# Coordinated movement in PMAC through EPICS

## **Overview**

Axes can only be specified by letters **[X, Y, Z], [U, V, W],** and **[A, B, C]**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Axis | A | B | C | U | V | W | X | Y | Z |
| Address | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| Usage | Rotary | Rotary | Rotary | 2nd linear | 2nd linear | 2nd linear | 1st linear | 1st linear | 1st linear |
|  | Roll |  | Pitch |  |  |  | Slits-Center | Slits-Opening |  |

## **Motion program**

|  |
| --- |
| CLOSE  ;---------------------------------------------------  ; Generic program for moving any co-ordinate system  ; Original Author: Tom Cobb  ; Used variables: Q71..Q79, Q81..Q89  ; You should do a $$$ or type the following the first  ; time you load this prog.  ; If you type the commands in, ignore any errors. If you  ; $$$, PLC1 will do it automatically for you at startup:  ;&16 DEFINE LOOKAHEAD 50,10  ;&15 DEFINE LOOKAHEAD 50,10  ;&14 DEFINE LOOKAHEAD 50,10  ;&13 DEFINE LOOKAHEAD 50,10  ;&12 DEFINE LOOKAHEAD 50,10  ;&11 DEFINE LOOKAHEAD 50,10  ;&10 DEFINE LOOKAHEAD 50,10  ;&9 DEFINE LOOKAHEAD 50,10  ;&8 DEFINE LOOKAHEAD 50,10  ;&7 DEFINE LOOKAHEAD 50,10  ;&6 DEFINE LOOKAHEAD 50,10  ;&5 DEFINE LOOKAHEAD 50,10  ;&4 DEFINE LOOKAHEAD 50,10  ;&3 DEFINE LOOKAHEAD 50,10  ;&2 DEFINE LOOKAHEAD 50,10  ;---------------------------------------------------  ; This program defines the program needed to do a move  ; using the motor record  ;on a CS axis  ;to do a move we need to do &{CS}R  ;to stop a move we need to do &{CS}A  ;demand values are in &{CS}Q71..79  ;The following are set by the motor record:  ;Isx87 = Acceleration  ;Isx88 = S curve  ;Isx89 = Feedrate  ; Set some defaults for all CS axes  i5213,15,100=50 ; segmentation time (needed for lookahead)  i5220,15,100=50 ; lookahead length (needed to limit max  ; velocity to max set in CS)  i5250,15,100=1 ; Enable kinematics  OPEN PROG 10  CLEAR  LINEAR  ABS  ; need to think about the consequences of including all axes  ; in frax  FRAX(A,B,C,U,V,W,X,Y,Z)  A(Q71)B(Q72)C(Q73)U(Q74)V(Q75)W(Q76)X(Q77)Y(Q78)Z(Q79)  DWELL0  CLOSE |

