Quatech, Inc.

Quatech, Inc.
5675 Hudson Industrial Pkwy.
Hudson, Ohio 44236

Phone 330 655 9000
Fax 330 655 9100
www.quatech.com

# SDS – Serial Device Server

## Intellisock SDK

P/N 940-0187-100

# Table of Contents

# Introduction

## Overview of Sockets Interface

This manual describes how to setup, send, and receive data from the serial ports of a network attached Serial Device Server (SDS) using TCP/IP sockets programming.

### Reader is expected to be familiar with...

IP Sockets programming for the operating system that will be running on the host system. This manual will not describe how to open, close, and communicate over a TCP/IP socket connection.

Serial port communications using a standard 16550 UART. This manual assumes that the reader is familiar with the register set of a standard 16550 compatible UART.

# TCP Sockets Interface

The SDS and host communicate using standard TCP/IP protocols. The host computer will act as a client initiating a connection to the SDS that will be acting as a server listening for a connection request from possibly multiple hosts. Each serial port on an SDS is controlled by a single TCP port starting at port 5000. See example scenario in figure 1. As long as the TCP port is opened by a host the same port can not be opened by another host. After the host opens a TCP port specially formatted commands and data packets are used to setup the serial port operating parameters and exchange data. The command structures used to communicate with individual serial ports on the SDS are described in the following sections.

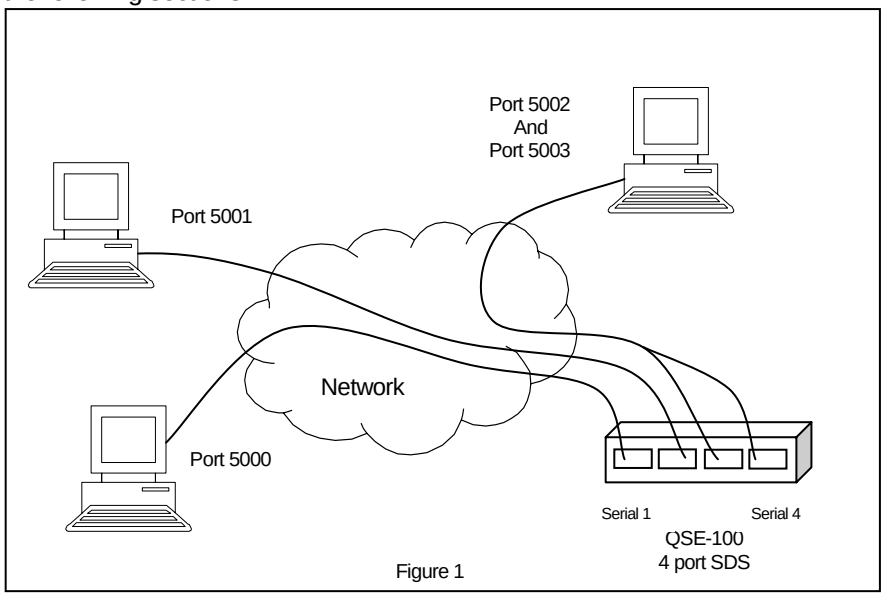



Figure 1

## Serial Port Control Interface

After a TCP Port connection has been opened to an SDS. Serial Port Control Commands must be sent to the SDS to configure the operating parameters of the serial port UART. Figure 2 shows the sequence of commands that need to be sent to setup and send data to an SDS serial port. See the **Command Reference** section for the format and description of commands exchanged with the SDS.

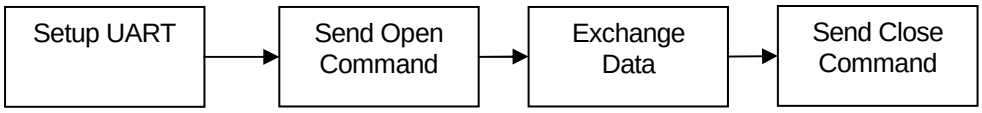



Figure 2

### Setup UART

Before the UART is opened to allow data exchange the operating parameters of the UART must be setup. During this phase commands should be sent to set baud rate, parity, number of start bits, and number of stop bits, Hardware flow control, and Software flow control.

