Shannon Way
Tewkesbury
Gloucester GL20 8ND
Ph: 011.44.1684.292333
Fx: 011.44.1684.297929

1000 Gamma Drive
Suite 206
Pittsburgh, PA 15238
Ph: 412.968.9744
Fx: 412.968.9746
www.triomotion.com

A P P L I C A T I O N N O T E

Doc No.:	**TN30-12**
Date:	**24 Jul 2001**
Version:	**1.0**
Subject:	**Coordinate Transformation – FRAME=2**

Another useful feature built-in to all Trio controllers is the ability to do frame coordinate transformation.

Applications for this feature would include positioning a 2D or 3D mechanism using standard TrioBASIC move functions.

Frame transformations are used to allow movements to be specified in a multi-axis coordinate frame of reference, which do not correspond one-to-one with the axes. A good example would be a SCARA robot. To position the end tip of a jointed SCARA robot arm and perform straight-line movements in X-Y, the motors need to move in a pattern determined by the robots geometry.

A machine system can be specified with several different "frames". The currently active FRAME is specified with the FRAME system parameter. The default FRAME is 0, which corresponds to a one-to-one transformation.

FRAME=0 (default)

The illustration on the following page shows the default one-to-one frame transformation with FRAME=0 for a conventional X-Y system. In this mode, an interpolated move of MOVE(0,100) produces motion on the Y-motor only to raise the load vertically. Note that the weight of the Y-motor must be carried by the X-motor.

FRAME=1

(Reserved)

FRAME=2 (Axes 0 and 1 only)

Switching to FRAME=2 will allow X-Y motion using a single-belt configuration (see illustration). In this mode, an interpolated move of MOVE(0,100) produces motion on both motor1 and motor2 to raise the load vertically, based on the transformed position. Note that the two motors are located on the X-axis. The mass of the Y-axis can be minimized in this configuration. The equations for the transformed position of the X and Y axes are as follows:

$$X_{transformed} = (MPOS\ AXIS(0) + MPOS\ AXIS(1)) * 0.5$$

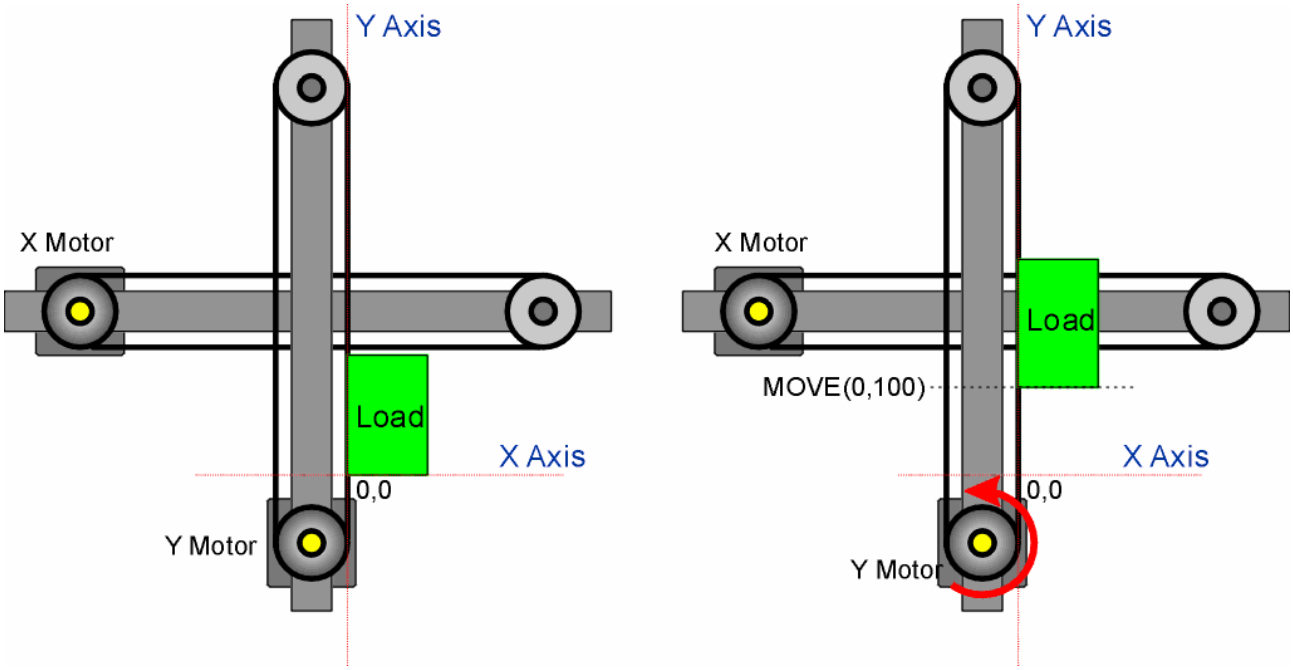
$$Y_{transformed} = (MPOS\ AXIS(0) - MPOS\ AXIS(1)) * 0.5$$

