http://groups.google.com/d/topic/comp.dcom.modems/7gxaiOs_P14/discussion

comp.dcom.modems ›

Telebit modem questions answered

9 posts by 7 authors

Super user

3/4/88

A BRIEF TECHNICAL OVERVIEW OF THE TELEBIT TRAILBLAZER MODEM
By Michael Ballard, UNIX Program Manager, Telebit Corp.

Before starting on this document, a caveat: this document is intended to address many of the questions and comments about Telebit modems that have appeared from the user community. We are striving to provide as much information as possible, in such a way that will be useful to the widest group of readers. This is NOT intended to be a Marketing Article, but rather a technical overview for the more sophisticated reader. Its purpose is to inform, not to sell product. If anyone is offended by Telebit taking this action, please mail directly to me first, to avoid cluttering the newsgroup. Thank you.

I would like to provide some background for Unix users considering the use of Telebit's TrailBlazer Plus high speed dialup modem. I served as project manager and principal programmer for Telebit's protocol support developement. The UUCP "g", Kermit, Xmodem and Ymodem protocols are directly supported in the TrailBlazer modem's firmware. Peter Honeyman, co-developer of ATT's HoneyDanBer/BNU UUCP, coded those portions of the TrailBlazer firmware which support the "g" protocol.

The Telebit modem employs a patented multicarrier modulation scheme coined DAMQAM (Dynamically Adaptive Multicarrier Quadrature Amplitude Modulation). A CRC-16 based sliding window protocol with selective retransmission runs on top of this modulation scheme insuring data integrity across the phone line. This telephone line protocol is known as the Packetized Ensemble Protocol or PEP. PEP is the trademark by which all modems employing this technique can be recognized.

This technique (DAMQAM) divides the voice bandwidth into 511 individual channels each capable of passing 2, 4, or 6 bits per baud based on the measured characteristics of the individual frequencies associated with each channel. On a typical phone connection, the modem uses a subset of about 400 of those channels.

Each time the modem connects to a circuit established on the dialup Public Switched Telephone Network (PSTN), the TrailBlazer measures the quality of the connection, and determines the usable subset of the 511 carriers. The aggregate sum of bits modulated on this subset of carriers multiplied times the baud rate yields a bit per second rate that on a local telephone connection (i.e. round trip through your local telco) is 18031 bps. This 18031 bps is then reduced by about 20% to allow for the CRC overhead, to about 14400 bps of data throughput.

Long distance line quality varies with location and carrier, but you can expect this number to be in the 10000 to 17000 bps range under most conditions domestically. By choosing a high quality long distance carrier, you will ensure the best throughput overall.

The modem operates at 7.35 and 88.26 baud, transparently changing baud rates to accomodate the pace and quantity of data traffic. When in "interactive mode" the modem sends data using 11 msec packets (which run at 88.26 baud). Each packet contains 15 bytes of data. In "file transfer mode" the modem uses 136 msec packets (that transfer at 7.35 baud) that contain 256 bytes of data. The TrailBlazer decides which packet size to use on an ongoing dynamic basis. No intervention from the user is required.

At lower speeds, such as 300, 1200, and 2400 bps, the TrailBlazer provides emulation (performed in the DSP section, not by a "chip"

V.22. Both are supported. At 2400 bps, the standard is called CCITT V.22 bis. These speeds are all available with or without MNP Class 3 Error Correction.

The TrailBlazer employs a Motorola 68000 and a Texas Instuments TMS32010 digital signal processor to accomplish this performance. Because of this substantial computer horsepower (about 7.5 MIPS), the TrailBlazer is really a communications processor, rather than a conventional modem.

The software defined architecture produces a flexible product platform that allows broad feature development capabilities while allowing the product's installed base to benefit from those developments by installing upgrade EPROM sets.

All four protocols (Kermit, Xmodem/Ymodem, UUCP), V.22bis support, MNP at low speeds, multiple releases to improve the interactive performance (earlier TrailBlazers utilized only one baud rate), a multitude of RS-232 behavior related features, leased line capabilities, remote command processor access, echo suppressor compensation, increased data rates, and a myriad of user requested features have found their way into current production modems and are available to earlier revisioned modems via the EPROM uprgrade kits.

PEP modems provide a full duplex serial interface to an attached computer, however they employ a half duplex implementation on the telephone line. Telebit refers to this half duplex technique as "Adaptive Duplex". As the name implies, the ownership of the line (i.e. the ability to transmit) adapts to the quantity of data available to send at any single moment. Maximum efficiency is achieved by sending data in a nonstop data stream at 19.2Kbps relying on serial interface flow control to moderate the data flow into and out of the modem.