### Send Open Command

Sending the open command enables the serial port to start sending and receiving serial data.

### Exchange Data

Data present on the serial port will automatically be formatted into data transfer packets as described in the **Command Reference** and sent to the host. The host can format data transfer packets and send them to the SDS to be transmitted on the open serial port.

### Send Close Command

When the data exchange is complete the close command should be sent. This command will stop the transmission and reception of serial port data.

## Advanced Options

### Packet Acknowledgements

The packet acknowledgment feature can be enabled to allow a host application to have control over the amount of data that has been transmitted to the SDS but not sent out over the serial port. It also controls the amount of data that the SDS will transmit to the host before the packets of data are acknowledged by the host. When the packet acknowledgment feature is enabled a packet is acknowledged by a packet ACK message sent to the host when all of the bytes in a packet have been successfully sent out the

serial port. The host also must send a packet ACK to the SDS when a data packet received has been processed.

### MSR/LSR Status Insertion

When the MSR/LSR status insertion feature is enabled the host is notified of changes in the UART Modem Status Register (MSR) or Line Status Register (LSR) by embedding the status changes into the incoming data stream. The method of embedding the MSR/LSR status change notifications into the data stream are described in the **Command Reference** section under the Data Transfer command.

### Heartbeat Timer

The Heartbeat Timer is used by the SDS to verify that a connection is still good and in use by a host computer. When the Heartbeat timer is enabled the host computer must periodically send down a Connection Integrity Check command to keep the SDS from timing out a TCP connection and closing it.

## Command reference

Communication between the host and the SDS will occur as outlined in the following sections.

### Message format

All messages transported over the TCP connection will follow the format as show below. The SDS will respond to any command requiring a response by echoing the command back to the host with the data portion filled in. In its command to the SDS, the host should set the Length field as appropriate for the expected response.

| 0x1b | 0x1b | 0x1b | Reserved | Sequence # | Command | Destination Port | Length | Data |
|---|---|---|---|---|---|---|---|---|
| 8 bits | 8 bits | 8 bits | 8 bits | 16 bits | 8 bits | 8 bits | 16 bits | Length x 8 bits |
| Header | | | | | | | | |

**Header** Set of three 0x1b characters that indicate the beginning of a packet.

**Reserved** Reserved 8 bit field

**Sequence #** 16 bit sequence number incremented for each packet sent in big-endian format. Sequence number is only incremented for data packets. Zero is not valid.

**Command** Action to be performed (see section 3.2 for description)

**Destination Port** Unused. Set to Zero.

**Length** Number of bytes in data portion of packet in big-endian format.

**Data** Command specific data.

## Serial Port Setup Commands

### SET BAUD RATE = 0x42 = 'B'

This request is used to set the baud rate. The value passed is a 32-bit unsigned integer in big-endian format representing the actual baud rate. The device will use this value to calculate the appropriate baud rate divisor. The SDS will echo this command back to the host with the actual baud rate set or zero if the baud rate could not be set.

| Data Byte | Value | Description |
|---|---|---|
| 0-3 | 0 – 0xFFFFFFFF | Baud rate |

### GET BAUD RATE = 0x62 = 'b'

This request is used to get the baud rate. The value returned is a 32-bit unsigned integer in big-endian format representing the actual baud rate.

| Data Byte | Value | Description |
|---|---|---|
| 0-3 | 0 – 0xFFFFFFFF | Baud rate |

### HARDWARE FLOW CONTROL MASK = 0x48 = 'H'

This function notifies the SDS that it must react to hardware flow control.

Data byte 0 is a bit mask containing the bits of the Modem Status Register that must be SET in order for data to be transmitted. Transmit flow control can be set to use one of the following DSR, CTS, or CD control lines.

Data byte 1 is a bit mask containing the bits of the Modem Control Register that the SDS may CLEAR in order to prevent the remote device from transmitting more data. Receive flow control can use RTS or DTR.