## **Coordinate system definition in PMAC controller**

### **Slits**

#### Kinematics

|  |
| --- |
| CLOSE  ;---------------------------------------------------  ;; \file  ;; Define motion for top/bottom blades  ;;  ;; Original Author: Tom Cobb  ;;  ;; Defined axes:  ;; - X (addr 6) = CENTER in same EGUs as blade  ;; - Y (addr 7) = gap in same EGUs as blade  ;;  ;; Macros:  ;; - COORD = $(COORD) CS number, e.g. 2  ;; - PLC = $(PLC) PLC number, should be CS  ;; number+15, e.g. 17  ;; - TOP = $(TOP) Axisnum for Blade+, e.g. 4  ;; - BOTTOM = $(BOTTOM) Axisnum for Blade-, e.g. 5  ;---------------------------------------------------  #define COORD 2  #define PLC 17  #define TOP 4 ; Top blade on Axis 4  #define BOTTOM 5 ; Bottom blade on Axis 5  ; Change to CS$(COORD)  &$(COORD)  ; Set relevant axes to use kinematics  #$(TOP)->I ; +ve blade  #$(BOTTOM)->I ; -ve blade  ;---------------------------------------------------  ; These are set by motor\_in\_cs.template  #define TOP\_MOVE P(4700+$(TOP))  #define TOP\_MRES P(4800+$(TOP))  #define TOP\_OFF P(4900+$(TOP))  #define BOTTOM\_MOVE P(4700+$(BOTTOM))  #define BOTTOM\_MRES P(4800+$(BOTTOM))  #define BOTTOM\_OFF P(4900+$(BOTTOM))  ;---------------------------------------------------  ; Calculate gap and CENTER from blade positions  #define TOP\_POS (TOP\_MRES\*P$(TOP)+TOP\_OFF)  #define BOTTOM\_POS (BOTTOM\_MRES\*P$(BOTTOM)+BOTTOM\_OFF)  #define CENTER Q7  #define GAP Q8  OPEN FORWARD  CLEAR  CENTER = (TOP\_POS + BOTTOM\_POS)/2  GAP = TOP\_POS - BOTTOM\_POS  CLOSE  ;---------------------------------------------------  ; Calculate blade positions in cts from gap and CENTER  #define TOP\_POS Q228  #define BOTTOM\_POS Q229  OPEN INVERSE  CLEAR  ; calculate in EGUs  TOP\_POS = CENTER + GAP/2  BOTTOM\_POS = CENTER - GAP/2  ; then in cts  P$(TOP) = (TOP\_POS - TOP\_OFF)/TOP\_MRES  P$(BOTTOM) = (BOTTOM\_POS - BOTTOM\_OFF)/BOTTOM\_MRES  CLOSE  ;---------------------------------------------------  ; A PLC(sx+15) needs to be made to do position reporting  ; Readbacks should be in &{axisnum}Q81..89  ; As forward kinematic, but with Px = mx62/(Ix08\*32)  #define TOP\_POS (TOP\_MRES\*m$(TOP)62/(I$(TOP)08\*32)+TOP\_OFF)  #define BOTTOM\_POS (BOTTOM\_MRES\*m$(BOTTOM)62/(I$(BOTTOM)08\*32)+BOTTOM\_OFF)  #define CENTER Q87  #define GAP Q88  OPEN PLC$(PLC)  CLEAR  ADDRESS&$(COORD)  CENTER = (TOP\_POS + BOTTOM\_POS)/2  GAP = TOP\_POS - BOTTOM\_POS  ; If blade motor record did the last move,  ; set demands = readbacks  if (TOP\_MOVE = 1)  or (BOTTOM\_MOVE = 1)  TOP\_MOVE = 0  BOTTOM\_MOVE = 0  Q77 = Q87  Q78 = Q88  endif  CLOSE  ENABLE PLC$(PLC)  CLOSE  ;===================================================  ;; \file  ;; Define motion for Inboard/Outboard blades  ;;  ;; Defined axes:  ;; - X (addr 6) = CENTER in same EGUs as blade  ;; - Y (addr 7) = gap in same EGUs as blade  ;;  ;; Macros:  ;; - COORD = $(COORD) CS number, e.g. 3  ;; - PLC = $(PLC) PLC number, should be CS  ;; number+15, e.g. 18  ;; - OUT = $(OUT) Axis num for Blade+, e.g. 6  ;; - IN = $(IN) Axis num for Blade-, e.g. 7  ;---------------------------------------------------  #define COORD 3  #define PLC 18  #define OUT 6 ; Outboard blade on Axis 6  #define IN 7 ; Inboard blade on Axis 7  ; Change to CS$(COORD)  &$(COORD)  ; Set relevant axes to use kinematics  #$(OUT)->I ; +ve blade  #$(IN)->I ; -ve blade  ;---------------------------------------------------  ; These are set by motor\_in\_cs.template  #define OUT\_MOVE P(4700+$(OUT))  #define OUT\_MRES P(4800+$(OUT))  #define OUT\_OFF P(4900+$(OUT))  #define IN\_MOVE P(4700+$(IN))  #define IN\_MRES P(4800+$(IN))  #define IN\_OFF P(4900+$(IN))  ;---------------------------------------------------  ; Calculate gap and CENTER from blade positions  #define OUT\_POS (OUT\_MRES\*P$(OUT)+OUT\_OFF)  #define IN\_POS (IN\_MRES\*P$(IN)+IN\_OFF)  #define CENTER Q7  #define GAP Q8  OPEN FORWARD  CLEAR  CENTER = (OUT\_POS + IN\_POS)/2  GAP = OUT\_POS - IN\_POS  CLOSE  ;---------------------------------------------------  ; Calculate blade positions in cts from gap and center  #define OUT\_POS Q228  #define IN\_POS Q229  OPEN INVERSE  CLEAR  ; calculate in EGUs  OUT\_POS = CENTER + GAP/2  IN\_POS = CENTER - GAP/2  ; then in cts  P$(OUT) = (OUT\_POS - OUT\_OFF)/OUT\_MRES  P$(IN) = (IN\_POS - IN\_OFF)/IN\_MRES  CLOSE  ;---------------------------------------------------  ; A PLC(sx+15) needs to be made to do position reporting  ; Readbacks should be in &{axisnum}Q81..89  ; As forward kinematic, but with Px = mx62/(Ix08\*32)  #define OUT\_POS (OUT\_MRES\*m$(OUT)62/(I$(OUT)08\*32)+OUT\_OFF)  #define IN\_POS (IN\_MRES\*m$(IN)62/(I$(IN)08\*32)+IN\_OFF)  #define CENTER Q87  #define GAP Q88  OPEN PLC$(PLC)  CLEAR  ADDRESS&$(COORD)  CENTER = (OUT\_POS + IN\_POS)/2  GAP = OUT\_POS - IN\_POS  ; If blade motor record did the last move, set demands = readbacks  if (OUT\_MOVE = 1)  or (IN\_MOVE = 1)  OUT\_MOVE = 0  IN\_MOVE = 0  Q77 = Q87  Q78 = Q88  endif  CLOSE  ENABLE PLC$(PLC) |

