SERVOnet®

A PTS SYSTEM USING DISTRIBUTED CONTROL UNITS

This manual also covers SynchroLink and using CANopen devices (encoder and I/O) on CANbus enabled PTS systems

MAN 529

Issue 6

August 2004

Copyright Notice

Copyright © 2004 Quin Systems Limited. All rights reserved.

Reproduction of this document, in part or whole, by any means, without the prior written consent of Quin Systems Limited is strictly prohibited.

SERVOnet is a registered trade mark of Quin Systems Ltd.

Software Version

This manual reflects the following firmware/software versions:

- PTS Firmware version 1.9.3.1 or later for SERVOnet®
- PTS Firmware version 1.9.1.1 or later for SynchroLink
- PTS Firmware version 1.9.1.1 or later for CANopen devices

Important Notice

Quin Systems reserves the right to make changes without notice in the products described in this document in order to improve design or performance and for further product development. Examples given are for illustration only, and no responsibility is assumed for their suitability in particular applications.

Although every attempt has been made to ensure the accuracy of the information in this document, Quin Systems assumes no liability for inadvertent errors.

Suggestions for improvements in either the products or the documentation are welcome and should be addressed to:-

Quin Systems Limited
Oaklands Business Centre
Oaklands Park
Wokingham
Bershire
RG41 2FD

Telephone:	0118 977 1077
Facsimilie:	0118 977 6728
E-Mail:	Support@quin.co.uk
WWW Site:	http://www.quin.co.uk/

Contents

Copyright ©2004 Quin Systems Ltd.

 Copyright ©2004 Quin Systems Ltd.

Copyright ©2004 Quin Systems Ltd.

 Copyright ©2004 Quin Systems Ltd.

1. Introduction

SERVOnet is a method of connecting PTS units together to form a multiple axis control system which does not have to reside in a single cabinet but can be strategically placed throughout the machine. The system is programmed as a single unit, with all user interface through a master QManager/Machine Manager.

This manual also covers SynchroLink, a subset of SERVOnet used for linking separate and independent PTS controllers, and for including LinMot actuators within the system: refer to Appendix A. on page 36 for this.

CANopen I/O and encoder devices can be used with SERVOnet and SynchroLink. The use of these is detailed in Appendix B. on page 45.

Issue 6 of this manual introduces:

- Updated wiring diagrams for long or many-noded SERVOnet systems.

Issue 5 of this manual introduces:

- SERVOnet node table (PTS V2.1 and higher)
- SERVOnet error line control (PTS V2.2.1.1 and higher)
- Re-ordered Appendix B. on page 45
- General changes for SRV300 product and other updates.

1.1 Technical Specification

SERVOnet supports the following features:

- Up to 58 axis modules and one QManager/Machine Manager on a SERVOnet network. SERVOnet uses CANbus as the network transport medium.
- Control up to 24 motors using a Mini Machine Manager. Control up to 180 motors using a QManager/Machine Manager. Use any combination of MiniPTS 3 SERVOnet axis modules (3 real motors + 1 optional virtual motor), Q-Drive SERVOnet axis modules (1 real motor + 1 optional virtual motor) and SRV300 (ABB ACS600/ACS800 series with Quin inside) axis modules (1 real motor + 1 optional virtual motor). Future developments will add more types of Axis Modules that can be used on SERVOnet.
- Network length maximum 100m {minimum module-module cable length 10cm}.
- Up to 8 map masters transmitting simultaneously across CANbus.
- Many map slaves can receive one map master signal (see next point).
- Each axis module can support up to 8 unique map masters/slaves (more than one channel can receive the same map slave information within a module, whilst only requiring one unique map slave reception from CANbus).
- Requires only a handful of extra PTS instructions to configure and use SERVOnet.
- Module number clash and network error detection provided using both hardware and software.

1.2 What this Manual Contains

This manual is intended to be used in conjunction with the PTS Reference Manual and PTS User Manual for the PTS system you are installing/using. All the extra information required to install and use a SERVOnet based PTS system is detailed in the following chapters:

- Wiring details for SERVOnet: chapter 3. on page 8
- Initial configuration of SERVOnet modules: chapter 4. on page 14
- New PTS commands for configuring and diagnosing SERVOnet: chapter 5. on page 18
- Hints and tips for writing PTS programs on a SERVOnet system: chapter 6. on page 23
- Detecting and handling hardware errors on SERVOnet: chapter 7. on page 24
- SERVOnet node table; operating SERVOnet with ‘optional’ modules: chapter 8. on page 28
- PTS error codes for SERVOnet: chapter 9. on page 32
- Explanation of LED display codes for SERVOnet: chapter 10. on page 34
- **Appendix A. containing details of SynchroLink - wiring, configuring and using.**
- **Appendix B. containing details of using CANopen encoders and I/O devices on SERVOnet and SynchroLink.**

 Copyright ©2004 Quin Systems Ltd.

2. Glossary

A number of new phrases and words are used throughout this document to refer to SERVOnet features and functions. A glossary of these is provided here:

Term	Explanation
CANbus	The network hardware and transport protocol used for SERVOnet
SERVOnet	Proprietary name used by Quin Systems to describe the linking of multiple axis control systems over CANbus
SynchroLink	Proprietary name used by Quin Systems to describe a subset of SERVOnet that is available on some PTS systems
CANopen	A protocol for CANbus for connecting devices such as encoders and digital and analogue I/O modules together with a master controller
Module	A single PTS unit that connects to SERVOnet
(Mini) Machine Manager	The man/machine interface. One QManager/Machine Manager is required per SERVOnet installation. The PTS program is held on the QManager/Machine Manager and all operator interfaces (such as Operator's Panel) are processed through it.
Axis Module	A PTS unit that provides the motor control functions. This unit can be placed on or near the motors in question and will not need operator access. Many (up to 60) Axis Modules can be connected to one SERVOnet

Table 1: Glossary of Terms used for SERVOnet

Copyright ©2004 Quin Systems Ltd.

3. Wiring SERVOnet

Wiring SERVOnet requires the CANbus cable to be connected to each module to form a single continuous cable with terminators at each end. There is no requirement for modules to be positioned or connected in numerical order. Cable spurs should not be used.

SERVOnet uses additional wires to standard CANbus to provide hardware error detection (fail safe mode of operation). When long runs of cable/many nodes are used this error line requires additional circuitry to function.

This section presents the cable types recommended, pin out descriptions and then wiring scenarios for different lengths of SERVOnet.

3.1 Cable types

The choice of cable is determined primarily by the volt drop due to the supply current to the node buffers: each node takes about 35mA, and the maximum drop in the 0 volt return is 2 volts. The heaviest practicable screened lead is 16/0.2 mm (such as DEF STAN 61-12 pt 5, 16-2-8C), with core resistance about 40 milliohms/metre; 30 nodes and a 50 metre run is the limit for this, allowing centre-fed supply to a maximum network. For lighter loads on a smaller network, Hiperlink patch cable has about 90 milliohms/metre, and is probably adequate for most likely applications. It is possible to split the net into sections, each centre-fed from its own supply, enabling large networks with the thinner cable.

CA2049	4 screened pair Hiperlink cat 5 (7/0.16mm, 26AWG), grey, 100 M reel. Farnell 296-788
CA2050	8 core screened to 61-12 part 5 (16/0.2 mm), black, 100 M reel. Farnell 715-839

Note that for short links it may be simpler to use a ribbon lead with IDC connectors.

The CANbus cable should not exceed 100m total, and individual module to module connections should not be less than 10cm long.

3.2 Connectors

Each PTS module (with the exception of SRV300 and PTSQ5xxx series QDrives) has a male and female 9 pin sub-D type connector for CANbus - thus a simple daisy-chain wiring system can be employed. SERVOnet includes a hardware error detection method built into the CANbus cable itself.

Certain modules (SRV300 and PTSQ5xxx series QDrives) are constrained for space, and use RJ45 connectors rather than D-type. Link cables use pre-made assemblies: drawing QDV-2-2-011 shows these, and the necessary adaptors and termination. Note that RJ45 connectors do not accommodate 'Hot-swap' wiring.

 Copyright ©2004 Quin Systems Ltd.

3.3 Connection table

The connections for both male and female 9 pin sub-D type CANbus connectors are:

Pin	Description
1	*reserved*
2	CAN L (signal)
3	CAN ground (0V)
4	SERVOnet Continuity line
5	CAN shield
6	CAN ground (0V)
7	CAN H (signal)
8	SERVOnet Error line
9	CAN V+ (12V)

Table 2: Pin descriptions for SERVOnet Cable

A 12V DC power supply is required to provide bus power (all PTS modules are optoisolated from the bus) at about 35 mA per node. Each end of the CANbus should be terminated with a 120Ω resistor placed between CAN H and CAN L. The SERVOnet error line may use (this is recommended) additional circuitry placed in the terminators, to provide a fail safe cable continuity check.

3.3.1 Fail safe or hot swapping?

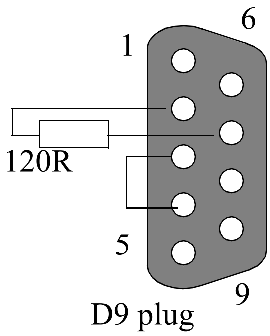
It is possible to wire SERVOnet to two configurations: ‘fail safe’ and ‘hot swapping’.

- Fail safe: All modules are connected using daisy chain wiring. Any hardware failure (see chapter 7. on page 24 for a full discussion of these) will be detected.
- Hot swapping: By wiring Axis modules using a ‘T’ connection it is possible to remove an Axis Module for service/replacement without causing a hardware error state on SERVOnet

Copyright ©2004 Quin Systems Ltd.

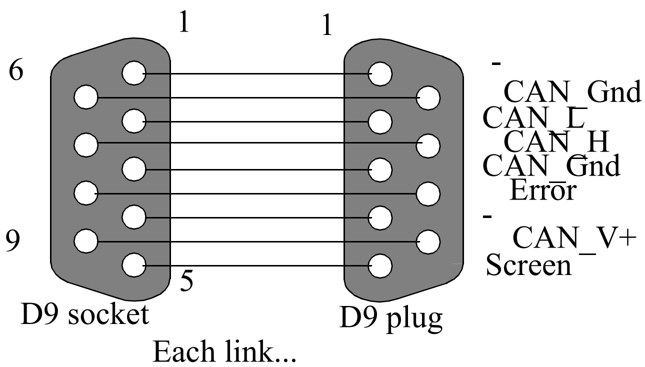
3.4 Wiring for up to 15 nodes

Wiring diagrams for both fail safe and hot swapping configurations are given below:



One terminator requires the 120Ω resistor between CAN H and CAN L and a wire link between pins 3 and 4 for the SERVOnet continuity line.

FAIL SAFE wiring



The SERVOnet cable linking each module together is a straightforward pin to pin connection

The second terminator provides a 120Ω resistor between CAN H and CAN L, the 12V bus power supply and additional circuitry for the SERVOnet continuity line. This consists of a 100Ω 2W resistor and a BAV21 diode

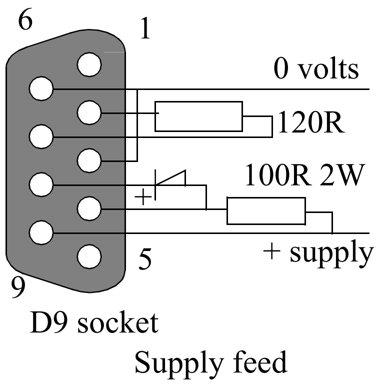


FIG.1: SERVONET WIRING (UP TO 15 NODES) (FOR FAIL SAFE SCENARIO)

To provide an Axis Module with 'Hot swapping' capability, the male sub-D type of the unit should be used, with the SERVOnet cable socket for each Axis Module having two wires for each pin. The resultant cable has a 'T' connection to the Axis Module with 'Hot swapping' capability. For short cable runs within a single cabinet in this mode, a ribbon cable with IDC connectors may be used.

 Copyright ©2004 Quin Systems Ltd.

Both methods require 120 ohm termination resistors at both ends of the bus, placed between CAN H and CAN L.