The SDS will echo this command back to the host.

To use RTS/CTS hardware flow control the following values should be used.

Data Byte 0 = 0x10 Data Byte 1 = 0x02

To use DTR/DSR hardware flow control the following values should be used.

Data Byte 0 = 0x20 Data Byte 1 = 0x01

| Data Byte | Value | Description |
|---|---|---|
| 0 | 0x00 – 0xFF | MSR bit mask |
| 1 | 0x00 – 0xFF | MCR bit mask |

### BREAK CONTROL = 0x4B = 'K'

This function notifies the SDS that it must assert or remove the BREAK condition. All transmission, including TX_IMMEDIATE operations, is suspended while BREAK is asserted. The SDS will echo this command back to the host.

| Data Byte | Value | Description |
|---|---|---|
| 0 | 0x00, 0x01 | 0=clear BREAK<br>1=set BREAK |

MODIFY REGISTER = 0x4D = 'M'

This request is used to set or clear bits in a UART register. A UART register bit will be set to '1' if the corresponding bit in the SET mask is set to '1'. A UART register bit will be set to '0' if the corresponding bit in the CLEAR mask is set to '1'. '0' bits in the SET and CLEAR masks have will have no effect on the UART register. It is illegal to indicate both SET and CLEAR for any particular bit. The SDS will echo this command back to the host.

| Data Byte | Value | Description |
|---|---|---|
| 0 | 0x00 – 0x07 | UART Register Number |
| 1 | 0x00 – 0xFF | SET Mask |
| 2 | 0x00 – 0xFF | CLEAR Mask |

SET REGISTER = 0x52 = 'R'

This request writes the value to the UART register. The SDS will echo this command back to the host.

| Data Byte | Value | Description |
|---|---|---|
| 0 | 0x00 – 0x07 | UART register number |
| 1 | 0x00 – 0xFF | Register value |

GET REGISTER = 0x72 = 'r'

This request reads the current status of a UART register. The SDS will echo this command back to the host with the register value filled in.

| Data Byte | Value | Description |
|---|---|---|
| 0 | 0x00 – 0x07 | UART register number |
| 1 | 0x00 – 0xFF | Register value |

SOFTWARE FLOW CONTROL ENABLE = 0x53 = 'S'

This function notifies the SDS that it must enable and manage software flow control.

When the SDS receives an XOFF character data transmission will be halted. When the XON character is received, data transmission will resume. The third byte of the command determines whether Xon/Xoff characters received by the SDS are included in the data stream or thrown away.

The SDS will send Xoff and Xon characters to impose receive flow control in the case where an overrun would soon occur because the host has not cleared previously received data. The device may stop and start the received data flow at its discretion without notifying the host.

The SDS will echo this command back to the host.

TX_IMMEDIATE operations are NOT suspended by software flow control.

Typical values for Xoff and Xon characters are 0x13 and 0x11.

| Data Byte | Value | Description |
|---|---|---|
| 0 | 0x00 – 0xFF | XOFF character |
| 1 | 0x00 – 0xFF | XON character |
| 2 | 0,1 | 0 = Keep Xon/Xoff Characters<br>1 = Remove Xon/Xoff Characters |

SOFTWARE FLOW CONTROL DISABLE = 0x73 = 's'

This command notifies the SDS that it must disable software flow control applied to it's transmit and receive operations.

Set Word length, Stop bits, and Parity = 0x50 = 'P'

This command is used to set the serial ports word length, number of stop bits, and parity. The SDS will echo this command back to the host. *Not supported in SDS firmware revision 4.01 and earlier.*

| Data Byte | Value | Description |
|---|---|---|
| 0 | 0,1,2 | 0 = No parity<br>1 = Even parity<br>2 = Odd parity |
| 1 | 5,6,7,8 | Word length |
| 2 | 0,1 | 0 = 1 stop bit (Word length = 5,6,7,8)<br>1 = 1.5 stop bits (Word length = 5)<br>1 = 2 stop bits (Word length = 6,7,8) |

## Connection Management Commands

Max Outstanding Packet Count = 0x41 = 'A'

