# **Motor Control Using PMAC**

Use NSLS-II default configure file GB\_Startup.pmc which can be downloaded at

<https://wiki.bnl.gov/nsls2controls/index.php/Delta_Tau_PMAC>

PMAC variables:

* I: motor/controller variables
* M: memory in controller
* P/V: user variables

## Preparation

* 1. Commutation enable/disable

|  |  |
| --- | --- |
| **Variable** | **Explanation** |
| Ixx01 | Check/set limit switch.  0 ; disable communication  1 ; enable commutation |

* 1. Limit switch check

|  |  |
| --- | --- |
| **Variable** | **Explanation** |
| Flag Modes  ixx24 | Check/set limit switch.  $800401 ; using limit switch  $820001 ; limit switch not used |

## Move the motor

* 1. Set magnetization current

|  |  |
| --- | --- |
| **Variable** | **Explanation** |
| Magnetization current  ixx77 | Set current. Usually 2A for stepper motors. |

* 1. Set close-loop

|  |  |
| --- | --- |
| **Variable** | **Explanation** |
| Position-Loop Feedback Address  ixx03 | Enable/disable close-loop.  $3501 ; Conversion Table Line 0  $3501 ; Conversion Table Line 1 |
| Velocity-Loop Feedback Address  ixx04 |  |

* 1. Set motor direction

|  |  |
| --- | --- |
| **Command** | **Explanation** |
| Number of commutation cycles (N)  ixx70 | 1/-1 to change direction.  Direction of motor (pos/neg) is decided by pos/neg limit. |

* 1. Move

|  |  |
| --- | --- |
| **Command** | **Explanation** |
| #1o.1 | Move motor 1 in positive direction using 0.1% current in open-loop mode. |
| #1o-.1 | Move motor 1 in negative direction using 0.1% current in open-loop mode. |

## Set encoder direction

|  |  |
| --- | --- |
| **Variable** | **Explanation** |
| Encoder decode control  i7mn0  i7mn0=7 | Set the direction of the encoder n on Servo IC m. Incorrectly set encoder direction can result in following error.  3: clocl wise(4 counts per encoder cycle)  7: conter-clock wise (4 counts per encoder cycle)   |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | | Motor | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | | mn | 01 | 02 | 03 | 04 | 11 | 12 | 13 | 14 | |

## Set motor direction

If the motor rotate in opposite direction of the given command, change ixx70.

|  |  |
| --- | --- |
| **Variable** | **Explanation** |
| |  | | --- | | Commutation Cycle Size |   ixx70 | Set Ixx70 equal to the number of pole pairs of the motor  Set to a negative value to change the direction of motor rotation. |

## Homing

|  |  |
| --- | --- |
| **Variable** | **Explanation** |
| Motor Flag Mode Control  Ixx24 |  |
| Motor Position Capture & Trigger Mode  Ixx97 | 0/2: Use hardware captured position as the trigger position  1/3: Use software captured position for the trigger position  0/1: Use input capture trigger flag as the trigger  2/3: Use the warning following error status bit as the trigger |
| Motor Flag Address  Ixx25 |  |
| I7mn2 |  |
| I7mn3 | * 0: HOMEn (Home Flag n) * 1: PLIMn (Positive End Limit Flag n) * 2: NLIMn (Negative End Limit Flag n) * 3: USERn (User Flag n) |
|  |  |

