PSR Quarterly

July 1985 #16

PACKET
STATUS
REGISTER

TUCSON AMATEUR PACKET RADIO CORPORATION

PRESIDENT'S CORNER

Lyle Johnson, WA7GXD

Welcome to the new PSR Quarterly! I certainly hope you enjoy the format of this issue. As noted elsewhere, Gwyn Reedy, W1BEL, has assumed editorship of TAPR's Official Publication. This is Gwyn's first PSR, but I think you'll agree it's a fine start.

Remember, PSR is what YOU make it! If you want to see an article about something, write one and send it to the address indicated on the back cover...

The big news this month is, of course, TNC 2. While we announced this device in the last issue, this PSR has more detailed information on the hardware, software and philosophy of the new TNC. If all goes well, this month (July) should see the release ot the first 300 production units.

I think projects like the TNC 2 development effort demonstrate one ot the finest aspects ot TAPR -- people from many places working together under a common banner to accomplish something positive for Amateur radio. TNC 2 is a major undertaking. But dedicated packeteers in New Jersey, Florida, Illinois, Arizona and California brought it about, assisted in testing by others in Texas, Colorado, Maryland, Virginia, Missouri and Connecticut.

As you can see from the above list, TAPR is not merely Tucson. TAPR utilizes talents from a large geographic area. TAPR is a reflection of what we can all do if we work together. Working together, we will bring about Networking. Working together, we will develop low-cost, high-speed radio/modems. Working together, we will assist disaster and other emergency communications. Working together, we will assist in developing highly reliable satellite links. Working together, we will sustain and spread the packet revolution!

TNC 2 also brings out one of TAPR's major limitations. TAPR is unable to crank up a massive production line and make sufficient TNC 2s for everyone to be satisfied. Being a volunteer-based R & D organization, we simply lack the twin resources ot time and money that such an effort would require. Thus, we have to start with an initial production of 300 units, then let those sales generate the revenues to get 300 more ready tor sale, and so on.

Problem: How do you spread 300 units equitably among 1,000 buyers? The answer is obvious: you can't. But this is answer is no solution. A common suggestion is to make them only available to members ot TAPR. This fails for two reasons.

The first is that, in order to remain tax-exempt (sales are for the primary purpose of generating revenues for R & D), we have to make the results of R & D (TNCs) available to non-members on an equal basis with members. Thus, it puts our status with the IRS in jeopardy to make the first units available to members only. Not a pleasant prospect.

The second is manpower. Assuming we could tind a way to make the federal government happy, it takes time (meaning people-power) to weed out orders and verify membership claims. How do you stop someone from just joining when they order and circumventing the whole idea? How do you do it equitably? What is you have 300 units to sell and only 100 members want to buy? Or 400 members want to buy? Which members get cut off? As you can see, Pandora has nothing on us! All I can ask is that you be patient.

We have settled on the following method. We will establish a 'call-in' day and publish the date in advance via electronic media. There will be enough lead time for the word to get around. On the call-in day, the telephone system will be allowed to mediate who gets through. Each caller will be allowed to order only one TNC2.

We are cranking on the TNC 2 project as hard as we can, and will make as many available to everyone as fast as practical. Meanwhile, check out the offerings from other sources. Of course, be sure to ask plenty of questions before you obtain any TNC! Be sure that the unit you purchase meets your needs!

Finally, I would like you all to join me in welcoming Christina Kurz as our new office manager. She will be handling your orders and generally trying to servie your needs at the TAPR Office. As before, don't ask her for technical advice. Send all technical enquiries to the TAPR PO Box with an SASE.

Thank you, and enjoy this PSR!

EDITOR'S COLUMN

Gwyn Reedy, W1BEL

I am honored to have been selected for the editorship of this important publication. I thank the TAPR directors for their confidence.

PSR has now become the PSR Quarterly. Since the first PSR was published many changes have taken place in the packet community. Many monthly local packet newsletters are now published and the ARRL GATEWAY provides coverage of non-technical packet news on a frequent basis. Rather than try to cover all areas of packet activity, PSR Quarterly will concentrate on technical topics and news of TAPR projects and activities. You are encouraged to read one or more of the fine newsletters referenced above, including one that I edit, the FADCA>BEACON.

This issue of PSR has two general themes, the TNC2 and networking. The TNC2 is very exciting to use (I have a Beta test version) and there is much worth waiting for. There are articles by designers of the hardware and software to give you additional insight into this new product. There are a number of approaches to the short term networking solution, and several are presented here for your consideration. Not present in this collection is the 'GATOR-1' and follow-on networking plans of the folks in Florida (a rudimentary level 3), the two port linking plan for California, and many other networking attempts underway throughout the country. The next PSR Quarterly will continue this networking discussion.

There is a review of the Kantronics TNC, because with its new low price, it is bound to have a large impact on the packet community. I estimate the current number of amateur TNCs in existence at about 4000, and it could well double in a year or less. Come on network!

Packet history has been relatively peaceful to date, with only the normal problems of various humans competing for recognition. The efforts at standardization have helped a lot. The packet market is potentially very large, and everyone's efforts should be directed to expanding it. There is no need to compete head-on with any other producer of equipment as is true in a more mature market. Let the strife and hard feelings in the SSTV arena be a lesson to everyone of the need for cooperation. You'll note the lack of recent growth in that mode...

Please begin now to prepare inputs for the October PSR Quarterly. We can use inputs about linking, homebrewed equipment, modifications to existing packet gear, etc. The size of the next issue is only limited by the number and quality of the inputs that are submitted.

A quick word about the status of some TAPR projects.

- The K9NG prototype high speed modem cards have gone through extensive testing, the board has been redrawn, and the boards should be available in August.
- No work has been done on a 220 MHz RF deck design as was previously forecast. Serious volunteers are needed in that area.
- A new run of PAD (frame assembler-disassembler) boards has been made. These piggyback on the Xerox 820 to allow it to function as a TNC (with external modem) or digipeater.
- Improved software for earlier TAPR TNCs is making development progress.

Harold Price sent the following word about his work on updated software for the TNC1:

"Due to other projects, nothing much has been done on 4.0 software in the last few months. But I've actually spent 8hrs/day the last three days on it, so we're finally making progress.

I'm going to try to spent at least 4 hours a day on it till it gets done. I now have the software running under Turbo Pascal, which makes for easy debugging (IBM PC only, haven't tried 8 bit). I've also built a protocol validation testbed, a manual way to generate frames to bounce off the software. It's an easy way to check the degenerate cases.

I'm holding out the multiple connect stuff for last and am hurrying along to get ax25 version 2.0 compatability out at the same time tnc-2 comes out. I'll have the major portion of the bug fixes in as well.

Just to let you know that progress is in fact being made, here is some output from the TNC-1 4.0 code, running on the simulator. The frame types (<SABM>, etc.) are enabled with the MFRAME command, and are more of a trace/education feature. You can see exactly what is happening with a bit less data than a full trace.

```
cmd:C NK6K V NK6K
cmd:NK6K>NK6K,NK6K<SABM> <C>
NK6K>NK6K,NK6K*<SABM> <C>
NK6K>NK6K,NK6K<UA>
NK6K>NK6K,NK6K*<UA>
*** CONNECTED to NK6K VIA NK6K
THIS IS A TEST
NK6K>NK6K,NK6K<I 0,0>:
THIS IS A TEST

NK6K>NK6K,NK6K*<I 0,0>:
THIS IS A TEST

NK6K>NK6K,NK6K<RR 1>
NK6K>NK6K,NK6K*<RR 1>
ANOTHER TEST
NK6K>NK6K,NK6K<I 1,1>:
ANOTHER TEST

NK6K>NK6K,NK6K*<I 1,1>:
ANOTHER TEST

NK6K>NK6K,NK6K<RR 2>
NK6K>NK6K,NK6K*<RR 2>
WB6YMH>NK6K,NK6K<SABM> <C>
WB6YMH>NK6K,NK6K*<SABM> <C>
*** connect request:WB6YMH VIA NK6K
NK6K>WB6YMH,NK6K<DM>
NK6K>WB6YMH,NK6K*<DM>
@
cmd:D
cmd:NK6K>NK6K,NK6K<DISC> <D>
NK6K>NK6K,NK6K*<DISC> <D>
NK6K>NK6K,NK6K<UA>
NK6K>NK6K,NK6K*<UA>
*** DISCONNECTED
```

The connect from WB6YMH was a canned packet sent to the simulator."

****

THE TAPR TNC 2 - WHY?

Lyle Johnson, WA7GXD

The TAPR TNC 1 was an embarrassing success. It established itself as a standard in only a few months. It has spawned a series of clones and work-alikes. It was a vehicle whereby packet gained widespread acceptance among the Amateur radio community worldwide.

TNC 1 accomplished the goals its design team envisioned. But TNC 1 was expensive. $326 with cabinet and shipping in the US, more elsewhere. It was a bit bulky. It only ran on AC power.

There were those packeteers who wanted a less expensive, full-performance TNC. There were many potential packeteers who expressed strong interest in packet radio, but less than $200 worth!

And there were those Amateurs living beyond US borders who drove home the point that importation of equipment is often expensive and time-consuming.

With AEA and Heath "clone" TNCs on the market, it seemed reasonable that TAPR should withdraw. These units sell for $299 to around $490, covering the middle- and upper-price ranges. Certainly, two TNCs in this class is sufficient at this time.

It was time to enter a challenge in the low-cost arena. Something to address the needs of the serious packet enthusiast on a budget. Students. Overseas operators. Public service communicators. Remote digipeater sites. Portable users. Something to draw in those on the fringe of the crowd.

Enter TNC 2. Lightweight (1 kilogram...er, 2 lbs 3 oz), low cost ($195 to your door in the US) and based on easily obtainable parts (for overseas and ham-swap specialists). 12-volt operation for ease of use in home, portable or mobile operation. CMOS ICs in most circuitry for low power consumption.

TNC 2 will be TAPR's source of R & D funding for Networking development. It is an evolutionary step in TNC development. While it didn't achieve one design goal (small enough to "velcro" to the back of an HT!), it seems to have met the other criteria outlined above.

The people at Kantronics had similar goals in mind in development of their Packet Communicator. They chose a slightly different approach than TAPR, and have produced a piece of equipment that will serve the packet revolution well. The recent price reduction for this assembled unit is an exciting event for packet radio.

****

AN OVERVIEW OF TNC 2 SOFTWARE

Howard Goldstein, N2WX

You wouldn't be completely mistaken to credit the Lone Star state (as only their natives can) for the reason TNC 2 exists. The Xerox (tm) 820 boards (practically cheaper than air) had almost enough capability as-is to be a "super-TNC" with but two little things lacking: NRZI related converters and the software. The first difficulty was overcome with Skip Hanson's (WB6YMH) very usable state machine/NRZI flip flop circuit. The second item is the subject of the rest of this article.

WHAT MAKES TNC 2 SO SPECIAL?

Good question! TNC 2 is the only product on the market to fully support the AX.25 level 2 version 2.0 link layer protocol (also referred to as AX25L2V2) as approved by the ARRL. This is important since previous AX.25 level two implementations (supported by TNC 2 as well) were not clearly defined. And besides a thorough definition, version 2.0 link layer offers increased throughput for most connections by eliminating retries of data unless they are specifically asked for ("C" bits"), and preventing some collisions by delaying ACK packets a short amount of time to allow the sender to finish a multiple frame transmission (T2 timer).

