



# 6513

## RS485/RS232 Communications Interface

HA466357U001 Issue 7  
Technical Manual

aerospace  
climate control  
electromechanical  
filtration  
fluid & gas handling  
hydraulics  
pneumatics  
process control  
sealing & shielding





---

# RS485/RS232 Communications Interface

Technical Manual  
HA466357U001 Issue 7

Compatible with Version 5.x Software

2011 Parker SSD Drives, a division of Parker Hannifin Ltd.

All rights strictly reserved. No part of this document may be stored in a retrieval system, or transmitted in any form or by any means to persons not employed by a Parker SSD Drives company without written permission from Parker SSD Drives, a division of Parker Hannifin Ltd. Although every effort has been taken to ensure the accuracy of this document it may be necessary, without notice, to make amendments or correct omissions. Parker SSD Drives cannot accept responsibility for damage, injury, or expenses resulting therefrom.

## WARRANTY

Parker SSD Drives warrants the goods against defects in design, materials and workmanship for the period of 24 months from the date of manufacture, or 12 months from the date of delivery (whichever is the longer period), on the terms detailed in Parker SSD Drives Standard Conditions of Sale IA500504.

Parker SSD Drives reserves the right to change the content and product specification without notice.

**FAILURE OR IMPROPER SELECTION OR IMPROPER USE OF THE PRODUCTS DESCRIBED HEREIN OR RELATED ITEMS CAN CAUSE DEATH, PERSONAL INJURY AND PROPERTY DAMAGE.**

This document and other information from Parker-Hannifin Corporation, its subsidiaries and authorized distributors provide product or system options for further investigation by users having technical expertise.

The user, through its own analysis and testing, is solely responsible for making the final selection of the system and components and assuring that all performance, endurance, maintenance, safety and warning requirements of the application are met. The user must analyze all aspects of the application, follow applicable industry standards, and follow the information concerning the product in the current product catalog and in any other materials provided from Parker or its subsidiaries or authorized distributors.

To the extent that Parker or its subsidiaries or authorized distributors provide component or system options based upon data or specifications provided by the user, the user is responsible for determining that such data and specifications are suitable and sufficient for all applications and reasonably foreseeable uses of the components or systems.

# Safety Information



## Requirements

**IMPORTANT:** Please read this information BEFORE installing the equipment.

### Intended Users

This manual is to be made available to all persons who are required to install, configure or service equipment described herein, or any other associated operation.

The information given is intended to highlight safety issues, EMC considerations, and to enable the user to obtain maximum benefit from the equipment.

Complete the following table for future reference detailing how the unit is to be installed and used.

INSTALLATION DETAILS	
<b>Model Number</b> <i>(see product label)</i>	
<b>Where installed</b> <i>(for your own information)</i>	
<b>Unit used as a:</b> <i>(refer to Certification for the Inverter)</i>	<input type="checkbox"/> Component <input type="checkbox"/> Relevant Apparatus
<b>Unit fitted:</b>	<input type="checkbox"/> Wall-mounted <input type="checkbox"/> Enclosure




### Application Area

The equipment described is intended for industrial motor speed control utilising DC motors, AC induction or AC synchronous machines

### Personnel

Installation, operation and maintenance of the equipment should be carried out by qualified personnel. A qualified person is someone who is technically competent and familiar with all safety information and established safety practices; with the installation process, operation and maintenance of this equipment; and with all the hazards involved.

### Product Warnings

	<b>Caution</b> Risk of electric shock		<b>Caution</b> Refer to documentation		<b>Earth/Ground</b> Protective Conductor Terminal
---	--	---	--	---	--

# Safety Information



## Hazards

### **DANGER! - Ignoring the following may result in injury**

1. This equipment can endanger life by exposure to rotating machinery and high voltages.
2. The equipment must be permanently earthed due to the high earth leakage current, and the drive motor must be connected to an appropriate safety earth.
3. Ensure all incoming supplies are isolated before working on the equipment. Be aware that there may be more than one supply connection to the drive.
4. There may still be dangerous voltages present at power terminals (motor output, supply input phases, DC bus and the brake, where fitted) when the motor is at standstill or is stopped.
5. For measurements use only a meter to IEC 61010 (CAT III or higher). Always begin using the highest range. CAT I and CAT II meters must not be used on this product.
6. Allow at least 5 minutes for the drive's capacitors to discharge to safe voltage levels (<50V). Use the specified meter capable of measuring up to 1000V dc & ac rms to confirm that less than 50V is present between all power terminals and earth.
7. Unless otherwise stated, this product must NOT be dismantled. In the event of a fault the drive must be returned. Refer to "Routine Maintenance and Repair".

### **WARNING! - Ignoring the following may result in injury or damage to equipment**

#### **SAFETY**

Where there is conflict between EMC and Safety requirements, personnel safety shall always take precedence.

- Never perform high voltage resistance checks on the wiring without first disconnecting the drive from the circuit being tested.
- Whilst ensuring ventilation is sufficient, provide guarding and /or additional safety systems to prevent injury or damage to equipment.
- When replacing a drive in an application and before returning to use, it is essential that all user defined parameters for the product's operation are correctly installed.
- All control and signal terminals are SELV, i.e. protected by double insulation. Ensure all external wiring is rated for the highest system voltage.
- Thermal sensors contained within the motor must have at least basic insulation.
- All exposed metalwork in the Inverter is protected by basic insulation and bonded to a safety earth.
- RCDs are not recommended for use with this product but, where their use is mandatory, only Type B RCDs should be used.

#### **EMC**

- In a domestic environment this product may cause radio interference in which case supplementary mitigation measures may be required.
- This equipment contains electrostatic discharge (ESD) sensitive parts. Observe static control precautions when handling, installing and servicing this product.
- This is a product of the restricted sales distribution class according to IEC 61800-3. It is designated as "professional equipment" as defined in EN61000-3-2. Permission of the supply authority shall be obtained before connection to the low voltage supply.

### **CAUTION!**

#### **APPLICATION RISK**

- The specifications, processes and circuitry described herein are for guidance only and may need to be adapted to the user's specific application. We can not guarantee the suitability of the equipment described in this Manual for individual applications.

#### **RISK ASSESSMENT**

Under fault conditions, power loss or unintended operating conditions, the drive may not operate as intended.

In particular:

- Stored energy might not discharge to safe levels as quickly as suggested, and can still be present even though the drive appears to be switched off
- The motor's direction of rotation might not be controlled
- The motor speed might not be controlled
- The motor might be energised

A drive is a component within a drive system that may influence its operation or effects under a fault condition.

Consideration must be given to:

- Stored energy
- Supply disconnects
- Sequencing logic
- Unintended operation

# Contents

Contents

Page

## **RS485/RS232 COMMUNICATIONS INTERFACE 1**

<b>A System Overview .....</b>	<b>1</b>
Protocols.....	1
• El Bisynch ASCII/Binary .....	1
• MODBUS RTU .....	2
Product Features .....	2
Product Code.....	2
<b>Installation.....</b>	<b>3</b>
RS485/RS232 Communication Module (650V Frames 1, 2 & 3).....	3
• LED Indications.....	4
RS485 Communications Option (650V Frames C, D, E & F).....	5
Terminators .....	5
System Recommendations .....	6
• PLC/SCADA Supervisor .....	6
<b>Initial Set-up for El Bisynch ASCII.....</b>	<b>7</b>
Configuring the Drive.....	7
Configuring the PLC/SCADA Supervisor .....	10
ASCII Communications.....	11
• What Information Can I Transfer? .....	11
• How is the Information Transferred? .....	11
• Programmer's Information .....	13
• El Bisynch ASCII Message Protocol.....	14
• El Bisynch ASCII Parameter Mapping .....	15
• El Bisynch ASCII Sequence Diagrams .....	18
• Transferring Data - ASCII Example Messages .....	19
Character Definitions .....	24
Control Character Definitions .....	24
Last Error Code (EE) .....	25
<b>Initial Set-up for MODBUS RTU .....</b>	<b>26</b>
Configuring the Drive.....	26
Configuring the PLC/SCADA Supervisor .....	29
MODBUS RTU Communications .....	29
• How is the Information Transferred? .....	29
• RTU Mode of Transmission.....	30
• Cyclic Redundancy Check .....	30
• Function Codes .....	34
• Typical Transmission Line Activity.....	42
• MODBUS RTU Parameter Mapping .....	43
ASCII Table.....	46



# RS485/RS232 COMMUNICATIONS INTERFACE

## A System Overview

The RS485/RS232 Communications Interface provides a serial data port, allowing VSDs (variable speed drives) to be linked to form a network. Using a PLC/SCADA or other intelligent device, this network can be continuously controlled to provide supervision and monitoring for each VSD in the system.

With each unit under local control, the central supervisor performs only periodic setpoint updating, control sequencing and data collection.

In the system, the PLC/SCADA supervisor acts as the Master, and the VSD as the Slave.

The network of VSDs can be set-up using just one unit's MMI/Keypad, or connection to ConfigEd Lite (or other suitable PC programming tool).

### Advantages with this type of control system

1. Multi-wire analog transmission from a central programmable controller is replaced by a bussed digital system using serial data transmission over 3 wires (RS232) or differential twisted-pair wires (RS485).
2. Digital transmission is fundamentally less noise-prone than analog methods, and the accuracy of the transmitted data is unaffected by the transmission medium. The use of intelligent devices at either end of the data link allows error checking to be used. This virtually eliminates the effects of electrical noise on data integrity. It is therefore possible to issue setpoints to drives with much higher accuracy using this method.
3. The RS485 communication standard allows multiple drives to be connected to a single link which can be driven from a computer serial port. Additional drives can be readily accommodated through additional ports. The RS232 communication standard allows for a single drive to be connected to the master. Most computers are equipped with RS232 serial ports which can be easily converted to accommodate the RS485 standard. Modules are available from Parker SSD Drives to make this conversion.
4. The chosen standard and protocol are compatible with other Eurotherm Group products. Temperature controls, process controls, data loggers and drives can communicate easily with a common supervisory system.

## Protocols

### EI Bisynch ASCII/Binary

*Note: The RS485/RS232 Communications Interface supports EI Bisynch ASCII only, not Binary.*

These communications protocols come under the heading of Binary Synchronous Communications Data Link Control (BSCDLC).

This is all part of an internationally recognised ANSI standard protocol called BISYNCH (Binary Synchronous) and is known by the abbreviation x3.28.

They are widely used by manufacturers of computers, computer peripherals, and communications equipment.

EI BISYNCH, the specific form of communication used, corresponds with the following full American National Standard definition:

- ANSI Standard: x3.28, Revision: 1976
- Establishment and Termination Control Procedures Sub-category 2.5:  
*Two-way Alternate, Non-switched Multi-point with Centralised Operation & Fast Select*
- Message Transfer Control Procedure Sub-category B1:  
*Message Associated Blocking with Longitudinal Checking & Single Acknowledgement*

This is known by the abbreviation ANSI - x3.28 - 2.5 - B1.



## MODBUS RTU

The MODBUS RTU (Remote Terminal Unit) protocol is an efficient binary protocol. Each message must be transmitted in a continuous stream.

### Product Features

- Suitable for use with:  
650/650V software version 4.x onwards
- Connection using 2-wire shielded twisted pair (RS485)
- Connection using 3-wire un-shielded cable (RS232)
- Configured using Function Block inputs
- Software-selectable Baud Rate
- Software-selectable Slave Address
- Direct tag access for all parameters

### Product Code

The Parker SSD Drives' product is fully identified using an alphanumeric code which records how the product was assembled, and its various settings when despatched from the factory.

Product	Product Code when supplied with the Drive	Product Code when supplied separately
650 Frames 1, 2 & 3	Supplied separately	<b>6513-****</b> plug-in Communications Module
650V Frames 1, 2 & 3	Supplied separately	<b>6513-****</b> plug-in Communications Module
650V Frames C, D, E & F	<b>Legacy code</b> (where X is the Frame size letter) 650VX/xxxx/xxx/xxxx/xx/x/RS485/x/x/x <b>New product code</b> 650V-xxxxxxxx-xx2xxx-xx	Factory-fitted Communications Option - not supplied separately

# Installation

## WARNING!

Before installing, ensure that the drive and all wiring is electrically isolated and cannot be made "live" unintentionally by other personnel.

Wait 5 minutes after disconnecting power before working on any part of the system or removing the covers from the Drive.

## RS485/RS232 Communication Module (650V Frames 1, 2 & 3)

You can create a network of drives by linking a Master (PC/PLC) to one or more 650V drives fitted with this module.