The transformed X-Y coordinates are derived from the measured encoder position (MPOS) of Axis(0) and Axis(1). This conversion is automatically accomplished by the *Motion Coordinator*™ when FRAME=2.

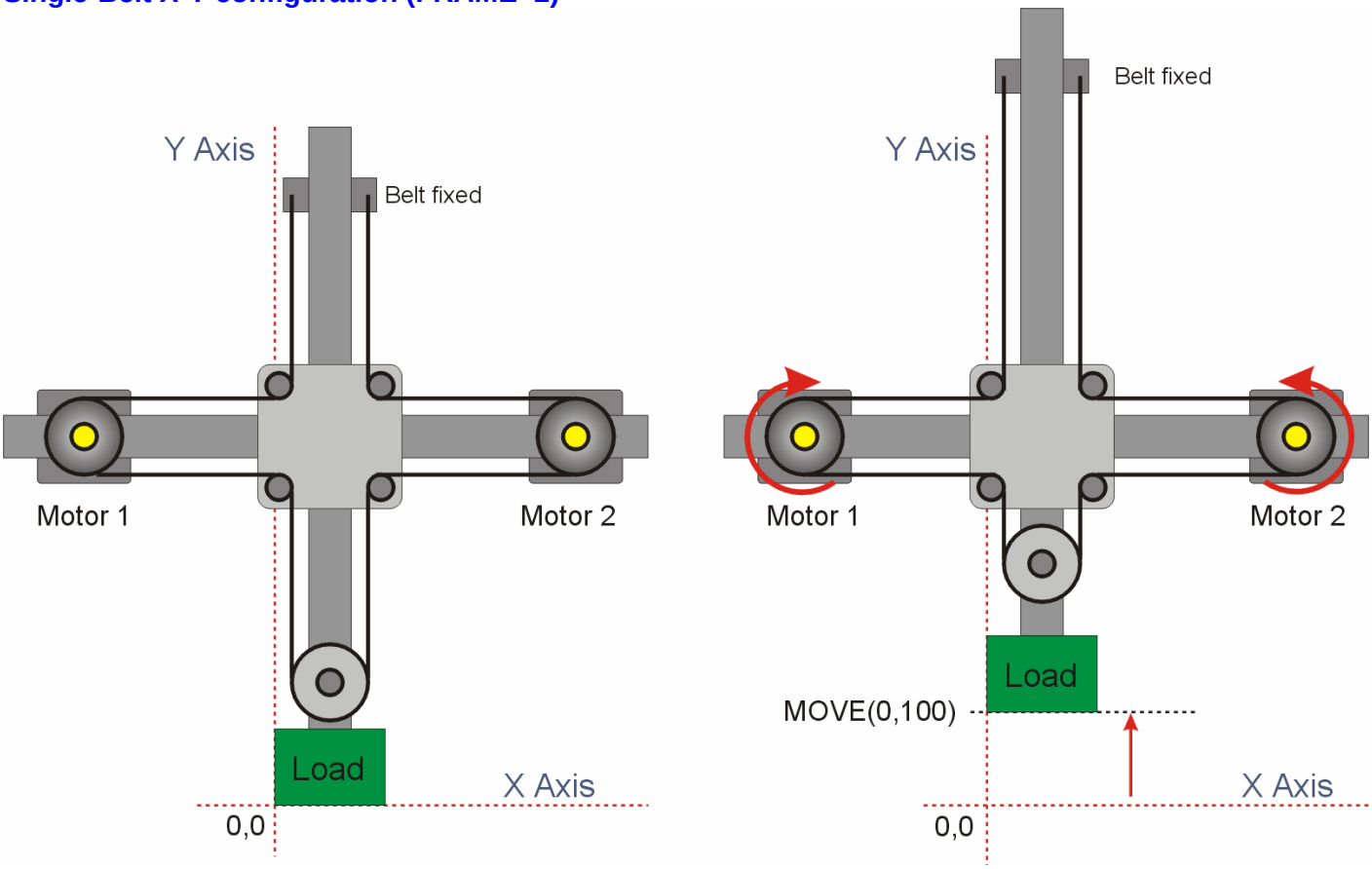
FRAME=?