Numerous features, many not found anywhere else, enhance the users' ability to follow channel activity, interface the TNC to applications, and control the TNC's actions.

MONITOR ENHANCEMENTS

- TNC 2 maintains a list of the callsign-ssids of up to the last eighteen different stations copied by the board. Also captured is whether the station was heard directly or through a digipeater. Previously this capability was never provided in a user TNC - one had to query a FADCA "-7 like" digipeater or an application running somewhere else to get this function. When the clock is set (calender desribed below) TNC 2 also captures the date and time that stations are copied, collating it into a MONITORED HEARD (MHEARD) list the user may query at any time to gauge channel activity, propagation, or just whether one's buddy is around or not.

- A clock-calender is provided which is used by the TNC firmware to: "Stamp" (i.e. print the date and time of) all printed packets or "Stamp" all *** CONNECTED status messages, in addition to its use in the MHEARD log described above. These "Stamping" functions may be selectively enabled by the user as he sees fit, as they are switched by individual commands.

- Packets containing connect and disconnect commands, that is SABM and DISC commands, are now displayed by the monitor:

```
W1BEL>N2WX  <D>      shows W1BEL sending a
                     DISC packet to N2WX
```

- A whole class of commands and functions replace the MTO and MFROM lists of the TNC 1 firmware. TNC 2 allows you to specify up to eight callsign-ssid pairs in a LANCED CALLS (LCALLS) list. The user may filter, as far as the monitor is concerned, frames from stations one prefers not to see. Bulletin boards, file transfers, even overly prodigous BEACONers may be selectively squelched simply and effectively.

```
cmd:LCALLS AD7I,WB9FLW,N2WX-7
LCALLS   was
cmd:
```

Now the user has specified that if any of these three stations are heard, their frames are intercepted ("lanced") before they travel on to the monitor routines.

```
cmd:BUDLIST ON
BUDLIST  was OFF
cmd:
```

Our user has now instructed the TNC to monitor ONLY the lanced frames, that is, to think of the lanced call list as a list of buddies (BUDLIST). TNC 1 owners recognize this as equivalent to setting MFROM AD7I,WB9FLW,N2WX-7.

- In addition to the familiar ESCAPE translation command and function of TNC 1, TNC 2 can recognize and filter characters the user doesn't want to leave the TNC as information from a monitored packet. Clear screen, cursor positioning, bells (whistles?), in short any of (up to four) characters can be selectively stripped from anything monitored.
- The path taken by monitored frames is (if desired) displayed without going into trace mode. In addition to the list of digipeaters, when the actual retransmission is copied the callsign of the digipeater that sent it is marked with an asterisk:

```
W2VY>W1BEL,N2WX-7,KC2FF-7:printer on go ahead
W2VY>W1BEL,N2WX-7*,KC2FF-7:printer on go ahead
W2VY>W1BEL,N2WX-7,KC2FF-7*:printer on go ahead
```

OPERATIONAL ENHANCEMENTS

- Every parameter related to the actual use of the TNC is preserved by the battery backed up RAM (including BTEXT, UNPROTO, and all the new features except the clock).
- 9600 baud (synchronous and asynchronous side) sustained in full duplex without any mods.
- One software controllable LED is used to show if the TNC is connected, another LED lights it there are unacknowledged packets in the TNC.
- A new mode (NEWMODE) that controls the TNC's entries between "cmd:" and CONMODE according to the connect state.
- True BREAK detection permits an immediate, untimed escape from Transparant connect mode.
- Software RECEIVE flow control in transparent mode is available with the TRFLOW command.
- A CONNECT MESSAGE (CMSG) is sent to anyone who connects with you, presenting up to 120 characters of text that the operator has stored.
- In an AX25L2V2 connection, TNC 2 checks an inactive link at (T3) intervals with the other station to verify the link's existence. If not AX25L2V2, as for instance if TNC 2 is talking to an older TNC, an interval of inactivity causes the TNC to try to disconnect. In either case the user specifies the inactive limit.
- Calibration includes a "dotting" command useful for setting transmit levels and tuning demodulators.
- /R identification in ASCII at proper intervals occurs automatically for digipeaters
- The destination for transmitted beacons is taken from the UNPROTO setting, along with the path.

UNCLASSIFIED FEATURES

- An enhanced version of the same user interface in the TAPR TNC 1, AEA PKT-1, Heath 4040 and Kantronics Packet Communicator.
- Written entirely in Z-80 (tm) assembly language for speed, compactness, ease of access to implemenation tools.
- TNC 2 has its prompts and error messages in one location (with extra room) to ease language translation problems. This design attitude is also present in the calander, where European format for date is also available, as it is throughout the whole TNC 2 project. Adherence to international standards (where available) was paramount. Moreover, the hardware was designed specifically to avoid using difficult-to-obtain parts. However, TAPR welcomes input from "DX" stations on how we can make the TNC more useful for amateurs in those locations.
- Functions in TNC 1 not supported by TNC 2 include: FSK CW ID, EPROM programmer code, MTO, and Vancouver protocol.

****

AN OVERVIEW OF TNC 2 HARDWARE

Paul Newland, AD7I

The micro-computer hardware for TNC 2 tries to do only a few things, but do those things very well. It doesn't provide all the features of other full blown TNCs but what it does provide is not compromised.

TAPR chose a Z80 (tm) processor for TNC 2 based on several considerations. Some of those considerations include its low cost, availability in CMOS, multiple vendors, its maturity, and perhaps most importantly, the large amount of development software available for it. Some folks think the Z80 is not an "advanced processor" and that is probably a true statement. However, that doesn't change the considerations listed above.

ROM and RAM are simple byte-wide (tm) devices with the exception that the RAM is backed up with a lithium battery so that its data is retained when the 12 volt supply is removed from TNC2. Special care was taken with the battery protection circuitry and it represents a third generation design.

There is only one I/O device for TNC2: a SIO/0. One channel of the SIO provides asynchronous communications to the user's terminal. The other channel provides a HDLC port (with external state machine for clock recovery) for packet reception and transmission. Again, the SIO was chosen because of its maturity, CMOS availability, dual channel capability and low cost. Several unused modem status indicators are used to drive two of the four front panel LEDs that act as status indicators.

The modem is almost a direct copy from TNC1. TAPR decided to use this modem instead of a chip like the AMD7910 because we felt that it provided us with the necessary flexability to do HF (narrow shift or wide shift) as well as allowing us to control current consumption. Also, parts for this modem (as well as the rest of TNC2) would not be difficult to get.

The remaining function, the power supply, is conventional. The user supplied 12 volts is used to drive a 5 volt regulator as well as one side of the RS232 drivers. The output of the 5 volt regulator drives the microcomputer and the modem. An LM556 is used to invert the +12 volts to about -7 volts (under load) to drive the negative side of the RS232 drivers. A low drop-out voltage regulator, formed by a transistor and an op-amp, is used to give a regulated voltage of -5 volts for the modem.

That is a briet description of TNC2 hardware. The important thing to keep in mind about this system is that it is designed for low cost and, for the functions it was designed to provide, it gives high performance.

****

2764 EPROM PROGRAMMER FOR THE TNC 1

Mel Whitten, K0PFX
3219 Haas Ave.
Bridgeton, MO 63044

This is the design information for the TAPR EPROM programmer. It was originally planned for production as a kit so that TNC1 owners could reburn their EPROMs with updated firmware as it became available, however the device was never produced. Herewith is the information needed to construct the programmer. There is a possibility I may be able to produce some printed circuit boards. Please write to me at the above address if you are interested.

DESIGN FEATURES

-Sequential programming can begin from any specified starting address rather than from the Zero Base address only.
-Non-sequential addresses may be programmed or verified using the direct addressing mode.
-Actual address applied to the EPROM can be logically verified.
-Operating and programming supply voltage can be checked for presence.

CIRCUIT DESCRIPTION

The 2764 programmer consists of two controlled power supplies, four quad binary latches, two octal tri-state buffers, mode decode logic, a 28 pin ZIF (zero insertion force) socket, and a 26 pin header.

Power and control is supplied to the programmer by a flat-ribbon cable to the 26 pin header located on the printed circuit board. Data lines PA0-PA7 are bi-directional. They are used to send data to the programmer board during programming, address data to be set or initialize the EPROM address registers. These same lines are also used to read back Vcc, Vpp and address register status.

Control lilnes PB0-PB7 are used to set the modes for the various programmer operations.

A positive 5-volt DC supply is brought to the board through the header for control logic and Vcc of the 2764. The Vcc line is ON/OFF controlled by the logic signal PB5 (pin 21). When PB5 is in its low state (0 state), Q1 (VFET) remains off which removes bias from Q2 (TIP-32) and effectively turns off the 5 volts at the Vcc pin of the 2764. When PB5 is high (1 state), then Q1 conducts and forward bias is applied to Q2 which becomes saturated causing 5 volts to appear at Vcc. A Vcc ON indicator (LED) is provided allowing the status of Vcc to ge read back as a logic 1 Bit 6 (PA 6).

Vpp for the programming mode is supplied by three 9-volt transistor (216) batteries connected in series. During programming, Vpp is plus 21.0 volts and should not exceed 21.5 volts. In the READ modes, Vpp is normally maintained at the Vcc level of plus 5 volts. Diode D1 supplies normal bias for the READ mode. Control line PB4 keeps the 21.0 volt line shut off when it is in the low state. The operation and biasing of Q3 and Q4 is similar to the operation of Q1 and Q2 as previously discussed. An additional address control line (A12) is provided to allow for programming the 27128 EPROM.

A10 (LM317) is an adjustable 3-terminal regulator set within the range of 21 volts output and trimmed to exactly 21.0 volts by the Vpp adjust control. Diodes D3 and D4 protect the regulator from excessive reverse voltage when Q4 is turned off. D2 prevents loading of the Vpp bias when Q4 is turned off. When PB4 goes high (1 state), Q3 and Q4 turn on allowing the LM317 to regulate a programming Vpp bias of 21.0 volts. Diode D1 during this time prevents Vpp from damaging the Vcc supply. The Vpp programming level is logic sensed by A6 (E & F) and is used to operate the Vpp ON indicator by turning on Q5. Vpp level status (0 = ON) can be monitored through bit 7 (PA7).

The following discussion of the Logic Control Modes of the programmer will refer to Table 1. Logic signals /PGM (PB6), /CE (PB0), and /OE (PB1) are used to control the various modes in which the programmer can be operated. Auxiliary signals ASEL (PB3) and CLK (PB2) are used with the state control signals to address and generate timing strobe pulses.

