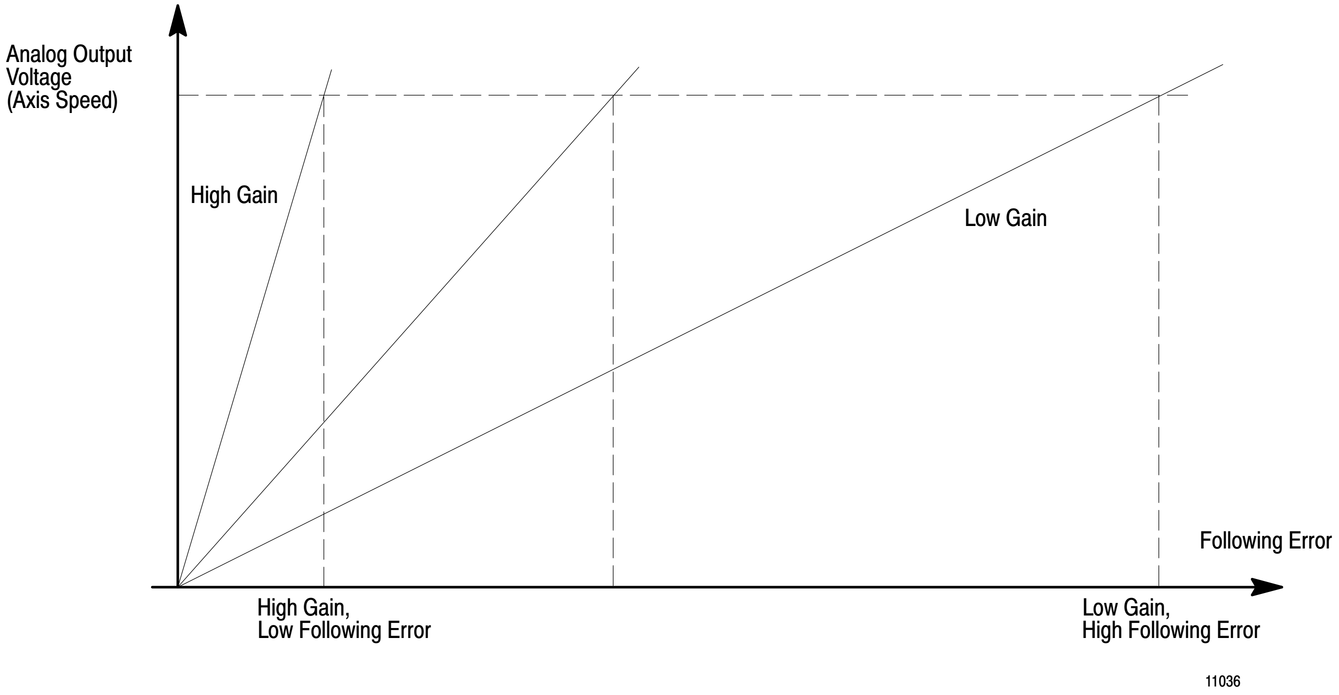
$$\text{Gain} = \frac{\text{Axis Speed}}{\text{Following Error}}$$

Following error is the difference between the axis position commanded by the servo expander and the actual axis position indicated by encoder feedback.

Servo gain affects axis response to positioning commands from the 1771-ES expander module. Figure 7.14 shows how different gain values affect system responsiveness. The horizontal axis represents following error. The vertical axis represents analog output voltage. Since analog output voltage is directly proportional to axis speed, you can use the vertical axis to represent either variable.

If gain is relatively high, following error will be relatively small, because the system will be more sensitive to changes in following error. If gain is low, following error becomes relatively larger, because the system is not as responsive to changes in following error. Choose a gain value to match the capability of your axis drives, motors, and mechanics, and provide adequate system response.

Figure 7.14
Following Error Vs. Speed for Various Gains



Parameter block values for gain and in-position band must provide a stable system and maintain desired positioning accuracy. If gain is too high, the axis may overshoot programmed endpoints and oscillate, or “hunt,” about them. If gain is too low, the axis may stop before it is within the desired in-position band. You can increase in-position band, but this decreases positioning accuracy.

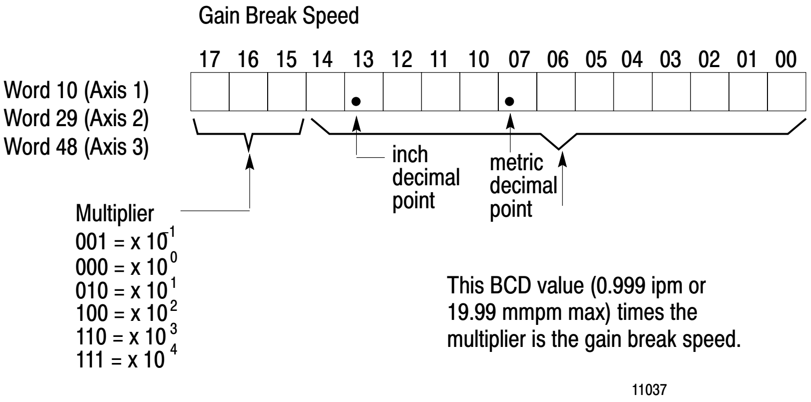
Use bit 15 of this word to select the encoder lines multiplier. This encoder lines multiplier you select, times the encoder lines value you select in the previous word, must match the number of lines per revolution of the encoder.

Use bits 16 and 17 of this word to select the feedback multiplier. The feedback multiplier you select affects the value you must enter for the feedback resolution word.

Gain Break Speed

At axis speed below the gain break value you enter into the gain break word (Figure 7.15), servo gain is the initial gain programmed in the preceding word.

Figure 7.15
Gain Break Speed Word



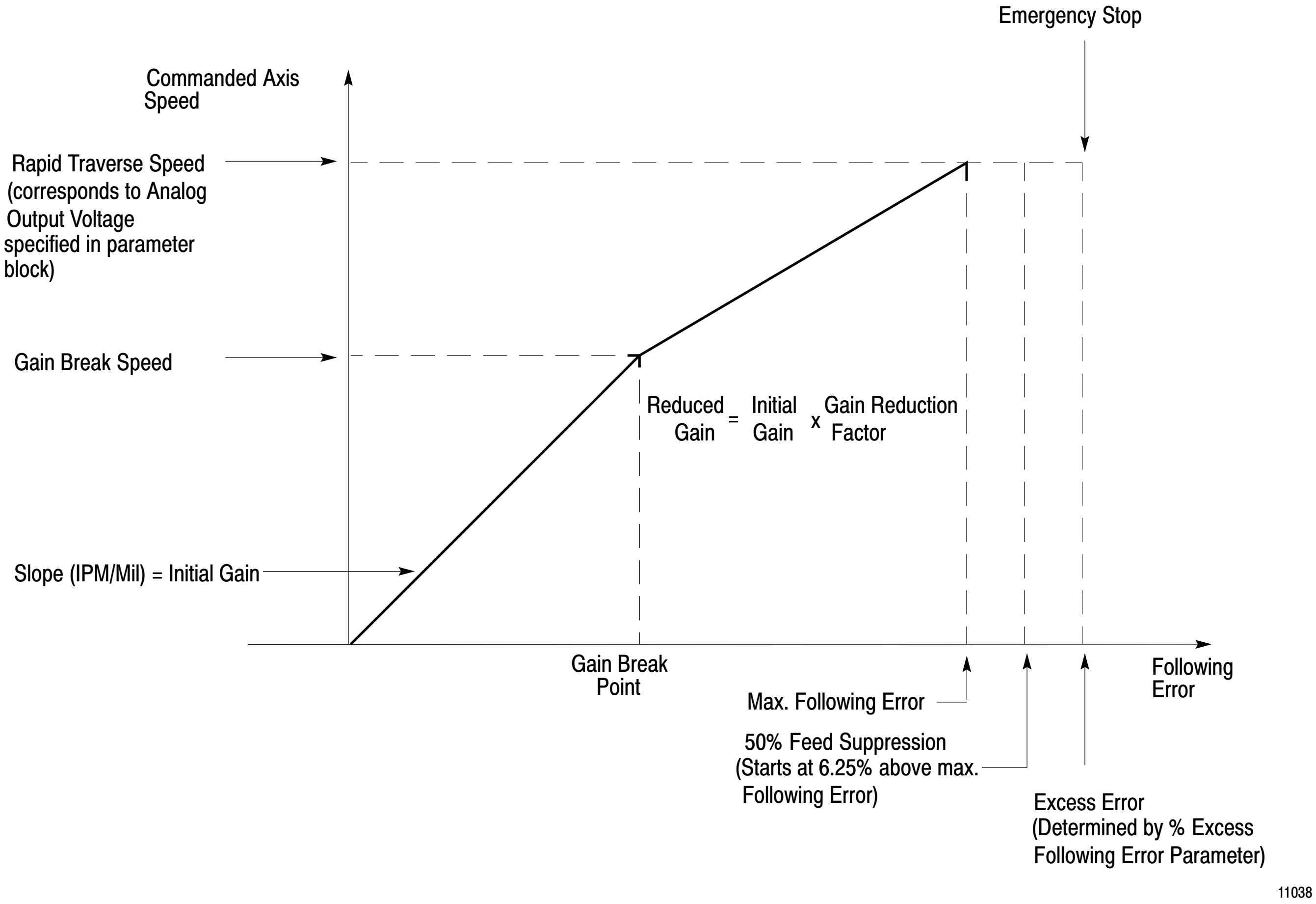
At speeds equal to and above the gain break value you enter into this word, the servo positioning assembly reduces servo gain by the gain reduction factor specified in the next word of the parameter block.

The gain break plot of Figure 7.16 illustrates the concept of gain break.

Typically, gain at axis speeds below the gain break velocity is relatively high to allow precise axis positioning. Reduced gain at axis speeds above gain break velocity allows for better stability at higher axis speeds.

Gain break velocity can be no greater than rapid traverse rate. If there is to be no gain break point for an axis, program the rapid traverse speed in this word.

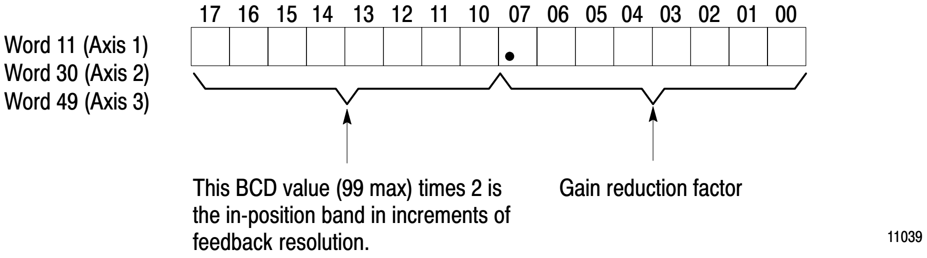
Figure 7.16
Gain Break Plot



Gain-Reduction Factor

Bits 0-7 of the in-position-band and gain-break-factor word (Figure 7.17) specify the gain reduction factor. The initial gain of the axis is multiplied by this factor to obtain the reduced gain value for axis speeds above the gain break speed.

Figure 7.17
In-position Band, Gain Reduction Factor Word



$$\text{Gain Reduction Factor} = \frac{\text{Reduced Gain}}{\text{Initial Gain}}$$

For example, if the initial gain is one, and you want the reduced gain to be 0.5, program .50 as the value for gain reduction factor.

$$\text{Gain Reduction Factor} = \frac{.5}{1} = .50$$

The gain reduction factor must be less than 1.0. If you program zero, the system gain for any axis speed will be the initial gain. If gain break velocity is zero and you program a non-zero gain reduction factor, system gain for any axis speed is the initial gain times the gain reduction factor. Enter the gain reduction factor in BCD format.

In-Position Band

The size of the in-position band is measured in increments of the feedback resolution of the axis. Program a 2-digit BCD value that is half the desired in-position band in bits 10-17 of the in-position-band and gain-break-factor word (Figure 7.17). If you program zero as the in-position band parameter value, the servo positioning assembly automatically makes the active in-position band ±2 feedback increments.

The 1771-M3 controller turns on the in-position bit when the done bit is on in the status block and the axis is within the in-position band.

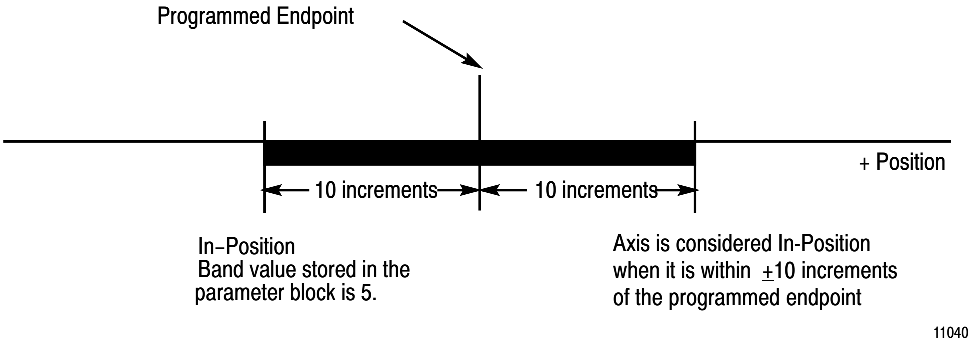
The axis must be in-position before the following actions can take place:

- Manual mode commands are not executed unless the in-position bit is on.
- In auto mode, the start command is not executed unless the in-position bit in the status block is on.

- When the direction of axis motion is reversed, the in-position bit in the status block must be on before axis motion in the reverse direction can occur.

Note that the value you enter for the in-position band is actually half the desired in-position band value. For example, if the in-position band value you enter is 5, then the servo positioning assembly considers the axis in-position if it is within ±10 increments of feedback resolution of the programmed endpoint. Figure 7.18 illustrates the concept of in-position band.

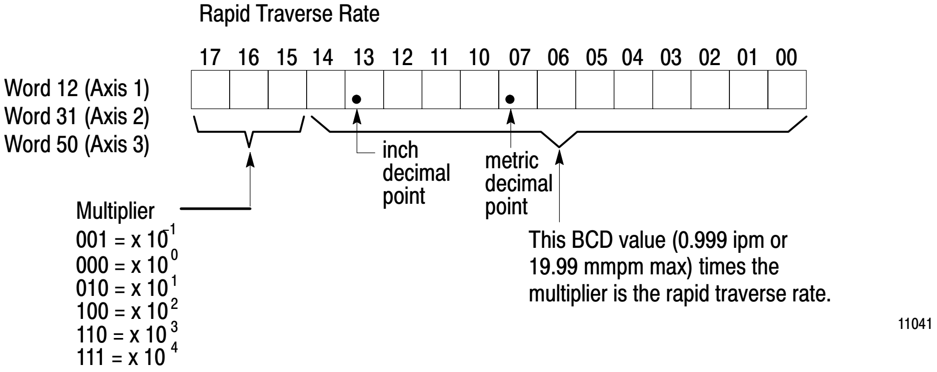
Figure 7.18
In-position Band Example



Rapid Traverse Rate

The rapid traverse rate you enter (Figure 7.19) is the highest feedrate the axis can attain. It is associated with open travel of the axis. The servo positioning assembly uses this rate for the go home operation and for moves that you program to use the global feedrate.

Figure 7.19
Rapid Traverse Rate Word



This BCD value (0.999 ipm or 19.99 mmpm max) times the multiplier is the rapid traverse rate.

The rapid traverse rate is limited by several parameters. The servo positioning assembly detects a programming error and inhibits axis motion when you enter a rapid traverse rate that violates any of the following formulas:

$$RR < \frac{12{,}500}{1.28} \text{ x } FR \text{ x } FM \text{ x } EL \qquad \text{(Table 7.A, code 15)}$$

$$RR < \frac{1.5 \text{ x } 10^7}{1.28} \text{ x } FR \text{ x } FM \qquad \text{(Table 7.A, code 14)}$$

$$RR < 4 \text{ x } 10^6 \text{ x } FR \text{ x } D/A \qquad \text{(Table 7.A, code 12)}$$

$$RR < \frac{6.5 \text{ x } 10^7}{1.28} \text{ x } FR \text{ x } IG \qquad \text{(Table 7.A, code 13)}$$

Where the following are parameters you enter:

RR = rapid traverse rate

FR = feedback resolution

FM = feedback multiplier (1, 2, or 4)

EL = encoder lines per revolution

D/A = maximum D/A voltage

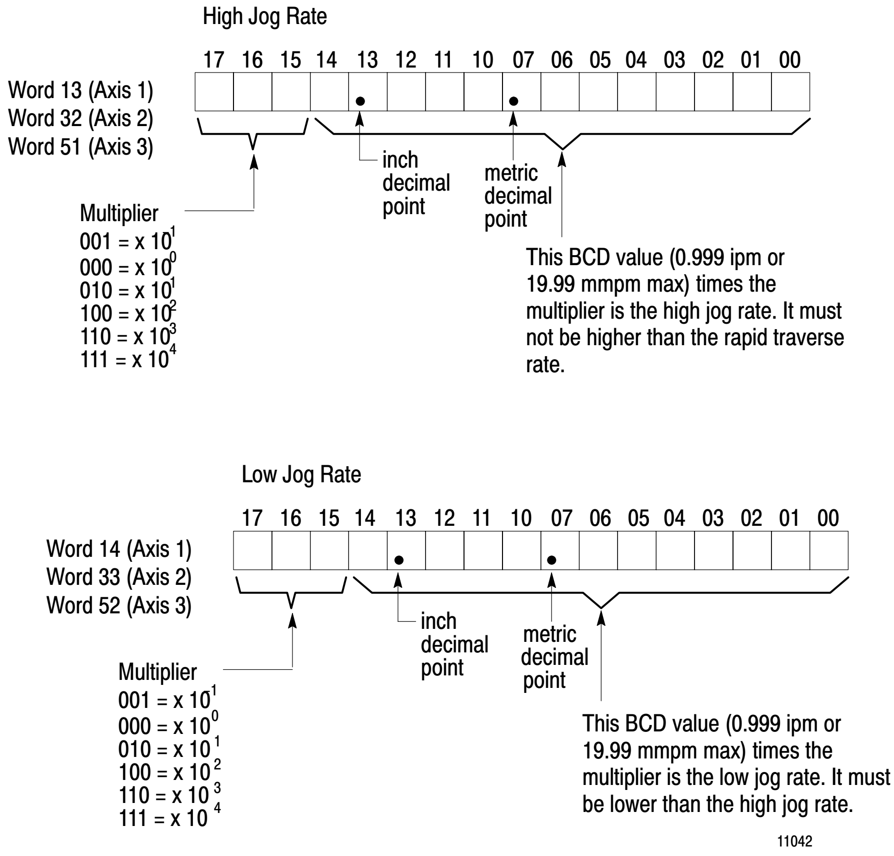
IG = initial gain

These formulas include an allowance for a 127% feedrate override factor. These formulas apply to both ipm and mmpm.

Jog Rate

The high and log jog rate words (Figure 7.20) specify the speeds at which you can jog the axis. You can jog the axis only in manual mode. Program the values in BCD format. The operator can select jog speed (high or low) by controlling the jog rate select bit in the command block.

Figure 7.20
Jog Rate Words

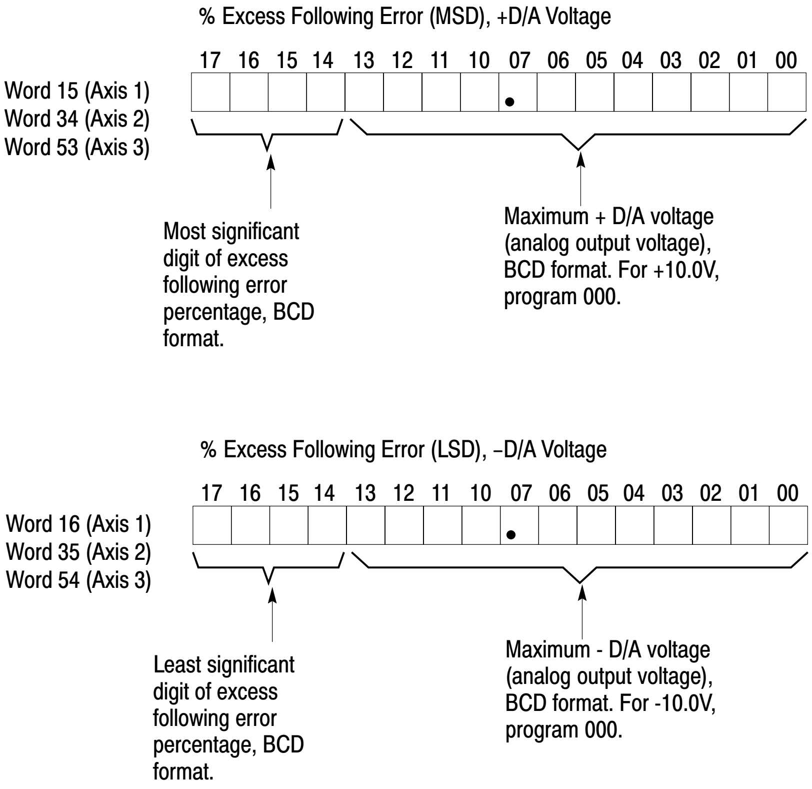


% Excess Following Error

The % excess following error parameter is a 2-digit BCD number that the 1771-ES expander interprets as a percentage above the following error allowed at the rapid traverse rate. Programmable excess following error values can thus range from 0 through 99. Program the most significant digit in bits 14 through 17 of the first word, and the least significant digits in bits 14 through 17 of the second word (Figure 7.21).

This parameter specifies maximum allowable axis following error. When the following error reaches the maximum value permitted as specified by the % excess following error parameter, the servo positioning assembly stops axis motion by commanding immediate stop (Figure 7.16).

Figure 7.21
Excess Following Error, D/A Voltage Words


% Excess Following Error (MSD), +D/A Voltage
17 16 15 14 13 12 11 10 07 06 05 04 03 02 01 00
Word 15 (Axis 1)
Word 34 (Axis 2)
Word 53 (Axis 3)
Most significant digit of excess following error percentage, BCD format.
Maximum + D/A voltage (analog output voltage), BCD format. For +10.0V, program 000.
% Excess Following Error (LSD), –D/A Voltage
17 16 15 14 13 12 11 10 07 06 05 04 03 02 01 00
Word 16 (Axis 1)
Word 35 (Axis 2)
Word 54 (Axis 3)
Least significant digit of excess following error percentage, BCD format.
Maximum - D/A voltage (analog output voltage), BCD format. For -10.0V, program 000.

Excess following error percent should be greater than or equal to 6%. The value entered here is the percent above rapid traverse following error at which Emergency Stop is to occur.

11043

Feedrate Reduction

When axis following error reaches 106.25% of rapid traverse following error, the servo positioning assembly automatically reduces feedrate by 50% of the feedrate value. This feedrate reduction provides an opportunity for following error to decrease. Feedrate returns to the programmed value when following error is reduced to less than or equal to 106.25% of rapid traverse following error and the current move is completed.

Note that if the excess following error value you enter is less than or equal to 6%, the axis executes immediate stop before following error reaches 106.25% of rapid traverse following error.

+ and - D/A Voltage

Bits 00 thru 13 in words 15 and 16 (34 and 35, 53 and 54) specify the maximum servo output voltage that is available to command rapid traverse feedrate in the positive and negative directions (Figure 7.21).

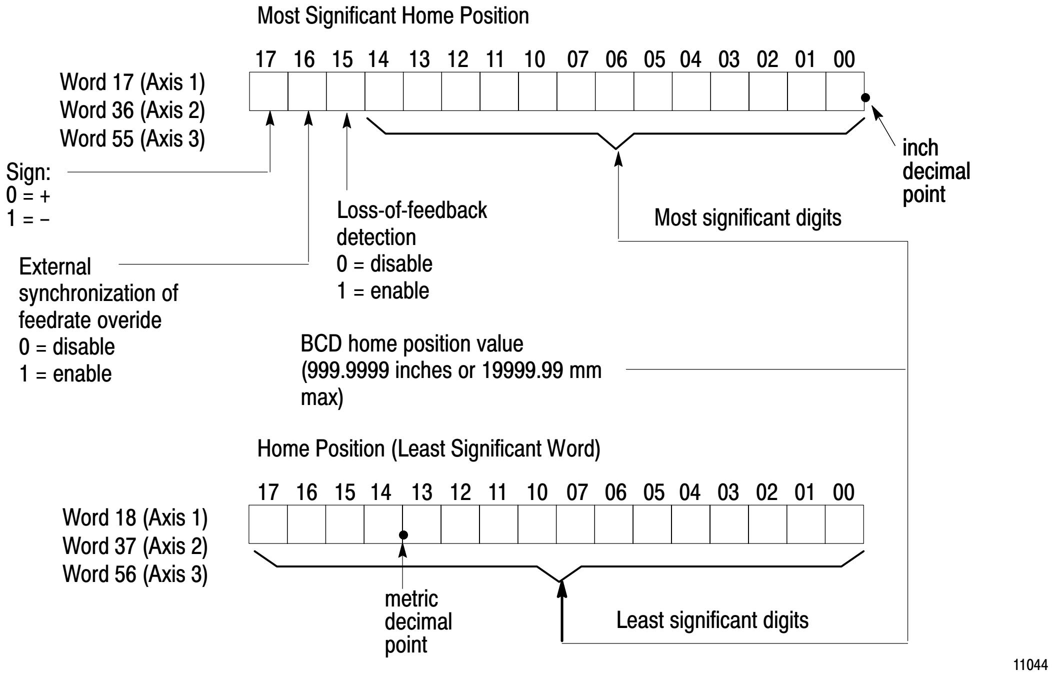
Enter values for these parameters in BCD format in the range of 0.01V to 9.99V. Programming 0 causes the D/A voltage value to default to 10V.

Initially set them to the maximum value the servo drive will accept. The plus and minus D/A voltage values needn't be equal. You can enter them as different values to compensate for directional differences in drive performance during axis integration (chapter 9).

Home Position Value

Words 17 and 18 (36 and 37, 55 and 56) specify the axis home position value (Figure 7.22). Bits 0 through 14 of the first word contain the most significant digits. Bit 17 of the first word specifies the sign of the home position value. When the servo positioning assembly performs a search home or initialize home operation, it sets the axis position register to the value you enter for this parameter.

Figure 7.22
Home Position Words



Loss-of-Feedback Detection Enable

Turn bit 15 off until you complete the open-loop and closed-loop axis integration procedures (chapter 9).

Then turn on bit 15 of the most significant home position word to enable the loss-of-feedback detection feature of the 1771-ES expander.

WARNING: Once you have completed the axis integration procedures, **never** turn this bit **off**. Without loss-of-feedback detection, if encoder or tachometer feedback is lost, unexpected axis motion can occur, resulting in damage to equipment and/or injury to personnel.

External Synchronization of Feedrate Override

Turn on bit 16 of the most significant home position word to have the 1771-ES expander recognize the feedrate override enable input (Figure 7.22).

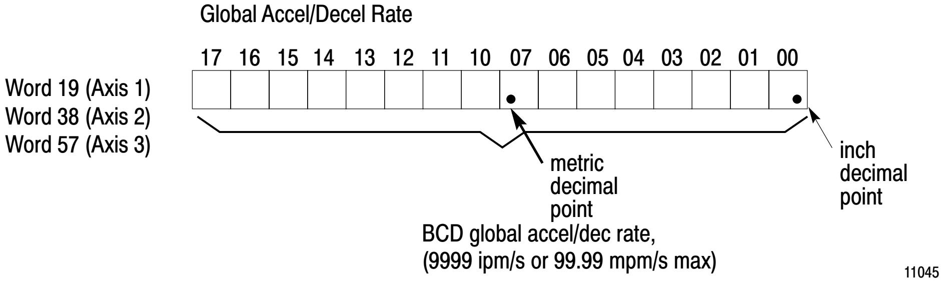
With this bit off, you change the feedrate by the percentage you enter in the command block when you enable feedrate override in the command block. However, you change the feedrate for a particular move only if you had enabled feedrate override in the move block.

With the bit on, you must still enable feedrate override in the command block and move block before feedrate changes. However, the 1771-ES expander will not change the feedrate until you close the feedrate-override-enable input. This allows you to synchronize the feedrate override of several axes.

Global Accel/Decel Rate

Word 19 (38, 57) specifies the acceleration and deceleration rate the servo positioning assembly uses for all jogs and for moves in movesets for which you do not enter local acceleration and deceleration rates. It is also the deceleration value used when executing a slide stop during manual mode operation of an axis or when you issue a reset command during axis motion. (Figure 7.23).

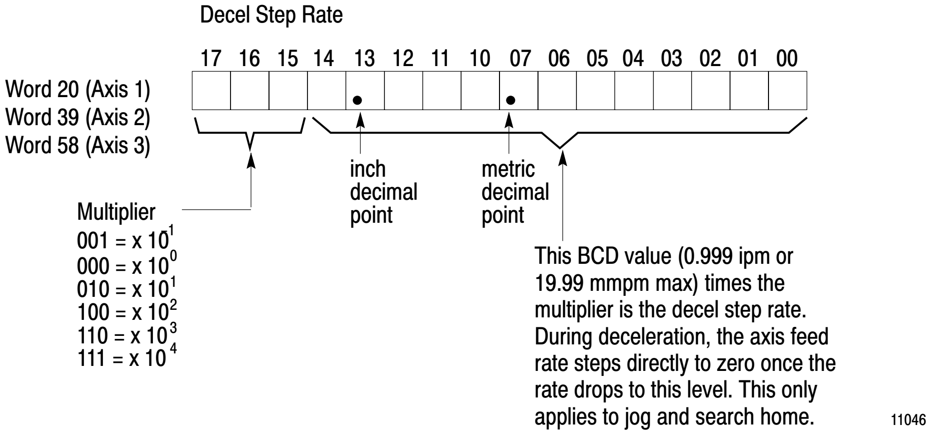
Figure 7.23
Global Accel/Decel Rate Word



Decel Step Rate

Word 20 (39, 58) specifies the deceleration step rate (Figure 7.24). This parameter applies to deceleration of axis motion when the servo positioning assembly is in the manual mode. At axis feed rates equal to or less than that specified by this word, the servo positioning assembly ignores the programmed deceleration rate, and steps axis feed rate directly to zero. This parameter is not effective in auto mode, and applies only to jog and search home operations.

Figure 7.24
Deceleration Step Rate Word



This BCD value (0.999 ipm or 19.99 mmpm max) times the multiplier is the decel step rate. During deceleration, the axis feed rate steps directly to zero once the rate drops to this level. This only applies to jog and search home.

Software Travel Limits

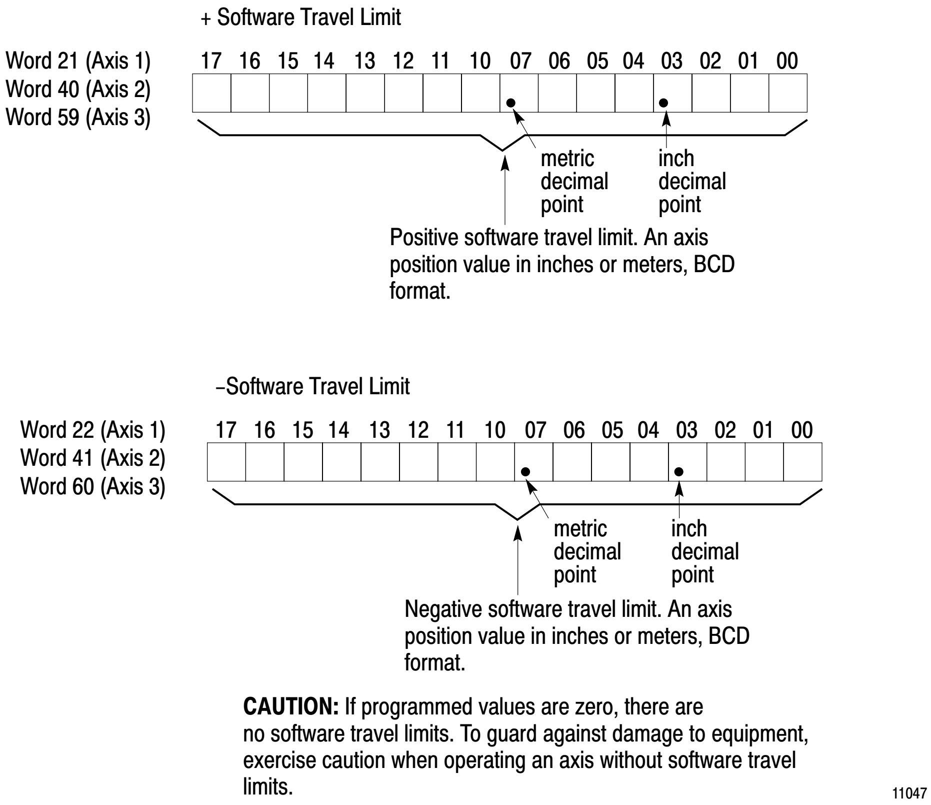
Words 21 and 22 (40 and 41, 59 and 60) specify the axis position values of the axis software travel limits (Figure 7.25). When a programmed move calls for the axis to move beyond a software travel limit, if there is time, the servo positioning assembly automatically decelerates the axis at the programmed rate for the current move so that it stops at or before the software travel limit position.

If there is no time to decelerate the axis before the limit is exceeded, the servo positioning assembly executes an immediate stop. This could occur following a continuous move because the next move starts with the feedrate of the previous move rather than zero.

Software travel limit values are axis position values. Note that if zero is the programmed travel limit value, there is no software travel limit. The absolute positions of the software travel limit vary with changes in axis position value due to preset or home commands. In addition to the software travel limit you must have extreme axis overtravel limit switches wired in the master control relay circuit.

CAUTION: If programmed values for software travel limits are zero, there are no software travel limits. To guard against damage to equipment, use caution when operating an axis without software travel limits.

Figure 7.25
Software Travel Limit Words



Positive software travel limit. An axis position value in inches or meters, BCD format.

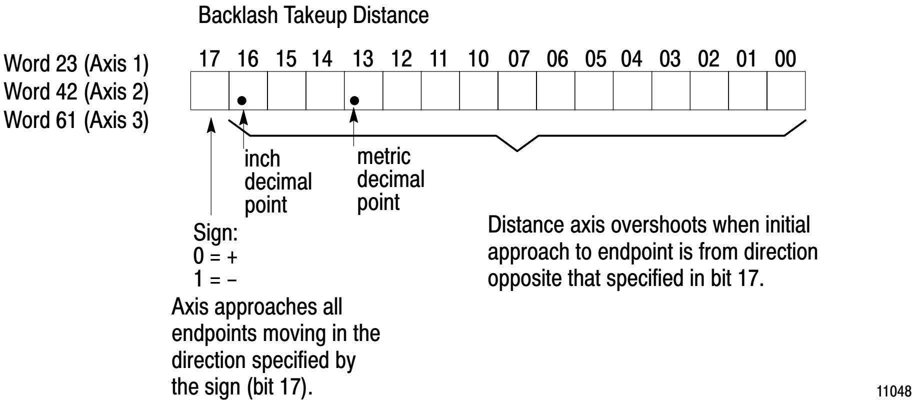
Negative software travel limit. An axis position value in inches or meters, BCD format.

CAUTION: If programmed values are zero, there are no software travel limits. To guard against damage to equipment, exercise caution when operating an axis without software travel limits.

Backlash Takeup

Backlash takeup helps minimize axis positioning inaccuracy caused by mechanical play in the axis positioning system. Word 23 (42, 61) is the backlash takeup word (Figure 7.26).

Figure 7.26
Backlash Takeup Word


Backlash Takeup Distance
Word 23 (Axis 1)
Word 42 (Axis 2)
Word 61 (Axis 3)
17 16 15 14 13 12 11 10 07 06 05 04 03 02 01 00
inch decimal point
metric decimal point
Sign:
0 = +
1 = –
Axis approaches all endpoints moving in the direction specified by the sign (bit 17).
Distance axis overshoots when initial approach to endpoint is from direction opposite that specified in bit 17.
11048

In this word, bit 17 specifies the direction the axis is to move in approaching all programmed endpoints. When the axis approaches an endpoint at which it is to stop while moving in the specified direction, it simply stops at the endpoint. If the axis approaches the endpoint from the opposite direction,it overshoots the endpoint by the amount you specify in bits 00 thru 16, then returns to the endpoint from the opposite direction.

Consider the example of your entering +.0010 in the backlash takeup word:

- If the axis is moving in the positive direction it stops at the programmed endpoint without overshoot.
- If the axis is moving in the negative direction, it overshoots the endpoint by 0.001 inch, then returns to the programmed endpoint.

Backlash takeup affects only halt moves that command the axis to stop at a move endpoint. For blended moves, backlash takeup has no effect. Also, backlash takeup is active only in auto mode. Backlash takeup has no effect on axis motion in the manual mode.

This parameter has a 4-digit BCD value in the range of 0.0001 to 0.7999 inches or 0.001 to 7.999 mm.

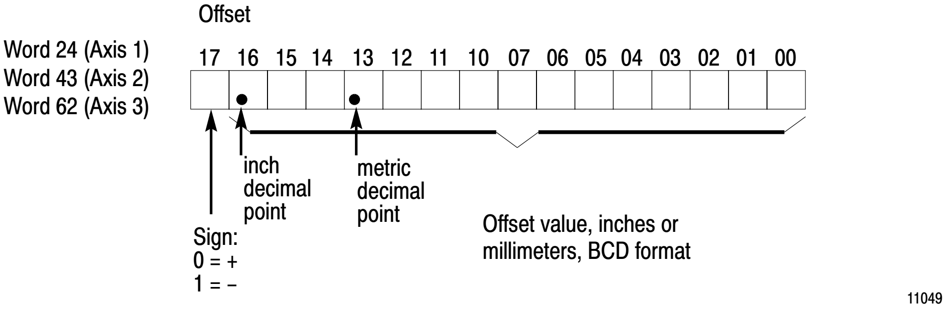
Offset

Word 24 (43, 62) specifies the value the servo positioning assembly adds to the offset accumulator when the servo positioning assembly executes one of the following:

- a position-with-offset move in a moveset
- an offset command from the command block

This parameter has a 4-digit BCD value that can be in the range ±0.0001 to ±0.7999 inches or ±0.001 to ±7.999 mm (Figure 7.27).

