

CHAPTER SIX



# Following

## IN THIS CHAPTER

This chapter will help you understand Ratio Following:

- Introduction to Ratio Following ..... 166
- Implementing Ratio Following ..... 168
- Master Cycle Concept ..... 183
- Technical Considerations for Following ..... 189
- Troubleshooting for Following ..... 199
- Following Commands (list)..... 200

## Ratio Following – Introduction

---

### Compiled Profiles

You can pre-compile Following profiles (saves processing time). See page 139 for details.

As part of its standard features, the 6K Series Controller family allows you to solve applications requiring *Ratio Following*.

Ratio Following is, essentially, controlled motion based on the measurement of external motion. This includes concepts such as an electronic gearbox, trackball, follower axis feed-to-length, as well as complex changes of ratio as a function of master position. Ratio Following can include continuous, preset, and registration-like moves in which the velocity is replaced with a ratio.

The follower axis can follow in either direction and change ratio while moving, with phase shifts allowed during motion at otherwise constant ratio. Ratio changes or new moves can be dependent on master position or based on receipt of a trigger input. Also, a follower axis can perform Following moves or normal time-based moves in the same application because Following can be enabled and disabled at will. *Product cycles* (operations that repeat with periodic master travel) can be easily specified with the master cycle concept (see page 183).

In Ratio Following, acceleration ramps between ratios will take place over a user-specified master distance. Product cycles can be easily specified with the master cycle concept.

This chapter highlights the capabilities of the 6K Following features and provides application examples. If you need more details on the operation or syntax of a particular command, please see the *6K Series Command Reference*.

Before delving into the specifics of Ratio Following, read on to gain a basic understanding of how the 6K controller *follows*.

## What can be a master?

Any axis on the 6K controller can be made to any of the master input (“master”) sources listed in the table below. All of the 6K controller’s axes can be following at the same time with the same or different master.

- Commanded position
- Incremental encoder. The encoder can be one of the axis-related “ENCODER” ports, or it can be the “MASTER ENCODER” port.
- Analog input (servo axes only). This requires an ANI SIM be installed on an expansion I/O brick (see your product’s *Installation Guide* for instructions). ANI SIMs and expansion I/O bricks are sold separately.
- Internal count source (a “virtual master” option)
- Internal sine wave source (a “virtual master” option)
- Integer variable (VARI)

A servo axis cannot follow its own feedback device input (encoder or analog input). A stepper axis can follow its own encoder input, as long as that axis does not use the Stall Detect feature (ESTALL1 mode).

For instructions on assigning a master for a particular follower axis, see *Define the Master and Follower Axes* (page 168), or see the FOLMAS command description in the *6K Series Command Reference*.

## Following Status (TFSE, TFS & FS Commands)

Many of the Following features described in this document have associated status bits that can be displayed (with the **TFSE** and **TFS** commands) or used in assignment or comparison operations (with the **FS** operator). The portions of this document that describe those features also summarize the related status bits.

---

**FS Bit Function (YES = 1; NO = 0)**

- 1..... Follower in Ratio Move..... A Following move is in progress.
- 2..... Ratio is Negative ..... The current ratio is negative (i.e., follower counts counting in the opposite direction from master).
- 3..... Follower Ratio Changing.. The follower is ramping from one ratio to another (including a ramp to or from zero ratio).
- 4..... Follower At Ratio ..... The follower is at constant non-zero ratio.

Bits 1-4 indicate the status of the follower axis in Following motion.

- \* 5..... FOLMAS Active..... A master is specified with the **FOLMAS** command.
- \* 6..... FOLEN Active..... Following has been enabled with the **FOLEN** command.
- \* 7..... Master is Moving ..... The specified master is currently in motion.
- 8..... Master Dir Neg ..... The current master direction is negative. (Bit must be cleared to allow Following move in preset mode—**MC0**).

Bits 5-8 indicate the status required for Following motion (i.e., a master must be assigned, Following must be enabled, the master must be moving, and for many features, the master direction must be positive). Unless the master is a commanded position of another axis, minor vibration of the master will likely cause bits 7-8 to toggle on and off, even if the master is nominally "at rest". These bits are meant primarily as a quick diagnosis for the absence of master motion, or master motion in the wrong direction. Many features require positive master counting to work properly.

- 9..... OK to Shift ..... Conditions are valid to issue shift commands (**FSHFD** or **FSHFC**).
- 10..... Shifting now ..... A shift move is in progress.
- 11..... Shift is Continuous..... An **FSHFC**-based shift move is in progress.
- 12..... Shift Dir is Neg..... The direction of the shift move in progress is negative.

Bits 9-12 indicate the shift status of the follower. Shifting is super-imposed motion, but if viewed alone, can have its own status. In other words, bits 10-12 describe only the shifting portion of motion.

- 13..... Master Cyc Trig Pend..... A master cycle restart is pending the occurrence of the specified trigger.
- 14..... Mas Cyc Len Given ..... A non-zero master cycle length has been specified with the **FMCLN** command.
- 15..... Master Cyc Pos Neg..... The current master cycle position (**PMAS**) is negative. This could be by caused by a negative initial master cycle position (**FMCP**), or if the master is moving in the negative direction.
- 16..... Master Cyc Num > 0..... The master position (**PMAS**) has exceeded the master cycle length (**FMCLN**) at least once, causing the master cycle number (**NMCY**) to increment.

Bits 13-16 indicate the status of master cycle counting. If a Following application is taking advantage of master cycle counting, these bits provide a quick summary of some important master cycle information.

- 17..... Mas Pos Prediction On..... Master position prediction has been enabled (**FPPEN**).
- 18..... Mas Filtering On ..... A non-zero value for master position filtering (**FFILT**) is in effect.
- 19..... <RESERVED>
- 20..... <RESERVED>

Bit 17 and 18 indicate the status of master position measurement features.

- 21..... <RESERVED>
- 22..... <RESERVED>
- 23..... OK to do **FGADV** move..... OK to do Geared Advance move (master assigned with **FOLMAS**, Following enabled with **FOLEN**, and follower axis is either not moving, or moving at constant ratio in continuous mode).
- 24..... **FGADV** move underway .... Geared Advance move profile is in progress.

Bits 23 and 24 indicate the status of a Geared Advance move.

- 25..... <RESERVED>
- 26..... **FMAXA/FMAXV** limited..... The present Following move profile is being limited by **FMAXA** or **FMAXV**.
- 27..... <RESERVED>
- 28..... <RESERVED>

Bits 23 and 24 indicate the status of a Geared Advance move.

---

\* All these conditions must be true before Following motion will occur.

# Implementing Ratio Following

This section covers the basic elements of implementing Ratio Following:

- Applying Following setup parameters
- Move profiles
- Performing phase shifts (FSHFC and FSHFD)
- Geared advanced (FGADV)
- Application scenarios:
  - Electronic gearbox
  - Trackball

## Ratio Following Setup Parameters

Prior to executing a Following move, there are several setup parameters that must be specified. These parameters can be established:

⇒  
*Programming examples — see application examples later in this chapter.*

- Define the *master* and *follower axes* (FOLMAS)
- Define master & follower scaling factors (SCLMAS, SCLA, SCLD, and SCLV) — *if required*
- Define the follower-to-master Following ratio (FOLRN and FOLRD)
- Define the master distance (FOLMD) — *define scaling first*
- Enable the Following Mode (FOLEN)

**Following Status**  
 (see TFSF, TFS and FS commands)

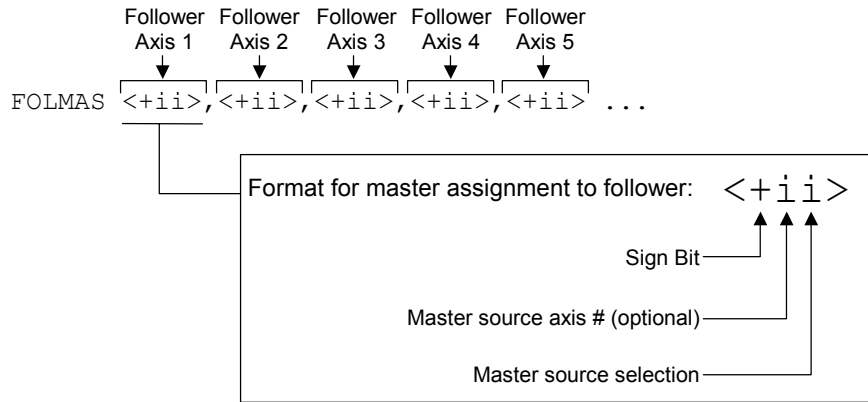
Following Status bits 5-8 (see table below) are meant to indicate the status required for Following motion (i.e., a master must be assigned, Following must be enabled, the master must be moving, and for many features, the master direction must be positive).

Bits 7-8 represent master motion and master direction respectively. Unless the master is a commanded position of another axis, it is likely that minor vibration of the master will cause these bits to toggle on and off, even if the master is nominally “at rest”. These bits are meant primarily as a quick diagnosis for the absence of master motion, or master motion in the wrong direction. Many features require positive master counting to work properly.

Bit No.	Function (YES = 1; NO = 0)
5	FOLMAS Active ..... A master is specified with the FOLMAS command.
6	FOLEN Active ..... Following has been enabled with the FOLEN command.
7	Master is Moving ..... The specified master is currently in motion.
8	Master Dir Neg ..... The current master direction is negative. (bit must be cleared to allow Following move in preset mode—MCØ).

**Define the Master and Follower Axes (FOLMAS)**

The FOLMAS command defines the masters and the followers. The command syntax is:



- **Sign bit** ( $\pm$ ): Specifies the count direction of the master source that will result in positive master travel counts. The sign bit is not meant to be used simply to change the direction of follower motion. That function can be done with the sign of the `D` command. Rather, the sign bit is used to allow forward motion of the physical master (e.g., conveyor belt, rotating wheel, or the continuous feed of material or product) to result in positive counts. Several features described later in this document require increasing master counts for proper operation. These include Following motion in preset positioning mode (`MCØ`), master cycle counting, and executing `GOWHEN` based on master cycle position.
- **Master source axis number** (1<sup>st</sup> `i`): Selects the axis number of the master source. This number is not required when selecting the Master Encoder as the master source. If the master source is “8” (`VARI`), this represents the integer variable (`VARI`) number.
- **Master source selection** (2<sup>nd</sup> `i`): Selects the master source (according to the master source axis number). The options, selected by their respective number, are:
  - 1.... Incremental encoder connected to one of the axis-related “ENCODER” ports, or the “MASTER ENCODER” port. If the master encoder is selected, the master source input number must be omitted from the syntax.
  - 2.... Analog input (servo axes only). This requires an ANI SIM be installed on an expansion I/O brick (see your product’s *Installation Guide* for instructions), and the ANI input must be designated as a master input source with the `ANIMAS` command. ANI SIMs and expansion I/O bricks are sold separately.
  - 4.... Commanded position
  - 5.... Internal count source (see “Following a Virtual Master” below for details)
  - 6.... Internal sine wave source (see “Following a Virtual Master” below for details)
  - 8.... `VARI` variable (see “Following an Integer Variable” below for details)

#### NOTES

- Servo controllers: The follower axis cannot use its own commanded position or its currently selected feedback device (encoder or ANI) as the master input.
- Stepper controllers: The follower axis cannot use its own commanded (motor) position as the master input. Also, a follower axis that is using the Stall Detect mode (`ESTALL1`) cannot use its own encoder as the master input.
- Multiple axes can follow the same count source (e.g., encoder) from the same master. However, multiple axes cannot follow different count sources (e.g., encoder and commanded position) from the same master.
- If scaling is enabled (`SCALE1`), the measurement of the master is scaled by the `SCLMAS` value.

**As an example**, suppose the **ENCODER 3** input is to be the master input for follower axis 2, and forward travel of the physical master (e.g., conveyor belt) results in negative counts on **ENCODER 3**. Given these operating constraints, you would use the `FOLMAS, -31` command.



Each count of the count frequency changes the angle by one tenth (0.1) of a degree. For example, a FVMFRQ value of 3600 would create an angular frequency of 3600 tenths degrees per second, or 1 cycle per second. When used as a source for the sine wave, the maximum value for FVMFRQ is 144000. This results in a maximum of 40 Hz angular frequency. Higher frequencies are not allowed because they can be subject to aliasing.

The associated commands are:

```
SINANGi,i,i,i,i,i,i,i..... Where "i" is the new angle (range is 0.0-360.0 degrees)
SINAMPi,i,i,i,i,i,i,i..... Where "i" is the amplitude (range is 0-9,999,999 —
                             maximum peak-to-peak is 16384))
SINGObbbbbbbb..... Where "b" is the binary start/stop for each master axis:
                             1 = restart from zero degrees
                             0 = stop the sine wave
```

The SINAMP command is included in the specification in order to allow a change in slave amplitude without changing the center of oscillation. Changing the ratio (with FOLRN) will also change the slave amplitude, but unless the command is given at exactly the master zero crossing, it will also change the center of slave oscillation. The center of oscillation can be changed by a controlled amount by using the FSHFD command. The SINAMP command affects the sine wave immediately, without any built in ramp in amplitude. If a gentle change is desired, a user program should be written that repeatedly issues the command with small changes in value, until the desired value is reached.

Using SINGO with a "0" parameter abruptly stops the sine wave, without changing its current magnitude. Using SINGO with a "1" parameter abruptly starts the sine wave, also without changing its current magnitude. To gently pause the slave output, use an FVMFRQ value of zero, with a moderate FVMACC value; to resume, restore the FVMFRQ value.