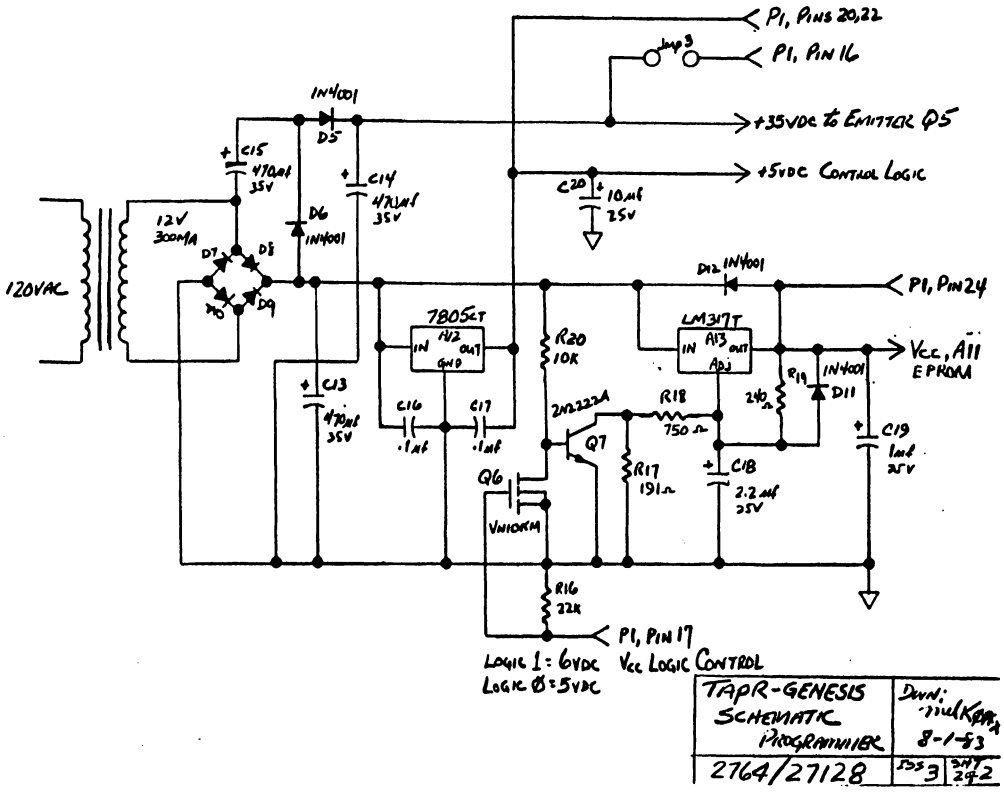
When /PGM and /CE are both in the high state, the 2764 is effectively deselected. This allows the other control signals to write and read data to and from the Address Latch Registers (A1-A4). The /OE signal determines whether address data is written to or read from the programmer. When /OE is a logic high, the PA0-7 lines should contain data to be written into the Address Latch Registers. When /OE is a logic low, the power supply status and Address Latch Registers can be read with the PA0-7 data lines. When the ASEL signal is a logic low, it selects the lower port register and when it is a logic high, it selects the upper port registers. The CLK signal is used as a Write or Read strobe and must be toggled form a low to a high and back to a low state. During read operations, CLK must be held high to read stable data, then returned low to tri-state the PA0-7 lines. The /CE determines when a CLK is used as a Write or a Read control signal in the Address/Status mode. When a /CE is in the low state (logic 0), the programmer is switched into the EPROM operation mode. Address latch registers are now set and toggling the CLK signal will cause the registers to increment (add 1 to the current address number) in a binary sequence for each low to high transition (toggle). The CLK signal must be stable or in a steady state during the Read or Program operation on the 2764. All 2764 operations are defined for /CE in the LOW state.

****

The high altitude, wide area coverage digipeater serving the Pike Peak region of Colorado was put in service in May ot 1984. Service extends from Cheyenne WY on the north, east to Kansas and south to Raton Pass on the New Mexico border. Plans are afoot to link westerly to Grand Junction and Salt Lake.

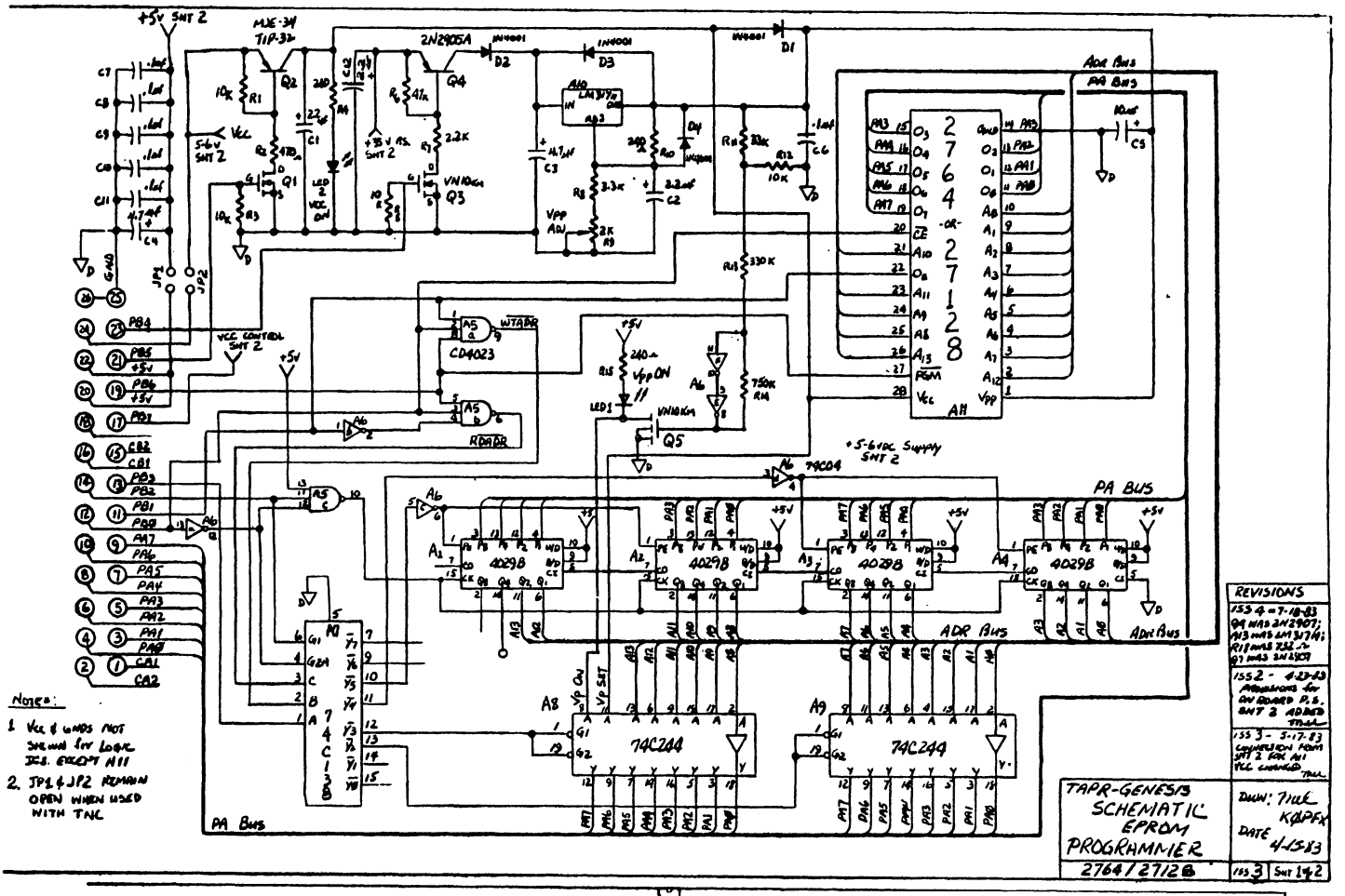
The equipment is the TAPR TNC1 Beta board of N0CCZ operating as N0CCZ-1 on 145.01 mhz. RF is provided by an Icom 22A. The installation was engineered by John Conner, WD0FHG.

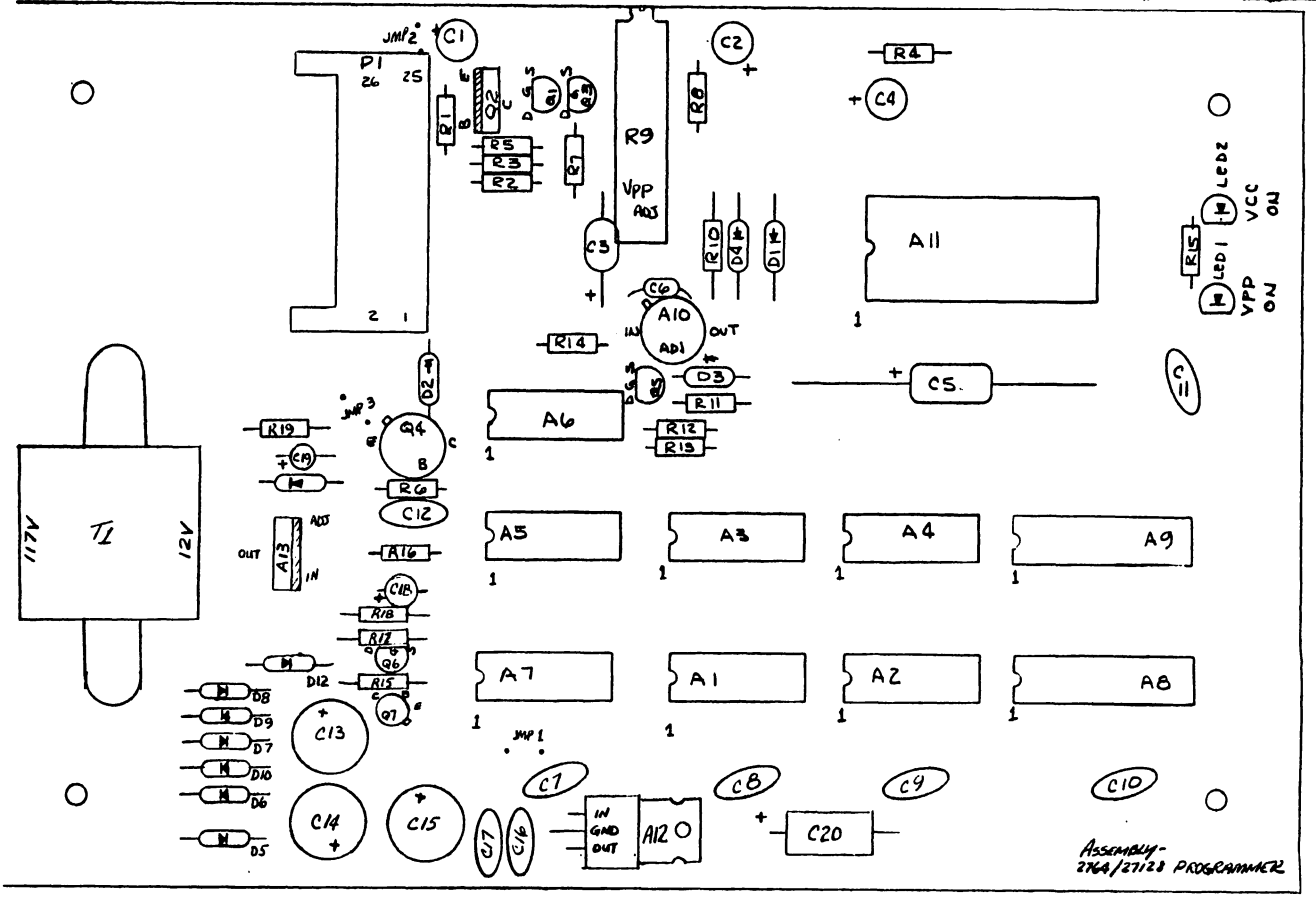
In the 13 plus months that the TAPR Beta board has been on the air it has never required a reset. Does this tell you something about the TAPR TNC hardware and sottware?



