

## Technical Note TN-8

## Subject: A Definition of SLIP

### Background

Serial Line Interface Protocol (SLIP) defines a method of framing messages containing binary data, on asynchronous channels. SLIP has its origins in the UNIX™ world, and was first implemented in Berkley Version 4.2 UNIX around 1985. Whilst there are standards for the carriage of TCP/IP datagrams over LAN's, X25 lines etc, there is no official standard for frame encapsulation on RS232 links. Although SLIP is not an officially recognised standard its use is so widespread that it now an accepted defacto standard.

With SLIP, network interconnections can be implemented at modest data rates without the need for Ethernet, Token ring, or other elaborate hardware and software. Slip drivers are available from many vendors as a means of interconnecting TCP/IP networks via asynchronous RS232 ports on many different computer types including IBM PC's running under DOS. SLIP can also be used by network application writers as a means of parsing packets directly to modem equipment via standard asynchronous ports for transmission in Synchronous format.

Because of its simplicity and direct network access, Trio have implemented SLIP as a configurable driver in the D, E, M & K Series of serial data radio products to facilitate use of the diagnostics and trunked user data.

### SLIP PROTOCOL DESCRIPTION / DEFINITION

The asynchronous serial channel to be used must be configured for eight bit character size, no parity, and one stop. A specific binary code called FEND (Frame End, hexadecimal value=C0) is reserved to define a frame boundary. Should this same code occur in the data message to be transferred across the channel controlled under SLIP, then an escape sequence is used so that the message byte will not be confused for a FEND. This escape sequence, involves replacing the message hexadecimal C0 code with a two byte sequence FESC, TFEND. FESC (Frame Escape) is the binary code hexadecimal DB, and TFEND (Transposed FEND) is binary code hexadecimal DC. Likewise, if the FESC character ever appears in the user data, it is replaced with the two character sequence FESC, TFESC (Transposed FESC). The TFESC is the binary code hexadecimal DD. The following table clarifies this.

ABBREVIATION	DESCRIPTION	HEX.VALUE
FEND	Frame end	C0
FESC	Frame escape	DB
TFEND	Transposed frame end	DC
TFESC	Transposed frame escape	DD

As characters arrive at the SLIP receiver, they are appended to a buffer containing the current frame. Receiving a FEND marks the end of the frame, and consequently, succeeding bytes are considered part of the next frame.

Receipt of a FESC code puts the SLIP receiver into "escaped mode", causing it to translate a following TFESC or TFEND back to a FESC or FEND code, appending it to the buffer, and resuming it's normal state. Receipt of any byte other than TFESC or TFEND while in escaped mode, is an error. No translation occurs, and the SLIP receiver leaves escaped mode. A TFESC or TFEND received while not in escaped mode is treated as an ordinary character and stored accordingly. Reception of consecutive FEND characters, causes no action to be taken (i.e. is not interpreted as zero length frames).

An example of a typical SLIP frame is shown below. The message consists of the string DA,C4,C0,C5,DB,20,BD,DC,DD. The SLIP frame will be:-  
 DA,C4,<FESC>,<TFEND>,C5,<FESC>,<TFESC>,20,BD,DC,DD,<FEND>  
 ==> DA,C4,DB,DC,C5,DB,DD,20,BD,DC,DD,C0