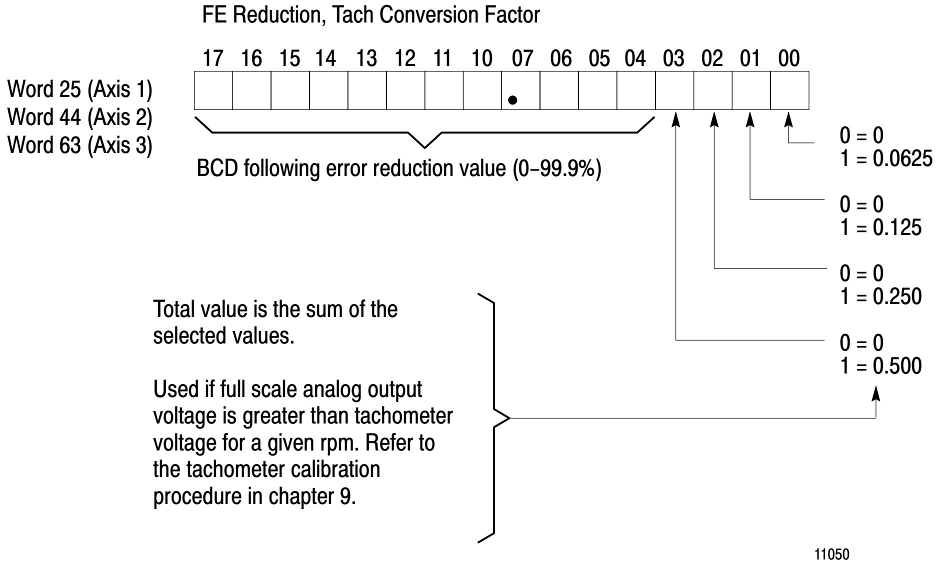
Figure 7.27
Offset Word



Tachometer Conversion Factor

The servo positioning assembly uses bits 00 thru 03 of word 25 (44, 63) for its loss-of-feedback detection feature (Figure 7.28). The tachometer calibration procedure explains how to select the value for this word.

Figure 7.28
Following Error Reduction/Tachometer Conversion Factor Word


FE Reduction, Tach Conversion Factor
17 16 15 14 13 12 11 10 07 06 05 04 03 02 01 00
Word 25 (Axis 1)
Word 44 (Axis 2)
Word 63 (Axis 3)
BCD following error reduction value (0–99.9%)
0 = 0
1 = 0.0625
0 = 0
1 = 0.125
0 = 0
1 = 0.250
0 = 0
1 = 0.500
Total value is the sum of the selected values.
Used if full scale analog output voltage is greater than tachometer voltage for a given rpm. Refer to the tachometer calibration procedure in chapter 9.
11050

Each of bits 0 thru 3 corresponds to a factor. The total factor used by loss-of-feedback detection equals the sum of the individual factors selected.

If you are not using the loss-of-feedback detection feature, or if tachometer voltage is greater than or equal to 10V, program zero for all bits of this word.

Following Error Reduction

The servo positioning assembly accepts the BCD value you enter into bits 04 thru 17 as the following error reduction value (Figure 7.28).

You can command a reduction of the following error by 0 through 99.9%. The 1771-ES expander reduces the following error through feed forwarding without increasing the positioning loop gain.

Consider an example in which you have entered an initial gain value of 1.00. With an axis speed of 10 ipm, without following error reduction, the following error would be 10 mils.

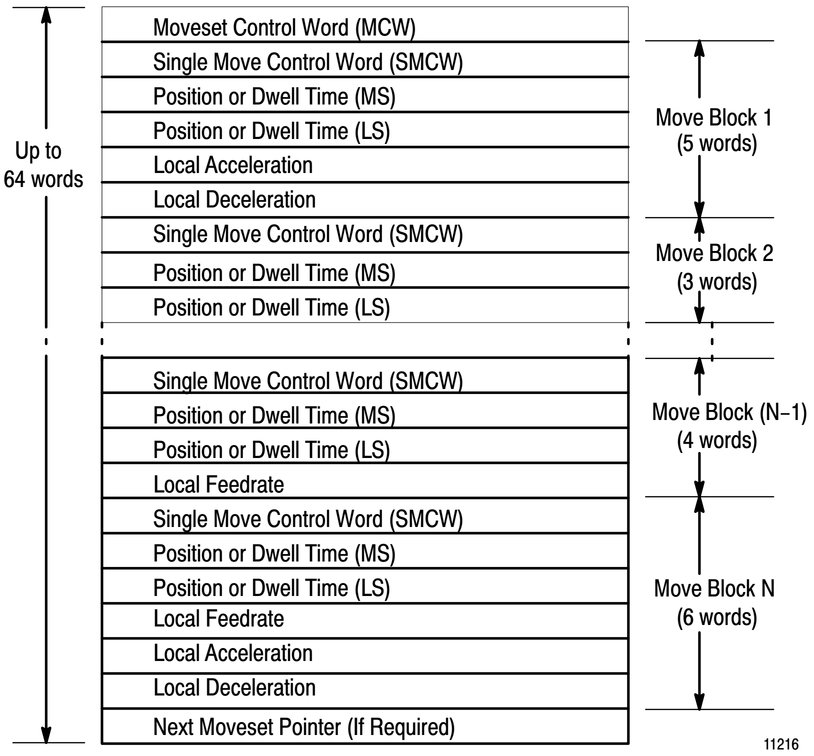
$$FE = \frac{\text{speed}}{\text{gain}} = \frac{10 \text{ ipm}}{1 \text{ ipm/mil}} = 10 \text{ mil s}$$

However, if you enter a following error reduction value of 70.0 the following error at 10 imp is reduced from 10 mils to 3 mils.

Moveset Block

A moveset block contains a number of move blocks through which it describes axis motion for a sequence of moves (Figure 7.29).

Figure 7.29
Moveset Block- Showing Word Assignments



Moveset Control Word (MCW)
Single Move Control Word (SMCW)
Position or Dwell Time (MS)
Position or Dwell Time (LS)
Local Acceleration
Local Deceleration
Single Move Control Word (SMCW)
Position or Dwell Time (MS)
Position or Dwell Time (LS)
Single Move Control Word (SMCW)
Position or Dwell Time (MS)
Position or Dwell Time (LS)
Local Feedrate
Single Move Control Word (SMCW)
Position or Dwell Time (MS)
Position or Dwell Time (LS)
Local Feedrate
Local Acceleration
Local Deceleration
Next Moveset Pointer (If Required)

Each move requires a move block of at least three words (a single move control word, and two words to define position or dwell time). A move block may have as many as six words (a single move control word, two position words, a rate word, an accel word, and a decel word). In addition, two words (moveset control word and next moveset pointer) apply to the entire moveset block. Since the moveset block may contain no more than 64 words, the largest possible number of moves a single block can describe is 21. All 21 moves would have to use global accel/decel and final rate values.

Upon request from the status block, the PC processor writes a moveset block to the 1771-M3 controller, which transfers the move blocks to the 1771-ES expander one at a time. The servo expander generates analog voltage to command axis motion as programmed.

The first word of a moveset block is the moveset control word (MCW).

Following the MCW are the move blocks. Each move block consists of a single move control word (SMCW), two position (or dwell) words, and may contain words for local feedrate, accel rate, and decel rate, depending on whether you select local or global rates by the SMCW.

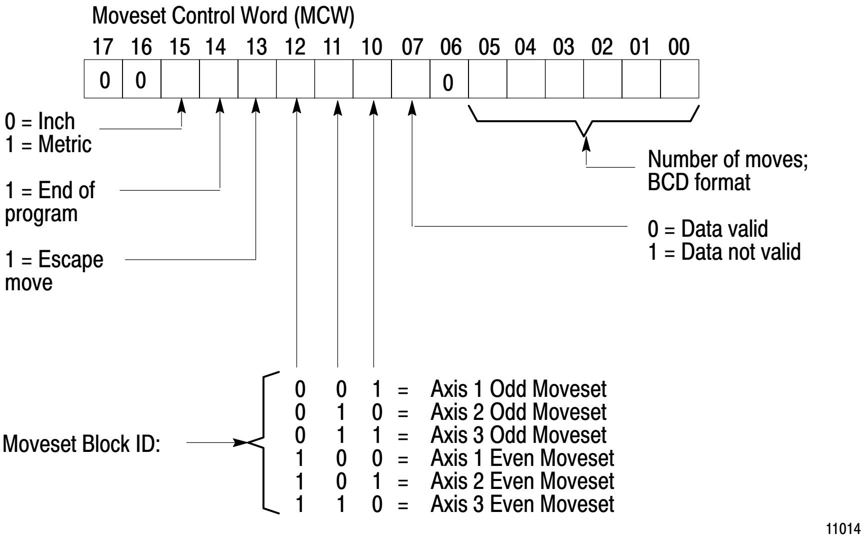
You can leave words of zeros before and after move blocks. This gives you flexibility. For example, you could remove a move block without changing the location of the other move blocks within the data table. However, you must respecify the number of moves in the moveset control word.

If the end of program bit is off, the last word in a moveset block must be an address pointer (in BCD format) to the moveset block that should be executed next.

Moveset Control Word

The MCW word identifies the block as a moveset block, indicates whether you program the axis in inch or metric units, specifies the number of moves in the moveset and whether or not the moveset defines an escape move (Figure 7.30).

Figure 7.30
Moveset Control Word (MCW)


Moveset Control Word (MCW)
17 16 15 14 13 12 11 10 07 06 05 04 03 02 01 00
0 0
0
0 = Inch
1 = Metric
1 = End of program
1 = Escape move
Number of moves; BCD format
0 = Data valid
1 = Data not valid
Moveset Block ID:
0 0 1 = Axis 1 Odd Moveset
0 1 0 = Axis 2 Odd Moveset
0 1 1 = Axis 3 Odd Moveset
1 0 0 = Axis 1 Even Moveset
1 0 1 = Axis 2 Even Moveset
1 1 0 = Axis 3 Even Moveset
11014

Bits 0 thru 5 Number of Moves

Bits 0-5 specify the number of moves in this moveset in BCD format. Due to the maximum block size of 64 words, the maximum number of moves you can program in a moveset block is 21. If you use local rates for any moves, the maximum number of moves is reduced.

To accommodate more moves, program additional movesets.

Bit 7 Data Valid

If bit 7 is off, it tells the 1771-M3 controller that data in the moveset block is valid and can be executed.

This bit can be used by the operator when making on-line moveset changes. If the bit is on when the 1771-M3 controller receives the moveset, the controller does not execute the moveset. The controller continues to request the moveset until the data valid bit is off.

Bits 12, 11, 10 Moveset ID

Bits 12, 11, 10 tell the 1771-M3 controller whether it is an odd or an even moveset block and also identifies the axis to which the moveset applies.

	Axis 1	Axis 2	Axis 3
Odd Moveset Block	001	010	011
Even Moveset Block	100	101	110

These rules apply to moveset block ID assignment (Figure 7.31):

1. You must identify the first moveset block for an axis as odd (001 for axis 1, 010 for axis 2, 011 for axis 3).

2. The next-move-set pointer of **any** moveset block may point to the first moveset block for the axis. Consequently, the first moveset can point to itself.

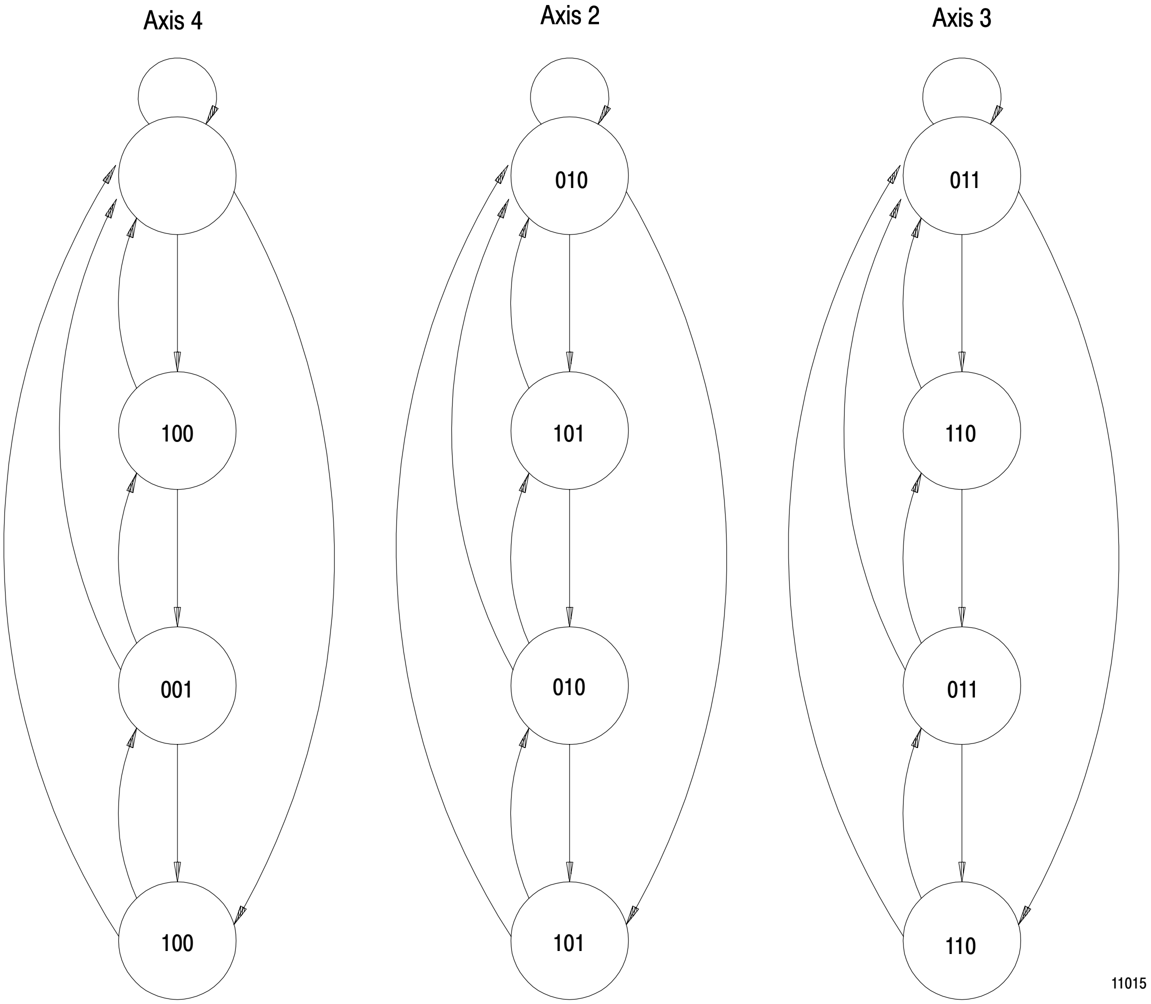
3. Except for rule 2, any moveset block with ID 001 must point to moveset blocks with ID 100, and moveset blocks with ID 100 must point to moveset blocks with ID 001.

4. Except for rule 2, moveset blocks with ID 010 must point to moveset blocks with ID 101, and moveset blocks with ID 101 must point to moveset blocks with ID 010.

5. Except for rule 2, a moveset block with ID 011 must point to a moveset block with ID 110 and a moveset block with ID 110 must point to a moveset block with ID 011.

Consequently, the movesets of a series for an axis typically alternate between even and odd IDs (Figure 7.32).

Figure 7.31
Moveset Block IDs - Showing Allowed Moveset Sequencing



Bit 13 Escape Move

If you turn on this bit, the controller identifies this as an escape moveset block.

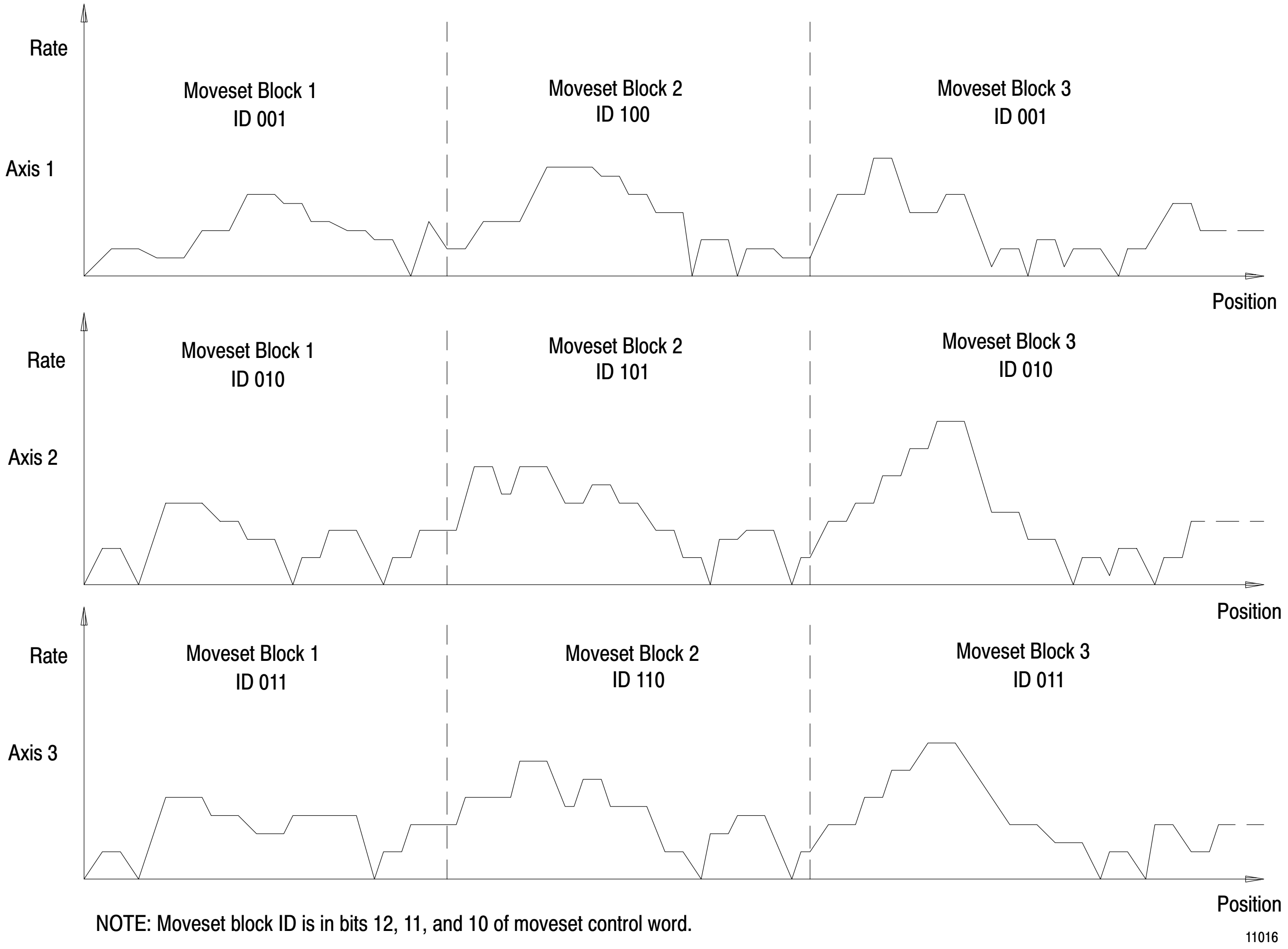
Follow these rules when programming an escape moveset block:

1. The escape moveset must be the **first** moveset transferred to the 1771-M3 controller at power up or reset. Consequently, it must have moveset ID 001 (axis 1), 010 (axis 2), or 011 (axis 3). The next moveset pointer must point to a moveset block with ID 100 (axis 1),

101 (axis 2), or 110 (axis 3) that is the moveset that you want to normally execute first.

2. The moveset that contains the escape move must contain **no** other moves. The escape moveset thus contains only one move, and a next moveset pointer that identifies the moveset that you want to normally execute first.

Figure 7.32
Moveset Profiles - Showing Alternating Moveset IDs



NOTE: Moveset block ID is in bits 12, 11, and 10 of moveset control word.

3. The escape moveset block cannot be executed as part of a sequence of moveset blocks.

The PC processor transfers the escape moveset block to the 1771-M3 controller at power up and after a reset or escape command is executed. When it receives the escape moveset block, the 1771-M3 controller immediately transfer the escape moveset block to the 1771-ES expander, which stores it on-board. The escape move is not executed unless you issue an escape command via the command block.

When you issue the escape command, the 1771-ES expander executes the escape move previously stored on-board. If another move is being executed when you issue the escape command, the 1771-ES expander stops executing the current move and starts to execute the escape move.

Immediately after you issue the escape command, the 1771-M3 controller requests transfer of the moveset for that axis controller requests transfer of the moveset for that axis identified in the parameter block. If this is still an escape block, the 1771-M3 controller transfers it to the 1771-ES expander for storage and requests transfer of another moveset from the PC processor. The moveset requested is the one specified by the next moveset pointer of the escape moveset. If you issue a command such as start, begin, or next-move after execution of the escape moveset is completed, the axis begins executing the moveset that was transferred.

You can issue a slide stop, hardware stop, or software stop command to stop an escape move in progress.

When the escape move bit is on, the 1771-M3 controller ignores the setting of the end of program bit in the MCW (bit 14).

Bit 14 End of Program

If you turn on bit 14, the 1771-M3 controller stops move execution after it completes execution of the current moveset block. The 1771-M3 controller does **not** look for a next moveset pointer. You must issue a begin command to repeat execution of the profile. If you issue a start or next-move command, the 1771-M3 controller turns on the insufficient data bit in the status block.

The insufficient data bit goes on also if you issue the escape command when no escape move has been programmed. However, an escape command causes a request for the moveset block identified in the parameter block for the axis.

If the end-of-program bit is off, a start command can start execution of the moveset specified in the next moveset pointer.

Bit 15 Inch/Metric

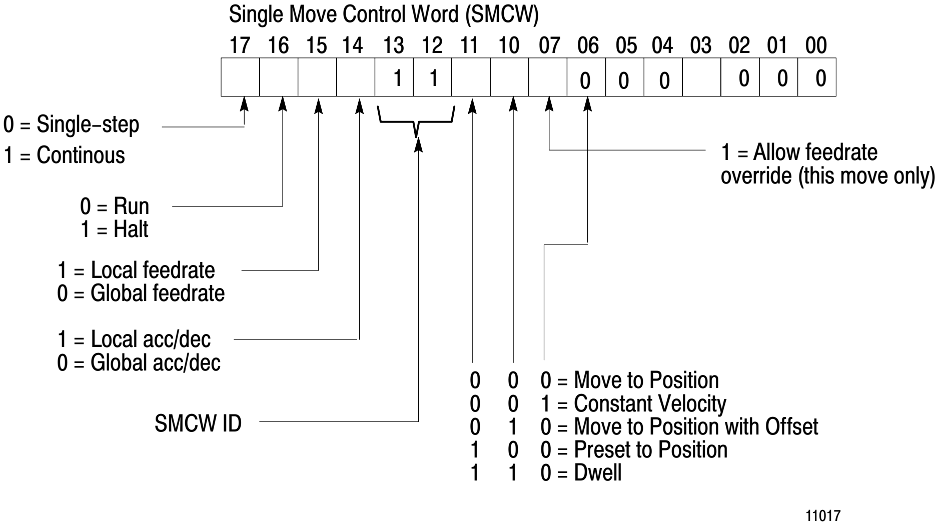
Bit 15 determines how the servo positioning assembly interprets positions, feedrates, and accel/decel rates entered in the moveset block.

If you want the units to be:	Then set bit 15 to:
inch	0
mm	1

Single Move Control Word (SMCW)

For each move in a moveset, you must enter a single move control word (SMCW). The SMCW specifies move characteristics and determines how the servo positioning assembly interprets the words that follow it (Figure 7.33).

Figure 7.33
Single Move Control Word (SMCW)



Bit 7 Allow Feedrate Override

With bit 7 on, you can enable feedrate override for this move by turning on the axis feedrate override enable bit in the command block. The programmed feedrate for this move is modified by the feedrate override value programmed in the command block. Feedrate override value is typically an operator controlled input.

With bit 7 off, this move executes at the programmed feedrate regardless of the command block.

This bit affects only the move controlled by the SMCW. For example, if feedrate override is to apply to several moves, you must turn on bit 7 of the SMCW for each of those moves.

Bits 11, 10, 06 Set to 000 Move to Position

Turn off bit 11, 10, and 6 to generate a move to position. The axis will move to the position specified by the sum of the value in the position words and the value in the offset accumulator register.

Bits 11, 10, 06 Set to 001 Constant Velocity

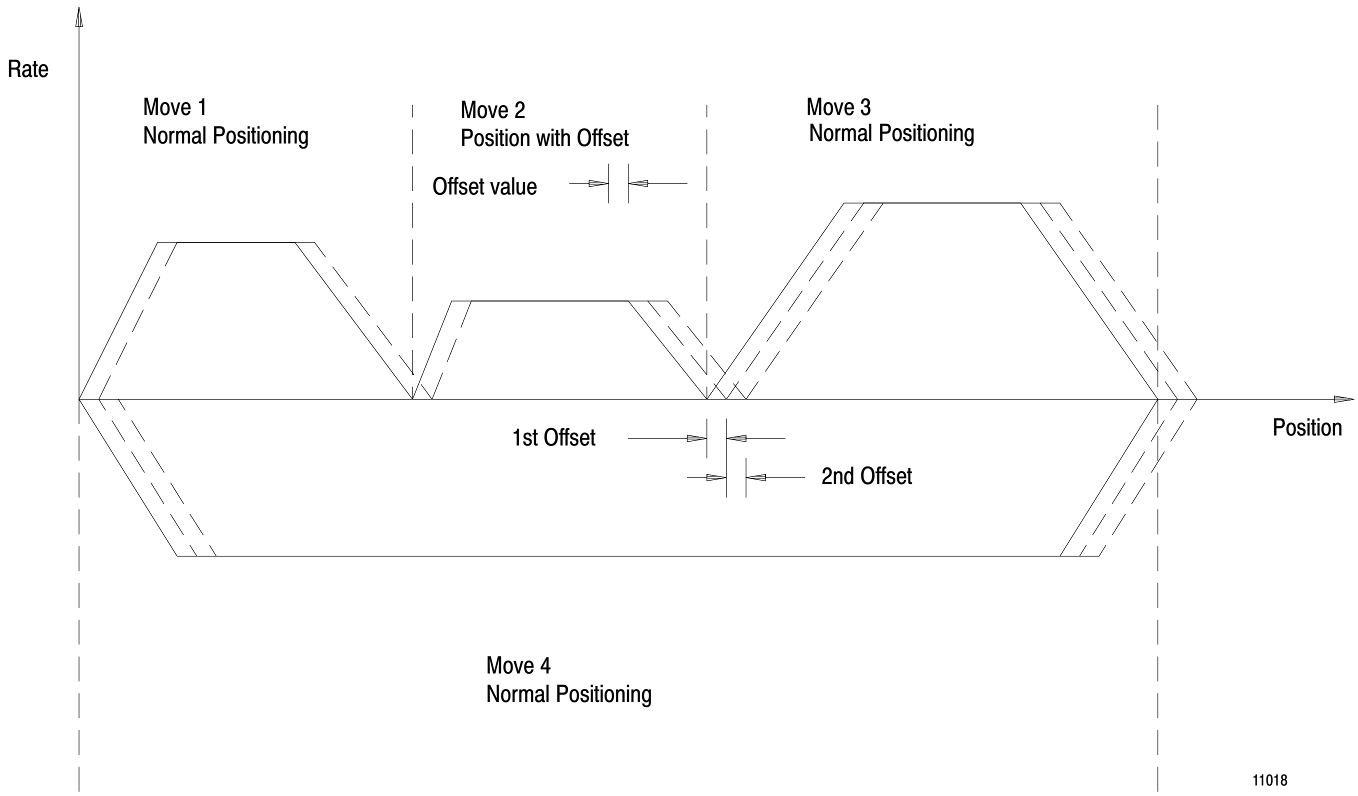
Turn bit 11 off, bit 10 off, and bit 6 on to generate a constant velocity move. This command clears the position register to zero before moving the axis to a position specified by the value in the position words. By repeatedly generating continuous constant velocity moves you can cause uninterrupted motion that you could apply to conveyors, winders, coilers and spindle-type controls.

Bits 11, 10, 06 Set to 010 Move to Position with Offset

Turn bit 11 off, bit 10 on, and bit 06 off to generate a move to position with offset. This command first adds the offset value in the parameter block to the offset accumulator register. Then the axis moves to the position specified by the sum of the value in the position words and the value in the offset accumulator register.

The endpoints of all following moves are also modified by the offset value, which remains in the offset accumulator. If you execute the moveset again, the offset accumulator value increments again when the move-to-position-with-offset command executes. Figure 7.34 shows the effect of position with offset.

Figure 7.34
Moveset Profile - Showing Position with Offset



Changing the offset value in the parameter block has no effect on the two move blocks already stored on board the 1771-ES expander module, but only on subsequent move blocks.

Bits 11, 10, 06 Set to 100 Preset to Position

Turn bit 11 on, bit 10 off, and bit 06 off to generate a preset to position. This command clears the accumulator register to zero and sets the position register to the value specified in the position words. Instead of generating axis motion, this command redefines the present axis position.

Bits 11, 10, 06 Set to 110 Dwell

Turn bit 11 on, bit 10 on, and bit 06 off to generate a dwell. This command inhibits the 1771-ES expander from executing the move block that follows for the amount of time specified by the value in the position or dwell words.

Bits 12 and 13 SMCW ID

Bits 12 and 13 must be on to identify the word as an SMCW. If they are not on, the status block indicates programming error 20.

Bit 14 Accel/Decel

If you turn on bit 14, you must include local acceleration and deceleration rate words in this move block. Note that if you select local accel/decel, you enter separate acceleration and deceleration values for the move.

If you turn off this bit, the servo positioning assembly uses the global accel/decel rate in the parameter block. Note that if you select global accel/decel, then the move uses the single value programmed in the parameter block for both acceleration and deceleration rates. For dwells and presets, this bit must be off; if not, error code 21 appears in the status block.

Use of local accel and decel rates lowers the number of moves you can program in a moveset block (Table 7.B).

You must select either the global accel/decel value or a local acceleration and deceleration value for each move, even though you may anticipate that the moves will execute in the continuous run mode so that some programmed accelerations and decelerations do not take place. This is because the slide stop command in the command block uses the deceleration rate of the current move that is executing.

Bit 15 Feedrate

If you turn on bit 15, you must enter a local feedrate word in the move block.

If you turn off this bit, the move is executed at the global feedrate in the parameter block (rapid traverse rate), and this move block must not include a local feedrate word.

For dwells and presets, turn off this bit or the programming error bit will go on in the status block.

Use of local feedrates lowers the number of moves you can program in a moveset block (Table 7.B).

Table 7.B
Moveset Programming

	Word	No. of Words	Comments
Move Block	MCW	1	
Move 1	SMCW #1 Bit 14 = 1 Local Rate Bit 15 = 1 Local Acc/Dec	6	Separate words required to program Local Feedrate. Local Accel, Local Decel.
	Position (MSW)		
	Position (LSW)		
	Local Feedrate		
	Local Acc		
	Local Dec		
Move 2	SMCW#2 Bit 14 = 1 Local Rate Bit 15 = 0 Global Acc/Dec	4	Separate word required to program Local Feedrate. No moveset words required to program Accel and Decel--values programmed in parameter block are used.
	Position (MSW)		
	Position (LSW)		
	Local Feedrate		
Move 3	SMCW #3 Bit 14 =0 Global Rate Bit 15 = 0 Global Acc/Dec	3	No Moveset words required to program Feedrate Accel and Decel--values programmed in parameter block are used. No Moveset words required to program Feedrate, Accel or Decel--values programmed in parameter block are used.
	Position (MSW)		
	Position (LSW)		
	SMCW #4 Bit 14 = 0 Global Rate Bit 15 = 1 Local Acc/Dec		Separate words required to program Local Accel and Local Decel.

	Word	No. of Words	Comments
Move 4	Position (MSW)	5	No Moveset words required to program Feedrate--value programmed inn parameter block used.
	Position (LSW)		
	Local Acc		
	Local Dec		
	Next Moveset Pointer	1	This word is optional

Bit 16 Run/Halt

Bit 16 determines whether or not the move block after this one will execute automatically without a motion command (start, next move, or begin) from the command block. Bit 16 functions in conjunction with bit 17, single-step/continuous.

- **Run** - If you turn off the run/halt bit, the servo positioning assembly begins execution of the next move block immediately after completing the current move block without waiting for a start or begin command. If bit 17 is on (continuous), the moves are blended smoothly. If bit 17 is reset (single step), the axis decelerates to zero velocity before starting the next move.
- **Halt** - If you turn on the run/halt bit, the servo positioning assembly decelerates the axis to zero velocity at the programmed endpoint, then waits for a motion command before executing the next move block, regardless of the setting of bit 17 (single-step or continuous).

The run/halt bit applies to all move blocks, including presets and dwells.

Figure 7.35 shows a moveset profile with examples of various bit-16/bit-17 combinations. Table 7.C explains the various combinations with reference to the figure.

Figure 7.35
Moveset Profile - Showing Single-step/Continuous and Run/Halt Combinations (Refer to Table 7.C)

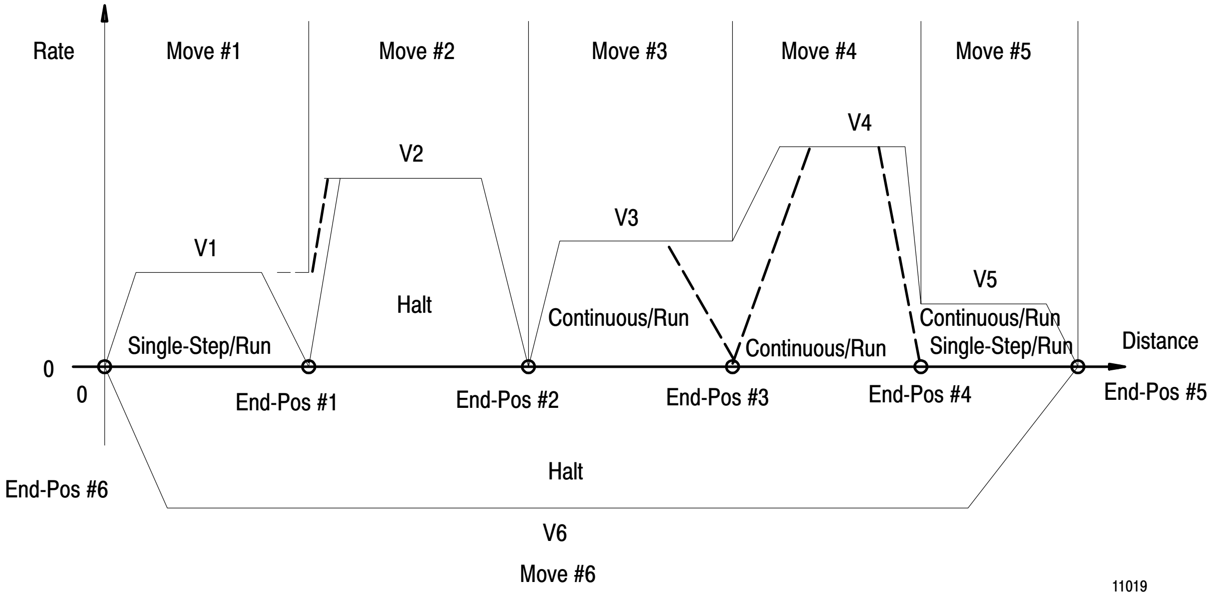


Table 7.C
Single-Step/Continuous and Run/Halt Combinations

Move	Bit 17 0=SS 1=Cont	Bit 16 0=Run 1=Halt	Command Combination	Axis Motion
1	0	0	Single-Step Run	1. Axis decelerates at the programmed rate for move 1. It reaches zero velocity at the programmed endpoint for move 1. 2. Immediately after stopping, the axis begins executing move 2. 3. Single Step requires the axis to be at zero velocity at the programmed endpoint. 4. Run allows the axis to start executing move 2 without waiting for a go command in the motion control block. 5. Compare moves 3 and 4, continuous/run. Dashed lines on the move profile show how moves 1 and 2 would blend if move 1 were continuous/run.
2	0 1	1 or 1	Halt	1. The axis decelerates at the programmed rate for move 2. It reaches zero velocity at the programmed endpoint for move 2, then stops and waits. 2. Execution of the next move begins only after a motion command is received in the motion control block. 3. The halt command has priority over both the single step and run commands. Consequently, the axis stops at the programmed endpoint and waits for a go command. Single step or continuous commands have no effect when the halt command is programmed.
3	1	0	Continuous Run	1. The axis moves at the programmed final rate to the programmed endpoint, then immediately begins executing move 4 by accelerating to the move 9 final rate. 2. Compare move 2, single step/run and move 2, continuous/halt and move 4.
4	1	0	Continuous Run	1. The axis decelerates from the programmed final rate for move 4 so it reaches the final rate for move 5 at the endpoint programmed for move 4. The axis then immediately begins execution of move 5 at the programmed final rate. 2. Compare move 3, for which the final rate of the next move is higher than the final rate for move 3. 3. Dashed lines from move 4 or move 5 on figure show execution if move 4 were single step/run or halt.
5	1 0	0 or 0	Continuous Run or Single-Step Run	1. Because move 6 is in the opposite direction, the axis decelerates at the rate programmed for move 5 so it reaches zero velocity at the programmed endpoint. Execution of move 6 begins immediately. In this case, execution is the same for both continuous and single step commands.
6 or	0	1 or	Halt	1. The axis decelerates at the programmed rate so that it stops at the programmed endpoint.
	1	1		2. When on bit 16 has priority over bit 17. Single-step or Continuous has no effect on this move. 3. No further axis motion occurs until a command is received in the motion control block.

Bit 17 Single-Step/Continuous

Bit 17 determines how the 1771-ES expander executes moves when the run/halt bit is off (run mode):

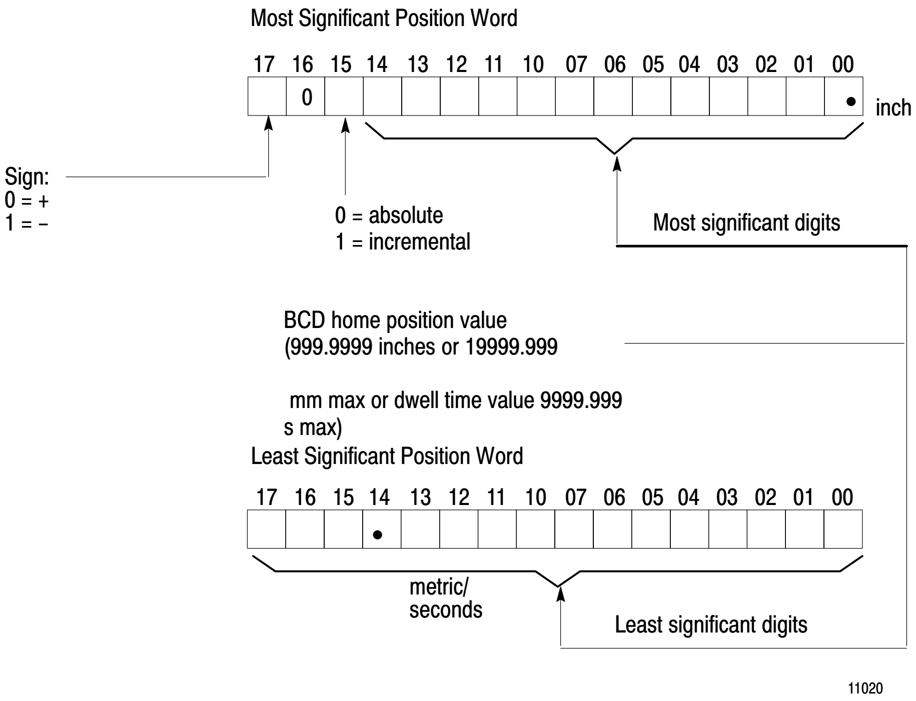
- **Single Step** - If you turn off this bit, the move ends with deceleration to zero velocity. If you turn off bit 16 (run command), the axis decelerates to zero velocity, then continues moveset execution without waiting for a motion command from the command block. If you turn on bit 16 (halt command), the servo positioning assembly ignores bit 17.
- **Continuous** - If you turn on bit 17 and turn off bit 16, the move blends smoothly with the next move. That is, the axis accelerates or decelerates from its final feedrate to the final feedrate of the next move.