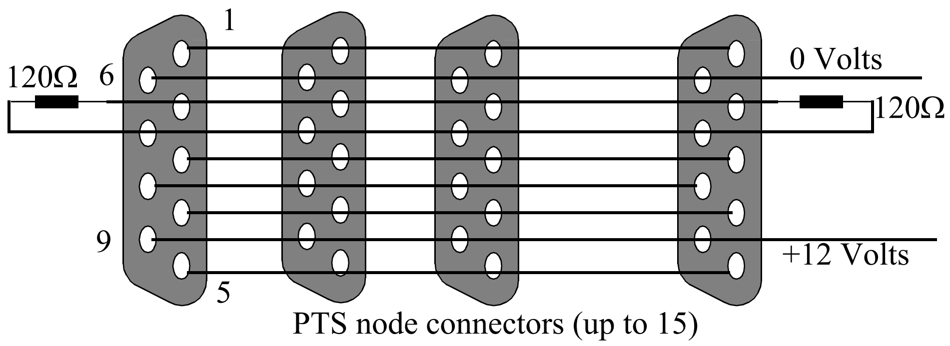


FIG.2: SERVONET WIRING (UP TO 15 NODES) (HOT SWAPPING)

3.5 Cabling for more than 15 nodes (fail safe)

When wiring more than 15 nodes together, or where long cable runs are to be used, it is advisable to "centre feed" the cabling to reduce voltage drop (sockets viewed on mating face).

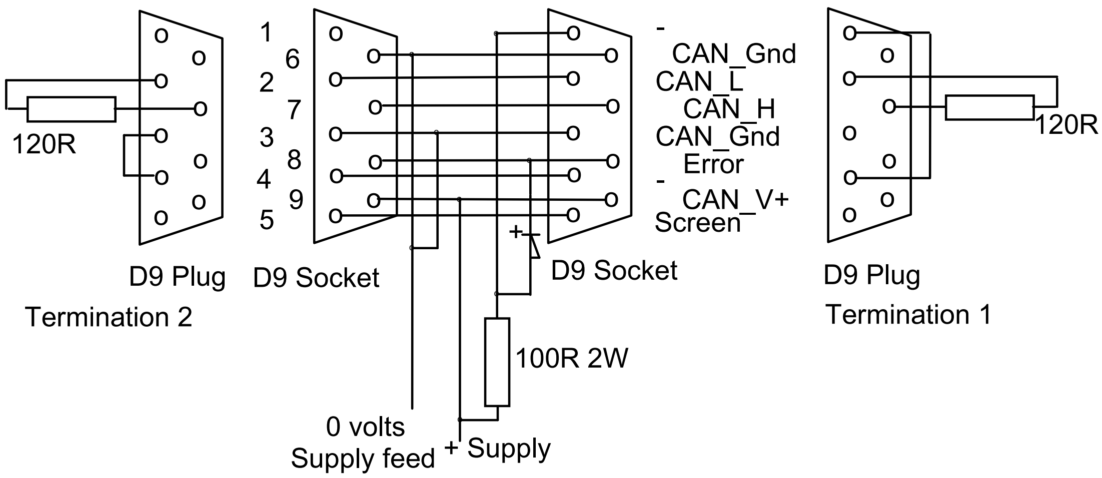


FIG.3: SERVONET WIRING (MORE THAN 15 NODES) (FOR FAIL SAFE SCENARIO)

A centre-feed link should be fitted with two sockets, and power runs out to the ends, with the free socket always carrying the supply and a free plug always 'dead' when disconnected: this avoids accidents due to pins of plugs contacting ground.

Note this cabling is not compatible with the SRV300/PTSQ5xxx product, please contact Quin Systems for more information if required.

3.6 Wiring using an "active" terminator

Quin can supply an "active" terminator for networks containing many nodes (up to 60). This terminator boosts the error line signal and ensures consistent operation. This wiring is only suitable for QDrives/QMotion products with sub-D type connectors. Please contact Quin for details if using the SRV300/PTSQ5xxx product.

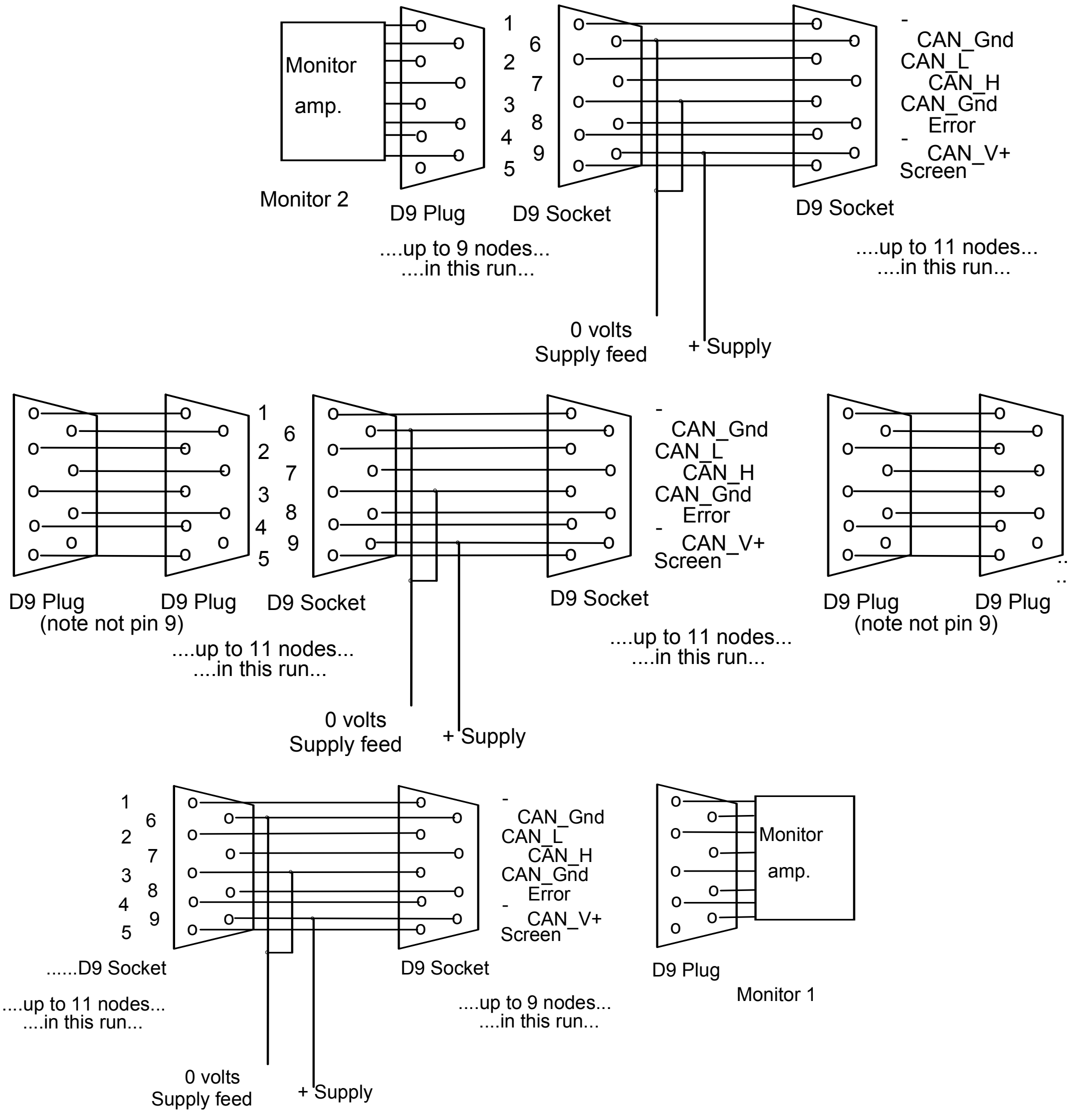


FIG.4: SERVONET WIRING USING ACTIVE TERMINATOR (UP TO 60 NODES)

 Copyright ©2004 Quin Systems Ltd.

Here three power feeds are used, each powering a section of the network. Special link cables are used to join the sections (pin 9, the power feed, is not connected). The active terminators boost and control the error line signal. Note the careful choice of plug and sockets to ensure connection/disconnection does not cause problems with shorting to ground (earth).

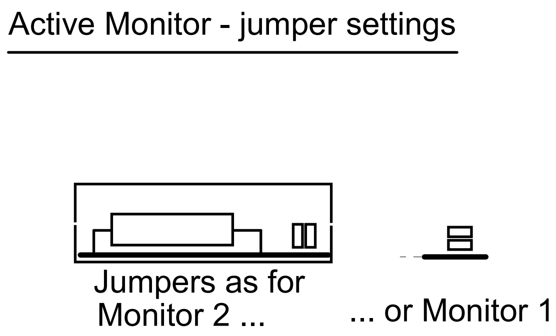


FIG.5: ACTIVE MONITOR JUMPER SETTINGS

3.7 Error Line Logic

The logic of the standard error line and continuity wiring is given in the following figure:

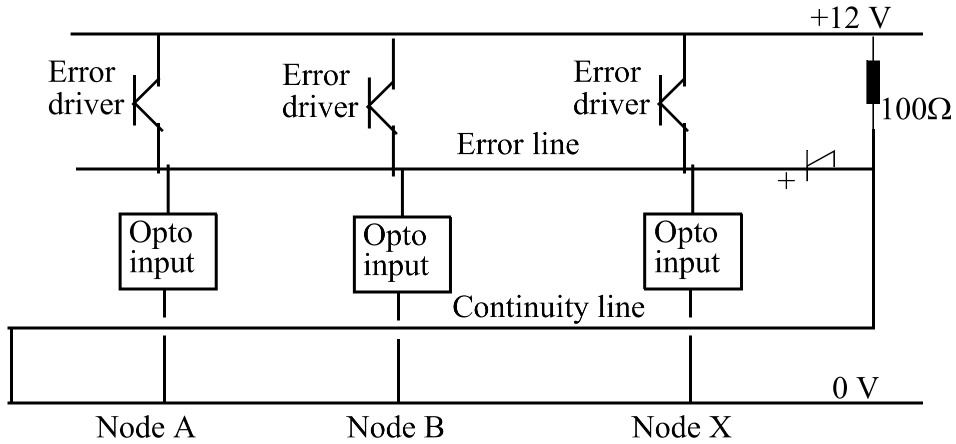


FIG.6: CANBUS WIRING

A node may give an error output by reason of logical faults or lack of node power, and error from any node signals to all nodes. A break in the continuity line also creates an error, which signals to all nodes on the powered side of the break. Nodes unpowered because of the break see a "No CANbus power" error.

4. Configuring SERVOnet

Once the SERVOnet wiring is complete the system can be configured.

4.1 Axis Module Numbering

Each Axis Module requires a unique module number to identify it to the QManager/Machine Manager. The module number of an axis module determines the channel numbers that will be used to refer to its motors. The following rules apply:

- There must be at least one Axis Module on a SERVOnet system
- There must be an Axis Module with module number 1 on a SERVOnet system
- Channel 1 will be the first axis on module 1
- The second Axis Module must have module number 2
- The channel number of the first axis on module 2 will depend upon what module 1 was, and how it was configured: if module 1 is an MiniPTS 3 SERVOnet Axis Module with the virtual channel enabled (see section 4.2 on page 17 for details of this) then channel 5 will be the first axis on module 2, if module 1 is an Q-Drive SERVOnet Axis Module with the virtual channel disabled then channel 2 will be the first axis on module 2
- For further Axis Modules higher module numbers should be used BUT there should never be a missing module number in the sequence
- The channel number for any given axis on any given Axis Module will depend upon how many lower numbered Axis Modules there are and how many axes each of these modules have (channel numbering always starts at 1 and increments with no missing numbers, in the same manner as module numbers)

Note for SERVOnet node table (see chapter 8. on page 28) - the above rules still apply, but instead of detecting these settings from the network a pre-written table is used.

The current module number of an axis module is displayed on the LED(s) as the unit is switched on. For a MiniPTS 3 SERVOnet Axis module the module number will be either ‘Nxx’ or ‘nxx’ (where xx is the module number). For a Q-Drive SERVOnet the same display uses a single LED with the ‘Nxx’ or ‘nxx’ cycling round:

- A capital N indicates that the module number being used is stored in NVM (non volatile memory - memory that remembers its settings whether or not the Axis Module is provided with electricity): this module will always use this number
- A lowercase n indicates that a temporary module number is being used (NVM does not contain a module number), this temporary number could be different every time the unit is switched on

For the SRV300 the node number and other configuration settings are available via the ABB front panel, in parameter group 51 (Comm. Mod. data):

- Param 51.02: Axis module number. If the number is greater than 100 this indicates a temporary module number; i.e. module+100. For example 102 is temporary module number 2

 Copyright ©2004 Quin Systems Ltd.