Frame transformations to perform additional functions that can be defined need to be compiled from "C" language source and loaded into the controller system software. Contact Trio if you need to do this. A Texas Instruments DSP development system will be required plus the appropriate compilers.

Conventional Dual-belt X-Y configuration (FRAME=0)



Single-Belt X-Y configuration (FRAME=2)



Software end of travel limits (FRAME=2)

When switched to frame coordinate transformation the software end of travel limits RS_LIMIT and FS_LIMIT cannot be set in the conventional sense. Since MPOS for both axes are transformed to give an X and Y coordinate, the real world coordinate must be calculated and monitored instead. Once an end of travel has been detected, it is then possible to force the FS_LIMIT or the RS_LIMIT to the transformed value allowing detection of the forward and reverse limits via the AXISSTATUS command on each axis.

Code example:

```
'Setup SOFT limit position in "Real World" positions (FRAME=2)

BASE(0,1)'               Set reference axes
pos_limit_x=100 '        real world EOT pos in user units - x axis forward
neg_limit_x=-100 '       real world EOT pos in user units - x axis reverse
pos_limit_y=100 '        real world EOT pos in user units - y axis forward
neg_limit_y=-100 '       real world EOT pos in user units - y axis reverse

'============================== MAIN LOOP ====================================
loop:

    'Convert individual encoder positions to Transformed positions (actual)
    mposx=(MPOS AXIS(0) + MPOS AXIS(1))*0.5
    mposy=(MPOS AXIS(0) - MPOS AXIS(1))*0.5
```

```
   'Check for over travel in the X DIMENSION
    IF (mposx < neg_limit_x) OR (mposx > pos_limit_x) THEN
       IF SGN(mposx)=1 THEN
             FS_LIMIT=MPOS AXIS(0) '          Force axis to go into Soft Forward Limit
       ELSE
             RS_LIMIT=MPOS AXIS(0) '          Force axis to go into Soft Reverse Limit
       ENDIF
       GOSUB stopallmoves
    ENDIF

    'Check for over travel in the Y DIMENSION
    IF (mposy<neg_limit_y) OR (mposy>pos_limit_y) THEN
       IF SGN(mposy)=1 THEN
             RS_LIMIT=MPOS AXIS(1) ‘          Force axis to go into forward Soft Limit
       ELSE
             FS_LIMIT=MPOS AXIS(1) '          Force axis to go into Reverse Soft Limit
       ENDIF
       GOSUB stopallmoves
    ENDIF
GOTO loop
STOP
' ================================ END MAIN LOOP ==========================
' Subroutines
stopallmoves:
    BASE(0)  '             Cancel moves on base axis
    old_decel=DECEL
    DECEL=10000000  '      Increase Decel to reduce overshoot
    REPEAT
       RAPIDSTOP  '        Cancel the remaining move(s)
       WAIT IDLE
    UNTIL ((MTYPE=0) AND (NTYPE=0))
    DECEL=old_decel '      set back the old decel rate
RETURN

' End of subroutines
```