The SINGO with a "1" parameter always starts at the previous angle, which might not be the desired start of oscillation. The SINANG command will instantly change the angle and corresponding sine of the angle. This represents an abrupt change in master position. If the slave is still following when this occurs, there will be an abrupt change in commanded slave position. To start the slave properly, move the slave to the desired start position first (using MC0, D, GO), then issue SINANG, then MC1, GO1, and finally SINGO. If SINANG is issued without any parameters, the current angle is reported, not the most recent user SINANG command value.

### Sinewave Example

In the example below, an analog input (ANI) is used for feedback on axes 1 and 2, without scaling, so distance is in ANI counts of 205 counts/volt. The ANI voltage range (ANIRNG) is left at its default of -10V to +10V. Axis 1 is supposed to oscillate at 1 Hz, centered around -4096, with a peak-to-peak amplitude of 2048. Axis 2 is supposed to oscillate at 0.1 Hz, centered around 7000, with a peak-to-peak amplitude of 2000. Because both the master axis and the follower axis use the same units, we can use a Following ratio of 1:1, and control the amplitude with the SINAMP command. In order to provide a gentle start, the oscillation is started at the bottom of the cycle, where the velocity is zero.

```
D-3072,6000      ; Establish desired bottom of oscillation
MC00             ; Put axes into preset move mode
GO11            ; Move axes to desired center locations
FOLMAS16,26     ; Axis 1 follows sine 1, axis 2 follows sine 2
FOLRN1,1        ; Set both axes to 1:1 ratio
FOLRD1,1
FOLEN11        ; Put axes into following mode
MC11           ; Put axes into continuous following
SINAMP1024,1000 ; Establish center to peak amplitudes
FVMFRQ3600,360 ; Establish frequencies
@FVMACC9999999 ; Reach frequencies rapidly
SINANG270,270   ; Start angles at bottom of cycle
GO11           ; Lock follower axes to master (not started yet)
SINGO11        ; Start sine waves (and hence follower axis motion)
```

In observing operation, it is determined that for axis 2, the bottom of the cycle is correct

(minimum value 6000), but the top of cycle needs to change from 8000 to 8192. In order to correct this, both the center and amplitude need to increase by 96 ( $192/2 = 96$ ):

```
FSHFD,96 ; Shift center of axis 2 only
SINAMP,1096 ; Increase amplitude of axis 2 only
```

### Following an Integer Variable

Syntax is FOLMASn8. This option allows an axis to follow the integer variable (VARI) specified with “n”; that is, VARIn. The range for “n” is 1-8 (VARI1 – VARI8). For example, FOLMAS, 48 assigns axis 2 to follow VARI4.

This option is particularly useful in conjunction with the INVARI (Map Inputs to a VARI Variable) feature. INVARI continuously updates a specified VARI variable with the value of a specified group of digital inputs, allowing an axis to follow a binary input pattern. Another useful way to update the value of the VARI variable is to calculate its value in a PLCP program (launched with the SCANP command).

The INVARI and SCANP options for updating VARI are good choices, because both are performed every system update, thus facilitating smooth Following motion. It is also possible to use an extra task (multi-tasking) to calculate VARI values, but the resulting updates will not be as fast (not perfectly periodic); consequently, Follow motion will be less smooth.

### Define the Master and Follower Scaling Factors (SCLMAS) ... if required

If you will be scaling your motion parameters (distance, velocity, acceleration/deceleration), be aware that the only variance from non-Following scaling is that the master source (distance) is scaled by the SCLMAS value. Virtual master sources are not scaled.

Typically, the master and follower scale factors are programmed so that master and follower units are the same, but this is not required. Consider the scenario below as an example.

The master is a 1000-line encoder (4000 counts/rev post-quadrature) mounted to a 50 teeth/rev pulley attached to a 10 teeth/inch conveyor belt, resulting in 80 counts/tooth (4000 counts/50 teeth = 80 counts/tooth). To program in inches, you would set up the master scaling factor with the SCLMAS800 command (80 counts/tooth \* 10 teeth/inch = 800 counts/inch).

The follower axis is a servo motor with position feedback from a 1000-line encoder (4000 counts/rev). The motor is mounted to a 4-pitch (4 revs/inch) leadscrew. Thus, to program in inches, you would set up the follower scaling factor with the SCLD16000 command (4000 counts/rev \* 4 revs/inch = 16K counts/inch).

**NOTE:** Additional details on scaling and non-scaled units of measure are presented on page 48.

### Define the Follower-to-Master Following Ratio (FOLRN & FOLRD)

The FOLRN and FOLRD commands establish the goal ratio between the follower and master travel, just as the V command establishes the goal velocity for a typical non-Following move. The FOLRN command specifies the ratio's numerator (follower travel), and the FOLRD command specifies the ratio's denominator (master travel). If the denominator (FOLRD) is not specified, it is assumed to be 1.

*FOLRNF can be used to define a final ratio for compiled Following profiles (see page 139).*

FOLRN and FOLRD are specified with two positive numbers, but the resulting ratio applies to moves in both directions; the actual follower direction will depend on the direction commanded with the D command and master direction. Numeric variables (VAR) can be used with these commands for follower and/or master parameters (e.g., FOLRN (VAR1) : FOLRD3). The maximum value of the resulting quotient is 127 to 1.

For a preset Following move (MCØ mode), the FOLRN/FOLRD ratio represents the maximum allowed ratio. For a continuous move (MC1 mode), it represents the final ratio reached by the follower axis.

#### Example

As an example, assume the follower-to-master ratio is set to 5-to-3 for an axis (FOLRN5 : FOLRD3). The first parameter (5) is scaled by the SCLD value to give follower steps. The second parameter (3) is scaled by the SCLMAS value to give master steps. If the SCLD setting is 25000 and the SCLMAS setting is 4000, the follower-to-master step ratio would be  $5 * 25000$

to 3 \* 4000, or 125 follower steps for every 12 master steps.

## Define the Master Distance (FOLMD)

The “master distance” for moves in the Following mode (FOLEN1) is analogous to the move time for normal time-based moves with Following disabled (FOLENØ). For time-based moves, the time required to ramp to a new velocity (MC1 mode) or move to a new position (MCØ mode) is determined indirectly by the acceleration (A), deceleration (AD), and velocity (V) command values. For Following mode moves, a ramp to a new ratio (MC1 mode) or a move to a new position (MCØ mode) takes place over a specific master distance, not over a specific time. This distance is defined directly by the user with the FOLMD command.

In other words, the FOLMD command defines the master distance over which a preset follower move will take place, or the master distance over which a continuous follower move will change from its current ratio (including zero) to the commanded ratio (ratio established by FOLRN and FOLRD).

By carefully specifying a master distance (FOLMD), a precise position relationship between master and follower during all phases of the profile is ensured.

⇒ **HINT:** If the follower axis is in continuous mode (MC1) and the master is starting from rest, setting FOLMD to Ø will ensure precise tracking of the master's acceleration ramp.

If scaling is enabled (SCALE1), the FOLMD value is scaled by the SCLMAS parameter.

Examples and more information on this topic can be found below in the section titled *Follower vs. Master Move Profiles*.

## Enable the Following Mode (FOLEN1)

When an axis is configured as a follower with the FOLMAS command, it will continuously monitor the position and motion of its master, even if the follower is at rest. This allows subsequent motion to be related to the motion of the master via ratios (FOLRN/FOLRD) and ramping over master distances (FOLMD). Such moves are done with Following enabled (FOLEN1).

It is also possible, and sometimes desirable, to have the follower axis motion independent of master axis motion, yet still “aware” of master position. For example, a move might need to start at a specified master position, yet finish in a fixed time, independent of the master speed. This move would be performed with Following disabled (FOLENØ).

*Following can be enabled or disabled between moves, as needed, without affecting the monitoring of the master.*

If a move is performed with Following disabled, its motion profile is determined by the acceleration, deceleration, and velocity specified with the A, AD, and V commands. Its motion is the same as if the axis were not configured as a follower, but the axis does monitor the master.

If a move is performed with Following enabled, its profile is determined by the specified master distance (FOLMD) and Following ratio (FOLRN/FOLRD). The next section describes such profiles.

## Follower vs. Master Move Profiles

Following Status (TFSE, TFS, and FS) bits 1-4 indicate the status of follower axis in Following motion. They mimic the meaning and organization of Axis Status (TASF and AS) bits 1-4, except that each bit indicates the current state of the ratio, rather than the current state of the velocity:

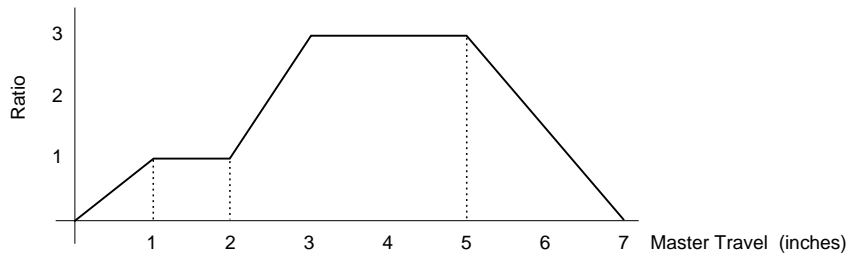
Bit No.	Function (YES = 1; NO = 0)
1	Follower in Ratio Move ..... A Following move is in progress
2	Ratio is Negative ..... The current ratio is negative (i.e., the follower axis counts are counting in the opposite direction from the master counts).
3	Follower Ratio Changing .. The follower axis is ramping from one ratio to another (including a ramp to or from zero ratio).
4	Follower At Ratio ..... The follower axis is at constant non-zero ratio.

## Continuous Positioning Mode Moves

For Following moves in the continuous positioning mode (MC1), FOLMD specifies the exact master travel distance over which the follower axis ratio changes. This will be required for any application that uses multiple ratios and continuous moves for the construction of precisely defined multi-segment moves.

⇒ **HINT:** If the follower is in continuous mode (MC1) and the master is starting from rest, setting FOLMD to 0 will ensure precise tracking of the master's acceleration ramp.

In the profile below, the first two moves each change ratio over one master inch, and the final ramp to zero takes place over two master inches.



### Example

In the sample 6K code below, assume the follower has a 1000-line encoder on axis 1, connected to a 2-pitch leadscrew. This gives 8000 follower axis steps per inch. The master is a toothed belt with a pulley connected to encoder 2, such that there are 800 master steps per inch.

```

SCALE1           ; Enable scaling
SCLD8000         ; Set axis 1 scaling so that follower commands are in inches
SCLMAS800        ; Set axis 1 scaling so that master commands are in inches
COMEXC1         ; Allow commands during motion
FOLMAS21         ; Define axis 1 master to be encoder input 2
FOLEN1          ; Enable Following on axis 1 (will follow encoder 2)
FOLMD1          ; Follower to change ratio over 1 inch of the master travel
FOLRD1          ; Set Following ratio denominator to 1 for subsequent ratios
D+              ; Set direction positive
MC1             ; Mode set to continuous clockwise moves
FOLRN1          ; Set Following ratio numerator to 1 (ratio set to 1:1)
FMCNEW1         ; Restart master cycle counting
GO1             ; Start axis 1 from rest to reach velocity of master
                ; (encoder 2)

WAIT(1FS.4=B1)  ; (for stepper axes) Wait until follower axis is at ratio
FOLRN3          ; Set Following ratio numerator to 3 (ratio set to 3:1)
GOWHEN (1PMAS>=2) ; Enable motion pre-processing so that the next ramp
                ; begins at master position 2
GO1             ; Follower starts ratio change to 3 to 1 at master position 2
FOLRN0          ; Set Following ratio numerator to zero (ratio is 0 to 1)
FOLMD2          ; Follower axis changes ratio over 2 inches of master travel
WAIT(1AS.26=b0) ; Wait until the previous ramp is started
                ; (GOWHEN bit in axis status register is cleared)
WAIT(1FS.3=b0)  ; Wait until the previous ramp is finished
GOWHEN(1PMAS>=5) ; Enable motion pre-processing so that follower motion
                ; begins at master position 5
GO1             ; Wait for master to reach 5 revolutions before
                ; follower axis starts ratio change to 0 (zero)

```

## Preset Positioning Mode Moves

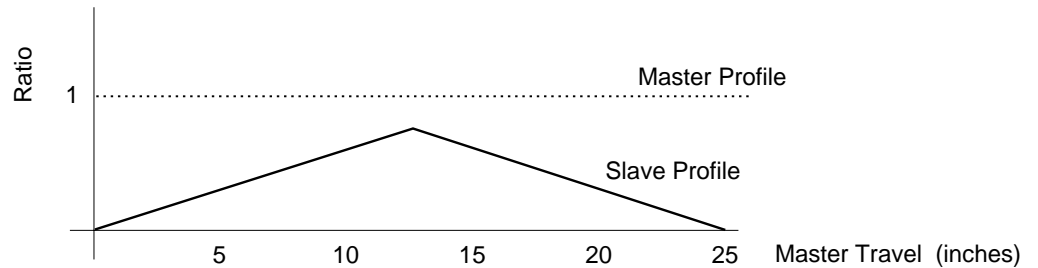
For preset positioning mode (MCØ) moves, the FOLMD parameter is the master distance over which the entire follower axis move is to take place.