4.1.1 Module number clashing

No two modules can have the same module number, so the Axis Modules communicate their module number over CANbus when they are switched on and any module number clashes are sorted out using the rules in the following table:

Module A and Module B have the same number:	**Module A NVM**	**Module A temporary**
Module B NVM	Both Axis Modules will turn themselves off	Module A will turn itself off, increment its module number and turn itself on again
Module B temporary	Module B will turn itself off, increment its module number and turn itself on again	*One* module will turn itself off, increment its module number and turn itself on again

Table 3: How module number clashes are resolved

Note that for the case of two temporary modules it is not possible to determine which module will increment its module number: this depends upon which module switches on first, which module communicates over CANbus first, which module waits longest before deciding to increments its module number.

The module number of an Axis Module can be configured a number of ways:

4.1.2 Using Local Serial Port

Serial Port A on an Axis Module can be used to configure the module number. Connect a correctly wired serial cable to Port A, and use PTS Terminal (part of PTS Toolkit 2000) or equivalent. The serial port settings are: 9600baud, 8 data bits, 1 stop bit, no parity, software handshaking. When the Axis Module is switched on the following message (or similar) will appear:

```
Axis Module (MiniPTS 3 SERVOnet) Version 2.2.1.1 Date  2 Dec 02
(press any key)
```

Enter **m** to change the module number, as per the following example:

```
Axis Module (MiniPTS 3 SERVOnet) Version 2.2.1.1 Date  2 Dec 02
(press any key)

M for Module, S for Serial Number, C for Channels, T for Type? m↵
Module number: 1
New Module number ? 2↵

Changes made, REBOOTING...
```

```
Axis Module (MiniPTS 3 SERVOnet) Version 2.2.1.1 Date  2 Dec 02
(press any key)
```

This module number is then stored in NVM. Note that the serial port cannot be used when SERVOnet is working (i.e. when the QManager/Machine Manager has talked to the Axis Module, or when the LED display does not display the module number).

4.1.3 SRV300: Using ABB Front Panel

For the SRV300 product fitted to ABB ACS600 or ACS800 the SERVOnet configuration can be performed from the ABB front panel, parameter group 51 (Comm. Mod. data):

- 51.02: Module number. This may display a value of 100+module number indicating a temporary module number
- 51.05: Channels. Select 1 (real motor only) or 2 (real motor + virtual motor)

If no ABB front panel is available use a serial connection, as described above, instead.

4.1.4 PTSQ5xxx Module number setting

The PTSQ5xxx series of QDrives have a rotary dial switch to set module number:

- Set dial switch to 0 to use software module number (set as for QDrive documented above)
- Set dial switch to 1 - 15 to select module number

4.1.5 Using Auto Module Numbering

It is possible for a fresh SERVOnet system to successfully auto module number itself without any user intervention. This happens because Axis Modules automatically sort out module number clashes. However there is no guarantee that each time the system is turned on the module numbering will be the same, or work at all. Therefore use the CN function from the QManager/Machine Manager to set the correct module numbers into NVM on each Axis Module, and then cycle the power on all units.

4.1.6 Using QManager/Machine Manager

Once a SERVOnet system has been initialised, in other words both the Axis Modules and the QManager/Machine Manager switched on and successfully communicated with each other the CN command (PTS V1.9) or SI command (PTS V2.1 and higher) can be used to change module numbers in Axis Modules. Care should be taken to ensure that the changes do not make the system invalid: the rules governing Axis Module module numbers still apply.

 Copyright ©2004 Quin Systems Ltd.

4.2 Axis Module Virtual Channel Enabling/Disabling

Each Axis Module has one virtual channel. A virtual channel is a PTS motor channel without the ability to connect a motor: the motor is ‘virtual’. These channels are used to provide timing, buffering, mapping and other software features which allow complex machines to be controlled. A MiniPTS 3 SERVOnet Axis Module will control 3 real motors and has one additional virtual channel. A Q-Drive SERVOnet Axis module has one real motor and one virtual channel. On both types of Axis Module there is some functionality available on the virtual channel, such as reading an encoder - for example a master line speed/position feedback.

It is unlikely that more than a handful of virtual channels will be necessary on any one SERVOnet so it is possible to switch off the virtual channels you do not require. This means that you can use the channel numbering more efficiently - and also have a SERVOnet without any virtual channels if required. In this manner a Mini Machine Manager can have 24 real motors using either 8 MiniPTS 3 SERVOnet Axis Modules or 24 Q-Drive SERVOnet Axis Modules.

The virtual channel in each axis module is enabled/disabled using the local serial port, in the same way as the axis module number can be configured (see section 4.1.2 on page 15). Select ‘C’ for channels and then ‘E’ to enable the virtual channel or ‘D’ to disable the virtual channel.

4.3 The QManager/Machine Manager

No specific configuration of the QManager/Machine Manager is required (it is module number 61, and this cannot be changed).

Copyright ©2004 Quin Systems Ltd.

5. PTS commands for SERVOnet

To configure and maintain a SERVOnet system a number of commands have been added to the PTS language.

Note the CK command is not present on SERVOnet. This is because the QManager/Machine Manager is by definition a clock master (CK1), and this cannot be changed.

5.1 CQ CAN Query

CQ performs a network wide analysis of the current state of SERVOnet/CANbus. Each module will be queried as to their status and this will be displayed as text messages, for example:

```
1: CQ↵
ServoNet Status Report
======================
Module  TX map  RX map  Trace  Status
   1       1       2      0    CANbus is working ok
   2       2       1      4    CANbus is working ok
   3       0       1      0    CANbus is working ok
Bus load 42.4% (3 map txns 4 trace txns)
Other module(s) found: 59
Machine Controller: CANbus is working ok
PTS is not a SynchroLink module
```

As well as the CANbus status of each axis module the usage of mailboxes for map TX and RX and trace data is displayed. CQ also searches the CANbus network for PTS modules that are not currently being used by the QManager/Machine Manager and will report their existence as a further line of text, giving their module numbers. The following error states can be encountered and reported by CQ:

- If the SERVOnet error line is in the error state this will be reported
- If the QManager/Machine Manager is unable to communicate over CANbus it will not report the status of any Axis Modules
- An axis module can have three error states reported in CQ; a) the axis module has 'some errors' on CANbus which implies that there have been some problems but is still functioning as an Axis Module, b) the Axis Module is alive but not communicating with the QManager/Machine Manager which implies that the Axis Module has suffered a power failure and c) the Axis Module is neither alive or communicating which means that its power is off.

 Copyright ©2004 Quin Systems Ltd.

- For any module the CANbus status can be working OK, ‘some errors’ or bus off (applicable to QManager/Machine Manager only). The error source given is specific to CANbus. For the vast majority of the time any given module should be ‘working OK’, the ‘some errors’ state should be transient and is normally related to CANbus (or other) power failure. If a system is regularly experiencing both errors and bus off due to errors the CANbus wiring should be inspected: paying attention to screening of electrically noisy equipment.

NOTE: If the QManager/Machine Manager fails to find any Axis Modules when it is switched on the CQ display will be given - this gives a network diagnosis to help solve any SERVOnet problems.

5.2 SI CAN module Number

SI allows module number configuration of an entire SERVOnet network from the QManager/Machine Manager. Care should be taken when using this function to obey the rules set in section 4.1 on page 14. (Version 1.9.2 and earlier used CN command here).

5.3 LK master map LinK over CANbus

The LK function exists on SERVOnet to allow SynchroLink and LinMot operations to be performed. See section A.5.4 on page 42 for more details.

5.4 ML Map Link

The ML command has two alternative syntaxes: ML<channel> for use on SERVOnet and ML<node>:<channel> for use with SynchroLink. See section A.5.5 on page 42 for more details.

5.5 NL differeNtial Link

The NL command has two alternative syntaxes: NL<channel> for use on SERVOnet and NL<node>:<channel> for use with SynchroLink. See section A.5.6 on page 43 for more details.

5.6 RN Reset module Number

RN is used to recover an Axis Module that has suffered a power failure without restarting the whole SERVOnet. As such it is a specialised function and requires careful implementation from a programming perspective. It is not seen as an operational feature but as a maintenance feature.

Syntax: RN<module> where <module> is between 1 and 60.

RN performs error checking to determine if the module is alive (able to use CANbus to communicate) and to decide whether the module is already communicating with the QManager/Machine Manager. These tests take a few seconds during which the message ‘Working’ is displayed. If the target module is alive but not communicating with the QManager/Machine Manager then RN attempts to begin communication with the module. If this is successful the axis parameters stored in the module are reloaded. This gives the message ‘Restoring channels 1 to 4 .... OK’ or similar. Once this process is complete the module is ready for use. During this process the RN function also sends a CANopen NMT wakeup command to any CANopen device with the same node number as the Axis Module in question. This is to recover any CANopen encoder that might have been used with the Axis Module.

RN does not restore the current state of any user program. Sequences, Maps and Profiles will need downloading when necessary by means of the CM/TM/TP commands, any variables storing channel level parameters will need resetting, as will any DI line definitions that were modified by the program.

Notes:

- RN cannot be used to add channels to a SERVOnet system: it is for recovering channels lost due to module power failure.
- RN will not work if Axis Module suffered power failure during the CM command (or similar). To properly implement RN ensure that the PTS system is not currently accessing the channels in question before turning the Axis module off, performing maintenance, and restarting it.

Syntax: RN61.

This special case of RN is used to send a global CANopen NMT wakeup command to any CANopen device on the CANbus. This is provided so that the PTS can recover any CANopen devices that have failed, without requiring the PTS system to be re-booted.

5.7 XN eXecute sequence on remote Node

Syntax: XN<seq. no.> where <seq. no.> is between 1 and 65535 inclusive.

This function is used to execute a PTS sequence on a number of SynchroLink nodes simultaneously. A SERVOnet (QManager/Machine Manager + n Axis Modules) forms one SynchroLink node as far as XN is concerned. Issuing XN from the QManager/Machine Manager will have no affect on any Axis Module. See section A.5.7 on page 43 for more information on XN on SynchroLink.

For LinMot linear motors connected to SERVOnet the XN command and PTS variable $XN form a means of issuing command/response communications between a PTS and a LinMot system. Further documentation of this is provided in the LinMot manual.

 Copyright ©2004 Quin Systems Ltd.

5.8 FW Fault options Word

From PTS V2.2.1.1 and higher FW Set fault options word allows advanced control over the SERVOnet error line:

Bit	Meaning, when set
0	Don't abort on motor error if ME defined
1	Don't abort on user error if UE defined
2	**Don't force motor off if Servonet error line detected**
3	**Don't force motor off if another Servonet error has occurred**
4	**Trip Servonet error line if a motor error occurs**

Table 4: FW Fault word option bit definitions

The FW command is channel specific; i.e. it needs defining on the PTS motor channel for which the modified behaviour is required.

The following table describes how FW affects the behaviour of the SERVOnet error line:

Fault	Default behaviour	Modified behaviour using FW
Error line goes into error state (various causes: power failure or another axis module fails)	Motor off: the PTS automatically goes into MO state	FW bit 2; if set: the motor continues to be enabled. This should be used if stopping the motor is more dangerous than continuing
SERVOnet power fails (12V dc failure)	Motor off: the PTS automatically goes into MO state	FW bit 3; if set: the motor continues to be enabled. This should be used if stopping the motor is more dangerous than continuing
SERVOnet protocol failure (message corruption)	Motor off: the PTS automatically goes into MO state	FW bit 3; if set: the motor continues to be enabled. This should be used if stopping the motor is more dangerous than continuing
Motor error occurs (such as position error, i.e. SE exceeded)	Motor stops. ME routine triggered	FW bit 4; if set: the motor error will cause the SERVOnet error line to trigger - making ALL other PTS channels respond (by motor off or not, depending on FW bit 2 on individual channel)

Table 5: FW fault option word actions and behaviour

This is an advanced facility to provide user control over the SERVOnet error line and its behaviour, see chapter 7. on page 24 for more details about SERVOnet error handling.