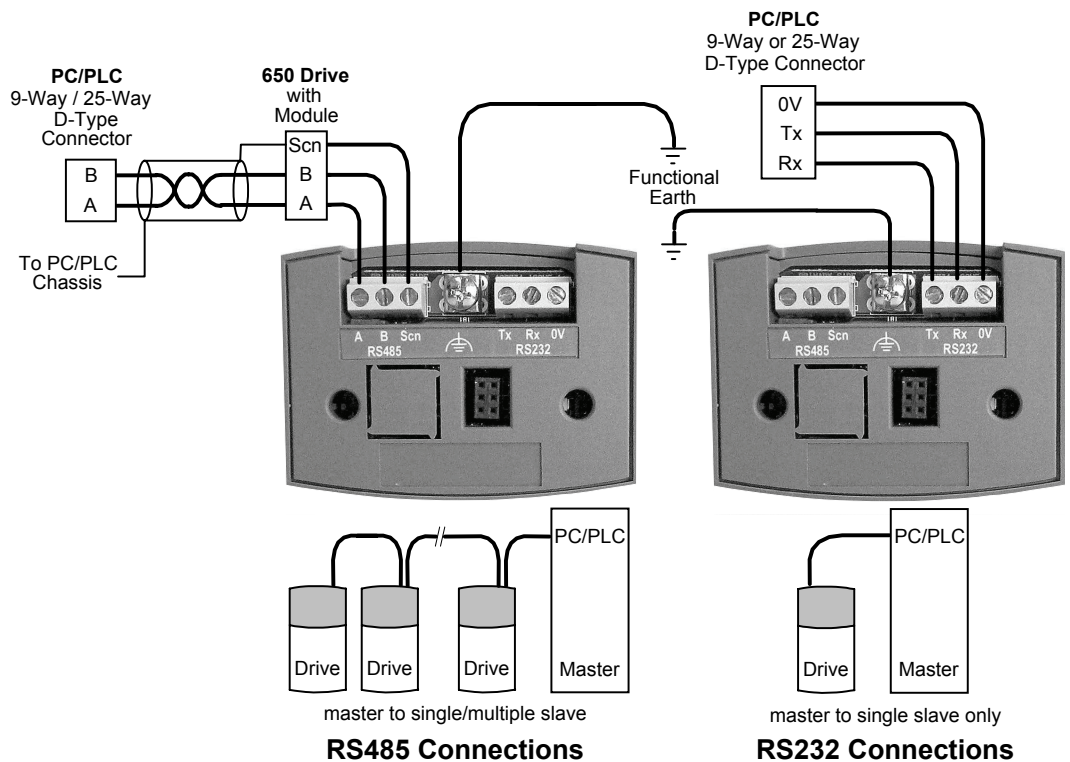
Plug this Communication Module on to the front of the 650V drive, replacing the keypad.

It converts signals from the host 650V drive into RS485 or RS232, and vice versa, so that information can be shared between the Master and 650V drive(s).

Wiring is very simple - all connections are SELV (Safe Extra Low Voltage). Select to use RS485 or RS232 by wiring to the appropriate terminal on the module.

**Note:** RS485 and RS232 terminals cannot be used simultaneously.

We recommend you ground the module to the system earth using the Functional Earth terminal.



<b>Wiring Specifications</b>		
	<b>RS485 Connections</b>	<b>RS232 Connections</b>
<b>Network Type</b>	2-Wire Shielded Twisted-Pair	3-Wire Un-Shielded Cable
<b>Connections</b>	A=RxA/TxA, B=RxB/TxB, Shield	Rx, Tx, Ground (0V)
<b>Signal Levels</b>	To RS485 Standard	To RS232 Standard
<b>Receiver Input Impedance</b>	¼ Unit Load	3 kΩ minimum 7kΩ maximum
<b>Maximum Cable Length</b>	1200m (4000ft)	3 metres
<b>Maximum Baud Rate</b>	57.6kbaud	57.6kbaud
<b>Maximum Number of Units</b>	32 including slaves and masters	2: 1 master and 1 slave only

### LED Indications

The module has three LEDs providing diagnostic information about the 650V host drive's 'Health', 'Receive' and 'Transmit' activity.

HEALTH = Green, Rx = Red, Tx =Red



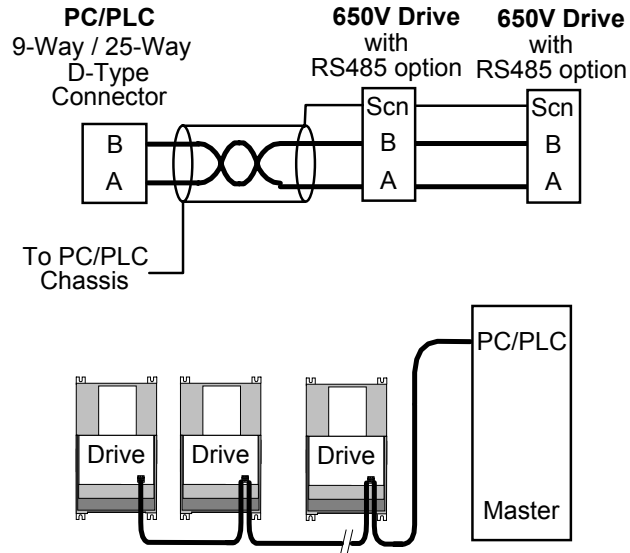
<b>LED Name</b>	<b>LED Duty</b>	<b>Drive State</b>
<b>HEALTH</b>	SHORT FLASH	Re-configuration, or corrupted non-volatile memory at power-up
	EQUAL FLASH	Tripped
	ON	Healthy
	LONG FLASH	Braking
	OFF	No drive power, or serious hardware fault
<b>Rx</b>	INTERMITTENT	Indicates activity on the 'receive' line carrying data from the Master
<b>Tx</b>	INTERMITTENT	Indicates activity on the 'transmit' line carrying data to the Master

## RS485 Communications Option (650V Frames C, D, E & F)

You can create a network of drives by linking a Master (PC/PLC) to one or more 650V drives fitted with this additional 3-way terminal. It is factory-fitted to the right hand side of the control board.

Signals from the host 650V drive are converted into RS485, and vice versa, so that information can be shared between the Master and 650V drive(s).

Wiring is very simple - all connections are SELV (Safe Extra Low Voltage).



master to single/multiple slave

### RS485 Connections

Wiring Specifications	
	RS485 Connections
Network Type	2-Wire Shielded Twisted-Pair
Connections	A=RxA/TxA, B=RxB/TxB, Scn = Screen (shield)
Signal Levels	To RS485 Standard
Receiver Input Impedance	¼ Unit Load
Maximum Cable Length	1200m (4000ft)
Maximum Baud Rate	57.6kbaud
Maximum Number of Units	32 including slaves and masters

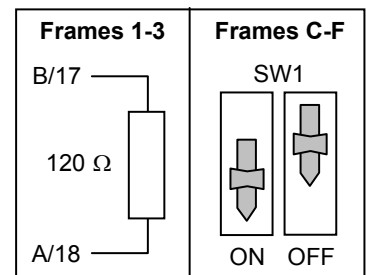
## Terminators

The last drive in a system must have a terminating resistance. All other drives in the system should not have a terminator.

Frames 1-3 drives require a 120Ω terminating resistor fitting to terminals 17 and 18 on the Control Board (resistor is ±1%, minimum ¼ Watt).

Frames C-F drives are fitted with an integral resistor, switched in and out by switch SW1 on the Control Board.

**IMPORTANT:** Failing to use a terminating resistance may result in unreliable operation.



RS485

## System Recommendations

**Note:** *It is possible to make serial communications operate without adhering to the following recommendations, however, the recommendations will promote greater reliability.*

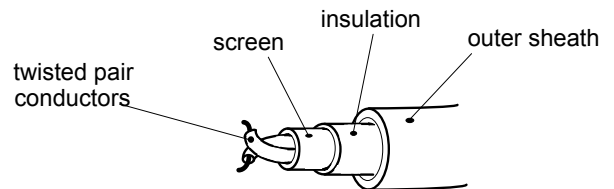
- An RS485 two-wire system can only be used in a network in which all devices use their tri-state capability. Data flow is restricted, i.e. transmit and receive cannot be simultaneous (half duplex). The driver in an RS485 system has tri-state capability (i.e. its output can be disabled) which allows multiple transmitters to be connected to the same bus. RS485 thus supports “multi-drop” operation. In multi-drop systems there is always one device which is a “Master” and which sends messages to or requests data from the “Slaves”. A Slave never initiates a communication.
- An RS232 three-wire system always has a “Master” which sends messages to or requests data from the “Slave”. The Slave never initiates a communication. There is only one “Master” and one “Slave” in the system.

### PLC/SCADA Supervisor

If possible, avoid using a PLC/SCADA supervisor which take its transmitter to a high impedance state (tri-state) when idling. If it is unavoidable, then it is essential to use properly screened cable.

### RS485 Cable Specification

Use cable which has twisted pairs and one overall screen, as shown. The characteristic impedance should be in the range 100 to 165 Ohms.



Recommended Cable Specification	
Characteristic Impedance	100-165Ω at 3-20MHz
Cable Capacitance	<30pF/m
Core Diameter	0.34mm <sup>2</sup> (22 AWG)
Cable Type	Twisted pair cable
Resistance	<110Ω/km
Shielding	Copper braid, or braid & foil

**Note:** *Belden 9841 cable meets the above specification, but there are others.*

### RS232 Cable Specification

There are no special requirements for RS232 cabling, but we do recommend a maximum length of 3 metres between Master and Slave.

# Initial Set-up for EI Bisynch ASCII

## Configuring the Drive

**Note:** The RS485/RS232 Communications Interface can only be used on drives using software version 4.1 or higher (indicated on power-up, i.e. "r4.1").

You must configure the drive to your system.

If you are using the keypad (MMI), the parameters to edit are in the SERIAL menu, <sup>S</sup>SE01 to <sup>S</sup>SE09.

If you are using ConfigEd Lite (or other suitable PC programming tool) the same parameters are contained in the COMMS PORTS and COMMS CONTROL function blocks.

*ConfigEd Lite is Parker SSD Drives' Windows-based block programming software.*

**Note:** To view all parameters available on the MMI, FULL menu detail must be selected in the DETAILED MENUS parameter (<sup>ST</sup>99): 1 = FULL.

Comms Ports	
0	[102] GROUP ID (GID)
0	[103] COMMS ADDRESS
9600	[1062] BAUD RATE
NONE	[1061] PARITY
5	[1260] REPLY DELAY
AUTOMATIC	[1060] OP PORT PROTOCOL
AUTOMATIC	[1059] P3 PORT PROTOCOL
MODBUS	[117] RS485 PROTOCOL
FALSE	[129] SWITCH OP PORT
FALSE	[90] SWAP WORD ORDER

Comms Control	
	COMMS SEQ [295] FALSE
	COMMS REF [270] FALSE
	COMMS STATUS [272] 0
	COMMS COMMAND [273] 0
FALSE	[300] REMOTE COMMS SEL
TERMINALS/COMMS	[307] REMOTE SEQ MODES
TERMINALS/COMMS	[308] REMOTE REF MODES
0.0 s	[309] COMMS TIMEOUT

## Parameter Descriptions: COMMS PORTS

This function block configures the programming ports that allow connection to the keypad, or to a personal computer.

The parameters below are used to identify the drive to the controlling software for drive configuration and storage of parameters.

**Note:** The unit will always respond to GID = 0 and UID = 0, as this is the "broadcast" address used by the keypad.

### GROUP ID (GID)

Range: 0 to 7

The Parker SSD Drives protocol group identity address.

### COMMS ADDRESS

 SET\SERL SE03

Range: 0 to 255

The Parker SSD Drives protocol unit identity address (UID) or the Modbus node address.

Note: if set to 0, it will only respond to broadcast messages.

### BAUD RATE

 SET\SERL SE04

Range: Enumerated - see below

Selects the Baud Rate for the MODBUS protocol.


Enumerated Value : Baud Rate

- 0 : 1200
- 1 : 2400
- 2 : 4800
- 3 : 7200
- 4 : 9600
- 5 : 14400
- 6 : 19200
- 7 : 38400
- 8 : 57600

**PARITY**  *SET\SERL SE05* *Range: Enumerated - see below*

Selects the Parity for the MODBUS protocol.

*Enumerated Value : Parity*  
0 : NONE  
1 : ODD  
2 : EVEN


**REPLY DELAY**  *SET\SERL SE06* *Range: 0 to 200*

The time in milliseconds between the drive receiving the complete request from the communications master (PLC/PC) and replying to this request.

**OP PORT PROTOCOL**  *SET\SERL SE07* *Range: Enumerated - see below*


Selects the protocol to be used by the keypad port on the front of the drive. When EIBISYNC ASCII is selected, BAUD RATE is 19200 and PARITY is EVEN.

*Enumerated Value : Protocol*  
0 : AUTOMATIC - checks for keypad or EI ASCII  
1 : KEYPAD  
2 : EIBISYNC ASCII  
3 : MODBUS  
4 : FIELDBUS

**P3 PORT PROTOCOL**  *SET\SERL SE08* *Range: Enumerated - see below*

Selects the protocol to be used by the RS232 programming port on the drive's control board. When EIBISYNC ASCII is selected, BAUD RATE is 19200 and PARITY is EVEN.

*Enumerated Value : Protocol*  
0 : AUTOMATIC - checks for keypad or EI ASCII  
1 : KEYPAD  
2 : EIBISYNC ASCII  
3 : MODBUS  
4 : FIELDBUS

**RS485 PORT PROTOCOL**  *SET\SERL SE09* *Range: Enumerated - see below*

**This parameter is not available on the 650V Frame 1, 2 & 3. For these frame sizes, use "OP PORT PROTOCOL" to select the protocol for the comms option.**

Selects the protocol to be used by the RS485 programming port on the drive's control board.