As an example, a follower axis is to move 20 inches over a master distance of 25 inches with a maximum ratio of 1:1 (ratio set with the FOLRN1 and FOLRD1 commands). The program and a diagram of the move profiles are provided below.

```
FOLMAS31      ; Define axes 1 master to be encoder input port 3
FOLMD25      ; Define follower to perform the move over 25 inches
              ; of the master (encoder 3)
MC0          ; Set positioning mode to preset
MA0          ; Set preset positioning mode to incremental
D20          ; Set follower distance to 20 inches
FOLRN1       ; Set Following ratio numerator to 1
FOLRD1       ; Set Following ratio denominator to 1 (ratio is 1:1)
GO1         ; Perform the follower move
```



If the master distance specified is too large for the follower distance and ratio (FOLRN and FOLRD) commanded, the follower will never actually reach the commanded ratio, and the move profile will look similar to that below. Here, the FOLMD is 25 inches and the follower is commanded to move 10 inches:



If the master distance is too small for the follower distance and ratio commanded, the 6K controller will not perform the move at all. For example, if the FOLMD is 25, the ratio is 1:1, and the follower is commanded to move 30 inches, the move will not even be attempted. The error message “INVALID DATA” will be displayed (depending on the ERRVLV setting) and program execution will continue.

## Performing Phase Shifts

Following Status (TFSF, TFS and FS) bits 9-12 indicate the shift status of the follower axis. Shifting is super-imposed motion, but if viewed alone, can have its own status. In other words, bits 10-12 describe only the shifting portion of motion.

Bit No.	Function (YES = 1; NO = 0)
9	OK to Shift ..... Conditions are valid to issue shift commands (FSHFD or FSHFC).
10	Shifting now ..... A shift move is in progress.
11	Shift is Continuous..... An FSHFC-based shift move is in progress.
12	Shift Dir is Neg..... The direction of the shift move in progress is negative.

When a follower axis is following a master continuously, it might be necessary to adjust, or *shift*, the Following *phase* (follower's position with respect to the master) independent of motion due to ratio moves. The FSHFC and FSHFD commands allow time-based follower moves to be superimposed upon ratio Following moves. Because phase shifts are time-based, they are independent of master motion; in fact, the master can be at rest and a shift can still be performed.

Use the FSHFD command to perform a preset shift move with a specific change in follower phase. The FSHFD distance value will be scaled by SCLD if scaling is enabled (SCALE1).

Use the FSHFC command to superimpose a continuous shift move in the positive (FSHFC1) or negative (FSHFC2) direction. The FSHFC parameters stop (0) and kill (3) can be used to halt a continuous FSHFC move or a preset FSHFD move without affect the ratio motion.

The most recently defined velocity and acceleration (i.e., the V and A values) for the follower will determine the basis for the superimposed shift move profile for both FSHFC and FSHFD moves. The commanded velocity of the FSHFC or FSHFD move will be added to the current velocity at which the follower is performing the Following move. For example, assume a follower is traveling at 1 rps in the positive direction as a result of following a master. If a FSHFC move is commanded in the positive direction at 2 rps, the follower's actual velocity (after acceleration) will be 3 rps.

For servos, shifting can be performed whenever Following is enabled (FOLEN1). For steppers, this can only be done while Following is enabled and the follower is either not in a move, or is in continuous positioning mode (MC1) and moving at constant ratio. For both products, TFS/FS bit 9 indicates when a shift is allowed.

The current follower position (TPSLV value) and the net follower shift accumulated since the most recent FOLEN1 command (TPSHF value) can be read into numeric variables (VAR) using the PSLV and PSHF commands, respectively (e.g., VAR6=2PSLV). They can also be used for subsequent decision making (e.g., IF (3PSHF<6), GOWHEN (1PSLV>VAR2), etc.). The TPSHF and PSHF values are set to zero each time the Following mode is enabled (FOLEN1), even if the follower is already in Following mode. This provides a way of clearing these values for programming convenience.

Note that the distance traveled during the time-based deceleration due to stop, kill, or limits is included in the PSHF value. By comparing "before and after" values of PSHF, a 6K program can calculate how much shift was required to perform visual- or sensor-based alignment of a master/follower phase relationship.

## Phase Shift Examples

An FSHFC or FSHFD move can be needed to adjust the follower position *on the fly* because of a load condition that changes during the continuous Following move. Below are programming examples to demonstrate both shift methods.

### FSHFC Example

An operator is visually inspecting the follower's continuous Following motion with respect to the master. If he notices that the master and follower are out of synchronization, it might be desirable to have an interrupt programmed (e.g., activated by pressing a push-button switch) that will allow the operator to move the follower at a superimposed correction speed until the operator chooses to have the follower start tracking the master again (by releasing the push-button). The programming example below illustrates this.

Assume all scale factors and set-up parameters have been entered for the master and follower. In this example, the follower (axis 1) is continually following the master at a 1:1 ratio. If the operator notices some mis-alignment between master and follower, he can press 1 of 2 pushbuttons (connected to onboard trigger inputs 1 and 2) to shift the follower in the positive or negative direction until the button is released. After the adjustment, the program continues on as before.

```
DEL SHIFT          ; Delete program before defining
DEF SHIFT          ; Begin definition of program called SHIFT
COMEXS1           ; Continue command execution after stop
COMEXC1           ; Continue command execution during motion
FOLMAS21          ; Axis 2 encoder input is the master for axis 1
FOLRN1            ; Set follower-to-master Following ratio numerator to 1
FOLRD1            ; Set follower-to-master Following ratio denominator to 1
                  ; (ratio set to 1:1)
FOLEN1            ; Enable Following mode on axis 1
A25               ; Set acceleration
AD18              ; Set deceleration
V5               ; Set velocity
D+               ; Set direction to positive
MC1               ; Select continuous positioning mode
GO1               ; Start following master continuously
VARB1=b10         ; Define onboard input pattern 1 and assign to VARB
VARB2=b01         ; Define onboard input pattern 2 and assign to VARB
$TESTIN           ; Define label called TESTIN
IF(IN=VARB1)      ; IF statement (if input 1 is activated, do the jump)
JUMP SHIFTP       ; Jump to shift follower axis in the positive direction
                  ; when pattern 1 active
NIF               ; End of IF statement
IF(IN=VARB2)      ; IF statement (if input 2 is activated, do the jump)
JUMP SHIFTN       ; Jump to shift follower axis in the negative direction
                  ; when pattern 2 active
NIF               ; End of IF statement
JUMP TESTIN       ; Return to main program loop
$SHIFTP           ; Define label called SHIFTP (subroutine to shift in
                  ; the positive direction)
FSHFC1            ; Start continuous follower shift in positive direction
WAIT(IN.1=B0)     ; Continue shift until input bit 1 is deactivated
FSHFC0            ; Stop shift move
WAIT(1FS.10=B0)   ; (stepper axes only) Wait until the shift is completed
JUMP TESTIN       ; Return to main program loop
$SHIFTN           ; Define label called SHIFTN (subroutine to shift
                  ; in the negative direction)
FSHFC2            ; Start continuous follower axis shift move in the
                  ; negative direction
WAIT(IN.2=B0)     ; Continue shift until bit 2 is deactivated
FSHFC0            ; Stop shift move
WAIT(1FS.10=B0)   ; (stepper axes only) Wait until the shift is completed
JUMP TESTIN       ; Return to main program loop
END               ; End definition of program called SHIFT
```

In this example, the follower axis follows a master that moves in a continuous cycle. Once each cycle, the master and follower both pick parts. The master's part is detected by a sensor connector to trigger 1B, and the follower's part is detected by a sensor connected to trigger 1A. After both parts are detected, they must be aligned. The sensors are mounted 2 inches apart from each other, so that proper alignment would result in 2 inches of follower axis travel between detection of the master's part and detection of the follower's part.

The follower axis position is sampled when each of the sensors activates. The difference between the follower axis positions is compared to the required 2 inches. If the measured difference is greater than or less than 2 inches, then a shift move to correct the alignment is made. At that point, the follower axis will then start tracking the master again. The follower axis (axis 1) is continually following the master at a 1:1 ratio.

```

DEL ALIGN          ; Delete program before defining
DEF ALIGN          ; Begin definition of program called ALIGN
COMEXC1           ; Allow continuous command execution during motion
FOLMAS31          ; Axis 3 Encoder input is the master for follower axis 1
FOLRN1            ; Set follower-to-master ratio numerator to 1
FOLRD1            ; Set follower-to-master ratio denominator to 1
                  ; (ratio set to 1:1)
FOLEN1            ; Enable Following mode on axis 1
MC1               ; Enable continuous positioning mode
D+                ; Set direction to positive
GO1               ; Start Following master continually
INFNC1-H          ; Enable trigger input 1A to latch position of follower
                  ; when the follower's part is detected
INFNC2-H          ; Enable trigger input 1B to latch position of follower
                  ; when the master's part is detected
$SYNCLP           ; Main loop where synchronizing moves occur
WAIT(TRIG.1=b1 AND TRIG.2=b1)
                  ; Wait for both follower and master inputs to occur
VAR10=1PCCA       ; Load VAR10 with the follower commanded position due
                  ; to follower input activation
VAR11=1PCCB       ; Load VAR11 with the follower commanded position due
                  ; to master input activation
VAR12=VAR10-VAR11 ; Load VAR12 with the offset distance
VAR13=VAR12-2     ; Calculate the required shift
FSHFD(VAR13)      ; Perform synchronization move of distance in VAR13
JUMP SYNCLP       ; Return (jump) to main program loop
END               ; End of program

```

## Geared Advance Following

The FGADV command provides the ability to super-impose an advance or retard on Following motion. This is the same ability provided by the FSHFD command (see *Performing Phase Shifts* above), except that the super-imposed motion is also geared to master motion. The FGADV command has the positive or negative “advance” distance as a parameter, but it initiates motion instead of simply setting up the distance. The shape of the super-imposed profile is determined by the FOLMD, FOLRN, and FOLRD commands (just as a normal preset Following move).

The FGADV command profile can be delayed with the GOWHEN command.

A FGADV move can be performed only while the conditions below exist (Following status bit 23, reported with the FS, TFS, and TFSF commands, indicates that it is “OK to do FGADV move”):

- Master is specified with a FOLMAS command
- Following is enabled with the FOLEN command
- The follower axis is either not moving, or moving at constant ratio in continuous mode (MC1)

A FGADV move cannot be performed:

- During a preset (MC0) move
- In a compiled profile or program Following Status (FS, TFS, and TFSF) bit 24 reports if a “FGADV move is underway”.

*Example*

```

COMEXC1      ; All command processing during motion
FOLRN25     ; Set numerator of follower-to-master Following ratio
FOLRD10     ; Set denominator of follower-to-master Following ratio
FOLMD1000   ; Set master distance to 1000 units
MC1         ; Enable continuous positioning mode
D+          ; Set direction to positive
FOLEN1      ; Enable Following
GO          ; Ramp up to a 2.5:1 ratio over 1000 master distance units
FOLMD500    ; Set master distance to 500 units
FOLRN13     ; Superimposed ratio will be 1.3 (added to 2.5 = 3.8 total)
WAIT(FS.23=B1) ; Wait for OK to do geared advance
              ; (in this case, ramp is complete)
FGAVD400    ; Advance the follower axis 400 counts over a distance
              ; of 500 master counts
WAIT (FS.23=B1) ; Wait for OK to do geared advance (in this case,
              ; FGADV400 super-imposed profile is complete)
FGADV-400   ; Retard the follower axis 400 counts over a distance of
              ; 500 master counts (2.5 - 1.3 = 1.2 net ratio)

```

## Summary of Ratio Following Commands

ANIMAS.....Assigns an analog input to be used as a master in a FOLMAS assignment (requires a ANI SIM located on an expansion I/O brick)

FGADV.....Defines the geared advance distance

FOLEN.....Enables or disables Following mode

FOLMAS.....Defines masters for follower axes

FOLMD.....Defines the master distance over which follower acceleration or moves are to take place

FOLRN and FOLRD.....Establishes the maximum follower-to-master ratio for a preset move or the final ratio for a continuous move (FOLRN for the numerator and FOLRD for the denominator)

FSHFD.....Initiates preset advance or retard (shift) of follower position during continuous Following moves

FSHFC.....Initiates continuous advance or retard (shift) of follower position (or kills or stops the shift portion of motion) during continuous Following moves

FVMACC.....Establishes the rate at which the virtual master count frequency (FVMFRQ) can change for an axis.

FVMFRQ.....Defines the frequency of the virtual master count.

SCLD.....Sets the follower distance scale factor

SCLMAS.....Sets the master distance scale factor

SINAMP.....Defines the amplitude of the internal sine wave

SINANG.....Defines the phase angle of the internal sine wave

SINGO.....Initiates the internal sine wave

TPSHF or [ PSHF ] .....Transfers or assigns the net position shift since constant ratio

TPSLV or [ PSLV ] .....Transfers or assigns the current follower position

TVMAS or [ VMAS ] .....Transfers or assigns the velocity of the master axis

## Electronic Gearbox Application for Ratio Following

An electronic gearbox is a classic application for Ratio Following. Suppose we need a three-output gearbox, with all three outputs geared off the same input. Also, each gear ratio must be individually programmed. In this example, a 1000-line encoder is mounted to the input shaft of a master motor, giving 4000 master counts per revolution after quadrature. This encoder is fed into the encoder input on axis 4 (**ENCODER 4** connector) of the 6K controller. The motors on axes 1, 2, and 3 have resolutions of 2000, 4000, and 5000 steps/revolution.

