

Re: SocketModem query (3)

Source: <http://coding.derkeiler.com/Archive/General/comp.arch.embedded/2004-08/1481.html>

From: Paul Keinanen (keinanen_at_sci.fi)

Date: 08/19/04

Date: Thu, 19 Aug 2004 11:56:26 +0300

On 18 Aug 2004 14:53:23 GMT, Grant Edwards <grante@visi.com> wrote:

>
><http://groups.google.com/groups?hl=en&lr=&ie=UTF-8&selm=telecom17.86.10%40massis.lcs.mit.edu>
>
> *Pretty close. All high-speed (>=9600bps) modems use*
> *synchronous links between the modems. On top of this, they*
> *use LAP-M (which is an HDLC-based protocol) and V.42 or*
> *V.14 to adapt asynchronous traffic to the modem's*
> *synchronous interface.*
>
> *The LAP-M protocol is basically a reliable transport*
> *protocol which uses retransmission timers and sequence*
> *numbers to achieve reliability. In this protocol, flow*
> *control is done by having the receiver indicate that it's*
> *willing to receive more data, and not necessarily by a*
> *"flow off" message as with XON/XOFF. (Though, confusingly,*
> *V.42 does provide a way to carry RS-232 signals end-to-end.*
> *Sigh!*

This end to end flow control can be quite "elastic", especially when XON/XOFF handshaking is used.

When the receiving DTE (uC) wants to suspend the reception by dropping DTE or sending XOFF, the DCE (receiving modem) sooner or later stops sending bytes to the DTE.

With XOFF, it can take a while before the XOFF has propagated through the modem Rx FIFO and handled by the modem software and only then no new bytes are inserted into the Tx FIFO. Thus the previous contents of the Tx FIFO is transmitted. Thus, after the uC has sent the XOFF, a number of characters will still be received, the number is greater than the sum of the modem UART Rx and Tx FIFO sizes. Thus the uC ISR should be capable of storing at least these characters.

Even if HW handshaking is used, the DCE can send at least one byte, if it has already been transferred to the Tx shift register in the DCE UART. Thus, the uC must still be able to receive one additional

character.

What happens between DTE and DCE in the receiving end does not necessary have any direct implication in the phone line communication between the two DCEs (modems). The transmitting DCE will send frames and the receiving DCE will acknowledge these frames until the internal buffers are filled in the receiving DCE. At this point the receiving DCE stops acknowledging the HDLC frames.

Since the transmitting DCE does not get acknowledges, it tries to resend the frames. If the transmitting DTE (PC) still send bytes, sooner or later the sending DCE buffers are filled and it requests the sending DTE to stop sending by dropping the CTS signal or sends XOFF to the sending DTE.

Thus, the flow control from the receiving DTE to the sending DTE can take seconds if large buffers are used.

However, I would look at the OP's problem in a different way.

Any interrupt service routine in any current uC should be capable of handling the data rates from any phone line modem.

In large downloads, which should include some mechanism for error detection and recovery, the data is divided into blocks and the receiving system acknowledges the reception of the block by sending some kind of positive or negative acknowledge to the sender.

In the simplest case, each received block is acknowledged immediately and only after this acknowledge, the sender resends the old block (if negative acknowledge) or sends the next block (if positive acknowledge from the previous block). Thus, the receiving end interrupt service routine only needs to be able to store the full block size. In many protocols this size can be negotiated at startup. If the normal routines handling the received block needs more time to process it, it simply delays the transmission of the acknowledgement.

Sending the acknowledgement after each received data block greatly reduces the throughput, so many protocols send a few blocks before demanding an acknowledgement. In some protocols up to 7 frames can be sent before demanding an acknowledgment (window size 7), but multiple blocks can acknowledged at once, so usually the data transfer is continuous. This also means that the interrupt service routine Rx buffer must be capable of holding all 7 blocks.

In some protocols this window size can be negotiated to a smaller value. With window size 2, the data flow is usually continuous, but now only two frames need to be buffered. Using two separate buffers, in which the interrupt service routine fills up one buffer, while the other buffer containing the previous block is processed by the normal

comp.arch.embedded: Re: SocketModem query (3)

code. For the next block, the role of these buffers are reversed. The interrupt service routine must be capable of understanding the frame format, so it know, when a complete frame has been received.

Paul