*Enumerated Value : Protocol*  
0 : AUTOMATIC  
1 : KEYPAD (not applicable)  
2 : EIBISYNC ASCII  
3 : MODBUS  
4 : FIELDBUS

**SWITCH OP PORT**  *SET\SERL SE10* *Range: FALSE / TRUE*

**This parameter is not available on the 650V Frame 1, 2 & 3.**

When TRUE, the keypad port on the front of the drive is disabled when the communications equipment is connected to the RS232 programming port on the drive's control board.

When FALSE, the RS485 programming port is disabled when the communications equipment is connected to the RS232 programming port. Both ports are on the drive's control board.

**SWAP WORD ORDER** *Range: FALSE / TRUE*

Controls the word order used with 32-bit data access with the Modbus RTU protocol.

## Parameter Descriptions : COMMS CONTROL

This block switches between Remote Terminal and Remote Comms operating modes.

The drive must be in Remote operating mode for selection to be made - REMOTE mode is enabled in the LOCAL CONTROL function block (REF MODES) or selected by the keypad.

**REMOTE COMMS SEL**      SET\SERL SE01      *Range: FALSE / TRUE*

Selects the type of remote communications mode:

0 : FALSE, and in REMOTE mode then control is from the terminals.

1 : TRUE, and in REMOTE mode then control is from the communications.

**REMOTE SEQ MODES**      *Range: Enumerated - see below*

Selects the type of remote sequencing mode:

*Enumerated Value : Mode*

0 : TERMINALS/COMMS

1 : TERMINALS ONLY

2 : COMMS ONLY

**REMOTE REF MODES**      *Range: Enumerated - see below*

Selects the type of remote reference mode:

*Enumerated Value : Mode*

0 : TERMINALS/COMMS

1 : TERMINALS ONLY

2 : COMMS ONLY

**COMMS TIMEOUT**      SET\SERL SE02      *Range: 0.0 to 600.0 s*

Sets the maximum time allowed between refreshing the COMMS COMMAND parameter. The drive will trip if this time is exceeded. Set the time to 0.00 seconds to disable this feature.

**COMMS SEQ**      *Range: FALSE / TRUE*

Diagnostic indicating if operating in Remote Sequencing Comms Mode.

If FALSE (0), the drive may be in Local Sequencing mode or Remote Sequencing Terminal mode.

**COMMS REF**      *Range: FALSE / TRUE*

Diagnostic indicating if operating in Remote Reference Comms Mode.

If FALSE (0), the drive may be in Local Reference mode or Remote Reference Terminal mode.

**COMMS STATUS**      *Range: 0000 to FFFF*

Diagnostic showing the 16-bit Status word as seen by the communications.

Refer to Chapter 4: "Sequencing Logic" in the 650 or 650V Software Product Manual.

**COMMS COMMAND**      *Range: 0000 to FFFF*

Diagnostic showing the 16-bit Command as written by the communications.

Refer to Chapter 4: "Sequencing Logic" in the 650 or 650V Software Product Manual.



## Configuring the PLC/SCADA Supervisor

By referring to the Parameter Specification Table in the 650 or 650V Software Product Manual, you can enter the parameter information you require.

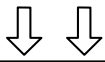
It provides the information in the following way:

### Type

The first page of the Parameter Specification Table chapter details parameter types. The Type column indicates each parameter's type.

### ID

The ID column provides the parameter mnemonic (of the tag number).



TAG	Pref	CELITE Name	MMI Name	Function Block Name	Range	Type	ID
104	33.01	V/F SHAPE	PAR 11	FLUXING	0 : LINEAR LAW 1 : FAN LAW	ENUM	2w
107	33.02	FIXED BOOST	PAR 13	FLUXING	0.00 to 25.00	REAL	2z
108	33.03	AUTO BOOST	SET\CTRL CL08	FLUXING	0.00 to 25.00	REAL	30
110	66.05	SPEED SCALE	SET\ENC EN05	ENCODER	0.00 to 300.00	REAL	32
111	66.06	SPEED	SET\ENC EN06	ENCODER	Output	REAL	33
112	53.02	BASE VOLTS		VOLTAGE CONTROL	0.00 to 115.47	REAL	34
117	18.07	RS485 PROTOCOL	SET\SERL SE08	COMMS PORTS	0 : AUTOMATIC 1 : KEYPAD 2 : EIBISYNC ASCII 3 : MODBUS 4 : FIELDBUS	ENUM	39
119	35.14	STATOR RES	SET\CTRL CL17	MOTOR DATA	0.0000 to 250.0000	REAL	3b
	35.15	LEAKAGE INDUC	SET\CTRL CL18	MOTOR DATA	0.00 to 300.00		
	16	MUTUAL INDUC			0.00 to 3000.00		

*Example only*

## ASCII Communications

**Note:** The RS485/RS232 Communications Interface supports *El Bisynch ASCII* only, not Binary.

### What Information Can I Transfer?

The data transfer sequence in the ASCII mode offers the following facilities:

- i) Parameter enquiry (known as polling)
  - a. Single Parameter Poll
  - b. Continuous Polling of a Parameter
  - c. Sequential Polling (fast polling down the parameter list)
- ii) Setting parameters (known as selection)
  - a. Single Parameter Selection
  - b. Continuous Selection of a Parameter
  - c. Sequential Selection (fast selection down the parameter list)

**Note:** For examples of all the above refer to "Transferring Data - ASCII Example Messages", page 19.

### How is the Information Transferred?

There are two types of data transfer message:

1. Reading information from the Drive
2. Writing information to the Drive

In both cases the supervisor must have an established connection with the device, which will then respond. The role of master and slave exchanges during the transfer.

A message consists of a sequence of characters which we identify as

- Control Characters
- Instrument Address
- Parameter Mnemonic
- Data

**Note:** Refer to "El Bisynch ASCII Message Protocol" page 14, where these four types of character are discussed in detail.

The following events take place in transmitting a successful message:

- Establish Connection
- Enquiry or Set Parameter
- Response
- Further Transmission and/or Termination

### Establish Connection

Connection is established with a particular device by sending its two-digit address (i.e. INSTRUMENT ADDRESS as above). This comprises the GROUP ID (GID) - first digit, and the COMMS ADDRESS (<sup>SE</sup>03) - second digit.

**Note:** The GROUP ID (GID) parameter is not available on the keypad and so the first digit is always "0" when using only the keypad. Over the Comms, it can be set from 0 to 7.

### Enquiry or Set Parameter

The message is either an enquiry (reading information from the Drive), or a message to set a parameter (writing information to the Drive).

## Response to a 'Set Parameter' Message

The Drive will respond to a Set Parameter message in one of three ways:

1. Positive Acknowledgement (ACK)
2. Negative Acknowledgement (NAK)
3. No Reply: Under certain circumstances the supervisor may not receive a reply from the Drive. This could be due to any of the following reasons:
  - Group/Unit address identifiers not recognised.
  - An error (e.g. parity) is found in one or more of the characters up to and including (ENQ).
  - Communications loop failure perhaps due to noise or wrong Baud Rate being selected.
  - Hardware failure.
  - Serial link is disabled on the Keypad.

In these cases the supervisor should be programmed to "time-out", i.e. wait for a response for a short time (160 msec minimum) before trying again.

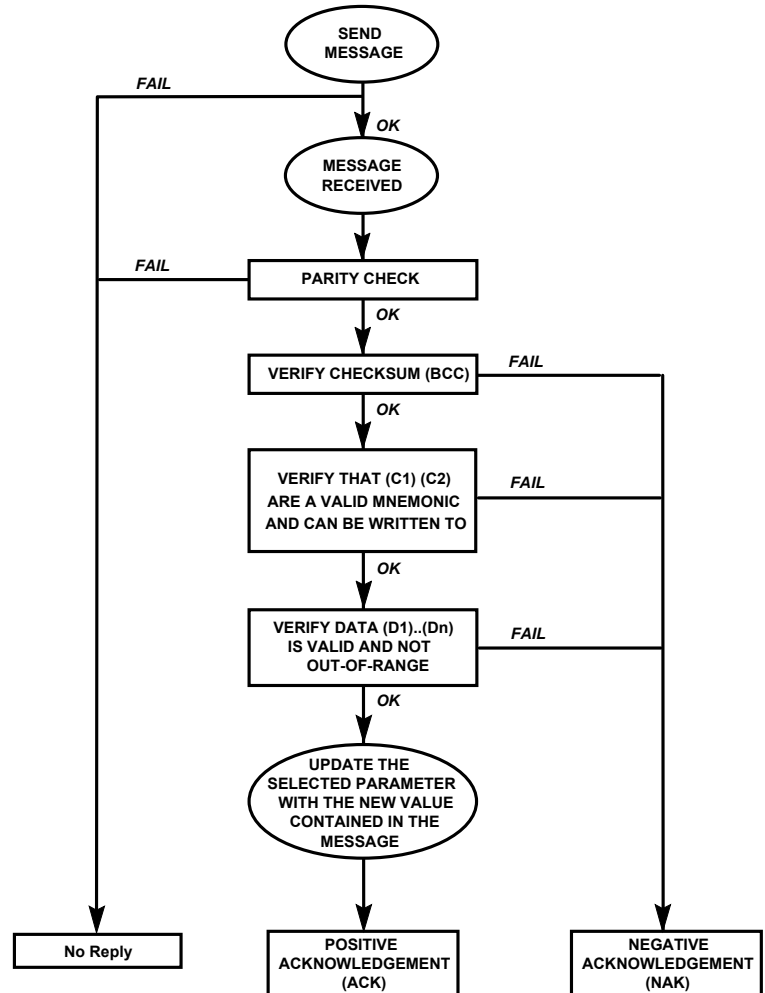


Figure 1 Drive Response Sequence to an ASCII Selection Message

## Further Transmission and/or Termination

### Further Transmission

If the supervisor still has an established connection with the device, you can repeat the previous message without re-establishing connection.

In both cases, writing to or reading from the device, you can use this to re-select the previous parameter or to select the next parameter in the parameter list. Refer to "Transferring Data - ASCII Example Messages", page 19 for further explanation.

### Termination (EOT)

If you wish to terminate connection with a particular device and establish connection with another, send the 'Establish Connection' sequence preceded by the (EOT) control character, (End Of Transmission).

The (EOT) character resets all devices on the data link to be responsive to the next four characters, i.e. the (GID)(GID)(UID)(UID) address.

In 2-wire operation, an (EOT) can only be sent when the supervisor has Master status.

## Programmer's Information

### ASCII (American Standard Code for Information Interchange)

The RS485 Option communicates using ASCII, a binary code which represents letters, digits, and control signals (collectively called characters).

The code, originated by the American National Standards Institute (ANSI), has become a world-wide standard for information interchange. It uses a seven bit binary word to represent all the letters, digits, punctuation marks and control signals.

#### Handling of Numerical Data

(Format 21 - Free Format Numeric)

Numerical Data is transferred as a string of characters. The drive will accept any format but will transmit an interpreted value that always contains a decimal point, and with no trailing zeros i.e.

1.00, 1.0, 1. or 1      is interpreted as      1.  
 -2.20 or -2.2          is interpreted as      -2.2

#### Handling of Status Information

(Format 23 - Hexadecimal)

Status Information is transmitted by first encoding the data into a hexadecimal format. The length of a string is then determined by the number of characters in the encoded data. The hexadecimal data is preceded by a '>' sign to differentiate it from numerical data.

