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

TECHNICAL NOTE

Doc No.:	**TN30-25**
Date:	**March 22, 2007**
Version:	**1.0**
Subject:	**CANopen communications to FESTO CPX**

OVERVIEW:

This document demonstrates a simple method of reading and writing I/O on two FESTO CPX-FB14 modules using CANopen. In this application it was desired to control some pneumatic valves directly with the FESTO hardware instead of implementing Trio CAN I/O modules. Shown below is a picture of the temporary system used to test the TrioBASIC program.

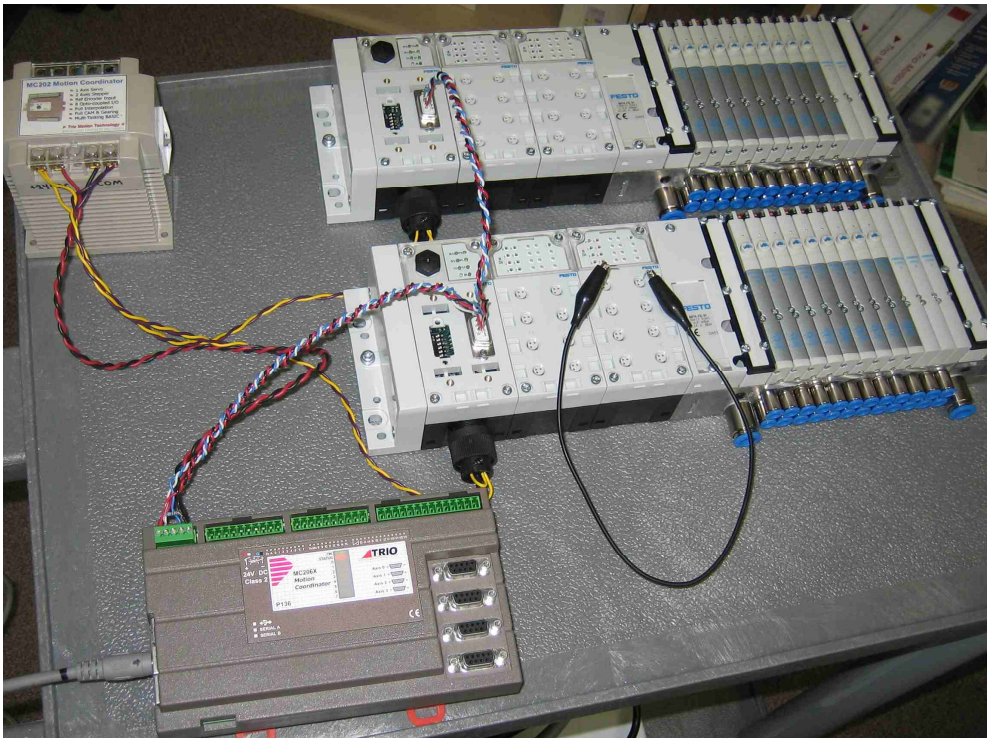

Fig. 1 - FESTO CANopen test system.

Since the on-board CAN port of the Trio controller is used here for CANopen communications, it would not be available for use with the Trio CAN I/O modules.

There are some DIL switches on the Festo modules that must be set for the appropriate CAN address and baudrate.

Fig. 2 - Festo DIL switches.

The program can be tested by using the MotionPerfect VR Viewer window. See the program listing below for further operation details.