## PLC

Foreground PLCs: PLC 0 and PLCC 00

Background PLCs: PLC 1-31 and PLCC 1-31

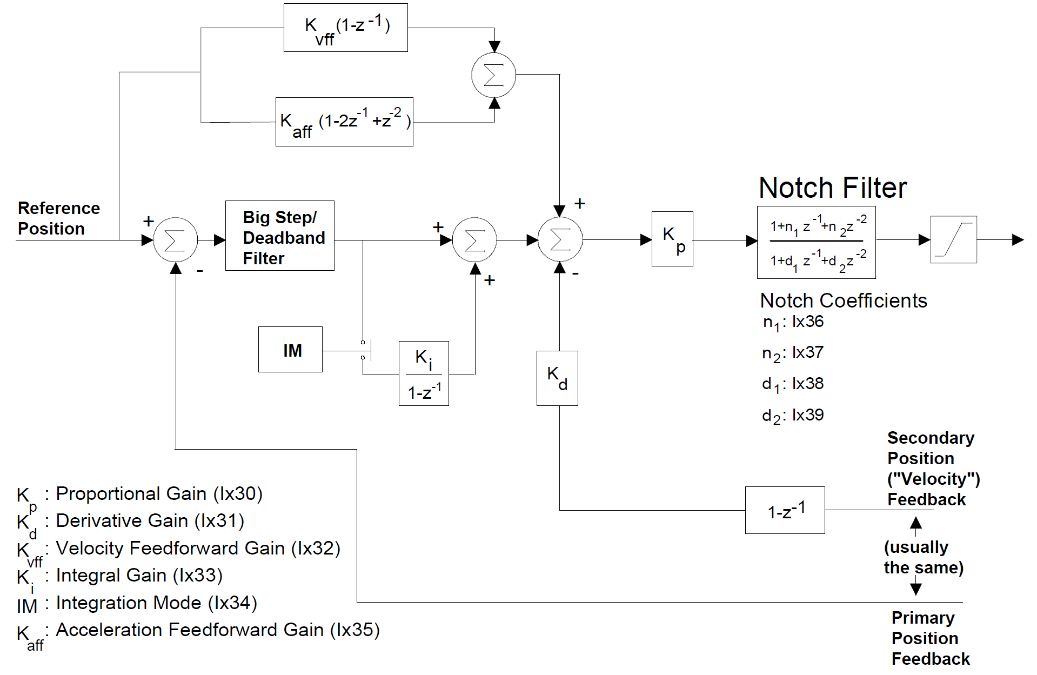
|  |  |
| --- | --- |
| **Variable** | **Value** |
| PLC program control  I5 | 0: Foreground PLCs off, background PLCs off  1: Foreground PLCs on, background PLCs off  2: Foreground PLCs off, background PLCs on  3: Foreground PLCs on, background PLCs on |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

## Motor training

* 1. Parameters

|  |  |
| --- | --- |
| **Variable** | **PID Parameter** |
| ixx30 | Proportional Gain (Kp) |
| ixx31 | Derivative Gain (Kd) |
| ixx32 | Velocity Feedforward (Kvff) |
| ixx33 | Integral Gain (Ki) |
| ixx34 | Integration Mode |
| ixx35 | Acceleration Feedforward (Kaff) |
| ixx68 | Friction Feedforward (Kfff) |

* 1. PMAC PID + Low-pass Notch filter



* 1. Current loop
  2. Position loop
  3. Limit check

Command: #1j+ ; jog in positive direction

#1j- ; jog in negative direction

* 1. Find home (index pulse)

Command: #1j^100000000 ; jog in positive direction until index pulse is found or positive limit reached

#1j^-100000000 ; jog in negative direction until index pulse is found or positive limit reached

* 1. Set current position as home (zero)