**Note:** *Hexadecimal refers to the common practice of counting to the base of 16 in computing rather than the base of 10. The sixteen 'numbers' used being 0 to 9, A to F. Thus an 8 bit byte is represented by two characters in the range 00 to FF, while a 16 bit word is represented by four characters in the range 0000 to FFFF.*

#### Block Check Character (BCC)

This is a checksum value generated by taking the exclusive OR (XOR) of the ASCII values of all the characters transmitted after and excluding (STX) up to and including (ETX). For example, the shaded characters are included in the (BCC) of the following message:

(EOT)	(GID)	(GID)	(UID)	(UID)	(STX)	(C1)	(C2)	(D1)	(D2)	(D3)	(D4)	(D5)	(ETX)	(BCC)
-------	-------	-------	-------	-------	-------	------	------	------	------	------	------	------	-------	-------

*Example 1: EI Bisynch Prime Set*

#### For Beginners:

You can calculate this easily by converting the ASCII values to Binary and progressively adding the Binary values together, obeying the following rules:

$$\begin{array}{cccc} 0^+ & 1^+ & 1^+ & 0^+ \\ 0 & 1 & 0 & 1 \\ \hline 0 & 0 & 1 & 1 \end{array}$$

Referring to Example 1 on page 23, the calculation of (BCC) becomes:

As Characters	HEX	ASCII	Binary
(C1)	49	I	0 1 0 0 1 0 0 1
(C2)	49	I	0 1 0 0 1 0 0 1
(D1)	3E	>	0 0 1 1 1 1 1 0
(D2)	32	2	0 0 1 1 0 0 1 0
(D3)	36	6	0 0 1 1 0 1 1 0
(D4)	35	5	0 0 1 1 0 1 0 1
(D5)	30	0	0 0 1 1 0 0 0 0
(ETX)	03	(ETX)	0 0 0 0 0 0 1 1
(BCC)	3C	<	0 0 1 1 1 1 0 0 (TOTAL)

## EI Bisynch ASCII Message Protocol

<b>Transmission Standard</b>	:	RS485
<b>Protocol</b>	:	ANSI-X3.28-2.5-B1
<b>Data Rates</b>	:	300, 600, 1200, 2400, 4800, 9600 or 19200 Baud
<b>Character Format</b>	:	1 start + 7 bit ASCII data + 1 parity + 1 stop bit (10 bits)
<b>Parity</b>	:	Even

The Protocol defines the string or sequence of characters (called a Message) which must be sent between communicating instruments to produce specific responses. The message usually comprises:

- Control Characters
- Instrument Address
- Parameter Mnemonic
- Data

### Control Characters

Control Characters are ASCII codes that define actions rather than information. Six ASCII codes are supported:

<i>Keyboard</i>	<i>HEX</i>	<i>ASCII</i>	
^B	02	(STX)	<i>Start of Text</i>
^C	03	(ETX)	<i>End of Text</i>
^D	04	(EOT)	<i>End of Transmission</i>
^E	05	(ENQ)	<i>Enquiry</i>
^F	06	(ACK)	<i>Positive Acknowledge</i>
^U	15	(NAK)	<i>Negative Acknowledge</i>

### Instrument Address

The Drive has a two-digit address, the first digit being the “group” ID number (GID) in the range 0 to 7, the second digit is a “unit” ID number (UID) in the range 0 to F. There are therefore 128 different addresses from 00 to 7F.

The Instrument Address (01 for example) is repeated in the message (i.e. 0011) for security as it is not included in a Checksum.

### Parameter Mnemonic

Each parameter in the Drive’s menu system is identified by a unique Tag Number. Information is exchanged across the system by use of a two character Mnemonic that is derived from the Tag Number.

Examples are:

- 81 : the TRIPPED parameter from the SEQUENCING LOGIC function block
- 3b : the STATOR RES parameter from the MOTOR DATA function block

**Note:** Refer to the 650 or 650V Software Product Manual, Chapter 2 for a full listing of Tag Numbers and Mnemonics.

## EI Bisynch ASCII Parameter Mapping

### 1. EI Bisynch ASCII Prime Set

The following prime set parameters are supported:

Mnemonic	Description	Range (HEX encoding)	Access
II	Instrument Identity	>0650, >1650 or >2650 0650 = 650 Frames 1, 2 & 3 1650 = 650V Frames 1, 2 & 3 2650 = 650V Frames C, D, E & F	Read Only
V0	Main Software Version	>0000 to >FFFF	Read Only
V1	Keypad Software Version	>0000 to >FFFF (>0000 if not fitted)	Read Only
EE	Last Error Code	>0000 to >FFFF (Writing any value resets this to >00C0)	Read/Write

### 2. Command/Status

The following Command/Status parameters are supported:

Mnemonic	Description	Range (Hex encoding)	Access
!1	Command	see below	Write Only
!2	State	see below	Read Only
!3	Save Command	see below	Write Only
!4	Save State	see below	Read Only

#### !1 : Command

Write-only: used to modify the state of the drive and to load configuration data from non-volatile memory.

HEX Value	Description
>7777	Reset Command. Acknowledges failed restore. Loads and saves default Product Code and default Configuration (Application 1).
>0101	Restores Saved Configuration from drive's non-volatile memory.
>0110	Restores Default Configuration (Application 0)
>0111	Restores Default Configuration (Application 1)
>0112	Restores Default Configuration (Application 2)
>0113	Restores Default Configuration (Application 3)
>0114	Restores Default Configuration (Application 4)
>0115	Restores Default Configuration (Application 5)
>4444	Exit Configuration Mode
>5555	Enter Configuration Mode

#### !2 : State

Read-only: used to determine the major state of the Inverter.

HEX Value	Description
>0000	Initialising. (Powering up )
>0001	Corrupted Product Code and Configuration
>0002	Corrupted Configuration
>0003	Restoring Configuration
>0004	Re-Configuring Mode
>0005	Normal Operation Mode

<b>!3 : Save Command</b>	
Write-only: used to save the configuration and product code in non-volatile memory.	
HEX Value	Description
>0000	Reset Command. Acknowledges (clears) any previous save error.
>0001	Saves Configuration to drive's non-volatile memory.
>0100	Saves Product Code to drive's non-volatile memory.

<b>!4 : Save State</b>	
Read only: used to determine the progress of a non-volatile saving operation.	
HEX Value	Description
>0000	Idle
>0001	Saving
>0002	Failed

### 3. Tag Access

Each parameter in the drive's menu system is identified by a unique Tag Number. Information is exchanged across the system by use of a two character Mnemonic that is derived from the Tag Number.

**Note:** Refer to the Parameter Specification Table in the 650 or 650V Software Product Manual for a full list of tag mnemonics - see the ID column. Refer to the Notes column which gives access information about each parameter.

#### Parameter Mapping

##### 650V Algorithm

The algorithm to convert between tag number and 2 character mnemonics is:

```

if (TagNo < 1296)
{
    m = INT (TagNo / 36) (INT: the integer part)
    n = TagNo MOD 36 (MOD: the remainder)
    if m > 9 then
        char_1 = 'a' + (m - 10)
    else
        char_1 = '0' + m
    end_if
    if n > 9 then
        char_2 = 'a' + (n - 10)
    else
        char_2 = '0' + n
    }
else
{
    m = INT (TagNo - 1296) / 126
    n = (TagNo - 1296) MOD 26
    char_1 = 'a' + n
    char_2 = 'A' + m
}
end_if

```

The algorithm generates mnemonics containing only the characters '0' to '9' and 'a' to 'z'.

#### 4. Encoding

Type	Description	Encoding	Comments
BOOL	Boolean	FALSE >00 TRUE >01	Will accept >0 and >1
WORD	16-bit Bitstring	>0000 to >FFFF	Will accept leading zero suppression, except >0
REAL	Signed Integer	-XXXX. to XXXX. -XXXX.X to XXXX.X -XXX.XX to XXX.XX -XX.XXX to XX.XXX -X.XXXX to X.XXXX	Leading zeroes suppressed up to digit before decimal point. Trailing zeroes suppressed after decimal point.
ENUM	Enumerated Value ( 0 to 99)	XX.	Leading zeroes suppressed, except 0.
INT	Unsigned Integer	XXXXX.	Leading zeroes suppressed up to digit before decimal point.
TAG	Tag number	-XXXXX. to XXXXX.	Leading zeroes suppressed up to digit before decimal point.

*Note: The "." in the above formats is not optional. It must be sent to conform to the EI-BISYNCH standard.*



### EI Bisynch ASCII Sequence Diagrams

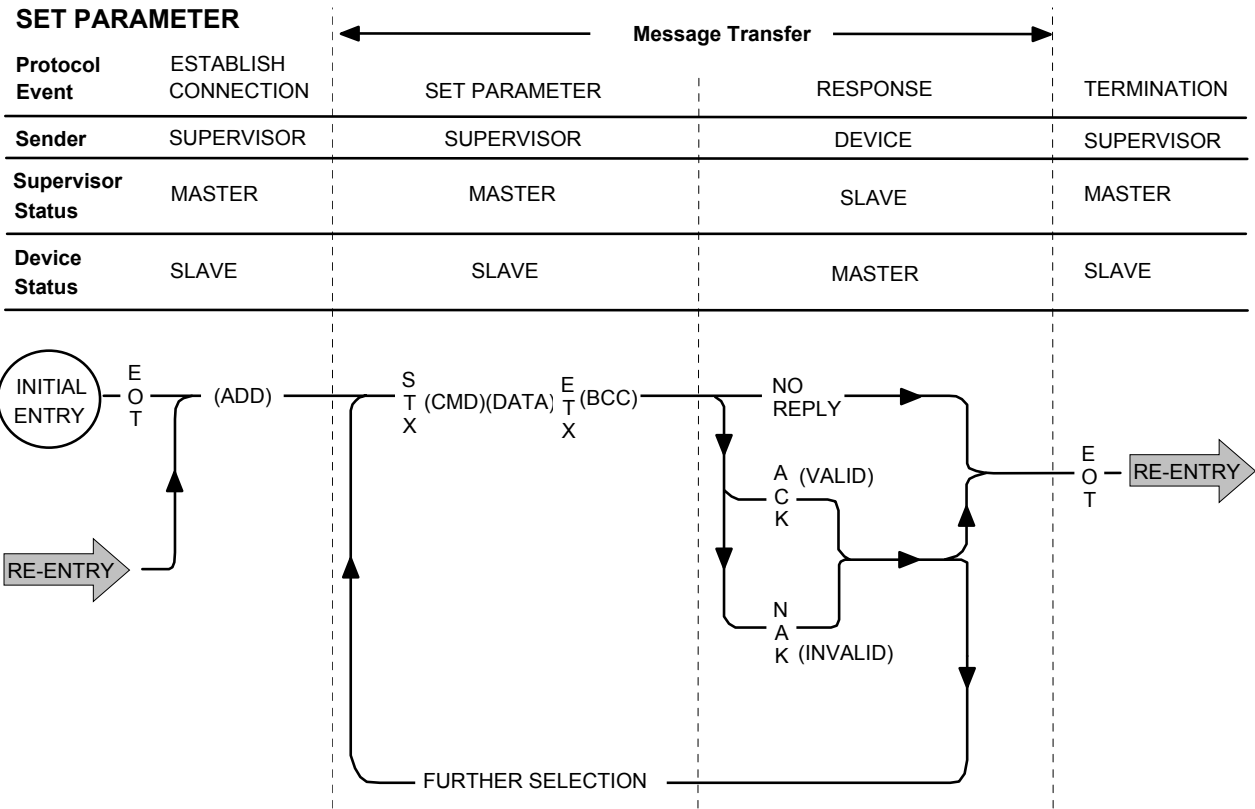


Figure 2 Selection Sequence for Writing Information to the Drive

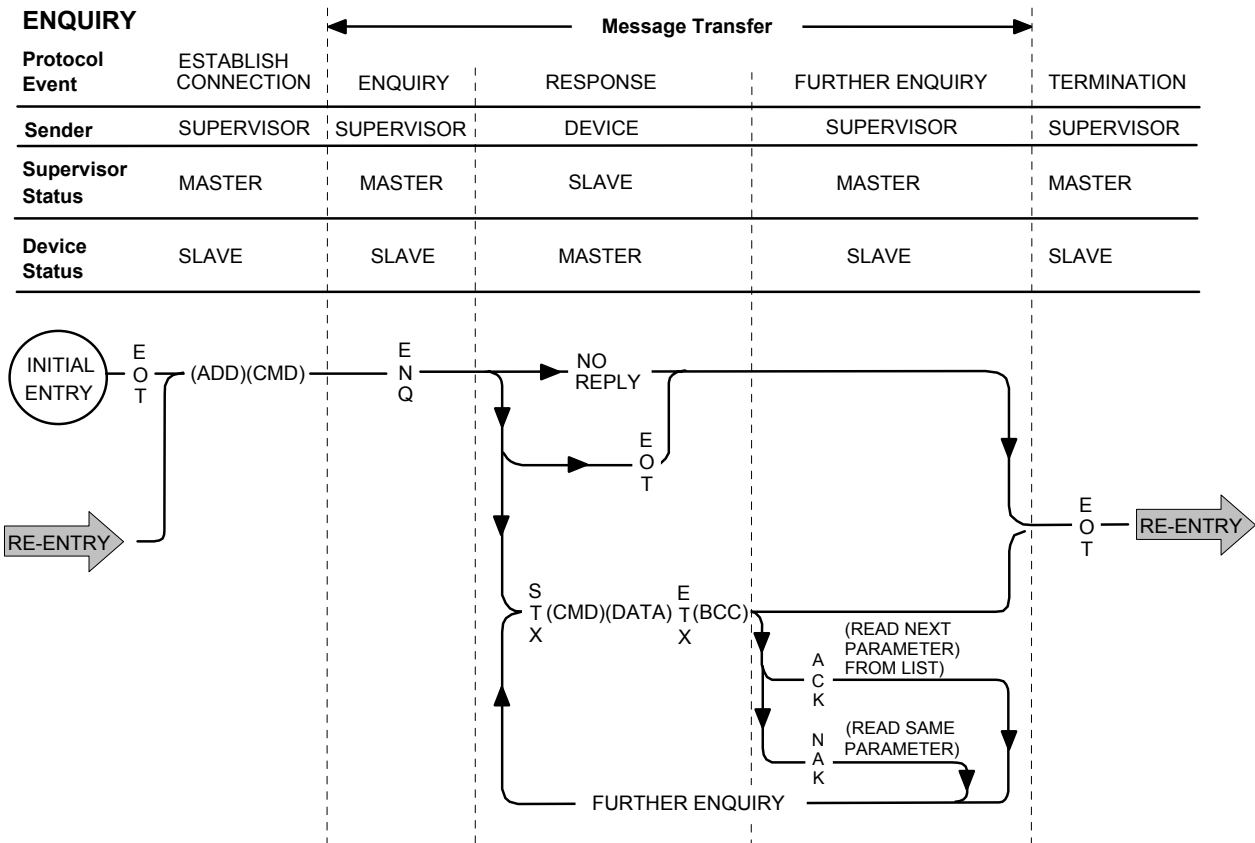


Figure 3 Poll Sequence for Reading Information from the Drive

## Transferring Data - ASCII Example Messages

The following examples show how data transfer takes place using the network, they will also help to verify your communications if you using the RS485/RS232 Communications Interface for the first time. Many users will not become involved in generating low-level code, but for those experienced in programming, the examples include ASCII, HEX and Control Character information.