This command enables the packet ACK feature and sets the maximum number of un-acknowledged packets that can be sent between the host and SDS before halting transmission and waiting for packets to be acknowledged. This command is sent by the host with the maximum number of packets it can buffer and returned by the SDS with its maximum packet count it can buffer. The value is a 16-bit unsigned integer in big-endian format. *In SDS Firmware revision 4.01 and earlier the packet acknowledgment feature is automatically enabled and cannot be disabled.*

| Data Byte | Value | Description |
|---|---|---|
| 0,1 | 0-65534 | Max outstanding packets |

Packet ACK = 0x61 = 'a'

This command is sent by the host or SDS to acknowledge packets that have been processed. The packet being acknowledged is referenced by the sequence number contained in data byte 0 of this packet. The SDS will acknowledge a packet when the last byte has been placed in the UART FIFO. The length will contain the number of bytes being acknowledged. A single command can be sent acknowledging multiple packets. In

the case of acknowledging multiple packets the length field will only contain the number of bytes being acknowledged in the last packet. Command packets will not be acknowledged and should not cause the sequence number to be incremented. If a packet is only partially processed (due to a flush command, for instance), the length field will reflect the number of bytes actually processed. The remainder of the packet is considered to be permanently discarded.

| Data Byte | Value | Description |
|---|---|---|
| 0, 1 | 0x00 – 0xFF | ACK # |
| 2, 3 | 0x00 – 0xFF | Length |

CLOSE = 0x43 = 'C'

This function notifies the SDS that the host wishes to close a UART channel. Its purpose is to allow the host and the SDS to synchronize state information. After processing this command the SDS will consider the UART port free to be acquired by the same or another host computer application wishing to open the UART port.

At close the SDS will process no further data transfer requests, disable receive interrupts from the UART, and flush any buffers containing received data waiting to be sent to the host. The SDS will immediately stop transmission of any transmit data received from the host before the close.

The host can specify UART register bits to be set or cleared by the SDS as part of the processing of the close function. The purpose is to allow emulation of typical behavior such as unilaterally dropping RTS and DTR when a port is closed. Adding this information to the Close Channel Command allows the SDS to perform this step at the appropriate time. The format for the "set" and "clear" data fields follows that defined for the Modify Register command. The SDS will echo this command back to the host.

Typically all of the data bytes would be set to zero leaving the UART registers unchanged at close.

| Data Byte | Value | Description |
|---|---|---|
| 0 | 0x00 – 0xFF | SET mask for Line Control Register |
| 1 | 0x00 – 0xFF | CLEAR mask for Line Control Register |
| 2 | 0x00 – 0xFF | SET mask for Modem Control Register |
| 3 | 0x00 – 0xFF | CLEAR mask for Modem Control Register |

DATA TRANSFER = 0x44 = 'D'

Variable length data transfer packet with a maximum packet size of 1460 bytes including header.

Line Status and Modem Status information is inserted into the received data stream by the SDS unless the Disable Status Insertion Command is sent. The embedded status information will move transparently from the SDS through the TCP connection to the host.

When the SDS determines that a status change has occurred, it inserts an escape sequence of 0x1B, 0x1B followed by status information into the received data stream.

| 0x1B | 0x1B | Status type | Register value |
|---|---|---|---|

| | | |
|---|---|---|
| **Status type** | 0x00 | indicates a Line Status change |
| | 0x01 | indicates a Modem Status change |
| | 0x02-0xFE | are reserved for future use |
| | 0xFF | no status |
| **Register Value** | Line or Modem Status Register value | |

If actual data matching the escape sequence is received, the SDS will insert a byte of 0xFF immediately after the 0x1B, 0x1B to indicate that the pattern is real data rather than an escape sequence.

The SDS will ensure that an escape sequence is entirely contained within a single data command packet.

If Status insertion has not been disabled the application on the host computer must scan the received data in each data command packet to look for status changes. If the host finds the escape sequence pattern of two 0x1b characters in a row, it should check the following byte. If that byte is 0xFF, the escape sequence should be treated as real data and the 0xFF should be discarded. Otherwise, the status type and code should be processed and all four bytes of the embedded status information should be discarded.