Figure 7.35 shows a moveset profile with examples of various bit-16/bit 17 combinations. Table 7.C explains the various combinations with reference to the figure.

Position Words

Two position words follow the SMCW in each move block. Together, these words specify an axis motion endpoint, a position preset value, or a dwell time, depending on the move block type entered in bits 11, 10, and 6 of the SMCW (Figure 7.36).

Figure 7.36
Position/Dwell-time Words



For dwell values, these words express a 7-digit number in BCD format. The maximum programmable value is 9999.99 seconds. The minimum programmable value is 0000.020 seconds.

For inch values, these words express a 7-digit number in BCD format. The maximum programmable value is 999.9999 inches. If bit 14 of the most significant position word is on when the SMCW is programmed for inch units, the status block indicates a programming error to inform the operator that a position value greater than 999.9999 inches has been programmed.

For metric values, these words express an 8-digit number in BCD format. The maximum programmable value is 19999.999 mm.

For position values, use bit 15 of the most significant position word to specify the positioning mode:

- **Absolute** - If you turn off this bit, the position value is an axis position coordinate relative to the zero position of the axis. For example, if the axis is at position coordinate +4.5 and the position words call for motion to absolute endpoint +3.5, the servo positioning assembly moves the axis one unit in the negative direction to position coordinate +3.5.

- **Incremental** - If you turn on this bit, the position word specifies the move endpoint relative to the most recently achieved programmed endpoint. For example, if the axis is at position +4.5 and the position word value is +3.5, the axis moves +3.5 units in the positive direction to position +8.0.

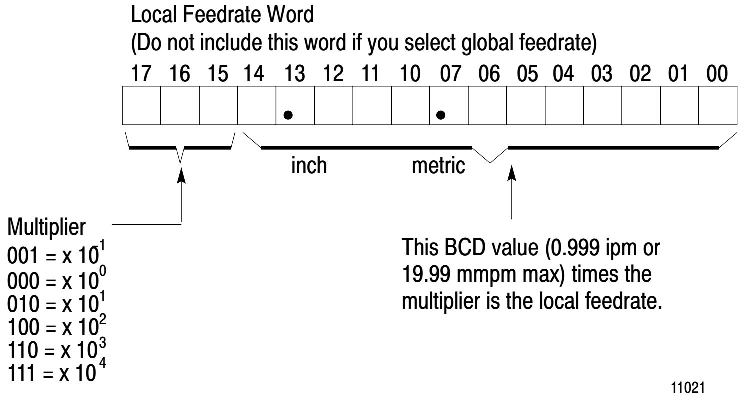
Local Feedrate Word

If you turn on the local feedrate bit in the SMCW, you must include a local feedrate word after the position words. Acceptable values depend on units (Figure 7.37):

- Inch: 0.0001 to 9990 ipm (bit 14 must be 0 for inch values)
- Metric: 0.001 to 199900 mmpm

For example, to program a local feedrate of 100 ipm, you program .100 x 10# ipm.

Figure 7.37
Local Feedrate Word



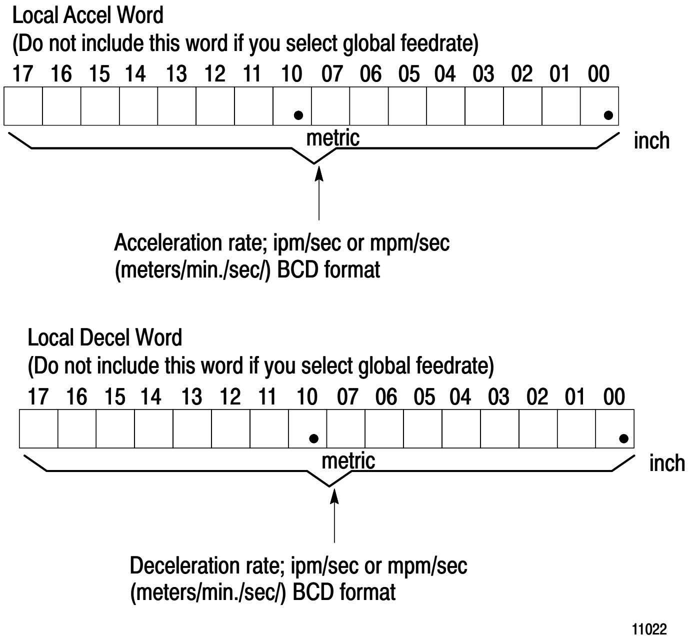
Local Accel and Decel Words

If you turn on the local accel/decel bit in the SMCW, you must include local accel and decel words at the end of the move block. These words specify acceleration and deceleration rates as 4 digit BCD values (Figure 7.38):

- Inch: maximum value is 9999 ipm/s

- Metric: maximum value is 99.99 mpm/s (Note that the units for this value are meters/minute/second, **not** millimeters/minute/second.)

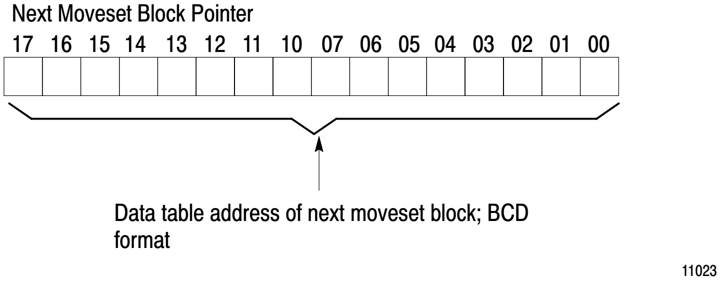
Figure 7.38
Local Accel Word and Local Decel Word


Local Accel Word
(Do not include this word if you select global feedrate)
17 16 15 14 13 12 11 10 07 06 05 04 03 02 01 00
metric
inch
Acceleration rate; ipm/sec or mpm/sec
(meters/min./sec/) BCD format
Local Decel Word
(Do not include this word if you select global feedrate)
17 16 15 14 13 12 11 10 07 06 05 04 03 02 01 00
metric
inch
Deceleration rate; ipm/sec or mpm/sec
(meters/min./sec/) BCD format
11022

Next-Moveset Pointer

The final word of the moveset block contains the PC data table address of the next moveset (Figure 7.39).

Figure 7.39
Next Moveset Block Pointer Word


Next Moveset Block Pointer
17 16 15 14 13 12 11 10 07 06 05 04 03 02 01 00
Data table address of next moveset block; BCD format
11023

You must include a next moveset pointer word if you turn off the end of program bit in the moveset control word.

When the 1771-M3 controller has transferred the last move of a moveset to the 1771-ES expander, the 1771-M3 controller immediately requests transfer of the next moveset, using the next moveset pointer.

Because the time required for status block transfer and moveset block transfer, you must be sure that the last two moves of any moveset that is to be followed immediately by another moveset have sufficient execution time to allow the necessary block transfers. If you allow insufficient time, a momentary decrease in axis velocity, or an unintended dwell, may occur between continuous moves in two consecutive movesets. If the delay is greater than 30ms, the next move will not execute until you issue a motion command. The insufficient-data bit will be on in this condition.

Important: The moveset block specified by the next moveset pointer must have an allowable block ID.

Command Block

During normal operation, the command block is repeatedly transferred to the 1771-M3 controller. Unless the 1771-M3 controller requests a moveset block or the parameter block, it requests the command block (Figure 7.40).

Command block size depends on whether there are one, two, or three axes. For each axis, the command block contains two control words. It may also include two position preset words, depending on whether a position preset is programmed. The following sections describe the command block words.

Figure 7.40
Command Block - Showing Word Assignments

a) Single–Axis

Word	
1	Control Word 1
2	Control Word 2
3	Position Preset (MS) Word
4	Position Preset (LS) Word

b) Two–Axis

Word	
1	Control Word 1 – Axis 1
2	Control Word 2– Axis 1
3	Control Word 1 – Axis 2
4	Control Word 2 – Axis 2
5	Position Preset (MS) Word – Axis 1
6	Position Preset (LS) Word – Axis 1
7	Position Preset (MS) Word – Axis 2
8	Position Preset (LS) Word – Axis 2

c) Three–Axis

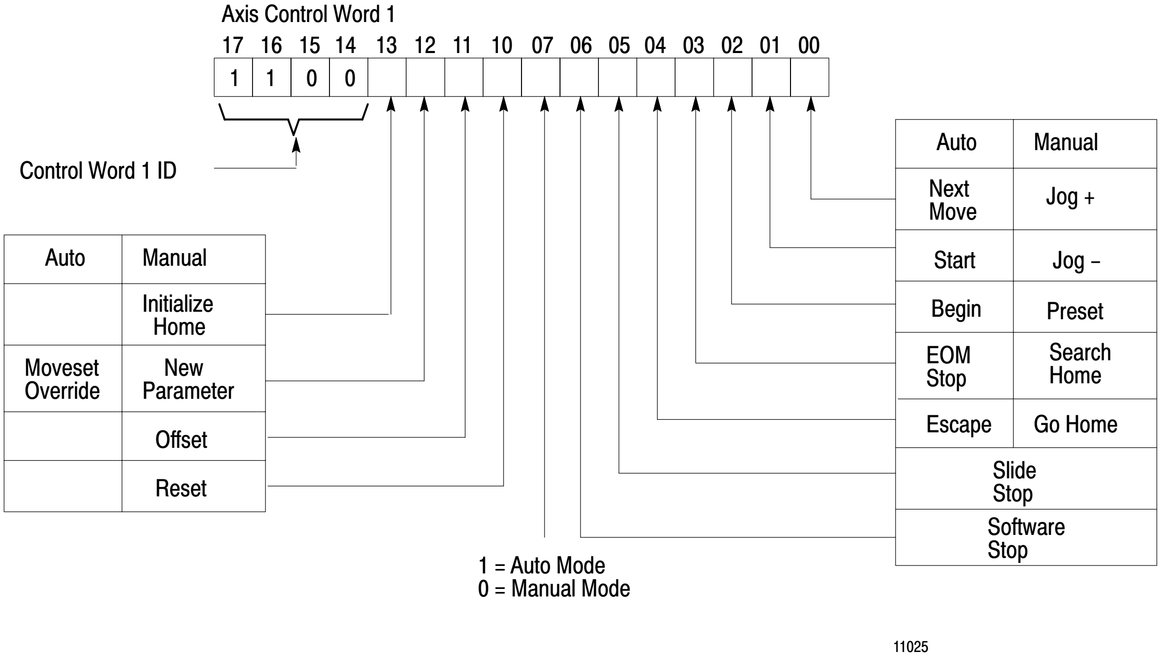
Word	
1	Control Word 1 – Axis 1
2	Control Word 2– Axis 1
3	Control Word 1 – Axis 2
4	Control Word 2 – Axis 2
5	Control Word 1 – Axis 3
6	Control Word 2 – Axis 3
7	Position Preset (MS) Word – Axis 1
8	Position Preset (LS) WOrd – Axis 1
9	Position Preset (MS) Word – Axis 2
10	Position Preset (LS) Word – Axis 2
11	Position Preset (MS) Word – Axis 3
12	Position Preset (LS) Word – Axis 3

11815

Axis Control Word 1

The functions of many of the bits in axis control word 1 depend on the state of bit 7, which determines whether the mode of operation is auto or manual (Figure 7.41).

Figure 7.41
Axis Control Word 1



Bit 0 Next Move

In the auto mode, turn on bit 0 to generate a next move command. The next move command causes the servo positioning assembly to discontinue executing the current move and start executing the next move.

If the endpoints for the current move and the next move are in the same direction from the current axis position, the axis accelerates or decelerates to the final velocity for the next move at the programmed acceleration rate for the next move.

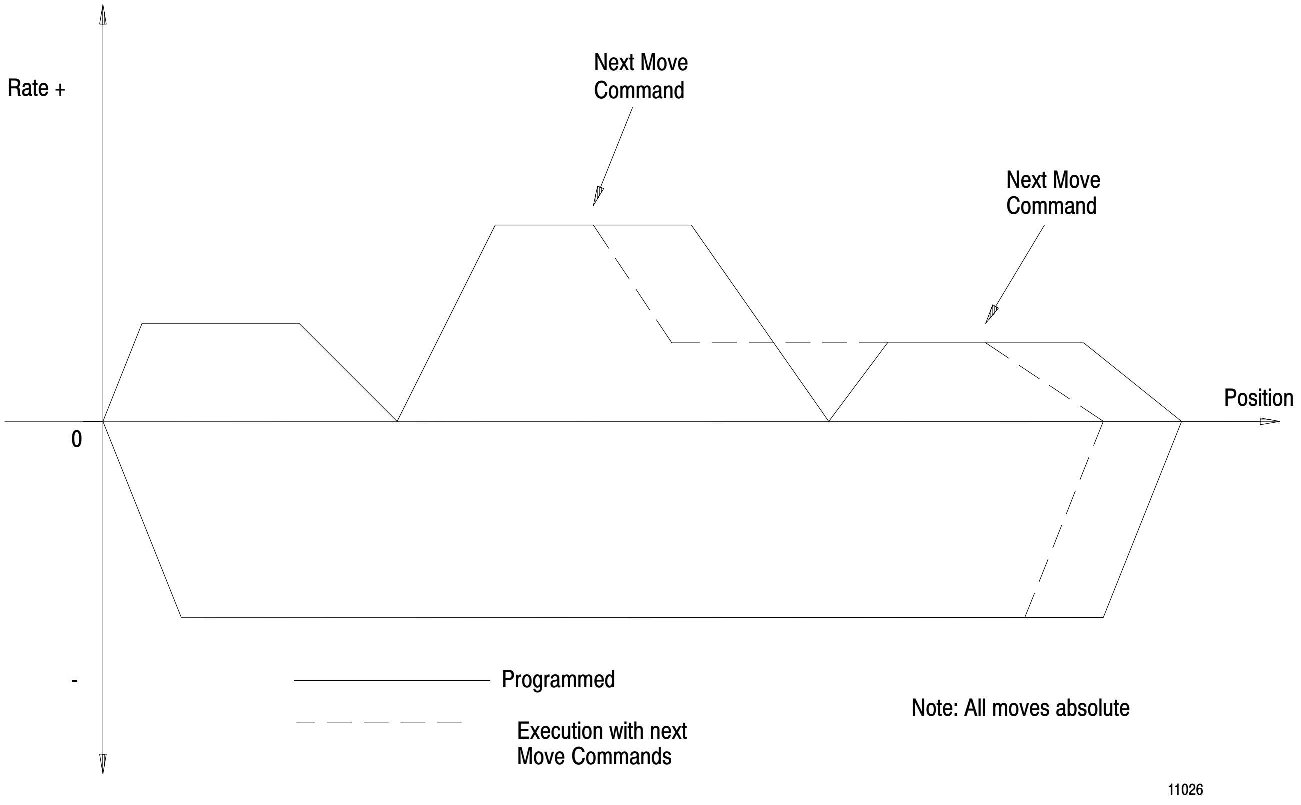
If the endpoint for the next move is in the opposite direction from the endpoint of the current move, the servo positioning assembly performs a slide stop using the decel rate programmed for the current move, then executes the next move (bit 5, slide stop).

If you issue a slide stop command before the next move command, so that the axis is stopped when you issue the next move command, the servo positioning assembly starts execution of the next move without completing execution of the current move.

If you issue the next move command at a time when the axis cannot decelerate at the programmed rate without passing the endpoint of the next move, the servo positioning assembly executes a slide stop that carries the axis past the next move endpoint, then moves the axis in the opposite direction to the desired endpoint.

If no data for a next move is available when you issue a next move command, the 1771-M3 controller sets the insufficient data bit in the status block, and performs a slide stop if the axis is in motion.

Figure 7.42
Moveset Profile - Showing Next Move Command



Bit 0 Jog +

In the manual mode, turn on bit 0 to generate a jog plus (+) command. The axis jogs in the positive direction as long as this bit is on, unless the axis reaches the software travel limit. Jog speed is either high or low, as determined by bit 14 of the second control word for the axis. You specify high and low jog speed values in the parameter block. The axis accelerates to jog speed and decelerates from it at the rate you specified in the parameter block. If you use the decel step velocity (parameter block) it applies to deceleration from jog rates.

Bit 1 Start

In the auto mode, turn on bit 1 to generate a start command. When you turn on this bit, the servo positioning assembly continues moveset execution starting at the point it was last stopped (by completion of a halt move, EOM stop, or slide stop). The start command can also be used to start moveset execution after an escape move is executed. This command can start moveset execution at the beginning of the cycle or after the interruption of the moveset. The 1771-M3 controller only recognizes a start command when the axis is in position. (Also refer to the begin command, bit 2.)

Bit 1 Jog-

In the manual mode, turn on bit 1 to generate a jog minus (-) command. The axis jogs in the negative direction as long as this bit is on unless the axis reaches the software travel limit. Jog speed is either high or low, as determined by bit 14 of the second control word for this axis. You specify high and low jog speed values in the parameter block. The axis accelerates to jog speed and decelerates from it at the global rate value you specified in the parameter block. If you use the decel step velocity (parameter block) it applies to deceleration from jog rates.

Bit 2 Begin

In the auto mode, turn on bit 2 to generate a begin command. This command causes the servo positioning assembly to discontinue executing the current move and start executing the active moveset at its first move block, regardless of what move block is executing.

If the axis is moving, and the endpoint of the first move of the moveset is in the same direction as the axis motion, then the axis will smoothly accelerate or decelerate to the final rate for the first move in the current moveset, and continue moveset execution.

If the axis is moving, and the endpoint of the first move of the moveset is in the direction opposite that of axis motion, the servo positioning assembly:

1. executes a slide stop

2. moves the axis to the endpoint of the first move in the current moveset using the values programmed for that move

3. continues moveset execution

If the servo positioning assembly is executing an escape move when you issue the begin command, it ignores the begin command. If you issue the begin command after the escape move is complete, the servo positioning assembly executes the first move of the normally executed moveset that follows the escape moveset. (That is, the moveset specified by the next moveset pointer of the escape moveset.)

Bit 2 Preset

In the manual mode, turn on bit 2 to generate a preset command. The axis must be stopped for this command to be executed. If a get-new-preset-value command (second control word) has been previously executed, the servo positioning assembly sets its current position value to the current preset position values for the axis in the command block. If no get-new-preset-value command has been executed since power-up or reset, the servo positioning assembly sets its current axis position value to zero.

In addition, execution of a preset command clears the axis offset accumulator, setting its value to zero.

The servo positioning assembly does not request write block transfer of the preset words of the command block unless you issue a get-new-preset-value command. Otherwise, only the first and second control words for each axis are transferred in the command block. Furthermore, when you include the preset words in the command block in response to a get-new-preset-value command, only the axis requesting the new preset value recognizes its new values.

Bit 3 EOM Stop

In the auto mode, turn on bit 3 to generate an end-of-move (EOM) stop command. When this bit is on, the servo positioning assembly completes the current move by decelerating to zero velocity and stopping at the programmed endpoint for the move. If the EOM stop command is received when the axis is beyond the point where it can stop at the programmed endpoint after decelerating, the servo positioning assembly decelerates axis motion to zero velocity, then moves the axis in the opposite direction to the programmed endpoint. When axis motion has stopped, the in-position, done, and ready bits in the status block are on.

You can continue moveset execution after an EOM stop with a start command. When you issue the start command, the axis attempts to move to the endpoint of the move following the EOM stop move. However, a begin command causes the axis to execute the first move block in the current moveset block.

In a typical use of EOM stop, an operator commands an EOM stop, switches to manual mode, then jogs the axis to a convenient location for an operation such as a tool change. Before restarting moveset execution, the operator typically issues a return-to-position command, which causes the axis to move to the endpoint of the last executed move at the selected jog rate. If you issue no return-to-position command before the start command, the axis attempts to accelerate to the final feedrate of the next move in the moveset and continues moveset execution. Figure 7.43 shows three examples of what can happen in this situation.

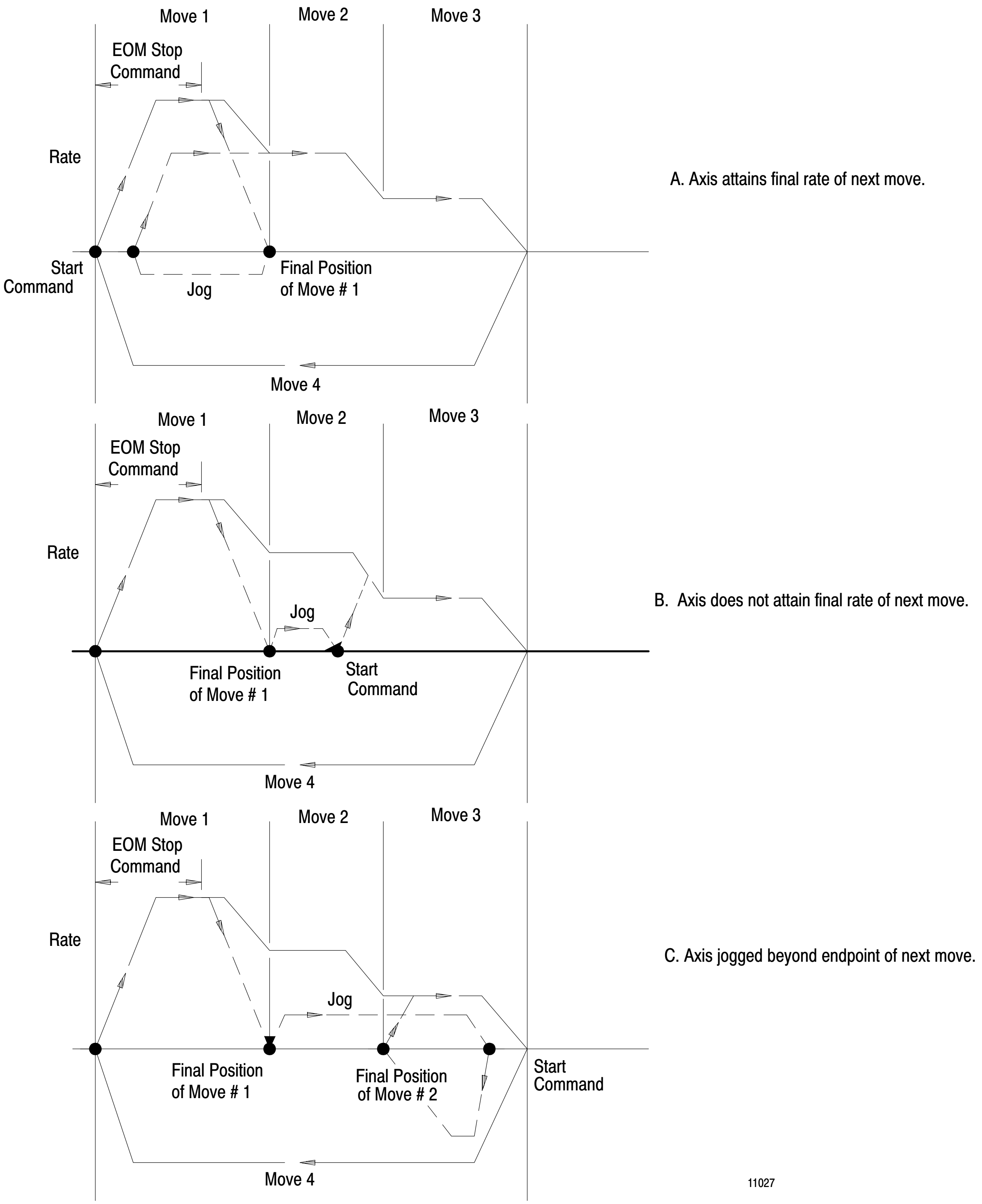
In Figure 7.43a, the axis is able to accelerate to the final rate for the next move and continue moveset execution.

In Figure 7.43b, the axis cannot accelerate to the final rate for the next move, but does continue moveset execution.

In Figure 7.43c, the servo positioning assembly moves the axis to the endpoint of the next move, using the accel, final and decel values for that move, then continues moveset execution.

Note that in all cases, the axis attempts to move to the endpoint of the move that follows the EOM stop move.

Figure 7.43
Moveset Profiles - Showing 3 Possibilities of Jogging After and EOM Stop


Move 1
Move 2
Move 3
EOM Stop Command
Rate
Start Command
Jog
Final Position of Move # 1
Move 4
A. Axis attains final rate of next move.
Move 1
Move 2
Move 3
EOM Stop Command
Rate
Jog
Final Position of Move # 1
Start Command
Move 4
B. Axis does not attain final rate of next move.
Move 1
Move 2
Move 3
EOM Stop Command
Rate
Jog
Final Position of Move # 1
Final Position of Move # 2
Start Command
Move 4
C. Axis jogged beyond endpoint of next move.
11027

Bit 3 Search Home

In the manual mode, turn on bit 3 to generate a search-home command. When you issue the search-home command, the servo positioning assembly moves the axis in the direction you specify by bit 7 of the second control word for the axis (search home direction) and at the speed specified by bit 14 of the second motion control word (jog speed select, high or low). The axis keeps moving until it activates the home limit switch. When the axis trips the home limit switch, it decelerates to zero, then proceeds at the low jog rate to the position at which the first encoder marker occurs. If the axis cannot decelerate to zero velocity at the first marker location, it decelerates past it, stops, then returns to the marker location. This marker location is the axis home position. When the axis reaches its home position, the servo positioning assembly assigns the value in the home position words of the parameter block as the current position.

For a firmware revision F or earlier 1771-ES expander, at power-up, the axis must be positioned at least one encoder revolution away from the home-limit-switch transition before you issue a search-home command (so that a marker can be found before the switch transition). Otherwise, the switch transition will cause a slide stop without establishing a home position.

For a firmware revision G 1771-ES expander, if you issue a search-home command while the home limit switch is closed, the axis first moves away from the home limit switch until the switch has opened and the first market has been found or one encoder revolution has been completed. Once this occurs, the 1771-ES expander decelerates the axis to zero; then if the marker was found, it executes the search home command by moving the axis toward the limit switch to establish home position. However, at power-up, if the home limit switch is open, position the axis at least one encoder revolution away from the home limit switch before issuing a search-home command (so that a marker can be found before the switch transition).

If, while executing a search home command, the axis encounters a software travel limit before it reaches the home limit switch, the servo positioning assembly:

- commands a slide stop
- turns on the appropriate travel limit bit and the slide stop bit in the status block
- sets the current position value to zero

You can then issue another search home command.

You must perform a search home operation after system power-up to initialize the axis position scale. Also, be sure that the home limit switch is in the direction the axis moves when you issue the search home command, as you specify in the parameter block.

Bit 4 Escape

In the auto mode, turn on bit 4 to generate an escape command. This command causes the servo positioning assembly to stop executing the current move and execute the escape move stored on the 1771-ES expander.

If the escape move endpoint is in the direction of current axis motion, the servo positioning assembly smoothly accelerates or decelerates the axis to the escape move velocity, then continues escape move execution.

If the escape move endpoint and current axis motion are in opposite direction, the servo positioning assembly commands a slide stop. After axis motion stops, the escape move executes.

If no escape moveset is stored on the 1771-ES expander and the axis is not in motion when you issue the escape command, the servo positioning assembly turns on the insufficient-data bit in the status block, and requests a new moveset block. If the axis is in motion, the servo positioning assembly performs a slide stop before turning on the insufficient-data bit and requesting a new moveset block.

You can issue a start, begin, or next-move command to start execution of the new moveset.

Bit 4 Go Home

In the manual mode, turn on bit 4 to generate a go-home command. When this bit is on, this command causes the axis to move to the home position at rapid traverse rate, using the accel/decel rate you specify in the parameter block.

If you issue the go-home command before execution of any search-home or initialize-home command, the axis moves to an erroneous home position.

Bit 5 Slide Stop

In either mode, turn on bit 5 to generate a slide stop command. This command causes the axis to decelerate to a stop at the programmed deceleration rate (local or global) for the current move when the servo positioning assembly is in auto mode. After completion of a slide stop, status block bits for slide stop, done, in-position and ready turn on.

If you issue the slide stop command while the servo positioning assembly is in manual mode and the axis is in motion, the axis stops at the global decel rate programmed in the parameter block.

Bit 6 Software Stop

Turn on bit 6 to generate a software stop command. This command causes the servo positioning assembly to go into the immediate stop condition. In this condition it immediately clamps the analog output voltage to zero and turns off the drive disable circuit to disable the servo drive.

The servo positioning assembly executes the software stop command in both auto and manual modes.

The servo positioning assembly goes into the immediate stop condition in response to:

- loss of power
- software stop command from command block
- hardware stop input open
- excess following error
- timeout of a firmware or hardware watchdog
- loss of feedback

To recover from an immediate stop condition, you must either issue the reset command or cycle I/O chassis power. Note that the 1771-ES expander continues to monitor current axis position.

Bit 7 Auto/Manual

Bit 7 selects the mode of operation of the servo positioning assembly:

- **On = Auto Mode** - In auto mode, the servo positioning assembly can execute movesets you have programmed, according to operator commands such as start, begin, next move, EOM stop, escape, slide stop.
- **Off = Manual Mode** - In manual mode, moveset execution is suspended, and the servo positioning assembly can execute jog, preset, home and other manual mode commands.

Bit 10 Reset

Turn on bit 10 to generate a reset command. This command re-initializes all axes when the servo positioning assembly is in manual mode. Issue this command to recover from an immediate stop condition. This command is not recognized, and the status block indicates a programming error, when this bit is on with the servo positioning assembly in auto mode.

If an axis is in motion when this command is issued, the axis performs a slide stop at the global deceleration rate in the parameter block. After the axis stops, the 1771-M3 controller acknowledges the command through the status block.

Important: The reset command affects all axes controlled by a 1771-M3 controller. Motion of **all** axes must stop before the reset command is acknowledged.

The reset command is similar to power-up. That is, all 1771-M3 controller memory is cleared by a reset, including presets and accumulated offsets. However, it is unlike power-up in that the actual position is maintained on the 1771-ES expander. Of course, if an external power-supply loss occurs, you must perform a search home operation after a reset to re-initialize the axis.

Bit 11 Offset

Turn on bit 11 to generate an offset command. When on, this bit commands the servo positioning assembly to add the offset increment value in the parameter block to an offset accumulator. The offset accumulator value is added to each programmed endpoint during move execution. The command-taken bit in the status block remains on as long as this offset bit is on. A preset, initialize home, or reset command clears the offset accumulator.

Important: An offset command has no effect on the two moves already stored on board the 1771-ES expander module. Only moves sent to the 1771-ES expander after you issue the offset command are affected.

The offset command can be executed only when the servo positioning assembly is in the manual mode. When the servo positioning assembly is in auto mode, use a move-to-position-with-offset move block instead of the offset command to increment the offset accumulator.

Bit 12 New Parameter

In the manual mode, turn on bit 12 to generate a new parameter command. This command is typically used to indicate that the parameter block has been changed and the 1771-M3 controller should request it again through the status block. The servo positioning assembly must be in the manual mode, and the axes stopped, for this command to be acknowledged by turning on the command-taken bit. If you issue the command while an axis is in motion, the status block indicates a programming error.

The PC processor transfers the entire parameter block to the 1771-M3 controller in response to the new parameter command. However, only the axis for which the new parameter command was issued receives updated parameter values (words 7 thru 25 for axis 1, words 26 thru 44 for axis 2, words 45 thru 63 for axis 3).

You cannot use the new parameter command to change the parameter block control word or address pointers. This change requires reprogramming of the parameter block and re-initialization of the system.

Bit 12 Moveset Override

In the auto mode, turn on bit 12 to generate a moveset override command. This command provides you with a means of modifying the remaining move blocks of a moveset block while one of its moves is executing.

When you issue the moveset override command, the 1771-M3 controller turns on the command-taken bit and requests another transfer of the current moveset block (unless the last move is executing). After the servo positioning assembly receives the new copy of the moveset block to replace the old copy, it continues executing the moveset block, starting with the next move block. If you had changed the remaining move blocks in the data table before transfer, the 1771-ES expander executes the changed move blocks.

If the current block is completed before the servo positioning assembly receives the new copy of the moveset block, the next move will not execute immediately. This would cause an unintended delay for run moves.

Bit 13 Initialize Home

In the manual mode, turn on bit 13 to generate an initialize home command. This command is functional only when the servo positioning assembly is in the manual mode and the axis is stopped. The status block indicates a programming error if you issue the initialize home command while the axis is moving.

To execute this command, the servo positioning assembly sets the axis current position register to the home position value in the parameter block and clears any accumulated offsets.

Bits 17 thru 14 Control Word 1 ID

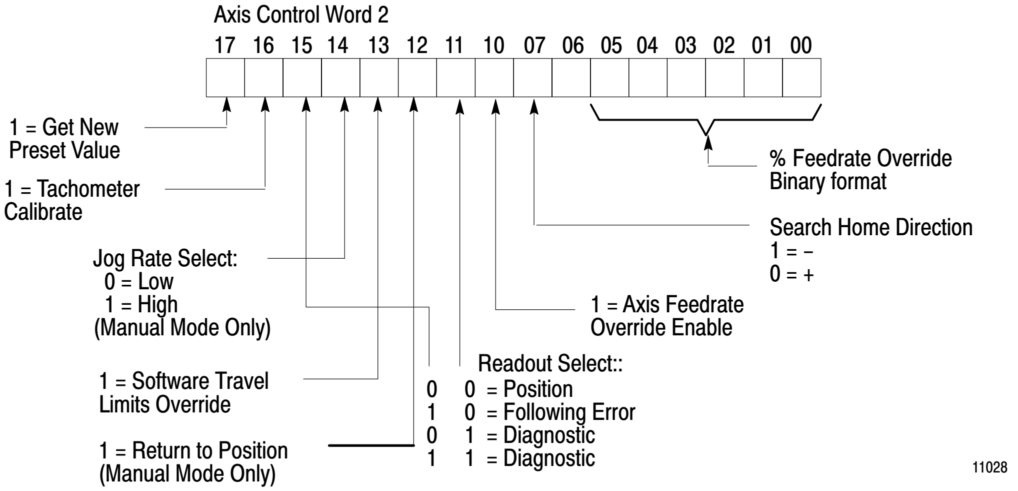
Set bits 17 thru 14 to 1100 to identify this as axis control word 1.

If the control word 1 ID bits for any of the axes are incorrect, the status block indicates a programming error for the axis with the incorrect ID, and all axes execute slide stops.

Axis Control Word 2

Figure 7.44 shows axis control word 2. Bits 12 and 14 apply only to the manual mode.

Figure 7.44
Axis Control Word 2



Bits 0 thru 6 Axis Feedrate Override

Enter the axis feedrate override value into bits 0-6. This specifies the percentage of the programmed feedrate at which moves will be executed if feedrate override is enabled. The value can range from 0% through 127%, expressed in binary form.

Two bits must be on for axis feedrate override to affect enabled moves in auto mode:

- bit 7 of the SMCW (feedrate override enable for current move only)
- bit 10 of the axis control word 2 (axis feedrate override enable)

In manual mode, only bit 10 of axis control word 2 must be on to enable feedrate override for all manual mode axis motion. Note that feedrate override affects only the feedrate value for a move, **not** accel/decel values.

For axis motion to occur, the feedrate override value must be greater than zero, or the axis feedrate override enable bit must be off to disable the feedrate override function.

You can select the moves to be affected by the feedrate override value by turning on bit 7 of the SMCW. If this bit is off, feedrate override does not affect the move.

If you turn on bit 16 of the most significant home-position word, (see section titled “External Synchronization of Feedrate Override”) feedrate override will not start until you close the feedrate-override-enable input. This allows you to synchronize the feedrate override on several axes.

Bit 7 Search Home Direction

Use bit 7 to specify the direction the axis moves in a search home operation:

To make the axis move in this direction:	Then set bit 17 to:
Negative (-)	0
Positive (+)	1

Bit 10 Axis Feedrate Override Enable

Turn on bit 10 to enable feedrate override for the axis.

If this bit is off, feedrate override is disabled for all axis motion. If it is on, feedrate override is enabled only on those moves for which bit 7 of the SMCW is on.

Bit 11, 15 Readout Select

The third and fourth status words for an axis provide either current axis position, following error, or diagnostic information. You can select which status to display by controlling the state of bits 11 and 15 of axis control word 2:

- Turn off bits 11 and 15 to display the current axis position.
- Turn off bit 11 and turn on bit 15 to display the following error.
- Turn on bit 11 to display the diagnostic status.

Bit 12 Return to Position

In the manual mode, turn on bit 12 to generate a return-to-position command. When the servo positioning assembly is in the manual mode and this bit is on, the axis jogs at the selected jog rate (high or low) to the position where the axis was last stopped during execution of a moveset.

If the axis had stopped in the middle of a move, the return-to position command would return the axis to the point where it had stopped, not the endpoint of the move block.

The status block indicates a programming error if you issue the return-to-position command with the servo positioning assembly in auto mode.

Bit 13 Software Travel Limits Override

Turn on bit 13 to override software travel limits. With the servo positioning assembly in the auto or manual mode, and this bit on, software axis travel limits in the parameter block have no effect on axis motion. This allows axis motion to continue beyond the travel limits.

⚠ **CAUTION:** If values for software travel limits are zero, there are no software travel limits. To guard against damage to equipment, exercise caution when operating an axis without software travel limits.

Bit 14 Jog Rate Select

Bit 14 determines the jog rate for jogs, the search home operation, and the return-to-position operation.

- Turn off this bit to select the low jog rate.
- Turn on this bit to select the high jog rate.

You enter the high and low jog rates in the parameter block.

If this bit changes state during a jog operation, the axis accelerates or decelerates to the newly commanded rate at the global accel/decel rate programmed in the parameter block and continues jogging at the newly commanded jog rate.

During a search home or return-to-position operation, the axis ignores changes in this bit.

Bit 16 Tachometer Calibrate

Turn on bit 16 for the tachometer calibration procedure described in chapter 9. At all other times, leave it off.

During tach calibration, bit 15 of the most significant home position word in the parameter block (loss-of-feedback detection enable) must also be on.

This bit is ignored when the servo positioning assembly is in auto mode.

Bit 17 Get New Preset Value

When you turn on bit 17, the 1771-M3 controller requests a command-block transfer to include the position preset for the axis. The servo positioning assembly uses the position preset value when you issue a preset command for the axis.

The command-taken bit of the status block is on as long as the get-new-preset-value bit is on (unless the preset data results in a programming error).

When bit 17 is off, the position preset words are not transferred to the 1771-M3 controller during command block transfer.

Note that the preset command causes the 1771-M3 controller to use the position preset value that was transferred to the 1771-M3 controller during execution of the most recently issued get-new-preset-value command.