**Note:** Refer to "Control Character Definitions", page 24 for a more detailed explanation of all control characters.

### Example 1: EI Bisynch Prime Set

**Note:** Refer to the 650 or 650V Software Product Manual for a full list of Tag No's/ID's (mnemonics).

Using this set of mnemonics, you can enquire about the drive. For instance, you could enquire about the Instrument Identity:

#### ENQUIRY

- **For software users:**

Enter the known address of the Drive (say 01), II, and that it is an enquiry.

- **For programmers, in ASCII:**

(EOT)	0	0	1	1	I	I	(ENQ)
-------	---	---	---	---	---	---	-------

- **For programmers, in HEX:**

04	30	30	31	31	49	49	05
----	----	----	----	----	----	----	----

- **As Characters - Establish Connection | Ask Question:**

(EOT)	(GID)	(GID)	(UID)	(UID)	(C1)	(C2)	(ENQ)
-------	-------	-------	-------	-------	------	------	-------

**Note:** The (GID)(UID) address is always entered twice.  
 Refer to "Instrument Address", page 14 for a more detailed explanation.

#### RESPONSE

- **For software users:**

The Instrument Identity will be returned, in our case 2650 (representing a 650V drive, size C to F - 1650 represents a 650V drive, size 1 to 3)

- **For programmers, in ASCII:**

(STX)	I	I	>	2	6	5	0	(ETX)	<
-------	---	---	---	---	---	---	---	-------	---

- **For programmers, in HEX:**

02	49	49	3E	32	36	35	30	03	3C
----	----	----	----	----	----	----	----	----	----

- **As Characters - Valid Response:**

(STX)	(C1)	(C2)	(D1)	(D2)	(D3)	(D4)	(D5)	(ETX)	(BCC)
-------	------	------	------	------	------	------	------	-------	-------

**Note:** The BCC checksum (XOR) of the data after and excluding (STX) up to and including (ETX) is "1" and >31. Refer to "Block Check Character (BCC)", page 13 for a more detailed explanation.

In Example 1, connection to a new device is being made, i.e. the "Establish Connection" information is transmitted. However, these examples can be transmitted without the "Establish Connection" information if connection to the correct device is already established. This is shown by Examples 3, 5 & 6.

### Example 2: Tag Access (Single Parameter Poll)

Here we ask a question of a single parameter: *what is the value of SETPOINT?*

(Tag 254, SETPOINT, ID 72, Type REAL - see the Parameter Specification Table in the 650 or 650V Software Product Manual for this information)

#### ENQUIRY

- *For software users:*

Enter the known address of the drive (say 01), 72, and that it is an enquiry.

- *For programmers, in ASCII:*

(EOT)	0	0	1	1	7	2	(ENQ)
-------	---	---	---	---	---	---	-------

- *For programmers, in HEX:*

04	30	30	31	31	37	32	05
----	----	----	----	----	----	----	----

- *As Characters - Establish Connection | Ask Question:*

(EOT	(GID)	GID)	(UID)	(UID)	(C1)	(C2)	(ENQ)
------	-------	------	-------	-------	------	------	-------

**Note:** The (GID)/(UID) address is always entered twice.  
Refer to "Instrument Address", page 14 for a more detailed explanation.

#### RESPONSE

- *For software users:*

The SETPOINT value will be returned, say 30. (representing 30.00%)

- *For programmers, in ASCII:*

(STX)	7	2	3	0	.	(ETX)	`
-------	---	---	---	---	---	-------	---

- *For programmers, in HEX:*

02	37	32	33	30	2E	03	2B
----	----	----	----	----	----	----	----

- *As Characters - Valid Response:*

(STX)	(C1)	(C2)	(D1)	(D2)	(D3)	(ETX)	(BCC)
-------	------	------	------	------	------	-------	-------

**Note:** The BCC checksum (XOR) of the data after and excluding (STX) up to and including (ETX) is `` and >2B. Refer to "Block Check Character (BCC)", page 13 for a more detailed explanation.



**Example 4: Tag Access (Single Parameter Selection)**

Here we are writing a value to a single parameter: *the value of PRESET INPUT 1 is 30.00%.*

**SET PARAMETER**

(Tag 348, PRESET INPUT 1, ID 9o, Type REAL - see the Parameter Specification Table in the 650 or 650V Software Product Manual for this information)

**For software users:**

Enter the known address of the drive (say 01), (STX), 90, 30. and (ETX).

- **For programmers, in ASCII:**

(EOT) 0 0 1 1 (STX) 9 0 3 0 . (ETX) (

- **For programmers, in HEX:**

04 30 30 31 31 02 39 6F 33 30 2E 03 78

- **As Characters - Establish Connection | Data Transfer:**

(EOT) (GID) (GID) (UID) (UID) (STX) (C1) (C2) (D1) (D2) (D3) (ETX) (BCC)

**Note:** The (GID)/(UID) address is always entered twice.

**Note:** Refer to "Instrument Address", page 14 for a more detailed explanation.

**Note:** The BCC checksum (XOR) of the data after and excluding (STX) up to and including (ETX) is "(" and >78. Refer to "Block Check Character (BCC)", page 13 for a more detailed explanation.

**RESPONSE**

- **For software users:**

The response will be either (ACK), (NAK) or no reply. If (ACK), the parameter value will be updated at the drive.

- **For programmers, in ASCII:**

either (ACK), (NAK) or no reply

- **For programmers, in HEX:**

either 06, 15 or no reply

- **As Characters:**

either (ACK), (NAK) or no reply

### Example 5: Tag Access (Continuous Selection of a Parameter)

You can repeat a valid selection (from Example 4) without having to re-establish connection to the drive. You can use this to continuously update a parameter. Lets say the new value is 35. (representing 35.00%).

#### SET PARAMETER

- *For software users:*  
Send (STX), 90, 35. and (ETX).
- *For programmers, in ASCII:*

(STX)	9	0	3	5	.	(ETX)	-
-------	---	---	---	---	---	-------	---

- *For programmers, in HEX:*

02	39	6F	33	35	2E	03	7D
----	----	----	----	----	----	----	----

- *As Characters - Data Transfer:*

(STX)	(C1)	(C2)	(D1)	(D2)	(D3)	(ETX)	(BCC)
-------	------	------	------	------	------	-------	-------

**Note:** The BCC Checksum is the result of the new value you are sending to the drive.

**Note:** Refer to "Block Check Character (BCC)", page 13 for a more detailed explanation.

#### RESPONSE

- *For software users:*  
The response will be either (ACK), (NAK) or no reply. If (ACK), the parameter value will be updated at the drive.

- *For programmers, in ASCII:*

either (ACK), (NAK) or no reply
---------------------------------

- *For programmers, in HEX:*

either 06, 15 or no reply
---------------------------

- *As Characters:*

either (ACK), (NAK) or no reply
---------------------------------

### Example 6: Tag Access (Sequential Selection)

You can also repeat a valid selection (as Example 5) without having to re-establish the connection to the drive to update any other specified parameter. Lets say the next parameter you want to update is DIGITAL INPUT 1 INVERT whose new value is to be TRUE.

(Tag 30, DIGITAL INPUT 1 INVERT, ID 0u, Type BOOL - see the Parameter Specification Table in the 650 or 650V Software Product Manual for this information)

#### SET PARAMETER

- *For software users:*  
Send (STX), 0u, 1 and (ETX).
- *For programmers, in ASCII:*

(STX)	0	u	>	0	1	(ETX)	m
-------	---	---	---	---	---	-------	---

- *For programmers, in HEX:*

02	30	F5	3E	30	31	03	F9
----	----	----	----	----	----	----	----

- *As Characters - Data Transfer:*

(STX)	(C1)	(C2)	>	(D1)	(D2)	(ETX)	(BCC)
-------	------	------	---	------	------	-------	-------

**Note:** The BCC Checksum is the result of the new information you are sending to the Drive.

#### RESPONSE

The response will be as for Example 5.

## Character Definitions

Standard Character Definitions	
(GID)	The Group address Identifier (repeated for security)
(UID)	The Unit address identifier (repeated for security)
(C1) (C2)	The two characters of the parameter mnemonic (from the Tag number)
(D1)..(Dn)	The value of the requested parameter (string may be any length, determined by the data).
(BCC)	Block Check Character: a character generated by taking the exclusive OR (XOR) of the ASCII values of all the characters transmitted after and excluding (STX) up to and including (ETX)

## Control Character Definitions

Standard Control Character Definitions	
(STX)	Start of text
(ETX)	End of text
(EOT)	End of Transmission: resets all instruments on the link and causes them to examine the next four transmitted characters to see if they correspond with their Group/Unit address identifiers  Also sent to terminate communication with a particular device.

Control Character Definitions when Reading Information	
(ENQ)	Indicates the end of the message, and that it is an enquiry
(ACK)	Sequential Polling: when transmitted after a valid response, this fetches data from the next parameter in the parameter list
(NAK)	Continuous Polling: when transmitted after a valid response, this fetches data from the previously requested parameter
(EOT)	The information received contained an error

Control Character Definitions when Writing Information	
(ACK)	Positive Acknowledgement: the message was correctly received and the parameter updated
(NAK)	Negative Acknowledgement: the message received by the drive contained an error and the parameter was not updated

## Last Error Code (EE)

The EI-BISYNCH Prime Set contains the EE mnemonic. The following values are returned if an enquiry (reading information from the drive) is performed on this Read/Write parameter.

Writing any value to this parameter will set the value to >00C0. Clearing the last error value may be useful in seeing a repetitive error re-occurring.

Value	Description
>00C0	No error
>01C7	Invalid Mnemonic
>02C2	Checksum (BCC) error
>04C8	Attempt to read from a write-only parameter
>05C8	Attempt to write to a read-only parameter
>07C8	Invalid Data (Encoding error)
>08C8	Data out of range



# Initial Set-up for MODBUS RTU

**Note:** Pages 10, 11 and 12 are repeated here as pages 29, 30 and 31 for your convenience.

## Configuring the Drive

**Note:** The RS485/RS232 Communications Module can only be used on drives using software version 4.1 or higher (indicated on power-up, i.e. "r4.1").

You must configure the drive to your system.

If you are using the keypad (MMI), the parameters to edit are in the SERIAL menu, <sup>S</sup>SE01 to <sup>S</sup>SE09.

If you are using DSELite (or other suitable PC programming tool) the same parameters are contained in the COMMS PORTS and COMMS CONTROL function blocks.

*DSELite is Parker SSD Drives' Windows-based block programming software.*

Comms Control			
		COMMS SEQ [295]	FALSE
		COMMS REF [270]	FALSE
		COMMS STATUS [272]	0
		COMMS COMMAND [273]	0
FALSE		[300] REMOTE COMMS SEL	
TERMINALS/COMMS		[307] REMOTE SEQ MODES	
TERMINALS/COMMS		[308] REMOTE REF MODES	
0.0 s		[309] COMMS TIMEOUT	

Comms Ports			
0		[102] GROUP ID (GID)	
0		[103] COMMS ADDRESS	
9600		[1062] BAUD RATE	
NONE		[1061] PARITY	
AUTOMATIC		[1060] OP PORT PROTOCOL	
AUTOMATIC		[1059] P3 PORT PROTOCOL	
MODBUS		[117] RS485 PROTOCOL	
FALSE		[129] SWITCH OP PORT	
FALSE		[[90]] SWAP WORD ORDER	

**Note:** To view all parameters available on the MMI, FULL menu detail must be selected in the DETAILED MENUS parameter (<sup>ST</sup>99): 1 = FULL.

## Parameter Descriptions : COMMS CONTROL

This block switches between Remote Terminal and Remote Comms operating modes.

The drive must be in Remote operating mode for selection to be made - REMOTE mode is enabled in the LOCAL CONTROL function block (REF MODES) or selected by the keypad.

**REMOTE COMMS SEL**      SET\SERL SE01      *Range: FALSE / TRUE*

Selects the type of remote communications mode:

0 : FALSE, and in REMOTE mode then control is from the terminals.

1 : TRUE, and in REMOTE mode then control is from the communications.