A change of status reported by the SDS will apply to all bytes following the status change until another status is reported. For example, if a parity error is signaled, the device driver will treat all subsequent received data as having a parity error until another line status is found with the parity error indication removed.

RECEIVER FLUSH = 0x46 = ‘F’

This request causes a complete flush of the receive FIFO of the UART and any received data queued by the firmware. The SDS will echo this command back to the host.

TRANSMITTER FLUSH = 0x66 = ‘f’

This request causes a complete flush of the transmit FIFO of the UART and any transmit data queued by the SDS. The SDS will return a packet ACK for any partially transmitted data packets indicating the amount of data transmitted. All packet ACK messages will be transmitted before the transmitter flush command is echoed back to the host.

OPEN = 0x4F = ‘O’

This function notifies the device that the host wishes to open a particular UART channel. Its purpose is to allow the device driver and the device to synchronize state information. Receiving this message causes the SDS to enable UART interrupts and start processing transmit and receive data.

The six bytes of data are all parameters returned by the SDS for use by the host. The SDS does not use any of the data parameters received from the host.

The heartbeat timer is used by the SDS to detect an idle or broken TCP connection as directed by the user via the web configuration interface. If the value returned is non-zero, the host must send a Connection Integrity Check command ("?") to the SDS using the indicated timer interval. The heart beat timer value is a 32-bit unsigned integer in big-endian format.

| Data Byte | Value | Description |
|---|---|---|
| 0 | 0x00 – 0xFF | Current LSR |
| 1 | 0x00 – 0xFF | Current MSR |
| 2 | 1,2 | 1 = Port acquired<br>2 = Failed |
| 4-5 | 0x0000<br>0x0001 – 0xFFFE<br>0xFFFF | Heartbeat timer disabled<br>Heartbeat timer, in seconds<br>Feature not supported |

### Enable Status Insertion = 0x51 = 'Q'

Enable insertion of MSR/LSR status changes into data stream. If this feature is enabled status insertion strings must be searched for and removed from the data stream. See the Data Command for a description of the Status insertion string. Default state of opened port is status insertions are enabled. The SDS will echo this command back to the host. *Not supported in SDS firmware revision 4.01 and earlier.*

### Disable Status Insertion = 0x71= 'q'

Disable insertion of MSR/LSR status changes into data stream. The SDS will echo this command back to the host. Default state of opened port is status insertions are enabled. *Not supported in SDS firmware revision 4.01 and earlier.*

### TRANSMIT IMMEDIATE = 0x54 = 'T'

This request causes the given character to be the next character transmitted; bypassing any characters queued ahead of it external to the UART.

| Data Byte | Value | Description |
|---|---|---|
| 0 | 0x00 – 0xFF | Character to be transmitted |

### Disable Heartbeat Timer = 0x78= 'x'

This command stops the Heartbeat timer. The Heartbeat timer is automatically started for each new connection unless the Heartbeat Timer is disabled by setting the timer to zero via the Web interface or this command is sent down before the timer expires and the SDS closes the TCP connection. *Not supported in SDS firmware revision 4.01 and earlier.*

### Connection Integrity Check = 0x3F = '?'

Sent by the host computer and echoed back by the SDS to verify connection. The Connection Integrity Check commands can be sent down before the Heartbeat timer expires to keep the SDS from terminating the connection. When the SDS receives the

Connection Integrity Check command the Heartbeat timer is restarted starting the process over again. The Heartbeat Timer can be disabled by setting the timer to zero via the Web interface or by sending the Stop Heartbeat Timer command before the timer expires and the SDS closes the TCP connection.