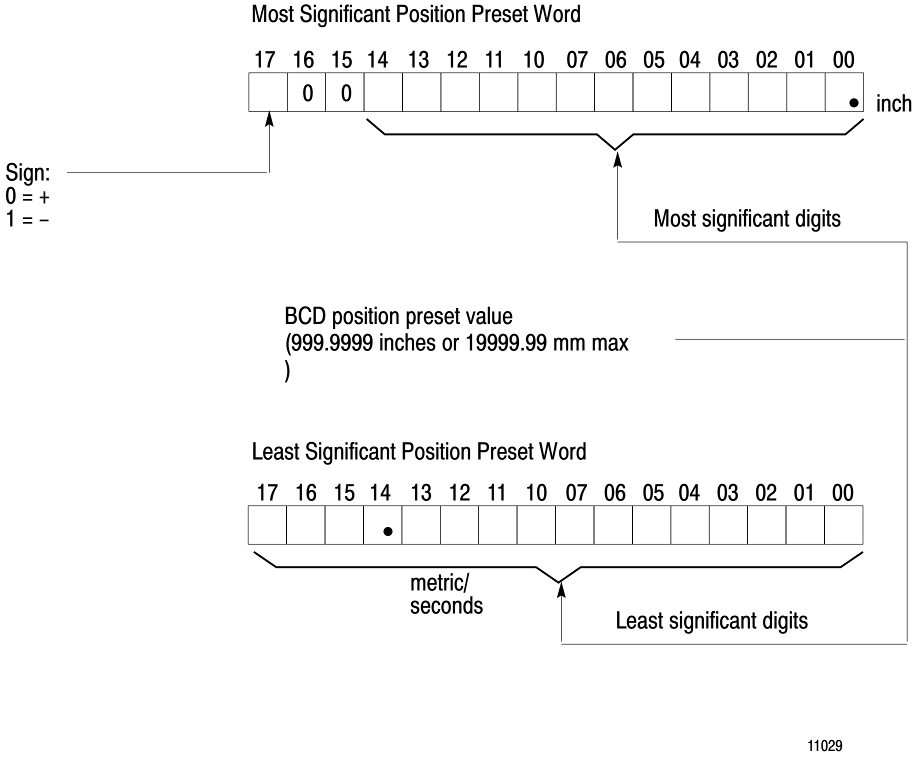
Position Preset Words

Two words of the command block are used to specify a new position preset value for the axis (Figure 7.45). These two words are requested by the 1771-M3 controller when the get-new-preset-value bit is on.

Note that bit 17 of the most significant position preset word specifies the sign of the preset value (0=+, 1=-).

Preset values are in BCD format. The maximum value is 999.9999 in. or 19999.999 mm.

Figure 7.45
Position Preset Words



Illegal Combinations

Simultaneously setting certain bits in control words 1 and 2 for an axis results in a programming error, sometimes with slide stop. Table 7.D shows these bit combinations.

Table 7.D
Command Block Illegal Command Combinations

	Word #		1										2
Word #	(word #1, bit #7 is off) Manual Mode	Bit Meaning	Initialize Home	New Parameters	Offset	Reset	Manual	Go Home	Search Home	Preset	Jog –	Jog +	New Preset
	Bit Meaning	Bit	13	12	11	10	7	4	3	2	1	0	17
2	Ret. to Position	12	PE	(PE)				(PE)	(PE)	PE	(PE)	(PE)	
	New Preset	17											
1	Jog +	0	(PE)	(PE)				(PE)	(PE)	(PE)	(PE)		
	Jog –	1	(PE)	(PE)				(PE)	(PE)	(PE)			
	Preset	2	PE		PE			PE	PE				
	Search Home	3	PE	(PE)				(PE)					
	Go Home	4	PE	(PE)									
	Manual	7											
	Reset	10											
	Offset	11		PE									
	New Parameters	12	PE										
	Initialize Home	13											

	Word #		1										2
Word #	(word #1, bit #7 is on) Auto Mode	Bit Meaning	Initialize Home	Moveset Override	Offset	Reset	Auto	Escape	EOM Stop	Begin	Start	Next Move	New Preset
	Bit Meaning	Bit	13	12	11	10	7	4	3	2	1	0	17
2	Ret. to Position	12	PE	PE	PE	PE	PE	PE	PE	PE	PE	PE	PE
	New Preset	17	PE		PE	PE							
1	Next Move	0	PE	PE	PE	PE		(PE)	(PE)	(PE)	(PE)		
	Start	1	PE	PE	PE	PE		(PE)	(PE)	(PE)			
	Begin	2	PE	PE	PE	PE		(PE)	(PE)				
	EOM Stop	3	PE	PE	PE	PE		(PE)					
	Escape	4	PE		PE	PE							
	Auto	7	PE	PE	PE	PE							
	Reset	10	PE	PE	PE								
	Offset	11	PE	PE									
	New Parameters	12	PE										
	Moveset Override	13											

(PE) = Programming error with slide stop

PE = Programming error without slide stop

Here are some additional notes about illegal programming:

- Even though the programming error bit in the status block is on, the 1771-M3 controller responds to the following commands and information:
    - emergency stop
    - slide stop
    - reset
    - software travel limit override
    - tachometer calibrate
    - feedrate override value
    - jog rate select
- If you issue a preset, new parameter, or initialize home command while the axis is in motion, the status block indicates a programming error. If the command is still present after axis motion stops, the 1771-M3 controller acknowledges it, and executes it.
- The status block indicates a programming error if you change the servo positioning assembly mode from auto to manual or from manual to auto under either of the following conditions:
    - If you change the mode while an axis is in motion, the status block indicates a programming error and the 1771-M3 controller commands a slide stop. The programming error bit clears when axis motion stops. If the mode change is still commanded after axis motion stops, the mode changes.
    - If you attempt a mode change when any of bits 0 thru 4 of the first axis control word or bit 12 of the second axis control word are on, the 1771-M3 controller sets a programming error in the status block for the axis. If the axis is in motion, a slide-stop is executed. The mode change occurs only after bits 0 thru 4 of the first axis control word and bit 12 of the second axis control word are off and only the command for the mode change is still present.

Summary

In this chapter we told you what information to put into the parameter, moveset, and command blocks to direct the servo positioning system. We also told you how to monitor the servo positioning system thru the status block.

Now you need to learn how to generate a ladder diagram program to transfer these blocks of information between the data table and the servo positioning system.

Chapter 8

Programming

Chapter Objectives

The previous chapter told you what information to put into and monitor from the data blocks. This chapter tells you how to generate a ladder diagram program to transfer these blocks between the data table and the servo positioning assembly.

Programming Objectives

The main objectives of a program for the servo positioning assembly are to:

- transfer the status block from the 1771-M3 controller to the data table
- transfer the parameter, moveset, and command blocks from the data table to the 1771-M3 controller

You can use a block transfer read instruction to continually transfer the status block from the 1771-M3 controller to the data table.

You can use a single block transfer write instruction to transfer either a parameter, moveset, or command block, from the data table to the 1771-M3 controller. Your program must manipulate the address of the block to be transferred by the block transfer write instruction so that the right block gets there at the right time.

After power-up, the first block to send is the parameter block. In the parameter block, you must include address pointers for (Figure 8.1):

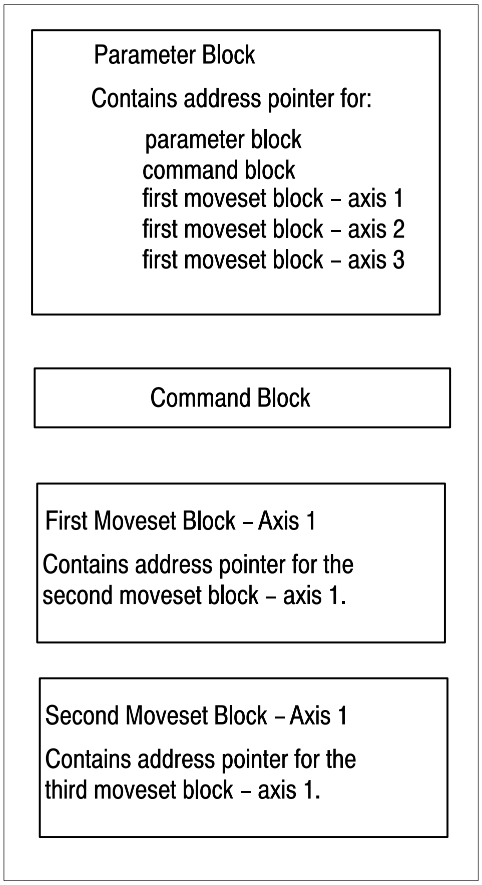
- parameter block
- command block
- first moveset block for each axis

Once it receives this information, the 1771-M3 controller can start to request the block it needs by sending its address pointer in the status block. After receiving the parameter block, the 1771-M3 controller requests the first moveset block for each axis.

In each moveset block, you must include the address pointer for the next moveset block if one exists for the axis.

Once it has a moveset block for each axis, unless it is down to the last two moves, the 1771-M3 controller normally requests the command block through the status block.

Figure 8.1
Data Blocks Sent to the 1771-M3 Controller - Showing Where Address Pointers are Given


Parameter Block
Contains address pointer for:
parameter block
command block
first moveset block – axis 1
first moveset block – axis 2
first moveset block – axis 3
Command Block
First Moveset Block – Axis 1
Contains address pointer for the
second moveset block – axis 1.
Second Moveset Block – Axis 1
Contains address pointer for the
third moveset block – axis 1.

When it gets down to its last two moves of a moveset, the 1771-M3 controller requests another moveset block.

After the parameter block is sent, your program must use the address pointer from the status block to direct the block transfer write instruction so that the requested block always transfers to the 1771-M3 controller.

For detailed information on programming block transfer instructions, refer to the appropriate programming manual as listed in our Publication Index (publication 499).

PLC-2 Family Block Transfer Instructions

Figure 8.2 shows the formats of block transfer instructions for Mini-PLC-2/15 and PLC-2/30 controllers. For each block transfer instruction for these PCs, you must specify:

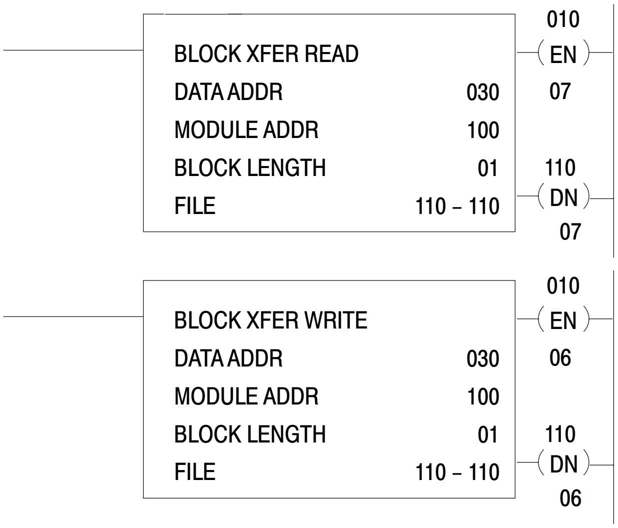
- Data Address - The address of a word in the timer/counter accumulated value area of the data table. This word contains the 1771-M3 controller module location address, in BCD format.

  For bidirectional block transfer, there must be two data address words in consecutive data table locations: one for the write transfer, and one for the read transfer. These words contain the same module location address.

- Module Address - A 3-digit number of the form RGS, where R is the rack number, G is the module group number, and S is the slot number (0 or 1). This value is stored in the data address word.
- Block Length - Specifies the number of words to be transferred. For the servo positioning assembly, program 00 as the block length. This is the default value. When the PC executes the block transfer instructions, the 1771-M3 controller automatically sends or requests the correct number of words.
- File - The data table address of the first word in the block to be transferred. For read transfer, this address is constant. For the write transfer, your program changes the address according to the address pointer in the status block. Note that the address of the first word in the block to be transferred is stored in a data table word 100_8 above the data address.

Figure 8.2 also shows enable (EN) and done (DN) outputs for each block transfer instructions. The PC automatically enters addresses for these output according to the module location address you enter. These addresses specify bits of the input and output image table bytes that correspond to the 1771-M3 controller location.

Figure 8.2
Block Transfer Instructions for P-2/30 and Mini PLC-2/15



Data Address : First possible address in accumulated vaue area of data table.

Module Address : RGS - R = rack, G = module group, S = Slot number. (This value is stored in the data address word.)

Block Length : Number of words to be transferred. (00 can be entered for default value or for 64 words.)

File : Address of first word in the file. (This value is stored in the 100_8 above the data address.)

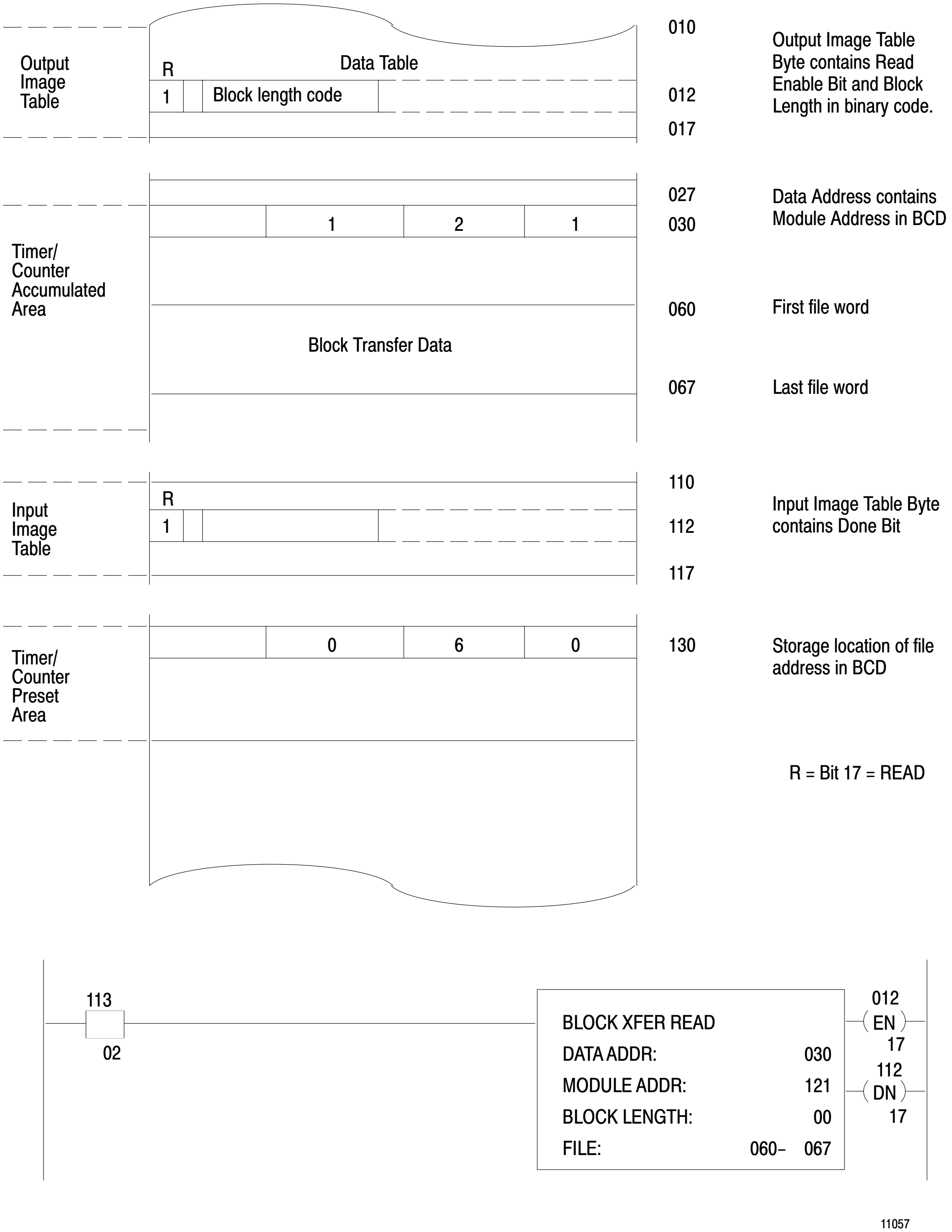
Enable Bit (EN) : In the output image table word for the module. Set on when rung containing the instruction is true.

Don Bit (DN) : In the input image table word for the module. Remains on for 1 scan following successful transfer.

Figure 8.3 shows an example of data table arrangement for a read block transfer.

Data table words assigned for block storage must not include reserved processor work areas. That is, you must ensure that you assign starting addresses for the block so that the words requested by the 1771-M3 controller do not include words in the processor work area. If the PC attempts to transfer processor work area words to the 1771-M3 controller, the PC can lock up, out of communication with the 1771-M3 controller. For PLC-2-family processors, store blocks at data table addresses above 200_8, and no block should begin less than 64 words from the start of the user program.

Figure 8.3
Block-transfer-read-instruction Example (PLC-2/30 or Mini-PLC-2/15)



PLC-2-Family Block Transfer Timing

Because the servo positioning assembly relies on bidirectional block transfer for communication with the PC processor, the time required for block transfer operations may be critical in certain situations. For example, operator commands are transmitted to the 1771-M3 controller via command block transfer. Depending on when a command is issued, up to four block transfers (two read, two write) may occur before the 1771-M3 controller can act on it.

Another example involves continuous execution of consecutive movesets. The 1771-M3 controller stores one moveset block on-board. It transfers individual moves to the 1771-ES expander one at a time. The 1771-ES expander stores two moves on-board: the current move it is executing and the next move. After the 1771-M3 controller transfers the last move of a moveset to the 1771-ES expander, it requests transfer of the moveset block from the PC processor. This request is transmitted via the status block. In the worst case, four block transfers (two read, two write) may occur before the 1771-M3 controller can send the next move to the 1772-ES expander. If the 1771-ES expander completes execution of the first two moves of a moveset before it receives the next move, an unintentional dwell may occur in move execution.

Factors involved in block transfer timing include:

- system scan time
- block length
- system I/O configuration
- the number of enabled block-transfer instructions in a given program scan

To calculate worst case block transfer times, assume maximum block length (64 words write, 14 words read), and that a block transfer instruction is enabled for each block transfer module in the system in every program scan.

The following sections describe calculation of worst case block transfer times for PLC-2/30 remote and local systems, and for the Mini-PLC-2/15 system.

PLC-2/30 Remote System

To find the time between block transfers for a given module in a PLC-2/30 remote system, perform the following steps:

1. Find the block transfer times of each block transfer module in the system.

2. Determine the sequence of block transfers for the system.

3. Sum block transfer times according to the system sequence.

Block Transfer Time

System scan time for a PLC-2/30 remote system is the sum of processor program scan time, processor I/O scan time, and remote I/O scan time. For worst case calculation, assume that the Remote I/O Scanner (cat. no. 1771-SD2) can process only one block transfer operation per remote I/O scan.

To calculate worst case block transfer time for a module, perform the following steps:

1. Write down known facts:

- program length
- number of chassis
- number of block transfer modules
- block lengths (W)

2. Calculate system values determined by system configuration:

- program scan time in ms
  PS = (5ms/K word) x (program length)
- processor I/O scan time in ms
  PIO = (0.5ms/chassis) x (number of chassis)
- remote I/O scan time in ms
  RIO = (7ms/chassis) x (number of chassis)

3. Calculate individual block transfer times:

- write transfer time in ms
  TW = PS + PIO + 2 RIO + 0.5W + 13
- read transfer time in ms
  TR = PS + PIO + 2 RIO + 0.5W + 4

These equations are valid for a data transfer rate of 57.6k bits/s, or 115.2k bits/s.

For worst case calculations, use the longest block transfer time.

Block Transfer Sequence

As stated above, the remote I/O scanner can process only one block transfer per remote I/O scan, worst case. If a system has N I/O chassis with block-transfer modules, a block transfer for a given chassis occurs once each N system scans. If a given chassis contains X block-transfer modules, block transfer for any one of them occurs once each (N) x (X) remote I/O scans.

For example, consider a system with 4 I/O chassis, each of which contains one or more block transfer modules:

- chassis 1 - modules A, B, C
- chassis 2 - modules D, E
- chassis 3 - module F
- chassis 4 - modules G, H, I

The block-transfer sequence for this system is:

A D F G B E F H C D F I A E F G B D F H C E F I...

Note that block transfer for modules A, B, and C occurs once each 12 system scans (4 chassis x 3 modules). Block transfer for modules D and E occurs once each 8 system scans (4 chassis x 2 modules), and so on.

Example Calculation

Consider the PLC-2/30 remote system of Figure 8.4. This system has four I/O chassis, each of which holds one 1771-M3 controller module. The ladder diagram program is 4K words long. Calculate the worst case time between write block transfers for one of the 1771-M3 controller modules in the following steps:

1. Write down known facts:

- Program Length = 4K words
- Number of Chassis = 4
- Block Length - 64 words write, 6 words read

2. Calculate system values:

- Program Scan Time
  PS - (5ms/K word)(4 K words) = 20ms
- Processor I/O Scan Time
  PIO = (0.5ms/chassis)(4 chassis) = 2ms
- Remote I/O Scan Time
  RIO = (7ms/chassis)(4 chassis) = 28ms

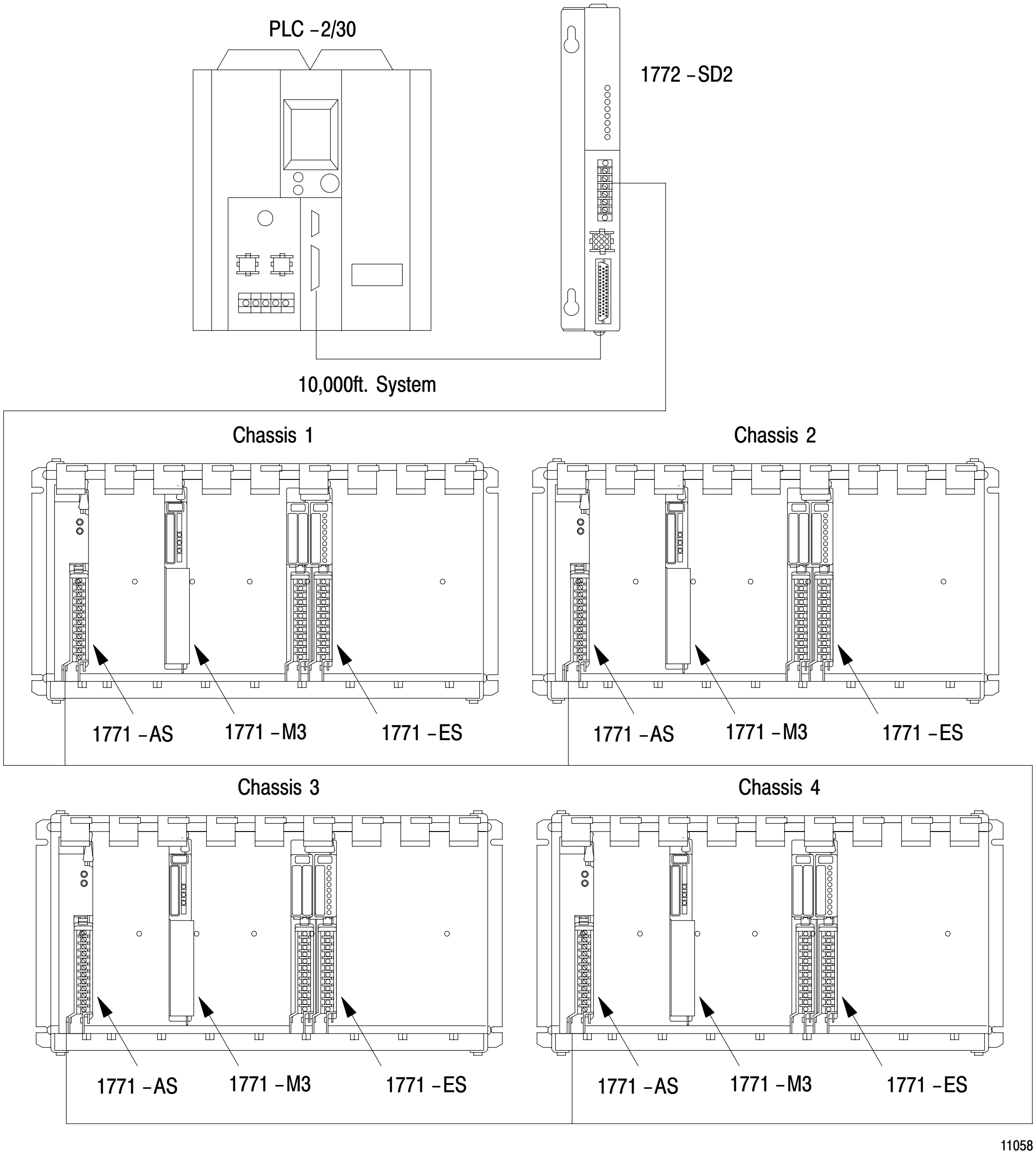
3. Calculate Block Transfer Times:

Write, TW = PS + PIO + 2 (RIO) + .5 W + 13ms
= 20 + 2 + 2 (28) + .5(64) + 13ms
= 123ms

Read, TR = PS + PIO + 2 (RIO) + .5 W + 4ms
= 20 + 2 + 2 (28) + .5(6) + 4ms
= 87ms

These times apply to all four 1771-M3 controller modules. If the system included other block transfer modules, separate calculations would be required for each.

Figure 8.4
PLC-2/30 Remote System Example



4. Calculate worst case time between write transfers:

Since each chassis contains one block transfer module, each 1771-M3 controller is serviced once every four remote I/O scans. Because the 1771-M3 controller module uses bidirectional block transfer, a read transfer occurs between consecutive write transfers. Consequently, a write transfer for a given 1771-M3 controller occurs every eight system scans. In those eight scans, four write and four read transfers occur:

$$\begin{aligned}\text{Time Between Write Transfers} &= 4TW + 4TR \\ &= 4(123) + 4(87) \\ &= 840\text{ms}\end{aligned}$$

PLC-2/30 Local System

In a PLC-2/30 local system, all block-transfer modules are serviced once in each system scan. Time between consecutive block transfers for a given module is, consequently, system scan time plus the sum of the individual block transfer times of the modules in the system.

Time Between Transfers = (system scan time) + T1 + T2...+Tn

where T1, T2..., Tn are individual block transfer times.

System scan time for a PLC-2/30 local system is program scan time (PS) plus processor I/O scan time (PIO):

PS = (5ms/K word) x (program length)

PIO = (1ms/chassis) x (number of chassis)

You can calculate individual block-transfer times with this formula:

T = 0.1ms + (0.075ms/word) x (block length)

The same formula applies to both read and write transfers.

As an example, consider a PLC-2/30 local system with four I/O chassis, each of which holds one 1771-M3 controller module and no other block-transfer modules. User program length is 4k words. To calculate the worst case time between consecutive write block transfers for one of the 1771-M3 controllers follow these steps:

1. Write down known values:

- program length = 4K words
- number of chassis = 4
- block length = 64 words write, 10 words read

2. Calculate system values:

- program scan time
  PS = (5ms/K word)(4K words) = 20ms
- processor I/O Scan time
  PIO = (1ms/chassis)(4 Chassis) = 4ms

3. Calculate individual block transfer times:

- Write
  TW = 0.1ms + [(0.075ms/word)(64 words)] = 4.9ms
- Read
  TR = 0.1ms + [(0.075ms/word)(10 words)] = 0.85 ms

These times apply to all four 1771-M3 controller modules in the system. If the system includes other block-transfer modules, you will have to make separate calculations for each.

4. Calculate the time between consecutive write transfers. Since a read transfer occurs between write transfers, you must include 2 system scans, 4 write transfer times and 4 read transfer times in the formula:

Time between write transfers = 2(PS + PIO) + 4 TR + 4 TW
= 2(24ms) + 4(0.85ms) + 4(4.9ms)
= 71ms

Mini-PLC-2/15 Controller

Block transfer timing for the Mini-PLC-2/15 controller is similar to that for the PLC-2/30 local system. The program scan and processor I/O scan are consecutive and are considered as one scan. Scan time typically varies from 18 to 24ms per 1K word of user program:

- processor scan time

  PS = (24ms/K word) x (program length)

- Individual block transfer times can be calculated from this formula:

  T = 0.1ms + (0.16ms/word) x (block length)

For example, consider a Mini-PLC-2/15 controller with one 1771-M3 controller in its I/O chassis. There are no other block transfer modules, and program length is 2K words.

To calculate worst case time between write block transfers for this system follow these steps:

1. Write down known values:

- program length = 2K words
- block length = 64 (write) or 10 (read)

2. Calculate processor scan time:

- PS = (24ms/K word)(2K words) = 48ms

3. Calculate block transfer times:

- TW = 0.1ms + [(0.16ms/word)(64 words)] = 10.34ms
- TR = 0.1ms + [(0.16ms/word)(10 words)] = 1.7ms

4. Calculate time between write transfers:

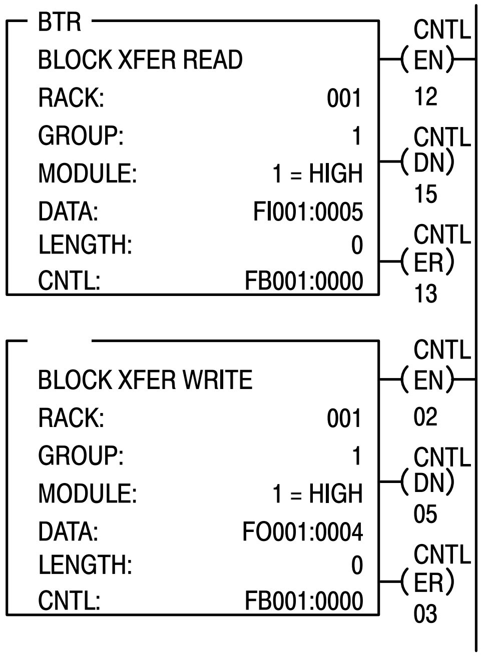
- TBT = 2(PS) + TR + TW = 96 + 10.34 + 1.7 = 108.04ms

PLC-3 Block Transfer Instructions

Figure 8.5 shows the formats of block transfer instructions for a PLC-3 processor. For each block transfer instruction you must specify:

- I/O rack
- I/O module group within the I/O rack
- I/O module slot within the I/O module group

Figure 8.5
Block-transfer Instructions for PLC-3 Controllers



Block transfer instructions use two files when transferring data and commands between the block transfer module and the PLC-3 processor:

- a data file that contains data being transferred
- a control file that contains control bits, module location, data table address and length of the data file

The I/O scanner directs communication between the block-transfer module and processor. Once the block-transfer instruction is enabled, the scanner directs the transfer of data to or from the enabled block transfer module according to the information contained in the instruction's control file. Once the instruction is enabled, the PLC-3 processor automatically sets and resets the control bits in accordance with the various steps required to execute the read or write operation.

PLC-3 Block Transfer Timing

The execution time required to complete a read/write block transfer depends on factors that include the number of:

- words of user program
- active I/O channels on the scanner
- I/O chassis entries on the I/O chassis scanning sequence list for the channel

- I/O channels on the scanner that contain block-transfer modules
- block transfer modules on the channel (if the I/O chassis containing a block transfer module appears more than once in the I/O chassis scanning sequence list count the module once each time the chassis appears in the list)

Typical time required for the module to complete a read/write (bidirectional) block transfer depends on the program scan and the I/O scan as follows:

time (read/write) = program scan + 2 (I/O scan)

Program Scan

The program scan is approximately 2.5ms per 1K words of user program when using examine on/off and block instructions.

I/O Scan

The time required for the scanner to complete a read or write block transfer depends on the number of other block-transfer modules on the same scanner channel that the program enable simultaneously. Use the following procedure to calculate the time required for the PLC-3 processor to perform all block transfers on the channel and be ready to perform the first transfer again.

Block transfer times typically are similar regardless of the type of block transfer module, whether a read or write operation, or the number of words transferred. To calculate the I/O scan time for block transfer, follow these steps:

1. Determine the number of active I/O channels on the scanner.

2. Determine the number of I/O channels with block-transfer modules.

3. Use this table to determine the nominal block transfer time using the number from steps 1 and 2.

Nominal Time (ms) per Block Transfer

Channels with Block-Transfer Modules	1 Active Channel	2 Active Channels	3 Active Channels	4 Active Channels
1	40	52	54	58
2	--	67	68	76
3	--	--	98	99
4	--	--	--	123

4. Count the number of block-transfer modules on the channel. If a chassis containing block-transfer modules is repeated in the chassis scanning sequence list, count the modules as often as listed.

5. Count the number of I/O chassis entries in the chassis scanning list for the channel.

6. Calculate the time between block transfers for the scanner as follows:

scanner time = [nominal time x BT modules on the channel] + [I/O chassis in list -1) x 9ms

Example Computation

As an example, we compute the read/write block transfer time for each of two 1771-M3 controllers in the following system:

- User program contains 20K words.
- Channel 1 contains five I/O chassis, with a total of seven block-transfer modules including one 1771-M3 controller.
- Channel 2 contains two I/O chassis with no block-transfer modules.
- Channel 3 contains two I/O chassis with one 1771-M3 controller.
- Channel 4 is made inactive thru LIST.

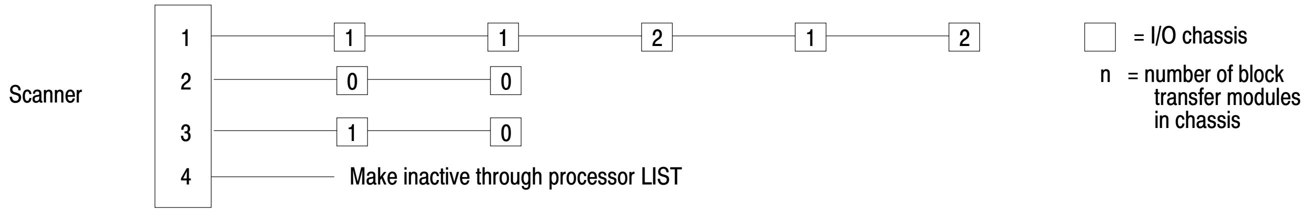
You can compute the read/write block transfer times for the 1771-M3 controllers in this example in four steps. An accompanying figure explains each of the following four steps.

1. Diagram the I/O channels of your PC system (Figure 8.6), showing the number of:

- block transfer modules in each I/O chassis
- block transfer I/O channels
- I/O chassis entries in the chassis scanning sequence list for each block transfer I/O channel
- active I/O channels per scanner

Figure 8.6
Diagram of PLC-3 I/O Channels

Step 1 - Diagram the chassis connected in series to each channel (up to 4) of your scanner module. Then, fill in the information called for below. Example values have been added.


Scanner
1
2
3
4
1
1
2
1
2
0
0
1
0
Make inactive through processor LIST
= I/O chassis
n = number of block transfer modules in chassis

Description	Number	Ch 1	Ch 2	Ch 3	Ch 4
Active I/O channels	3				
Block transfer I/O channels	2				
Block transfer modules on each I/O block transfer channel		7	0	1	0
I/O chassis on each block transfer I/O channel (I/O chassis in rack list		5	0	2	0

A block transfer I/O channel is a channel that contains one or more block transfer modules located in any chassis connected to the channel.

An I/O chassis can appear more than once in a chassis scanning sequence list. Count it and the block transfer module(s) that it contains as often as it is listed.

2. Step 2 - Using information from Figure 8.6, look up the nominal time from the table in Figure 8.7.

Figure 8.7
Nominal Time Table

Step 2 - Determine a time from the table. Example values have been added.

Number of Active I/O Channels

		1	2	3	4
Active I/O channels containing one or more block transfer modules.	1	40	52	54	58
	2		67	68	76
	3			98	99
	4				123
		Time (ms)			

Example:

Number of active I/O channels:	3
Number of active I/O channels ccontaining one or more block transfer module:	2
Time, from table:	68ms

3. Compute the approximate transfer time for each block-transfer I/O channel. You will use a value from the table, values from your diagram (Figure 8.6), and the formula (Figure 8.8).

Figure 8.8
Channel Time Computation

Step 3 - Compute the scanner times for each block transfer channel. Example values have been added.

```
CT  =  Channel Time

CT  =  [Time] x [ # BT  modules] + [#  I/O chassis -1] x 9ms
       (table)     on BT channel         on BT channel

  CT1 =  [68] x [7] + [5 -1] x 9
      =  [68] x [7] + [4] x 9
      =  476 + 36
      =  512ms
  CT2 =  Not a block transfer channel
  CT3 =  [68] x [1] + [2 - 1] x 9
      =  [68] x [1] + 1 x 9
      =  68 + 9
      =  77ms
  CT2 =  Not an active channel
```

4. Compute the approximate read/write block transfer time for the 1771-M3 controller in channel 1 and in channel 3 (Figure 8.9).

Figure 8.9
Block-transfer Time Computation

Step 4 – Compute the 1771–M3 controller read/write block transfer time. Example values have been added.

Program Scan:

Time (program) = 2.5ms per 1K words x 20K words
= 2.5 x 20
= 50ms

Scanner Time:

Time (read or write) = 512ms for channel 1 and 77ms for channel 3 (from results of sep 3)

Read/Write

Time
(1771–M3 Controller
in Channel 1

= Program scan + 2[Scanner Timer]
= 50 + 2(512)
= 50 + 1024
= 1074ms
= 1.1 seconds

Time
(1771–M3 Controller
in Channel 3)

= Program scan + 2[Scanner Time]
= 50 + 2[77]
= 204ms

Reducing Scan Time

Due to the asychronous scan relationship between program scan and I/O scan, and the serial operation of each channel in the scanner, we suggest that you optimize the overall scan time. Although recommendations are application dependent, we make the following recommendations as general guidelines:

- Whenever possible, control the manner in which block-transfer instructions are enabled. For example, if only a few block transfer modules require frequent transfer of data (as the 1771-M3 controller), program them to run continually. Inhibit block transfer instructions of those modules that require less frequent transfer until enabled by a timer and/or some application dependent condition.
- Distribute your block transfer modules equally between all four scanner channels.
- Distribute block transfer instructions equally throughout your program. Place an equal number of non-block-transfer rungs between block-transfer rungs. Consider the last rung adjacent to the first.
- For large numbers of block transfer instructions, distribute groups of block transfer rungs equally throughout your program. Place no more than four block transfer rungs consecutively in one group (one block transfer instruction per rung). Within each group, condition the next rung using the done bit of the previous block transfer instruction.

- Consider an additional I/O Scanner Module (cat. no. 1775-S4A, -S4B) if you cannot otherwise reduce the block transfer times to meet your timing requirements.
- During a write handshake, the processor also can transfer write data; and during a read handshake, the processor also can transfer read data.

Special Considerations

When using one 1775-S4A I/O scanner with thumbwheel switchset to 1, only part of its data handling capacity is available for block transfers. This scanner can store and transfer a maximum of 72 words at any one time, from up to four block transfer modules, across any of the active channels.

If a block-transfer-read instruction is enabled but the scanner's buffer cannot accept the instruction's block length (the scanner is processing other blocks of data), the block transfer instruction must wait for a subsequent scan when the scanner's buffer can accept all the words that the module has to transfer. The same applies for a write block transfer instruction.