**REMOTE SEQ MODES**      *Range: Enumerated - see below*

Selects the type of remote sequencing mode:

*Enumerated Value : Mode*

0 : TERMINALS/COMMS

1 : TERMINALS ONLY

2 : COMMS ONLY

**REMOTE REF MODES**      *Range: Enumerated - see below*

Selects the type of remote reference mode:

*Enumerated Value : Mode*

0 : TERMINALS/COMMS

1 : TERMINALS ONLY

2 : COMMS ONLY

**COMMS TIMEOUT**      SET\SERL SE02      *Range: 0.0 to 600.0 s*

Sets the maximum time allowed between refreshing the COMMS COMMAND parameter. The drive will trip if this time is exceeded. Set the time to 0.00 seconds to disable this feature.

## COMMS SEQ

*Range: FALSE / TRUE*

Diagnostic indicating if operating in Remote Sequencing Comms Mode.

If FALSE (0), the drive may be in Local Sequencing mode or Remote Sequencing Terminal mode.

## COMMS REF

*Range: FALSE / TRUE*

Diagnostic indicating if operating in Remote Reference Comms Mode.

If FALSE (0), the drive may be in Local Reference mode or Remote Reference Terminal mode.

## COMMS STATUS

*Range: 0000 to FFFF*

Diagnostic showing the 16-bit Status word as seen by the communications.

Refer to Chapter 4: "Sequencing Logic" in the 650 or 650V Software Product Manual.

## COMMS COMMAND

*Range: 0000 to FFFF*

Diagnostic showing the 16-bit Command as written by the communications.

Refer to Chapter 4: "Sequencing Logic" in the 650 or 650V Software Product Manual.

## Parameter Descriptions: COMMS PORTS

This function block configures the programming ports that allow connection to the keypad, or to a personal computer.

The parameters below are used to identify the drive to the controlling software for drive configuration and storage of parameters.

*Note: The unit will always respond to GID = 0 and UID = 0, as this is the "broadcast" address used by the keypad.*

### GROUP ID (GID)

*Range: 0 to 7*

The Parker SSD Drives protocol group identity address.

### COMMS ADDRESS

 SET\SERL SE03

*Range: 0 to 255*

The Parker SSD Drives protocol unit identity address or the Modbus node address.

Note: if set to 0, it will only respond to broadcast messages.

### BAUD RATE

 SET\SERL SE04

*Range: Enumerated - see below*

Selects the Baud Rate for the MODBUS protocol.

*Enumerated Value : Baud Rate*

0 : 1200

1 : 2400

2 : 4800

3 : 7200

4 : 9600

5 : 14400

6 : 19200

7 : 38400

8 : 57600

### PARITY

 SET\SERL SE05

*Range: Enumerated - see below*


Selects the Parity for the MODBUS protocol.

*Enumerated Value : Parity*

0 : NONE

1 : ODD


2 : EVEN

**OP PORT PROTOCOL**     *SET\SERL SE06*    *Range: Enumerated - see below*

Selects the protocol to be used by the keypad port on the front of the drive. When EIBISYNC ASCII is selected, BAUD RATE is 19200 and PARITY is EVEN.

*Enumerated Value : Protocol*


- 0 : AUTOMATIC - checks for keypad or EI ASCII
- 1 : KEYPAD
- 2 : EIBISYNC ASCII
- 3 : MODBUS
- 4 : FIELDBUS

**P3 PORT PROTOCOL**     *SET\SERL SE07*    *Range: Enumerated - see below*

Selects the protocol to be used by the RS232 programming port on the drive's control board. When EIBISYNC ASCII is selected, BAUD RATE is 19200 and PARITY is EVEN.

*Enumerated Value : Protocol*

- 0 : AUTOMATIC - checks for keypad or EI ASCII
- 1 : KEYPAD
- 2 : EIBISYNC ASCII
- 3 : MODBUS
- 4 : FIELDBUS

**RS485 PORT PROTOCOL**     *SET\SERL SE08*    *Range: Enumerated - see below*

**This parameter is not available on the 650V Frame 1, 2 & 3.**

Selects the protocol to be used by the RS485 programming port on the drive's control board.

*Enumerated Value : Protocol*

- 0 : AUTOMATIC
- 1 : KEYPAD (not applicable)
- 2 : EIBISYNC ASCII
- 3 : MODBUS
- 4 : FIELDBUS

**SWITCH OP PORT**     *SET\SERL SE09*    *Range: FALSE / TRUE*

**This parameter is not available on the 650V Frame 1, 2 & 3.**

When TRUE, the keypad port on the front of the drive is disabled when the communications equipment is connected to the RS232 programming port on the drive's control board.

When FALSE, the RS485 programming port is disabled when the communications equipment is connected to the RS232 programming port. Both ports are on the drive's control board.

**SWAP WORD ORDER**    *Range: FALSE / TRUE*

Controls the word order used with 32-bit data access with the Modbus RTU protocol. When this parameter is FALSE the lower, even address of a pair of registers will contain the least significant 16 bits of the parameter data. The higher, odd address will contain the most significant 16 bits of parameter data.

When this parameter is true, the most significant 16 bits of parameter data are held in the lower, even register, while the least significant 16 bits of parameter data are held in the higher, odd register.

## Configuring the PLC/SCADA Supervisor

By referring to the Parameter Specification Table in the 650 or 650V Software Product Manual, you can enter the parameter information you require.

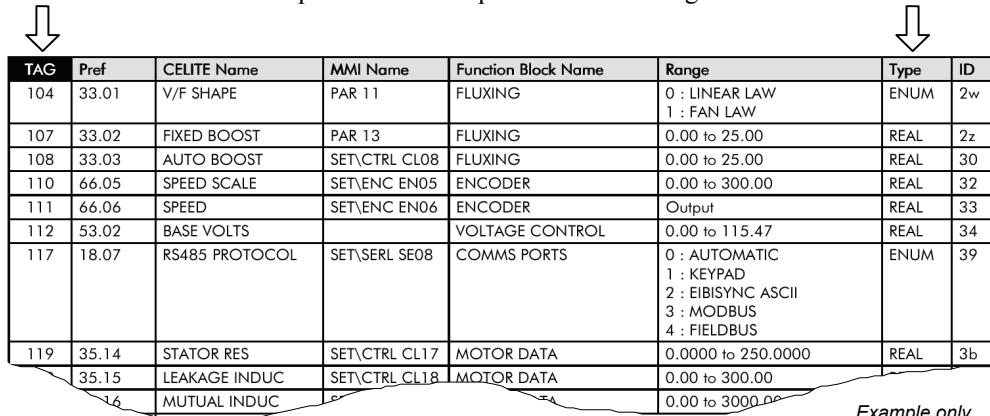
It provides the information in the following way:

### Type

The first page of the Parameter Specification Table chapter details parameter types. The Type column indicates each parameter's type.

### TAG

The TAG column provides the unique identification tag number.



TAG	Pref	CELITE Name	MMI Name	Function Block Name	Range	Type	ID
104	33.01	V/F SHAPE	PAR 11	FLUXING	0 : LINEAR LAW 1 : FAN LAW	ENUM	2w
107	33.02	FIXED BOOST	PAR 13	FLUXING	0.00 to 25.00	REAL	2z
108	33.03	AUTO BOOST	SET\CTRL CL08	FLUXING	0.00 to 25.00	REAL	30
110	66.05	SPEED SCALE	SET\ENC EN05	ENCODER	0.00 to 300.00	REAL	32
111	66.06	SPEED	SET\ENC EN06	ENCODER	Output	REAL	33
112	53.02	BASE VOLTS		VOLTAGE CONTROL	0.00 to 115.47	REAL	34
117	18.07	RS485 PROTOCOL	SET\SERL SE08	COMMS PORTS	0 : AUTOMATIC 1 : KEYPAD 2 : EIBISYNC ASCII 3 : MODBUS 4 : FIELDBUS	ENUM	39
119	35.14	STATOR RES	SET\CTRL CL17	MOTOR DATA	0.0000 to 250.0000	REAL	3b
	35.15	LEAKAGE INDUC	SET\CTRL CL18	MOTOR DATA	0.00 to 300.00		
	35.16	MUTUAL INDUC	SET\CTRL CL19	MOTOR DATA	0.00 to 3000.00		

*Example only*

## MODBUS RTU Communications

A MODBUS RTU communication network can have only one Master, and one or more Slave devices.

- Each Slave has a unique “device address”
- The device address “0” is a special case and is used for messages that are broadcast to all Slaves. This is restricted to parameter write operations.
- The unit supports a subset of MODBUS RTU function codes.
- The data includes parameters referenced by a “parameter address”.
- Sending a communication with a unique device address causes only the device with that address to respond. That device will check for errors, perform the requested task and then reply with its own address, data and check sum.
- Sending a communication with the device address “0” is a broadcast communication that sends information to all devices on the network. Each device performs the required action but does not transmit a reply.

### How is the Information Transferred?

A typical transaction consists of a request sent from the Master followed by a response from the Slave.

A message consists of a sequence of characters which we identify as:

- Device Address
- Function Code
- Data
- Error Check Data
- End of Transmission

### Device Address

Each Slave has a unique 8-bit device address. The Gould MODBUS Protocol defines the address range limits as 1 to 247 (device address 0 is the broadcast message to all slaves simultaneously).

## Parameter Address

Data bits or data words exchange information between Master and Slave devices. This data consists of parameters. All parameters communicated between Master and Slaves have a 16-bit parameter address.

The MODBUS parameter address range is 0001 to FFFF.

## RTU Mode of Transmission

The MODBUS RTU definition of the mode of transmission for a single character is:

*A start bit, eight data bits, a parity bit, one or two stop bits*

All Parker SSD Drives' units use one stop bit.

Parity may be configured to be NONE, ODD or EVEN (if NONE, no parity bit is transmitted)

The RTU mode of transmission for a single character is represented as follows:

Start	d7	d6	d5	d4	d3	d2	d1	d0	Parity	Stop
-------	----	----	----	----	----	----	----	----	--------	------

## Message Frame Format

A message frame format consists of a number of correctly sequenced characters, as shown below.

Frame Start	Device Address	Function Code	Data	CRC	EOT
3 bytes	1 byte	1 byte	n bytes	2 bytes	3 bytes

### Frame Start

The frame start is a period of inactivity at least 3.5 times the single character transmission time. For example, at 9600 baud a character with a 1 start, 1 stop and 8 data bits will require 3.5ms frame start. This period is the implied EOT of a previous transmission.

### Device Address

The device address is a single byte (8-bits), unique to each device on the network.

### Function Code

Function codes are a single byte instruction to the Slave describing the action to perform.

### Data

The Data segment of a message will depend on the function code and the number of bytes will vary accordingly. Typically, the data segment will contain a parameter address and the number of parameters to read or write.

### CRC

The CRC (Cyclic Redundancy Check) is an error code and is 2 bytes (16-bits) long.

### EOT

The EOT (End Of Transmission) segment is a period of inactivity 3.5 times the single character transmission time. The EOT segment at the end of a message indicates to the listening device that the next transmission will be a new message and therefore a device address character.

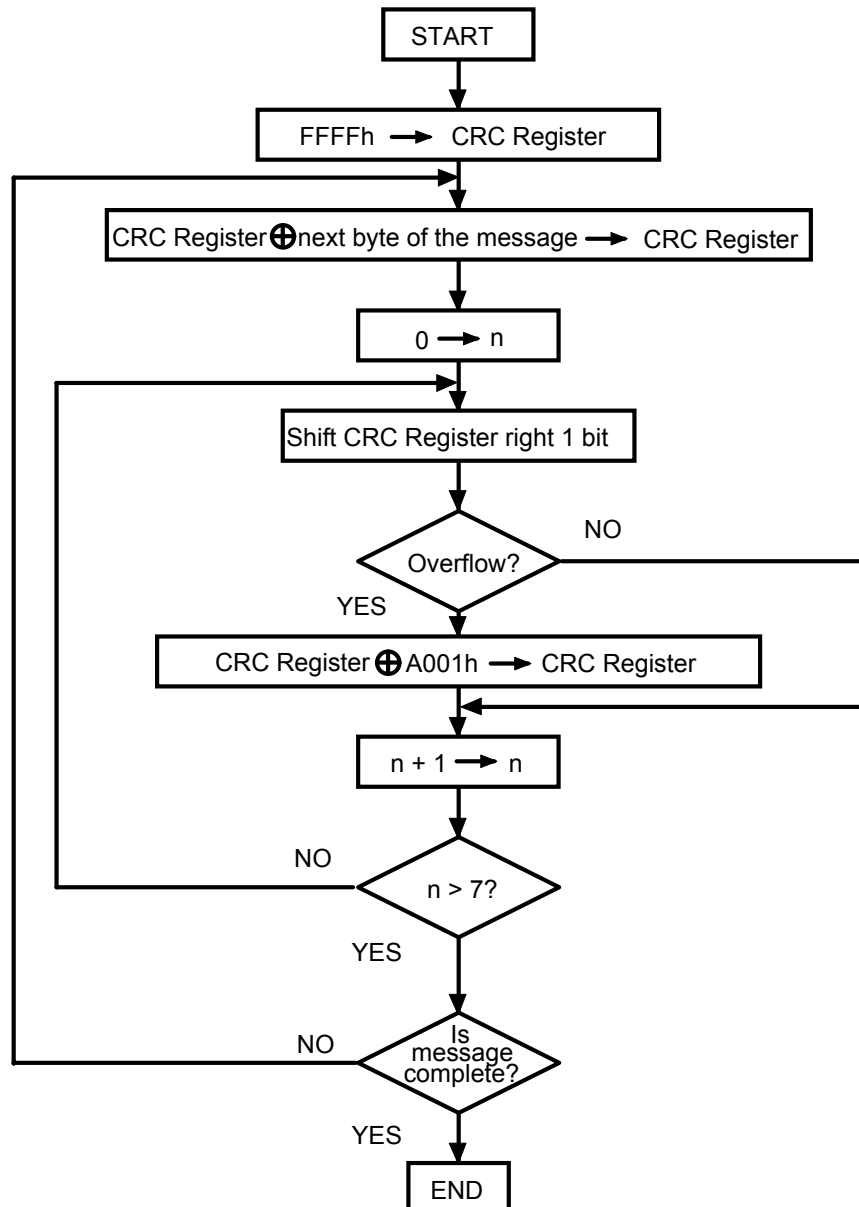
## Cyclic Redundancy Check

This is an error check code and is 2 bytes (16-bits) long. After constructing a message (data only - no start, stop or parity bits), the transmitting device calculates a CRC code and appends this to the end of the message. The receiving device also calculates a CRC code from the received message. If this CRC code is not the same as the transmitted CRC there has been a communication error. Units do not reply if they detect a CRC error in messages sent to them.