## Diagnostic Commands

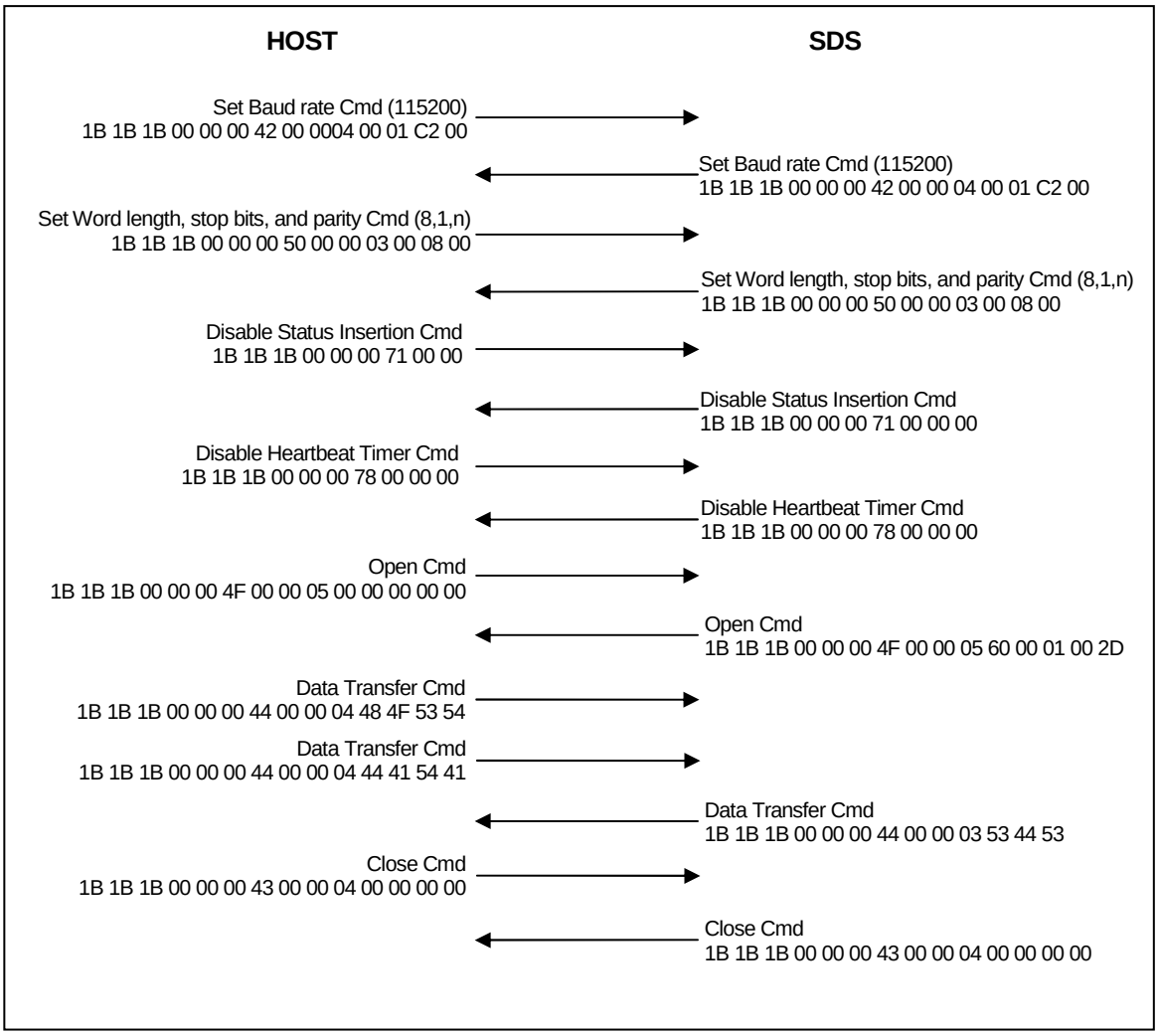
### Device Information = 0x49 = 'I'

Host query the SDS for information.

| Data Byte | Value | Description |
|---|---|---|
| 0,1 | 0x00 – 0xFFFF | Hardware Product ID |
| 2 | 0x00 – 0xFF | Firmware Major Rev |
| 3 | 0x00 – 0xFF | Firmware Minor Rev |

# Appendix

## Simple Message Exchange



# Index