Command: #1homez

This is usually used if index pulse is not found

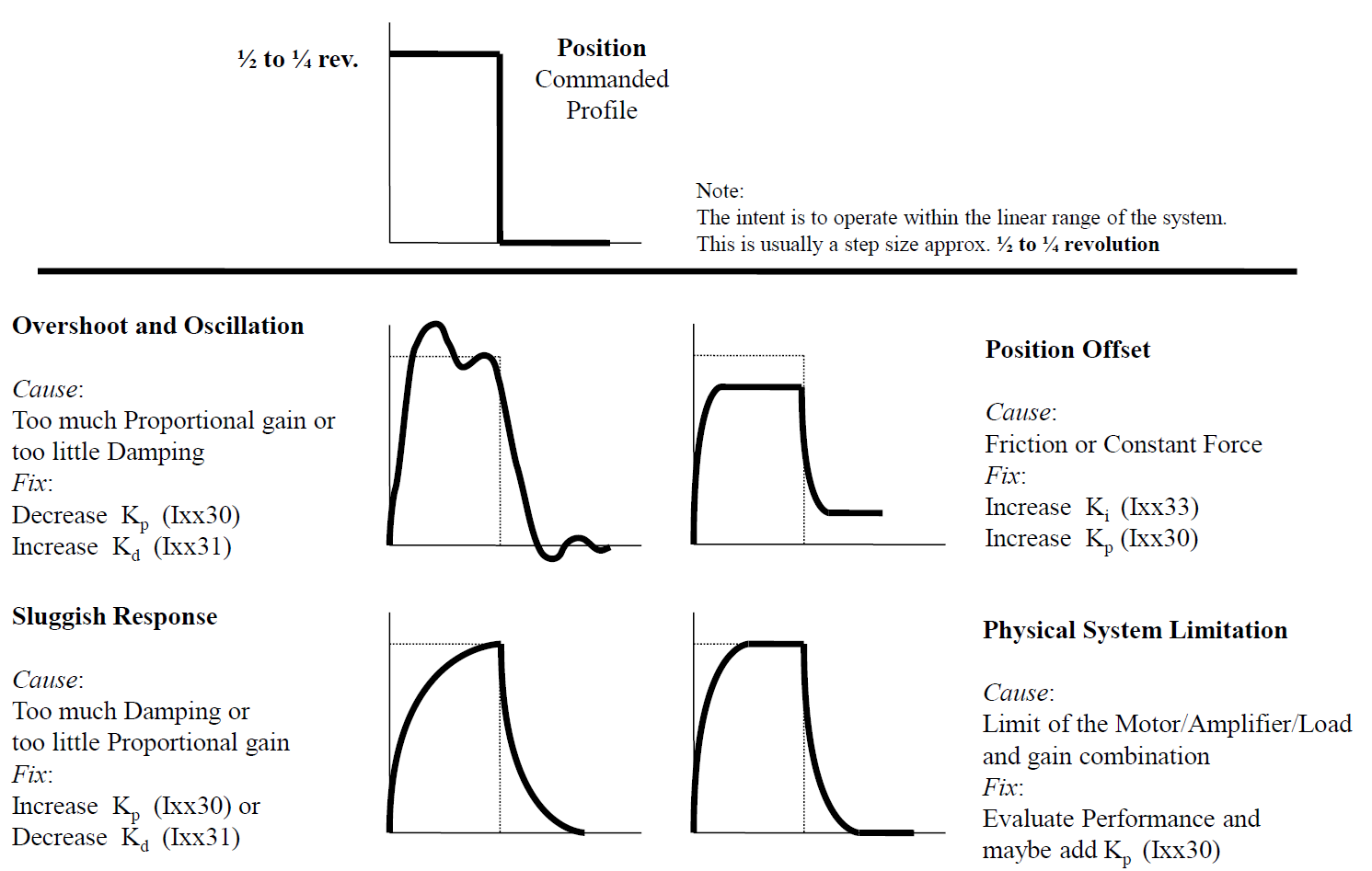
## Delta-Tau Suggested Steps for Tuning

* 1. Perform the DAC Calibration as described in the Motor Setup section
  2. Set Ixx34 (Motor xx PID Integration Mode) – can be changed on the fly as needed

=1, position error integration is performed only when Motor xx is not commanding a move (when desired velocity is zero)

=0, position error integration is performed always

* 1. Using the Step Response, tune the following parameters in this order:
     1. Proportional Gain, Kp (Ixx30). Let Ki =0 and kd=0. Set input as 60%~70% of maximum allowed input, increase Kp from 0 until oscillating. Decrease Kp until oscillation disappears. Set Kp to 60%~70% of the current reading.
     2. Derivative Gain, Kd (Ixx31). Set Kd to a large value, decrease Kd until the system oscillates. Increase Kd until oscillation disappears. Set Kd to 150%~180% of current reading.
     3. Integral Gain, Ki (Ixx33). Set Ki to 0, or use the methodology for Kp and set it to 30% of the reading when oscillation disappears.



