Filtering ORMEC Variables

Abstract

Some applications need to display the real time value of a constantly changing ORMEC variable such as DRV.CMD@ (commanded torque) or VEL.ACT@ (actual speed). In their raw form, these variables return a snapshot of the variable taken during one DSP update tick and may fluctuate from each reading to the next. This Tech Note shows you how to implement a software filter to "smooth" out the readings.

Description

The example application has a requirement to display a constantly updated value of commanded motor torque as a percentage of the motor maximum continuous torque. The value is to be updated five times per second in a timed interrupt routine.

While the example displays the filtered value on the screen, the same technique can be used in applications where the value is used to change process parameters or to check for mechanical overloads. The technique can be used to filter any variable in your program.

Implementation

The following code sets up an interrupt routine which will handle the update:

```
TIME.INT = 1 'pick a DIO point from 1-16
IO.MODE@(TIME.INT) = "O" 'define the DIO point as an output
ON EVENT DIO@(TIME.INT) GOSUB INT.ROUTINE
PERIOD = -200 'the interrupt period will be 200 ms
CONSTANT = 10 'Filter constant
```

The first three lines assign an I/O point to be the interrupt, configure the point as an output and set up the interrupt subroutine. While you cannot use the I/O point for any other purpose, you don't need to install an actual I/O module in this location to use it as a timed interrupt.

Line 4 sets up a variable for the interrupt period in milliseconds. It is used in a timed DIO@ statement to turn the TIME.INT output on after the specified time and should therefore be proceeded by a minus sign.

CONSTANT is a variable for the filter constant. The larger the number, the more filtering you will have.

This code enables the interrupt:

```
EVENT ON 'enable all events
EVENT DIO@(TIME.INT) ON 'turn the timer event processing on
DIO@(TIME.INT) = PERIOD 'turn the interrupt output ON after PERIOD ms
```
The interrupt routine is:

```
INT.ROUTINE:
  IF MODE@(axis) THEN
    ACC! = ACC! + DRV.CMD@(axis) - ACC! / CONSTANT
  ELSE
    ACC! = 0 'zero the accumulated value if the drive is disabled
  ENDIF
  PRINT@ 1,1 USING "#####"; ACC! / CONSTANT 'print the value
  DIO@(TIME.INT) = PERIOD 'restart the timed output
RETURN
```

If the drive is enabled, the routine updates the filter accumulator. If it is not enabled, it zeros the accumulated value. After printing the value, it sets up the interrupt input to turn on again after PERIOD milliseconds.

If the start up response of the filter is too slow, you may be able to improve it by changing INT.ROUTINE as shown below. The penalty for this change will be a slightly longer execution time for INT.ROUTINE.

```
INT.ROUTINE:
  IF MODE@(axis) THEN
    IF ACC! = 0 THEN ACC! = DRV.CMD@(axis) * CONSTANT
    ACC! = ACC! + DRV.CMD@(axis) - ACC! / CONSTANT
  ELSE
    ACC! = 0 'zero the accumulated value if the drive is disabled
  ENDIF
  PRINT@ 1,1 USING "#####"; ACC! / CONSTANT 'print the value
  DIO@(TIME.INT) = PERIOD 'restart the timed output
RETURN
```

To print the value as a percentage of the motor maximum torque, calculate a conversion factor by adding the following line to the initial setup code.

```
FACTOR! = (PK.TRQ! / CONT.TRQ!) * (100 / (DRV.MAX@ * CONSTANT))
```

PK.TRQ! and CONT.TRQ! are the motor’s maximum peak and continuous torques respectively.

To display the percentage value, replace the PRINT statement in INT.ROUTINE with:

```
PRINT@ 1,1 USING "### %"; ACC! * FACTOR! 'convert to % and print
```

Additional Information

For additional information on MotionBASIC® EVENT and timed DIO@ features refer to the MotionBASIC® Hypertext Manual.