In this example, a precise position relationship is not required between master and followers, but a ratio change during motion is required. The ratio change will take place over one master revolution in order to avoid abrupt acceleration of the follower. The followers will accelerate to their initial ratios (in terms of revolutions), and after 10 seconds the gear ratio on each axis will change to their final ratios.

In this example, **ENCODER 4** is specified as the master. This means that this *external* master encoder is wired to the 6K controller's axis 4 encoder input.

### Program (code portion)

```
*****  
; execute scaling before the program is executed  
SCALE1          ; Enable scaling  
SCLD2000,4000,5000 ; Set follower scale factors equal to the  
                  ; motor resolutions  
SCLA2000,4000,5000 ; Set acceleration scale factors equal to the  
                  ; motor resolutions  
SCLMAS4000,4000,4000 ; Master scale factor to number of pulses per rev  
*****  
COMEXC1        ; Allow continuous command execution during motion  
FOLMAS+41,+41,+41 ; Encoder 4 is master axis for follower axes 1-3.  
                  ; The + sign indicates that the master input is not  
                  ; inverted before it is read as master counts.  
  
FOLEN111       ; Enable followers to follow  
FOLMD1         ; Change to new ratio over one master revolution  
FOLRN1,3,2     ; Set follower-to-master ratio numerators to 1, 3 & 2  
FOLRD1,1,1     ; Set all follower-to-master ratio denominators to 1  
                  ; (initial Following ratio for axis 1 is 1:1, axis 2  
                  ; is 3:1, and axis 3 is 2:1)  
  
MC111         ; Enable continuous mode  
GO111         ; Begin follower continuous Following move  
TIMST0        ; Reset and start the timer  
FOLRN10,6,1   ; Set follower-to-master ratio numerators to 10, 6 & 1  
FOLRD1,7,2    ; Set follower-to-master ratio denominators to 1, 7 & 2  
                  ; (change Following ratios: axis 1 is 10:1,  
                  ; axis 2 is 6:7, axis 3 is 1:2)  
WAIT(TIM>=10000) ; Wait 10 seconds to change to new ratio  
GO111         ; Start moving to new ratio
```

## Trackball Application for Ratio Following Motion

A trackball is a two-axis, two-dimensional positioning device; just as a mouse is used to position the cursor on a computer screen, a trackball could be used to position an X-Y stage.

In this example, a two-axis trackball is needed that can do fine and coarse positioning of an X-Y stage. The fine or coarse setting is selected by the user with a two-position switch connected to onboard trigger input 1 on the 6K controller. Onboard trigger input 2 on the 6K controller is used to switch back and forth from trackball to standard point-to-point positioning mode. *Unlocking* the stage from the trackball is necessary because of other point-to-point move requirements elsewhere in the 6K controller program.

The trackball housing has two encoders mounted at 90 degrees to each other that are driven by rubber wheels in contact with the ball. The stage is driven by motors and leadscrews.

For one inch of trackball motion to result in one inch of stage motion, the follower-to-master ratio must be 10-to-1; this will be the ratio for coarse positioning. The fine positioning ratio will be one tenth of that, or 1-to-1. When programmable input 1 is low, coarse positioning is selected, and when trigger input 2 goes low, the stage becomes locked to the trackball. Each change of state of trigger inputs 1 and 2 calls a different subroutine in the 6K controller program; however, the ratios can only change if the stage is locked to the trackball positioning mode.

The trackball is initially unlocked and fine positioning is selected.

### Program

```
SCALE1          ; Enable & define scale factors before loading program
SCLD4000,4000   ; Follower axes 1 and 2 have 4000 counts per rev
                ; resolution post-quadrature
SCLV4000,4000   ; Set velocity scaling factors
SCLA4000,4000   ; Set acceleration scaling factors
SCLMAS200,200   ; Master axis 3 and 4 have 200 counts per rev
                ; resolution post-quadrature
DEL UNLOCK      ; Delete program before defining
DEF UNLOCK      ; Program that unlocks the stage from the trackball
S11             ; Stop moves
WAIT(1AS.1=B0 AND 2AS.1=B0) ; (for steppers only)
                ; Wait for motion to stop on both axes
ONIN.2-1        ; Set up onboard input 2 to lock trackball to stage
FOLEN00         ; Stop Following mode
ONP LOCK        ; Select LOCK as an ON program
JUMP WAITLP     ; Return to main loop
END             ; End program definition

DEL LOCK
DEF LOCK        ; Program that locks the stage to the trackball
FOLEN11        ; Enable Following on both axis
IF(VAR1=0)     ; If in the FINE mode, set ratio to 0.5:1
FOLRN.5,.5
ELSE
FOLRN1.5,1.5   ; If in the COARSE mode, set ratio to 1.5:1
NIF
ONIN.2-0       ; Set up input 2 to unlock trackball from stage
GO11           ; Start following the trackball
ONP UNLOCK     ; Select UNLOCK as ON program
JUMP WAITLP   ; Return to main loop
END           ; End program definition

DEL COARSE
DEF COARSE     ; Declare COARSE label
FOLRN1.5,1.5   ; Coarse positioning ratio of 1.5 to 1
GO11           ; Move to begin travel at new ratio
VAR1=1         ; Flag to indicate we are in coarse mode
END           ; Return to main loop
```

*(Continued on next page)*

**Trackball Program (Continued from previous page)**

```
DEL FINE
DEF FINE          ; Subroutine to assign fine positioning
FOLRN.5,.5       ; Fine positioning ratio is 0.5:1
GO11            ; Move to begin travel at new ratio
VAR1=0          ; Flag to indicate we are in fine mode
END              ; Return to main loop

DEL TRACK
DEF TRACK        ; Main track ball program
IF(IN.1=b1 AND VAR1=0) ; If input 1 is set to 1 and we are
                    ; currently in fine mode, then enter coarse mode
    GOSUB COARSE ; Set high ratio
NIF
IF(IN.1=b0 AND VAR1=1) ; If input 1 is set to 0 and we are
                    ; currently in coarse mode, then enter fine mode
    GOSUB FINE    ; Set low ratio
NIF
IF(LIM.1=b0 OR LIM.2=b0) ; If a limit is hit, allow track ball to move off
D~                      ; Back off of the limit, axis 1
GO1
NIF
IF(LIM.4=b0 OR LIM.5=b0)
D,~                      ; Back off of the limit, axis 2
GOX1
NIF
END

DEL MAIN
DEF MAIN        ; Begin definition of main program
V1,1           ; Set non-Following move parameters
A99,99
LH3,3,0,0
FOLMAS+31,+41 ; Encoder 3 is master axis for follower axes 1 and
                ; Encoder 4 is master axis for follower axes 2.
                ; The follower axes will move in the same direction as
                ; the master.

FOLRN.5,.5
INDEB250       ; Noisy switch debounce of 250 milliseconds for
                ; onboard trigger inputs
FOLRD1,1       ; Initial follower-to-master ratio is set to fine
                ; positioning (0.5:1)
VAR1=0         ; Flag set to fine positioning
MC11          ; Set both axes 1 and 2 to continuous positioning mode
FOLEN00       ; Following is initially disabled
COMEXC1       ; Continue command execution during motion.
COMEXS1       ; Continue command execution after stop
COMEXL11      ; Continue command execution after a limit is hit
SGP20,20      ; Set servo gains
SGV5,5
DRIVE11       ; Enable drives
DRFEN11      ; Enable checking for drive fault input states
DRFLVL11     ; Set drive fault level for Compumotor 670-T drives
ONP LOCK     ; Select LOCK as an ON program
ONIN.2-1     ; Trigger 1B locks trackball to stage
ONCOND1000   ; Inputs enabled for interrupts
$WAITLP      ; Main program loop
IF(IN.2=b1)  ; If trigger input 1B is set to 1 (stage locked),
                ; enter trackball mode
    GOSUB TRACK
NIF          ; End of IF statement

; *****
; * Other user programs can be added here for performing *
; * motion when the stage is not locked to the trackball. *
; *****

JUMP WAITLP   ; Return to main loop
END           ; End program definition
```

# Master Cycle Concept

Ratio Following can also address applications that require precise programming synchronization between moves and I/O control based on master positions or external conditions. The concept of the master cycle greatly simplifies the required synchronization.

A master cycle is simply an amount of master travel over which one or more related follower axis events take place. The distance traveled by the master in a master cycle is called the *master cycle length*. A *master cycle position* is the master position relative to the start of the current master cycle. The value of master cycle position increases as positive-direction *master cycle counts* are received, until it reaches the value specified for master cycle length. At that point, the master cycle position becomes zero, and the *master cycle number* is incremented by one—this condition is called *rollover*.

The master cycle concept is analogous to minutes and hours on a clock. If the master cycle is considered an hour, then the master cycle length is 60 minutes. The number of minutes past the hour is the master cycle position, and current hour is the master cycle number. In this analogy, the master cycle position decrements from 59 to zero as the hour increases by one.

By specifying a master cycle length, periodic actions can be programmed in a loop or with subroutines that refer to cycle positions, even though the master can be running continuously. To accommodate applications where the feed of the product is random, the start of the master cycle can be defined with trigger inputs. Two types of *waits* are also programmable to allow suspension of program operation or follower moves based on master positions or external conditions.

## Master Cycle Commands

Following Status (TFSE, TFS and FS) bits 13-16 indicate the status of master cycle counting. If a following application is taking advantage of master cycle counting, these bits provide a quick summary of some important master cycle information:

Bit No.	Function (YES = 1; NO = 0)
13	Master Cyc Trig Pend.....A master cycle restart pending the occurrence of the specified trigger.
14	Mas Cyc Len Given .....A non-zero master cycle length has been specified with FMCLLEN.
15	Master Cyc Pos Neg.....The current master cycle position (PMAS) is negative. This could be by caused by a negative initial master cycle position (FMCP), or if the master is moving in the negative direction.
16	Master Cyc Num > 0.....The master position (PMAS) has exceeded the master cycle length (FMCLLEN) at least once, causing the master cycle number (NMCY) to increment.

### Master Cycle Length (FMCLLEN)

The FMCLLEN command is used to define the length of the master cycle. The value entered with this command is scaled by the SCLMAS parameter to allow specification of the master cycle length in user units. This parameter must be defined before those commands that wait for periodically repeating master positions are executed.

The default value of FMCLLEN is zero, which means the master cycle length is practically infinite (i.e., 4,294,967,246 steps, after scaling). If a value of zero is chosen, the master cycle position will keep increasing until this very high value is exceeded or a new cycle is defined with the FMCNEW command (or triggered after a TRGFNCx1 command) described below. If a non-zero value for FMCLLEN is chosen, the internally maintained master cycle position will keep increasing until it reaches the value of FMCLLEN. At this point, it immediately rolls over to zero and continues to count.

The master cycle length can be changed with the FMCLLEN command even after a master cycle has been started. The new master cycle length takes affect as soon as it is issued. If the new master cycle length is greater than the current master cycle position, the cycle position will

not change, but will rollover when the new master cycle length is reached. If the new master cycle length is less than the current master cycle position, the new master cycle position becomes equal to the old cycle position minus one or more multiples of the new cycle length.

*Example Code*

```
FMCLen23,10,12,22 ; Set master cycle length for all four axes:
                  ; (axis 1: 23 units; axis 2: 10 units;
                  ; axis 3: 12 units; and axis 4: 22 units)
```

Restart Master Cycle Counting (FMCNEW or TRGFNCx1xxxxxx)

Once the length of the master cycle has been specified with the FMCLen command, master cycle counting can be restarted immediately with the FMCNEW command, or based on activating a trigger input as specified with the TRGFNCx1 command. The new master cycle count is started at an initial position specified with the FMCP command (see below).

When the TRGFNCx1 command is used, the restart of master cycle counting is pending activation of the specified trigger. If an FMCNEW command is issued while waiting for the specified trigger to activate, counting is restarted immediately with the FMCNEW command, and the TRGFNCx1 command is canceled.

**When using TRGFNCx1**

- Before the TRGFNC command can be used, you must first assign the trigger interrupt function to the specified trigger input with the INFNCi-H command, where “i” is the input number of the trigger input desired for the function (input bit assignments vary by product).
- Because the 6K controller program will not wait for the trigger to occur before continuing on with normal program execution, a WAIT or GOWHEN condition based on PMAS will not evaluate true if the restart of master cycle counting is pending the activation of a trigger. To halt program operation, the WAIT command can be used.

A new master cycle will restart automatically when the total master cycle length (FMCLen value) is reached. This is useful in continuous feed applications.

*Example Code*

```
FMCNEW11xx ; Restart new master cycle counting on axes 1 and 2
INFNC2-H ; Assign input 2 (TRG-1B) the trigger interrupt
          ; function (prerequisite to using the TRGFNC features)
1TRGFNBx1 ; When trigger input 2 (TRG-1B) goes active,
2TRGFNBx1 ; restart new master cycle counting on axes 1 and 2
```

Initial Master Cycle Position (FMCP)

The FMCP command allows you to assign **for the first cycle only**, an initial master cycle position to be a value other than zero. When master cycle counting is restarted with the FMCNEW command or with the trigger specified in the TRGFNCx1 command, the master cycle position takes the initial value previously specified with the FMCP command. The value for FMCP is scaled by SCLMAS if scaling is enabled (SCALE1)

FMCP was designed to accommodate situations in which the trigger that restarts master cycle counting occurs either before the desired cycle start, or somewhere in the middle of what is to be the first cycle. In the former case, the FMCP value must be negative. The master cycle position is initialized with that value, and will increase right through zero until it reaches the master cycle length (FMCLen). At that point, it will roll over to zero as usual.