TAPR-GENESIS SCHEMATIC PROGRAMMER	Dwn: MulKey 8-1-83
2764/27128	1353 247

ITEM	QTY	DESCRIPTION	PART NUMBER	DESIGNATION	REMARKS
54	6	SILICON RECT., 50V	1N 4001	D5-D12	T.I.
53	1	NPN SILICON TRANS TO-18	2N2222A	Q7	T.I.
52	1	N-CHAN VMOS FET TO-237	VN10KM NONE	Q6	SILICONIX
51	1	TANT, RADIAL, 10µf, 25V	199D199D106K0025CA2	C20	SPRAGUE
50	1	FILM RES, 240Ω, 1/4W, 5%	CR25-241	R19	MEPCO/ELECTRA
49	1	MF RES, 750Ω, 1/8W, 1%	RN55D 7500	R18	MEPCO/ELECTRA
48	1	MF RES, 191Ω, 1/8W, 1%	RN55D 1910F	R17	MEPCO/ELECTRA
47	1	FILM RES, 22K, 1/4W, 5%	CR25-223	R16	MEPCO/ELECTRA
46	1	FILM RES, 10K, 1/4W, 5%	CR25-103	R20	MEPCO/ELECTRA
45	1	TANT, RADIAL, 1µf, 25V	199D105K0025CA2	C19	SPRAGUE
44	1	TANT, RADIAL, 2.2µf, 25V	199D225K0025CA2	C18	SPRAGUE
43	1	CER DISC, Y5T, .1µf, 16V	UK16-104	C17	CENTRALAB
42	1	CER DISC, Y5T, .1µf, 50V	UK50-104	C19	CENTRALAB
41	1	ELECT, RADIAL, 470µf, 35V	ECE-A1VV4715	C13,14,15	PANASONIC
40	1	3-TERM REG, ADJ, TO-39	LM317T	A13	NATL
39	1	3-TERM REG, 5V, TO-220	LM340T5	A12	NATL
38	1	TRANSFORMER, 117V TO 12V	RS 273-1385A	T1	RADIO SHACK
37	1	LED LAMP, GRN, T-1¾	CM4-3846	LED-2	GEN INST
36	1	LED LAMP, RED, T-1¾	CM4-2848	LED-1	GEN INST
35	1	HEADER, SOLDER TAIL, R/A	3429-1002 26 PIN	P1	3M ELECT PROD
34	1	ZIF DIP RECEPTACLE, 28 PIN	228-3596	A11	3M TEXTOOL
33	1	ZIF DIP SOCKET, 28 PIN	228-3545	A11	3M TEXTOOL
32	2	DIP SOCKET, TIN, 20 PIN	DILB20P-108	A8,9	BURNDY
31	2	DIP SOCKET, TIN, 14 PIN	DILB14P-108	A5,6	BURNDY
30	5	DIP SOCKET, TIN, 16 PIN	DILB16P-108	A1-4,6	BURNDY
29	4	SILICON RECT, 50V	IN 4001	D1-D4	T.I.
28		TRIMPOT, CERMET, 2K, POT	3006Y-1-202	R9	BOURNS
27	1	PNP SILICON PWR TRANS TO22	TIP32	Q2	~~MOT~~ NATL
26	2	PNP SILICON TRANS. TO-5	2N2905A	Q4	T.I.
25	3	N-CHAN VMOS FET TO-237	VN10KM	Q1,3,5	SILICONIX
24	1	FILM RES, 750K, 1/4W, 5%	CR25-754	R14	MEPCO/ELECTRA
23	1	FILM RES, 330K, 1/4W, 5%	CR25-334	R13	MEPCO/ELECTRA
22	1	FILM RES, 33K, 1/4W, 5%	CR25-333	R11	MEPCO/ELECTRA
21	1	FILM RES, 3.3K, 1/4W, 5%	CR25-332	R8	MEPCO/ELECTRA
20	1	FILM RES, 2.2K, 1/4W, 5%	CR25-222	R7	MEPCO/ELECTRA
19	1	FILM RES, 47K, 1/4W, 5%	CR25-473	R6	MEPCO/ELECTRA
18	2	FILM RES, 240Ω, 1/4W, 5%	CR25-241	R4,10,15	MEPCO/ELECTRA
17	1	FILM RES, 470Ω, 1/4W, 5%	CR25-471	R2	MEPCO/ELECTRA
16	4	FILM RES, 10K, 1/4W, 5%	CR25-103	R1,3,5,12	MEPCO/ELECTRA
15	5	CER DISC, Y5T, .1µf, 16V	UK16-104	C7-11	CENTRALAB
14	1	CER DISC, Y5T, .1µf, 50V	UK50-104	C6	CENTRALAB
13	1	TANT, AXIAL, 10µf, 10V	162D106X0010D2	C5	SPRAGUE
12	1	TANT, RADIAL, 4.7µf, 35V	199D475X0035DA2	C4	SPRAGUE
11	1	TANT, AXIAL, 4.7µf, 10V	162D475X0010D2	C3	SPRAGUE
10	1	TANT, RADIAL, 2.2µf, 35V	199D225X0035CA2	C2, C12	SPRAGUE
9	1	TANT, RADIAL, 2.2µf, 10VDC	199D225X0010AA2	C1	SPRAGUE
8	1	UV ERASABLE PROM	2764/27128	A11	INTEL/AMD
7	1	3-TERM ADJ REG TO-39	LM317H	A10	NATL
6	2	CMOS OCT BUF & LINE DRV	MM74C244N	A8,9	NATL
5	1	CMOS 7b DECODE-DEMULT	H74SC138P	A7	SUPERTEX INC.
4	1	CMOS HEX INVERTER	MM74C04	A6	NATL
3	1	COS/MOS NAND GATE	CD4023B	A5	RCA
2	4	COS/MOS PRESET UP/DWN CTR	CD4029B	A1-A4	RCA
1	1	PRINTED CIRCUIT BOARD-PTH	2764/27128	PROGRAMMER	


TAPR-GENESIS
SCHEMATIC
EPROM
PROGRAMMER
2764/27128
DWN: MIKE KAPEX
DATE 4-15-83
ISS 3 SHT 1 of 2
REVISIONS
ISS 4 = 7-18-83
ISS 2 = 4-23-83
ISS 3 = 5-17-83
NOTES:
1. VCC & GNDS NOT SHOWN FOR LOGIC ICs. EXCEPT A11
2. JP1 & JP2 REMAIN OPEN WHEN USED WITH TNC
PA BUS
ADR BUS
74C244
4029B
CD4023
74C04


A11
TI
12V
A10
A6
A5
A3
A4
A9
A7
A1
A2
A8
A12
LED2 VCC ON
LED1 VPP ON
Assembly-
2764/27128 PROGRAMMER

HF OPERATION WITH THE TAPR/HEATH/AEA TNC

John Dewey KA9CAR
4904 Drive In Lane
Crystal Lake, Il 60014

I like to operate HF packet on the weekends; but I still want to have access to the local VHF network during the week and Sunday evenings. Retuning the modem is an easy but time consuming task. This modification lets you change from the wide VHF to the narrow HF tone sets with the flip of a switch.

If you compare the values of the components for the HF and VHF headers, there are only two differences. These are R46 which determines the demodulator bandwidth and C21 which is part of the post-detection (low pass) data filter. The values of R36 and R38 set the transmit tones. The receive center frequency is the same for both HF and VHF.

With the addition of a few components you can quickly switch from the VHF tone pair to the HF tone pair. The schematic shows how. All connections to the modem, except the grounds, were made on the headers (U34 and U35). TAPR designed the TNC with only one ground connection between the audio and digital circuits. Don't mess that up by using the wire wrap area ground for the modification!

I purchased the 3PDT switch and 2 multi-turn resistors at Radio Shack. Mount the resistors by bending the "rear" leads and using a drop of instant glue on the pots so that they are standing up with one soldered onto the ground connector for R59 and the other at the plated through hole near C15. Drops of colored nail polish can be used to pair R36 with R36A and R38 with R38A. Cut the "top" leads off the resistors and solder hook up wire to the "center"pins. R35, and R37 are lifted from the headers at one end only and wired to the switch. R46 is removed and mounted on the switch with R46A. Unless you have a 4 pole switch, leave the post-detection data filter (C21) at the value for 1200 baud.

Tune up follows the normal routine with the entries of 1/1152 and 2/1024 being used when the switch is in the "A" position. While you are at it recheck the standard tones. A tuning indicator such as the one in PSR #11 is a great help when trying to tune in packets on HF. I found that connecting the "spare" LED to go out when the regular DCD is on provided a very useful "bar DCD" which is adequate for casual packet tuning.

The HF parameters of HBAUD 300 and MAXFRAME 1 should be stored in one of the 2 NOVRAM banks. If you are a BBS or MSO the DWAIT should be set to 4. (Ed. note - See "HF Packet Parameter Recommendations" following this article.) When resetting the HF parameters the error message "HDLC can't init" will come up. There is an error in the software. Version 3.2 seemed to work anyway; but in order to get 3.3 to work you must switch to "conversation" mode and send a few characters to your dummy load. You can then return to command mode and the TNC will operate correctly.

For improved input filtering I was able to get the 500 hz CW filter on my Kenwood TS-520SE to work in SSB by changing the existing diode switch wiring. Due to the difference in carrier frequency used for CW and LSB it came out with a passband that passes the TNC tones in both transmit and receive. If you need information on that modification contact me. It is a 30 minute no cost modification.

Because the two tones on HF are only 200 hz apart it does not matter what header you have on the input filter to the TNC. Calling frequencies for HF packet are 14.103, 10.149, 7.097, 3.642 and 3.630 mhz.

HF Packet Parameter Recommendations

In case you haven't noticed, the packet activity on 14.103 has reached the point of forced QSYs of individual QSOs because of BBS activity.

Due to this increased activity, some thought should be given to the differences between HF and VHF operation. A few parameter changes will significantly improve channel throughput not only for BBS stations but for individual QSOs on the same frequency.

When the frequency is relatively quiet and a good path exists, everything proceeds just as it would on VHF. But, as is often the case with HF, the trouble starts when things are less than perfect.

What happens when the channel is in a condition that requires retries? The following examples apply to a BBS or file transfer and can occur if you manually type in data while your TNC is holding off due to a DCD.

The TAPR software MAXFRAME is set to 4 by default. This means that up to a maximum of 4 frames (packets) can be sent in a continuous transmission. On a good channel this is very efficient as the end flags for one packet become the start flags for another saving some bit time on the frequency. Also, the receiver CAN send one ACK for all 4 packets.

Lets assume the sending station sets his MAXFRAME at the maximum value of 7 and has a total backlog of 25 packets. Due to propagation many packets will collide with unheard packets and others will suffer bit errors (fading, static, etc). The system sends a transmission of 7 packets and receives an ACK for the first but an error report on the second. The TNC dutifully resends all packets after the first and adds another up to the MAXFRAME value of 7. Due to conditions, on the next try, the first packet is in error but some of the others make it through. All 7 packets are again re-transmitted as the retry is set up to resend all packets after the last one received correctly. This process continues until the sending station melts down his finals, gets all 25 packets through, or retries out. If there is an error in the second frame of each transmission, the end result would be sending 7*25 or 175 packets to get the file through. Had MAXFRAME been set to 1 and every packet sent twice, the total would have been only 2*25 or 50 packets. Watch a BBS or file transfer some time on HF or a noisy VHF circuit and see what happens. The result is the same lengthy data transfered over and over.

Recommendations:
1. Set MAXFRAME to 1.
2. Set your BBS DWAIT to 4.
3. Set PACLEN to 80 or less.
4. PERM, PERM, PERM.
5. QSY for long file transfers.

As conditions improve, the recommended values can be changed to optimize throughput.

