

17 October 2003

31 March 2004 rev. B

Thruster Controller Protocol

Programming Model

The software interface is based on a register model of the controller, with commands to read and write registers. This results in a very simple and efficient protocol.

The host sends requests of the form

```
Read <Register Number> and Write <Register Number> <Value>
```

and receives responses of the form

```
Acknowledge OK <Value>
```

```
Acknowledge NOTOK <Reason>
```

- Form realisation varies according to the serial interface used.
- Register numbers are in the range 0 – 255.
- Values are in the units specified individually for the registers.
- Reasons returned when commands fail are one of¹:
 - 1 Register not implemented or not accessible
 - 2 Register not writeable
 - 3 Value out of range
 - 4 Unrecognised command
 - 5 Other

¹ First release: Reasons 2 and 3 are given as 1.

Serial Communications Options

The controller can be supplied built with the following communications interfaces:

Option 1A	RS-232C full duplex, no handshake 57600 baud
Option 1B	RS-485 full duplex, no handshake 57600 baud
Option 2	CAN bus 1 Megabaud and 250kbaud.

Only one of Option 1A and 1B may be specified.

Serial Realisations

Option 1, Hardware link set to ASCII

Communication is conversational, commands may be typed by hand using a terminal console program and the response read. The command itself is not echoed².

Register number and values are in decimal or hexadecimal integer format:

33 decimal

0x33 hex

(See individual register definitions for whether signed or unsigned.)

The maximum command length is 50 characters.

Letter case is not significant.

Commands and replies are terminated with <TERM> characters <cr><lf>³.

Fields are separated by one or more spaces. <TERM> does not require a preceding separator..

Upon receipt of any invalid character, including too many characters, the controller will wait for <TERM> before parsing for a new command.

	Console	Replies
Read	R <Register> <TERM>	A <Value> <TERM> N <Reason> <TERM>
Write	W <Register> <Value> <TERM>	A <Value> <TERM> N <Reason> <TERM>
Read Block	G <Register> <TERM>	A <Value0>... <Value7> <TERM> N <Reason> <TERM>
Write Block	P <Register> <Value0>... <Value7> <TERM>	A <Value0>... <Value7> <TERM> N <Reason> <TERM>

² See later for a diagnostic modes.

³ First release: <cr> is sufficient for commands.

Option 1, Hardware link set to BINARY

Communication is in binary bytes.

Register numbers are single unsigned bytes.

Values are 16-bit words, transmitted most significant byte first. See individual register definitions for whether signed or unsigned.

Reasons are unsigned 16-bit words.

Commands are ASCII characters.

Commands and replies are terminated with <TERM> character EOT.

There are no bytes between fields – the read command is three bytes long and the reply is four bytes long.

Upon receipt of any invalid character, including too many characters, the controller will wait for <TERM> before parsing for a new command.

	Console	Replies
Read	R <Register> <TERM>	A <Value> <TERM> N <Reason> <TERM>
Write	W <Register> <Value> <TERM>	A <Value> <TERM> N <Reason> <TERM>
Read Block	G <Register> <TERM>	A <Value0>... <Value7> <TERM> N <Reason> <TERM>
Write Block	P <Register> <Value0>... <Value7> <TERM>	A <Value0>... <Value7> <TERM> N <Reason> <TERM>

Option 2

The TSL distributed communication protocol is used – refer to separate documentation. Each intelligent module in a system has a CAN bus address. Module registers can be read and written at will on a peer-peer basis with high reliability and low latency.

Register Model

The controller has a number of registers, all of which are readable for information and some of which are writeable for parameters or commands. The provisional list of registers is given in the table.

Registers corresponding to all channels may be read in a block by using the block read command on the register marked with an asterisk.

Register	R/W	Name	Description
0		COMMAND	Controller operation word. Commands in the range 0 – 255 are structured as flags for channel operation: Bits 0 – 7 set: causes thruster 0 – 7 to run in any combination, eg 0x82 starts thrusters 1 and 7.
1		STATUS	Flags for controller status Bits 0 – 7 set confirms thruster running. Bit 8 indicates shutdown due to supply under voltage. Bit 9 indicates shutdown due to supply over voltage. Bit 10 indicates excessive board supply power Bit 11 indicates excessive board supply current. Bit 12 indicates shut down due to excessive temperature. Bit 13 indicates communications polling timeout. Bit 14 indicates gate array 0 failed sanity test. Bit 15 indicates gate array 1 failed sanity test.
2		AUTH	Factory code sequence for extended testing. 0x0123 – permits writing of command codes above 255. 0x0124 – permits access to hidden registers. 0x0AAA – permits backup of calibration constants.
3		VERSION	Firmware version
4			
5		LINKV	Supply Voltage (mV)

Register	R/W	Name	Description
6		LINKI	Supply Current (mA)
7		TEMP	Board temperature (C)
8			
9			
10			
11			
12		MODE	0=Current (mA), 1=Speed (rpm) control. Default is current control.
13		K	PID Gain
14		TI	PID Integration Time (ms). 0 ms disables.
15		TD	PID Differentiation Time (ms) – not recommended for use. 0 ms disables.
16			
17			
18			
19			
20			
21			
22			
23			
24*		Set0	Channel 0 Set point (mA or rpm according to mode)
		Set1	Channel 1 Set point (mA or rpm according to mode)
		Set2	Channel 2 Set point (mA or rpm according to mode)
		Set3	Channel 3 Set point (mA or rpm according to mode)
		Set4	Channel 4 Set point (mA or rpm according to mode)
		Set5	Channel 5 Set point (mA or rpm according to mode)
		Set6	Channel 6 Set point (mA or rpm according to mode)
		Set7	Channel 7 Set point (mA or rpm according to mode)
32*		Speed0	Channel 0 Speed (rpm)