The continuous cut-to-length example below illustrates the use of a negative FMCP (a trigger that senses the motion of the master is physically offset from the master position at which some action must take place). If it is desired that the first cycle is defined as already partially complete when master cycle counting is restarted, the FMCP value must be greater than zero, but less than the master cycle length.

To give a value for FMCP that is greater than master cycle length is meaningless since master cycle positions are always less than the master cycle length. The 6K controller responds to this case as soon as a new master cycle counting begins by using zero instead of the initial value specified with FMCP.

Transfer and  
Assignment/  
Comparison of Master  
Cycle Position and  
Number

The current master cycle position and the current master cycle number can be displayed with the `TPMAS` and `TNMCY` commands, respectively. These values can also be read into numeric variables (`VAR`) at any time using the `PMAS` and `NMCY` commands (e.g., `VAR6=NMCY`). If position capture is used, the master cycle position can be captured, and the value is available with the `TPCMS` and `PCMS` commands.

Very often, the master cycle number will be directly related to the quantity of product produced in a manufacturing run, and the master cycle position can be used to determine what portion of a current cycle is complete.

The master cycle number is sampled once per *position sampling period* (see note, left). If the master cycle length (`FMCLLEN`) divided by the master's velocity (`VMAS`) is less than the position sampling period (2 ms), then the sample (`TNMCY` or `NMCY` value) cannot be accurate.

Details on using `PMAS` in conditional expressions is provided below in *Using Conditional Statements with PMAS*.

Using Conditional  
Statements with  
Master Cycle Position  
(`PMAS`)

The current master cycle position (`PMAS`) value can be used in comparison expressions, just like other position variables such as `PC`, `PE`, and `FB`. `PMAS` is a special case, however, because its value rolls over to zero when the master cycle length (`FMCLLEN`) is met or exceeded. This means that `PMAS` values greater than or equal to the master cycle length will never be reported with the `TPMAS` command, or with expressions such as (`VAR1=1PMAS`).

The other fact that makes `PMAS` special is that master cycle counting can be restarted after the command containing the `PMAS` expression has been executed. Either the `FMCNEW` command or the `TRGFNCx1` command can be used to restart counting, each with a different effect on the evaluation of `PMAS`.

The treatment of `PMAS` in comparison expressions depends on the command using the expression, as described below. `WAIT` and `GOWHEN` are treated as special cases.

`IF`, `UNTIL`, and  
`WHILE`

These commands evaluate the current value of `PMAS` in the same way that `TPMAS` does (i.e., `PMAS` values will never be greater than or equal to the master cycle length). With these commands, avoid comparing `PMAS` to be greater than or equal to variables or constants that are nearly equal to the master cycle length, because rollover can occur before a `PMAS` sample is read that makes the comparison true. If such a comparison is necessary, it should be combined (using `OR`) with a comparison for master cycle number (`NMCY`) being greater than the current master cycle number.

Also, master cycle counting restart can be pending activation of a trigger, but this will not affect the evaluation of `PMAS` for `IF`, `WAIT`, and `WHILE`. It is simply evaluated based on counting currently underway.

`WAIT` and `GOWHEN`

These commands evaluate the current value of `PMAS` differently than `TPMAS` does, in such a way that it is possible to compare `PMAS` to variables or constants that are greater than or equal to the master cycle length and still have the comparison be reliably detected.

Effectively, `PMAS` is evaluated as if the master cycle length were suddenly set to its maximum value ( $2^{32}$ ) at the time the `WAIT` or `GOWHEN` command is encountered. It eliminates the need to `OR` the `PMAS` comparison with a comparison for master cycle number (`NMCY`) being greater than the current master cycle number. Such multiple expressions are not allowed in the `GOWHEN` command, so this alternative evaluation of `PMAS` offers the required flexibility.

This method of evaluation of `PMAS` allows commands that sequence follower axis events through a master cycle to be placed in a loop. The `WAIT` or `GOWHEN` command at the top of the loop can execute, even though the actual master travel has not finished the previous cycle. If it is desired to `WAIT` or `GOWHEN` for a master cycle position of the next master cycle, the variable or constant specified in the command should be calculated by adding one master cycle length to the desired master cycle position.

Finally, master cycle counting restart can be pending activation of a trigger (`TRGFNCx1`), and

Synchronizing  
Following Moves with  
Master Positions

this will suspend the evaluation of PMAS for these commands. PMAS is not sampled, and the comparison evaluates as false. During this time, if the pending status of master cycle counting restart is aborted with FMCNEWØ, the GOWHEN condition is also cleared, and any motion profile of any axis waiting on *that* PMAS comparison will be canceled. Otherwise, when master cycle counting is restarted by a trigger, evaluation takes place as described above. This allows GOWHEN to include waiting on a trigger without explicitly including it in the GOWHEN expression.

A final special case allows perfect synchronization between the start of a Following motion profile of a follower axis and a specified position of its master. If a GOWHEN (nPMAS >= xxx) expression is used to synchronize a follower with its own master, with the operator specifically ">=", a special synchronization occurs. Although it can be impossible for the 6K product to sample the exact master position specified, the Following motion profile is calculated from master travel based on that position. This allows for the construction of profiles in which the synchronization of master and follower positions is well defined and precisely maintained. This feature requires positive travel of the master, which can be achieved with the appropriate sign for the FOLMAS specification.

## Summary of Master Cycle and Wait Commands

FMCLen.....	Defines the length of the master cycle
FMCNEW.....	Immediately restart master cycle counting
FMCP.....	Defines the initial position of a new master cycle
GOWHEN.....	GOWHEN suspends execution of the next move on the specified axis or axes until the specified conditional statement (based on T, IN, LIM, FB, NMCY, PC, PE, PMAS, PSLV, or PSHF) is true
TNMCY or [ NMCY ] .....	Transfers or assigns the current master cycle number
TPCMS or [ PCMS ] .....	Transfers or assigns the captured master cycle position
TPMAS or [ PMAS ] .....	Transfers or assigns the current master cycle position
TRGFN.....	aTRGFNc1x initiates a GOWHEN, suspending execution of the next follower move on axis (a) until the specified trigger input (c) goes active. aTRGFNcx1 causes master cycle counting to restart when the specified trigger input (c) goes active.
WAIT.....	WAIT suspends program execution until the specified conditional statement (based on PMAS, FS, NMCY, PCMS, PSHF, PSLV, or VMAS) is true; WAIT (SS.i=b1) suspends program execution until a trigger input is activated. The "i" is the programmable input bit corresponding to the trigger input. (Input bit assignments vary by product; see the Programmable I/O Bit Patterns table on page 76 to determine the correct bit pattern for your product.)
AS and TAS bit 26.....	AS and TAS (and TASF) bit 26 is set when there is a profile suspended pending GOWHEN condition, initiated either by a GOWHEN command or a TRGFNc1 command; this bit is cleared when the GOWHEN condition is true or when a stop or kill command is issued.
ER and TER bit 14.....	ER, TER, and TERF bit 14 is set if the GOWHEN condition is already true when the GO, GOL, FGADV, FSHFC, or FSHFD command is given (ERROR bit 14 must first be enabled to check for this condition)

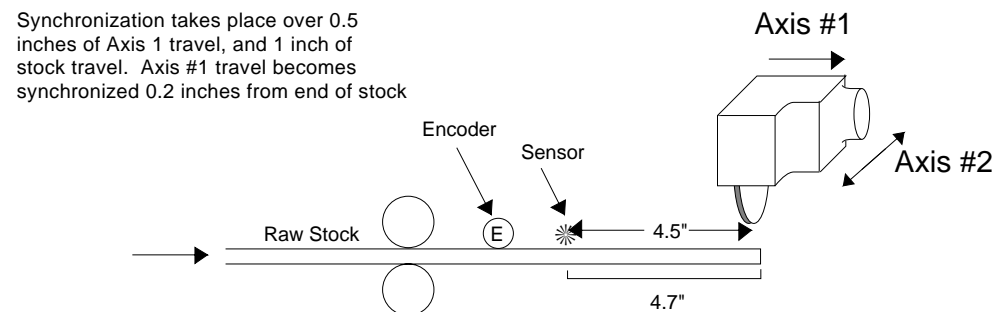
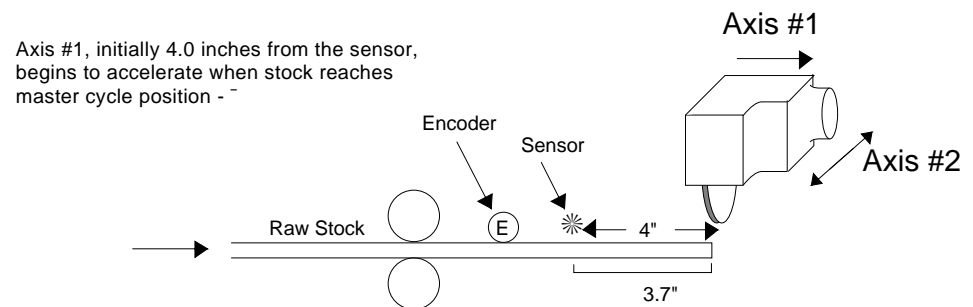
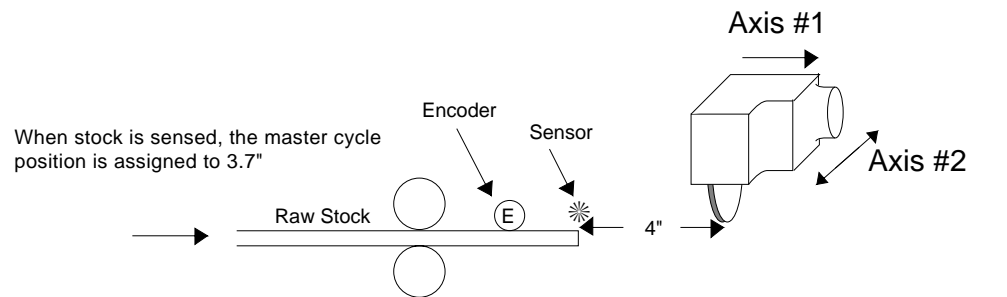
### NOTE

The continuous cut-to-length application example below illustrates the use of the master cycle concept and the commands above.

# Continuous Cut-to-Length Application

This application requires automobile trim to be cut to a pre-defined length. The saw is controlled by axes 1 and 2 on the 6K controller. It must be moving with the material while the cut is being made (axis 1), and also move perpendicular to the trim (axis 2) to actually make the cut. The trim comes in long stock that moves continuously under the cutting area.

The leading edge of the trim stock is detected with a sensor connected to trigger 1 that is located 4 inches from the home position of the saw. Axis 1 will be following the trim based on an encoder mounted on the trim via a friction wheel. The encoder is a 1000-line encoder and the wheel is geared to give 2 revolutions per inch of trim, resulting in 8,000 post-quadrature steps per inch of trim. Axis 1 has a resolution of 4,000 steps per rev and is connected to a 2-pitch leadscrew 24 inches in length. Axis 2 is similar in mechanics but its length is 10 inches. The travel on Axis 1 will be controlled by the speed at which axis 2 makes its cut. The travel on axis 2 is a fixed speed of 5 inches per second, with a fixed cross stroke of 5 inches. Limit switches are in place for safety.



The master cycle length will be set equal to the desired cut length (36" in the example below), which the operator can change by modifying variable VAR1. The cut cycle will be a continuous loop, but the first cut will be made 0.2 inches from the end of the stock to ensure an even first edge. Axis 1 will accelerate to the desired tracking ratio over 1 inch of master travel for all cuts. Assume that the home position of both axes is at position 0 inches.

The Cut-to-Length example takes advantage of being able to change master cycle length, while being careful to change it only at the beginning of a current cycle. This ensures that the current master cycle position will be less than the new master cycle length, and will not change as a result of a change in cycle length. In this example, the master cycle length and corresponding waits are redefined every cycle to the current value of VAR1. The value of VAR1 becomes the cut length, and can be changed via remote command during program execution. With minor modifications, the cut lengths and the number of iterations could be read from DATA commands (e.g., in a teach mode application).