(This article also appeared in the June FADCA>BEACON)

Schematic appears on page 19

BULLETIN BOARDS, HOSTS, GATEWAYS AND GLUE

Dr. Tom Clark, W3IWI
6388 Guilford Rd.
Clarksville MD 21029

This series of articles will address one exciting form of packet communication, the Packet Bulletin Board System (PBBS). I will begin by telling the story of the operation of one PBBS-- W3IWI in Clarksville MD. Perhaps it will prove of interest to those of you in other parts of the country. To tell this story, perhaps you will excuse me while I reminisce about the history of packet radio as seen thru my myopic eyes.

The Washington DC area was one of the early hotbeds of packeteering in the USA, due largely to the efforts of Paul Rinaldo (W4RI) and the organization he helped found, AMRAD. I became involved in these activities in 1981 when I got my first Vancouver TNC. At that time there were 3 or 4 of us active and on the air. AMRAD's repeater served as a congregating point with intermixed packets and voice. Dave Borden (K8MMO) occasionally put his CP/M "host" system on the air and we showed that this new system worked by playing Adventure. Packet Radio showed promise but it was clearly below "critical mass" in terms of being a tool that did something useful for the user. It was in the hands of the experimenters and innovators; the applicators had not yet arrived.

During 1982, the bulk of my time was tied up in AMSAT, trying to recover from our loss of the Phase-3A satellite, building its replacement and trying to obtain a ride for the first UoSAT satellite. As OSCAR-10 neared completion I began thinking about a follow-on and dreamed up the concept of PACSAT. About the same time, TAPR was beginning to develop its TNC and I had a phone call from Den Connors (KD2S). The discussions with Den resulted in AMSAT hosting a meeting in autumn, 1982 wherein two significant developments occurred. The first was the adoption of the proposal by WB4JFI/AMRAD outlining the AX.25 protocol, and the second was a preliminary design for PACSAT.

TAPR's software mavens (KV7D, NK6K and KD4NL) picked up the AX.25 ball and ran with it. Meanwhile the Tucsonians (WA7GXD, N0ADI, KD2S, KV7B et al) and St. Louisians (WB9PLW et al) got nearly 200 of the TAPR "Beta" TNCs out before spring of 1983 and I served as DC-area "Beta" coordinator and I put my CP/M system on the air as an occasional "host", set the terrestrial packet DX record (ZL on 10 meters), and piddled around. The applicators STILL hadn't arrived.

By late '83 we had shown that packets could be sent thru the OSCAR-10 satellite and we had logged onto "host" systems in Los Angeles and San Francisco. Mike Bruski (AJ9X), Rod Hart (WA3MEZ) and Terry Fox (WB4JFI) had put occasionally put RCPM systems (originally designed for telephone modem use) on the air when it was convenient. However, none of these systems were dedicated to packet use. A few new users came on the air and saw that there was no "magnet" to keep their interest level up; we were languishing below the "critical mass" level.

Then in the spring of 1984 came some very interesting news from Texas and Arkansas; Xerox was having a "clearance sale" on their 820 280 computer boards at $50 each. The news spread like wild-fire. Everybody and his brother (including me) bought a few (why a few? -- at that price, I might as well buy a couple of spares!). For me the justification (as if I needed one) for getting another computer was that it would be a good "engine" for putting a dedicated PBBS on the air. But it ended up on the shelf gathering dust.

Meanwhile, one of those who had also bought some 820's was an old VHF buddy of mine, Hank Oredson (W0RLI). Hank had recently moved to Massachusetts and was getting the packet radio bug. Hank had always been a super assembly language programmer, and he decided to write a BBS program specifically for use on packet radio. In late August I dropped in on Hank and saw his system on the air. We cloned a disk and I returned to Maryland. I started getting all the pieces off the shelf, dusting them off, and making cables. The W3IWI PBBS went on the air Sunday night, September 7th, 1984. Since that time, it has been in full-time (well, actually there has been about 3-4% down-time) service for the packet radio users in the Baltimore/Washington area. During the fall and winter we started building EASTNET in earnest. In October, Jon Pearce (WB2MNF) brought a similar system up in Medford NJ (southeast of Philly, about 100 miles away). With the help of Bob Bruninga's WB4APR-6 digipeater (near the MD/DE border) and Joe Fisher (KC2TN), Jon and I started passing BBS files back and forth between us manually. We made a proposal to Hank on having the BBS's forward mail automatically; lo and behold, by Thanksgiving Hank had implemented automatic forwarding between BBS's. We were really starting to roll now -- Philly/SNJ and Balto/Wash were linked! And 145.01 was starting to feel the pressure of too many users, too much QRM.

Dick Kutz (KS3Q, about 8 miles from me) and Gary Hoffmann (AK3P, near Harrisburg PA) then brought their W0RLI PBBS clones on the air and all 4 BBS's were inter-linked. The New England/NY area was similarly being whipped into shape by W0RLI, K1BC, KE3Z, WA2RKN, K8KA, WB2KMY and a host of others. Our bottleneck in linking was in northern New Jersey -- an area I unflatteringly called "the Black Hole of Passaic"! This hole was plugged by the WA2SNA-2 NNJ digipeater working with Mike Gullo (WB2RVX) in SNJ. By March, our links were complete (but very fragile!). A string linked PBBS's from Massachussets (W0RLI, K1BC and others) and Ontario (VE3FXI) extending southwest thru NY (WA2RKN-2), CT (W1AW-4), NJ, PA and MD was routinely passing message traffic.

In 1843, Samuel Finley Breese Morse established the first digital link from Washington to Baltimore. Since that is my turf, I can perhaps review the effect of our packet network by asking "What Hath God Wrought?"

I believe that the W3IWI PBBS can lay claim to being the busiest system in the world even though it is only 9 months old. In that time, it has logged about 190 different users and has handled over 5300 messages. In recent months, typically 20-30 messages per day flow thru the system, and about 30 connections per day are logged. The system is connected to a user (i.e."the phone is off the hook") about 23% of the time. It has accomplished what I intended it to do -- it has helped to serve as the "glue" that has bound together packet radio in the mid-Atlantic states.

In other areas attempting to develop packet radio, I have seen a similar pattern develop. The first few experimenters try out packet radio and prove that it works. But until there is a dedicated PBBS activity remains low. Local area network (LAN) activity requires a PBBS as a hub. Applications of packet radio like NTS 3rd party traffic, civil defense require a PBBS function.

In future contributions I plan to discuss how to get a PBBS on the air, how LAN (ab)users should use a PBBS, hints 'n kinks, sottware support, PBBS linking thru the JAS-1, PACSAT and OSCAR-10 satellites, and a myriad of other topics. I solicit contributions and questions from any of you.

[Ed. note: Tom has thrown the gauntlet by his claims for BBS usage and entertaining recollection of packet radio development in the Northeast section of the nation. Who will respond with histories of other areas that Tom has not been involved in ..The WD0ETZ dedicated BBS in St. Louis, the WB6YMH Apple BBS in Calif, and the Stuart, FL BBS for a few...? It is great to chronicle some of this rapid development for the benefit of the recent members.]

****

PACKET MADE EASY

By N4ZQ and KB4LLO
(Reprinted from the June FADCA>BEACON)

Kantronics, long noted for their popular RTTY, ASCII and AMTOR software - hardware, has emerged onto the packet scene. First rumored in the March BEACON from conversations with Travis Brann, at the Orlando Hamcation, and offically unveiled at the Dayton show, the Kantronics Packet Communicator (KPC) offers another way to join in on all the fun of packet radio.

During the Orlando hamfest, FADCA arranged to receive and review two KPC units. One has been in operation daily on both 2 and 20 meters from the N4ZQ QTH near Tampa, and the other some 45 miles away at the KB4LLO shack in Bradenton. Yes I did say 20 meters, and more on that later. First impressions are not always the most accurate. The evening I was handed the secure shipping package from Lawrence, Kansas, my reaction was one of doubt. Before opening the small, thin box I questioned if the TNC was in there. Believed that they had only sent the manual. Surprised would best describe my feelings upon finding not only the manual, but the TNC, power pack (an unusual inclusion for Kantronics) and a bag of cables and connectors. Housed in an attractive cream color case with a two tone blue front panel, the TNC measures only 6 x 8 x 1 3/4 inches. A quick glance through the 41 page manual, found the entire schematic diagram on one page with print large enough not to cause eye strain. Hardly could wait to get home with it and put it through its paces. Believed that there weren't enough parts in it to operate.

Once back in the shack we proceeded to read the manual in order to see what was needed for interfacing. One personal observation that we have made over the years, is that Hams don't like to read the manual. And the quickest way to discourage them is to include a big, heavy 2 inch thick book. The Communicator manual is well written, easy to understand and won't weigh you down carrying it. After a short introduction and brief description of amateur packet radio, installation instructions were found on the third page. All cable connections (audio in, rig, computer and power) are attached to the rear panel of the TNC where the power switch is located. Operating power is supplied from a plug in wall transformer. Care must be exercised, as the transformer has a switch selectable multiple voltage output and polarity switch. Although you can not select more than 12 volts out, you must make certain it's set for positive output. Also steps should be taken to make sure it remains set correctly and is not accidentally changed. Radio interfacing is accomplished by means of 3 jacks. A pair of mini jacks for audio in and out, and a 5 pin DIN connector for PTT, transmit audio and ground. The DIN connector and 3 feet of shielded cable are factory constructed. All you need to supply is a suitable mike plug for your transceiver. Computer or terminal connection is via a standard DB- 25 connector. One plus of the KPC is that it is RS 232 or TTL compatible. Changing the computer voltage levels is accomplished by moving simple jumper on the PC board. This feature should be welcomed by all Commodore users, as the expensive TTL to RS232 intrerface is not necessary. N4ZQ already had the VIC 1011A interface in use with his TNC-1, so we chose the factory preset RS 232 level for testing. Hookup was very simple and straightforward, with excellent details in the manual.

The heart of the system is a 6803 CPU, running under control of a pair of 2764 EPROMS. Also included is 8K of RAM and a NOVRAM for holding changeable operating paramenters. A departure from other TNCs, and unique only to the KPC, is the AM7910 modem on a single 28 pin chip. No external filters or tuned circuits are required. Using only a crystal and a few noncritical components, all the features a modem should have are built right in. For more information on the 7910 see the 1985 ARRL handbook. The modem chip interfaces with the 6803 via an I/O port, which is shared with the NOVRAM and RS232 from the computer. The use of the 7910 gives the TNC one of its greatest features. That is the ability to software select between BELL 103 or 202 tones. BELL 202 tones are 1200 BPS, 1000Hz shift, used by all TNCs on VHF. Bell 103 are 300 BPS, 200 Hz FSK tones, found on 20 meters and other HF bands. Measured center frequency for the tones are 1700 Hz and 2125 for 202 and 103 tones respectively. The 2125 Hz center frequency is 400 Hz higher then most other 300 BPS TNCs on the air, but causes no problem when tuning in the lower sideband mode if one is aware of it.

The KPC is compatible with existing packet TNCs, and supports both AX.25 anad Vancouver Protocols. At first glance at the command list, you would be hard pressed to see much difference between the KPC and the TAPR. In fact, the bulk of the software is Tucson based. The KPC supports 61 commands, of which 90% are the same as you would find on the TNC-1,AEA or Heathkit. But closer inspection reveals some interesting commands, such a BELL for changing the modem configuration. Another is the EQUALIZE command which is used to combat high frequency roll off from some transceivers. This command is only operational in the 202 tone mode, and requires experimenting to see if you need it or not.