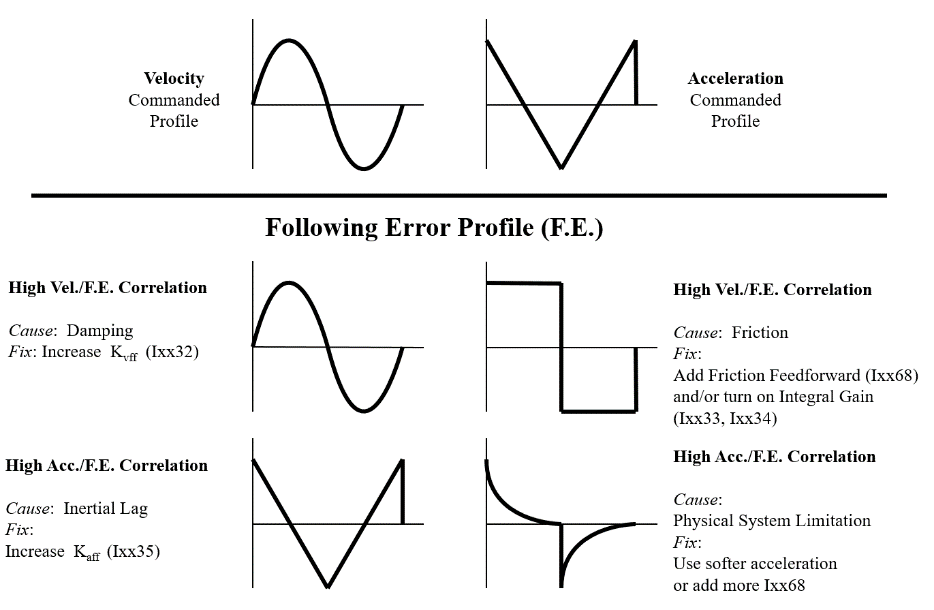
Step tunning (Ixx30, Ixx31, Ixx33)

* 1. Using the Parabolic Move, tune the following parameters, not necessarily in this order:

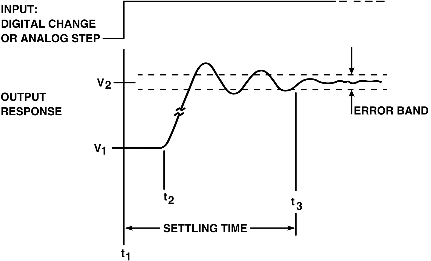
Velocity Feedforward, Kvff (Ixx32)

Acceleration Feedforward, Kaff (Ixx35)

Friction Feedforward, Kfff (Ixx68)