This allows the maximum amount of data to be available every time a transmitting modem takes ownership of the line. In this way the modem, not the DTE, controls the line turnarounds. The protocol provides a ceiling at about 3k of sent data before a transmitting modem must give up its turn and allow the other modem an opportunity to send. A continuous 19.2Kbps data flow into the modem is required to ensure that there is always 3k of data to send each time a transmitting modem takes its turn. The serial interface speed must exceed the telephone line speed, potentially 18,031 bps, or the maximum efficiency of the modems can not be reached).

UUCP's "g" protocol behavior on dialup lines was a clear contradiction of the desired behavior with the PEP protocol. "g" sends 3 small data packets at time and then waits for the remote UUCP to ACK or NAK their receipt. The resulting throughput when using UUCP and "g" with the TrailBlazer was only a little better than a standard 1200 bps modem. This was unacceptable.

What did we do about it?

The TrailBlazer can be configured to "spoof" the protocol by setting a register (S111) to one of several values. The spoof can support four different protocols: UUCP "g", Xmodem, Ymodem, and Kermit.

"Spoof" means to fool the various protocols into thinking that they are getting their acknowledgment packets from the remote computer, when in reality they are getting them from the modem.

All of these protocols are what are commonly referred to as "send and wait" protocols. This type of protocol builds a packet in computer A, sends it out through the modems, where it is received by computer B. Next, computer B looks at the packet to determine whether or not it arrived intact. If it did, it sends an ACK (acknowledgement) packet back to computer A. If it did not arrive intact, it sends a NAK (non-acknowledgement) packet. In either case, computer A can't send the next packet out until it gets the ACK from the first packet. This is slow!

Since our modems are error-free between the modems, the only place data could get "broken" is between the modems and their respective computers. Let me draw the connection diagram below:

```
          Ca <======> Ta <----------------> Tb <======> Cb

          Ca = Computer A                   Cb = Computer B
          Ta = Telebit Modem A              Tb = Telebit Modem B
```

```
          ====== RS-232 Cable
          ------ Phone Line
```

When we are running our protocol support, we look at the packet coming from Ca. Ta checks the packet for validity and sends the ACK or NAK. Ca can begin building the next packet immediatly upon receipt of Ta's ACK. This results in Ca building and sending packets as fast as it can. Many packets are now forwarded to Tb. Tb now delivers the packets to Cb, observing the rules of the protocol. Tb will deliver the next packet or retransmit the previous packet based on the ACK or NAK received from Cb. Cb ACKs and NAKs are then thrown away so as not to return to Ca.

Protocol support can be configured to run in parallel with data compression enabled. The real world result of this is to increase protocol transfers from 2-3 Kbps to 10-19.2 Kbps.

This covers most of the commonly asked questions about the TrailBlazer. If any of the above information is unclear, or you have questions regarding other aspects of modem technology or performance, send mail to:

```
==========================================================================
Richard Siegel                UUCP: {uunet,ames,hoptoad}!telebit!modems
Senior Systems Engineer       ARPA: telebit!modems@ames.ARPA
Telebit Corporation
==========================================================================
```

Click here to Reply

Bill Mayhew

3/7/88

1. Both my Trailblazer's have General Instruments DSP chips. One modem is about 8 months old with rev 3 firmware. The other is about 3 months old with rev 4 firmware. I take it that the GI DSP chip is a functional equivalent to the Texas Instruments chip?
2. The older modem has some SMD chips on a little daughter board; the newer modem has an AMD "Worldchip". I take it these are then strictly for DTMF generation and call progress detection?

3. Oh yes, here's another. What is the big square (Toshiba, I think) chip in the newer telebit design doing?

Thanks for posting the tech summary.

--Bill

Jonathan Clark

3/9/88

YATB

This one is really for Peter Honeyman to answer, since it is claimed that he did the relevant bit of the Telebit firmware, but the answer may prove to be of general interest.
Many of us have tweaked our uucico's to use a window of 7 outstanding packets, rather than 3. Given this, does it make sense to have two Telebit modes for the 'g'-protocol spoof (one for each window size); and iff it does, are there any plans to implement this?
--
Jonathan Clark jonathan.clark@mtune.att.com, attmail!jonathan
Any affiliation is given for identification purposes only.

The Englishman never enjoys himself except for some noble purpose.

Peter Honeyman

3/10/88

YATB

window size doesn't affect trailblazer throughput -- a window size of 1 gives the same performance as 3 or 7.
in fact, the modem converts a request for a large window size into a request for a window size of 3. it does this transparently, but you can watch it happen with -x9 debugging on.

eliminates the windowing inside the spoof code, but we were unable to assert unequivocally that all versions of uucp would work with a window size of 1. (we tested against a long list of uucp versions, but there are versions to which we didn't have access.) we were confident that 3 would work in all cases, so we stuck with that.

so to answer the question, no, you don't need multiple versions of spoof code to accommodate window size variants, the modem takes care of that (too!) for you.

peter

Carl S. Gutekunst

3/11/88

YATB

In article <38...@umix.cc.umich.edu> ho...@citi.umich.edu (Peter Honeyman) writes:
>window size doesn't affect trailblazer throughput -- a window size of 1
>gives the same performance as 3 or 7.