Block-Transfer Errors

Once enabled, a block-transfer instruction will set either a done bit or an error bit. The instruction indicates an error when it illuminates the -(ER)- symbol. Typical block transfer errors occur when:

- You do not correctly enter the instruction:
  - rack, group, and module numbers do not match the location of the installed module
  - you entered a file length greater than 64 words
  - you did not create the data file, or the address that you entered does not match the file you created

If the read and write error bits are on at the same time, the error source is the module-address entry or the file-length entry in the instruction block.

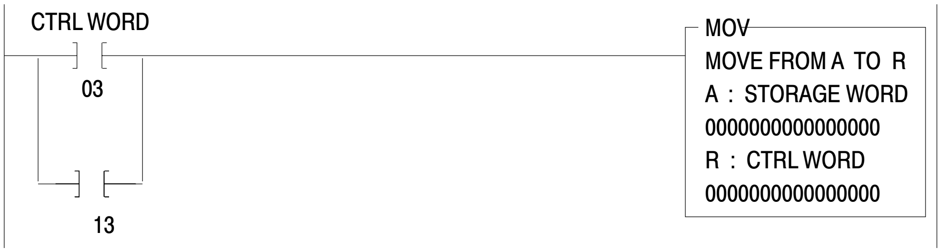
- You have a communication problem:
  - you did not correctly connect the twinaxial cable to the scanner
  - you did not connect a terminator resistor to each end of the twinaxial cable

When the scanner encounters a communication fault, it tries twice to complete the transfer. it sets the error bit after the second unsuccessful try.

When the scanner and/or processor detects a block-transfer error, it halts the transfer. Transfers from that module are prevented until:

- your program clears the instruction's control word (clears the error, Figure 8.10)
- you locate and correct the error

Figure 8.10
Example Rung to Clear the Control Word



Programming Example

Consider a servo positioning assembly that controls the motion of two axes. Figure 8.11, Figure 8.12, and Figure 8.13 show three moveset profiles. Assume that you are to program these profiles for execution by axis 1. When the servo positioning assembly receives a start command via the command block, it is to execute movesets 1, 2, and 3 in sequence. Axis motion is to stop after execution of moveset 3 is completed.

Figure 8.14 shows a single moveset profile. Assume that you are to program this profile for execution by axis 2. After axis 2 performs move block 2 of its moveset (a two second dwell), it is to wait until it receives a start command via the command block to execute move block 3.

Figure 8.11
Profile of Moveset 1 for Axis 1

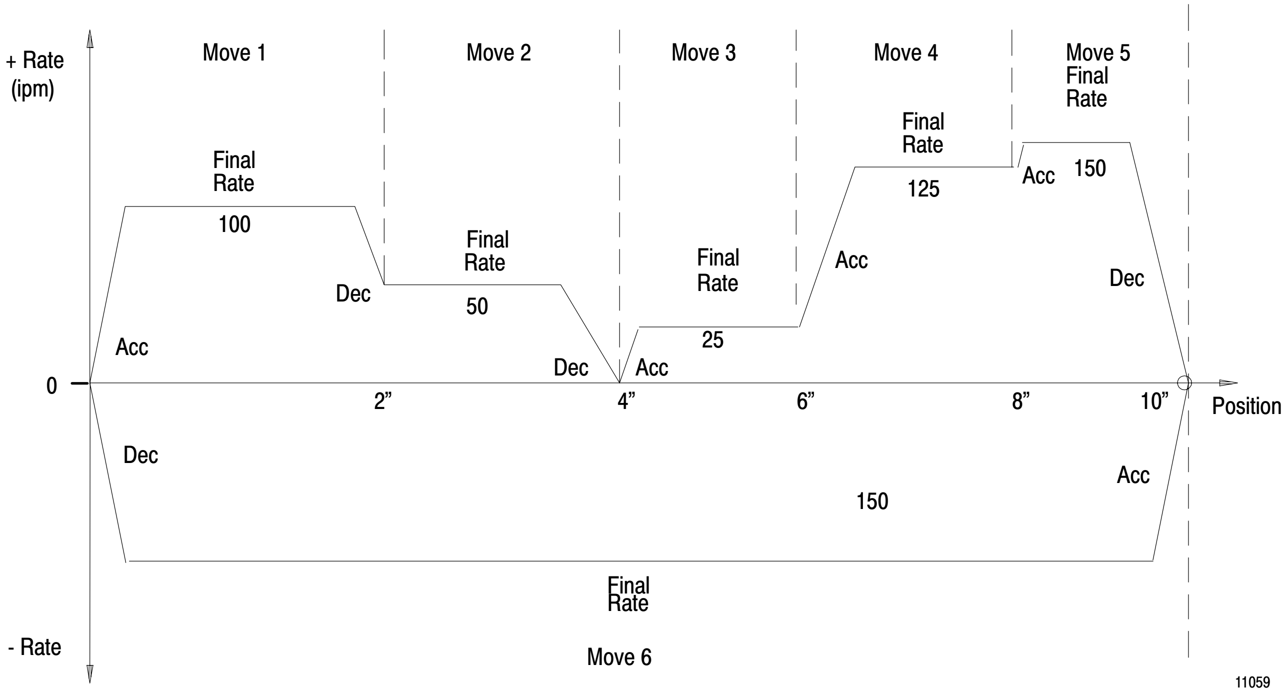


Figure 8.12
Profile of Moveset 2 for Axis 1

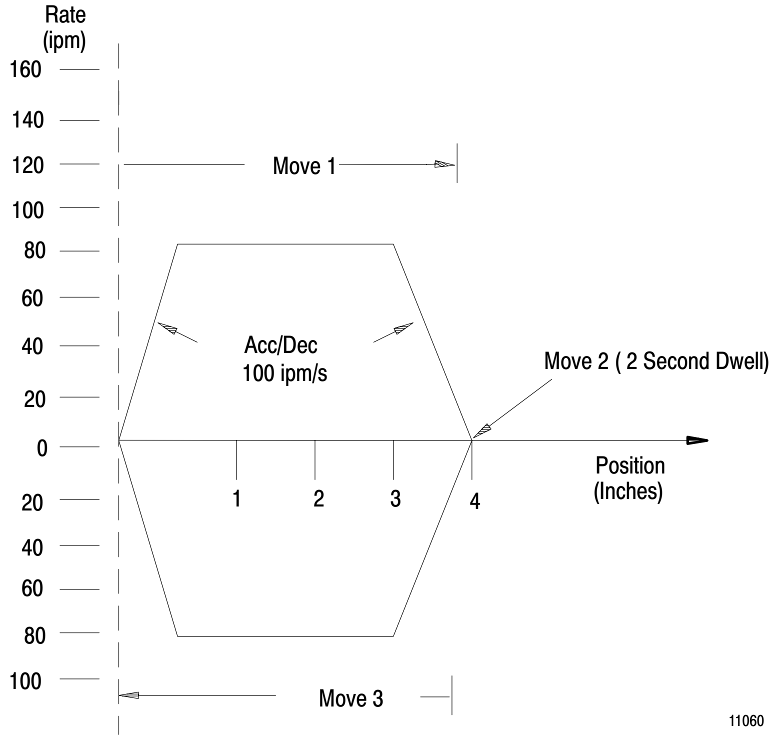


Figure 8.13
Profile of Moveset 3 for Axis 1

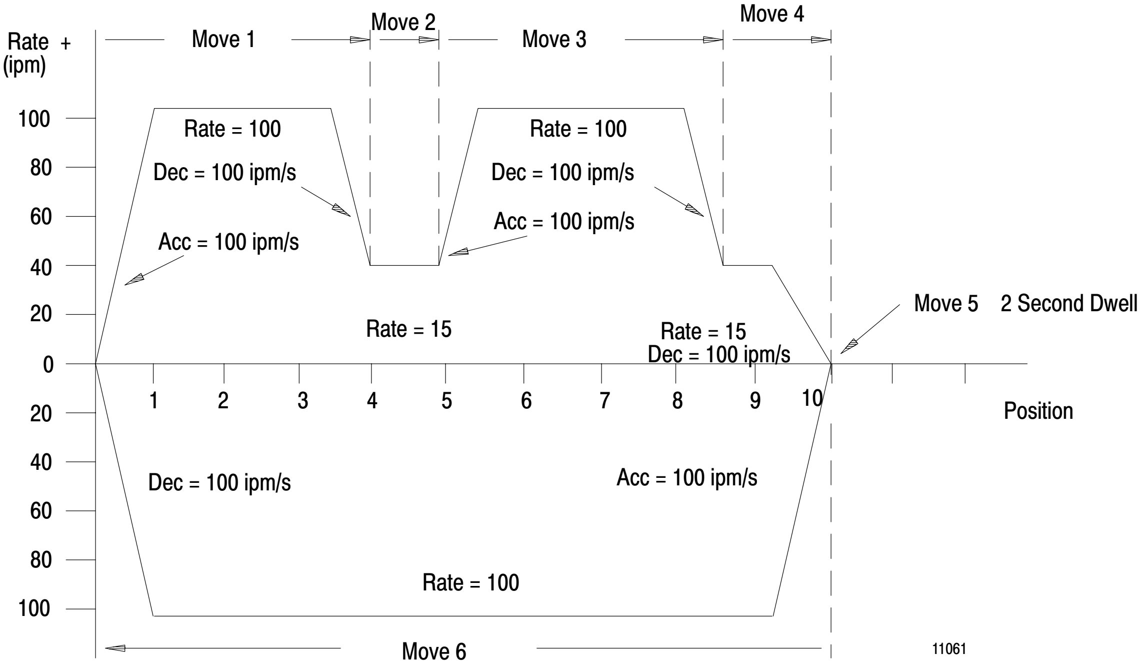
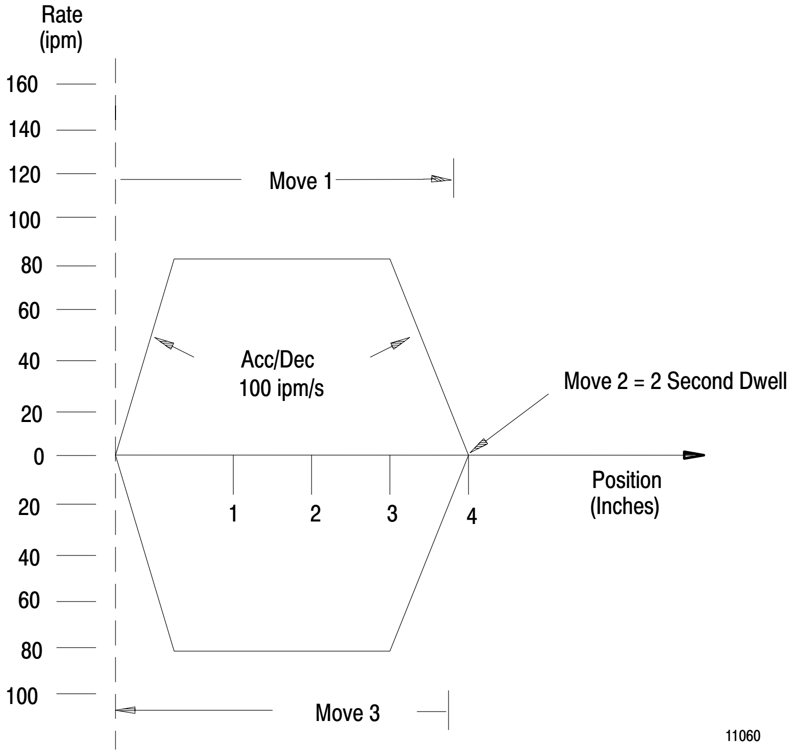


Figure 8.14
Profile of Moveset for Axis 2



Planning the Data Block for PLC-2/30

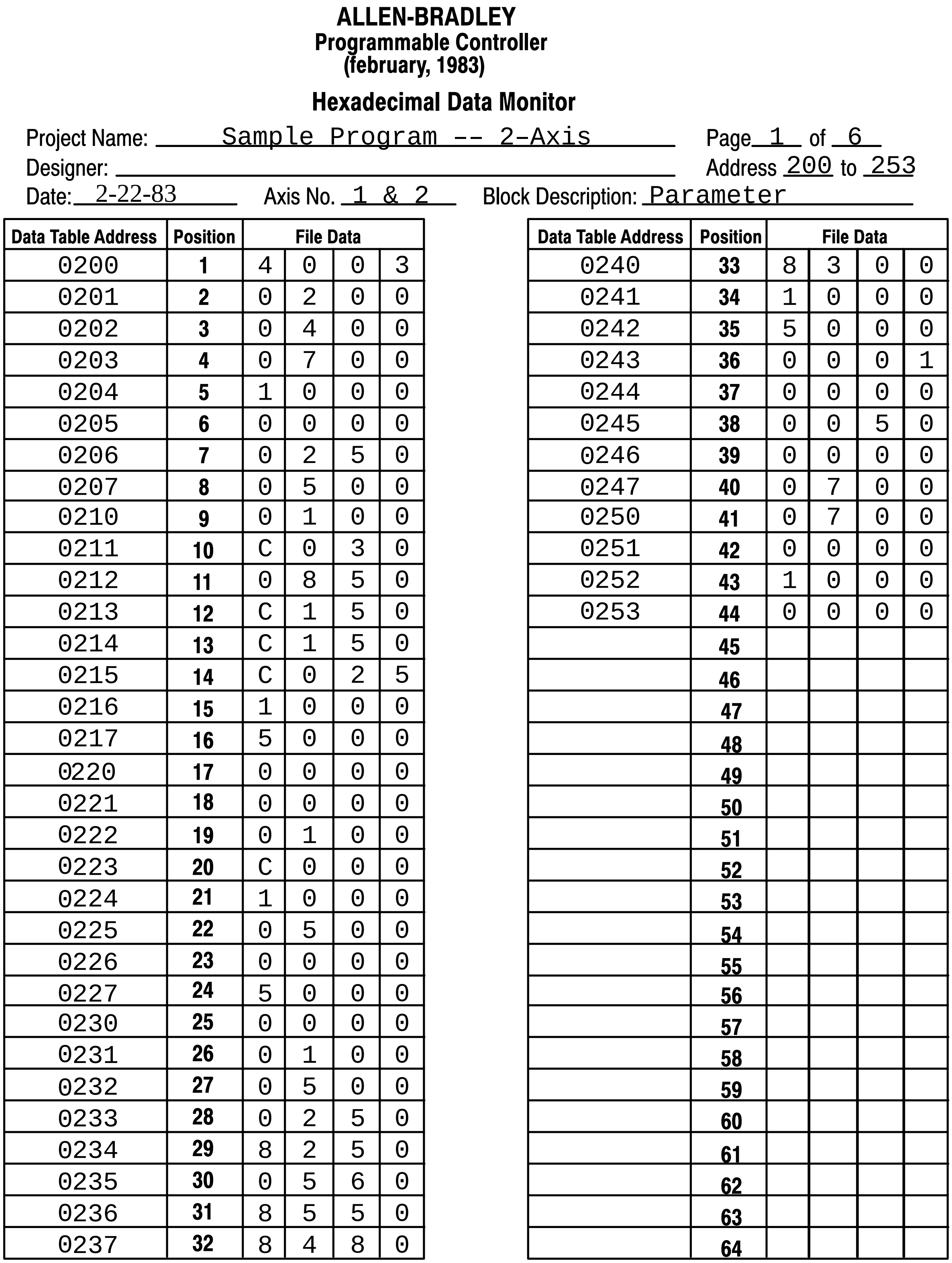
For this example, we assume a PLC-2/30 processor and assign the necessary data blocks to the following data table addresses:

Block	Address
Parameter	200 - 253
Command	400 - 407
Status	447 - 460
Moveset 1 for axis 1	700 - 727
Moveset 2 for axis 1	500 - 516
Moveset 3 for axis 1	600 - 642
Moveset 1 for axis 2	1000 - 1012

Once we have assigned data table addresses, we can plan how to enter values into the parameter, command, and moveset blocks to achieve the desired results. The most convenient way to enter values into the data table is with the industrial terminal in the hexadecimal data monitor mode.

Figure 8.15 shows forms filled in with the hexadecimal values for the parameter, moveset, and command blocks of this example. Note that the axes are independent in this example. Axis 1 motion has no effect on axis 2 motion, and vice versa.

Figure 8.15
Data Table Form Example for 2-Axis Program



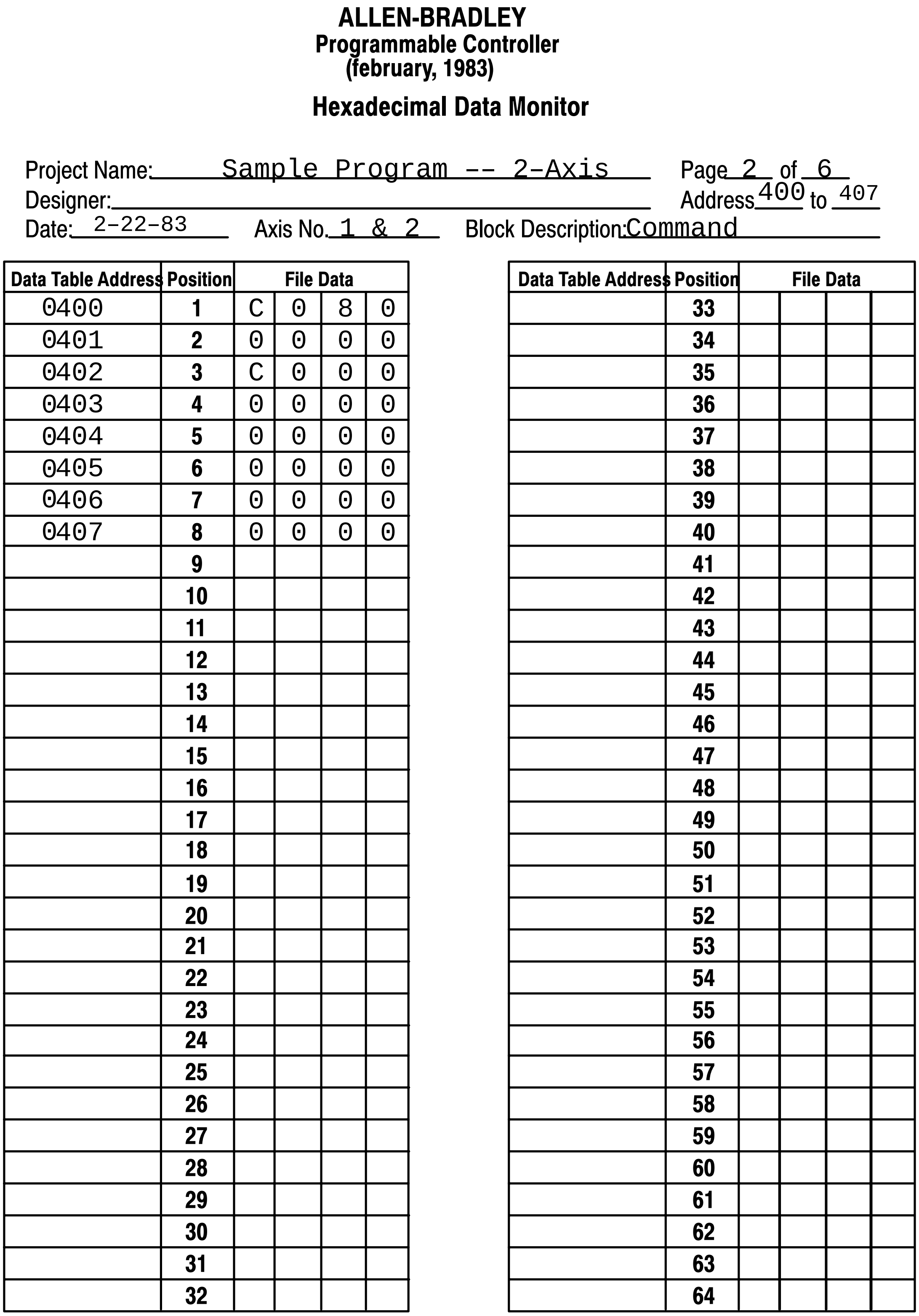
ALLEN-BRADLEY
Programmable Controller
(february, 1983)

Hexadecimal Data Monitor

Project Name: Sample Program -- 2-Axis Page 1 of 6
Designer: Address 200 to 253
Date: 2-22-83 Axis No. 1 & 2 Block Description: Parameter

Data Table Address	Position	File Data			
0200	1	4	0	0	3
0201	2	0	2	0	0
0202	3	0	4	0	0
0203	4	0	7	0	0
0204	5	1	0	0	0
0205	6	0	0	0	0
0206	7	0	2	5	0
0207	8	0	5	0	0
0210	9	0	1	0	0
0211	10	C	0	3	0
0212	11	0	8	5	0
0213	12	C	1	5	0
0214	13	C	1	5	0
0215	14	C	0	2	5
0216	15	1	0	0	0
0217	16	5	0	0	0
0220	17	0	0	0	0
0221	18	0	0	0	0
0222	19	0	1	0	0
0223	20	C	0	0	0
0224	21	1	0	0	0
0225	22	0	5	0	0
0226	23	0	0	0	0
0227	24	5	0	0	0
0230	25	0	0	0	0
0231	26	0	1	0	0
0232	27	0	5	0	0
0233	28	0	2	5	0
0234	29	8	2	5	0
0235	30	0	5	6	0
0236	31	8	5	5	0
0237	32	8	4	8	0
0240	33	8	3	0	0
0241	34	1	0	0	0
0242	35	5	0	0	0
0243	36	0	0	0	1
0244	37	0	0	0	0
0245	38	0	0	5	0
0246	39	0	0	0	0
0247	40	0	7	0	0
0250	41	0	7	0	0
0251	42	0	0	0	0
0252	43	1	0	0	0
0253	44	0	0	0	0
	45				
	46				
	47				
	48				
	49				
	50				
	51				
	52				
	53				
	54				
	55				
	56				
	57				
	58				
	59				
	60				
	61				
	62				
	63				
	64				

Figure 8.15
Data Table Form Eample for 2-Exis Program (continued)



ALLEN-BRADLEY
Programmable Controller
(february, 1983)

Hexadecimal Data Monitor

Project Name: Sample Program -- 2-Axis   Page 2 of 6
Designer: Address 400 to 407
Date: 2-22-83   Axis No. 1 & 2   Block Description: Command

Data Table Address	Position	File Data			
0400	1	C	0	8	0
0401	2	0	0	0	0
0402	3	C	0	0	0
0403	4	0	0	0	0
0404	5	0	0	0	0
0405	6	0	0	0	0
0406	7	0	0	0	0
0407	8	0	0	0	0
	9				
	10				
	11				
	12				
	13				
	14				
	15				
	16				
	17				
	18				
	19				
	20				
	21				
	22				
	23				
	24				
	25				
	26				
	27				
	28				
	29				
	30				
	31				
	32				

Data Table Address	Position	File Data			
	33				
	34				
	35				
	36				
	37				
	38				
	39				
	40				
	41				
	42				
	43				
	44				
	45				
	46				
	47				
	48				
	49				
	50				
	51				
	52				
	53				
	54				
	55				
	56				
	57				
	58				
	59				
	60				
	61				
	62				
	63				
	64				

Figure 8.15
Data Table Form Eample for 2-Exis Program (continued)

ALLEN-BRADLEY
Programmable Controller
(february, 1983)

Hexadecimal Data Monitor

Project Name: Sample Program -- 2-Axis Page 3 of 6
Designer: Address 500 to 531
Date: 2-22-83 Axis No. 1 Block Description: MOVE SET #2

Data Table Address	Position	File Data				
0500	1	0	4	0	3	MSCW
0501	2	A	C	8	0	Move #1
0502	3	0	0	0	4	
0503	4	0	0	0	0	
0504	5	C	1	0	0	
0505	6	8	F	8	0	Move #2
0506	7	0	0	0	0	
0507	8	2	0	0	0	
0510	9	B	C	0	0	Move #3
0511	10	0	0	0	0	
0512	11	0	0	0	0	
0513	12	C	1	0	0	
0514	13	0	1	0	0	
0515	14	0	1	0	0	
0516	15	0	6	0	0	NMSP
	16					
	17					
	18					
	19					
	20					
	21					
	22					
	23					
	24					
	25					
	26					
	27					
	28					
	29					
	30					
	31					
	32					

Data Table Address	Position	File Data			
	33				
	34				
	35				
	36				
	37				
	38				
	39				
	40				
	41				
	42				
	43				
	44				
	45				
	46				
	47				
	48				
	49				
	50				
	51				
	52				
	53				
	54				
	55				
	56				
	57				
	58				
	59				
	60				
	61				
	62				
	63				
	64				

Figure 8.15
Data Table Form Eample for 2-Exis Program (continued)

ALLEN-BRADLEY
Programmable Controller
(february, 1983)

Hexadecimal Data Monitor

Project Name: Sample Program -- 2-Axis　　Page 4 of 6
Designer:　　Address 600 to 642
Date: 2-22-83　　Axis No. 1　　Block Description Move Set #3

Data Table Address	Position	File Data				
0600	1	0	1	0	6	MSCW
0601	2	B	C	8	0	Move #1
0602	3	0	0	0	4	
0603	4	0	0	0	0	
0604	5	C	1	0	0	
0605	6	0	1	0	0	
0606	7	0	1	0	0	
0607	8	8	C	8	0	Move #2
0610	9	0	0	0	5	
0611	10	0	0	0	0	
0612	11	C	0	1	5	
0613	12	0	1	0	0	
0614	13	0	1	0	0	
0615	14	B	C	8	0	Move #3
0616	15	0	0	0	9	
0617	16	0	0	0	0	
0620	17	C	1	0	0	
0621	18	0	1	0	0	
0622	19	0	1	0	0	
0623	20	B	C	8	0	Move #4
0624	21	0	0	1	0	
0625	22	0	0	0	0	
0626	23	C	0	1	5	
0627	24	0	1	0	0	
0630	25	0	1	0	0	
0631	26	8	F	0	0	Move #5
0632	27	0	0	0	0	
0633	28	2	0	0	0	
0634	29	F	C	8	0	Move #6
0635	30	0	0	0	0	
0636	31	0	0	0	0	
0637	32	C	1	0	0	

Data Table Address	Position	File Data			
0640	33	0	1	0	0
0641	34	0	1	0	0
0642	35	0	7	0	0
	36				
	37				
	38				
	39				
	40				
	41				
	42				
	43				
	44				
	45				
	46				
	47				
	48				
	49				
	50				
	51				
	52				
	53				
	54				
	55				
	56				
	57				
	58				
	59				
	60				
	61				
	62				
	63				
	64				

Figure 8.15
Data Table Form Eample for 2-Exis Program (continued)

ALLEN-BRADLEY
Programmable Controller
(february, 1983)

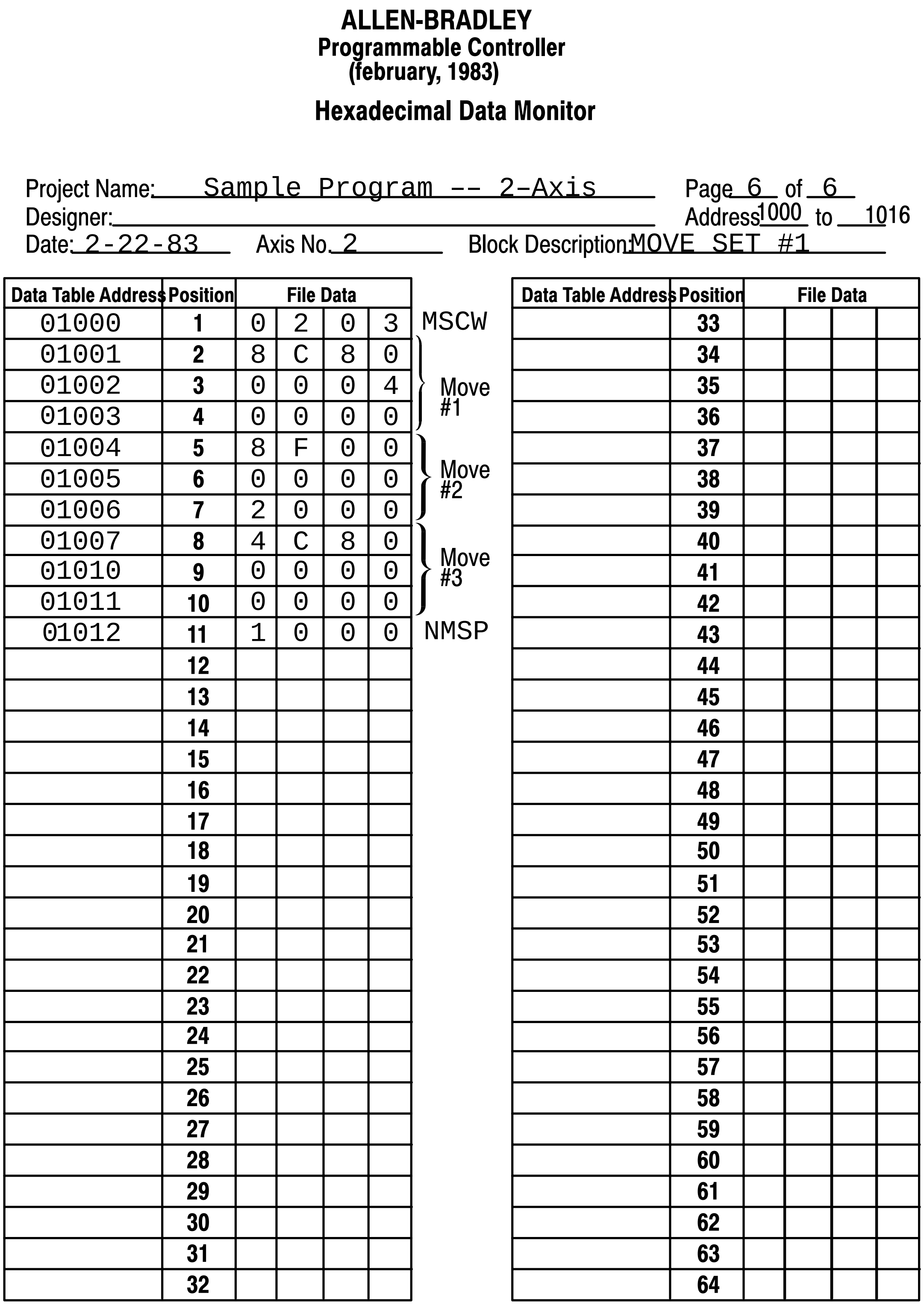
Hexadecimal Data Monitor

Project Name: Sample Program -- 2-Axis Page 5 of 6
Designer: Address 700 to 727
Date: 2-22-83 Axis No. 1 Block Description: Move Set #1

Data Table Address	Position	File Data				
0700	1	0	1	0	6	MSCW
0701	2	A	C	8	0	Move #1
0702	3	0	0	0	2	
0703	4	0	0	0	0	
0704	5	C	1	0	0	
0705	6	2	C	8	0	SMCW
0706	7	0	0	0	4	Move #2
0707	8	0	0	0	0	
0710	9	C	0	5	0	
0711	10	A	C	8	0	Move #3
0712	11	0	0	0	6	
0713	12	0	0	0	0	
0714	13	C	0	2	5	
0715	14	A	C	8	0	Move #4
0716	15	0	0	0	8	
0717	16	0	0	0	0	
0720	17	C	1	2	5	
0721	18	0	C	8	0	Move #5
0722	19	0	0	1	0	
0723	20	0	0	0	0	
0724	21	0	C	8	0	Move #6
0725	22	0	0	0	0	
0726	23	0	0	0	0	
0727	24	0	5	0	0	NMSP
	25					
	26					
	27					
	28					
	29					
	30					
	31					
	32					

Data Table Address	Position	File Data			
	33				
	34				
	35				
	36				
	37				
	38				
	39				
	40				
	41				
	42				
	43				
	44				
	45				
	46				
	47				
	48				
	49				
	50				
	51				
	52				
	53				
	54				
	55				
	56				
	57				
	58				
	59				
	60				
	61				
	62				
	63				
	64				

Figure 8.15
Data Table Form Example for 2-Axis Program (continued)



ALLEN-BRADLEY
Programmable Controller
(february, 1983)

Hexadecimal Data Monitor

Project Name: Sample Program -- 2-Axis Page 6 of 6
Designer: Address 1000 to 1016
Date: 2-22-83 Axis No. 2 Block Description: MOVE SET #1

Data Table Address	Position	File Data					Data Table Address	Position	File Data			
01000	1	0	2	0	3	MSCW		33				
01001	2	8	C	8	0	Move #1		34				
01002	3	0	0	0	4			35				
01003	4	0	0	0	0			36				
01004	5	8	F	0	0	Move #2		37				
01005	6	0	0	0	0			38				
01006	7	2	0	0	0			39				
01007	8	4	C	8	0	Move #3		40				
01010	9	0	0	0	0			41				
01011	10	0	0	0	0			42				
01012	11	1	0	0	0	NMSP		43				
	12							44				
	13							45				
	14							46				
	15							47				
	16							48				
	17							49				
	18							50				
	19							51				
	20							52				
	21							53				
	22							54				
	23							55				
	24							56				
	25							57				
	26							58				
	27							59				
	28							60				
	29							61				
	30							62				
	31							63				
	32							64				

You can directly convert between hexadecimal and binary as follows:

Binary	Hexadecimal
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9
1010	A
1011	B
1100	C
1101	D
1110	E
1111	F

Hexadecimal digits 0 thru 9 are the same as decimal digits 0 thru 9. Therefore, if a word is to contain only a BCD value, you can enter the decimal digits directly as hexadecimal digits.

If you make an individual selection with each bit of a word, write down the binary value and convert it to hexadecimal. For example:

binary	1001	1101	0100	0011
converts to hexadecimal	9	D	4	3

If a word is to contain a combination of a BCD value and individual bit selections:

- write down the BCD value
- fill in the individual bit selections
- convert the combined binary value to hexadecimal

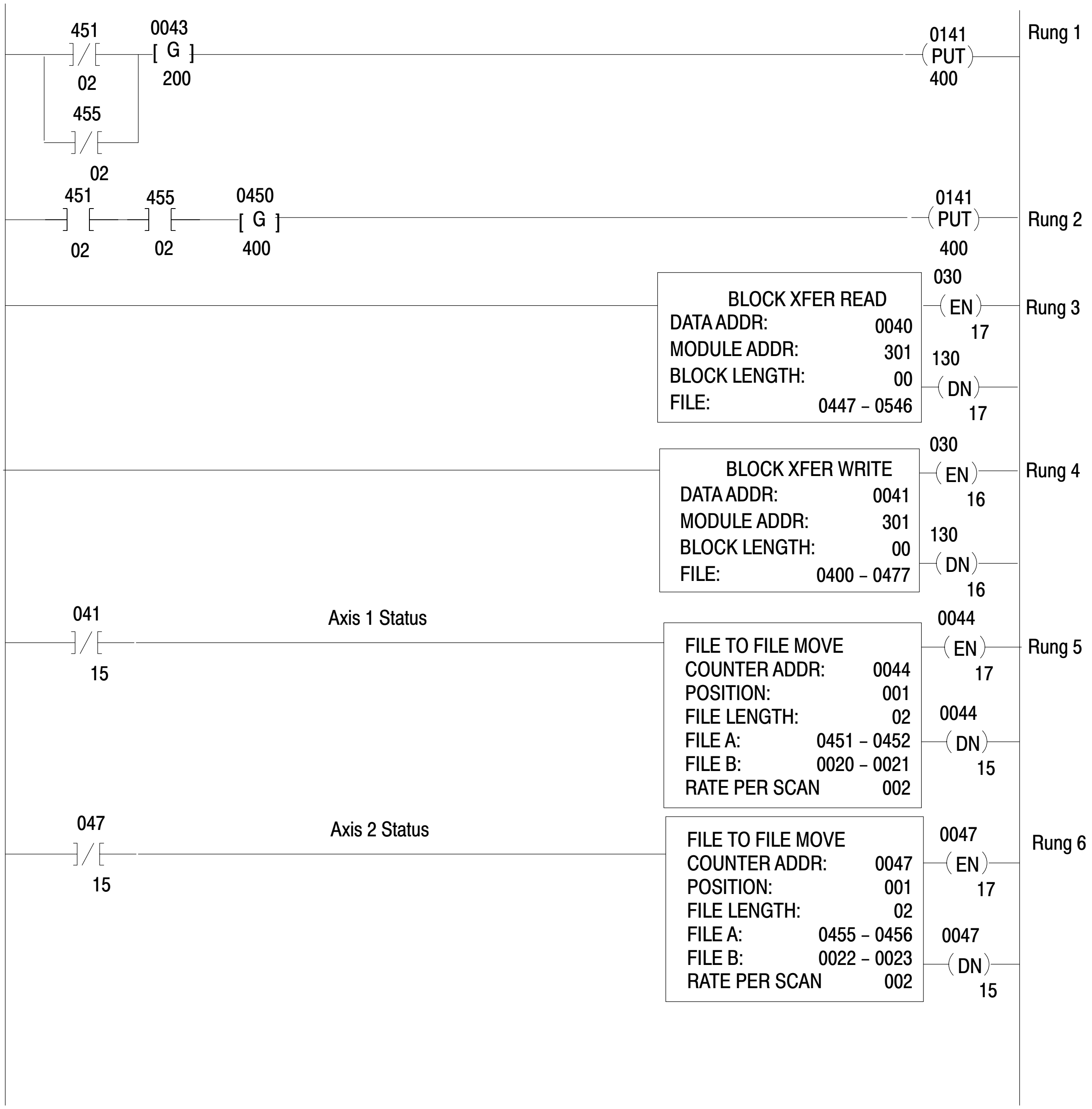
For example, if a word is to have bits 17, 16, and 15 on plus the decimal value 1972, you would:

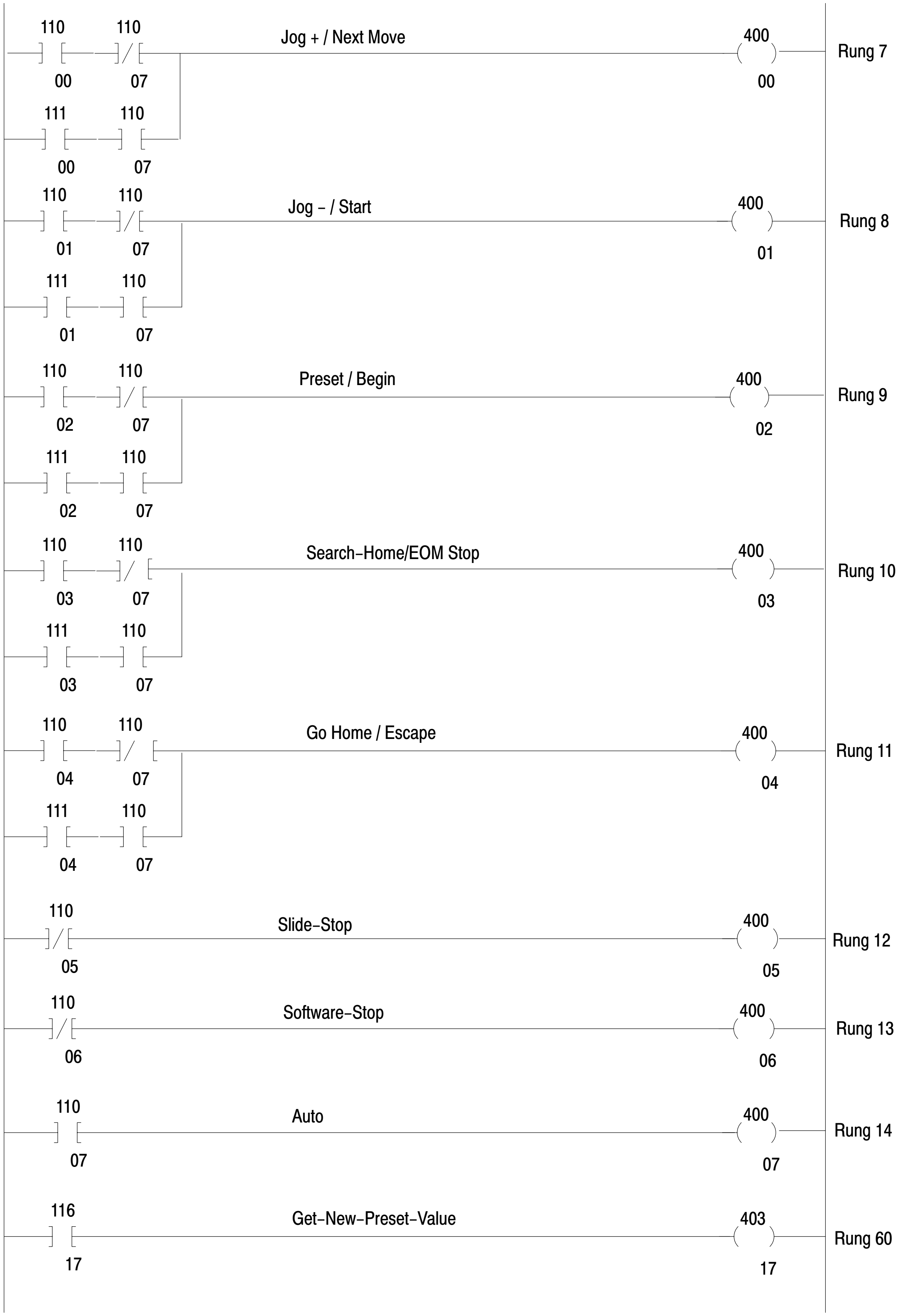
- write down the BCD value of 1972 as:
  _____1 1001 0111 0010
- fill in bits 17, 16, and 15 as:
  1111 1001 0111 0010
- convert it to hexadecimal as:
  F   9   7   2

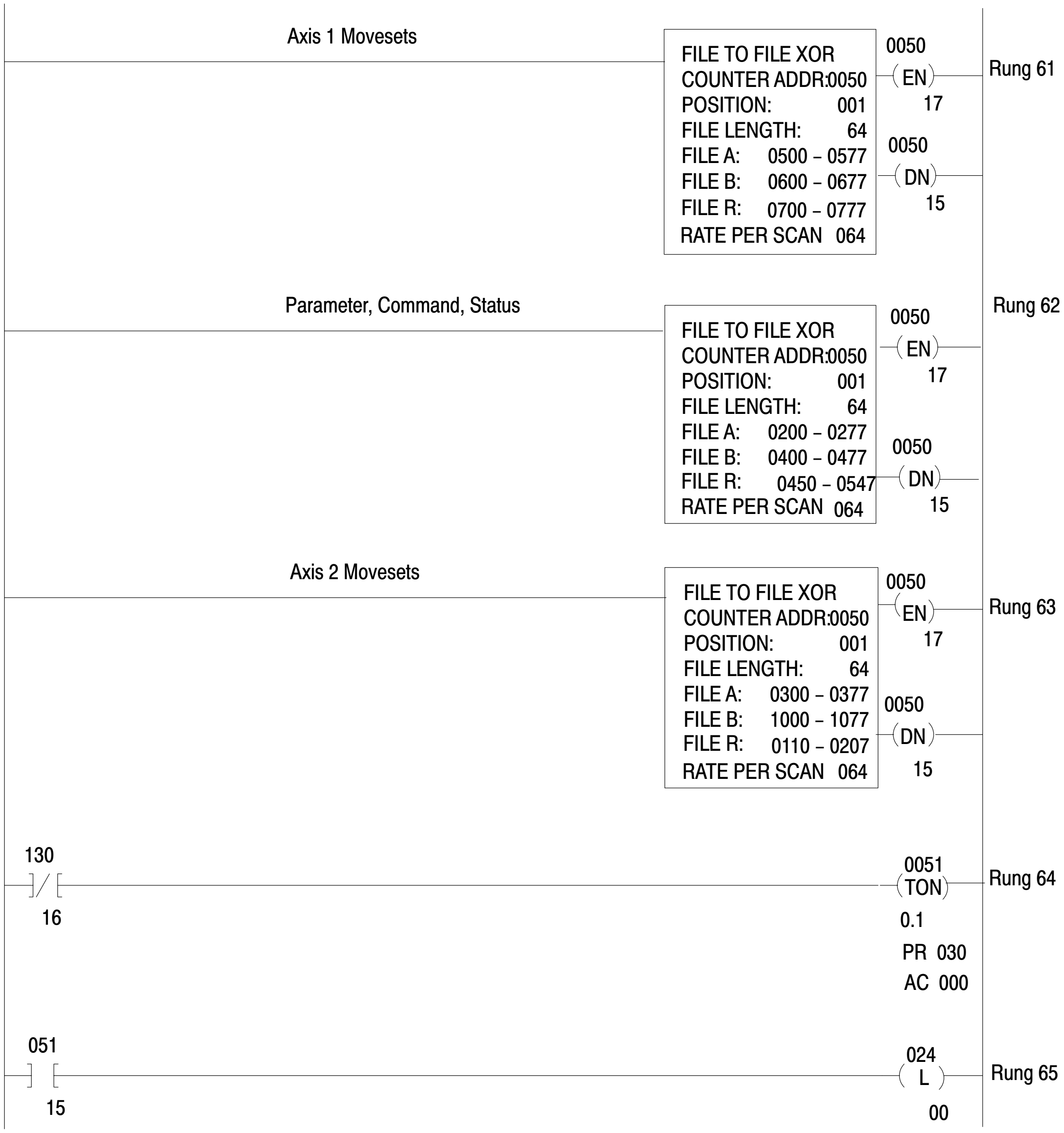
Program Rungs for PLC-2/30

Figure 8.16 shows the ladder diagram programming for this application for a PLC-2/30 system. Rungs 1 thru 4 of the program implement bidirectional block transfer. The remaining rungs are for data input and display. Here are individual rung descriptions:

Figure 8.16
PLC-2/30 Ladder-diagram Programming Example for Controlling 2 Axes




110
110
Jog + / Next Move
400
Rung 7
00
07
00
111
110
00
07
110
110
Jog – / Start
400
Rung 8
01
07
01
111
110
01
07
110
110
Preset / Begin
400
Rung 9
02
07
02
111
110
02
07
110
110
Search–Home/EOM Stop
400
Rung 10
03
07
03
111
110
03
07
110
110
Go Home / Escape
400
Rung 11
04
07
04
111
110
04
07
110
Slide–Stop
400
Rung 12
05
05
110
Software–Stop
400
Rung 13
06
06
110
Auto
400
Rung 14
07
07
116
Get–New–Preset–Value
403
Rung 60
17
17



Rung 1

Rung 1 assures transfer of the parameter block at power-up. This rung examines the ready bits (45102 and 45502) in the status block. The parameter block file address (200) is stored as a constant in storage word 0043. At power up, the parameter block automatically transfers to the 1771-M3 controller. If the parameter block is valid, the 1771-M3 controller turns on the ready bits (45102 and 45502) in the status block, thus inhibiting rung 1.

Rung 2

If both axes are ready (bits 45102 and 45502 both on), this rung gets the address pointer in the second word of the status block (0450) and puts it in the file-address word (0141) for the write-block-transfer instruction. The address pointer in the status block contains the address for the parameter, command, or a moveset block, as requested by the 1771-M3 controller. Because of the action of this rung, the block of data requested by the 1771-M3 controller is written to the 1771-M3 controller when the write-block-transfer instruction executes.

Rung 3

Rung 3 reads the status block from the 1771-M3 controller. The data address (0040) is located in the timer/counter accumulated area of the data table. The module location address (301) indicates that the 1771-M3 controller is in rack 3, module group 0, and the right slot of the module group. Block length (00) is the default length. This allows the 1771-M3 controller to control the number of words transferred. During a read operation, data loads into consecutive words starting with the designated address (0447).

Rung 4

Rung 4 writes the parameter, command, or moveset block to the 1771-M3 controller module, as requested by the 1771-M3 controller via the status block. The data address (0041) is in the timer/counter accumulated area of the data table. Module address (301) is explained in the rung 3 description. The block length (00) is the default length. This allows the 1771-M3 controller to control the number of words transferred according to the block it requests.

During a write operation, data transfers from consecutive data table words starting with the first word of the parameter, command, or moveset block. These blocks have starting addresses 0200, 0400, and 0500, respectively. Other moveset blocks start at addresses 0600 and 0700.

Rung 5

Rung 5 transfers the third and fourth words of the status block to digital outputs in module groups 0 and 1 of rack 2. We use these outputs to display the position, following error, and diagnostic codes for axis 1.

Rung 6

Rung 6 transfers the seventh and eighth words of the status block to digital outputs in module groups 2 and 3 of rack 2. We use these outputs to display the position, following error, and diagnostic codes for axis 2.

Rungs 7 thru 60

Rungs 7 thru 60 use discrete inputs to individually control bits of control words 1 and 2 for each axis in the command block. You would not need a rung to control bit 16 (tachometer calibrate) of axis control word 2, because you only need to turn on this bit when you calibrate the tachometer input. Also, you would not use rungs to control bits 17, 16, 15, and 14 (control word 1 ID) of axis control word 2, because you must always have them set to the binary value 1100.

Rung 61

Rung 61 performs no logical function in the program. However, this rung lets you display and enter values for all three moveset blocks for axis 1. You can display all three blocks at the same time on the industrial terminal by using its display function.

Rung 62

Rung 62 performs no logical function in the program. However, this rung lets you display the parameter, command, and status block and enter values for the parameter and command block. You can display all three blocks at the same time on the industrial terminal by using its display function.

Rung 63

Rung 63 performs no logical function in the program. However, this rung lets you display and enter values for the moveset block for axis 2. You can display the block on the industrial terminal by using its display function.

Rungs 64 and 65

Rungs 64 and 65 are block transfer timeout rungs. If a block transfer is not completed within three seconds, bit 15 of TON 0051 goes on, causing output 02400 to be latched on. This output can be used to turn on a warning device.

Planning Data Blocks for PLC-3

With a PLC-3 processor, the most straightforward way to arrange the data blocks is to put them all in the same file and use the address pointers to specify word offsets within the file as follows:

Block	Word
Status	1 - 10
Parameter	101 - 144
Moveset 1 for axis 1	201 - 224
Moveset 2 for axis 1	301 - 315
Moveset 3 for axis 1	401 - 435
Moveset 1 for axis 2	501 - 511
Command	601 - 608

Remember that the 1771-M3 controller will not accept an address pointer of 000. Therefore, never start a block at word 0.

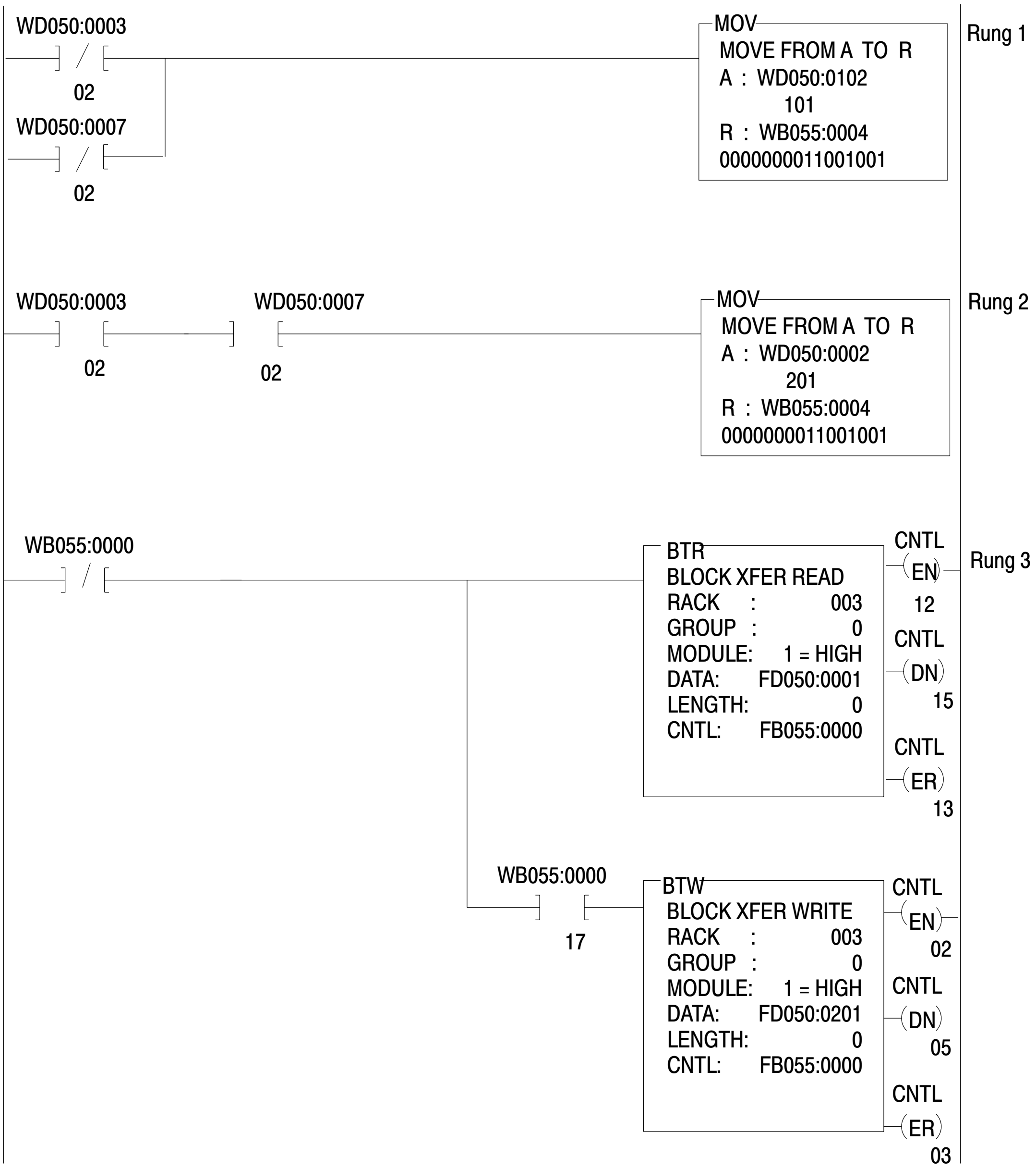
In this example, we started the blocks at words 1, 101, 201, 301, 401, 501 and 601 so that the numbers of their words would correspond to their descriptions in chapter 7.

In this example, we used a file in the decimal (BCD) section of the data table to keep the ladder-diagram program simple. Remember that the 1771-M3 controller will only accept BCD values for address pointers.

Program Rungs for PLC-3

Figure 8.17 shows ladder diagram programming rungs for this application for a PLC-3 system. The three rungs in Figure 8.17 perform the same function as the first four rungs in Figure 8.16.

Figure 8.17
PLC-3 Ladder-diagram Programming Example Rungs for Controlling 2 Axes



Rung 1

Rung 1 assures transfer of the parameter block at power up. This rung examines the ready bits (WD050:0003/02 and WD050:00007/02) in the status block. The parameter block address pointer (101) is stored in the second word (102) of the parameter block. At power up, the parameter block automatically transfers to the 1771-M3 controller. If the parameter block is valid, the 1771-M3 controller turns on the ready bits (WD050:0003/02) and WD050:0007/02) in the status block, thus inhibiting rung 1.

Rung 2

If both axes are ready (bits WD050:0003/02 and WD050:0007/02 both on), this rung moves the address pointer in the second word of the status block (WD050:0102) to the control file addresses word (WB055:0004) for the write-block-transfer instruction. The address pointer in the status block contains the word offset for the parameter, command, or a moveset block as requested by the 1771-M3 controller when the write-block-transfer instruction is executed.

Rung 3

Rung 3 controls both the block-transfer read and the block-transfer write. The examine-on instructions for the done bit (WB055:0000/15 and /05) and the block-transfer-read request bit (WB055:0000/17) assures that the read alternates with the write. The data file address for the block-transfer read is always the address of the status block. The data file address of the block-transfer write is controlled by rungs 1 and 2. The control file address must always be in the binary section of the data table.

In addition to the rungs in Figure 8.17, you need a rung to clear the block-transfer control word, as shown in Figure 8.10.

You also need rungs to perform the logic of rungs 5 thru 60, 64, and 65 in Figure 8.16.

Summary

In this chapter, we told you about block-transfer instructions and block-transfer timing. We also described programming examples. However, you must not run an axis with your program until you first follow the axis integration procedures we give you in chapter 9.

Chapter 9

Integrating Axes

Chapter Objectives

You must perform the procedures in this chapter before you have the servo positioning assembly in service: that is to help ensure that axes respond correctly to commands from the servo positioning assembly, and that adequate feedback is provided to the 1771-ES expander module.

You **must** enter the parameter block before you perform these integration procedures. In addition, the ladder diagram program for the axis must be loaded in the PC.

Important: For these procedures to work, servo drives and motors **must** be capable of controlling axis motion according to your requirements. The servo positioning assembly **cannot overcome inherent limitations** of drives, motors, or axis mechanisms.

We present axis integration procedures in the order in which you must perform them:

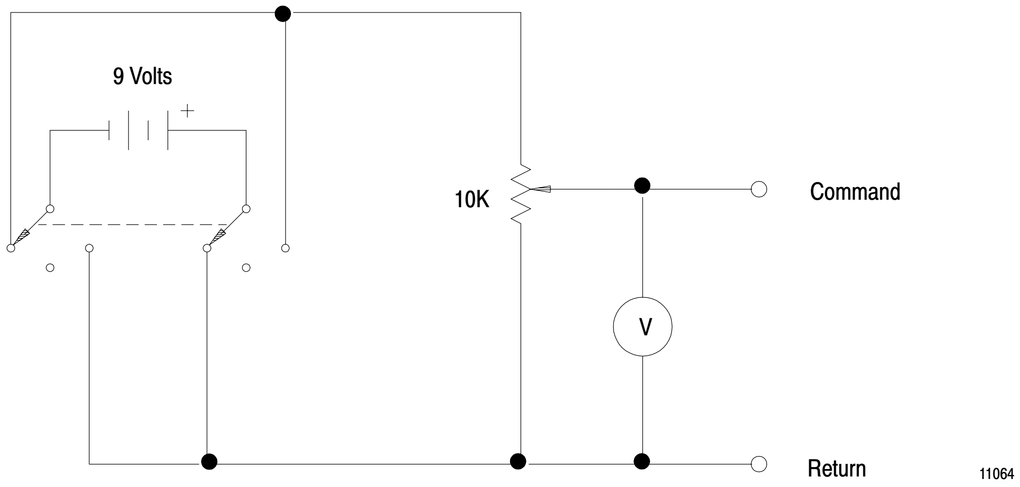
- **Open-Loop Procedure** - You disconnect the 1771-ES expander from the servo drive to open the positioning loop, and use a battery box to supply drive input. You check phasing of drive input and axis feedback. You check axis motion for smoothness and response.
- **Closed-Loop Procedure** - You close the axis positioning loop, and check axis response to commands from the 1771-ES expander.
- **Tachometer Calibration** - You must perform this procedure to ensure proper functioning of the loss-of-feedback detection feature.

Open-Loop Procedure

The following procedure requires a battery box to supply command voltage to the axis drive. If a commercial battery box is not available, you can make one according to the circuit of Figure 9.1. A 9-volt battery, a 10K ohm variable resistor, and a 2-pole 3-position switch are required. (You can use a dpdt toggle switch **with a center off position**.)

You can connect a voltmeter across the battery box output as shown to measure output voltage.

Figure 9.1
Diagram of Battery Box



The following steps form the open-loop axis integration **procedure**:

1. Disconnect power to the servo positioning assembly and servo drives.

2. Remove servo drive fuses to ensure that the servo drives are disabled.

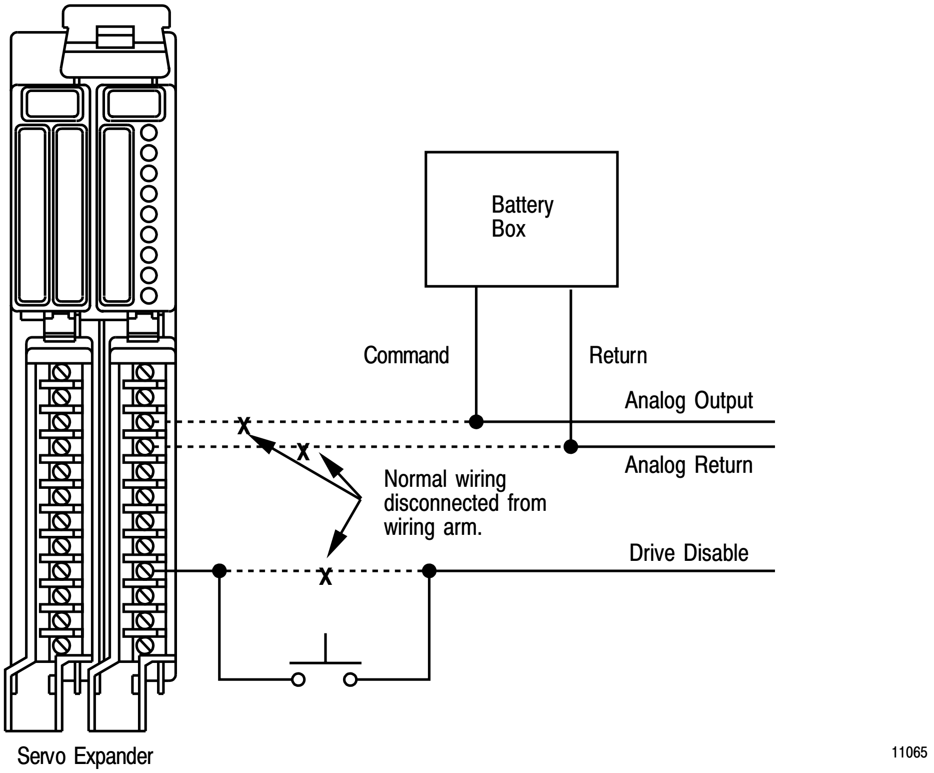
3. Verify that tachometer leads are correctly connected **at the servo drive**. The velocity feedback signal from the tachometer must be the opposite polarity to the velocity command signal.

WARNING: The velocity feedback loop **must** be closed at the servo drive. If the tachometer leads are reversed, or if either is disconnected, sudden high speed axis motion can occur, which can result in damage to equipment and/or injury to personnel.

4. Verify that axis overtravel limit switches are operational.

5. Disconnect the drive-disable lead from the right wiring arm on the 1771-ES expander (terminal 9 or 10, depending on type of circuit). Wire one side of a normally-open, momentary-contact switch to the drive disable terminal. Wire the other side of the switch to the drive disable lead that goes to the servo drive (Figure 9.2). To move the axis with the battery box, you must hold this switch closed.

Figure 9.2
Connections for Open-loop Testing



WARNING: To guard against possible injury, keep all personnel clear of the axis. In addition, have a competent person standing by to disconnect axis servo motor power if necessary.

If you hardwired an emergency stop circuit like that of Figure 6.8, the loop-contactor relay will disable the servo motor when one of the switches in the E-Stop string is opened. However, you must disconnect the drive-disable line for open-loop drive operation.

6. Disconnect the analog output and analog return leads from the 1771-ES expander wiring arm. Zero the output of the battery box, then connect its command output lead to the analog output lead just disconnected from terminal 3. Connect the battery box return lead to the analog return lead just disconnected from terminal 4 (Figure 9.2). Leave the wiring arm up, connected to the module.

⚠ **CAUTION:** Zero the output of the battery box before connecting it to a servo drive. Sudden application of command voltage to the drive could damage equipment. Axis speed must increase gradually.

7. Disconnect the servo motor from the leadscrew.

8. Replace the axis fuses, then re-apply power to the axis and servo positioning assembly. (Leave de-energized those drives for axes you are not integrating.)

9. Turn off bit 15 (enable-loss-of-feedback detection) of the home position value word in the parameter block.

10. While leaving the switch to the drive-disable line **open**, adjust the battery box output **away** from zero. If you can get the servo motor to rotate, there is something wrong with the drive-disable connections or the servo drive.

11. Adjust the battery box output **to zero**. The **close** the switch to the drive-disable line. If the motor starts to accelerate, the leads from the tachometer to the servo drive are reversed or disconnected. If the motor rotates without acceleration, you may need to adjust the drive balance, refer to the servo drive manufacturer’s instructions.

12. Disconnect power to the servo positioning assembly and servo drives.

13. Remove servo drive fuses to ensure that the servo drives are disabled.

14. Connect the servo motor to the leadscrew.

15. Replace the axis drive fuses, then re-apply axis and servo positioning assembly power. (Leave de-energized those drives for axes you are not integrating.)

16. While holding the switch in the drive disable line closed, adjust battery box output to move the axis slowly in each direction. Check for correct wiring of the analog output leads:

- Positive command voltage must move the axis in the positive direction.
- Negative command voltage must move the axis in the negative direction.

If phasing is incorrect, reverse the command connections either at the servo drive or at terminals 3 and 4 of the right servo expander wiring arm (**not both**).

CAUTION: Keep the axis near its center of travel. Running the axis into its mechanical stops could damage equipment.

If you inadvertently run an axis far enough that it trips an overtravel limit switch, causing emergency stop, shut power off, manually back the axis off the limit switch, then issue a reset command thru the command block to re-initialize the axis.

17. Run the axis at increasing speeds in both directions. Check for smooth axis motion. There should be no mechanical vibration or cogging. If there is, take appropriate corrective action.

18. Verify that 1/2 maximum servo output voltage (1/2 rapid traverse voltage) causes axis motion at approximately 1/2 maximum speed (1/2 rapid traverse speed). Check this for both directions of axis motion. You may have to adjust the servo drive to satisfy this requirement. Refer to the servo drive manufacturer’s instructions.

19. Turn off bits 11 and 15 of axis control word 2 in the command block. This selects current position readout, so the last two words of the status block indicate axis position.

20. While monitoring axis position in the status block, move the axis slowly in both directions. When the axis moves in the positive direction, axis position should increase in the positive direction. When the axis moves in the negative direction, axis position should change in the negative direction.

21. As the axis crosses the zero position while moving in the positive direction, the sign bit (bit 17) of the most significant axis position word should change from on (negative) to off (positive), and current position word values should begin increasing.

22. As the axis crosses the zero position while moving in the negative direction, the sign bit (bit 17) of the most significant axis position word in the status block should change from off (positive) to on (negative), and the current position word values should begin increasing.

23. If you obtain opposite results, interchange the channel A connections with the channel B encoder connections at the left wiring arm of the 1771-ES expander module. Repeat steps 20, 21, and 22 to ensure that you have corrected the problem.

This completes the open loop drive procedure. Repeat this procedure for all axes being integrated.

Closed-Loop Procedure

Follow the procedure presented in this section to close the axis positioning loop. You **must** perform this procedure in conjunction with the servo drive manufacturer’s instructions.

This procedure is **not** independent of the open-loop integration procedure. You must perform the open-loop drive procedure before you can perform the closed-loop procedure.

The following steps are the **closed-loop integration procedure**.

1. Turn off the axis power and remove drive fuses.

2. Disconnect the battery box from the analog output and analog return leads. Connect the analog output lead to terminal 3 of the right module wiring arm. Connect the analog return lead to terminal 4.

3. Disconnect the switch from the drive disable terminal and lead. Reconnect the drive disable lead to the drive disable terminal on the right module wiring arm (terminal 9 or 10, depending on type of circuit).

WARNING: To guard against injury to personnel and damage to equipment, the loop-contactor relay **must** remove the servo motor power when an emergency stop or overtravel limit switch opens. Refer to chapter 6 for information about power distribution.

4. Ensure that extreme overtravel limit switches are connected in series with the loop-contactor relay (chapter 6.)

5. Re-apply servo drive and servo positioning assembly power.

6. Verify that there is no axis motion. If necessary, adjust drive balance (at the servo drive) so axis following error is zero and no axis motion occurs when the 1771-ES expander is not commanding axis motion (analog output voltage is zero volts).

7. Turn on bit 15 (select readout) in the second control word for the axis so the status block provides following error information.

8. Jog the axis at a speed below the gain-break point. Record the following error. Calculate to see if the following error equals the feed rate divided by the initial gain. If it does not, adjust either the gain of the servo drive or the initial gain and the rapid traverse rate to achieve the proper performance.

9. Jog the axis in the positive direction at about 1/2 rapid traverse speed and note following error. Record the following error.

10. Jog the axis in the negative direction at the same rate used for step 9. Again record the following error. If following error is not the same as that recorded in step 9, lower one of the D/A voltage values in the parameter block to compensate. Repeat this step to verify the correction. Following error should be the same for both directions.

11. Observe following error for axis motion in both directions at various speeds by jogging the axis. At each speed, following error should be the same for axis motion in both directions.

12. Jog the axis back and forth within its range of travel. Use feedrate override to vary axis speed. Verify that axis motion is smooth and stable at all speeds, including rapid traverse, in both directions. If it is not, check parameter block values for initial gain, in-position band, gain break speed, and gain reduction factor. These parameters can influence axis stability and positioning accuracy and may require minor adjustment at this point. If necessary, adjust the servo drive according to the manufacturer's instructions to obtain the desired results.

Repeat this closed-loop integration procedure for each axis.

Tachometer Calibration

For the loss-of-feedback detection feature to function correctly, you must calibrate the 1771-ES expander. Follow these steps:

1. Turn on bit 15 (enable loss-of-feedback detection) of the home position value word in the parameter block (word 17 for axis 1, word 36 for axis 2, word 55 for axis 3).

⚠ **WARNING:** Once you have completed the axis integration procedures, **never** turn this bit **off**. Without loss-of-feedback detection, if encoder or tachometer feedback is lost, unexpected axis motion can occur, resulting in damage to equipment and/or injury to personnel.

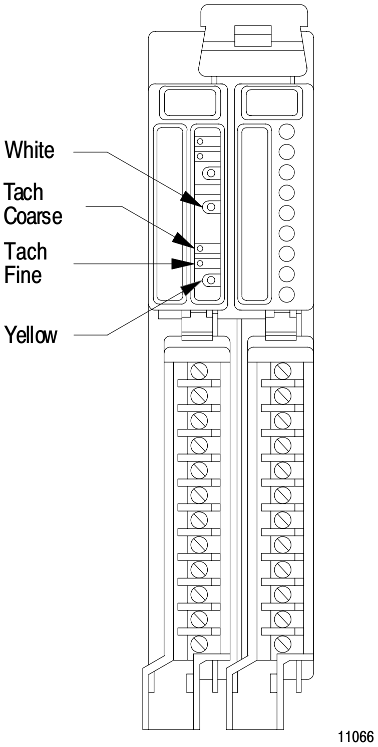
2. Turn on the tachometer calibrate bit (bit 16) of axis control word 2 in the command block.

3. Set the tach fine potentiometer on the 1771-ES expander to its fully clockwise position. Set the tach coarse potentiometer to its fully clockwise position (Figure 9.3).

4. Jog the axis in either direction at maximum speed (rapid traverse rate with 127% feedrate override). If you don't have enough axis travel in which to adjust the potentiometer, disconnect the servo motor from the leadscrew as in the open-loop procedure (see section titled "Open-loop Procedure," in this chapter). Set the potentiometer (see section titled "Connecting the Tachometer," in chapter 6) for a 50V maximum tachometer signal at the 1771-ES expander.

5. If the tachometer voltage is less than 10V, enter a conversion factor in the last word of the parameter block. The conversion factor multiplied by the full scale velocity command voltage must be a value less than the tachometer voltage.

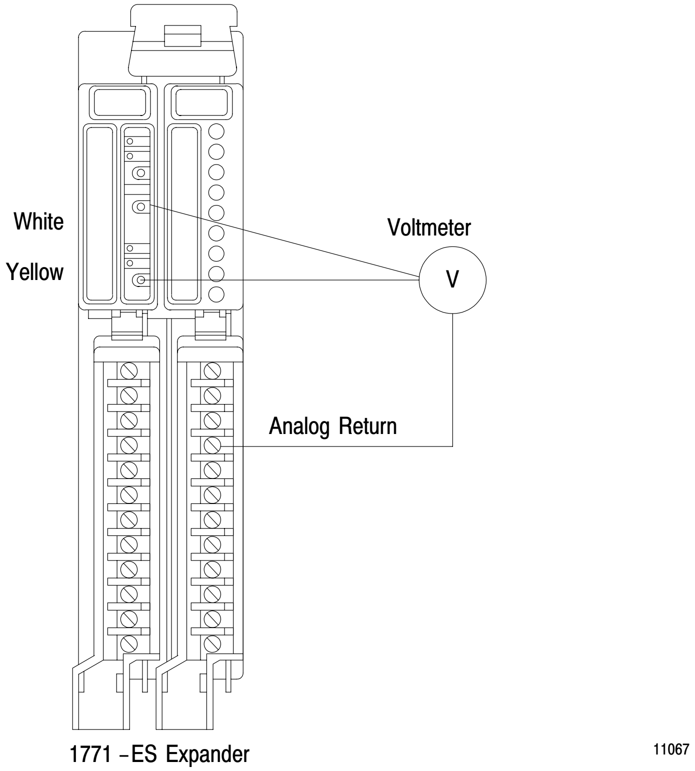
Figure 9.3
1771-ES Expander Test Points and Potentiometers



6. With power off, remove the potentiometer from between the tachometer signal and the 1771-ES expander. Measure the resistance the potentiometer was providing. Replace the potentiometer with a fixed resistor of equivalent value to limit the tachometer signal into the 1771-ES expander to 50V.

7. Restore power and jog the axis in either direction. While the axis is moving, measure the voltages at the white and yellow test points on the 1771-ES expander (DAC and tachometer voltages, respectively) with respect to terminal 4 (analog return) on the right wiring arm of the 1771-ES expander (Figure 9.4). These voltages should have the same polarity (+ when jogging in the positive direction, -while jogging in the negative direction).

Figure 9.4
Tachometer Calibration Procedure Voltage Reading



8. While jogging the axis, adjust the tach course potentiometer counterclockwise until the tach cal indicator just changes state. Then jog the axis in the other direction and turn the tach course potentiometer clockwise until the tach cal indicator begins to flicker.

9. Turn off the tachometer calibrate bit in the command block.

⚠ **WARNING:** To guard against possible injury or damage to equipment, before proceding with step 10, keep all personnel clear of the axis. In addition, have a competent person standing by to press an emergency stop switch if necessary.

10. Jog the axis or execute a programmed moveset at low axis speed. While the axis is moving, disconnect one of the encoder leads (terminals 1 thru 7 on the left 1771-ES expander wiring arm). Axis motion should stop. The status block should indicate both immediate stop and loss-of-feedback.

For single-ended encoders, loss-of-feedback is not detected if the channel A, channel B, or MARKER return connection is broken, since these 3 connections are common. (That is, all three leads must be disconnected for loss-of-feedback to be detected.) For these encoders, loss-of-feedback is detected only when the channel A, channel B or marker signal is disconnected.

11. Reconnect the encoder leads, and reset the system.

Summary

After you have performed all integration procedures for all axes, test execution of your intended move profile. For these tests, do not install tooling or workpieces. Use feedrate override at a low value so that you can more easily stop axis motion if necessary. When profile execution is verified, gradually increase axis speed over successive runs until you are satisfied with profile execution at full speed.

If you have problems with the servo positioning assembly during or after axis integration, refer to chapter 10 for troubleshooting information.

Chapter 10

Troubleshooting

Chapter Objectives

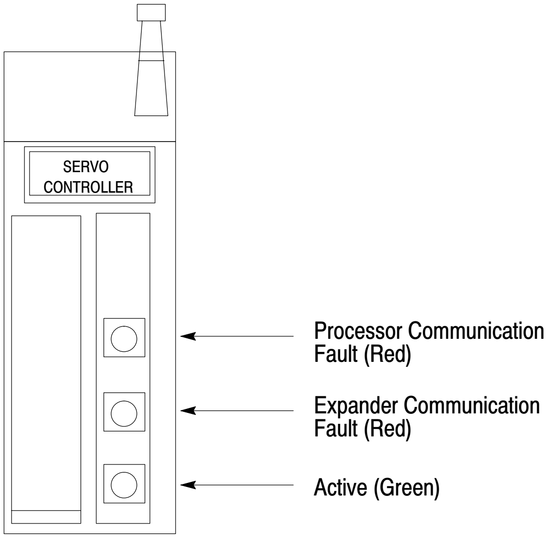
This chapter describes the LED indicators on the 1771-M3 controller and 1771-ES expander modules. It also presents a troubleshooting flowchart. The flowchart provides a logical sequence for evaluating servo positioning assembly condition. Used in conjunction with the indicators and the status block, it can help you detect problems in servo positioning assembly operation.

The status block provides continually updated information about axis and servo positioning assembly conditions. Refer to chapter 7, section titled "Status Block" (p. 7-4) for a detailed description of the status block.

Monitoring 1771-M3 Controller Indicators

The servo controller module has three indicators (Figure 10.1):

Figure 10.1
1771-M3 Controller Indicators


SERVO CONTROLLER
Processor Communication Fault (Red)
Expander Communication Fault (Red)
Active (Green)

- **Processor communication fault (PROC COMM FAULT)** - This red indicator is normally off. It turns on to indicate a communication fault between the PC processor and the 1771-M3 controller. Such a font could be caused by hardware, or by a data block overlapping a processor work area or starting less than 64 words before the user program area.

If a hardware fault is detected at power up, both the processor communication fault and expander communication fault indicators turn on.

- **Expander communication fault (EXPANDER COMM FAULT) -** This red indicator is normally off. It turns on to indicates a communication fault controller between the 1771-ES and the 1771-ES expander.

If a hardware fault is detected at power up, both the processor communication fault and expander communication fault indicators turn on.

- **Active** - This green indicator is normally on. It turns off to indicate a hardware fault on a 1771-ES expander. The active indicator blinks to indicate one of the following improper module configurations:
  - The I/O chassis contains no 1771-ES expander.
  - The I/O chassis contains two 1771-ES expanders with the same switch settings for axis identification.
  - The I/O chassis contains a 1771-ES expander with switches set for axis 2, but no 1771-ES expander for axis 1. (If the I/O chassis contains only one 1771-ES expander, it must be set for axis 1.)
  - The I/O chassis contains a 1771-ES expander with switches set for axis 3, but no 1771-ES expander for axis 2.
  - The I/O chassis contains more than one 1771-M3 controller.
  - The I/O chassis contains another master/slave module combination in addition to the servo positioning assembly. Master/slave combinations include the Analog Input Module (cat. no. 1771-IF), the Analog Output Module (cat. no. 1771-OF), the Stepper Controller Module (cat. no. 1771-M1), and the Thermocouple Input Module (cat. no. 1771-IX), and their respective slave modules.

