▶Trio Motion Technology◀

Trio Motion Technology Ltd.
Shannon Way,
Tewkesbury
Glos. GL20 8ND U.K.

Tel: 01684 292333
Fax: 01684 297929
Email: apps@triomotion.com
Web: www.triomotion.com

Application Note

Doc No.: **TN20-31**
Version: **1.1**
Date: **Feb 2003**
Subject: **Using the PROFIBUS Daughter Board (P297)**

Introduction

This document will allow a programmer to install and set-up the Trio BASIC program "P297DRXXX.BAS" in order to attach the Motion Coordinator to a Profibus fieldbus network as a slave using the P297 Profibus Daughter Board.

Scope of Operation

This document applies to the BASIC program developed for Motion Coordinator types MC204, MC206, MC216, MC224 and Euro205. The program is provided for evaluation and example purposes and no guarantee is made as to its suitability for a particular Profibus application.

In order to include the Motion Coordinator in a Profibus network the following components are required:

1. Trio BASIC program P297DRxxx.bas (where xxx is the version number of the program)
2. Profibus GSD file; TRIO0595.GSD. (Electronic Data Sheet for COM PROFIBUS)
3. This Document.
4. Motion Perfect and serial programming cable.

Installation and Set-up

1. Trio BASIC program.

The program must be loaded into the Motion Coordinator and set to run from power-up. The following variables must be set according to the system requirements.
node sets the Profibus Address for the Motion Coordinator on the network.

db is set to the slot number of the Profibus Daughter Board.

vbase is the base address pointing to a group of VRs that will be used for data transfer in and out of the profibus chip. 33 VRs are used for this purpose.

localtimeout sets the time in milliseconds after which the program will reset the Profibus chip if communications is lost with the Profibus master.

Once the SPC3 chip on the daughter board is initialised, the software is very efficient at transferring data in and out. In the MC204, however, a process number of 4 or 5 is recommended for optimum running. When using the MC206 and MC224, a low process number may be used with little impact on processing speed. It should be noted, however, that there is no guarantee that all 32 VRs will be transferred to/from the VRs in one bus cycle because the multi-tasking can interrupt VR transfer in to/out of the SPC3 chip at any time.

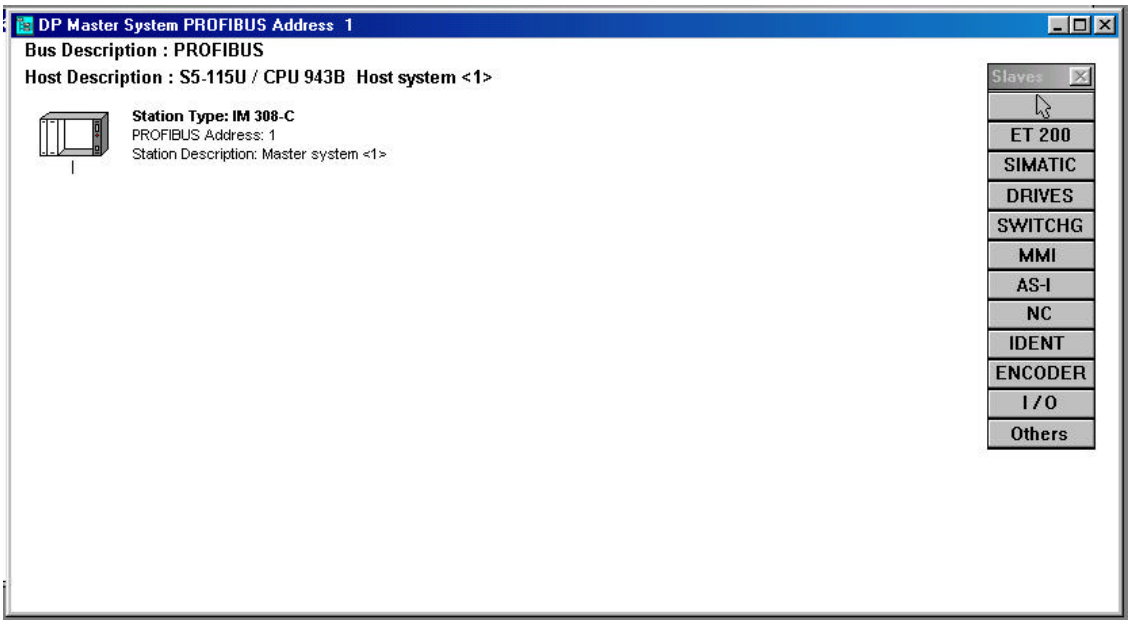
It is recommended that a system of flags is used to indicate that the data is ready for another process to use.