NOTE: FW bit 3 is potentially very dangerous. There is no guarantee that the PTS system will be able to communicate after either the 12V power fails or a communications corruption occurs. Therefore it may not be possible to *command* the motor to stop via the PTS language (as the QManager/Machine Manager cannot communicate with the axis module(s)).

5.9 SERVOnet node table (LN, NN, NT, NQ)

See chapter 8. on page 28 for detail of these commands.

 Copyright ©2004 Quin Systems Ltd.

6. PTS Program Writing for SERVOnet

Once SERVOnet has been installed and configured a machine control program can be written in much the same manner as for any other PTS system. However some attention needs to be paid to the coding to allow for SERVOnet, the following points have been noted:

- Use NC function to determine if all modules have started successfully (NC should give the expected number of channels). This is useful diagnostic information for the operator.
- Use CM and TM/TP in your program to transfer sequences and maps/profiles before using them (the XS and XM/XP commands). This will reduce processor/bus load at critical points.
- CQ can be queried into a variable i.e. $CQ=CQ, where 0 means CANbus OK, a non-zero value indicates an error, the number being a PTS error code (e.g. $CQ=141 means there is an error state on the hardware error line). Use this feature to produce information for machine diagnostics.
- After using the RN command certain variables may need resetting: especially if you are using variables to hold information about channel level parameters such as position or loop counting. It is also advisable to use CM/TM/TP as required after an RN to download sequences/maps/profiles to reduce the processor/bus load when restarting the machine.
- As with the PTS Mk2 you should be careful with I/O line definitions and other functions that are module specific (e.g. AO). Remember that input group 1 on one Axis module is not the same as input group 1 on another module (and the same applies for output groups and other functions). So keep a note of which PTS channels are on which Axis modules.
- Channel specific parameters are stored on the Axis Module - to save memory on the QManager/Machine Manager. This means if an axis module is exchanged or moved its parameters will go with it. To avoid this have all configuration parameters in sequences (stored in the QManager/Machine Manager) - which means 'hot-swapping' configurations can be constructed.

If any other points arise that would be useful please contact Quin Systems who will add them to the next release of this manual.

Copyright ©2004 Quin Systems Ltd.

7. SERVOnet Error Detection and Handling

SERVOnet has been designed to provide robust error detection and handling of CANbus and hardware failures. Due to the design and implementation of a SERVOnet system individual modules may well be powered from different sources, in differing electrical conditions and exposed to different mechanical risks. This contrasts with the PTS Mk2 which is a rack based system driven from a single power supply and is situated in a single place. This chapter deals with the possible errors and how they are detected and reported.

Two distinct implementations of SERVOnet are possible, the ‘fail safe’ system or the ‘hot swappable’ system.

- Fail safe means that any failure should automatically stop all motors
- Hot swappable means it is possible to remove one Axis Module from the SERVOnet for maintenance/replacement without stopping motors attached to other Axis Modules

From PTS V2.2.1.1 and higher advanced control over the error line and how PTS responds to it is provided by the FW command. See section 7.4 on page 26 for full details.

7.1 Error Detection in Fail Safe mode

In fail safe mode all SERVOnet errors are considered potentially serious - the only sensible response is to stop all motors. SERVOnet errors consist of CANbus protocol failures, CANbus power failures and module power failures. The means of transmitting these error states between modules is via the hardware error line and software messaging/detection. Therefore errors are detected very quickly and acted upon, whether or not an error is received by the PTS program.

7.2 Hot Swapping - Servicing a working machine

To enable hot swapping of Axis Modules the SERVOnet cable must be wired differently, as covered in chapter 3. on page 8. Once this has been done it is possible to remove an axis module without stopping all motors, by following this procedure:

i Use the PTS program to ensure that the machine is in the correct state for removing the Axis Module

ii Remove the SERVOnet ‘T’ socket from the front of the Axis Module.

iii Turn the power off on the Axis Module.

iv Perform maintenance/replacement as necessary

v Turn the power on to the Axis Module

vi Reconnect the SERVOnet ‘T’ socket to the Axis Module

vii Issue an RN function from the PTS program

viii Use the CN function to set the module number of the replacement Axis Module

 Copyright ©2004 Quin Systems Ltd.

If this order is not followed you are likely to cause a SERVOnet hardware error and cause all motors to stop.

7.3 What happens when a SERVOnet error occurs?

To describe what happens when a SERVOnet error occurs use the following two tables. The first table describes what will happen to the module local to the failure, the second table indicates what will happen to all the other modules on the SERVOnet as the result of this failure:

Error Source	If module was a:	
	Axis Module	**QManager/Machine Manager**
CANbus protocol failure	Motor off all axes on module, automatically attempt to regain CANbus as soon as possible. When bus regained send PTS error 128 to QManager/Machine Manager.	PTS error 128 generated, automatically attempt to regain CANbus as soon as possible.
CANbus power failure	Motor off all axes on module, automatically attempt to regain CANbus as soon as possible. When bus regained send PTS error 128 to QManager/Machine Manager.	PTS error 127 generated, automatically attempt to regain CANbus as soon as possible.
Module power failure	Motor off all axes on module. Module will not regain CANbus when power returns. QManager/Machine Manager can restart affected module using RN command	SERVOnet needs restarting, which will happen automatically when power is restored to QManager/Machine Manager.

Table 6: Effect of SERVOnet error on local module

Copyright ©2004 Quin Systems Ltd.

Error	Affect on other Axis Modules
Failure of an Axis Module	All motors stop [by means of the hardware error line]. The axis module will then wait for new instructions from the Machine Manager.
Failure of the QManager/Machine Manager	All motors stop [by means of the hardware error line]. The axis module will then wait for new instructions from the QManager/Machine Manager.

Table 7: Effect of SERVOnet error on another module

7.4 Advanced SERVOnet error handling using FW

The default behaviour of the PTS system to the SERVOnet error line and other error conditions can be modified by the use of the FW command. This is available from PTS version 2.2.1.1 and higher.

Care should be taken when modifying the default behaviour as there is no guarantee that it will be possible to *command* the axis module after a SERVOnet fault - in other words you may not be able to stop a motor without an EStop.

See section 5.8 on page 21 for detail on the FW command bits. This section provides two scenarios to demonstrate how FW can be used effectively:

7.4.1 Using FW on a multi-axis machine that mechanically interlocks

Consider a machine where the product and mechanism form a ‘gear train’ which is driven by many servo motors. One servo motor is the master, and the other servo motors are map slaves using the ML and XM commands of PTS.

If a slave motor should fail it is imperative that the machine stops quickly, and in sychronisation.

- To achieve this set FW bits 2 and 4 on all slave motors (leave the master with FW bits 2 and 4 unset)

If the master fails it will stop and all synchronised slaves will also come to a standstill via mapping.

If any (one) slave fails it will create a SERVOnet error which the master will respond to by stopping; and all (other) slave motors will also stop in synchronisation.

Motor error routines will operate for these failures and can be used to make the machine safe after the stop (e.g. issue global GF command).

 Copyright ©2004 Quin Systems Ltd.

7.4.2 Using FW on a multi-axis machine with significant product momentum

Consider a machine where the product has a significant mass and achieves a significant momentum through the process. In this case simply stopping a motor (via MO) would potentially cause damage - the product would back drive the motor/mechanics.

- Consider the use of PTS parameter EA
- FW bit 2 will allow the motor to continue after a SERVOnet error
- FW bit 3 will allow the motor to continue after a SERVOnet power or protocol error

The motor error/user error routines in PTS can be used to slowly ramp the machine down after an error has occurred.

NOTE: if FW bit 3 is used it may not be possible to communicate with the axis module to stop the motor; an EStop would be required.

Copyright ©2004 Quin Systems Ltd.

8. SERVOnet node table

8.1 Node table principles

When a SERVOnet system is switched on the QManager/Machine Manager waits 6/15 seconds and then scans the network to find axis module(s), using the scheme detailed in section 4.1 on page 14.

By default a SERVOnet system scans the network and *dynamically* builds the module (node) table from what is detected. This provides an efficient solution but has two potential pitfalls:

- What if an axis module fails to start? The SERVOnet will not have all it's required PTS channels - the NC command should be used to check for this situation.
- What if part of the machine is deliberately missing? A truly modular machine will have optional components. The *dynamic* module detection system will not allow for this.

Using the SERVOnet node table it is possible to pre-load the module configuration into the QManager/Machine Manager so that instead of building the table dynamically it compares what is found on the network with the table and reacts accordingly.

To build a node table a number of steps need to be performed:

- Decide how many modules there will be
- Decide for each module how many PTS channels it will have (e.g. whether the virtual channel is turned off or on)
- Decide for each module whether it needs to exist for the machine to operate (required) or whether it is optional
- Code this information into PTS via the node table commands, as explained below
- Add error checking in the PTS autostart to detect any failures during SERVOnet boot up (e.g. what is found on the network doesn't match the table)
- Add user code to report which PTS channels/axis modules exist to the operator (fault finding and status feedback)
- Add user code to handle optional modules being removed or added

The remainder of this chapter details the PTS commands used to perform these actions.

8.2 PTS commands for node table

These commands are also detailed in the PTS Reference manual (MAN533) issue 18 or higher, and in the PTS online help system for PTS Toolkit 2000.

8.2.1 LN List Node table

This command provides a textual list of the current status for the SERVOnet node table.

 Copyright ©2004 Quin Systems Ltd.

```
1: ln
Table is validated
Node  Type      State    Chans  First
  1   Optional  Asleep    2      1
  2   Required  Alive     2      3
1:
```

The first line of the display shows whether the node table is operational or not:

- "not validated" - this is a dynamic node table built by scanning the network. No NN or NT commands exist (in NVM) and SP100 will have no effect.
- "validated" - this is a node table built from NN and NT commands. These have previously been saved by SP100 and can be listed by LA100.

The node listing is a tabular form of NN, NT and NQ commands.

- "Type" - the NT value for the node
- "State" - the NQ value for the node
- "Chans" - the NN value for the node
- "First" - calculated first channel number for each node

8.2.2 NN Number of channels on Node

This sets the number of PTS channels on any given node. For example, for a QDrive this will be 1 or 2, depending upon whether the virtual channel is disabled or enabled.

Syntax NN<node>/<channels>; NN<node> may be used to query the setting of any node.

```
1: nn1/2        # Set node 1 to have 2 channels
1: nn1          # query node 1
NN2             # Note: response doesn't have node number
1: nn1/1        # Set node 1 to have 1 channel
1: nn1          # query
NN1             # response
```

8.2.3 NT Node Type

This sets whether any given node is optional or required. An optional node can be missing from the network and the QManager/Machine Manager will simply ignore commands to the PTS channels on that node; a required node should never be missing.

Syntax NT<node>/<type> where type is 0 for optional and 1 for required; NT<node> may be used to query the setting of any node.

```
1: nt2/0        # Set node 2 to be optional
1: nt2          # query node 2
NT0             # Note: response doesn't have node number
1: nt2/1        # Set node 2 to be required
1: nt2          # query
NT1             # response
```

Copyright ©2004 Quin Systems Ltd.

8.2.4 SP100 Save Node Table

To save the node table into the PTS use bit 2 of SP. To reset a node table to default (clear, no table) use RS bit 2 (e.g. RS100/SP100 to clear completely). Note: power cycle the whole SERVOnet system after saving the node table to check it is correctly configured.

8.2.5 NQ Query Node status

This has two uses: it can be used to tell if any given module (node) is present on the SERVOnet, or to tell the status of the node table itself.

NQ<node>: e.g. NQ1<return>. This will return -1, 0 or 1. Use $NQ=NQ1 or similar PTS code, and interpret the return value according to the following table:

NQ<node> returns	Meaning
-1	Node is not present on the network. Node was required and so shouldn't be missing. PTS user program will not function correctly - any commands sent to the channels on this node will fail with "device not ready" errors
0	Node is not present on the network. Node was optional and is asleep (PTS commands can be sent to the channels on this node but they won't do anything)
1	Node is alive and communicating. PTS instructions, such as PC, can be sent to the PTS channels on this node

Table 8: NQ return values (axis module)

Copyright ©2004 Quin Systems Ltd.

The second form of NQ queries the state of the node table. This is done by issuing the NQ command without a parameter e.g. NQ<return> or $NQ=NQ.