## **EPICS integration**

### **cs.substitutions**

cs.substitution is used to identify where to obtain the virtual motor information.

|  |
| --- |
| file "db/motor.db"  {  # ADDR ---> axis #  pattern  { P, M, MOTOR, ADDR, DESC, DTYP }  { "XF:28IDD-ES:2", "{Slt-Ax:YGap}Mtr", "CS2", 6, "Slit Y gap", asynMotor }  { "XF:28IDD-ES:2", "{Slt-Ax:YCtr}Mtr", "CS2", 7, "Slit Y center", asynMotor }  }{ "XF:28IDD-ES:2", "{Slt-Ax:XGap}Mtr", "CS3", 6, "Slit X gap", asynMotor }  { "XF:28IDD-ES:2", "{Slt-Ax:XCtr}Mtr", "CS3", 7, "Slit X center", asynMotor }  }  file "db/motor\_in\_cs.template"  {  pattern  { P , M , SPORT }  { "XF:28IDD-ES:2" , "{Pinhole-Ax:X}" , P0 }  { "XF:28IDD-ES:2" , "{Pinhole-Ax:Y}" , P0 }  { "XF:28IDD-ES:2" , "{Slt-Ax:I}" , P0 }  { "XF:28IDD-ES:2" , "{Slt-Ax:O}" , P0 }  { "XF:28IDD-ES:2" , "{Slt-Ax:T}" , P0 }  { "XF:28IDD-ES:2" , "{Slt-Ax:B}" , P0 }  } |

### **st.cmd**

Besides normal motor configuration in st.cmd, the following should be added.

|  |
| --- |
| # Initialize the coord-system(port, addr, cs, ref, prog#)  pmacAsynCoordCreate("P0", 0, 2, 2, 10)  pmacAsynCoordCreate("P0", 0, 3, 3, 10)  # setup the coord-sys(portName, drvel-name, ref#(from create), nAxes+1)  drvAsynMotorConfigure("CS2", "pmacAsynCoord", 2, 9)  drvAsynMotorConfigure("CS3", "pmacAsynCoord", 3, 9)  # Set coordinate system scaling  # Default value is 10000  # pmacSetCoordStepsPerUnit(CS, CS\_axis, scaling\_factor)  pmacSetCoordStepsPerUnit(2, 6, 1.0)  pmacSetCoordStepsPerUnit(2, 7, 1.0)  pmacSetCoordStepsPerUnit(3, 6, 1.0)  pmacSetCoordStepsPerUnit(3, 7, 1.0)  dbLoadTemplate("db/cs.substitutions") |