The CRC code is formed by the following steps:

1. Load a 16-bit CRC register with FFFFh.
2. Exclusive OR ( $\oplus$ ) the first 8-bit byte of the message with the high order byte of the CRC register. Return the result to the CRC register.
3. Shift the CRC register one bit to the right.
4. If the overflow bit (or flag) is 1, exclusive OR the CRC register with A001 hex and return the result to the CRC register.
5. Repeat steps 3 & 4 seven times (8 in total).
6. Exclusive OR the next 8-bit byte of the message with the high order byte of the CRC register.
7. Repeat step 3 through 6 until all bytes of the message have been exclusive OR'd with the CRC register and shifted 8 times.
8. The contents of the CRC register are the 2 byte CRC error code and are added to the message with the most significant bits first.

The flow chart below illustrates this CRC error check algorithm.



### Example of a CRC Calculation

This example is a request to read from the Slave unit at address 02, the fast read of the status (07).

Function	16 Bit Register				Carry Flag
	LSB		MSB		
Load register with FFFF hex	1111	1111	1111	1111	0
First byte of the message (02)			0000	0010	
Exclusive OR	1111	1111	1111	1101	
1st shift right	0111	1111	1111	1110	1
A001	1010	0000	0000	0001	
Exclusive OR (carry = 1)	1101	1111	1111	1111	
2nd shift right	0110	1111	1111	1111	1
A001	1010	0000	0000	0001	
Exclusive OR (carry = 1)	1100	1111	1111	1110	
3rd shift right	0110	0111	1111	1111	0
4th shift right (carry = 0)	0011	0011	1111	1111	1
A001	1010	0000	0000	0001	
Exclusive OR (carry = 1)	1001	0011	1111	1110	
5th shift right	0100	1001	1111	1111	0
6th shift right (carry = 0)	0010	0100	1111	1111	1
A001	1010	0000	0000	0001	
Exclusive OR (carry = 1)	1000	0100	1111	1110	
7th shift right	0100	0010	0111	1111	0
8th shift right (carry = 0)	0010	0001	0011	1111	1
A001	1010	0000	0000	0001	
Exclusive OR (carry = 1)	1000	0001	0011	1110	
Next byte of the message (07)			0000	0111	
Exclusive OR (shift = 8)	1000	0001	0011	1001	
1st shift right	0100	0000	1001	1100	1
A001	1010	0000	0000	0001	
Exclusive OR (carry = 1)	1110	0000	1001	1101	
2nd shift right	0111	0000	0100	1110	1
A001	1010	0000	0000	0001	
Exclusive OR (carry = 1)	1101	0000	0100	1111	
3rd shift right	0110	1000	0010	0111	1
A001	1010	0000	0000	0001	
Exclusive OR (carry = 1)	1100	1000	0010	0110	
4th shift right	0110	0100	0001	0011	0
5th shift right (carry = 0)	0011	0010	0000	1001	1
A001	1010	0000	0000	0001	
Exclusive OR (carry = 1)	1001	0010	0000	1000	
6th shift right	0100	1001	0000	0100	0
7th shift right (carry = 0)	0010	0100	1000	0010	0
8th shift right (carry = 0)	0001	0010	0100	0001	0
<b>CRC error check code</b>	<b>12h</b>		<b>41h</b>		

The final message transmitted including the CRC code is:

Device Address		Function Code		CRC MSB		CRC LSB	
02h		07h		41h		12h	
0000	0010	0000	0111	0100	0001	0001	0010

↑ First bit

Transmission order

Last bit ↑

### Example of a CRC Calculation in the "C" Language

This routine assumes that the data types "uint16" and "uint8" exist. These are unsigned 16 bit integer (usually an "unsigned short int" for most compiler types) and unsigned 8 bit integer (unsigned char).

"z\_p" is a pointer to a Modbus message, and z\_message\_length is its length, excluding the CRC.

Note that the Modbus message will probably contain "NULL" characters and so normal C string handling techniques will not work.

```
uint16 calculate_crc (uint8 *z_p, uint16 z_message_length)
/* CRC runs cyclic Redundancy Check Algorithm on input z_p */
/* Returns value of 16 bit CRC after completion and */
/* always adds 2 crc bytes to message */
/* returns 0 if incoming message has correct CRC */
{
    uint16 CRC = 0xffff;
    uint16 next;
    uint16 carry;
    uint16 n;
    uint8 crch, crcl;

    while (z_message_length--) {
        next = (uint16)*z_p;
        CRC ^= next;
        for (n = 0; n < 8; n++) {
            carry = CRC & 1;
            CRC >>= 1;
            if (carry) {
                CRC ^= 0xa001;
            }
        }
        z_p++;
    }
    crch = CRC / 256;
    crcl = CRC % 256;
    *z_p++ = crcl;
    *z_p = crch;
    return CRC;
}
```

### Example of a CRC Calculation in Basic Language

```
Function CRC (messages) as long
`` CRC runs Cyclic Redundancy Check Algorithm on input message$
`` Returns value of 16 bit CRC after completion and
`` always adds 2 crc bytes to message
`` returns 0 if incoming message has correct CRC

`` Must use double word for CRC and decimal constants

crc16& = 65535
FOR c% = 1 to LEN(message$)
    crc16& = crc16& XOR ASC(MID$(message$, c%, 1))
    FOR bit% = 1 to 8
        IF crc16& MOD 2 THEN
            crc16& = (crc16& \ 2) XOR 40961
        ELSE
            crc16& = crc16& \ 2
        END IF
    NEXT BIT%
NEXT c%
crch% = CRC16& \ 256: crcl% = CRC16& MOD 256
message$ = message$ + CHR$(crcl%) + CHR$(crch%)
CRC = CRC16&
END FUNCTION CRC
```



## Function Codes

Function codes are a single byte instruction to the Slave describing the action to perform.

The following communication functions are supported by Parker SSD Drives' units:

Function Code	Function
01 or 02	Read n bits
03 or 04	Read n words
05	Write 1 bit
06	Write 1 word
08	Loopback
15	Write n bits
16	Write n words

### Read n Bits

Function Code: 01 or 02, (01h or 02h)

#### Command:

Device Address	Function Code 01 or 02	Address of 1st bit		Number of bits to read		CRC	
1 byte	1 byte	MSB	LSB	MSB	LSB	MSB	LSB

The maximum number of bits that may be read is 512.

#### Reply:

Device Address	Function Code 01 or 02	Number of bits to read	First byte of data	....	Last byte of data	CRC	
1 byte	1 byte	1 byte	1 byte	....	1 byte	MSB	LSB

The first data byte contains the status of the first 8 bits, with the least significant bit being the first bit. The second data byte contains the status of the next 8 bits, etc. Unused bits are set to zero.

### Example

From the unit at device address 02, read 14 parameters, beginning at Tag 640:

#### Command:

Device Address	Function Code 01 or 02	Address of 1st bit		Number of bits to read		CRC	
02	01	02	7F	00	0E	8D	97

#### Reply:

Device Address	Function Code 01 or 02	Number of bytes read	First byte of data	Last byte of data	CRC	
02	01	02	27	03	A6	0D

An expansion of the data bytes illustrates the relationship between data and the parameter addresses.

Data byte	1st byte (27h)								2nd byte (03h)							
Param. address	647	646	645	644	643	642	641	640			653	652	651	650	649	648
Bit values	0	0	1	0	0	1	1	1	0	0	0	0	0	0	1	1

**Read n Words**

Function Code: 03 or 04, (03h or 04h)

**Command:**

Device Address	Function Code 03 or 04	Address of 1st word		Number of words to read		CRC	
1 byte	1 byte	MSB	LSB	MSB	LSB	MSB	LSB

The maximum number of words that may be read is 32.

**Reply:**

Device Address	Function Code 03 or 04	Number of bytes read	Value of 1st word		....	Value of last word		CRC	
1 byte	1 byte	1 byte	MSB	LSB	....	MSB	LSB	MSB	LSB

**Example**

For a 650V drive at device address 02, read 2 parameters beginning at Tag 254 (Speed Setpoint and Speed Demand). SPEED SETPOINT is 100.00% and SPEED DEMAND is 50.00%.

**Command:**

Device Address	Function Code 03 or 04	Address of 1st word		Number of words to read		CRC	
02	03	00	FD	00	02	55	C8

**Reply:**

Device Address	Function Code 03 or 04	Number of bytes read	Value of 1st word		Value of last word		CRC	
02	03	04	27	10	13	88	CF	14

**Write 1 Bit**

Function Code: 05, (05h)

**Command:**

Device Address	Function Code 05	Address of bit		Value of bit		CRC	
1 byte	1 byte	MSB	LSB	MSB	LSB	MSB	LSB

The LSB of “Value of bit” is always set to 00. The MSB is used to write the value of the addresses bit. To set a bit value of 1, either transmit 01h or FFh. To set a bit value of 0 transmit 00h.

A device address 00 will broadcast the data to all devices on the network.

**Reply:**

(There will be no reply to a command broadcast to the device address 00.)

Device Address	Function Code 05	Address of bit		Value of bit		CRC	
1 byte	1 byte	MSB	LSB	MSB	LSB	MSB	LSB

The reply to function 05 is the same as the command.

**Example**

Write to the unit at device address 02 setting the parameter with Tag 3 to be TRUE..

**Command:**

Device Address	Function Code 05	Address of bit		Value of bit		CRC	
02	05	00	02	01	00	6D	A9

**Reply:**

Device Address	Function Code 05	Address of bit		Value of bit		CRC	
02	05	00	02	01	00	6D	A9

**Write 1 Word**

Function Code: 06, (06h)

**Command:**

Device Address	Function Code 06	Address of word		Value of word		CRC	
1 byte	1 byte	MSB	LSB	MSB	LSB	MSB	LSB

A device address 00 will broadcast the data to all devices on the network.

**Reply:**

(There will be no reply to a command broadcast to the device address 00.)

Device Address	Function Code 06	Address of word		Value of word		CRC	
1 byte	1 byte	MSB	LSB	MSB	LSB	MSB	LSB

The reply to function 06 is the same as the command.

**Example**

For a 650V drive at device address 02, write 20.0 to ACCEL TIME (Tag 258).

**Command:**

Device Address	Function Code 06	Address of word		Value of word		CRC	
02	06	01	01	00	C8	D8	53

**Reply:**

Device Address	Function Code 06	Address of word		Value of word		CRC	
02	06	01	01	00	C8	D8	53

**Diagnostic Loopback**

Function Code: 08, (08h)

This function provides a means of testing the communications link by means of a “loopback” operation. The data sent to the unit is returned unchanged. Only diagnostic code 0 from the Gould Modicon Specification is supported.

**Command:**

Device Address	Function Code 08	Diagnostic Code 0000		Loopback Data		CRC	
1 byte	1 byte	MSB	LSB	MSB	LSB	MSB	LSB

**Reply:**

The reply to function 08 is the same as the command.

**Example**

Perform a loopback from the unit at address 02 using a data value of 1234h.

**Command:**

Device Address	Function Code 08	Diagnostic Code 0000		Loopback Data		CRC	
02	08	00	00	12	34	ED	4F

**Reply:**

Device Address	Function Code 08	Diagnostic Code 0000		Loopback Data		CRC	
02	08	00	00	12	34	ED	4F

### Write n Bits

Function Code: 15, (0Fh)

**Command:**

Device Address	Function Code 0F	Address of 1st word		Number of bits to write		Number of data bytes (n)	Data	CRC	
1 byte	1 byte	MSB	LSB	MSB	LSB	1 byte	n bytes	MSB	LSB

The maximum number of bits that may can be transmitted is 512.

A device address 00 will broadcast the data to all devices on the network.

**Reply:**

(There will be no reply to a command broadcast to the device address 00).