NQ returns	Meaning
-1	Node table not validated. This indicates a node table that has been *dynamically* determined from the network - no node table has been saved in the PTS
0	The node table doesn't match the network. There can be two reasons for this: a module has a different number of channels to the node table entry and/or a required module is missing. Use NQ<node> to find out which node(s) are not matching the node table correctly
1	The node table matches what was found on the network. N.B. This doesn't tell you whether an optional module is alive or asleep; use NQ<node> to determine this

Table 9: NQ return values (node table)

Copyright ©2004 Quin Systems Ltd.

9. Error Messages and Codes

Additional error codes have been added to the PTS language for SERVOnet.

Error	Description	Comments
126	CAN chip stuck in reset	The QManager/Machine Manager has a hardware problem: contact your supplier or Quin Systems Ltd.
127	No CAN network transceiver power	All motors will stop. Check CAN cabling and power supply.
128	This module is CANbus OFF	Reported when QManager/Machine Manager module goes off bus due to CANbus errors. All motors will stop (FW default). CANbus errors are probably caused by electrical noise on the CANbus cable.
129	Unable to log onto CANbus	Either there is insufficient bus idle for CAN chip to synchronise with other bus traffic or the wiring is faulty.
130	No free CAN mailbox	All eight mailboxes (used for mapping and trace functions) are in use: no more SERVOnet map links available, or no mailboxes are available for trace data; the axis module is fully occupied already.
131	No CAN mailbox assigned	ML, TR, DM or UL function failed. Contact Quin Systems.
133	No Clock signal being received	Axis modules are not receiving the clock signal that should be being broadcast from the QManager/Machine Manager. This error normally occurs after the QManager/Machine Manager has been off bus (error 128). All motors will stop (FW default)
136	Module *n* lost RX data on CANbus	A module is having problems receiving CANbus messages; this can happen when the bus load and processor load are both extremely high. A solution is to rewrite sequences to use TM for instance to reduce loading.
137	No more map links available	All the CAN mailboxes have been used or the maximum number (8) of map transmitters are being used - CQ will confirm this for you.

Table 10: SERVOnet Error codes and descriptions

 Copyright ©2004 Quin Systems Ltd.

Error	Description	Comments
138	MODULE *n* has been CANbus OFF	Axis module *n* has been off CANbus due to bus errors. To receive this message communication has been re-established; frequent errors like this imply CANbus wiring problems.
139	Module *n* is not responding	Axis module *n* is not responding to CANbus messages: check its power and cabling.
140	Module *n* has suffered power failure	This error code will be received when an Axis Module is switched back on after being switched off. Use for recovery from 'Hot swapping'
141	Module *n*: SERVOnet hardware detected	When the SERVOnet hardware error line enters the error state all motors stop. Each module that detects this will also report this error code.
142	RN: Module *n* has wrong number of channels	The RN function checks that the Axis Module you are trying to restart has the same number of channels as when it was first detected during startup. This error indicates that either the Axis Module is the wrong type (a MiniPTS 3 SERVOnet rather than a Q-Drive SERVOnet) or the virtual channel is enabled/disabled when it shouldn't be.

Table 10: SERVOnet Error codes and descriptions

10. SERVOnet LED Displays

10.1 MiniPTS 3 SERVOnet Axis Module, Mini Machine Manager and QManager

The MiniPTS 3 SERVOnet Axis Modules and Mini Machine Manager have three 7-segment LED displays mounted on the front panel. The QManager has four 7-segment LED displays mounted on the front panel. These are used to display SERVOnet/CANbus status as well as motor/error status:

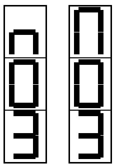
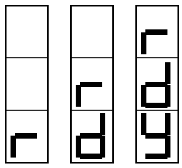
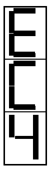
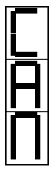
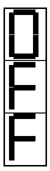
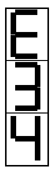
Axis Module	QManager/ Machine Manager
 Axis module is module 3 (n=temporary; N=NVM)	 indicates QManager/ Machine Manager is initialising
 Indicates a SERVOnet related error. See section 10.1.1 on page 35	 Indicates QManager/ Machine Manager is working OK
 Axis module is OFF due to module number clash	 Indicates PTS error number 34 (errors above 99 just display as a number)

Table 11: SERVOnet LED Displays for MiniPTS 3/QManager

For normal operation an axis module will display a number per channel indicating the current status of that channel or an error code (Exx) for a PTS error such as motor position error; the machine manager will display 'CAN'.

 Copyright ©2004 Quin Systems Ltd.

10.1.1 Axis Module SERVOnet LED error codes

The MiniPTS 3 SERVOnet axis module displays SERVOnet related errors on the 7-segment displays, which correspond to PTS error codes (but which might not be reported as the error will mean that the CANbus communications are not working properly)

LED Display	Description
EC0	Hardware fault. This is similar to PTS error 126 - contact your supplier or Quin Systems Ltd.
EC1	No CANbus power. PTS error 127. Check CAN cabling and power supply.
EC2	CANbus Error line in Error state. PTS error 141. All motors will stop. Check CAN cabling and power supply.
EC3	CANbus off due to bus errors. PTS error 128. All motors will stop. This is caused by electrical noise interfering with the CANbus messages.
EC4	No synchronisation clock being received. PTS error 133. All motors will stop. This means that the Machine Manager is not communicating on CANbus.

Table 12: MiniPTS 3 SERVOnet Axis Module LED error codes

NOTE: The local serial port on the Axis Module will output text in these error cases. This is intended for diagnostics of SERVOnet errors when the communications between the Machine Manager and the Axis Modules is unreliable.

10.2 Q-Drive SERVOnet Axis Module

The Q-Drive SERVOnet axis module has a single 7-segment LED display. This will display 'E' for errors (including SERVOnet errors), other display codes indicate the current motor status. When the Axis Module has been started but not talked to by the Machine Manager the LED will display 'Nxx' or 'nxx' (where xx = module number) by cycling the display every 0.5 seconds.

NOTE: The local serial port on the Axis Module will output text in the case of SERVOnet specific errors, in a similar manner to the MiniPTS 3 SERVOnet axis module. This is intended for diagnostics of SERVOnet errors when the communications between the QManager/Machine Manager and the Axis Modules is unreliable.

PTSQ5xxx has two 7 segment LED displays, one for each PTS channel. When displaying errors an E is displayed, and the node number is cycled in the manner as described above using only a single LED display.

10.3 Machine Manager

The machine manager has power status LEDs only.

Appendix A. SynchroLink

A.1 Introduction

SynchroLink is the ability for PTS systems to perform map linking between separate PTS units (nodes). This is an advance on the use of a link encoder to daisy chain PTS units together. A high speed communications bus, CANbus, has been used for SynchroLink, providing real time map linking for multiple master/slave combinations. Currently SynchroLink is supported on the MiniPTS 3, MiniPTS 1+1, Q-Drive 1+1 (receive only), Q-Drive MAP and PTS Mk2.

SynchroLink is a subset of SERVOnet, a distributed Master/Slave configuration of PTS units utilising CANbus as the communication medium.

A.2 Features of SynchroLink

SynchroLink supports the following features:

- Up to 60 nodes on one SynchroLink network.
- Network length maximum 100m.
- Up to 8 map masters transmitting simultaneously across CANbus.
- Many map slaves can receive one map master signal (see next point).
- Each node can support up to 8 unique map masters/slaves (more than one channel can receive the same map slave information within a node, whilst only requiring one unique map slave reception from CANbus).
- Global sequence execute for use during motor error sequences or similar.
- Requires only a handful of extra PTS instructions to configure and use SynchroLink.
- Node number clash, clock signal configuration and network error detection provided.

A.3 Wiring Requirements for SynchroLink

SynchroLink may be wired in an identical manner to SERVOnet, as described in chapter 3. on page 8, and must be so wired if a network is shared between SERVOnet and SynchroLink. However, for purely SynchroLink or CANopen nodes, it may be wired as follows:

 Copyright ©2004 Quin Systems Ltd.

Each PTS node has a male 9 pin sub-D type connector for CANbus. The cable should therefore have female 9 pin sub-D type connectors. The CANbus cable joins each node in turn (in the same manner as ethernet), with no cable spurs. Twisted pair cable is recommended. A 12V DC power supply is required to provide bus power, at about 30mA per node (all PTS nodes are opto-isolated from the bus). Each end of the bus should be terminated with a 120Ω resistor placed between CAN H and CAN L.

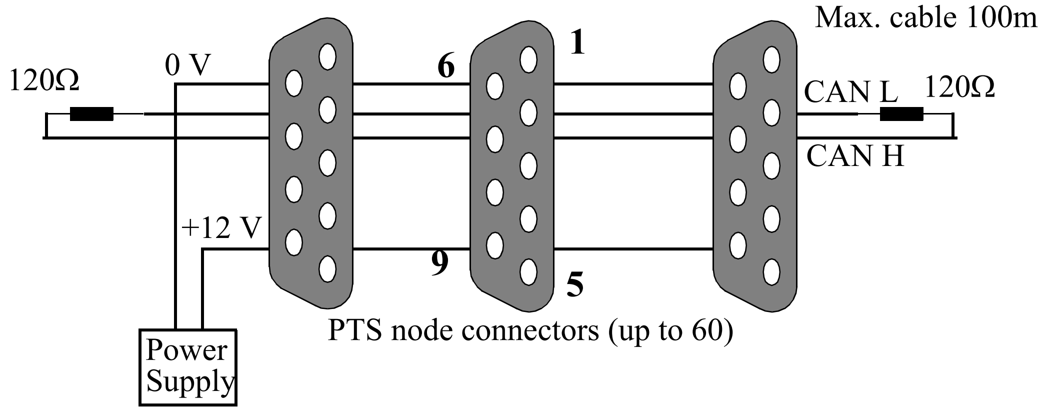


FIG.7: CANBUS WIRING

A LinMot controller uses a female 9 pin sub-D type connector which should be wired as below:

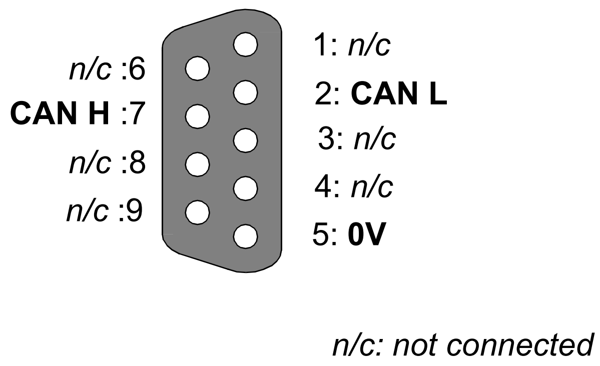


FIG.8: LINMOT CONNECTOR PINOUT

A.4 SynchroLink between two or more SERVOnets

On the QManager/Machine Manager it is possible to have a SynchroLink network on the second CAN port (upper connectors) as well as a SERVOnet network on the first CAN port (lower connectors). This means that a number of SERVOnet systems can be linked together with SynchroLink to increase flexibility and extend the physical size of the whole system. The data rate of the SynchroLink network can be reduced if required to increase the maximum distance between units although this also reduces the maximum number of map masters which can be accommodated.

In systems with two CAN ports the QManager/Machine Manager acts as a bridge between SERVOnet and SynchroLink. An axis module on one SERVOnet system can be map linked to a master axis on a different SERVOnet system with the SynchroLink network carrying the map data from one SERVOnet system to the other. The map data from the master axis is picked up by the QManager/Machine Manager which then broadcasts the data over SynchroLink. The data is then picked up from SynchroLink by the second system where the QManager/Machine Manager broadcasts the map data over SERVOnet to the map slave module.

This type of system can operate with any type of SynchroLink system including the PTS Mark 2, Mini-PTS 3, Q-Drive Map, etc. CANOpen I/O and encoders are supported as normal on the SERVOnet network.

The operation of the two networks is very similar to before. However some changes have been necessary to allow the two physical networks to co-exist as described below.

A.4.1 Enabling the Second CAN Port

To enable the second CAN port you must install a jumper link on the CPU360 and enter a software key. The link is fitted between pins 11 and 12 (the bottom two pins) of jumper J11 which is located just behind the second CAN port. To enable the software key, use the SK command to enter the feature and version as follows:

```
Feature: synchro2
Version: 0
```

After changing SK the system must be powered off and back on for the change to take effect. When the second CAN port is enabled DeviceNet is disabled since it also uses the second CAN port. If the second CAN port is not enabled the system still allows SynchroLink nodes to be placed on the SERVOnet network and accessed as map slaves, map masters or using XN.

