TRIO MOTION TECHNOLOGY
www.triomotion.com

Trio Motion Technology Ltd.
Shannon Way, Tewkesbury,
Gloucestershire. GL20 8ND
United Kingdom
Tel: +44 (0)1684 292333
Fax: +44 (0)1684 297929

1000 Gamma Drive
Suite 206
Pittsburgh, PA 15238
United States of America
Ph: +1 412.968.9744
Fx: +1 412.968.9746

Doc No.: **TN20-77**
Version: **1.0**
Date: **1st August 2005**
Subject: Techniques for Measuring Axis Speed

Application Note

1. Introduction:

In many applications there is a need to measure and display the actual speed of an axis. Although there is a *Trio*BASIC parameter for the speed measured from the axis' feedback, called MSPEED, there are some programming techniques that must be understood before it can be used effectively.

2. MSPEED limitations:

The MSPEED parameter reads the change in actual position (MPOS) over the elapsed time of one SERVO_PERIOD. It should be understood that the resolution of Measured Position, MPOS, is to the nearest increment of the feedback encoder or other measuring device. Therefore, the accuracy and resolution of MSPEED is very much limited by the number of MPOS counts per millisecond in a typical system.

The value returned to BASIC is also adjusted to report distance per second so that it is normalised and not affected numerically by changes in SERVO_PERIOD. If the SERVO_PERIOD is set to 1 millisecond, then MSPEED is given by:

```
MSPEED = change_in_MPOS * 1000
```

This is updated every millisecond and is a "snapshot" of the encoder movement in the previous millisecond.

The result of this is that at low speed or with low resolution feedback devices, the MSPEED values will seem to jump around excessively. For example you may see successive values like:

2000
2000
4000
1000
3000

The actual measured change in MPOS is 2, 2, 4, 1, 3 etc.

3. Example programs:

Here are some examples to help smooth out the effects caused by these limitations.

1.

```
 ` Rolling average function to smooth out the fluctuations in MSPEED
 ` Each msec, this function adds the latest MSPEED value to the total
 ` of n-1 values.  The smoothed value is then (total / n)
 `
n = 500
VR(20) = MSPEED * n ` insert starting value
VR(21) = VR(20) / n ` starting value for average
WHILE TRUE
  VR(20) = VR(20) - VR(21) ` reduce total to n-1 values
  VR(20) = VR(20) + MSPEED ` add latest measured speed
  VR(21) = VR(20) / 500 ` calculate the latest average
  WA(1) ` one millisecond wait
WEND

 ` VR(21) contains the average mspeed value to be used for display
```

2.

```
 ` measured speed calculated from 2 readings of position over a
 ` specified time period
 `
p = 500
VR(21) = 0 ` initialise the speed to zero
WHILE TRUE
  TICKS = p
  VR(20) = MPOS
  WAIT UNTIL TICKS<=0
  VR(21) = (MPOS - VR(20)) * 1000 / p ` calculate change per second
WEND
```

Example one gives the smoother reading but is slower to respond to changes as it is analogous to an RC circuit in electronics. The second example will be faster to react to changes in speed but will appear to change in “jumps” at the interval p. (in this case each half second.)