When the system is powered up, the three servo controller indicators all flash on then off. If the processor is in the run mode with no fault, the active indicator comes back on. If the processor is in the program or test mode with no fault, the indicators all turn off and stay off until the processor is taken out of the test or program mode.

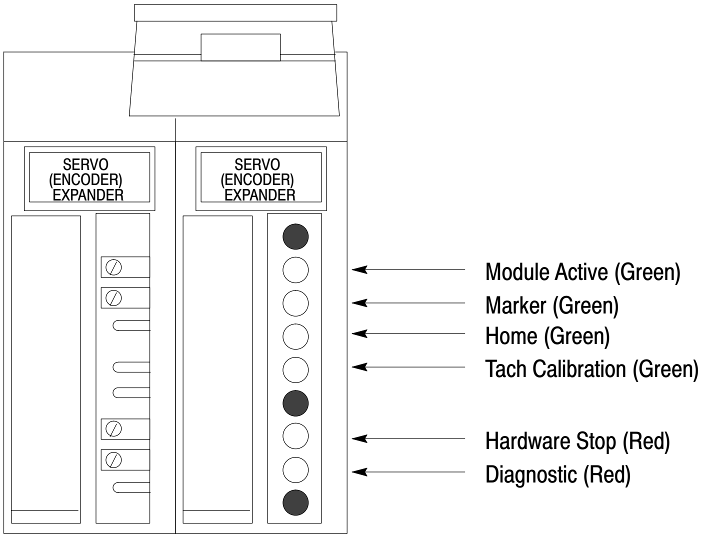
Monitoring 1771-ES Expander Indicators

The 1771-ES expander module has six indicators (Figure 10.2):

- **Module active** - This green indicator is on when the 1771-ES expander is operating normally.
- **Marker** - This green indicator is on when the 1771-ES expander detects the encoder marker signal. Note that this indicator turns on only if the signal from encoder channels A and B are true when the marker signal is true unless you set the marker-logic jumper for ungated.
- **Home** - This green indicator is on when the axis is at the home position.
- **Tachometer calibrate (TACH CAL)** - This green indicator is used in setting up the loss-of-feedback detection feature (see section titled "Tachometer Calibration," chapter 9).
- Hardware stop (HDW STOP) - This red indicator goes on when the hardware stop input is on. It stays on until the 1771-ES expander is reset.
- Diagnostic (DIAG) - This red indicator goes on when the 1771-ES expander detects a module fault.

At power up, the module active, marker, home, hardware stop, and diagnostic indicators all flash on then off. They stay off while the module performs power-up diagnostics. If diagnostics detect no problem, the active indicator turns on.

Figure 10.2
1771-ES Expander Indicators



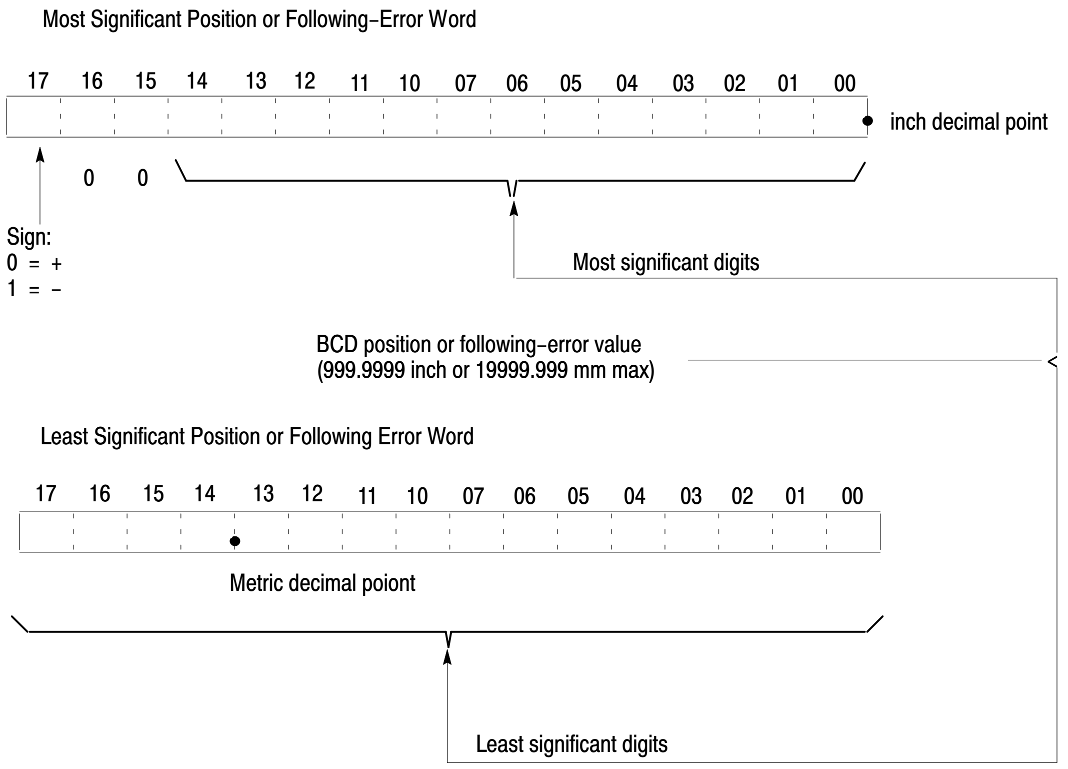
Monitoring the Status Block

The 3rd and 4th status words for an axis provide either current axis position, following error, or diagnostic information. You can select which status to display by controlling the state of bits 11 and 15 of the axis control word 2 of the command block.

Turn off bits 11 and 15 to display the current axis position as shown in Figure 10.3. The maximum value is 999.9999 inch or 19999.99 mm. If the axis exceeds the maximum, it displays the maximum, and the position-valid bit goes off.

Turn off bit 11 and turn on bit 15 to display the following error as shown in Figure 10.3. The maximum value is 999.9999 inch or 19999.99 mm. If the axis exceeds the maximum, it displays the maximum.

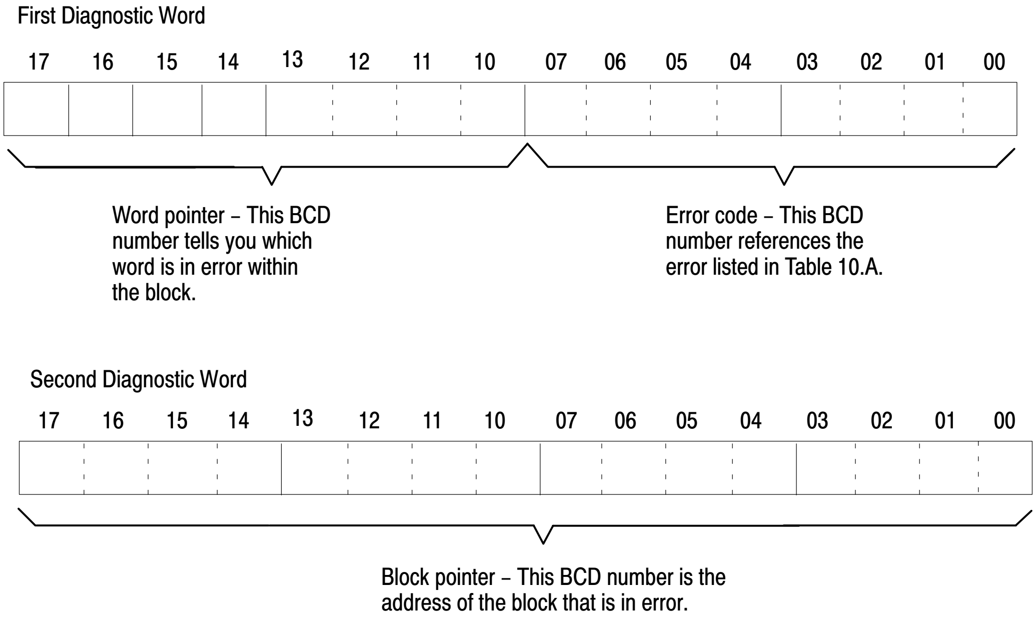
Figure 10.3
Position/Following-Error/Diagnostic Words - with Position or Following-error Selected



Turn on bit 11 to display the diagnostic status as shown in Figure 10.4.

Also, this diagnostic status displays automatically when the 1771-M3 controller detects an error in the parameter block immediately after power-up or an invalid ID in a command block. The diagnostic status displays automatically in that case because the error prevents you from selecting it thru the command block.

Figure 10.4
Position/Following-Error/Diagnostic Words - with Diagnostic Selected



Word pointer – This BCD number tells you which word is in error within the block.

Error code – This BCD number references the error listed in Table 10.A.

Block pointer – This BCD number is the address of the block that is in error.

The second diagnostic word is the block pointer. The block pointer is a BCD number that indicates the starting address of the block in error. The 1771-M3 controller gets these block pointers from the block pointers you enter into the parameter block or the moveset block.

The high byte (bits 10 thru 17) of the first diagnostic word is the word pointer. The word pointer is a BCD number (1 thru 64) that indicates which word is in error within the block.

The low byte (bits 00 thru 7) or the first diagnostic word is the error code. The error code is a BCD number that references the errors listed in Table 10.A.

Use the block pointer and word pointer to identify the location of the problem. Then use the error code to determine the nature of the problem.

Table 10.A
Diagnostic Code Definitions

Code	Definition
01	Invalid block identifier.
02	Non-BCD number entered.
03	Invalid bit setting unused bits must be zero.
04	MS “metric only” bit set in inch format.
05	Overflow: Converted data is too large for internal registers.
06	Can only change feedback multiplier from a power-up reset.
07	Invalid “axes used” programmed.
08	Invalid write block address pointer.
09	Invalid feedback resolution (<0.00001 in. or 0.0001 mm).
10	Invalid feedback multiplier bit setting.
11	(Counts per rev) x (feedback mult) x (encoder lines mult) >32767 decimal.
12	D/A voltage too small for selected rapid rate.
13	Initial gain too small for selected rapid rate.
14	Rapid rate entered exceeds 250 kHz maximum input frequency.
15	Rapid rate entered exceeds 1/2 revolution of encoder/2.4ms.
16	Programmed velocity $\geq$ rapid rate.
17	Invalid velocity exponent programmed.
18	Entered speed is too small for selected feedback resolution.
19	Accel velocity or decel value is too small for selected feedback resolution.
20	Not as many valid SMCWs as there were moves declared in the MCW.
21	Local parameters or run at velocity not allowed for a preset or dwell.
22	Invalid preset position (must be an absolute position).
23	Invalid dwell time (must be >20ms).
24	Escape move block can only have 1 move declared.
25	Invalid escape move block; only moveset blocks identified in the parameter block can be escape move blocks.
26	Cannot program a preset or dwell as an escape move.
27	A valid next-moveset pointer could not be found.
Code	**Definition**

28	Command results in overflow of offset accumulator.
29	Attempted context switch with dual meaning bit on.
30	Attempted context switch while axis is commanding motion.
31	Manual mode only bit(s) on while in auto mode.
32	Invalid motion command bit combination or command not allowed.
33	Invalid command (cannot process new parameters, preset, or offset commands while axis is in motion).
34	Attempted switch to auto mode before first marker is found.

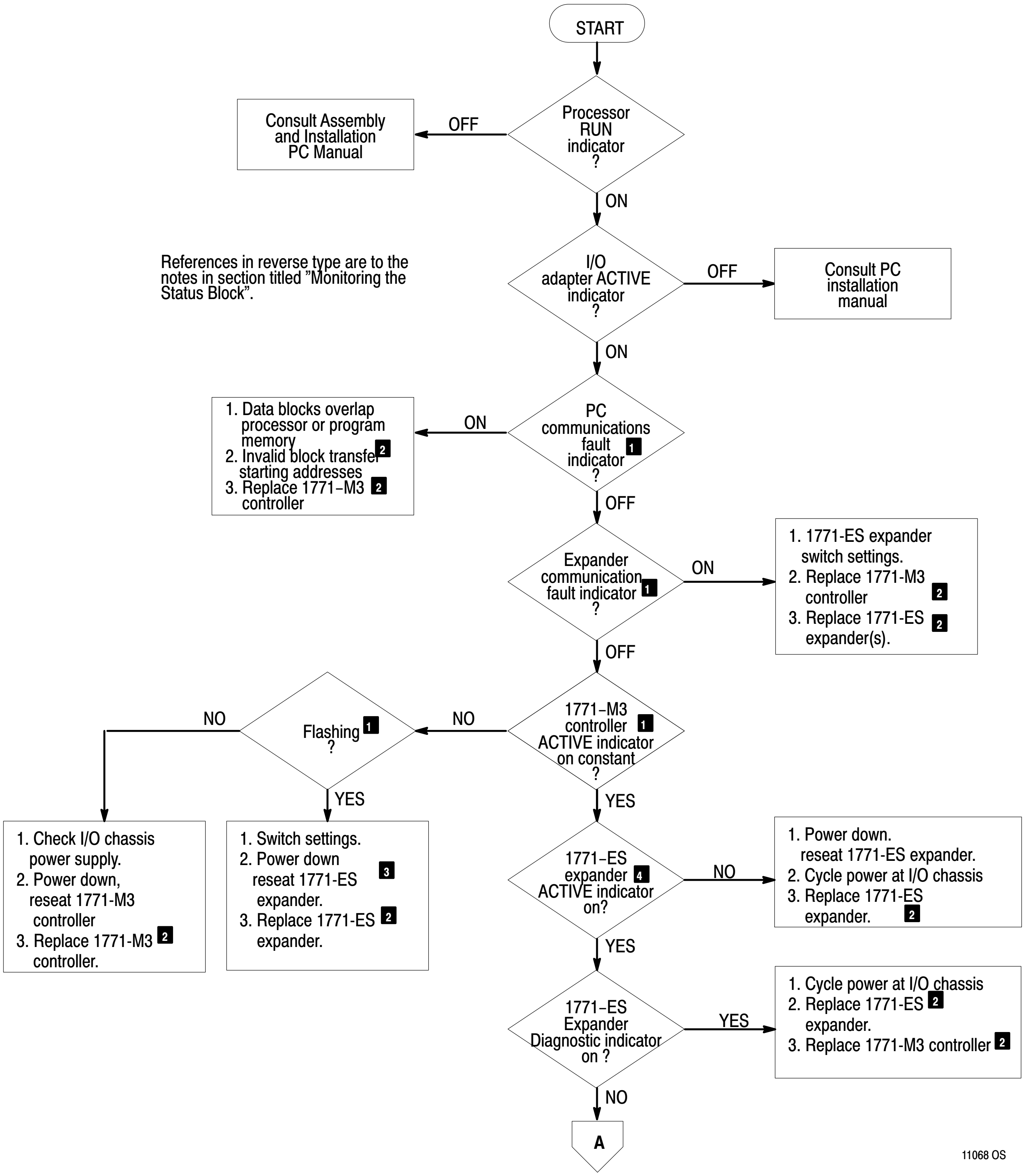
Troubleshooting Flowchart

The flowchart of Figure 10.5 provides a logical procedure to help isolate a problem with servo positioning assembly operation. You can also use it at system start-up to check out the servo positioning assembly for proper operation.

Many of the flowchart boxes that specify corrective actions contain more than one instruction. When using the flowchart, perform instruction 1 first. If this fails to correct the problem, perform instruction 2, and so on.

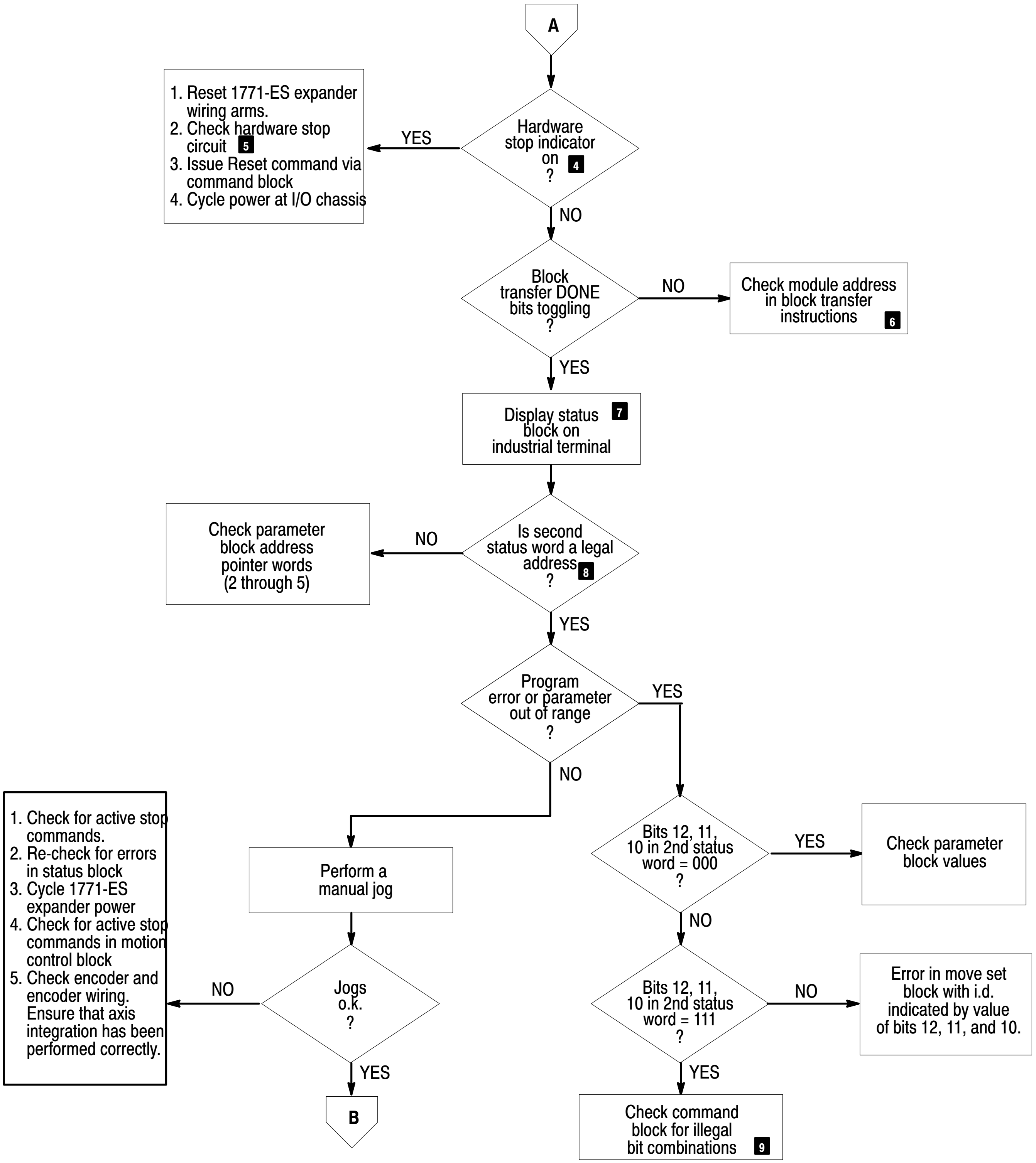
Your ladder diagram program should allow you to display parameter, moveset, command, and status block on an industrial terminal.

Figure 10.5
Troubleshooting Flowchart


START
Processor RUN indicator ?
OFF
Consult Assembly and Installation PC Manual
ON
I/O adapter ACTIVE indicator ?
OFF
Consult PC installation manual
ON
PC communications fault 1 indicator ?
ON
1. Data blocks overlap processor or program memory 2
2. Invalid block transfer starting addresses
3. Replace 1771–M3 2 controller
OFF
Expander communication 1 fault indicator ?
ON
1. 1771-ES expander switch settings.
2. Replace 1771-M3 controller 2
3. Replace 1771-ES 2 expander(s).
OFF
1771–M3 controller 1 ACTIVE indicator on constant ?
NO
Flashing 1 ?
NO
1. Check I/O chassis power supply.
2. Power down, reseat 1771-M3 controller
3. Replace 1771-M3 2 controller.
YES
1. Switch settings.
2. Power down reseat 1771-ES 3 expander.
3. Replace 1771-ES 2 expander.
YES
1771–ES expander 4 ACTIVE indicator on?
NO
1. Power down. reseat 1771-ES expander.
2. Cycle power at I/O chassis
3. Replace 1771-ES expander. 2
YES
1771–ES Expander Diagnostic indicator on ?
YES
1. Cycle power at I/O chassis
2. Replace 1771-ES 2 expander.
3. Replace 1771-M3 controller 2
NO
A
11068 OS

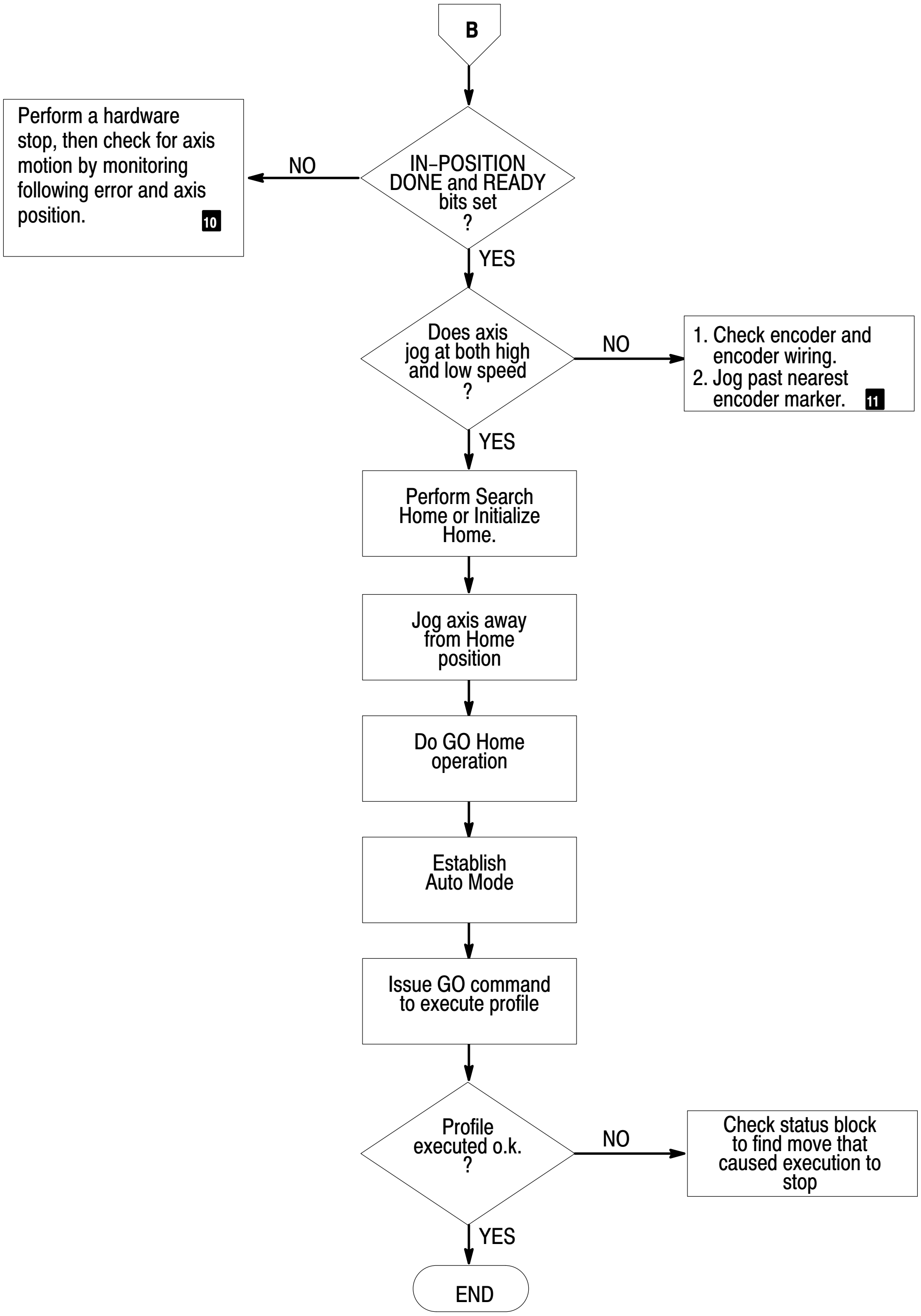
References in reverse type are to the notes in section titled ”Monitoring the Status Block”.

Figure 10.5 Troubleshooting Flowchart (continued)


A
Hardware stop indicator on 4 ?
YES
1. Reset 1771-ES expander wiring arms.
2. Check hardware stop circuit 5
3. Issue Reset command via command block
4. Cycle power at I/O chassis
NO
Block transfer DONE bits toggling ?
NO
Check module address in block transfer instructions 6
YES
Display status block on industrial terminal 7
Is second status word a legal address 8 ?
NO
Check parameter block address pointer words (2 through 5)
YES
Program error or parameter out of range ?
YES
NO
Bits 12, 11, 10 in 2nd status word = 000 ?
YES
Check parameter block values
NO
Bits 12, 11, 10 in 2nd status word = 111 ?
NO
Error in move set block with i.d. indicated by value of bits 12, 11, and 10.
YES
Check command block for illegal bit combinations 9
Perform a manual jog
Jogs o.k. ?
NO
1. Check for active stop commands.
2. Re-check for errors in status block
3. Cycle 1771-ES expander power
4. Check for active stop commands in motion control block
5. Check encoder and encoder wiring. Ensure that axis integration has been performed correctly.
YES
B

11068 OS

Figure 10.5 Troubleshooting Flowchart (continued)


B
IN–POSITION DONE and READY bits set ?
NO
Perform a hardware stop, then check for axis motion by monitoring following error and axis position. 10
YES
Does axis jog at both high and low speed ?
NO
1. Check encoder and encoder wiring.
2. Jog past nearest encoder marker. 11
YES
Perform Search Home or Initialize Home.
Jog axis away from Home position
Do GO Home operation
Establish Auto Mode
Issue GO command to execute profile
Profile executed o.k. ?
NO
Check status block to find move that caused execution to stop
YES
END

11068 OS

Flowchart Notes

The following notes are referenced by numbers of the flowchart of Figure 10.5.

1. Refer to section titled “Monitoring 1771-M3 Controller Indicators,” in this chapter.

2. **CAUTION**: To guard against damage to equipment, power down the system before removing or installing any module.

3. Refer to Figure 6.1

4. Refer to section titled “Monitoring 1771-ES Expander Indicators,” in this chapter.

5. On the left 1771-ES expander wiring arm, measure the voltage at terminal 1 with respect to terminal 12. It should equal the input supply voltage. If it doesn’t, check the power supply. If it does, measure the voltage at terminal 11. This voltage should be zero. If it’s not zero, check for a closed circuit between terminals 11 and 12. (Refer to Figure 5.1 and section titled “Discrete Inputs”).

6. The module address programmed in the block transfer instruction must be the address of the 1771-M3 controller. The rack number in the module address must match the setting of the I/O chassis switches.

7. Refer to the sample program of section titled “Programming Example” in chapter 8 for examples of rungs that perform this function.

8. The second word of the status block specifies the starting address of the block to transfer to the 1771-M3 controller. Be sure that this address specifies the actual starting address of a parameter, moveset, or command block. If it does not, check the corresponding address pointer in the parameter block and make necessary corrections. Also check that the parameter block I.D. bits are correctly programmed (bits 10 thru 17, word 1). If they are not, the 1771-M3 controller does not accept the parameter block.

9. Refer to Table 7.D. Note that in some cases, ladder diagram execution timing can cause illegal bit combinations to be on unintentionally. In such cases, monitoring the command block while executing the ladder diagram program can sometimes help isolate the problem.

10. With the axis in hardware stop, there should be no axis motion and zero following error. If there is axis motion or following error, assure that someone has correctly integrated the axis. (Refer to chapter 9.)

11. After system power-up, the axis jogs only at low speed until the 1771-ES expander receives an encoder marker signal. If high jog speed is initially selected thru the command block, the axis moves at low speed until a marker is detected, then accelerates to high jog speed.

Summary

In this chapter, we gave you information that should help you troubleshoot your system. As you gain experience with your system, the process of troubleshooting should become easier for you.

Glossary

Absolute Dimension: A dimension expressed with respect to the initial zero point of a coordinate axis.

Accumulator Register: A register that accumulates the axis feed increments to indicate the current commanded position for the axis to follow.

Adapter Module: A module that provides communication between an I/O chassis and the PC processor. It transmits I/O chassis input information to, and receives output information from, the processor.

Amplifier: A signal gain device whose output is a function of its input.

Analog: An expression of values that can vary continuously between specified limits.

Axis: A principal direction along which a movement of the tool or workpiece occurs.

Backlash: A relative movement between interacting mechanical parts, resulting from looseness.

Binary: A base 2 numbering system.

Binary Coded Decimal (BCD): A numbering system used to express individual decimal digits (0 thru 9) in four-bit binary notation.

Bit: Binary digit. The smallest unit of information. Represented by the digits 0 and 1. The smallest division of a PC word.

Block: A set of words handled as a unit.

Clear: To erase the contents of a storage device by replacing the contents with zeros.

Closed-Loop: A signal path in which results are fed back for comparison with desired values to regulate system behavior.

Current Sink: A signal sending device that shunts current to ground.

Current Source: A signal sending device that generates positive current.

Data Table: The part of processor memory that contains I/O values and files, where data is monitored, manipulated, and changed for control purposes.

Digital: Representation of data in discrete numerical form.

Digital-to-Analog (D/A) Conversion: Production of an analog signal, whose instantaneous magnitude is proportional to the value of the digital input.

Encoder (Incremental): A rotary device that transmits a fixed number of pulses per revolution.

Feedback: The signal or data transmitted to the PC from a controlled machine to denote its response to the command signal.

Feedback Device: An element of a control system that converts linear or rotary motion to an electrical signal for comparison to the command signal, e. g., encoder.

Feedback Loop: A closed signal path, in which feedback is compared with the commanded value to obtain a corrective error signal.

Feedback Resolution: The smallest increment of dimension that the feedback device can distinguish and reproduce as an electrical output.

Feedback Signal: The measurement signal indicating the value of a directly controlled variable, which is compared to the commanded value to obtain the corrective error signal.

Feedforward Control: Action in which information concerning upstream conditions is converted into corrective commands to minimize the effect of the disturbances.

Gain: The ratio of the magnitude of the output of a system with respect to that of the input.

Gate: A device that blocks or passes a signal, depending on the presence or absence of specified input signals.

Incremental Dimension: A dimension expressed with respect to the previous position of the coordinate axis.

Initialize: To cause a program or hardware circuit to return to an original state.

Instability: The state or property of a system where there is an output for which there is no input.

Instruction: A statement that specifies an operation and the values or locations of its operands.

Integrator: A device that integrates an input signal, usually with respect to time.

Jog: A control function that provides for the momentary operation of a servo drive for manual control of axis motion.

Manual Feedrate Override: The ability of the operator to manually change the feedrate.

Millisecond (ms): One thousandth of a second.

Noise: An extraneous signal in an electrical circuit capable of interfering with the desired signal.

Open Loop: A signal path with feedback.

Overshoot: The amount that a controlled variable exceeds the desired value after a change of input.

Point-to-Point Control System: A system that controls motion only to reach a given end point, but exercises no path control during the transition from one end point to the next.

Position Readout: A display of absolute slide position as derived from a position feedback device that is normally attached to the leadscrew of the machine.

Programmed Dwell: The capability of commanding delays in program execution for a programmable length of time.

Read: 1) To acquire data from a source. 2) Block Transfer; a transfer of data from an intelligent I/O module to the processor data table.

Register: A memory word or area used for temporary storage of data used within mathematical, logical, or transferral functions.

Shield: A conductive barrier that reduces the effect of electric and or magnetic fields.

Sign: The symbol or bit that distinguishes positive from negative numbers.

Signal: The event or electrical quantity that conveys information from one point to another.

Significant Digit: A digit that contributes to the precision of a value. The number of significant digits is counted beginning with the digit contributing the most value, called the most significant digit, and ending with the one contributing the least value, called the least significant digit.

Step Response: The time response of an instrument subjected to an instantaneous change in input.

Summing Point: A point at which signals are added algebraically.

Tachometer: A precision linear DC generator used to provide velocity feedback.

True: As related to PC instructions, an enabled logic state.

Write: 1) The process of loading information into memory. 2) Block Transfer; a transfer of data from the processor data table to an intelligent I/O module.

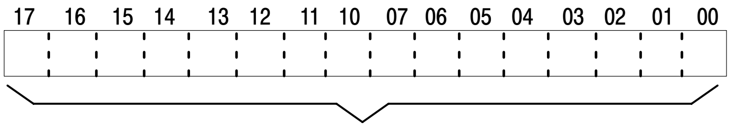
Status Block

Status Block Format

Word	
1	Future Use
2	Address Pointer
3	Status Word 1 (Axis 1)
4	Status Word 2 (Axis 1)
5	(MS) Position/FE/Diagnostic (Axis 1)
6	(LS) Position/FE/Diagnostic (Axis 1)
7	Status Word 1 (Axis 2
8	Status Word 2 (Axis 2)
9	(MS) Position/FE/Diagnostic (Axis 2)
10	(LS) Position/FE/Diagnostic (Axis 2)
11	Status Word 1 (Axis 3)
12	Status Word 2 (Axis 3)
13	(MS) Position/FE/Diagnostic (Axis 3)
14	(LS) Position/FE/Diagnostic (Axis 3)

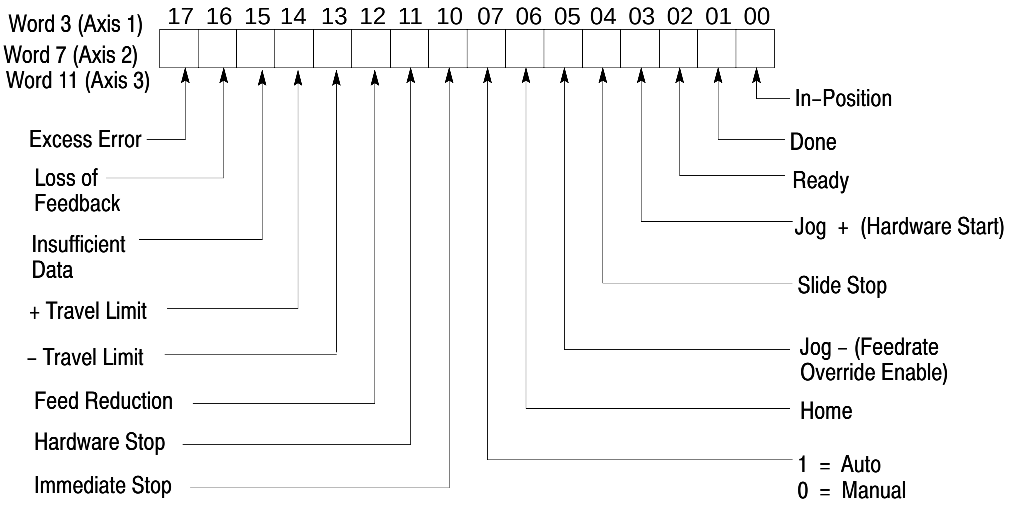
The module sends diagnostic information in this word when you request it thru the command block or when the module detects an error in the parameter block immediately after power-up.

Address Pointer
Word 2

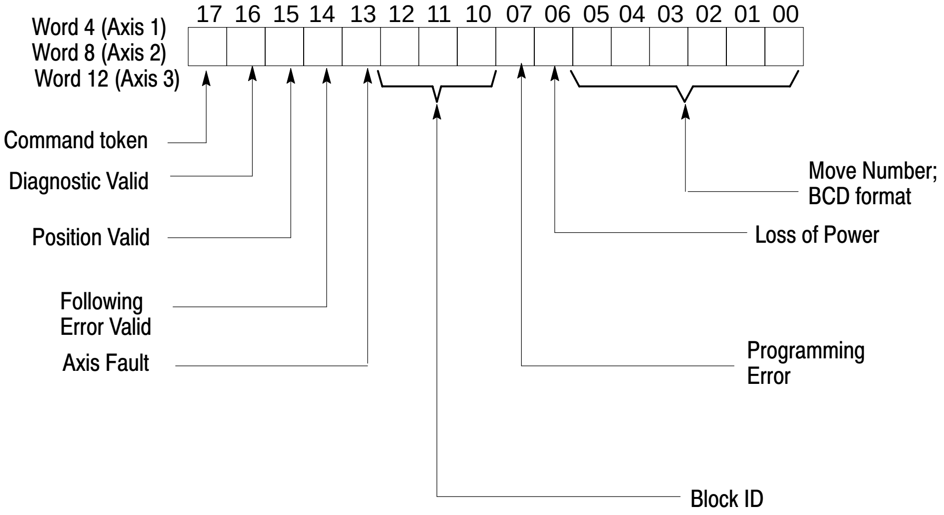


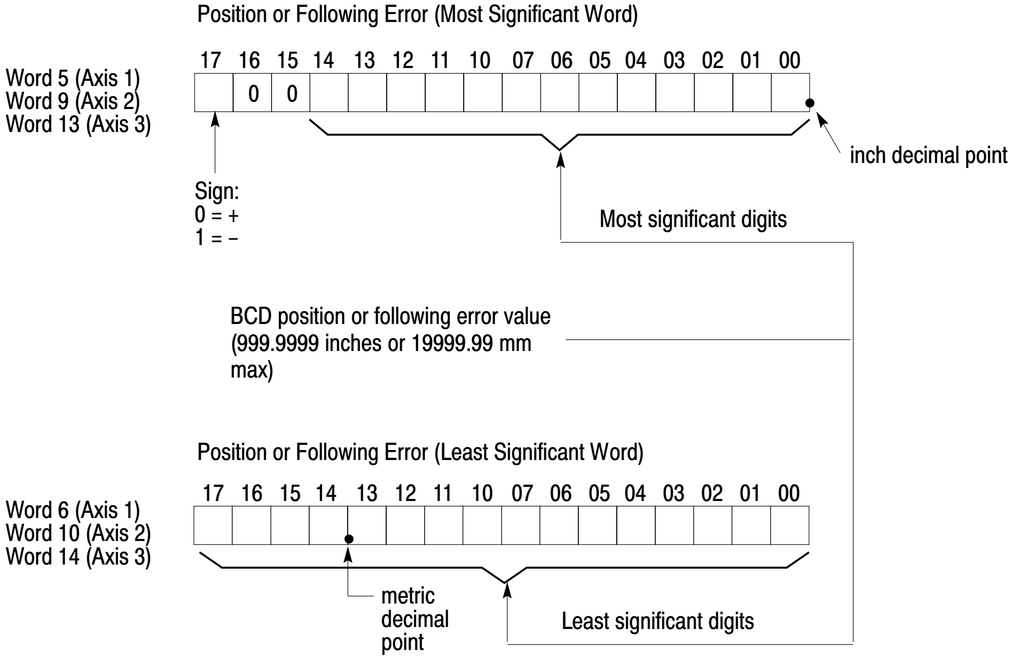
Address of next block to be write transferred to the 1771-M3 controller, BCD format.