A.4.2 SI & CN Commands

The SI command (SERVOnet node Id) has now replaced the CN command for SERVOnet networks. SI is used to set the SERVOnet axis module node numbers and CN is used to set the SynchroLink node number as before. CN is only available on systems having a separate SynchroLink network. The CN command is used both to set the SynchroLink node number and to start and stop SynchroLink and is also saved in non-volatile memory.

A.4.3 ML and NL Commands

On a QManager/Machine Manager using the second CAN port the ML (map link) and NL (differential link) commands use the node:channel syntax to link to an axis on SynchroLink where node is the remote SynchroLink node number and channel is the channel number on the remote system. The commands use the channel number only to link to an axis on the SERVOnet.

 Copyright ©2004 Quin Systems Ltd.

On a SynchroLink network the LK command is used on the master axis to enable map transmission. The receiving axis needs to be set up to match the master axis with MP, NP and MM commands.

A.4.3.1 CF Command

On the QManager/Machine Manager and PTS Mk2 the CF command now allows you to set the data rate on port C, the second CAN port. The data rate can be set to 125, 250 or 500 kbps. To change the data rate, enter the CF command, select port C and type in the new data rate. The system power must be cycled for the change to take effect. As the data rate drops the loading on the network grows since the time taken to transmit each message increases. This means that the maximum number of map masters possible reduces with the data rate. The maximum network length and number of map masters for the different data rates is shown in the following table.

Data Rate	Network Length	Map Masters
125 kbps	500 metres	2
250 kbps	250 metres	4
500 kbps	100 metres	8

Table 13: CF option C settings for SynchroLink

A.4.3.2 CQ Command

The output from the CQ command for SynchroLink has been modified to be similar to the SERVOnet report. It includes a table of the modules found with the number of map transmissions from each. The bus load calculation has also been improved. The binary parameter is unchanged.

When the second CAN port is enabled the CQ command shows information on both the SERVOnet and SynchroLink networks. The SERVOnet report now includes a bus load estimate similar to SynchroLink.

A.4.3.3 SERVOnet Node Number Limit

The highest node number for a SERVOnet axis module has been reduced from 60 to 59. This is to allow the QManager/Machine Manager to use node number 60 when it is transmitting map data arriving from a SynchroLink node.

Copyright ©2004 Quin Systems Ltd.

A.4.4 SERVOnet Clock

Normally the SERVOnet clock is generated by the QManager/Machine Manager. When the system also has SynchroLink on the second CAN port, the SERVOnet clock is locked to the SynchroLink clock master to ensure that all nodes on both networks are synchronized. This means that if the SynchroLink clock is lost for any reason, the SERVOnet networks will also lose their clock and any mapped axis modules will stop due to map time-out. The exception to this is at power up before the SynchroLink network has been started by the SI command. In this case the QManager/Machine Manager will generate the SERVOnet clock as normal. Once the SynchroLink network has been started the SERVOnet clock relies on it as described above.

The SynchroLink node chosen to provide the clock should be the source of most map data and receiver of least, to give the maximum time for data transmission. Note also that there is a trade-off between trace data mailboxes within SERVOnet, map data transferring to SynchroLink and other host activity. The SynchroLink clock is enabled with the CK command.

A.5 Additional/Modified PTS Instructions for SynchroLink

A.5.1 CK<val> Configure ClocK

Value 0 (default) or 1

SynchroLink requires clock synchronisation between nodes for position mapping to work. Therefore one node must be a clock *master* (CK1) and all other nodes must be clock *slaves* (CK0). The clock signal uses up approximately 3.5% of the available CANbus bandwidth.

When bus off (CN0) changing CK has no effect. The value of CK is applied when CN is set to a non-zero number and the node attempts to go CANbus on. If another clock master is detected on the bus then an error message (error 132) will be displayed and CK will revert to zero.

When bus on (CN non-zero) changing CK will have immediate effect. CK1 will attempt to make the node a clock master. If no other node is clock master then this will be successful, otherwise an error message (error 132) will be printed. CK0 will turn the node into a clock slave.

$ABC=CK will place either a zero or one in variable $ABC, to indicate clock slave and clock master respectively.

NOTE: there must always be a clock master for SynchroLink to work. Clock slave nodes check for the presence of a clock master and will not perform mapping across CANbus unless the clock signal is present (error 133 followed by map update timeout errors (error 100) will be reported if clock signal is not available). Therefore always configure one node to be the SynchroLink clock master. Use the CQ function to check for the presence of a clock signal. {A SERVOnet QManager/Machine Manager is a clock master by definition, so CK must not be used if SERVOnet is present on CANbus}.

 Copyright ©2004 Quin Systems Ltd.

A.5.2 CN<node> Configure Node

Value 0 (default) to 60 inclusive

To assign a node number and to go on bus for SynchroLink use CN<n> where <n> is the node number to be used. Setting CN0 switches SynchroLink off for this node. Setting CN<n> will start SynchroLink. If another node already uses <n> as its node number an error will be reported (error 134) and the node will go bus off (CN will also revert to zero). $ABC=CN will place the current node number (0 to 60) in the variable $ABC.

If at any time the node goes bus off (loss of CANbus power or loss of CANbus cable integrity) each affected node will report an error (error 128) and try to go back on bus immediately. If that fails the node will continue to try and regain the bus, waiting 1 minute between each attempt. No error messages will be displayed for success or failure of these attempts. The CQ function can be used to query the status of a node at any time.

There is no requirement for nodes to be numbered consecutively, or to start at node number 1; a node numbering scheme that suits the application should be implemented, and SERVOnet node numbers must not be used.

A.5.3 CQ CANbus status Query

Value none or binary code flags

To query the SynchroLink status of the node use CQ. CQ<return> displays the current status of the node. This information includes the node number being used, whether the node is clock master, the status of the CANbus interface and which node is clock master if the current node isn't.

The CQ function can display extra diagnostic information to aid CANbus configuration:

Bit	Meaning
0	Display this nodes bus usage
1	Find all nodes on CANbus
2	Display global bus load
3	Reserved
4	Reserved
5	Reserved
6	Reserved
7	Reserved

Bit 0 displays the number of unique map links being transmitted and received by this node. A maximum of eight transmissions/receptions is available.

Bit 1 counts all the nodes responding on CANbus. For a node to respond it must have been configured with CN non-zero, have successfully initialised its CANbus interface and be bus on (receiving power to the transceivers, and the CANbus cable be sound).

Bit 2 gives an indication of the percentage bus load on CANbus due to map links. Each master map link across CANbus occupies approximately 5% of the available bandwidth, with the clock signal occupying a further 3.5%. The figure presented is calculated by querying each node for the number of map transmissions.

A.5.4 LK<val> master map LinK over CANbus

Value 0 (default) or 1

For map linking across CANbus a map master is required. This is created by setting LK1 on the desired channel. Each master map link over CANbus uses up approximately 5% of the available bandwidth [CANbus map slaves do not use bus bandwidth as they receive messages only]. Testing has found that up to 8 map masters will be supported by CANbus (equivalent to 40% bus loading). The number of map slaves is limited by the CANbus interface on each node (maximum 8 unique CANbus map masters/slaves per node).

$ABC=LK will place zero or one in variable $ABC indicating map master off and map master on respectively.

A.5.5 ML<node:channel> Map Link

Value: node=1 to 60 inclusive, channel=1 to 48 inclusive

The ML command has been extended for SynchroLink. Using a node:channel format it is possible to specify that the required map master is on another node. [The standard form of ML<channel> still functions and performs a map link within the node] The existence of this master is not checked, however map timeout errors will occur if the master is not transmitting data. Any one node will support up to 8 CANbus map masters/slaves.

In addition to ML, two other functions must be called to complete the map link on the map slave axis, above the normal requirements:

Master axis, channel 5 on node 3:

CK1/CN3/CH5/SB20000/AR256/LK1/ZC/PC

Slave axis, channel 8 on node 5:

CK0/CN5/CH8/MP20000/MM256/SB1000/ML3:5/PC/XM56

In the above example channel 5 on node 3 is to be the map master for channel 8 on node 5. First node 3 is configured (here it is also the clock master). The master axis is then initialised by setting the bounds (SB) and the analogue range (AR, optional) before exporting the map data on CANbus using LK1. Secondly node 5 is configured. Channel 8 requires knowledge of the master bound (MP, using the SB value entered on channel 5, node 3), the master analogue range (MM, using the AR value entered on channel 5, node 3 [optional]) and its own bound (SB). The map link is established by ML3:5, and then the axis put into mapping using map table 56.

$ABC=ML will place a value of 256*node + channel in variable $ABC. In this manner it is possible to query and program ML using variables. For example to set ML to node 3 channel 5 the following are valid: ML3:5, ML773, $ABC=773/ML=$ABC.

 Copyright ©2004 Quin Systems Ltd.

A.5.6 NL<node:channel> differeNtial Link

Value: node=1 to 60 inclusive, channel=1 to 48 inclusive

The NL command has been extended for SynchroLink. Using a node:channel format it is possible to specify that the required map master is on another node. [The standard form of NL<channel> still functions and performs a map link within the node] The existence of this master is not checked, however map timeout errors will occur if the master is not transmitting data. Any one node will support up to 8 CANbus map masters/slaves.

Usage of NL follows the same format as ML.

A.5.7 XN<n> eXecute sequence on all other Nodes

Value 1 to 65535 inclusive

This function provides the ability for one node to request the execution of the said sequence number on all other nodes. This is equivalent to typing XS<n> on terminals connected to the serial ports of all other nodes and pressing all the enter keys at the same time. XN<n> does NOT perform an XS<n> on the source node. XN is a global broadcast with no confirmation from receiving nodes, and therefore no guarantee of reception (and execution of sequence <n> on remote node) can be given.

If sequence <n> does not exist on a receiving node no error is generated, the request is instead written into PTS variable $XN. XN<n> will generate errors if two or more XN's are received or transmitted before the firmware has finished operating the first XN function.

XN<n> is primarily intended for use in motor error sequences. If one channel on a node experiences a motor error the XN function should be used to inform all other nodes and perform a controlled shutdown of the plant, for example the XN could call the motor error sequence on other nodes. Other uses of XN are not recommended, due to the lack of acknowledgment between transmitter and receiver.

For LinMot linear motors connected to SynchroLink the XN command forms a means of issuing command/response communications between a PTS and a LinMot system. Further documentation of this is provided in the LinMot manual.

Copyright ©2004 Quin Systems Ltd.

A.6 Error Messages and Codes

Eleven additional error codes have been added to the PTS language for SynchroLink.

Error	Description	Comments
125	No CAN chip fitted to board	SynchroLink functionality requires a CAN interface IC to be fitted to the circuit board
126	CAN chip stuck in reset	Hardware fault on circuit board, contact Quin Systems
127	No CAN network transceiver power	Check CAN cabling
128	This node is CANbus OFF	Reported when node goes off bus due to errors or when already bus off and a command (such as LK) is issued
129	Unable to log onto CANbus	Either there is insufficient bus idle for CAN chip to synchronise with other bus traffic or the wiring is faulty
130	No free CAN mailbox	All eight mailboxes are in use: no more SynchroLink map links available
131	No CAN mailbox assigned	ML,LK or UL function failed
132	Node <n> is already clock master	Only one clock master is required
133	No Clock signal being received	Node is not receiving the clock signal that should be being broadcast from one node
134	Another node already uses our node number	Attempt to go bus on with CN<n> failed as <n> is already used, try another number
135	XN(s) lost on CANbus	Over ten XN's arrived before they could be processed

Table 14: CANbus Error codes and descriptions

 Copyright ©2004 Quin Systems Ltd.

Appendix B. CANopen Devices

B.1 Introduction

This document contains information for using CANopen devices (absolute encoders and/or additional digital and analogue I/O) on SERVOnet and SynchroLink. This covers hardware configuration, software configuration and PTS programme usage.