PROGRAM LISTING:

```
'====================================================================================
' Filename: CAN_2FESTO.BAS
' Abstract: Simple program that allows you to read and write FESTO CPX I/O
'           using CANopen.
'------------------------------------------------------------------------------------
' Notes:
' Open Terminal window #5 for debug messages.
' Open VR Viewer from 0 to 30 to see/set VRs easily.
' Set VR(0) for output byte 0 of FESTO CPX station #1
' Set VR(1) for output byte 1 of FESTO CPX station #1
' Set VR(2) for output byte 2 of FESTO CPX station #1
' Set VR(3) for output byte 0 of FESTO CPX station #2
' Set VR(4) for output byte 1 of FESTO CPX station #2
' Set VR(5) for output byte 2 of FESTO CPX station #2
' Set VR(vrbase1)=0 to read input bytes of FESTO CPX station #1
' Set VR(vrbase2)=0 to read input bytes of FESTO CPX station #2
' See TrioBASIC Help for more info on CAN command.
' Uses the built-in CAN port on Trio MC.
' Tested with FESTO modules: CPX-FB14, CPX-AB-8-M8, VMPA1-FB-EMS-8.
' The DIL switches on the CPX-FB14 must be set for CAN address & baud.
'====================================================================================
```

```
IF CANIO_ADDRESS<>33 THEN
      PRINT#5,"Must set CANIO_ADDRESS=33 and cycle power"
      STOP
ENDIF

WA(4000) 'make sure FESTO is ready after power up
GOSUB init_vars 'init some program variables
GOSUB init_canopen 'init canopen comms to FESTO (once after powerup)

'--Main loop that reads input bytes & writes output bytes.
WHILE 1=1

  '--Handle CPX terminal #1 I/O ------------------------------------------------
  '--Check inputs when VR(vrbase1) is set to 0, gets reset automatically.
  IF VR(vrbase1)=0 THEN
    CAN(ch,rd_msg,buf1,vrbase1)
    WA(10)
    PRINT#5,"in1_byte0=";VR(vrbase1+1)[0],"in1_byte1=";VR(vrbase1+2)[0]
  ENDIF

  '--Check VR(0) for change to send new output command.
  IF VR(0)<>vr0 THEN
    CAN(ch,wr_msg,buf2,VR(0),VR(1),VR(2)) 'write output image
    vr0=VR(0) 'save old value
    PRINT#5,"out1_byte0=";VR(0)[0],"out1_byte1=";VR(1)[0],"out1_byte2=";VR(2)[0]
  ENDIF

  '--Check VR(1) for change to send new output command.
  IF VR(1)<>vr1 THEN
    CAN(ch,wr_msg,buf2,VR(0),VR(1),VR(2)) 'write output image
    vr1=VR(1) 'save old value
    PRINT#5,"out1_byte0=";VR(0)[0],"out1_byte1=";VR(1)[0],"out1_byte2=";VR(2)[0]
  ENDIF

  '--Check VR(2) for change to send new output command.
  IF VR(2)<>vr2 THEN
    CAN(ch,wr_msg,buf2,VR(0),VR(1),VR(2)) 'write output image
    vr2=VR(2) 'save old value
    PRINT#5,"out1_byte0=";VR(0)[0],"out1_byte1=";VR(1)[0],"out1_byte2=";VR(2)[0]
  ENDIF

  '--Handle CPX terminal #2 I/O ------------------------------------------------
  '--Check inputs when VR(vrbase2) is set to 0, gets reset automatically.
  IF VR(vrbase2)=0 THEN
    CAN(ch,rd_msg,buf3,vrbase2)
    WA(10)
    PRINT#5,"in2_byte0=";VR(vrbase2+1)[0],"in2_byte1=";VR(vrbase2+2)[0]
  ENDIF

  '--Check VR(3) for change to send new output command.
  IF VR(3)<>vr3 THEN
    CAN(ch,wr_msg,buf4,VR(3),VR(4),VR(5)) 'write output image
    vr3=VR(3) 'save old value
    PRINT#5,"out2_byte0=";VR(3)[0],"out2_byte1=";VR(4)[0],"out2_byte2=";VR(5)[0]
  ENDIF
```

```
 '--Check VR(4) for change to send new output command.
  IF VR(4)<>vr4 THEN
    CAN(ch,wr_msg,buf4,VR(3),VR(4),VR(5)) 'write output image
    vr4=VR(4) 'save old value
    PRINT#5,"out2_byte0=";VR(3)[0],"out2_byte1=";VR(4)[0],"out2_byte2=";VR(5)[0]
  ENDIF

  '--Check VR(5) for change to send new output command.
  IF VR(5)<>vr5 THEN
    CAN(ch,wr_msg,buf4,VR(3),VR(4),VR(5)) 'write output image
    vr5=VR(5) 'save old value
    PRINT#5,"out2_byte0=";VR(3)[0],"out2_byte1=";VR(4)[0],"out2_byte2=";VR(5)[0]
  ENDIF
WEND

init_vars:
  PRINT#5,"init_vars"
  ch=-1 'set MC CAN channel to use
  cpx1=1 'set first FESTO CPX CAN station number
  cpx2=2 'set second FESTO CPX CAN station number
  vrbase1=10 'starting VR() address where CAN data is recieved
  vrbase2=20 'starting VR() address where CAN data is recieved
  baud=2
  rx_chk=3
  msg_init=5
  rd_msg=6
  wr_msg=7
  buf0=0
  buf1=1
  buf2=2
  buf3=3
  buf4=4

  FOR i = 0 TO 30 'zero some VRs
    VR(i)=0
  NEXT i
RETURN

init_canopen:
  PRINT#5,"Init CANopen...";
  CANIO_ENABLE=OFF
  CAN(ch,baud,1) 'set CAN baudrate (1=500k)

  '--Init MC CAN buffers (remove 6th parameter for MC's with older CAN IC).
  CAN(ch,msg_init,buf0,$0,2,1) 'NMT cob_id=0, 2 bytes
  CAN(ch,msg_init,buf1,$180+cpx1,8,0) 'TPDO1 (FESTO tx) 8 bytes
  CAN(ch,msg_init,buf2,$200+cpx1,8,1) 'RPDO2 (FESTO rx) 8 bytes
  CAN(ch,msg_init,buf3,$180+cpx2,8,0) 'TPDO1 (FESTO tx) 8 bytes
  CAN(ch,msg_init,buf4,$200+cpx2,8,1) 'RPDO2 (FESTO rx) 8 bytes

  '--Set FESTO CPX terminals to operational state.
  CAN(ch,wr_msg,buf0,1,cpx1)
  CAN(ch,wr_msg,buf0,1,cpx2)
  WA(1000)
  PRINT#5,"complete."
RETURN
'================================================================================
```