Register	R/W	Name	Description
40*		Speed1	Channel 1 Speed (rpm)
		Speed2	Channel 2 Speed (rpm)
		Speed3	Channel 3 Speed (rpm)
		Speed4	Channel 4 Speed (rpm)
		Speed5	Channel 5 Speed (rpm)
		Speed6	Channel 6 Speed (rpm)
		Speed7	Channel 7 Speed (rpm)
		Current0	Channel 0 Current (mA)
		Current1	Channel 1 Current (mA)
		Current2	Channel 2 Current (mA)
		Current3	Channel 3 Current (mA)
		Current4	Channel 4 Current (mA)
		Current5	Channel 5 Current (mA)
		Current6	Channel 6 Current (mA)
Current7	Channel 7 Current (mA)		
48*		Limit0	Channel0 Current Limit (mA)
		Limit1	Channel1 Current Limit (mA)
		Limit2	Channel2 Current Limit (mA)
		Limit3	Channel3 Current Limit (mA)
		Limit4	Channel4 Current Limit (mA)
		Limit5	Channel5 Current Limit (mA)
		Limit6	Channel6 Current Limit (mA)
		Limit7	Channel7 Current Limit (mA)
64*		Ramp0	Channel0 set point ramp rate (rpm/s according to mode)
		Ramp1	Channel1 set point ramp rate (rpm/s according to mode)
		Ramp2	Channel2 set point ramp rate (rpm/s according to mode)
		Ramp3	Channel3 set point ramp rate (rpm/s according to mode)
		Ramp4	Channel4 set point ramp rate (rpm/s according to mode)
		Ramp5	Channel5 set point ramp rate (rpm/s according to mode)

Register	R/W	Name	Description
		Ramp6	Channel6 set point ramp rate (rpm/s according to mode)
		Ramp7	Channel7 set point ramp rate (rpm/s according to mode)
...			
135		CoVersion0	Gate array 0 logic version
136		CoVersion1	Gate array 1 logic version

All register values are unsigned, apart from speeds and set points. For the latter, sign refers to direction of thrust. Actual direction needs to be verified on the ROV after installation, as there is the possibility of mis-phased wiring, and sensorless control cannot determine absolute direction.

The authorisation register codes are to prevent inadvertent corruption of hidden registers and non-volatile storage.

Watchdog

If a period of 500ms elapses without any register being accessed, the controller will turn off all channels. Therefore as a minimum during normal operation the status or other register should be polled or set points written on a frequent basis.

Typical Operation

Operation is very straightforward. Typically the thrust will be controlled as a demand from the ROV guidance system, so that current control mode is sufficient. Increasing the set point magnitude increases thrust. It is not really necessary to control speed or torque directly.

The controller board is turned on, a check is made by reading some registers, then the thruster parameters are set and finally the thrusters are started and controlled as required.

```
Apply power to controller.  
Wait 500ms.  
Flush serial port of random chars/start-up banner  
Read version register.  
Read status (0 if no problems).  
  
Set current limits for thrusters.  
Set operating control loop mode.  
Set control set points.  
Write command to start any combination of thrusters.  
Loop:  
    [Read status]  
    [Read actual speed]  
    Update set points  
Write command to stop thrusters.
```

Notes

Control

TSL's controller board implements sensorless control of the thrusters.

With sensorless control only three power wires are required, which for marine thrusters makes a major improvement in reliability and simplicity, compared to an additional five wires and Hall-effect sensors required by traditional controllers.

Sensorless control does pose some problems at low speed and during speed reversal. These are mitigated by special algorithms and the fact that thrust is negligible at low speed, ie low speed is not normally a useful part of the working range for a vehicle.

Speed Control

TSL's controller board handles speed reversal transparently to the user software. A speed recording will show that during reversal the speed first reduces to a 'dead-band' level, then brakes to a halt, followed by a restart in the other direction, with control picking up again at the opposite side of the dead-band.

If operation within the deadband is attempted the controller will simply work on its boundary; if for any reason the motor should stall it will be automatically restarted.

Current Control

(Speed control is actually a loop around current control.) This gives the fastest response; during a speed reversal the motor immediately brakes to a halt and restarts in the opposite direction.

Inertia

The combined rotor inertia and water swirl make speed reduction slower than speed increase. Faster speed reduction requires braking – the electrical absorption of rotor mechanical energy. Because neither the controller nor the user power supply can absorb power, the controller acts to dissipate energy in the resistance of the motor leads and windings. With the current controller this capability is limited due to the high currents involved and the danger of regenerating a high supply voltage, although the initial software release is not at the limit of what can be achieved.

Simultaneous starting

The user should note that the thruster current during starting is several amps. If all eight thrusters are started together, and particularly if the set point is large, it is possible the controller will stop with a supply power or current trip. Therefore staggered starting is advisable.

Supply Current

This is estimated as the sum of motor currents. Since the latter are measured from switching waveforms the estimate is reasonable but not exact. Measurements in mA are for consistency of units but do not imply high precision; similarly for supply voltage in mV.

Debugging

The communications interface has proven extremely reliable under very prolonged and intensive testing. Any problems are most likely to be in the commands transmitted and interpretation of the replies. If Option link 3 is made AFTER power up, then any controller NOTOK reply will be suffixed by the command that it is replying to. This can be very useful at a terminal.

If Option link 1 is made BEFORE power-up, then a general command-line interface with echo is presented on the serial port. This is limited for the end-user to a few commands that under instruction from the factory can give further diagnosis.

In all communications modes, after power up, a short banner is output on the serial port, giving the software version. Therefore in normal use the user's software should flush or initialise its serial input buffer just before issuing the first command.