This appendix is split into four sections;

- B.1 to B.5 is general information for all CANopen devices supported
- B.6 is specific information for CANopen absolute encoders
- B.7 is specific information for CANopen IO
- B.8 is programming information for SERVOnet and SynchroLink

B.2 Summary Table

PTS Product	CANopen devices available
SynchroLink Q-Drive, MiniPTS 1+1 & MiniPTS 3	***CANopen I/O:*** 64 digital inputs *or* 4 16bit analogue inputs *WITH* 64 digital outputs *or* 4 16bit analogue outputs ***OR*** 1 Absolute Encoder
SERVOnet (Mini) Machine Manager *PLUS* axis modules (of any sort)	***CANopen I/O:*** 64 digital inputs *or* 4 16bit analogue inputs *WITH* 64 digital outputs *or* 4 16bit analogue outputs ***PLUS*** Absolute Encoder (1 per Axis Module)

Table 15: Availability of CANopen on PTS systems

B.3 Important notes on CANopen compatibility

These notes form the basic rules for CANopen devices: if the CANopen device obeys these rules it will work with SynchroLink and SERVOnet.

1 General description of the type of CANopen: *A CANopen slave which supports the minimum CANopen bootup with node guarding and a pre-defined connection set.*

2 The CANbus hardware should be CAN specification V2.0B passive.

3 CANopen encoders must adhere to CiA DS406.

4 CANopen I/O must adhere to CiA DS401.

5 Require NO configuration from PTS systems during normal usage, with the exception of NMT bootup, as mentioned in point 1. The CANopen device must therefore store its own configuration data.

6 Module numbering of the CANopen devices will need to use the same number as the PTS unit it needs to talk to; i.e. 1 - 60 for Absolute Encoders, and 1 - 60, or 61 for External I/O (depending upon PTS system).

7 No CANopen error handling will be provided by PTS systems. All other CANopen issues, such as node guarding, dynamic allocation of messages etc. will be ignored.

Further notes regarding the impact of using CANopen devices on SynchroLink and SERVOnet:

8 Each Absolute encoder used will reduce the available CANbus bandwidth by the same amount as one SynchroLink/SERVOnet map transmission. Quin recommends 8 map transmissions as a sensible maximum bus load. Therefore there will be a trade off between Absolute encoders and PTS map sources, and a top limit of 8 Absolute encoders.

9 Extra Digital and Analogue inputs will also occupy CANbus bandwidth. The bus load will depend upon the number of events per second. The top limit of this will equate to one SynchroLink/SERVOnet map transmission; therefore the same rules as stated in the last point apply.

B.4 CANopen devices supported by PTS

B.4.1 CANopen absolute encoder

To facilitate absolute position measurement on Q-Drive systems the CANopen absolute encoder can be used. For completeness this option is available across all CAN enabled PTS products.

CiA DS406 is the specification required for a CANopen encoder, see section 4.

 Copyright ©2004 Quin Systems Ltd.

B.4.2 CANopen I/O

CANopen I/O adds additional digital and/or analogue I/O to a PTS system quickly and cheaply, without resorting to a PLC and other such external logic. Within PTS there is sufficient power to handle I/O tasks (such as opening/closing air valves) via digital I/O as part of a machine cycle.

This additional I/O does not support referencing, position triggers, or other channel-specific functions, and is slower than channel level I/O. CANopen I/O is intended for machine process control rather than machine motion control.

CiA DS401 is the specification required for CANopen I/O devices, see section 4. Digital I/O is supported in groups of 8, i.e. bytes of data. Analogue I/O is supported as 16bit conversions (2 byte data fields).

B.5 CANopen Devices Tested with SynchroLink and SERVOnet

- Fraba Absolute encoder. {usage details given}
- Beckhoff BK51xx series – digital and analogue I/O. {usage details given}
- Weidmuller CAN Bridge – 1 digital I/O module only.

B.5.1 CANbus Connection

Most CANopen devices give their own isolated power to their interface circuits; and most units make no provision for connection of the 9 standard CANbus leads. A simple 3-wire connection from the non-powered SERVOnet/SynchroLink terminator is generally recommended, with the 120Ω bus termination installed at the CANopen unit:

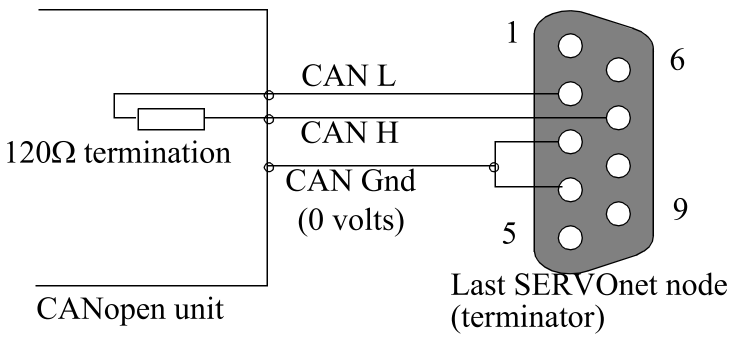


FIG.9: CANOPEN UNIT WIRING

For process and power connections refer to the CANopen module data sheets.

B.6 CANopen Encoder

B.6.1 Hardware configuration

The encoder needs configuring to transmit its data in a certain manner {specific details for Fraba encoder included here}:

1 Baud rate of CANbus for SERVOnet and SynchroLink is 500kBit/s.
FRABA: This means that the following DIP switches in the encoder should be set: DIP 8 ON, DIP 7 OFF and DIP 6 ON.

2 Node number for encoder should be the same as the Quin unit requiring the encoder data (CN setting on SynchroLink, Axis Module number on SERVOnet).
FRABA: This is set on DIP switches 1 to 5 as per encoder user manual section '6.2 Settings in the Connection Cap' (**Node number = 1 + binary switches**).

B.6.2 CANopen configuration

The encoder CANopen PDO configuration must be as follows:

1 Synchronous transmissions every sync. message on PDO2, with no inhibit time.

2 PDO1 must be suppressed.

FRABA: refer to section '7.2.7 Cyclic Mode' in the encoder user manual, specifically switching off cyclic mode transmissions. NOTE THAT THIS REQUIRES A SOFTWARE MOD TO ENCODER. If this mod has not been implemented set cyclic transmit period to 65535 (maximum). Also refer to section '7.2.8 Sync Mode' in the encoder user manual.

Once the configuration has been performed remember to save these settings in the encoder NVM.

CANopen objects and sub-indexes not mentioned here should be left at their factory defaults. It is quite possible to configure the encoder is such a way that position data is not sent to PTS correctly, so avoid doing any further configuration unless confident of the outcome. Other settings such as encoder direction, resolution and limit switches will be application specific (Index 2100).

B.6.3 PTS Configuration

A CANopen encoder is configured by the FS command. This will check that CANopen I/O is not being used on systems where one or the other will work but not both {Q-Drive, MiniPTS 1+1, MiniPTS 3}.

A CANopen clock signal is required for the encoder to transmit its data: this is manually enabled on SynchroLink using the CK command, but is fully automatic on SERVOnet.

 Copyright ©2004 Quin Systems Ltd.

B.6.4 Using a CANopen absolute encoder

The CANopen encoder is enabled/disabled by the use of the FS command [FS option 9 & 10]. The number of data bits from the encoder can be set using the same commands as used for an SSI absolute encoder (NB and NZ).

If the CANopen encoder fails to transmit data PTS error 102 will be generated.

The IN command works on the CANopen absolute encoder in the same manner as it works on an SSI absolute encoder.

Copyright ©2004 Quin Systems Ltd.

B.7 CANopen I/O

B.7.1 Hardware configuration

The CANopen I/O needs configuring to transmit its data in a certain manner {specific details for Beckhoff device included here}:

1 Baud rate of CANbus for SERVOnet and SynchroLink is 500kBit/s.
BECKHOFF: This means that the following DIP switches should be set:
DIP 7 ON and DIP 8 OFF.

2 The node number of the CANopen I/O device should be 61 for SERVOnet or to the CN number for SynchroLink.
BECKHOFF: DIP switches 1 to 6 inclusive set the node number in a binary format e.g. 61 = 111101 (switch order 654321).

B.7.2 PTS Configuration

A serial key (feature **canopen** version **0**) enables CANopen I/O. Once the serial key is present at power-on, the CANopen I/O configuration and host level I/O PTS commands will function, and FS9/10 will not {Q-Drive, MiniPTS 1+1, MiniPTS 3}. Once CANopen I/O is enabled it will be automatically detected and configured.

B.7.3 The CANopen configuration shell: PTS function QC

The PTS command **QC** allows manual configuration of CANopen I/O and also provides features to send and receive of CANopen messages for one time configuration and diagnostics of the CANopen device. During using this command PTS cannot perform motor control and all motors will remain in the motor off state. Upon entering this command the PTS prompt is replaced by CAN> and a new set of commands are available:

Configure Select CANopen I/O settings. Configure will automatically detect the CANopen I/O module & report what is found. This function is not available if the CANopen I/O license key does not exist.

Exit Exit this configuration tool ("**Quit**" also works).

Help Displays a simple listing of the available commands ("**?**" also works).

NMT Sends NMT message. You will be queried for the number of data bytes, and the value for each data byte before the message is sent. No transmit acknowledgement will be generated.

Node Choose CANopen node number to talk to (1 to 127 inclusive, but 1 to 61 only relevant for Quin products). This setting affects the send and receive commands, but not configure, which uses the node number (61 for SERVOnet, as CN for SynchroLink).

Query Displays CANopen configuration and CANbus diagnostics (similar to CQ command) including CAN state, bus load from synchronous transmissions and PTS units found on the network.

 Copyright ©2004 Quin Systems Ltd.

Receive Receive SDO (tx) messages from chosen node. This turns on (or off) the displaying of CANopen SDO messages from the chosen node. SDO messages will be interpreted and displayed giving R/W byte, index, sub-index and 4 data bytes, as per CANopen syntax.

Send Send an SDO (rx) message. This will be sent to the currently selected node. You will be prompted for the R/W flag, Index, sub index and D1, D2, D3 and D4 data bytes, as per CANopen syntax. No transmit acknowledgement will be generated. {won't work at the same time as FS9 or FS10 on Q-Drive, MiniPTS 1+1 and MiniPTS 3}

All commands are case insensitive, ignore spaces and can be shortened to their first character (except NMT and Quit).

B.7.4 CANopen configuration

The CANopen I/O PDO configuration must be as follows:

1 PDO1(tx) (digital inputs) asynchronous with inhibit time used to create 'debounce' for noisy inputs.
BECKHOFF: The default configuration is fine except for noisy inputs (need to change Index 1800, sub-index 03 in this case, and then save settings).

2 PDO2(tx) (analogue inputs) should be configured to transmit data when the analogue signal changes by a certain amount (event with threshold, use an inhibit time as well if system is noisy). In this manner the bus load will be restricted, and PTS trigger variables will not be continuously firing. If the analogue input can only be set to transmit synchronously then as low a bandwidth as possible should be selected, and PTS trigger variables avoided.

BECKHOFF: Analogue inputs can be configured by following this procedure, using the CANopen configuration shell (section B.7.3 on page 50):

a. Disable SynchroLink clock (CK0) [this is automatic on SERVOnet].
b. Set node switches on bus coupler as required and connect to CANbus etc.
c. Enter the configuration shell (QC) on the PTS system.
d. Send NMT node reset, for which the parameters are: length 2, data byte 1=81 hex, data byte 2=<node number> [which is 61 decimal, 3D hex, for SERVOnet].
e. Send SDO to reset Beckhoff unit, for which the parameters are listed in the Table on page 56.
f. Set PDO2 transmit to event, this is SDO index 1801, sub-index 2, data item 1 to FF (again see the Table on page 56).
g. Set Index 6423, sub-index 0 to 1 (specify event driven analogue inputs). This is done by sending the appropriate SDO command listed in the table.
h. For ALL analogue inputs set event threshold (Index 6424 and 6425 or 6426, sub-indexes 1 to number of analogue inputs). Index 6426 is the more common setup (set to 0F 00 say). Again use the table for reference.
i. Set inhibit time if required (Index 1801 sub-index 3).
j. Send the Store parameters SDO (Index 1010 sub-index 1). If this is successful there will be an SDO acknowledgement after a few seconds.
k. Exit the configuration shell, restart CK1 if applicable, and test using PTS trigger variables on $EI1 etc.