## Program

```

SCALE1           ; Enable scaling
SCLD8000,8000    ; Set axes 1 & 2 scale factors for programming
                 ; in inches
SCLV,8000        ; Axis 2 velocity scale factor for inches/sec
SCLA,8000        ; Axis 2 accel scale factor for inches/sec/sec
SCLMAS8000      ; Master scale factor for programming in inches

DEF CUTLEN       ; Start definition of Cut-to-Length program
COMEXC1          ; Enable continuous command execution mode
COMEXS1          ; Continue execution if stop is issued
INFNC1-H         ; Enable trigger input 1 (TRG-1A) for TRGFN use
A20,20           ; Set acceleration
V5,5             ; Set velocity
MA11             ; Absolute positioning mode for non-Following moves
VAR1=36          ; Desired cut length is 36"
VAR2=4           ; Sensor is 4" from home position of axis 1
VAR2=VAR2+0.2    ; 1st cut to be 0.2" from end of stock
VAR3=1           ; FOLMD to be set to 1"
VAR4=VAR3/2      ; Follower will travel 1/2" when accelerating to
                 ; 1:1 ratio while master travels 1"
VAR2=VAR4-VAR2   ; Take distance follower travels during accel into
                 ; account so we'll be up to speed at position = 0.2"
                 ; from end of stock. Then the cut will be made.
                 ; Initial master cycle position will be the negative
                 ; of the distance traveled during follower wait and
                 ; accel.
FOLMAS31         ; Encoder 3 is the master for follower axis 1
FMCP(VAR2)       ; Set initial master cycle position to wait length
FOLMD(VAR3)      ; Acceleration to constant ratio will take place
                 ; over VAR3" (1" here) of master travel

FOLRN1
FOLRD1           ; Following ratio is 1 to 1
1TRGFNA X1      ; Define a new master cycle on axis 1 when trigger
                 ; input 1 (TRG-A) is activated
WAIT(IN.1=b1)   ; Suspend program execution until stock is sensed
OUT.6-1         ; Turn on output for saw blade to move into position
GOWHEN(1PMAS>=0) ; Wait on first move for start of master cycle.
                 ; This will ensure being at 1:1 ratio at exactly
                 ; 0.2" from end of stock.
$NEWCUT         ; Subroutine DEF for continuous cutting
D,5             ; Axis 2 will move 5 inches across the stock
MC1             ; Axis 1 into continuous move mode.
FOLEN1          ; Enable Following on axis 1
VAR5=VAR1       ; Set VAR5 = VAR1 (Snapshot of VAR1)
FMCLN(VAR5)     ; New master cycle length is cut length
GO1            ; Start Following move on follower axis 1

```

(Continued on next page)

### Continuous Cut-to-Length Program (Continued from previous page)

```
WAIT(1FS.4=b1)      ; Wait for axis 1 to be in sync with the moving stock
GOx1                ; Once axis 1 is up to speed, move axis 2 across
                    ; stock to make the cut
WAIT(2AS.1=b0)      ; Wait for axis 2 cut to finish
S1                  ; Stop Following move on axis 1
WAIT(MOV=b0)         ; Wait until the move is complete on axis 1
FOLENO              ; Exit Following mode
OUT.6-0             ; Raise the saw blade
MCO                 ; Axis 1 into preset move mode.
D0,0                ; Move both axes back to home positions
GO11                ; Execute moves on axes 1 and 2 (execution will not
                    ; occur until motion from previous move is complete)
WAIT(MOV=b00)        ; Wait until the moves are complete on axes 1 and 2
OUT.6-1             ; Move saw blade into position for next cut
GOWHEN(1PMAS>=VAR5) ; Synchronize next move with next master cycle
GOTO NEWCUT         ; Repeat the cut cycle
END                 ; End program definition
```

## Technical Considerations for Following

---

In the introduction to Following (see page 166), the algorithm for 6K controller Following was briefly discussed. Here we will address some of the more technical aspects of Following:

*Keep in mind that in all cases, the follower position is calculated from a sampled master position.*

- Performance
- Master Position Prediction
- Master Position Filtering
- Following error
- Maximum acceleration and velocity (stepper axes only)
- Factors affecting Following accuracy
- Preset vs. Continuous Following moves
- Master and follower axis distance calculations
- Using other features with Following

## Performance Considerations

When a follower axis is following a master, the 6K controller does not simply measure the master velocity to derive follower axis velocity. Instead, the 6K controller samples the master position (*position sampling period = 2 ms*) and calculates the corresponding follower axis position command. This is true even if the follower axis is in the process of changing ratios. A follower axis is not simply following velocity, but rather position. With this algorithm, the master and follower position or phase relationship is maintained indefinitely, without any drift over time due to velocity measurement errors.

The 6K controller also measures master velocity by measuring the change in master position over a number of sample periods. The present master velocity and position can be used to calculate the next commanded follower position, so the follower has no velocity-dependent phase delay. This concept is known as *Master Position Prediction* and can be enabled or disabled as needed with the `FPPEN` command.

The 6K controller's default Following algorithm should work well for most applications; however, you can change the Following algorithm to meet application-specific needs. For instance, suppose that the speed of the master is very slow, or has some vibration. For a case like this, the 6K controller allows you to filter the master position signal to generate a smooth follower position command. This is known as *Master Position Filtering* and is programmed with the `FFILT` command.

## Master Position Prediction

*Master Position Prediction* is a technique used to compensate for the fact a follower's position command cannot be calculated and implemented infinitely fast.

The master position prediction mode is enabled by default (FPPEN1) in the Following algorithm, but can be turned off as desired with the FPPENØ command.

The 6K controller measures master position once per *position sampling period* (2 ms), and calculates a corresponding follower position command. This calculation and achieving the subsequent follower commanded position requires 2 sample periods (4 ms).

If master position prediction mode is disabled (FPPENØ), waiting 2 sample periods results in a follower position lag. That is, by the time the follower reaches the position that corresponds to the sampled master position, 2 sample periods have gone by, and the master can be at a new position. Measured in time, the lag is 2 sample periods. Measured in position, the lag is 2 sample periods \* current follower velocity.

For example, suppose the follower is traveling at a speed of 25000 counts per second. If master position prediction mode is disabled (FPPENØ), the follower will lag the master by 100 counts (25000 counts/sec \* 4 ms = 100 counts).

By measuring the change in master position over sequential sample periods, the master's present velocity is calculated. The present master velocity and position are used to predict future master position. If master position prediction mode is enabled (FPPEN1, the predicted future master position is used to determine the follower's position command. In this case the follower has no velocity-dependent phase delay. The follower's velocity for a given sample will always be the velocity required to move from its current position to the next calculated position command.

If the master motion is fairly smooth and velocity is not very slow, the measurement of its recent velocity will be very accurate, and a good way of predicting future position. But the master motion can be rough, or the measurements can be inaccurate if there is no filtering (see *Master Position Filtering* below). In this case, the predicted master position and the corresponding follower position command will have some error, which can vary in sign and magnitude from one sample to the next. This random variation in follower position command error results in rough motion. The problem is particularly pronounced if there is vibration on the master.

It can be desirable to disable the master position prediction mode (FPPENØ) when maximum follower smoothness is important and minor phase delays can be accommodated.

If master filtering is enabled (FFILT\_Ø), then the prediction algorithm would be used on the filtered master position, resulting in a smoother follower position command. However, due to the delay introduced by the filtering, the prediction algorithm would not compensate for the total delay in the follower's tracking command. (See also *Master Position Filtering* below.)

**Following Status** (TFSE, TFS, and FS) bit 17 indicates the status of where or not the master position prediction mode is enabled.

## Master Position Filtering

The follower axis' position command is calculated at each *position sample period* (2 ms). This calculation is a function of the master position and the master velocity estimated from the change in master position over 2 position sample periods.

The *Master Position Filter* feature allows you to apply a low-pass filter to the measurement of master position. Master position filtering is used in these situations:

- Measurement of master position is contaminated by either electrical noise (when analog input is the master) or mechanical vibration.
- Measurement noise is minimal, but the motion that occurs on the master input is oscillatory. In this case, using the filter can prevent the oscillatory signal from

propagating into the follower axis (i.e., ensuring smoother motion on the follower axis). The bandwidth of the low-pass filter is controlled with the `FFILT` command:

<code>FFILT</code> Setting	Low pass Filter Bandwidth
∅	infinite (no filtering) – <i>default setting</i>
1	120 Hz
2	80 Hz
3	50 Hz
4	20 Hz

When considering whether or how much master position filtering to use, consider the application requirement itself. The application requirements related to filtering can be categorized into these three types:

- Type I: If an application requires smooth motion but also high follower tracking accuracy, then a heavy filtering should **not** be used. It should not be used because it can introduce too much velocity phase lag, although the motion can be smooth. In other words, the master axis in the first place should produce very smooth motion and low sensor measurement noise such that a higher level of master filtering is not needed.
- Type II: If follower axis velocity tracking error is not critical but smooth follower axis motion is desired, then you can use a higher level of master filtering to deal with sensor noise or master vibration problems.
- Type III: If it is determined that under certain dynamic conditions the master position's oscillatory measurement is purely caused by its vibration motion (noise is insignificant), and it is necessary for the follower to follow such motion, then the filter command should not be used or only use the highest bandwidth (`FFILT1`).

**Following Status** (`TFSF`, `TFS`, and `FS`) bit 18 indicates the status of master position filtering.

## Following Error

As soon as an axis becomes configured as a follower, the follower's position command is continuously updated and maintained. At each update, the position command is calculated from the current master position and velocity, and the current ratio or velocity of the follower.

Whenever the commanded position is not equal to the actual follower position, a *Following error* exists. This error, if any, can be positive or negative, depending on both the reason for the error and the direction of follower travel. Following error is defined as the difference between the commanded position and the actual position.

$$\text{Following Error} = \text{Commanded position} - \text{Actual position}$$

If the follower is traveling in the positive direction and the actual position lags the commanded position, the error will be positive. If the follower is traveling in the negative direction and the actual position lags the commanded position, the error will be negative. This error is always monitored, and can be read into a numeric variable (`VAR`) at any time using the `PER` command. The error value in follower steps is scaled by `SCLD` for the axis. This value can be used for subsequent decision making, or simply storing the error corresponding to some other event.

## Maximum Velocity and Acceleration *(Stepper Axes Only)*

The follower's attempt to faithfully follow the master can command velocities and accelerations that the follower axis is physically not able to complete. Therefore, the `FMAXV` and `FMAXA` commands are provided to set the maximum velocity and acceleration at which the follower will be allowed to move.

If the follower is commanded to move at rates beyond the defined maximums, the follower will begin falling behind its commanded position. If this happens, a correction velocity will be applied to correct the position error as soon as the commanded velocity and acceleration fall within the limitation of `FMAXV` and `FMAXA`.

The `FMAXA` and `FMAXV` commands should be used only to protect against worst case conditions, and should be avoided altogether if they are not needed. If an axis is not able to follow its profile because of limitations imposed by these commands, some correction motion will occur. If the maximum acceleration (`FMAXA` value) is set very low, some oscillation about the commanded position can occur because the follower is not allowed to decelerate fast enough to prevent overshoot.

## Factors Affecting Following Accuracy

There are additional accuracy requirements of Following applications beyond those of standard positioning. The follower axis must maintain positioning accuracy while in motion, not just at the end of moves, because it is trying to stay synchronized with the master.

Assuming parameters such as master and follower scaling and ratios have been specified correctly, the overall positioning accuracy for an application depends on several factors:

- Resolution of the master
- Resolution of the follower axis
- Position sampling accuracy
- Accuracy of the follower axis motor and drive
- Accuracy of load mechanics
- Master position prediction
- Master velocity relative to master position prediction & master position filtering
- Tuning (servo axes only)
- Repeatability of the trigger inputs and sensors

Just as with a mechanical arrangement, the accuracy errors can build up with every link from the beginning to the end. The overall worst case accuracy error will be the sum of all the sources of error listed below. The errors fall into two broad categories, namely, master measurement errors and follower errors. These both ultimately affect follower accuracy, because the commanded follower position is based on the measured master position.

It is important to understand how master measurement errors result in follower position errors. In many applications, master and follower units will be the same (e.g., inches, millimeters, degrees). These applications will require linear speeds or surface speeds to be matched (i.e., a 1:1 ratio). For example, suppose that in a rotary knife application, there were 500 master steps per inch of material, so an error in master measurement of one encoder step would result in 0.002 inches of follower position error.

If the master and follower units are not the same, or the ratio is not 1:1, the master error times the ratio of the application gives the follower error. An example would be a rotary master and a linear follower. For instance, suppose one revolution of a wheel gives 4000 master counts, and results in 10 inches of travel on the follower. The ratio is then 10 inches/revolution. The follower error that results from one step of master measurement error is  $(1/4000) * 10$  inches/revolution = 0.0025 inches.

Resolution of the Master	The best case master measurement precision is the inverse of the number of master steps per user's master unit. For example, if there are 100 master steps/inch, then the master measurement precision is 0.01 inches. Even if all other sources of error are eliminated, follower accuracy will only be that which corresponds to 1 step of the master (e.g., 0.01 inches in the previous example).
Resolution of the Follower	The best case follower axis precision is the inverse of the number of follower steps per user's position unit. For example, if there are 1000 follower steps/inch, then the follower resolution is 0.001 inches. Even if all other sources of error are eliminated, follower positioning accuracy will only be that which corresponds to 1 step of the follower. This must be at least as great as the precision required by the application.
Position Sampling Accuracy	<p>The position sampling rate for the 6K controller depends on whether it is a servo or a stepper. The sample period for is 2 ms.</p> <p>The repeatability of the sampling rate, from one sample to the next, can vary by as much as 100 <math>\mu</math>s. This affect can be eliminated by using non-zero master position filter (<code>FFILT</code>) command values. Otherwise, measurement of master position can be off by as much as (20 to 100 microseconds * master speed). This can appear to be a significant value at high master speeds, but it should be noted that this error changes in value (and usually sign) every sample period. It is effectively like a noise of 200-600 Hz; if the mechanical frequency response of the motor and load is much less than this frequency, the load cannot respond to this error.</p>
Accuracy of the Follower Motor and Drive	<p>The precision also depends on how accurately the drive follows its commanded position while moving. Even if master measurement were perfect, if the drive accuracy is poor, the precision will be poor.</p> <p>In the case of stepper drives, this amounts to the specified motor/drive accuracy.</p> <p>In the case of servo drives, the better the drive is tuned for smoothness and zero Following error, the better the precision of the positioning. Often, this really only matters for a specific portion of the profile, so the drive should be tuned for zero Following error at that portion.</p>
Accuracy of the Load Mechanics	The accuracy (not repeatability) of the load mechanics must be added to the overall build up of accuracy error. This includes backlash for applications that involve motion in both directions.
Master Position Prediction	<p>The master position prediction mode can be enabled or disabled with the <code>FPPEN</code> command, but each state contributes a different error.</p> <p>Disabled (<code>FPPEN0</code>): The follower position command is based on a master position that is 2 sample periods (4 ms) old. This means that master measurement error due to disabling the position prediction mode will be (2 sample periods * master speed).</p> <p>Enabled (<code>FPPEN1</code>): If the position prediction mode is enabled (default setting), its accuracy is also affected by position sampling accuracy, master speed, and master position filtering. The error due to enabling the position prediction mode is about twice that due to sampling accuracy (i.e., 40 to 200 microseconds * master speed). As with the error due to position sampling accuracy, the error due to the position prediction mode being enabled is like a noise on the order of 200-600 Hz, which is not noticed by large loads.</p>

## Master Velocity Relative to Master Position Prediction

*Variation in Master Velocity:* Although increasing master position filtering (increasing the `FFILT` command value) eliminates the error due to sampling accuracy, it increases the error due to variations in master speed when the master position prediction mode is enabled (`FPPEN1`).