The CALIBRATE command is an interesting one, and not what you would expect. Having the modem self-contained, there are no alignment adjustments that have to be performed. The CAL command is used to

tune the transceiver for operation with the TNC. You are able to transmit a square wave mark/space tone at HBAUD BELL setting that the distant station then can tune to, using the receive option. Pressing the R key for receive measures the square wave at the EQUALIZE/BELL setting. The reading is displayed on the screen in the form of two numbers. The object is to get the two numbers to read as equal as possible. This function is a tremendous help when working single sideband 1200 baud packet. A tuning indicator of any type is a must with any SSB packet operation, and this is one of the best we have seen yet. The two KPC reviewers live just far enough apart to make simplex copy marginal at times. A late night experiment on 2 meter SSB at 1200 BPS, using the on-screen frequency counter was very sucessful. Signal level at times was just above the noise, but after both parties completed a receive calibration, a Connect was established first shot out of the barrel. If you have never tried to tune in the short burst of a 1200 SSB Packet, you're in for a real treat. We hope to try this tuning aid through the satellite, where signals are very touchy.

ABAUD or the rate for the I/O serial port, is set for 300, 1200 or 9600 BPS. In fact, you can not perm the rate into NOVRAM. Upon power up of the TNC, it goes into a auto baud routine everytime, supporting one of these rates. HBAUD selects the data rate for radio transmission. The KPC offers 4 choices, 300, 400, 600 or 1200 BPS. Although the command lists a FULLDUP for Full Duplex, the manual makes no mention of it and the TNC does not support it.

TAPR commands not found on the KPC are TRACE, DEBUG, IDTEXT and CWID. The lack of CWID is to be commended, from one that believes they only bog down the throughput on the frequency.

Probably one of the most difficult areas of any packet operation is the correct implemention of Flow Control. Kantronics has improved apon it greatly. The KPC will send the terminal an XOFF when there is room remaining for ten characters in the transmit buffer. If data continues to be sent, the TNC will send a XOFF after every character is recieved when there is room for only five more. The TAPR TNC-1 kit board sends only the one XOFF signal and my software program never seems to pick up on it, and I always over run the buffer.

Transmit audio is factory preset at 21 mv, which turned out to be just right level for the radios used for this review. However, because different transceivers require different AFSK tone levels, a jumper located on the TNC PC board can be adjusted to modify the level. If necessary, changing a common resistor can cause additional modification.

Operation of the KPC was very enjoyable. From the easy hookup, to the fun of 20 meter DX Packet, there's a lot to be said for the Communicator. FADCA wishes to thank Kantronics for the loan of the KPC for this review and demonstration at the Southnet Packet Conference. Probably they said it best in their advertisement, in that they "made getting on Packet as easy as getting on RTTY".

****

AN INTRODUCTION TO NETWORKS

by T.C. McDermott, N5EG
Networks SIG, Texas Packet Radio Society (TPRS)

This article is an introduction to the subject of a packet network. It describes what a network is, why a network is necessary to support amateur packet radio activity, and considerations that govern how a network may be constructed. There are many ways that a network can be designed, and it is beyond the scope of this article to elucidate them all. Rather, this article will focus on the simplifing assumptions that may be made in describing a network, and more specifically will concentrate on some suggestions for the Texas Packet Radio Society network which is called "TEXNET".

Most of us are familiar with packet radio activity through our operations with the TAPR TNC board. This board implements what is called a "Local Area Network", or LAN for short. When we wish to communicate, we ask our TNC to CONNECT to another station. If that station is not within range of our transmitter, then we may connect to that station through a digipeater, or through several digipeaters. This is a convenient extension of the X.25 protocol, and forms a large part of the difference between X.25 and the amateur version called AX.25.

It would be possible to construct a network of stations that are all within range of the next station, and then to connect to any station in the network using this digipeat method, up to 8 stations distant. This would not require the extension of any of the TAPR software, nor would it require the development of any new hardware. Why then is this not an acceptable method to construct a network? Basically this method, although simple to implement has a serious flaw, it lacks "robustness". That is, the method fails to support adequate communications in the presence of a radio path that is not perfect. Secondly, it assures communications integrity through a method known as "END-TO-END ACK".

To understand this, it is necessary to understand how a TAPR digipeater works. The TAPR digipeater is a "dumb" digipeater. That is - the digipeater does not understand anything about the state of the two stations that are trying to communicate to each other through it. When one station wishes to connect to another station through a digipeater, it simply adds the digipeater's address in series with the address field of each and every packet. When the digipeater recognizes it's callsign, it repeats the packet. The digipeater does not know what kind of packet is being digipeated, and does not really care. The packet could be a call-request packet, or user-data, or an acknowledge packet, it really doesn't matter, it digipeats them all, blindly. Why is this important? Because it affects how the transmission and acknowledgment of data is handled between the two end stations trying to communicate with each other.

When the sender, S, tries to send data to the receiver, R, through one or more intervening digipeaters, D1, D2, ... , Dn, it does this as follows:

```
S : sends a packet
D1: digipeats packet
D2: digipeats packet
R : receives packet,
R : sends acknowledge back
D2: digipeats acknowledge
D1: digipeats acknowledge
S : receives acknowledge,
S : sends next packet.
```

Although there can be several packets sent per acknowledge, it requires that a packet, and the acknowledge (ACK) make the round-trip from the sender to the receiver. Thus, the more digipeaters, the longer the round-trip time, and the lower the packet throughput.

The above example assumed that there were no errors in the transmission. What happens if one of the packets and one of the acknowldegments is corrupted during transmission? For example, using the terminology: S=sender, R=receiver, D1,D2, ... Dn = digipeaters.

```
S : sends packet
D1: digipeats packet
D2: doesn't hear packet from D1, so doesn't
      do anything
S : still waiting for ACK from the receiver
S : still waiting for ACK from the receiver
S : still waiting for ACK from the receiver
S : still waiting for ACK from the receiver
S : still waiting for ACK from the receiver
S : times out waiting for ACK, and re-trans
      mits packet
D1: digipeats the packet
D2: digipeats the packet
R : receives the packet,
R : sends ACK back to sender
D2: digipeats the ACK
D1: doesn't hear packet from D2, so doesn't
      do anything
S : still waiting for ACK from the receiver
S : times out waiting for ACK, and re-trans
      mits packet
D1: digipeats the packet
D2: digipeats the packet
R : receives the packet, but it's a duplicate
      - throw away
R : sends ACK back to sender
D2: digipeats the ACK
D1: digipeats the ACK
S : receives the ACK,
S : sends the next packet
```

How long did this take? About 25 packet times. The situation gets worse when 8 digipeaters are chained together. In fact with 8 digipeaters the round-trip time reduces the channel throughput by a factor of approximately 16 (8 hops to R, and 8 hops for the acknowledge to come back) if there are no channel errors. If the probability that any single transmission is corrupted is about 70 percent, then with 8 hops the average round trip will take about 1000 packet times. In other words, nothing will get through.

Why is the TAPR TNC built this way you might ask? For a very good reason - simplicity. To build a digipeater that behaves in a more coordinated fashion turns out to be a very complicated problem. The TAPR digipeater extension is far superior to the other alternative - no digipeater at all. The TAPR digipeater is elegantly simple, and a reliable way to improve the communications between two stations that are reasonably close, but not able to communicate directly. We have seen that one or two digipeaters may not degrade the throughput terribly, provided that the RF paths are highly reliable. Thus the digipeater solution may be called an LAN solution. That is, it is an acceptable network for small numbers of digipeaters, and high-quality circuits. A single digipeater in a superior location allows many stations within the coverage area of the digipeater to communicate. But the digipeater is not an acceptable solution when the need is to communicate over long distances, and with less than high-quality communications circuits.

Thus is born the requirement for a NETWORK.

Another method of information transfer possible is "HOP-TO- HOP" (HTH) acknowledgment. In this method, each packet, or small group of packets, is acknowledged by every receiving station along the path from the sender to the receiver. For example, using the same terminology as above: S=sender, R=receiver, D1,D2, ...Dn = digipeaters.

```
S  : send packet
D1 : ACK to S, repeat frame
D2 : ACK to D1, repeat frame
R  : ACK to D2.
```

Why does this method improve the throughput of the system ? Because now that the sender, S does not have to wait for the ACK to return from R, S may send another packet after the ACK from D1. That is, it may OVERLAP traffic.

```
S  : send packet 1
D1 : ACK S-1, repeat 1
D2 : ACK D1-1, repeat 1
R  : ACK D2-1
```

```
S  : send packet 2
D1 : ACK S-2, repeat 2
D2 : ACK D1-2, repeat 2
R  : ACK D2-2
```

In other words, once that S has received it's acknowledgment, it may transmit the next packet almost immediately (if D1 is on the same channel, it should wait for the D2 --> D1 ack first, if the D2 --> D1 link is on a different frequency (as in a network) then S could transmit the next packet immediately upon receiving the D1 ACK). What happens to the flow of information in the presence of errors in the transmission ? Lets look at an example:

```
S  : send packet
D1 : ACK S, repeat packet
D2 : gets garbled packet from D1
D1 : waiting for ACK from D2
D1 : retransmits packet to D2 after time-out
D2 : ACK D1, repeat packet
R  : ACK D2
```

In the earlier example, with this same exact scenaro, it took 25 packet times to accomplish the transfer of one packet from S to R through 2 digipeaters. In this example it took 6 packet times, a 417 % performance improvement in the transmission time. This performance improvement actually increases with more digipeaters, or worse RF paths. In fact with 8 digipeaters, and a 70% probability of a sucessful packet-hop, this approach offers about a 10,000 % performance advantage !

Additional to the transit time advantage (time delay per packet in seconds-from-S-to-R), there is the advantage in throughput (bytes/sec.). The throughput in the HOP-TO-HOP ack method is NOT dependent upon the number of

digipeaters. This is because as soon as the first digipeater has acknowledged the reception of the senders' packet, the sender is free to send the next packet, regardless of the number of hops in the path. Contrast this to the end-to-end hop method, where the throughput is very dramatically dependent upon the number of hops in the path.

What is required of the digipeaters in the network to handle this type of repeating function, i.e. HOP-TO-HOP digipeating? Each repeating station is required to contain a fair amount of memory, enough to buffer every packet that it digipeats until that packet is acknowledged by the next repeater. Since the repeater may receive packets from several different stations at nearly the same time, and perhaps some of them are occaisonally garbled in transmission to the next repeater, then they must be stored in repeater memory until they are sucessfully passed to the next repeater.

The repeaters must also implement some sort of flow control. If packets arrive faster than they can be sent, then the buffer memory could overflow. Thus the repeater must be able to tell the previous repeater, or sender, that the packet is rejected, and to stop sending. When the repeater clears the messages, and thus frees up some memory, then it re-initiates packet transfer from the previous station. This finite memory size limitation actually causes the END-TO-END performance of the network to become more heavily dependent upon the quality of the RF links. Thus performance of the HOP-TO-HOP system is dependent upon the probability of RF path-hop success, but is not heavily dependent upon the number of repeaters in the path, unlike the END-TO-END scheme. This was taken into account when I stated that the performance of the HOP-TO-HOP ack with 8 digipeaters, and 70% path-hop probability of success was about 10000% better than the END-TO-END method.