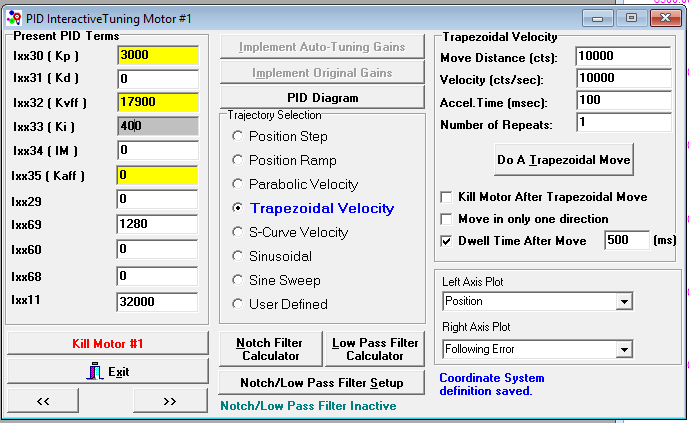


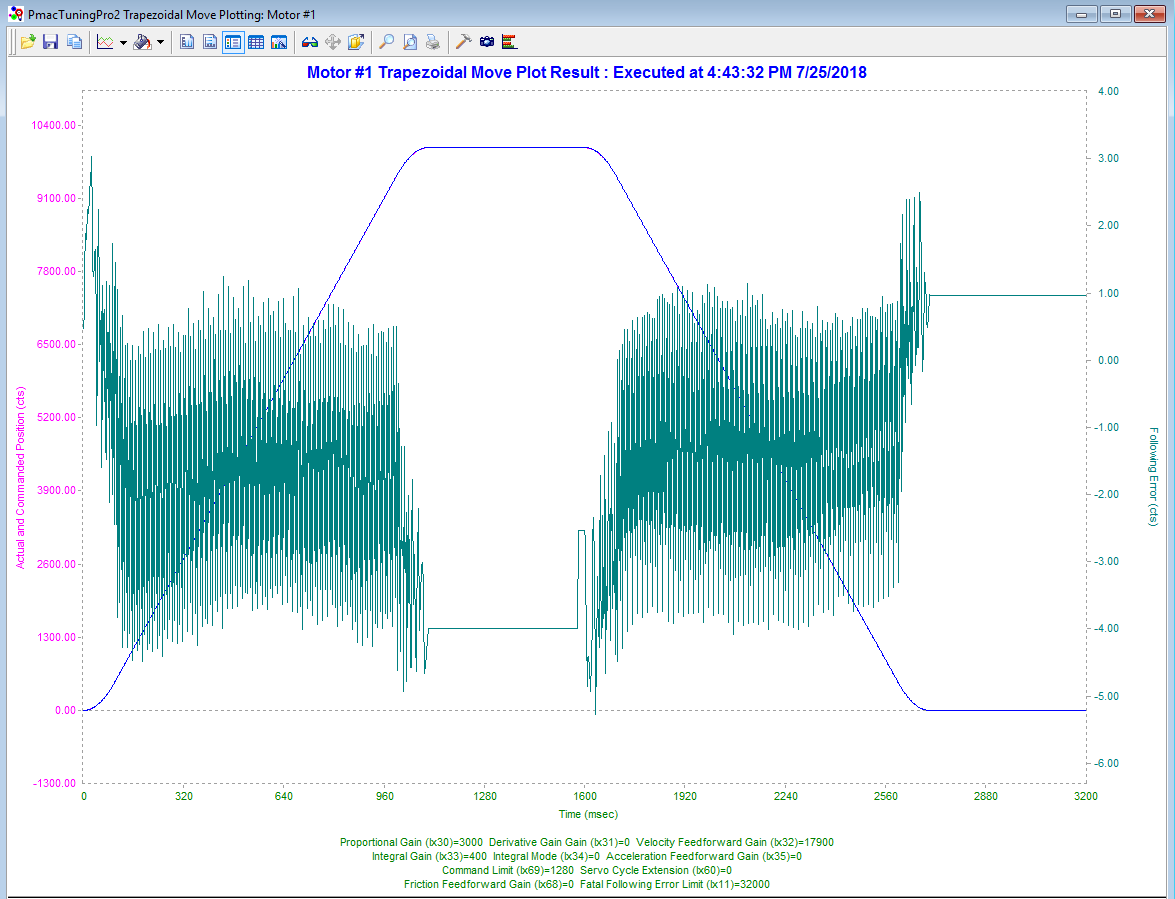
Parabolic tunning (Ixx32-Ixx35, Ixx68)