Most applications maintain a constant master speed, or change very slowly, so this effect is minimal. But if the master is changing rapidly, there can be a significant master speed measurement error. Because predicted master positions are in part based on master speed measurement, they can result in an error in master position prediction mode (`FPPEN1`). This effect will always be smaller than that due to the master position prediction mode being disabled (`FPPEN0`).

## Tuning (Servo Axes Only)

A servo system's tuning has a direct impact on how well the follower axis can track the master input. Overshoot, lag, oscillation, etc., can be devastating to following performance. The best tool to use for tuning the 6K series controller is Motion Planner's Servo Tuner utility.

## Repeatability of the Trigger Inputs and Sensors

Some applications can use the trigger inputs for functions like registration moves, `GOWHENS`, or new cycles. For these applications, the repeatability of the trigger inputs and sensors add to the overall position error.

Refer to page 84 for accuracy specifications on the trigger input position capture function. Refer to your sensor manufacturer's documentation for the sensor repeatability.

## Preset vs. Continuous Following Moves

When a follower performs a preset (`MC0`) move in Following mode (`FOLEN1`), the commanded position is either incremental or absolute in nature, but it does have a commanded endpoint. The direction traveled by the follower will be determined by the commanded endpoint position, and the direction the master is counting.

Let's illustrate this with an example. Assume all necessary set-up commands have been previously issued for our follower (axis 1) and master so that distances specified are in revolutions:

```
FOLRN3      ; Set Following follower-to-master ratio numerator to 3
FOLRD4      ; Set Following follower-to-master ratio denominator to 4
              ; (ratio is 3 revs on the follower to 4 revs on the master)
FOLEN1      ; Enable Following on axis 1
FOLMD10     ; Set preset move to take place over 10 master revolutions
MC0         ; Set follower to preset positioning mode
MA1         ; Set follower to absolute positioning mode
PSET0       ; Set current absolute position reference to zero
D5          ; Set move distance to absolute position 5 revolutions
GO1         ; Initiate move to absolute position 5
```

If the master is stationary when the `GO1` command is executed, the follower will remain stationary also. If the master begins to move and master pulses are positive in direction, the follower will begin the preset move in the positive direction. If the master pulses stop arriving before 10 master revolutions have been traveled, the follower will also stop moving, but that `GO1` command will not be completed. If the master then starts to count in the negative direction, the follower will follow in the negative direction, but only as far as its starting position. If the master continues to count negative, the follower will remain stationary. The `GO1` command will not actually be completed until the master has traveled at least 10 revolutions in the positive direction from where it was at the time the `GO1` command was executed. If the master oscillates back and forth between its position at the start of the `GO1` command to just under 10 revolutions, the follower will oscillate back and forth as well.

The master must be counting in the positive direction for any preset (`MC0`) `GO1`s commanded

on the follower to be completed. If mechanics of the system dictate that the count on the source of the master pulses is negative, a minus (-) sign should be entered in the FOLMAS command so that 6K controller sees the master counts as positive.

Continuous follower moves (MC1) react much differently to master pulse direction. Whereas a preset move will only start the profile if the master counts are counting in the positive direction, a continuous move will begin the ramp to its new ratio following the master in either direction. As long as the master is counting in the positive direction, the direction towards which the follower starts in a continuous move is determined by the argument (sign) of the D command. The follower direction is positive for D+ and negative for D-.

If the master is counting in the negative direction when follower begins a continuous move, the direction towards which the follower moves is opposite to that commanded with the D command. The follower direction is positive for D- and negative for D+.

If the master changes direction during a continuous follower move, the follower will also reverse direction. As with standard continuous moves, the GO1 will continue until terminated by S, K, end-of-travel limits, stall condition, or command to go to zero velocity. As with preset moves, the sign on the FOLMAS command determines the direction of master counting with respect to the direction of actual counting on the master input.

## Master and Follower Distance Calculations

The formulas below show the relationship between master move distances and the corresponding follower move distances. These relationships can be used to assist in the design of Following mode moves in which both the position and duration of constant ratio are important.

In such calculations, it is helpful to use SCLMAS and SCLD values that allow the master and follower distances to be expressed in the same units (e.g., inches or millimeters). In this case, many applications will be designed to reach a final ratio of 1:1, and the distances in these figures can be easily calculated.

⇒ **HINT:** For a trapezoidal preset follower axis move with a maximum ratio of 1:1, the master and follower distances during the constant ratio portion will be the same. The follower travel during acceleration will be exactly half of the corresponding master travel, and it will also occur during deceleration.

### Master Distance (FOLMD)

If the follower axis is in continuous positioning mode (MC1), FOLMD is the master distance over which the follower axis is to accel or decel from the current ratio to the new ratio. If the follower axis is in preset positioning mode (MC0), FOLMD is the master distance over which the follower's entire move will take place.

### When the Follower is in Continuous Positioning Mode (MC1)

$$D = \frac{\text{FOLMD} * (R_2 + R_1)}{2}$$

where:

FOLMD = Master distance

R<sub>2</sub> = New ratio (FOLRN ÷ FOLRD)

R<sub>1</sub> = Current ratio (FOLRN ÷ FOLRD)

D = Follower distance traveled during ramp

When the Follower is in Preset Positioning Mode (MCØ)

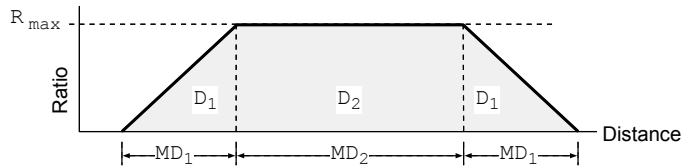
**Trapezoidal Follower Moves**

$D_{max} = (FOLMD * R_{max})$ $D_1 = \frac{(D_{max} - D)}{2}$ $MD_1 = \frac{2 * D_1}{R_{max}}$ $D_2 = D - (2 * D_1)$ $MD_2 = \frac{D_2}{R_{max}}$ $FOLMD = 2 * MD_1 + MD_2$	<p>where:</p> <p>FOLMD = Master distance</p> <p>MD<sub>1</sub> = Master distance during accel &amp; decel ramps</p> <p>MD<sub>2</sub> = Master distance during constant ratio</p> <p>D = Total follower axis preset distance commanded</p> <p>D<sub>1</sub> = Follower travel during accel and decel ramps</p> <p>D<sub>2</sub> = Follower travel during constant ratio</p> <p>D<sub>max</sub> = Maximum follower distance possible (assuming no accel or decel)</p> <p>R<sub>max</sub> = Maximum ratio = FOLRN ÷ FOLRD</p>
---	---

**Trapezoidal profile if:**

$$D = (D_2 + 2 * D_1) < D_{max}$$

These Profiles depict ratio vs. distance



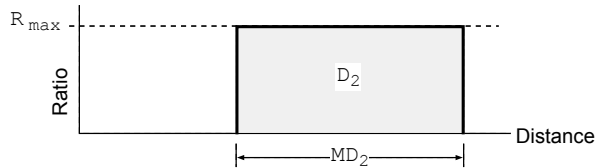
**Rectangular profile if:**

$$D_2 = D$$

$$D_1 = \text{zero}$$

$$MD_2 = FOLMD$$

$$MD_1 = \text{zero}$$



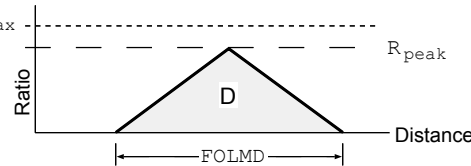
**Triangular Follower Moves**

$D = \frac{R_{peak} * FOLMD}{2}$	<p>where:</p> <p>FOLMD = Master distance</p> <p>R<sub>peak</sub> = Peak ratio reached during move</p> <p>D = Total follower distance</p>
----------------------------------	--

**Triangular profile if:**

$$D < 1/2 D_{max}$$

This Profile depicts ratio vs. distance



**Distance Calculation Example**

In the example below, the desired travel during constant ratio is already contained in numeric variable 1 (VAR1), and can have been read from thumbwheels or a DATA command. The corresponding follower move distance (D) and FOLMD are calculated as shown:

```

VAR2=2 ; Desired follower travel during accel and decel combined
VAR3=2 * VAR2 ; Required master travel during these ramps
VAR4=VAR1 + VAR2 ; Move distance is constant ratio portion plus ramps
VAR5=VAR1 + VAR3 ; Master travel for entire follower move
FOLMD (VAR5) ; Establish calculated master travel
D (VAR4) ; Establish calculated follower travel
GO1 ; Make desired follower move
    
```

Similar calculations can be done for a series of continuous move ramps to ratios, separated by GOWHEN for master cycle positions. These ramps can be repeated in a loop to create a continuous cyclical follower profile.

**Beware of Roundoff Error (Scaling only)**

Some potential for roundoff error exists if the scaling of a move distance or master distance by `SCLD` and `SCLMAS` does not result in an integer number of steps. Some additional care must be taken in the segment by segment construction of profiles using ramps to continuous ratio. The 6K controller maintains a follower position command that is calculated from the commanded constant ratios, the ramps to the new ratios, and the master travel over which these take place. At the end of each ramp or constant ratio portion, this commanded position is calculated to the nearest integer follower step. If the ratios and master travel result in a non-integer follower travel for a segment, the fractional part of that segment's calculated travel will be lost. In a cyclical application, repeated truncations could build up to a significant error. This can be avoided through careful attention to design of the profile.

## Using Other Features with Following

The 6K controller has many features that can be used in the same application as its Following features. In some cases, having configured an axis as a follower with the `FOLMAS` command will affect, or be affected by, the operation of other features, as described below.

**Setups used by `FOLMAS` (Stepper Axes Only)**

The `FOLMAS` command uses the drive resolution (`DRES`) and the velocity range (`PULSE`) data to configure an axis as a follower. In order for this data to be used correctly, these commands must be given before `FOLMAS`. These commands are not allowed after `FOLMAS` is given. After `FOLMAS` is executed, the `MC` command can be used to change from incremental to absolute positioning and back.

**S (Stop) & K (Kill) Commands**

Stop (`S`) and Kill (`K`) commands cause the follower to do a non-Following decel, even if the follower is in Following mode. A stop or a kill, buffered or immediate, will clear a pending `GOWHEN` condition (clears `AS` bit 26).

**Kill Conditions**

Under default operation (`FOLK0`), certain error conditions (i.e., drive fault input active, or maximum position error limit exceeded) will cause the 6K controller to disable the drive and kill the Following profile (follower's commanded position loses synchronization with the master).

If you enable Following Kill (`FOLK1`), these error conditions will still disable the drive (`DRIVE0`), but will not kill the Following profile. Because the Following profile is still running, the controller keeps track of what the follower's position should be in the Following trajectory. To resume Following operation, resolve the error condition (drive fault, excessive position error), enable the drive (`DRIVE1`), and command the controller to impose a shift to compensate for the lapse/shift that occurred while the drive was disabled and the follower was not moving. To impose the shift, assign the negative of the internally monitored shift value (`PSHF`) to a variable (e.g., `VAR1 = -1 * PSHF`) and command the shift using a variable substitution in the `FSHFD` command (e.g., `FSHFD (VAR1)`).

The `FOLK` command only preserves Following profiles; normal velocity-based profiles will be killed regardless of the `FOLK` command.

**GO Command**

If a follower axis is in Following mode (`FOLEN1`), moves will ramp to a ratio (set with `FOLRN` and `FOLRD`). If it is not in Following mode, moves will ramp to a velocity (`V`). Switching in and out of Following mode does not change the value for final ratio or final velocity goals, but simply changes which parameter is used as the goal.

**Registration Moves (see page 155)**

Registration inputs can be enabled with the `INFCi-H` command while an axis is a follower, and registration moves can interrupt either a Following mode move or a time-based move. The registration move itself, however, is always a time based move, and implements the registration velocity with respect to a stationary reference. Any trigger input can be used as a registration input or for any Following feature that uses triggers, even at the same time.