There is one interesting disadvantage to the HOP-TO-HOP scheme, although it is not a strong disadvantage, and that is the issue of data integrity. In the END-TO-END ack scheme when the receiver acquired the data, the ACK was sent. Thus when the sender receives the ACK, there is certainty that the data was in fact received. In the HOP-TO-HOP ack all that is known when the sender receives the ack, is that the first digipeater received the ack. A failure in the network could still block the receiver from receiving the data - thus the sender was ACK'ed even though the receiver had not received the data. This is not as serious a problem as it sounds at first, however, since there is still a method to determine whether the data is received at the final destination correctly.

This is handled by layer 4 of the OSI model - the transport layer. It is responsible for data integrity in the real world of unreliable networks. One protocol for doing this is familiar to those of us with AX.25 units, and this is the Virtual Circuit protocol. Each of us is intimately familiar with virtual circuits. Any time that you connect to another station, you have generated a virtual circuit. You and the receiver communicate on a common channel with everybody else. But your traffic only goes to your desired destination, not all destinations on the channel. Thus there is a circuit between you and the connected receiver on a channel with a (theoretically) unlimited number of circuits. This is called a virtual circuit. The only reason there is a circuit is because you and the connected receiver previously agreed to a connection. The circuit is dissolved when you and the receiver agree to this (disconnect).

Generally, concern about whether the data was received by the receiver is not a problem. It communications fails along a path we naturally tend to want a retransmission of the entire contents of whatever file we may have been sending. If we happened to be in the keyboard mode, then since the communications failed, we cannot continue to send anyway. Thirdly, our TNC's do not tell us how much data they have in their transmit buffers that has not been acknowledged, so the HOP-TO-HOP network really doesn't differ from the types of responses that we are used to from the TNC - LAN system.

If we must guarantee the absolute integrity of a file transfer, then we should implement some type of block numbering and sequencing program that controls the file transfer process. In essence, something like the MODEM7 protocol tacked on top of our existing TNC protocol would guarantee the complete integrity of files transfered. We would probably want to add onto the MODEM7 system a little bit, perhaps to record what was and was not sucessfully transfered, and perhaps a method of automatically reestablishing the connection to the other station, and continuing with the transfer process until it is sucessfully completed, and then tearing down the connection.

This additional program that we would run on each of our end-user computers (both sender and receiver) is the LAYER 4 of the OSI model - the transport layer. Since the network we are talking about constructing is thus releived of absolute ACK integrity by the presence of this additional program (in those rare instances where it is really needed) then our netork is only restricted to providing a reasonable guarantee of integrity, perhaps guaranteeing that packets arrive in the correct sequence, and without bit errors. Thus with an emphasis on the HOP-TO-HOP VS. the END-TO-END methods, we have decided to build a network that optimizes throughput and response, allowing for a layer 4 program in the event it is needed, but not sacrificing the performance of the network for the vast majority of the uses of the network. This tradeoff is usually described as a speed-integrity tradeoff in the literature.

Now that a decision has been reached on the desired attributes of the network, that is speed, and simplicity, we may concentrate on one final interface aspect, and that is how the LAN (i.e.the TNCs) are to interface and establish connection through the network. This linkage is the peer communication between two layer 3 processes that is described by Tannenbaum [1].

In the 7-layer OSI model, subnet communication (that is communication through a network) is established between two layer 3 processes - that is, the TNC AX.25 mode, and the network entrance and exit layers. Although AX.25 is sometimes discussed as a layer 2 protocol, in the LAN application it is really a layer 3 process. It establishes, communicates, and terminates, and thus it is a layer 3 process. Similarly the network, upon command, will establish, communicate, and terminate, thus it also is performing a layer 3 function.

Why the concern about which layers to call each other? Because it is desirable that the TNCs be able to use the network without any modification. Thus the network must be compatible to the way that the TNC establishes, maintains, and terminates connections, since the network must establish, maintain, and terminate connections to or from TNCs and itself. We are faced with the choices as to how the TNC and the network can talk with one another.

One method of network interface is to assume that the network is transparent, that is it looks like a digipeater. Then you would use your TNC as though the network were a single digipeater (even though it might have many hops, it would appear to the TNC as one digipeater).

Another method is to treat the network as a spearate LAN address. This is, you would connect to the network. Then the network would engage in an interactive session with you regarding the type of service that you needed, that is, who you wanted to talk to, and how to get through the network to that place. Once the network computer was satisfied, it would then engage in comunications between the endpoints. Your TNC would think it was connected to the network, not to your actual destination station.

Each of these connection methods has advantages and disadvantages. We will discuss some of them here. The digipeater emulation method is a very natural method to use, because the connection method is familiar to all of the TNC users. Let us establish the following scenario: WD0ETZ in Carrollton wishes to communicate with WD5GAZ in Houston. WD0ETZ knows that this distance will require the use of the network. So WD0ETZ proceeds as follows...

Connect WD5GAZ VIA DALLAS

This seems simple enough, but some interesting problems crop up almost immediately. How does the network know where to find WD5GAZ? What path is required to get there? WD0ETZ's TNC is going to want to see ACKs from WD5GAZ, not from DALLAS.

First problems first. One way for the network to know how to find WD5GAZ is for it to keep tables in all of the sites of each and every network user. This is really not practical in an amateur environment because hams move, come and go,and even change callsigns, and with very many users, it takes a lot of manual intervention to keep the tables current. It also takes a lot of computer power in the network to store and route messages based upon these tables. In the event of a network crash, the tables would have to be reloaded, etc. A simpler way would be for the originator, in this case WD0ETZ, to specify the network "hop-off" point, that is, the location in the network where WD5GAZ is likely to be found. For example:

Connect WD5GAZ VIA DALLAS,HOUSTON

Now the network knows that the entry point is this network (which is "DALLAS", the one hearing WD0ETZ) and the exit point is HOUSTON. Perhaps different types of names would be chosen for network nodes. Grid-squares and major city names seem to be two obvious choices. What about the route to take to get along the network? There is an incomplete but simple answer to this question - make a linear network (or a simple variant of linear). A linear network is one where the network is basically a straight line. Thus there is by definition only one path between any two points.

A few other questions. What if WD5GAZ is not within range of the HOUSTON node, but perhaps within range of a station that is near to the node - for example suppose that WA5AAA is between the HOUSTON node and WD5GAZ. Then...

Connect WD5GAZ VIA DALLAS,HOUSTON,WA5AAA

What if WD0ETZ is not within range of the DALLAS node, but is within range of WB5QNG, who is within range of DALLAS ?

Connect WD5GAZ VIA WB5QNG,DALLAS,HOUSTON, WA5AAA

Well, the addressing would work. But the network entry point has to do some strange things to the address field. Remember in the HOP-TO-HOP scheme it would be the address DALLAS that is actually ACKing WD0ETZ, and not WD5GAZ that would be ACKing WD0ETZ, so the network node has to play "fast-and-loose" with the address headers in the digipeat field.

The other method is fairly straight forward. The user connects to the network, and then enters an interactive Q & A session:

Connect DALLAS
Welcome to TEXNET - DALLAS node.
There are currently 4 other users connected to DALLAS.
Enter destination callsign ? (WD5GAZ would be entered here)
Enter network exit node ? (HOUSTON would be entered)
Enter destination digipeaters ? (WA5AAA would be entered)
CONNECTION ESTABLISHED - PROCEED

Notice one thing in the above scenario: more than one station may be connected to the network node entry and exit points. This is something that is a little foreign to the AX.25 protocol, that is MULTIPLE VIRTUAL CHANNELS to a single TNC. In this case it is still compatible with AX.25 since both source and destinations callsigns are part of the AX.25 standard. Only the network nodes have to have this special property of having to be connected to several different stations simultaneously - thus the AX.25 code for these controllers is a little different from a normal implementation. But only the network requires these special TNCs (actually they are built into the node controller, and aren't identifable as a separate device).

The reason that the network should allow for multiple virtual channels is to allow multiple people to simultaneously use the network. Since we will put high-speed radios in the network between nodes, we should take advantage of the bandwidth available.

The next article will deal with the type of hardware that will be required to support this concept of a network, and it turns out to be surprisingly modest. There are some other concerns about capacity, response time, channel utilization, reliability, and remote network "resuscitation" (in the event of software failure) that will also be addressed in part 2 of this series.

****

THE TWO PORT DIGIPEATER

Jon Bloom, KE3Z

Recently, I have heard from several people who seem to be confused by the mutli-port digipeater concept. What I hope to do with this article is to reduce the confusion level. I'll first explain the nature of the problem that prompted the development of the multi-port digipeater. Next, I'll explain what the digipeater does. Then I hope to show exactly how the multi-port digipeater solves, or at least reduces, the problem. Finally, I'll look at some interesting applications of the multi-port digipeater.

THE PROBLEM

Anyone who is more than casually interested in amateur packet radio will recognize the problem. I'll describe it from the perspective of a user of the northern part of EASTNET, although the same problem is found to a greater or lesser extent in all areas of the United States.

To understand the nature of the problem, we have to understand an entity known as a "collision" (please forgive me if I'm treading old ground for some of you). A collision occurs when a receiving station hears two transmitting stations at the same time. When this happens, the receiving station cannot properly receive the transmitted frame(s) of both transmitting stations. The nature of FM, which is the modulation method used on VHF packet, is such that the receiver MAY recive one of the signals if it is considerably stronger than the other, but often neither signal is received. For the purposes of our discussion, I will assume that neither transmitted frame is properly received.

When I wish to connect to another station using the local digipeater, there are three points at which a collision can occur. When my TNC transmits the initial SABM (connect request) frame, it may collide with another transmission at the receiver of the local digipeater. If that occurs, the digipeater will not retransmit the frame. My TNC will wait for a period of time and then try again. It I manage to transmit my SABM to the local repeater without a collision, the repeater will then retransmit the frame. Again, there is the possibility of a collision. This is usually not a problem because the local digipeater is by far the strongest signal on frequency at the receiving station. When the staion I'm calling responds to my SABM with a UA (acknowledge) frame, the same potential for collisions occurs. This time, the receivers involved are, once again, the digipeater, and finally my receiver. Because of the ARQ nature of packet, retries will usually "solve" the collison problem eventually, although a significant delay occurs because of the time between retries. Unfortunately, this is the simple case. Many packet connections occur over multiple-digipeater paths. In this case, there is the potential for collisions at the receiver of each digipeater in the path. As pointed out by W3IWI, the probability of reception at the destination station decreases exponentially as the number of digipeaters increases. For example, if the probablility of reception for each digipeater in a five-digipeater path is 90% (an optimistic value), the probability of reception at the end point of the path is 59% (.9^5). This lack of reliability results in significant network delays. Where do the frames that collide with mine come from? Most of them are NOT from other local users. Rather, they come from other digipeaters. As an example, the local repeater here in central Connecticut is W1AW-5, in South Windsor. This machine can hear four other wide-coverage digipeaters: KG1O-9, Mt. Ninham, NY; KY1H, Peru, MA; WA1TLN-1, Mt. Ascutney, VT; and, WA1IXU, Collinsville, CT. During the "prime time" hours of the evening, at least one of these digipeaters is transmitting at any time. This makes it difficult to get ANY data through the network.

The problem, therefore, comes down to one basic fact: with all of the digipeaters in the network on the same frequency, collisions are impossible to avoid.

THE MULTI-PORT DIGIPEATER

The multi-port digipeater was developed as a temporary solution to the problem. I say temporary because, if justice triumphs, it will be replaced with true network-layer machines in the .not-too-distant future. The major effect of the multi-port digipeater will be to get the network digipeaters, and hence their local users, onto different frequencies.