Peter and I have argued this one in private before -- and I still disagree.

I know the modem is perfectly capable of running flat out with window size 1. But a lot of computers are not -- and in fact, window size 3 at 9600 or 19200 bps is too small. We have many connections where the throughput is obviously constrained by the system's ability to get ack packets out; I think a bigger window size would help immensely.

Of course, a lot of machines that can barely buffer 3 packets at 19200 are going to gag on 7. No problem; they don't have to run a uucico with the window size pushed up.

<csg>

Peter Honeyman

3/11/88

YATB

carl, i'd buy your argument if uucico's throughout the land knew how to piggyback acks. it's admitted by the chesson spec, but i've never seen a version that does it.
if you're not convinced, why not recompile uucico with a window size of one and compare? i'm sure the results would be interesting.

peter

Jerry Aguirre

3/11/88

YATB

In article <16...@pyramid.pyramid.com> csg@pyramid.UUCP (Carl S. Gutekunst) writes:
>In article <38...@umix.cc.umich.edu> ho...@citi.umich.edu (Peter Honeyman) writes:
>>window size doesn't affect trailblazer throughput -- a window size of 1
>>gives the same performance as 3 or 7.
>

>I know the modem is perfectly capable of running flat out with window size 1.
>But a lot of computers are not -- and in fact, window size 3 at 9600 or 19200
>bps is too small. We have many connections where the throughput is obviously
>constrained by the system's ability to get ack packets out; I think a bigger
>window size would help immensely.

I have to agree. If the host were talking at 19200 and able to generate acks reasonably fast then throughput shouldn't suffer. But I have to run my interface at 9600 bps, to a heavily loaded slow CPU.

While it might be able to generate acks fast enough on average to keep the line going it can't do so at any given instant. There are other things like disks and other serial lines to service. Remember that the acks are not being generated by the tty driver at interupt level. They are being generated by a user process. It reduces overhead if the process can process and ack several packets at each context switch

The suggestion that full speed is possible with a window of 1 is ludicrous. The modem would be able to send only 1 packet. It would then have to wait for an ack before it could send another. At a minimum that means 8 bytes of idle time for every 64 bytes received. The line would be idle for 12% of the time. Any time to process the incomming packet and generate the ACK would add to that.

If the connection from the modem to the computer has additional delay then performance really suffers. By example consider dialing into a packet switched network like tymnet or PC persuit. Before switching to a window size of 7, I and others experienced terrible thruput on networks like that. I also run UUCP over lines with a satellite delay. Again, thruput is terrible without a window of 7.

```
3 packets * 64 char/packet * 10 bits/char = 1920 bits

1920 bits / 19200 bps = .1 second to send 3 packets.
```

If the system takes more than 100 ms. to return any ack then performance suffers. (Actually it is 66 ms. because I can't begin generating an ack until after the first packet is received.) A satellite circuit has 700 ms. of delay. Many packet switched networks have several hundred ms. of round trip delay.

If the modem really negotiates the window size with the host then why can't it use the value each host asks for? That way systems that can't buffer 7 packets can leave their window at 3 and systems that need a larger window can do so. (No, 3 is the magic number. Count ye not to 2, nor on to 4 ..... :-)

You are too locked into what the modems are doing internally and not to the host interface. If I ask for a window of 7 then you should give me a window of 7, I might have a good reason.

Jerry Aguirre

Peter Honeyman

3/12/88

YATB

if the wire is slower than the host, windowing is a clear win. but here, the wire is faster than the host, usually a lot faster. the modem immediately fills the window, whatever its size. thereafter, the modem sees a window of one -- host acks, modem sends a packet to fill the window.
your point about context switches is well taken -- it depends on the scheduler's behavior when the host calls write(ack). you can probably convince me that a window of two is worthwhile, but i don't see this argument extending to anywhere near seven.

someone should run an experiment here. (i can't, because i don't have a fast computer with a serial board.)

peter

William E. Davidsen Jr

3/14/88

YATB

In article <17...@oliveb.olivetti.com> jerry@oliveb.UUCP (Jerry Aguirre) writes:
| [...]

> The suggestion that full speed is possible with a window of 1 is
> ludicrous. The modem would be able to send only 1 packet. It would
> then have to wait for an ack before it could send another. At a minimum
> that means 8 bytes of idle time for every 64 bytes received. The line
> would be idle for 12% of the time. Any time to process the incomming
> packet and generate the ACK would add to that.

If the feed to the modem is at 19.2k, since the ack is generated at the modem, and accepted at the modem, the two systems should be able to drive the actual line at the maximum of 11-14k (depending on who's figures you see).

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.