## Enter/Exit Following Mode While Moving

The `FOLENØ` command can be executed while a follower axis is moving at constant ratio. In this case, the current velocity becomes the constant velocity, and the follower can accelerate or decelerate to other velocities. The `FOLEN1` command cannot be executed while the follower axis is moving in the non-Following mode. Attempting to do so will result in the error response “MOTION IN PROGRESS”.

## Pause and Continue

A program can be paused and continued, even if one or more axes is configured as a follower axis. Those axes do not lose track of the master input, even though motion is stopped. As usual, if a program finishes normally, or if `COMEXS` is set to zero (`COMEXSØ`), the program cannot be continued. If a program is resumed, partially completed Following moves will be completed; however, the remainder of the move is completed over the entire original master distance (`FOLMD` value).

## `TVEL` and `TVELA` Commands

The `TVEL` and `VEL` commands have always reported an unsigned value (i.e. magnitude), consistent with the fact that the `V` command is always positive, even if the move direction is negative. When Following is enabled (`FOLEN1`), `TVEL` and `VEL` report the net magnitude of commanded velocity due to following and/or shifting. For example, if the follower axis is Following in the positive direction at 1 rps, `TVEL` reports 1.000. If a shift in the negative direction of 1.5 rps is then commanded, `TVEL` will report 0.5 – still positive, even though the net direction is negative.

By contrast, `TVELA` and `VELA` have always been signed. In the above example, `TVELA` will report -0.5.

## `TASF`, `TAS` & `AS` Axis Status Bits

Axis Status (`TASF`, `TAS` and `AS`) bits 1-4 represent the motion status of an axis. Bits 1 and 2 represent the moving and direction status of the net motion of an axis, including the combination of following and shifting.

Bits 3 and 4 represent acceleration and velocity, respectively. With Following disabled (`FOLENØ`), the accel and velocity can only be due to a commanded time-based profile. With Following enabled (`FOLEN1`), the net commanded velocity and accel could be due to following the master and/or a simultaneous shift (time-based profile). Bits 3 and 4 only reflect the acceleration and velocity status of the time-based profile, not the combined command.

For example, if the follower axis is Following in the positive direction at 1 rps, `TAS` reports 1ØØØ. If a continuous shift (`FSHFC2`) of 0.8 rps in the negative direction is then commanded, `TAS` will report 1Ø1Ø, while the shaft is decelerating to 0.2 rps. When the continuous shift velocity is reached, `TAS` will report 1ØØ1.

## Changing Feedback Sources (Servo Axes only)

Changing feedback sources (`SFB`) can result in a follower axis Following its own feedback. For this reason, changing feedback sources is not allowed while a master is specified (`FOLMAS`).

## Following and Other Motion

Following motion is initiated with the `GO` command, just like normal motion. Other motion, that does not depend on the motion of an external master, cannot be used while a follower axis is in Following mode (`FOLEN1`). These motion types include jog mode, joystick mode, contouring, homing, streaming mode, and linear interpolation (`GOL`). To use these motion types, Following must be disabled (`FOLENØ`) — see *Enter/Exit Following Mode While Moving* above for precautions.

## Conditional Statements Using `PMAS`

The master cycle position (`PMAS`) value can be used in the comparison argument of these commands:

- `WAIT & GOWHEN`: If it is desired to `WAIT` or `GOWHEN` on a master cycle position of the next master cycle, one master cycle length (value of `FMCLEN`) should be added to the master cycle position specified in the argument. This allows commands that sequence

follower axis events through a master cycle to be placed in a loop. The `WAIT` or `GOWHEN` command at the top of the loop could execute, even though the actual master travel had not finished the previous cycle. This is done to allow a `PMAS` value that is equal to the master cycle length to be specified and reliably detected.

- `IF`, `UNTIL`, & `WHILE`: These arguments use the instantaneous `PMAS` value. Be careful to avoid specifying `PMAS` values that are nearly equal to the master cycle length (`FMCLEN`), because rollover can occur before a `PMAS` sample is read.

**Compiled Motion**      Following profiles can be pre-compiled to save processing time. For details, see page 139.

## Troubleshooting for Following *(also see Chapter 8)*

The table below offers some possible reasons for troubles that can be encountered in achieving the desired follower motion.

Symptom	Possible Causes
Follower axes do not follow master	<ul style="list-style-type: none"> <li>• Improper <code>FOLMAS</code></li> <li>• Poor connection if master is encoder</li> <li>• Master running backward</li> <li>• No encoder power (when the encoder is selected as the master)</li> </ul>
Follower motion is rough	<ul style="list-style-type: none"> <li>• <code>FFILT</code> command value too low</li> <li>• Unnecessary <code>FPPEN</code> amplifies master roughness</li> </ul>
Ratio seems wrong	<ul style="list-style-type: none"> <li>• <code>FOLRN</code> and <code>FOLRD</code> follower-to-master ratio values are inaccurate, possibly reversed</li> <li>• <code>SCLD</code> or <code>SCLMAS</code> wrong</li> <li>• Following limited by <code>FMAXV</code> or <code>FMAXA</code> (steppers only)</li> </ul>
Follower profile wrong, or un-repeatable	<ul style="list-style-type: none"> <li>• <code>WAIT</code> used where <code>GOWHEN</code> should be</li> <li>• Too little master travel between <code>GOWHEN</code> and <code>GO1</code>, desired <code>PMAS</code> is missed</li> </ul>
Master/follower alignment drifts over many cycles	<ul style="list-style-type: none"> <li>• Roundoff error due to fractional steps resulting from <code>SCLD</code> or <code>SCLMAS</code> and user's parameters</li> <li>• Ratios and master distances specified result in fractional follower steps covered during ramps, constant ratio</li> </ul>
Follower lags Following position (steppers only)	<ul style="list-style-type: none"> <li>• Inhibited by <code>FMAXV</code></li> <li>• <code>FMAXA</code> clips acceleration peaks resulting from attempt to follow rough master</li> </ul>

## Error Messages

If an illegal programming condition is discovered while programming or executing programs, the 6K controller responds with an error message. If a program execution error is detected, the program is aborted.

The table below lists all the error messages that relate to Following, and indicates the command and cause that can generate them. These error messages are displayed only if the error level is set to level 4 with the `ERRLVL4` command (this is the default setting).

Error Message	Cause
FOLMAS NOT SPECIFIED	No FOLMAS for the axis is currently specified. It will occur if FMCNEW, FSHFC, or FSHFD commands are executed and no FOLMAS command was executed, or FOLMASØ was executed.
INCORRECT DATA	Velocity (V), acceleration (A), or deceleration (AD) command is zero. (used by FSHFC & FSHFD)
INVALID CONDITIONS FOR COMMAND	<p>The FOLMD command value is too small to achieve the preset distance and still remain within the FOLRN/FOLRD ratio.</p> <p>A command phase shift cannot be performed:</p> <p>FSHFD.....Error if already shifting or performing other time based move.  FSHFC.....Error if currently executing a FSHFD move, or if currently executing another FSHFC move in the opposite direction.</p> <p>The FOLEN1 command was given while a profile was suspended by a GOWHEN.</p>
INVALID DATA	<p>The parameter supplied with the command is invalid.</p> <p>FFILT.....Error if: smooth number is not 0-4  FMCLLEN.....Error if: master steps &gt; 999999999 or negative  EMCP .....Error if: master steps &gt; 999999999 or &lt;-999999999  FOLMD.....Error if: master steps &gt; 999999999 or negative  FOLRD.....Error if: master steps &gt; 999999999 or negative  FOLRN.....Error if: follower steps&gt;999999999 or negative  FSHFC.....Error if: number is not 0-3  FSHFD.....Error if: follower steps&gt;999999999 or &lt;-999999999  GOWHEN.....Error if: position &gt; 999999999 or &lt;-999999999  WAIT.....Error if: position &gt; 999999999 or &lt;-999999999</p> <p>Error if a GO command is given in the preset positioning mode (MCØ) and:</p> <p>FOLRN = zero  FOLMD = zero, or too small (see <i>Preset Positioning Mode Moves</i> on page 175)</p>
INVALID FOLMAS SPECIFIED	An illegal master was specified in FOLMAS. A follower can never use its own commanded position or feedback source as its master.
INVALID RATIO	Error if the FOLRN:FOLRD ratio after scaling is > 127 when a GO is executed.
MASTER SLAVE DISTANCE MISMATCH	Attempting a preset Following move with a FOLMD value that is too small.
MOTION IN PROGRESS	The FOLEN1 command was given while that follower was moving in a non-Following mode.
NOT VALID DURING FOLLOWING MOTION	A GO command was given while moving in the Following mode (FOLEN1) and while in the preset positioning mode (MCØ).
NOT VALID DURING RAMP	A GO command was given while moving in a Following ramp and while in the continuous positioning mode (MC1). Following status (FS) bit 3 will be set to 1.
SYSTEM UPDATE OVERRUN, USE SYSPER4	The system update service has overrun the time allotted with the default 2-millisecond period. Use the SYSPER4 command to increase the system update period to 4 milliseconds.

## Following Commands

Detailed information about these commands is provided in the *6K Series Command Reference*.

- ANIMAS ..... Assigns an analog input to be used as a master in a FOLMAS assignment (requires a ANI SIM located on an expansion I/O brick).
- ERROR..... Enable bit 14 to check for when a GOWHEN condition is already true when a subsequent GO, GOL, FGADV, FSHFC, or FSHFD command is given.
- ERRORP ..... If ERROR bit 14 is enabled, a GOSUB branch to the error program occurs if a GOWHEN condition is already true

when a subsequent GO, GOL, FGADV, FSHFC, or FSHFD command is given.

FFILT..... Sets the bandwidth for master position filtering.

FGADV..... Defines the geared advance distance.

FMAXA..... Stepper axes only: Sets the maximum acceleration a follower can use while Following.

FMAXV..... Stepper axes only: Sets the maximum velocity at which a follower can travel.

FMCLEN..... Defines the length of the master cycle.

EMCNEW..... Restarts new master cycle counting immediately. To make this a trigger-based operation instead of issuing the EMCNEW command, use the TRGFNCx1 command.

FMCP..... Defines the initial position of a new master cycle.

FOLEN..... Enables or disables Following mode.

FOLK..... Allows drive fault or maximum position error to disable drive without killing the Following profile.

FOLMAS..... Defines masters for follower axes.

FOLMD..... Defines the master distance over which ratio changes or moves are to take place.

FOLRD..... Establishes the DENOMINATOR ONLY for the maximum follower-to-master ratio for a preset move or the final ratio for a continuous move (use in combination with the FOLRN command).

FOLRN..... Establishes the NUMERATOR ONLY for the maximum follower-to-master ratio for a preset move or the final ratio for a continuous move (use in combination with the FOLRD command).

FPPEN..... Allows master position prediction to be enabled or disabled.

FSHFC..... Allows continuous advance or retard (shift) of follower position during continuous Following moves.

FSHFD..... Allows preset advance or retard (shift) of follower position during continuous Following moves.

FVMACC..... Establishes the rate at which the virtual master count frequency (FVMFRQ) can change for an axis.

FVMFRQ..... Defines the frequency of the virtual master count.

GOWHEN..... A GOWHEN command suspends execution of the next follower move until the specified conditional statement (based on T, IN, LIM, PC, PE, PMAS, PSLV, or PSHF) is true. To make this a trigger-based operation instead of issuing the GOWHEN command, use the TRGFNC1x command.

SCLD..... Sets the follower distance scale factor.

SCLMAS..... Sets the master scale factor.

SINAMP..... Defines the amplitude of the internal sine wave.

SINANG..... Defines the phase angle of the internal sine wave.

SINGO..... Initiates the internal sine wave.

TRGFN..... aTRGFNC1x initiates a GOWHEN. Suspends execution of the next follower move on follower axis (a) until the specified trigger input (c) goes active.  
aTRGFNCx1 allows master cycle counting to be restarted on axis (a) when the specified trigger input (c) goes active.

### Status and Assignment Commands

TASF, TAS & [ AS ] ..... Bit 26 of each axis status is set (1) if a motion profile suspended by a GOWHEN (including TRGFNC1x) is pending on that axis. The bit is cleared when the GOWHEN is true, or when a stop (S) or kill (K) command is executed.

TERF, TER & [ ER ] ..... Bit 14 is set if the GOWHEN condition is already true when a subsequent GO, GOL, FGADV, FSHFC, or FSHFD command is given. (The corresponding error-checking bits must be enabled with the ERROR command before the error will be detectable.)

TFSF, TFS & [ FS ] ..... Transfers or assigns/compares the Following status or each axis.

TNMCY and [ NMCY ] ..... Transfers or assigns the current master cycle number.

TPCMS and [ PCMS ] ..... Transfers or assigns the current captured master cycle position.

TPMAS and [ PMAS ] ..... Transfers or assigns the current master cycle position.

TPSHF and [ PSHF ] ..... Transfers or assigns the net position shift since constant ratio.

TPSLV and [ PSLV ] ..... Transfers or assigns the follower's current commanded position.

TVMAS and [ VMAS ] ..... Transfers or assigns the current velocity of the master axis.

WAIT..... A WAIT command suspends program execution until the specified conditional statement (based on PMAS, FS, NMCY, PCMS, PSHF, PSLV, or VMAS) is true;  
WAIT(SS.i=b1) suspends program execution until a trigger input is activated ("i" is the input bit number corresponding to the trigger input—see the 6K Series Command Reference).