First Status Word

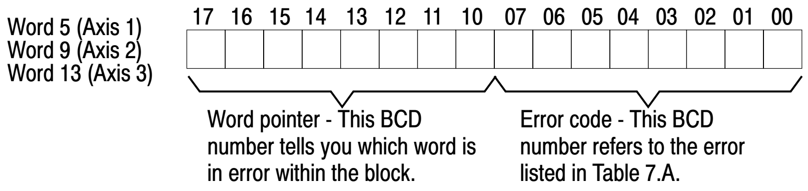


Second Status Word

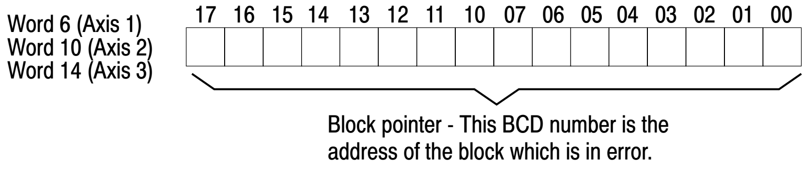




First Diagnostic Word


Word 5 (Axis 1)
Word 9 (Axis 2)
Word 13 (Axis 3)
17 16 15 14 13 12 11 10 07 06 05 04 03 02 01 00
Word pointer - This BCD number tells you which word is in error within the block.
Error code - This BCD number refers to the error listed in Table 7.A.

Second Diagnostic Word



Appendix C

Parameter Block

1	Parameter Block Control Word	Fixed Overhead
2	Parameter Block Pointer	
3	Command Block Pointer	
4	Moveset Block Pointer – Axis 1	
5	Moveset Block Pointer – Axis 2	
6	Moveset Block Pointer – Axis 3	
7	Feedback Resolution	Parameters for Axis 1
8	Encoder Lines	
9	Feedback Mult., Encoder Lines Mult., Initial Gain	
10	Gain Break Speed	
11	In–Position Band/Gain Reduction Factor	
12	Rapid Traverse Rate	
13	High Jog Rate	
14	Low Jog Rate	
15	% Excess Following Error, +D/A Vlotage	
16	% Excess Following Error, –D/A Voltage	
17	Home Position (MS)	
18	Home Position (LS)	
19	Global Accel/Decel Rates	
20	Decel Step Rate	
21	+Software Travel Limit	
22	–Software Travel Limit	
23	Backlash Take–up	
24	Offset	
25	FE Reduction, Tach Conversion Factor	
26 . . . 44	Words 26–44 specify same parameters as words 7–25 but for Axis 2. (Values may be different).	Parameters for Axis 2
45 . . . 63	Words 45–63 specify same parameters as words 7–25, but for Axis 3. (Values may be different).	Parameters for Axis 3

Parameter Block Control word

	17	16	15	14	13	12	11	10	07	06	05	04	03	02	01	00
Word 1	0	1	0	0	0	0	0	0		0	0	0	0			

Identifies this as
a parameter block

0 = Inch
1 = Metric

No. of Axes

0 0 1 = 1
0 1 1 = 2
1 1 1 = 3

Parameter Block Pointer

	17	16	15	14	13	12	11	10	07	06	05	04	03	02	01	00
Word 2																

Data table address of parameter
block, BCD format

Command Block Pointer

	17	16	15	14	13	12	11	10	07	06	05	04	03	02	01	00
Word 3																

Data table address of command
block, BCD format

Axis 1 Moveset Block Pointer

	17	16	15	14	13	12	11	10	07	06	05	04	03	02	01	00
Word 4																

Data table address of first moveset
block to be transferred for axis 1,
BCD format.

Axis 2 Moveset Block Pointer

	17	16	15	14	13	12	11	10	07	06	05	04	03	02	01	00
Word 5																

Data table address of first moveset
block to be transferred for axis 2,
BCD format.

Axis 3 Moveset Block Pointer

	17	16	15	14	13	12	11	10	07	06	05	04	03	02	01	00
Word 6																

Data table address of first moveset
block to be transferred for axis 3,
BCD format.

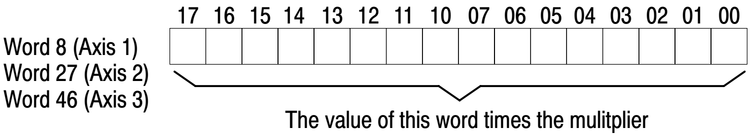
Feedback Resolution

	17	16	15	14	13	12	11	10	07	06	05	04	03	02	01	00
Word 7 (Axis 1) Word 26 (Axis 2) Word 45 (Axis 3)																

Feedback resolution, BCD format
(0010 minimum)

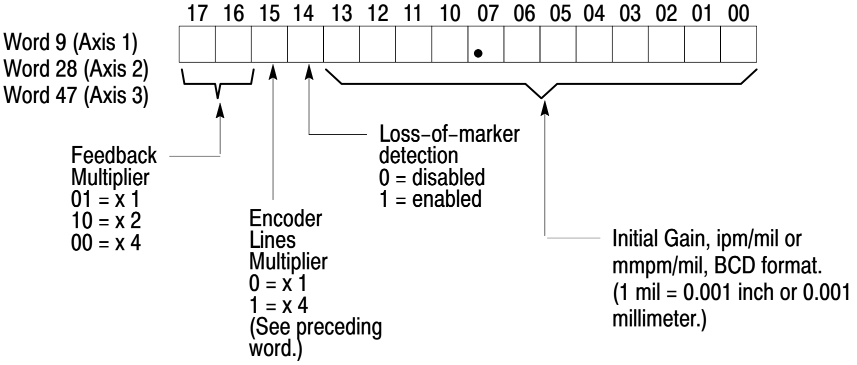
0 = inches x 10^{-6}
1 = millimeters x 10^{5}

Encoder Lines

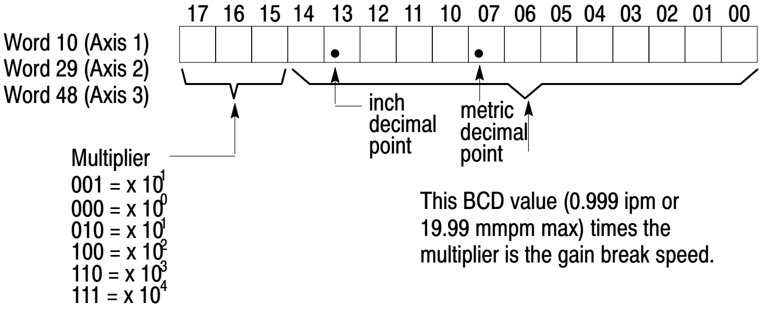


The value of this word times the mulitplier specified by bit 15 of the next word must equal the actual number of encoder lines, BCD format. For 10,000, program 0000.

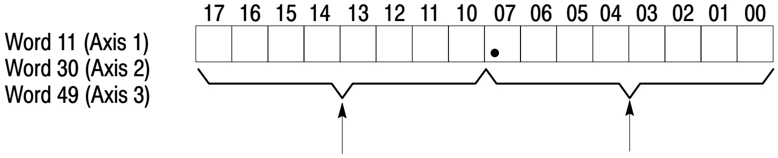
Feedback Multiplier, Encoder Lines Multiplier, Loss–of–marker, Initial Gain



Gain Break Speed



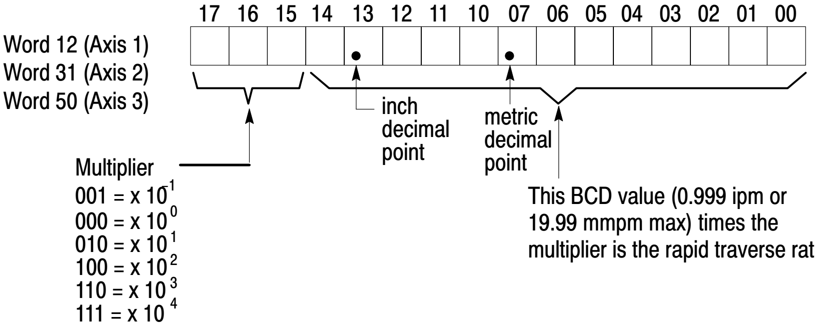
This BCD value (0.999 ipm or 19.99 mmpm max) times the multiplier is the gain break speed.



This BCD value (99 max) times 2 is the in-position band in increments of feedback resolution.

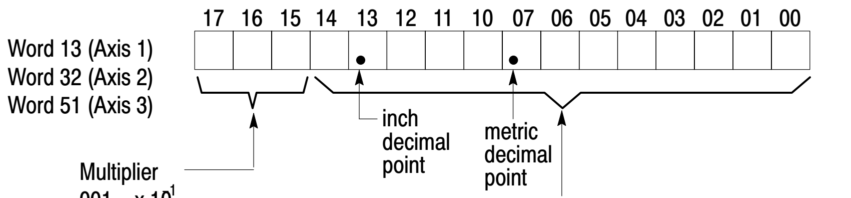
Gain reduction factor

Rapid Traverse Rate



This BCD value (0.999 ipm or 19.99 mmpm max) times the multiplier is the rapid traverse rate.

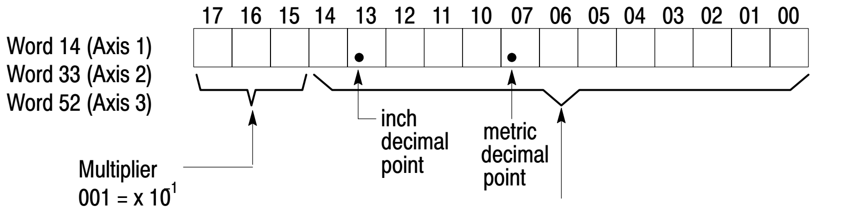
High Jog Rate



Multiplier
001 = x 10^1
000 = x 10^0
010 = x 10^1
100 = x 10^2
110 = x 10^3
111 = x 10^4

This BCD value (0.999 ipm or 19.99 mmpm max) times the multiplier is the high jog rate. It must not be higher than the rapid traverse rate.

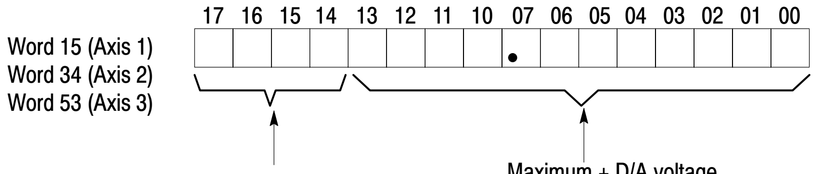
Low Jog Rate



Multiplier
001 = x 10^1
000 = x 10^0
010 = x 10^1
100 = x 10^2
110 = x 10^3
111 = x 10^4

This BCD value (0.999 ipm or 19.99 mmpm max) times the multiplier is the low jog rate. It must be lower than the high jog rate.

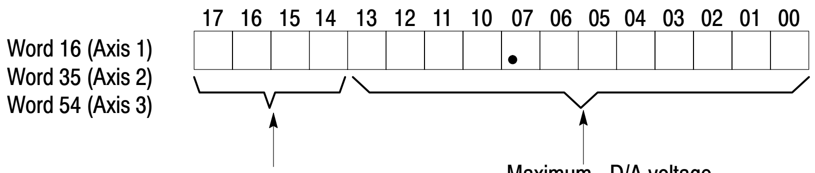
% Excess Following Error (MSD), +D/A Voltage



Most significant digit of excess following error percentage, BCD format.

Maximum + D/A voltage (analog output voltage), BCD format. For +10.0V, program 000.

% Excess Following Error (LSD), –D/A Voltage



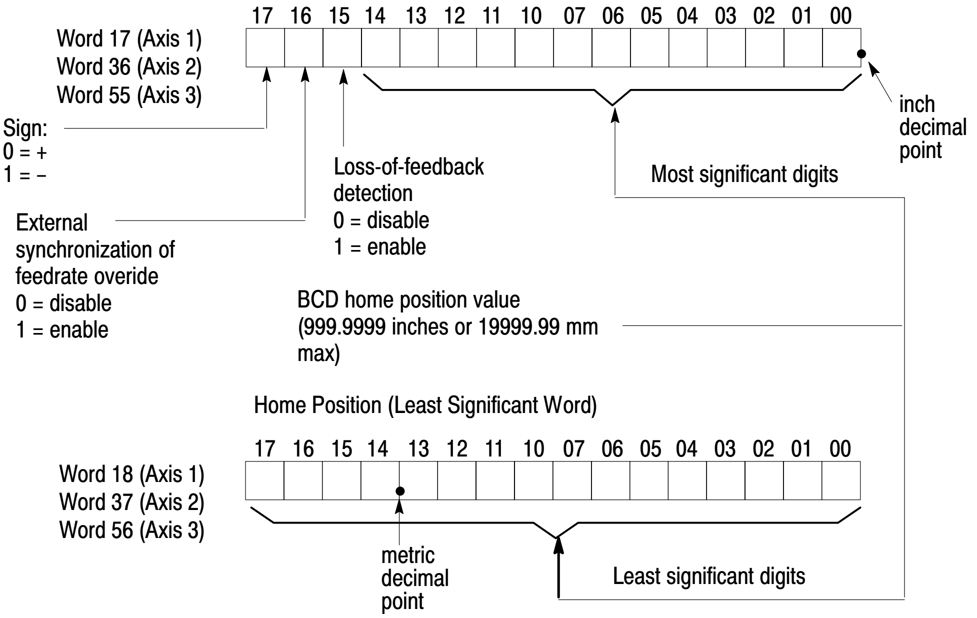
Least significant digit of excess following error percentage, BCD format.

Maximum - D/A voltage (analog output voltage), BCD format. For -10.0V, program 000.

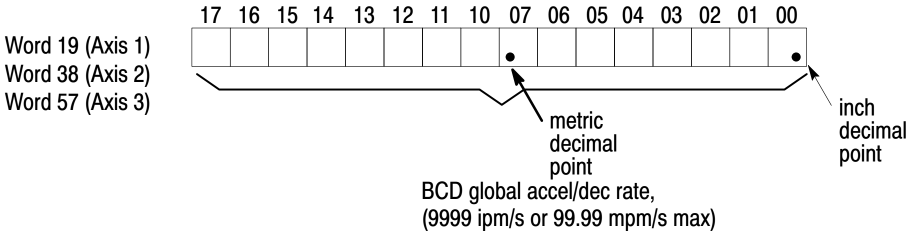
Excess following error percent should be greater than or equal to 6%. The value entered here is the percent above rapid traverse following error at which Emergency Stop is to occur.

Most Significant Home Position

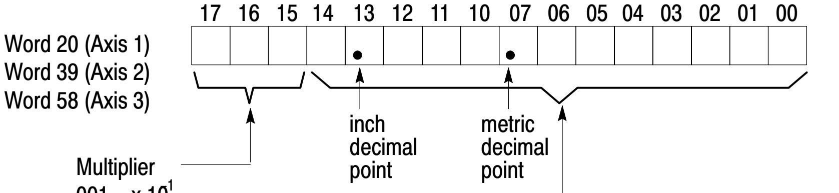
Home Position (Least Significant Word)



Global Accel/Decel Rate



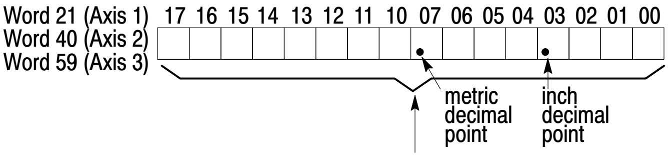
Decel Step Rate



Multiplier
001 = x 10^{1}
000 = x 10^{0}
010 = x 10^{1}
100 = x 10^{2}
110 = x 10^{3}
111 = x 10^{4}

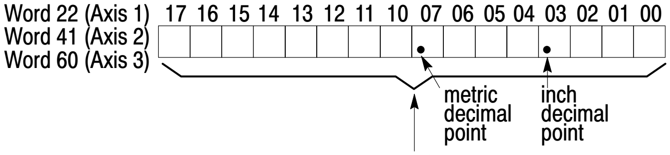
This BCD value (0.999 ipm or 19.99 mmpm max) times the multiplier is the decel step rate. During deceleration, the axis feed rate steps directly to zero once the rate drops to this level. This only applies to jog and search home.

+ Software Travel Limit



Positive software travel limit. An axis position value in inches or meters, BCD format.

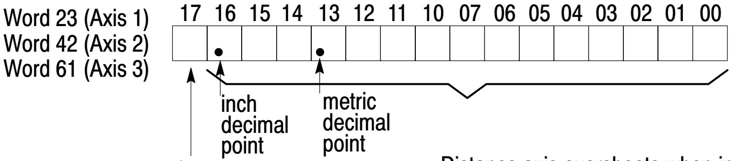
–Software Travel Limit



Negative software travel limit. An axis position value in inches or meters, BCD format.

CAUTION: If programmed values are zero, there are no software travel limits. To guard against damage to equipment, exercise caution when operating an axis without software travel limits.

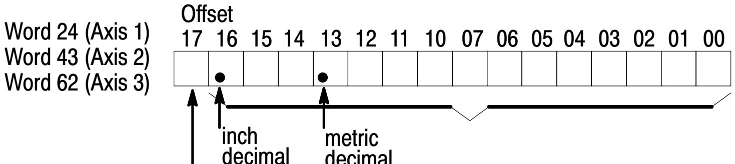
Backlash Takeup Distance



Axis approaches all endpoints moving in the direction specified by the sign (bit 17).

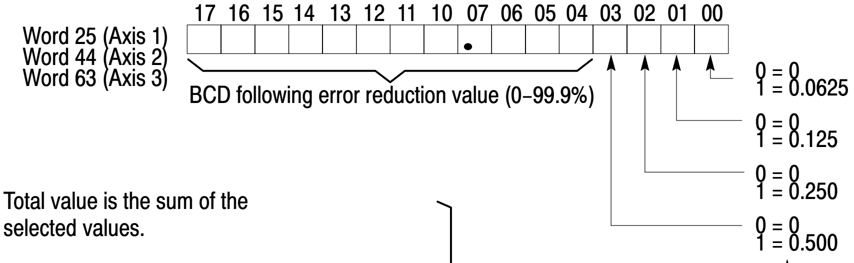
Distance axis overshoots when initial approach to endpoint is from direction opposite that specified in bit 17.

Offset



Offset value, inches or millimeters, BCD format

FE Reduction, Tach Conversion Factor



BCD following error reduction value (0–99.9%)

Total value is the sum of the selected values.

Used if full scale analog output voltage is greater than tachometer voltage for a given rpm. Refer to the tachometer calibration procedure in chapter 9.

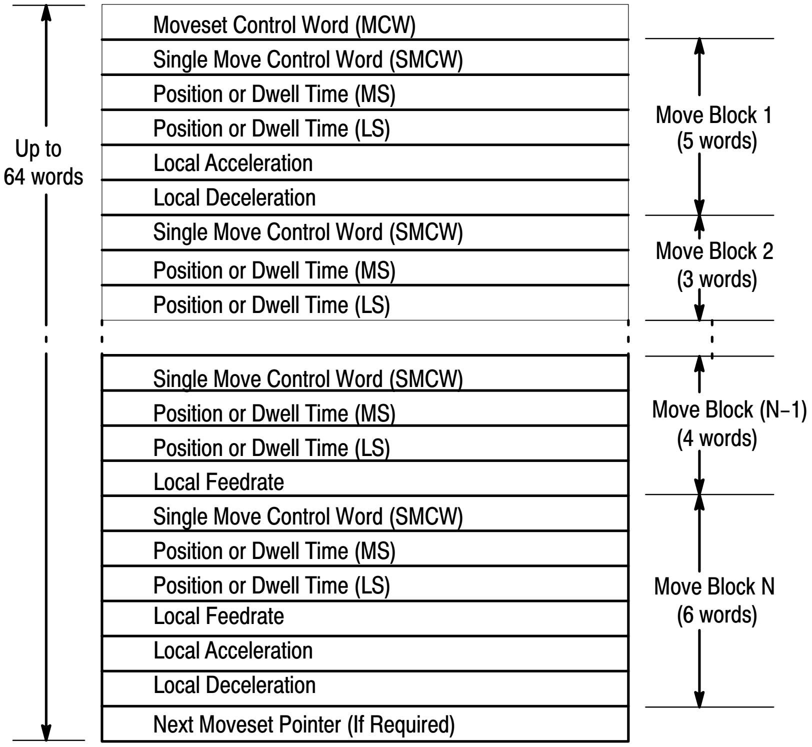
11050

Parameter Block Values

Parameter	Limits
Feedback Resolution	0.00010–0.07999 in. 0.0010–0.7999mm
Encoder Lines	1–32764 lines/revolution
Feedback Multiplier	X1, X2, X4
Initial Gain	0.01–9.99 ipm/mil 0.01–9.99 mmpm/mil
Gain Break Speed	0–9990 ipm 0–199900 mmpm
Gain Reduction Factor	1–100%
In–Position Band	0–198 increments (Even values only.)
Excess Following Error	06.00–99.99%
Rapid Traverse Rate	0.0001–9990 ipm 0.01–199900 mmpm
High Jog Rate	0.0001–9990 ipm 0.01–199900 mmpm
Low Jog Rate	0.0001–9990 ipm 0.01–199900 mmpm
Deceleration Step Rate	0–9990 ipm 0–199900 mmpm
Global Accel/Decel Rate	I–9999 ipm/s 0.01–99.99 mpm/s
Home Position Word	±999.9999 in ±19999.999 mm
Software Travel Limits	0–999.9 in 0 – 99.99 m
Maximum D/A Voltage	0.01–10.00V
Backlash Takeup	–0.7999 – +0.7999 in –7.999–+7.999mm
Offset	–0.7999 – +0.7999 in –7.999 – +7.999mm
Following Error Reduction	0–99.9%
External Synchronization of Feedrate Override	Enable–Disable
Loss–Of–Feedback Detection	Enable–Disable
Tachometer Conversion Factor	0–1111 binary

Appendix D

Moveset Block



Moveset Control Word (MCW)

Single Move Control Word (SMCW)

Position or Dwell Time (MS)

Position or Dwell Time (LS)

Local Acceleration

Local Deceleration

Single Move Control Word (SMCW)

Position or Dwell Time (MS)

Position or Dwell Time (LS)

Single Move Control Word (SMCW)

Position or Dwell Time (MS)

Position or Dwell Time (LS)

Local Feedrate

Single Move Control Word (SMCW)

Position or Dwell Time (MS)

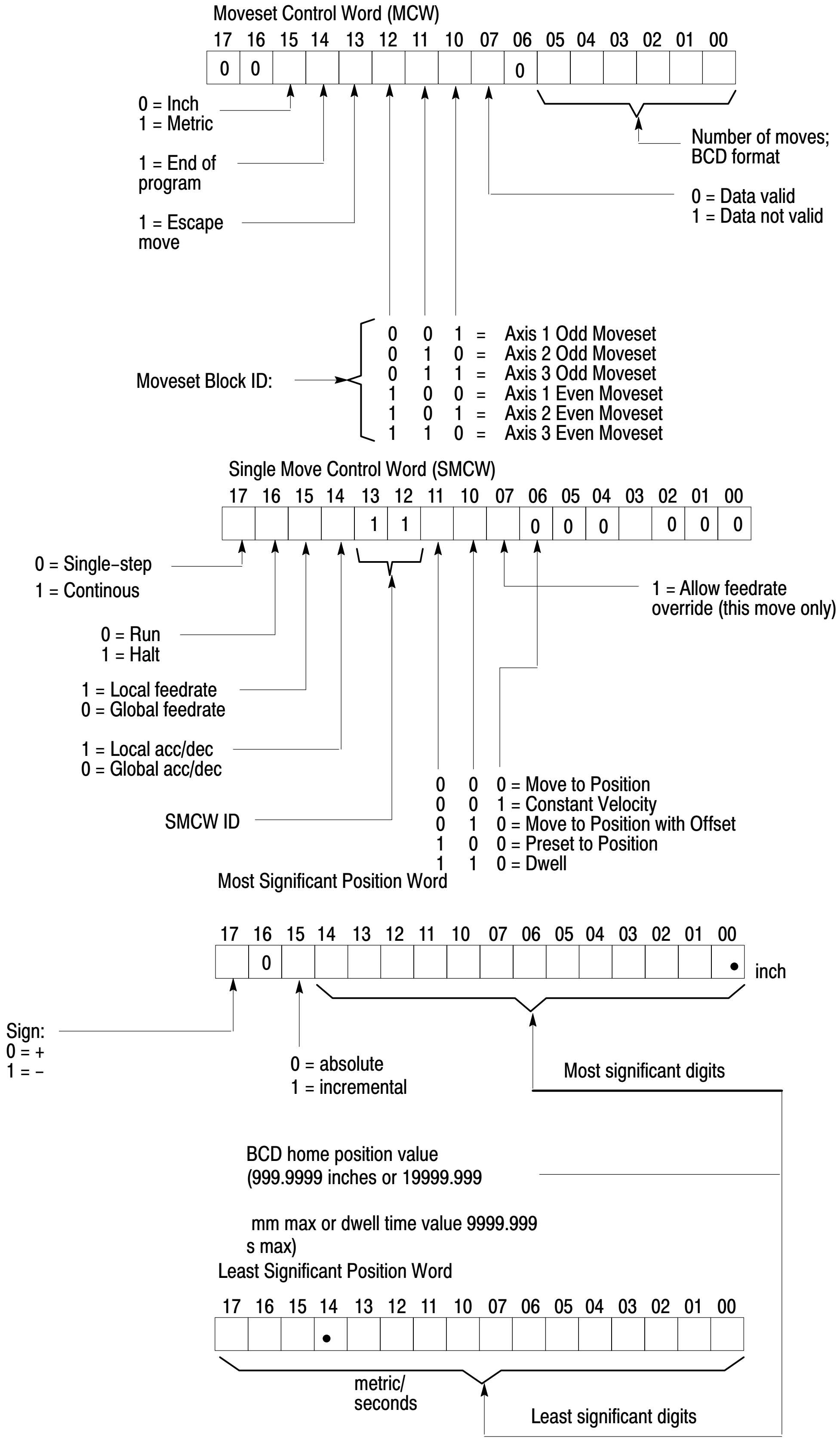
Position or Dwell Time (LS)

Local Feedrate

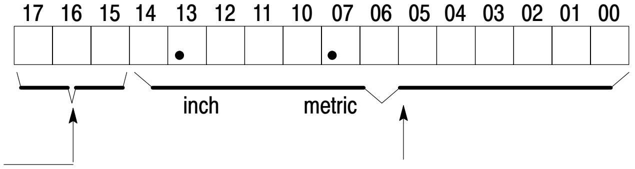
Local Acceleration

Local Deceleration

Next Moveset Pointer (If Required)


Moveset Control Word (MCW)
17 16 15 14 13 12 11 10 07 06 05 04 03 02 01 00
0 0
0
0 = Inch
1 = Metric
1 = End of program
1 = Escape move
Number of moves; BCD format
0 = Data valid
1 = Data not valid
Moveset Block ID:
0 0 1 = Axis 1 Odd Moveset
0 1 0 = Axis 2 Odd Moveset
0 1 1 = Axis 3 Odd Moveset
1 0 0 = Axis 1 Even Moveset
1 0 1 = Axis 2 Even Moveset
1 1 0 = Axis 3 Even Moveset
Single Move Control Word (SMCW)
17 16 15 14 13 12 11 10 07 06 05 04 03 02 01 00
1 1
0 0 0
0 0 0
0 = Single–step
1 = Continous
0 = Run
1 = Halt
1 = Local feedrate
0 = Global feedrate
1 = Local acc/dec
0 = Global acc/dec
SMCW ID
1 = Allow feedrate override (this move only)
0 0 0 = Move to Position
0 0 1 = Constant Velocity
0 1 0 = Move to Position with Offset
1 0 0 = Preset to Position
1 1 0 = Dwell
Most Significant Position Word
17 16 15 14 13 12 11 10 07 06 05 04 03 02 01 00
0
•
inch
Sign:
0 = +
1 = –
0 = absolute
1 = incremental
Most significant digits
BCD home position value
(999.9999 inches or 19999.999
mm max or dwell time value 9999.999
s max)
Least Significant Position Word
17 16 15 14 13 12 11 10 07 06 05 04 03 02 01 00
•
metric/
seconds
Least significant digits

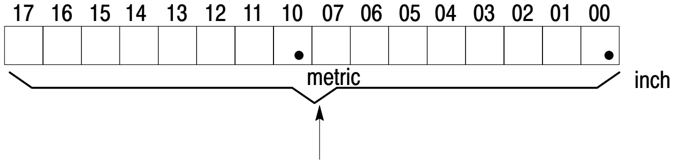
Local Feedrate Word
(Do not include this word if you select global feedrate)



Multiplier
001 = x 10^{-1}
000 = x 10^{0}
010 = x 10^{1}
100 = x 10^{2}
110 = x 10^{3}
111 = x 10^{4}

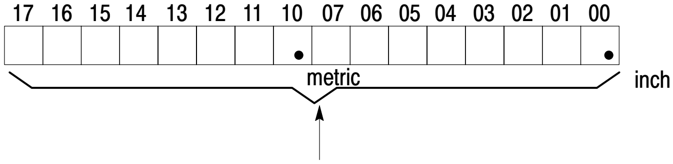
This BCD value (0.999 ipm or 19.99 mmpm max) times the multiplier is the local feedrate.

Local Accel Word
(Do not include this word if you select global feedrate)



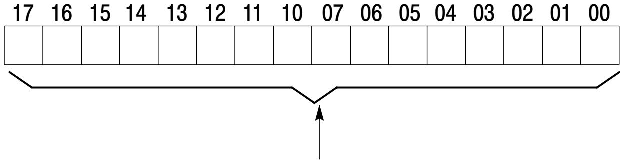
Acceleration rate; ipm/sec or mpm/sec (meters/min./sec/) BCD format

Local Decel Word
(Do not include this word if you select global feedrate)



Deceleration rate; ipm/sec or mpm/sec (meters/min./sec/) BCD format

Next Moveset Block Pointer



Data table address of next moveset block; BCD format

Appendix E

Command Block

a) Single–Axis

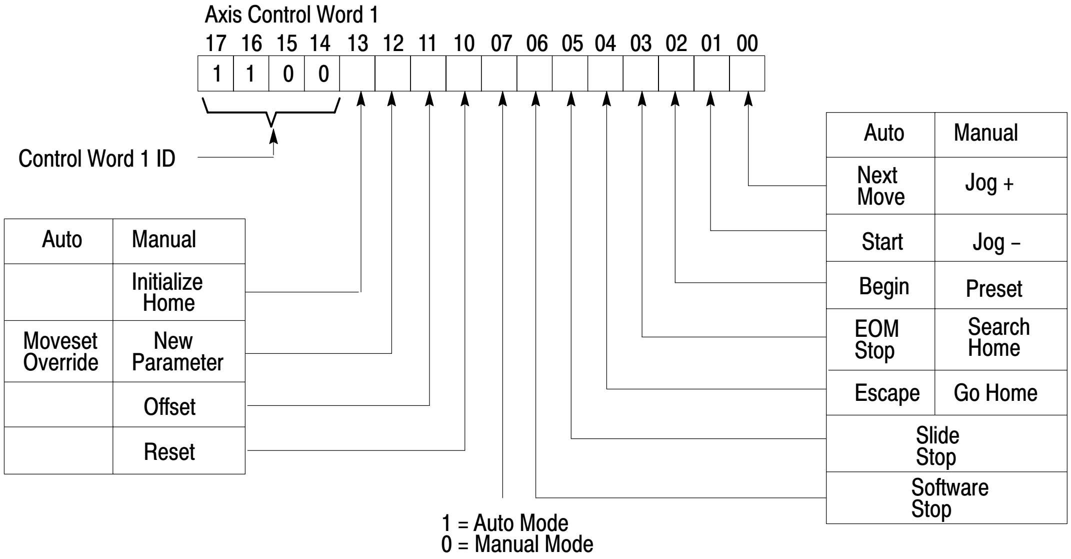
Word	
1	Control Word 1
2	Control Word 2
3	Position Preset (MS) Word
4	Position Preset (LS) Word

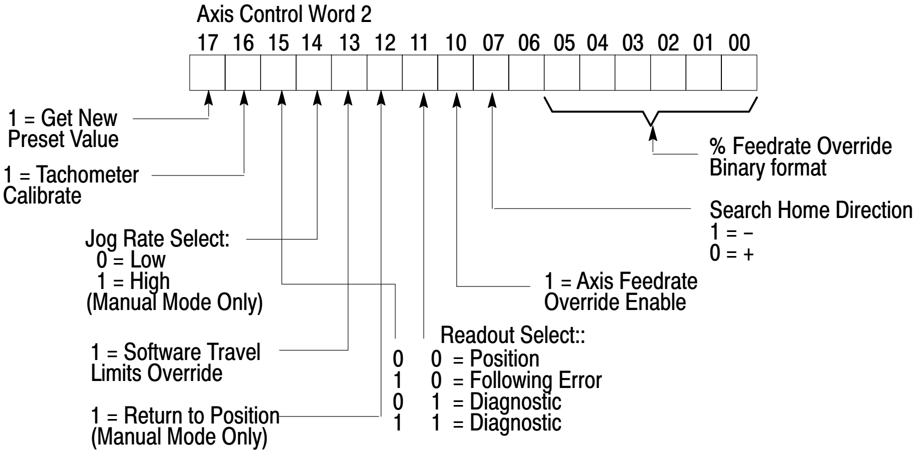
b) Two–Axis

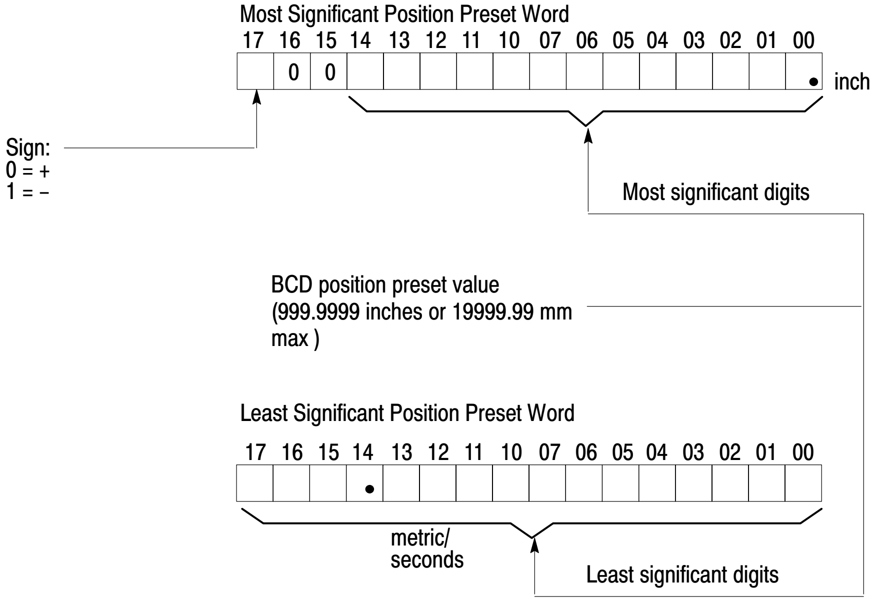
Word	
1	Control Word 1 – Axis 1
2	Control Word 2– Axis 1
3	Control Word 1 – Axis 2
4	Control Word 2 – Axis 2
5	Position Preset (MS) Word – Axis 1
6	Position Preset (LS) Word – Axis 1
7	Position Preset (MS) Word – Axis 2
8	Position Preset (LS) Word – Axis 2

c) Three–Axis

Word	
1	Control Word 1 – Axis 1
2	Control Word 2– Axis 1
3	Control Word 1 – Axis 2
4	Control Word 2 – Axis 2
5	Control Word 1 – Axis 3
6	Control Word 2 – Axis 3
7	Position Preset (MS) Word – Axis 1
8	Position Preset (LS) WOrd – Axis 1
9	Position Preset (MS) Word – Axis 2
10	Position Preset (LS) Word – Axis 2
11	Position Preset (MS) Word – Axis 3
12	Position Preset (LS) Word – Axis 3

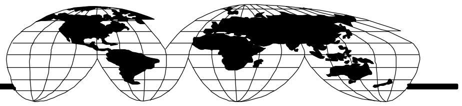

Axis Control Word 1
17 16 15 14 13 12 11 10 07 06 05 04 03 02 01 00
1 1 0 0
Control Word 1 ID
Auto
Manual
Initialize Home
Moveset Override
New Parameter
Offset
Reset
1 = Auto Mode
0 = Manual Mode
Auto
Manual
Next Move
Jog +
Start
Jog –
Begin
Preset
EOM Stop
Search Home
Escape
Go Home
Slide Stop
Software Stop


Axis Control Word 2
17 16 15 14 13 12 11 10 07 06 05 04 03 02 01 00
1 = Get New Preset Value
1 = Tachometer Calibrate
Jog Rate Select:
0 = Low
1 = High
(Manual Mode Only)
1 = Software Travel Limits Override
1 = Return to Position (Manual Mode Only)
Readout Select::
0 0 = Position
1 0 = Following Error
0 1 = Diagnostic
1 1 = Diagnostic
1 = Axis Feedrate Override Enable
% Feedrate Override Binary format
Search Home Direction
1 = –
0 = +


Most Significant Position Preset Word
17 16 15 14 13 12 11 10 07 06 05 04 03 02 01 00
0 0
•
inch
Sign:
0 = +
1 = –
Most significant digits
BCD position preset value
(999.9999 inches or 19999.99 mm max)
Least Significant Position Preset Word
17 16 15 14 13 12 11 10 07 06 05 04 03 02 01 00
•
metric/ seconds
Least significant digits

Rockwell Automation
Allen-Bradley

Allen-Bradley, a Rockwell Automation Business, has been helping its customers improve productivity and quality for more than 90 years. We design, manufacture and support a broad range of automation products worldwide. They include logic processors, power and motion control devices, operator interfaces, sensors and a variety of software. Rockwell is one of the worlds leading technology companies.



Worldwide representation.

Argentina • Australia • Austria • Bahrain • Belgium • Brazil • Bulgaria • Canada • Chile • China, PRC • Colombia • Costa Rica • Croatia • Cyprus • Czech Republic • Denmark • Ecuador • Egypt • El Salvador • Finland • France • Germany • Greece • Guatemala • Honduras • Hong Kong • Hungary • Iceland • India • Indonesia • Ireland • Israel • Italy • Jamaica • Japan • Jordan • Korea • Kuwait • Lebanon • Malaysia • Mexico • Netherlands • New Zealand • Norway • Pakistan • Peru • Philippines • Poland • Portugal • Puerto Rico • Qatar • Romania • Russia–CIS • Saudi Arabia • Singapore • Slovakia • Slovenia • South Africa, Republic • Spain • Sweden • Switzerland • Taiwan • Thailand • Turkey • United Arab Emirates • United Kingdom • United States • Uruguay • Venezuela • Yugoslavia

Allen-Bradley Headquarters, 1201 South Second Street, Milwaukee, WI 53204 USA, Tel: (1) 414 382-2000 Fax: (1) 414 382-4444

Publication 1771-6.5.25 – October, 1985
Supersedes 1771-832 – August, 1994

PN 955098-32

Copyright 1986 Allen-Bradley Company, Inc.   Printed in USA