2. Profibus GSD File.

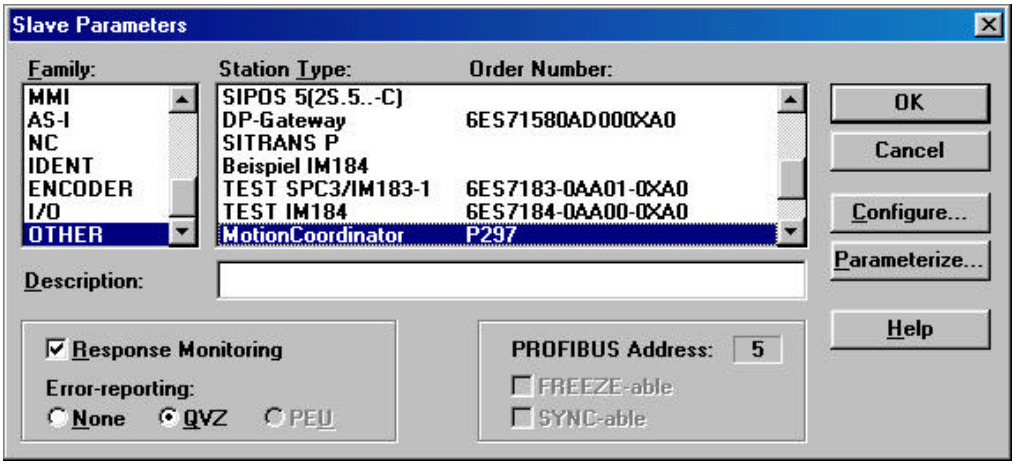
The GSD file supplied (TRIO0595.GSD) must be copied to the GSD folder used by COM PROFIBUS or the equivalent Profibus configuration tool supplied by the PLC vendor. The Motion Coordinator can then be added to the Profibus network by selecting OTHER and P297 Motion Coordinator from the list.

The following sequence shows how to include the Motion Coordinator in a fieldbus network using COM PROFIBUS.

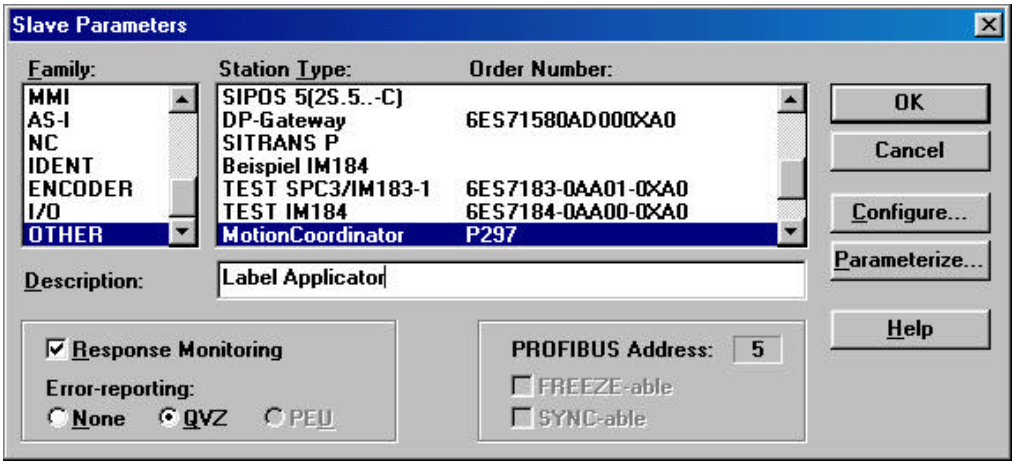
1. Launch the window shown below and click on Others.



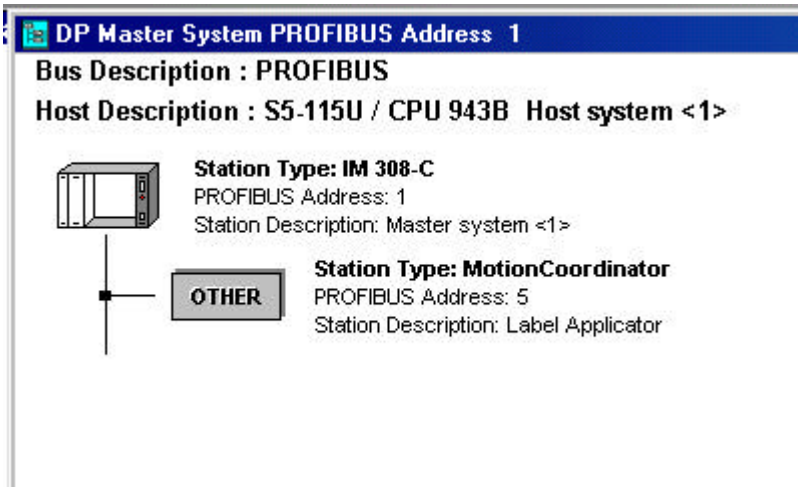
2. Add the Box Icon to the network on the left and the Slave Parameters dialogue will open. Choose Motion Coordinator P297 from the list as shown here:



3. Add your own description in the text box like this:



4. Click OK and the Motion Coordinator will appear on the diagram like this:



Now add any other nodes to the network that are required and close the window. Finally export the file in the required format, usually Binary, for use by the PLC or other Profibus Master. The master will now search for the Motion Coordinator on the Profibus network and when found will connect to it and start transferring the variable blocks.