Device Address	Function Code 0F	Address of 1st word		Number of bits written		CRC	
1 byte	1 byte	MSB	LSB	MSB	LSB	MSB	LSB

**Example**

Write to the Slave unit, at device address 02, 14 parameters beginning at Tag 640 the values 1, 1, 1, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0.

**Command:**

Device Address	Function Code 0F	Address of 1st word		Number of bits to write		Number of data bytes (n)	Data	CRC	
02	0F	02	7F	00	0E	02	see below	83	06

Data byte	1st byte (27h)							
Param. address	647	646	645	644	643	642	641	640
Bit values	0	0	1	0	0	1	1	1

Data byte	2nd byte (03h)							
Param. address			653	652	651	650	649	648
Bit values	0	0	0	0	0	0	1	1

**Reply:**

Device Address	Function Code 0F	Address of 1st word		Number of bits written		CRC	
02	0F	02	7F	00	0E	E4	5C

**Write n Words**

Function Code: 16, (10h)

**Command:**

Device Address	Function Code 10	Address of 1st word		Number of words to write		Number of data bytes (n)	Data	CRC	
		MSB	LSB	MSB	LSB			MSB	LSB
1 byte	1 byte					1 byte	n bytes		

The maximum number of words that may can be transmitted is 32.

The first 2 bytes are data with the required value of the first parameter, MSB first. Following pairs are data for the consecutive parameter addresses.

A device address 00 will broadcast the data to all devices on the network.

**Reply:**

(There will be no reply to a command broadcast to the device address 00).

Device Address	Function Code 10	Address of 1st word		Number of words written		CRC	
		MSB	LSB	MSB	LSB	MSB	LSB
1 byte	1 byte						

**Example**

650V drive: write to the Slave unit at device address 02

Tag 258           ACCEL TIME = 20.0

Tag 259           DECEL TIME = 15.0

**Command:**

Device Address	Function Code 10	Address of 1st word		Number of words to write		Number of data bytes (n)	Data	CRC	
		MSB	LSB	MSB	LSB			MSB	LSB
02	10	01	01	00	02	04	see below	31	27

Data (200) for Tag 258		Data (150) for Tag 259	
00	C8	00	96

**Reply:**

Device Address	Function Code 10	Address of 1st word		Number of words written		CRC	
		MSB	LSB	MSB	LSB	MSB	LSB
02	10	01	01	00	02	11	C7

## Error Response

The MODBUS protocol defines the response to a number of error conditions. A Slave device is able to detect a corrupted command or one that contains an incorrect instruction, and will respond with an error code.

With some errors, the Slave devices on the network are unable to make a response. After a wait period, the Master will interpret the failure to reply as a communications error. The Master should then re-transmit the command.

A Slave device that has detected a corrupted command, or a command that contains an incorrect instruction, will respond with an error message. The error message has the following syntax:

Device Address	Function Code	Error Response Code	CRC	
1 byte	1 byte	1 byte	MSB	LSB

The Function Code byte contains the transmitted function code but with the most significant bit set to 1. (This is the result of adding 128 to the function code.)

The error response code indicates the type of error detected. The following error response codes are supported by Parker SSD Drives' units:

Code	Error	Description
01	Illegal Function	The requested function is not supported by the slave.
02	Illegal Data Address	The address referenced in the data field is not an allowable address for the Slave
03	Illegal Data Value	The value referenced in the data field is not allowable in the addressed Slave location
06	Host Busy	The slave cannot process the request at this time. Try again later.
07	NAK	Rejected for an unspecified reason.

## Wait Period

There are several errors for which the Slave devices on the network are unable to make a response:

- If the Master attempts to use an invalid address then no Slave device will receive the message
- For a message corrupted by interference, the transmitted CRC will not be the same as the internally calculated CRC. The Slave will reject the command and will not reply to the Master.

After a wait period, the Master will re-transmit the command.

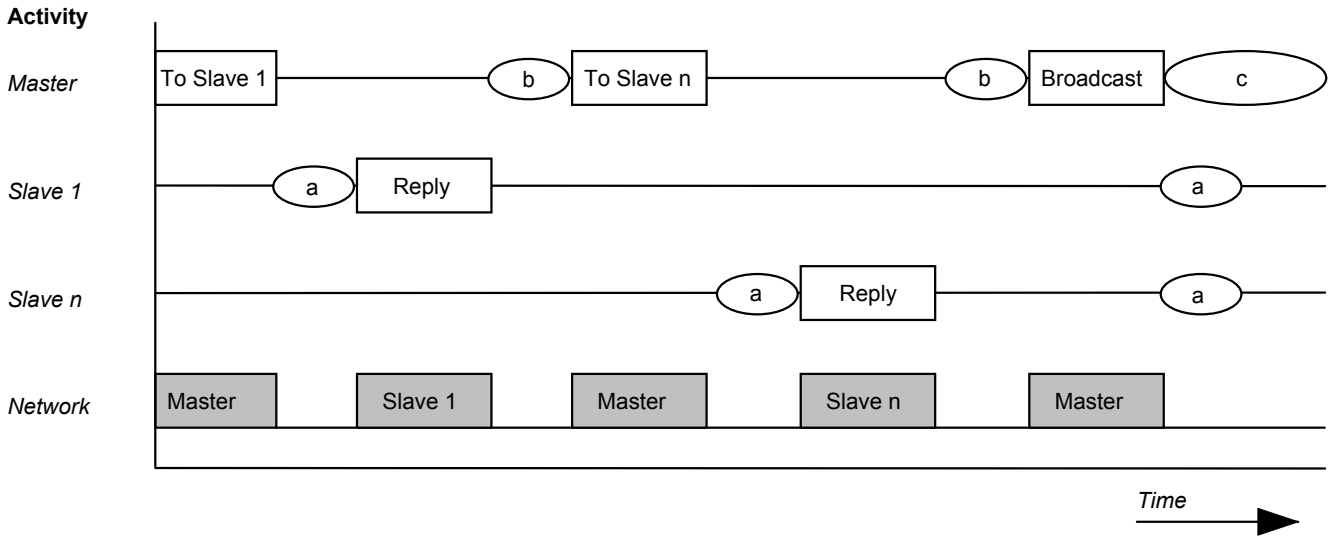
A wait period is also required after a broadcast communication to device address 0.

**IMPORTANT:** Failure to observe the wait period after a broadcast will negate the broadcast message.



## Typical Transmission Line Activity

This diagram illustrates a typical sequence of events on a Modbus transmission line.



- Period "a" The processing time (latency), required by the Slave to complete the command and construct a reply. This is typically 2 milliseconds.
- Period "b" The processing time required by the Master to analyse the Slave response and formulate the next command.
- Period "c" The wait time calculated by the Master for the Slaves to perform the operation. None of the Slaves will reply to a broadcast message.

## MODBUS RTU Parameter Mapping

### 1. MODBUS RTU Prime Set

Mnemonic	Description	Range (HEX values)	Access
9901	Instrument Identity	0650, 1650 or 2650 0650 = 650 Frames 1, 2 & 3 1650 = 650V Frames 1, 2 & 3 2650 = 650V Frames C, D, E & F	Read Only
9902	Main Software Version	0000 to FFFF	Read Only
9903	6051 Software Version	0000 to FFFF (0000 if not fitted)	Read Only
9904	Comms Interface Software Version	0000 to FFFF	Read Only
9908	Bootloader Software Version	0000 to FFFF	Read Only
9909	Last Tag Number	0000 to FFFF	Read Only

### 2. Command/Status

Mnemonic	Description	Range (HEX values)	Access
9911	Command	see below (!1)	Write Only
9912	State	see below (!2)	Read Only
9913	Save Command	see below (!3)	Write Only
9914	Save State	see below (!4)	Read Only

#### !1 : Command

Write-only: used to modify the state of the drive and to load configuration data from non-volatile memory.

HEX Value	Description
7777	Reset Command. Acknowledges failed restore. Loads and saves default Product Code and default Configuration (Application 1).
0101	Restores Saved Configuration from drive's non-volatile memory.
0110	Restores Default Configuration (Application 0)
0111	Restores Default Configuration (Application 1)
0112	Restores Default Configuration (Application 2)
0113	Restores Default Configuration (Application 3)
0114	Restores Default Configuration (Application 4)
0115	Restores Default Configuration (Application 5)
4444	Exit Configuration Mode
5555	Enter Configuration Mode

#### !2 : State

Read-only: used to determine the major state of the drive.

HEX Value	Description
0000	Initialising. (Powering up )
0001	Corrupted Product Code and Configuration
0002	Corrupted Configuration
0003	Restoring Configuration
0004	Re-Configuring Mode
0005	Normal Operation Mode

<b>!3 : Save Command</b>	
Write-only: used to save the configuration and product code in non-volatile memory.	
HEX Value	Description
0000	Reset Command. Acknowledges (clears) any previous save error.
0001	Saves Configuration to drive's non-volatile memory.
0100	Saves Product Code to drive's non-volatile memory.

<b>!4 : Save State</b>	
Read only: used to determine the progress of a non-volatile saving operation.	
HEX Value	Description
0000	Idle
0001	Saving
0002	Failed

### 3. Tag Access

Each parameter is directly mapped to four MODBUS registers: two of these represent it as a single data bit, and the other two represent it as a 16-bit signal or unsigned data word.

This allows a parameter to be read and written using the MODBUS bit functions (01, 02, 05 and 15) or word functions (03, 04, 06 and 16).

For example, the parameter with Tag 65 in the drive is mapped to register:

#### Bit Functions

{0}0065 as a COIL STATUS REGISTER for access using functions : (01, 05, 15)

{1}0065 as an INPUT STATUS REGISTER for access using function (02)

:

#### Word Functions

{4}0065 as a HOLDING REGISTER for access using functions : (03, 06, 16)

{3}0065 as an INPUT REGISTER for access using function : (04)

### 4. Encoding

Reading a parameter which is not of type BOOLEAN using a bit function (01 or 02) will return 1 if the value is non-zero. Writing to parameter which is not of type BOOLEAN using a bit function (05 or 15) will set the value to either 0 or 1 if the limits of the parameter allow this.

### 5. Reading parameter attributes

This feature is available in version 5.x onwards.

The following parameter attributes may be read by adding a corresponding offset to the parameter tag number as shown.

Offset	Modbus register range	Attribute
0	0 to 1999	Parameter value
2000	2000 to 3999	Minimum value
4000	4000 to 5999	Maximum value
6000	6000 to 7999	Data type and write qualifier

The data types are reported in the bottom 8 bits of the returned value. The write qualifiers are returned in the upper 8 bits of the returned value.

## Data types

ID	Data Type
0	32-bit fixed point
1	Boolean
2	32 bit integer
3	16 bit word
4	Null terminated character string.

## Write Qualifiers

Qualifier	Description
0	Parameter is writable
1	Parameter may only be written to when the drive is in configuration mode
2	Parameter is read-only.

## 6. Accessing data as 32 bits.

This feature is available in version 5.x onwards.

Within the 650 many parameters are held as 32 bit values. These may be read and written using pairs of Modbus registers. To access the parameters in this way the mapping of tag numbers to Modbus register addresses has been extended.

To access a parameter as a 32-bit value, use the Modbus register pair starting at the address given by  $((\text{parameter tag number}) * 2) + 10000$ . Always access the lower register first.

Parameters of data type 0, (32-bit fixed point), will be returned in IEEE floating point format when accessed as 32-bit value. These parameters should be written to in this format.

ASCII Table												
BINARY				b <sub>6</sub>	0	0	0	0	1	1	1	1
				b <sub>5</sub>	0	0	1	1	0	0	1	1
				b <sub>4</sub>	0	1	0	1	0	1	0	1
b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	HEX	0x	1	2	3	4	5	6	7
0	0	0	0	<b>x0</b>	NUL	DLE	SP	0	@	P	`	p
0	0	0	1	<b>1</b>	SOH	DC <sub>1</sub>	!	1	A	Q	a	q
0	0	1	0	<b>2</b>	STX	DC <sub>2</sub>	"	2	B	R	b	r
0	0	1	1	<b>3</b>	ETX	DC <sub>3</sub>	#	3	C	S	c	s
0	1	0	0	<b>4</b>	EOT	DC <sub>4</sub>	\$	4	D	T	d	t
0	1	0	1	<b>5</b>	ENQ	NAK	%	5	E	U	e	u
0	1	1	0	<b>6</b>	ACK	SYN	&	6	F	V	f	v
0	1	1	1	<b>7</b>	BEL	ETB	'	7	G	W	g	w
1	0	0	0	<b>8</b>	BS	CAN	(	8	H	X	h	x
1	0	0	1	<b>9</b>	HT	EM	)	9	I	Y	i	y
1	0	1	0	<b>A</b>	LF	SUB	*	:	J	Z	j	z
1	0	1	1	<b>B</b>	VT	ESC	+	;	K	[	k	{
1	1	0	0	<b>C</b>	FF	FS	,	<	L	\	l	
1	1	0	1	<b>D</b>	CR	GS	-	=	M	]	m	}
1	1	1	0	<b>E</b>	SO	RS	.	>	N	^	n	~
1	1	1	1	<b>F</b>	SI	US	/	?	O	_	o	DEL