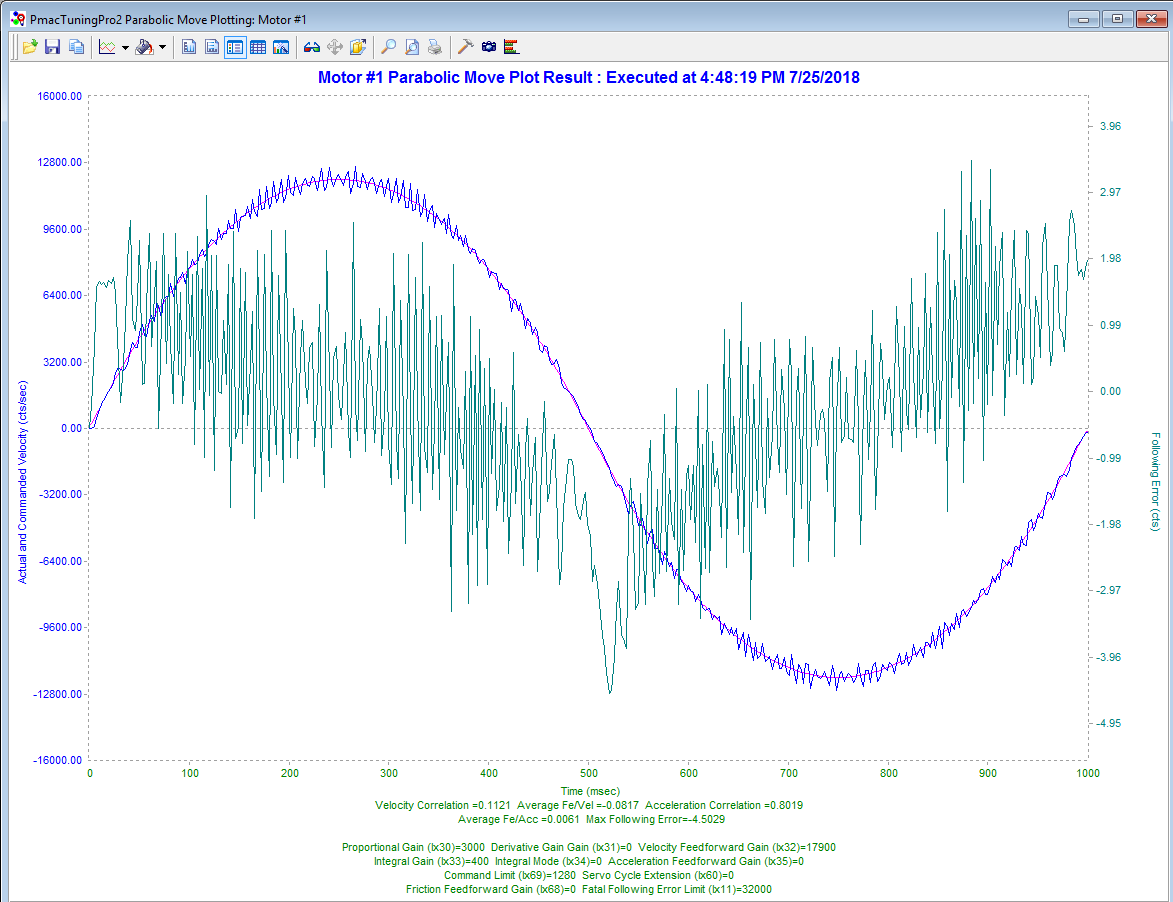


|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Parameter** | **Rise time** | **Overshoot** | **Settling time** | **Steady-state error** | **Stability** |
| **Kp** | Decrease | Increase | Small change | Decrease | Degrade |
| **Ki** | Decrease | Increase | Increase | Eliminate | Degrade |
| **Kd** | Minor change | Decrease | Decrease | No effect in theory | Improve if small |

For stepper motors at NSLS2, Kp, Kvff and Ki tuning could be enough.







## Homing

* 1. Homing/Jogging settings

I7mn2 determines which input signal or combination of signals for Channel n of a PMAC2-style Servo IC m, and which polarity, triggers a hardware position capture of the counter for Encoder n.

Ixx20 through Ixx23 are self-explanatory

|  |
| --- |
| // I7mn2 Servo IC m Channel n Capture Control  I7212,2,10 = 1 ; Motors 1 and 2 Capture on Index (CHCn) high  // Ixx20 Motor xx Jog/Home Acceleration Time  I120,2,100 = 0 ; Force Ixx21 to control jog accel. time  // Ixx21 Motor xx Jog/Home S-Curve Time  I121,2,100 = 25 ; msec  // Ixx22 Motor xx Jog Speed  I122,2,100 = 64 ; cts/msec  // Ixx23 Motor xx Home Speed and Direction  I123,2,100 = 64 ; cts/msec, sign is direction |

Ixx23 - Motor xx Home Speed and Direction [cts/msec]

Establishes the commanded speed and direction of a homing-search move for Motor xx.

Positive value – positive homing direction.

Negative value – negative homing direction.

* 1. Ixx26 - Motor xx Home Offset [1/16 ct]

Specifies the difference between the zero position of sensors for the motor and the motor’s own zero “home” position.

An easy way to set this value is to write a PLC to wait for the user to position the motor as desired, and then capture the home offset positions when the user sets a flag.

* 1. When using a PLC to home a motor, one should poll these three bits until they all become 1 before advancing:

Mxx33: Motor xx Desired-velocity-zero bit

Mxx40: Motor xx Background in-position bit

Mxx45: Motor xx Home-complete bit (use for homing only)

This example homes motor 1, waits for it to finish, and then jogs motor 1 to 2000 cts absolute :

|  |
| --- |
| M133->X:$0000B0,13,1 ; #1 Desired-velocity-zero bit  M140->Y:$0000C0,0,1 ; #1 Background in-position bit  M145->Y:$0000C0,10,1 ; #1 Home-complete bit  End Gat  Del Gat  Close  Open PLC 4 Clear  CMD"#1HM" ; Home the motor  I5111=1\*8388608/I10  While(I5111>0) EndW ; Force CMD execution before continuing  While(M133 = 0 or M140 = 0 or M145 = 0)EndWhile ; Poll status bits  CMD"#1J=2000" ; Jog to 2000 cts absolute  I5111=1\*8388608/I10  While(I5111>0) EndW ; Force CMD execution before continuing  While(M133 = 0 or M140 = 0)EndWhile ; Poll status bits  // Put subsequent code here  // that will execute only after the J=2000 move has completed  Disable PLC 4  Close |