Appendix I Trio BASIC Program Settings

```
'=====================================================================
' DESCRIPTION:
'  Profibus driver for Cyclic Data Transfer
'  This program sets up the SPC3 chip for transfer of 16 integers from
'  master and 16 intergers to master on a cylcic basis as determined by
'  master unit.
'  Variables:
'   VR(vbase) to VR(vbase+15)    : Data TO master (16bit int)
'   VR(vbase+16) to VR(vbase+31) : Data FROM master (16bit int)
'   VR(vbase+32) = PLC status. 0=running, 2=PLC in configure mode
'
' V1.00: 08/01/2001 Beta Release includes full 16+16 VR transfer
' V1.01: 15/03/2001 Reversed byte order for Word transfer
' V1.02: 24/04/2002 Added support for "Leave Data Ex". (S7 config sequence)
'                   Added local timout if no connection after BR detect.
' V1.03: 08/08/2002 Allows VR segment used to be changed by setting a variable
' V1.04: 23/10/2002 Made local timeout value settable.  See "localtimeout"

restart:
RESET
node = 5          <--- Set this to the required Profibus Address
debug = TRUE 'Set TRUE to get debug messages printed to terminal
db=0 '        Daughter Board slot number
vbase = 20 '  VRs for data transfer
localtimeout = 5000 'time in msec

GOSUB user_dps_reset

TICKS=localtimeout
REPEAT
    GOSUB dps2_ind
    'check local watchdog
    IF TICKS<0 THEN GOTO restart
    'has profbus watchdog timed out?
    IF timeout_flag=TRUE THEN GOTO restart
UNTIL FALSE
STOP
```

Appendix II Transfer of data between SPC3 and VRs

The routine shown below does the data transfer between the SPC3 chip and the VR variables. Each pair of bytes in the SPC dual-port ram is treated as the high and low bytes of a 16 bit integer. If other data formats are required then the routine can be edited as required.

Note 1: The PROFIBUS(db, 0, byte_pointer) command does a read, and PROFIBUS(db, 1, byte_pointer, data_byte) does a write to the SPC3 chip.

Note 2: The SPC3 has 2 buffers. While this program reads and writes to one buffer, the Profibus master is exchanging data with the second one. The command line: **PROFIBUS(db,0,9)' rotate input buffers**
swaps the 2 buffers so that the next read/write cycle uses the fresh data.

```
' Write Read Data Changed:
    IF intregh AND 32 THEN
        PROFIBUS(db,1,3,32) ' Acknowledge Interupt

        ' Handle update to outputs (from master):

        dout_buf_cmd=PROFIBUS(db,0,10)/16 AND 3
        dout_buf_ptr=PROFIBUS(db,0,27+dout_buf_cmd-1)*8
        vrb=vbase+16
        FOR ct=0 TO 15
          hibyte=PROFIBUS(db,0,dout_buf_ptr+(2*ct))
          lowbyte=PROFIBUS(db,0,dout_buf_ptr+(2*ct)+1)
          sint=INT(lowbyte+(hibyte*256))
          IF sint>32767 THEN sint=INT(sint-65536)
          VR(vrb+ct)=sint
        NEXT ct
        PROFIBUS(db,0,11)' update output buffers

        ' Handle update to inputs (to master):

        din_buf_cmd=PROFIBUS(db,0,8)/16 AND 3
        din_buf_ptr=PROFIBUS(db,0,31+din_buf_cmd-1)*8
        FOR ct=0 TO 15
          sint=VR(vbase+ct)
          lowbyte=INT(ABS(sint)+0.5)
          IF sint<0 THEN
            hibyte=(lowbyte/256) AND 127 OR 128
          ELSE
            hibyte=(lowbyte/256) AND 127
          ENDIF
          lowbyte=lowbyte AND 255

          PROFIBUS(db,1,din_buf_ptr+(2*ct),hibyte)
          PROFIBUS(db,1,din_buf_ptr+(2*ct)+1,lowbyte)
        NEXT ct
        PROFIBUS(db,0,9)' rotate input buffers

    ENDIF
```

This diagram shows the data paths between the PLC and the VR global variables in the Motion Coordinator. The example shows VRs used when vbase is set to 20.