3 PDO1(rx) is used to receive digital output data, and PDO2(rx) is used to receive analogue output data.

Once the configuration has been performed remember to save these settings in the CANopen device NVM.

CANopen objects and sub-indexes not mentioned here should be left at their factory defaults. It is quite possible to configure the CANopen I/O is such a way that data is not sent to PTS correctly, so avoid doing any further configuration unless confident of the outcome.

B.7.4.1 Sending Service Data Objects (SDO)

The CANopen configuration shell allows you to query and configure a CANopen device using the SDO system. Typing ‘S’ or ‘Send’ will prompt you for a number of parameters used to build an SDO message. This will be transmitted from the PTS and received by a CANopen module (target node number as set by the ‘N’ or ‘Node’ command). A response will be generated by the CANopen module which will be displayed by the PTS if ‘R’ or ‘Receive’ is set to on.

A CANopen SDO message has the following 8 bytes of data:

Byte:	1	2	3	4	5	6	7	8
Meaning:	R/W	Index		sub-ind	D1	D2	D3	D4

Entering the values to construct the SDO message is done using hexadecimal.

Read/Write Byte

Choose from the following table:

Value (hex)	Meaning
40	Read data (length independent, D1..D4 ignored)
22	Write 4 data bytes (D1..D4)
2A	Write 2 data bytes (D1 & D2)
2E	Write 1 data byte (D1)

Index

This is the object dictionary index that you wish to read/write. A four-figure number in hexadecimal should be entered here. This value is required for the SDO transmit to work.

Sub-Index

Each entry in the object dictionary has a number of sub-indexes, enter the appropriate value here (in hexadecimal). This value is required for the SDO transmit to work.

 Copyright ©2004 Quin Systems Ltd.

Data bytes 1 to 4

All four data bytes need values, in hexadecimal. The default setting of zero is provided for use when the data byte is not required by CANopen object.

B.7.4.2 Received SDO Data from CANopen module.

The received SDO data from a CANopen device is displayed as read/write code, index, sub-index and four data bytes:

Byte:	1	2	3	4	5	6	7	8
Meaning:	R/W	Index		sub-ind	D1	D2	D3	D4

The values displayed in the SDO message use hexadecimal notation.

Read/Write Byte

A response to a read request from the PTS gives one of the following codes:

Value (hex)	Meaning
43	Read 4 data bytes (D1..D4 contain data)
4B	Read 2 data bytes (D1 & D2 contain data)
4F	Read 1 data byte (D1 contains data)
60	Confirm write (D1..D4 irrelevant)
80	Error in SDO sent from PTS, see following section

Note that data bytes not containing valid data will possibly contain non-zero values, carried over from a previous command - these values should be ignored.

Copyright ©2004 Quin Systems Ltd.

B.7.4.3 SDO Data Bytes for error report from CANopen device

The four data bytes should be interpreted using the following tables.

Byte:	1	2	3	4	5	6	7	8
Meaning:	80	Index		Sub-ind	EE1	EE2	ECD	ECL

EE1	Additional information
0x	**00** No precise details for the reason for the error
1x	*Service parameter with an invalid value* **11** sub index doesn't exist **12** length of parameter too high **13** length of parameter too low
2x	*Service cannot currently be executed* **21** because of local control **22** because of present device state
3x	*Value range of parameter exceeded* **31** value written too high **32** value written too low **36** max. less than min.
4x	*Incompatibility with other values* **41** data cannot be mapped to PDO **42** PDO length exceeded **43** general reason **47** general internal

ECD Error code	ECL Error class
03 inconsistent parameter **04** illegal parameter	**05** Service Error
01 unsupported **02** object doesn't exist **06** hardware fault **07** type conflict **09** attributes inconsistent	**06** Access Error
00 (always)	**08** Other Error

 Copyright ©2004 Quin Systems Ltd.

TITLE	R/W	INDEX	SUB INDEX	VALUE & RANGE D1	D2	D3	D4	ADDITIONAL INFORMATION
Device Type	**40 /**	**1000**	**00**	**91**	**01**	**RR**	XX	RR = bit code for I/O present
PDO1 (tx) ID	**40 / 22**	**1800**	**01**	**80 + n**	**01**	**00**	**00**	N = node number
PDO1 (tx) Type	**40 / 2E**	**1800**	**02**	**FF**	XX	XX	XX	FF = asynchronous events
PDO1 (tx) Inhibit time	**40 / 2A**	**1800**	**03**	**00..FF**	**00..FF**	XX	XX	16 bit value (lowest byte first)
PDO2 (tx) Type	**40 / 2E**	**1801**	**02**	**00..F0, FF**	XX	XX	XX	0..F0 = synchronous at n sync. msgs; FF = asynchronous
PDO2 (tx) Inhibit time	**40 / 2A**	**1801**	**03**	**00..FF**	**00..FF**	XX	XX	16 bit value (lowest byte first)
N Digital Inputs	**40 /**	**6000**	**00**	**RR**	XX	XX	XX	RR = number of 8 bit groups
Read Digital Inputs	**40 /**	**6000**	**01..08**	**RR**	XX	XX	XX	Read 8 bits (by 8 bit group)
N Digital Outputs	**40 /**	**6200**	**00**	**RR**	XX	XX	XX	RR = number of 8 bit groups
Write Digital Outputs	**/ 2E**	**6200**	**01..08**	**00..FF**	XX	XX	XX	Write 8 bits (by 8 bit group)
N Analogue Inputs	**40 /**	**6401**	**00**	**RR**	XX	XX	XX	RR = number of analogue inputs
Read Analogue Inputs	**40 /**	**6401**	**01..04**	**RR**	**RR**	XX	XX	16 bit value (lowest byte first)
N Analogue Outputs	**40 /**	**6411**	**00**	**RR**	XX	XX	XX	RR = number of analogue outputs
Write Analogue Outputs	**/ 2A**	**6411**	**01..04**	**00..FF**	**00..FF**	XX	XX	16 bit value (lowest byte first)

TITLE	R/W	INDEX	SUB INDEX	VALUE & RANGE D1	D2	D3	D4	ADDITIONAL INFORMATION
Reset to defaults	**/ 22**	**1011**	**01**	**6C**	**6F**	**61**	**64**	ASCII 'load' to reset to defaults
PDO2 (tx) Type	**40 / 2E**	**1801**	**02**	**00..F0, FF**	XX	XX	XX	0..F0 = synchronous at n sync. msgs; FF = asynchronous
PDO2 (tx) Inhibit time	**40 / 2A**	**1801**	**03**	**00..FF**	**00..FF**	XX	XX	16 bit value (lowest byte first)
Analogue Input TX type	**40 / 2E**	**6423**	**00**	**0..1**	XX	XX	XX	0 means RTR (default) 1 means event driven
High limit trigger	**40 / 2A**	**6424**	**01..04**	**00..FF**	**00..FF**	XX	XX	Event generated when input exceeds set level (lowest byte first)
Low limit trigger	**40 / 2A**	**6425**	**01..04**	**00..FF**	**00..FF**	XX	XX	Event generated when input drops below set level (lowest byte first)
Input change trigger	**40 / 2A**	**6426**	**01..04**	**00..FF**	**00..FF**	XX	XX	Event generated when input changes by more than set level (lowest byte first)
Store Parameters	**/ 22**	**1010**	**01**	**73**	**61**	**76**	**65**	ASCII 'save' to store setup

Table 17: CANopen Object Dictionary for I/O (configuration of Analogue Inputs)

Copyright ©2004 Quin Systems Ltd.

B.7.5 Using CANopen I/O

The CANopen interface in the PTS will initialise after the autostart sequence has been triggered (assuming licence key ‘canopen’ exists). When a CANopen I/O module is successfully detected and configured a banner will be displayed. If detection is unsuccessful it will be repeated as a background task every 5 seconds, until success. Once detection has taken place the initial value of the inputs and outputs will be updated, ensuring that the PTS and CANopen device have matching data. The CANopen I/O is then ready for use.

CANopen I/O operates in two different modes, depending upon whether the I/O is digital or analogue. Digital I/O will appear as input and output groups 10 to 17 inclusive, i.e. host level digital I/O. Additionally the digital outputs (first 32 bits) can be addressed using variables. Analogue I/O will use variables to read/store/write the voltage settings from and to the CANopen device.

B.7.5.1 Digital Inputs and Outputs

CANopen digital I/O will appear as I/O groups 10 to 17 inclusive (depending upon the number of physical I/O present on the CANopen device). A subset of PTS I/O commands is supported:

SOn:x	COn:x	IOn:x	ROn:x	LOn			
DIn:x		IIn:x	RIn:x	LIn	MIn:x	BIn:x	EIn:x

There is one limitation: variables cannot be used to specify I/O groups 10 to 17. For example SO$SO will always refer to channel level I/O, whatever the value of $SO.

To provide the ability to set bytes of data onto digital output variable access is provided for the first 4 bytes (32 bits) of digital output, using $EO1..$EO4 (letter e and o, number 1 to 4). Writing these variables causes the digital outputs to be updated. If you need to read the current value of the digital inputs use $EI1=RI10: for example.

The following I/O functions are not supported for CANopen I/O: MG, BG, PO, OC, OX, DX, WI and all timer counter functions, nor channel-specific indications.

B.7.5.2 Analogue Inputs

When the CANopen device transmits a state change of the analogue inputs PTS variables $EI1 to $EI4 (letter e and i, number 1 to 4) will be updated with this information. Each of these four variables will contain 16 bits of information, relating to a analogue input (if the input exists). The value in each variable will be the ‘raw’ data transmitted from the CANopen device; PTS has no knowledge of any scale factors. Bi-polar inputs may need the sign-bit propagating, for example: $II1=(($EI1<<16)>>16)

Use a PTS trigger variable to respond to changes in analogue inputs, but only if the update frequency of the analogue input is low enough. It is sufficient to define a single trigger variable for all analogue inputs as the update event from any one analogue input causes all the relevant PTS variables to be updated.

B.7.5.3 Analogue Outputs

Write out the desired level of the CANopen analogue outputs using variables $EO1 to $EO4 (letter e and o, number 1 to 4). The scale factor of these variables is dependent upon the CANopen hardware; PTS has no knowledge of this. Writing to any one of these variables will cause the analogue output to be updated automatically. These variables will be initialised to zero when the CANopen interface starts.

 Copyright ©2004 Quin Systems Ltd.

B.8 Implementation Details for SynchroLink/SERVOnet

B.8.1 Implementation Details specific to SynchroLink

For either CANopen encoder or CANopen I/O to operate the SynchroLink system needs turning on using the CK and CN PTS commands. CQ can be used to query the state of CANbus (but will not return information about the state of any CANopen modules). A CANopen NMT wakeup command is sent whenever a nonzero CN command is issued – the node number of the CANopen device to wakeup is the same as the CN parameter value.

CK, CN and FS configuration are all stored in NVM. Therefore for the system to work properly these values should be saved correctly. During PTS boot the parameters are restored, which will start the SynchroLink features (CK & CN), which will issue an NMT wakeup, and then start receiving CANopen encoder data if so configured (FS9/10). After the autostart sequence is triggered the CANopen I/O will start (if enabled). This will determine the present state of the CANopen module. Therefore any CANopen device should be powered up before, or at the same time, as the PTS system.

If it is necessary to re-establish communications with a CANopen device (e.g. after device power failure) use PTS command string $CN=CN/CN0/CN$CN or similar. This will re-issue the CANopen NMT wakeup command. Any SynchroLink mapping will be temporarily interrupted and might cause PTS error messages.

B.8.2 Implementation Details specific to SERVOnet

CANbus is an integral part of SERVOnet so the necessary facilities for CANopen encoders and I/O will be initialised during PTS boot-up. Just before parameters are restored the QManager/Machine Manager will transmit a global NMT wakeup message. FS9/10 will be restored as parameters and will immediately start receiving CANopen encoder data. After the autostart sequence is triggered the CANopen I/O will be initialised if it is enabled. Therefore CANopen devices should be powered up before, or at the same time, as the QManager/Machine Manager.

If it is necessary to re-establish communications with a CANopen device (e.g. after device power failure) use PTS command RN61 which will transmit a CANopen NMT wakeup command.

Copyright ©2004 Quin Systems Ltd.

 Copyright ©2004 Quin Systems Ltd.

Index

T

V

W

X

 Copyright ©2004 Quin Systems Ltd.