The multi-port digipeater has two or more "ports" which are separate digipeater nodes. Typically, each port will be on a different frequency -- perhaps even on a different band. Each port is capable of acting as a stand-alone digipeater. That is, it can receive and transmit frames in the normal digipeater manner. But the multi-port digipeater can do more, as it can route frames received on one port out a different one. How does it do this?

Within the EPROM memory of the two-port digipeater are several constants and tables which determine the transmission port for a received frame. The digipeater determines the transmission port by examining the address field of the received frame (routing). If the frame is fully repeated (this digipeater is the last one in the address field of the received frame), the digipeater software first looks through a table of "destination stations" for a match with the address of the destination station in the received frame. If it finds a match, it retransmits the frame via the port indicated in the table.

If the received frame is NOT fully repeated (there is a digipeater in the address field after this digipeater), a different address table is scanned, looking for a match with the address of the next digipeater. If a match is found, the digipeater will, once again, retransmit the frame via the port indicated for that address by the table. If no match is found, the SSID of the next repeater address is examined. The multi-port digipeater can be configured to send all frames of a specific repeater SSID out a particular port. For example, all frames in which the next repeater address has an SSID of 8 could be routed out port 1. Each port can have its own "default" SSID.

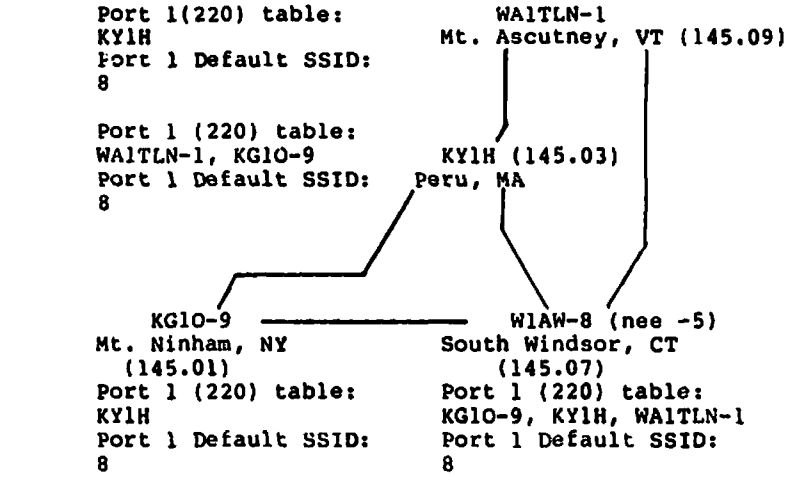
Should explicit routing fail (that is, the address/SSID was not found in the internal tables), the digipeater retransmits the frame on the default retransmission port for the receive port. If no explicit routing is determined the digipeater may retransmit a frame on the same port it was received on. Thus it will "look" like a normal single-port digipeater to that frame: received and

retransmitted on the same port, or frequency. In the proposed network, all frames that can't be routed by SSID or table look- up will be transmitted on the local, or 2-meter port.

This is all very confusing and you are probably wondering what purpose it all serves! In the next section we will see by example how the multi- port digipeater uses its routing capabilities to solve the network collision problem.

THE SOLUTION TO THE PROBLEM

Consider the network shown graphically below. Each digipeater is a two-port digipeater. The frequencies used on each port and the tables set up at each digipeater are shown. All of the inter-digipeater links are on a common 220-MHz frequency. Any frame which is not explicity routed is retransmitted on 2-meters.



Let's follow a few frames thru the network. WB2KMY in Poughkeepsie, NY wants to connect to KE1G in Goffstown, NH. WB2KMY is on 145.01 MHz and sends his connect request via the path: KG1O-9, KY1H, WA1TLN-1. Upon receiving this frame, KG1O-9 examines it, sees that the next repeater is KY1H and finds KY1H in its port-1 table. It therefore retransmits the frame on 220. KY1H receives it, sees WA1TLN-1 in the address, and retransmits it on 220 again. When WA1TLN-1 receives it, it is a fully repeated frame. Since there is nothing in the TLN-1 destination station address table, the frame is transmitted on the default transmission port, on 2 meters. This default action eliminates the need for a table of local 2-meter users; only the 220-Mhz stations (the "backbone" network) need be known to the digipeater.

Now, KE1G receives the frame and replies via WA1TLN-1,KY1H,KG1O-9. The same routing action occurs on this path, with WA1TLN-1 and KY1H transmitting on 220 and KG1O-9 on 2 meters, where WB2KMY receives it.

Let's say that another activity is occurring at the same time. In this case, KE3Z is reading a long message from the W1AW-4 PBBS, going through W1AW-5. When a frame is transmitted by W1AW-4, W1AW-5 receives it. Since W1AW-5 is the only repeater in the addres field, the frame is fully repeated. Neither the address KE3Z-0 nor the SSID 0 are to be found in the tables of W1AW-5. So, it retransmits the frame on the default, 2-meter port. In other words, all of this activity at W1AW-5 is occurring on 145.07. NO TRANSMISSION IS MADE ON THE 220 LINKING FREQUENCY. Since none of the other repeaters have a 2-meter frequency ot 145.07, none of the activity at W1AW-5 is heard by the other machines and vice versa.

So what happens when there are several connections active via the 220 linking frequency? Aren't we still going to have collisions? Yes, but far fewer. Remember, any connection which only makes use of a single digipeater uses no part of the available 220 capability. Also, since every station on the 220 frequency is a wide-area digipeater with a good VHF location, most of them can hear one another. This is a situation in which the Carrier- sense/Multiple-access (CSMA) protocol used in packet radio works well. Local users on 2 meters can transmit at the same time a distant digipeater is transmitting on 220 without collisions. In fact, W1AW-8 can digipeat packets on 220 (say between KG1O-9 and KY1H) while simultaneously digipeating the W1AW-4/KE3Z packets.

Note that none of the repeaters has W1AW-8 as an explicit address in its table. Instead, the fact that the SSID is 8 is used to perform the routing. Why have this capabiltity? Let's assume that a new station, WA1??? wishes to join the network as a two-port digipeater. Must all of the digipeaters which will communicate with this machine on 220 now update their EPROMs to add WA1??? to the tables? No. If WA1??? uses an SSID of 8, frames will be automatically routed to it on 220 by the other digipeaters without any change in tables. If all machines with a 220-MHz port agree on a common SSID, no specific call signs must be entered in any digipeater's tables.

Obviously, there are far too few 2-meter frequencies available to assign a unique one to each two-port digipeater. What we end up with is a situation somewhat analogous to that of the voice repeaters. We could, for example, have a machine in New Hampshire on 145.07 because it would not normally hear, or be heard by, W1AW-8. Still, there are far fewer frequencies available to 2-meter packet users than voice machines. Fortunately, packet repeaters can accept a small amount of co-channel interference better than voice machines can.

DEBUNKING THE SSID-TRANSLATION TECHNIQUE

When the multi-port digipeater concept is first considered, there is an intuitive belief that some sort of SSID translation is needed. That is, that the second port of the machine operates with a different SSID than the first port. I was bedazzled by this illusion too, but I eventually discarded it. To briefly explain the concept: When such a machine received a frame on port 0 (the 2-meter port), it would retransmit it on port 0 if the received SSID of its own address was 1. It would retransmit it on port 1 (the 220 port) if the SSID was 2. When it retransmitted the frame on 220, it would have translated its SSID in the frame to 1. This is so that any reply frames generated by the far endpoint station would have been generated with an SSID of 1, forcing the retransmission of the reply frame on port 0. (If this seems confusing to you it's because it's confusing.)

My main problem with this concept is that it requires the user to have an intimate knowledge of the ports used at each digipeater in the path. You have to specify the retransmission port of

each digipeater by selecting the proper SSID. This is not TOO awful as long as all of the digipeaters are two-port machines; all of the digipeaters are -2 (220 backbone) except the last one, which is -1 (for the 2-meter destination). If, however, one or more of the digipeaters is a three- or four-port machine, or if several repeaters are operating under the same call sign, it can be very hard to get the path right (we will discuss the function of digipeaters with more than two ports shortly). What the SSID-translation technique accomplishes is to remove the routing intelligence from the digipeater and force the user to supply it. This seems to me to be a giant step backwards.

The other major failing of the SSID-translation scheme is that it does not inhibit direct user access of the linking frequencies. If users start showing up on the "trunk" lines, we will rapidly be back just where we are now, with large numbers of collisions and a network that grinds to a halt. The SSID-translation machine doesn't protect against this, as it makes no difference which port you access. With the intelligent routing machine, a user accessing the machine via the network port never gets a reply from a called station. This is because when the reply frame is received back by the digipeater, it is fully repeated and goes out the 2-meter port by default.

THE TRUE MULTI-PORT DIGIPEATER

So far, we have shown only the two-port machine in operation. In the simplified network shown above, two-port machines handled everything nicely. In reality, a nagging problem crops up as the network is expanded. Let's say that the 220 link is expanded down across New Jersey from KG1O-9. Since the New England machines can't hope to hear the New Jersey machines, the potential for collisions between New Jersey frames and New England frames at the input of KG1O-9 is very high. This is not to say that the network wouldn't be much improved over the present state, just that the same old problem is starting to rear its ugly head again. One possible solution is to have the link between KG1O-9 and the New Jersey machines on yet another linking frequency -- one which would not interfere with the 220 frequency used in New England. This frequency could be on 220 or perhaps 450, which would ease the implementation at KG1O-9. At the southern end of New Jersey, where linking to the MD/DC/VA/PA areas occurs, the 220 frequency used in New England could once again be pressed into service. The three-port machines used at these "choke points" where much of the network traffic is handled would, of course, be key players in the network.

WB2KMY-1 digipeater is on Mt. Beacon, NY. It is very close to KG1O-9, but is isolated from much of the New England and New Jersey traffic by mountainous terrain. If it links to KG1O-9 on one of the two 220 linking frequencies, collisions will occur. Therefore, it would be ideal if the link between these two machines were on yet a fourth frequency. With the terrain and distances involved, 1296 MHz or even 10 GHz could be used. KG1O-9 would then be a four-port machine and WB2KMY-1 would need three ports.

Recently, I have heard some naysayers deriding this scheme because "there are too many frequencies involved" or some such specious argument. What these scoffers fail to realize is that even when we have true network-layer machines, the same frequency/topology problems will persist. Indeed, one of the motives behind the multi-port digipeater project is to get the RF media in place, ready for the next generation of the network: level 3.

NETWORK TRANSMISSION SPEEDS

I haven't mentioned the subject of transmission speeds on the inter-digipeater links before this because it isn't really germane to the basic concept of the multi-port digipeater; the advantages explained above would acrue to the network even of the links are still operated at 1200 bauds. Those of you who follow packet developments closely know that a revolutionary (in amateur terms, at least) new modem has appeared from the lab of K9NG. This modem is capable of operating at speeds greater than 9600 bit/s, although 9600 appears to be the developing standard. I anticipate that no inter-digipeater link will operate at below that speed.

The increase to 9600 bit/s provides an obvious advantage: eight times as much traffic can be carried by a single link. What may not be so obvious, at first glance, is the comcomitant benefit: lower delay. It should take less time to send a frame and get a reply through 8 digipeaters (4 1200-baud transmissions and 12 9600-baud transmissions) than it does through two digipeaters (6 1200-baud transmissions) now. Combine that with the infrequency of collisions and you have a network that really flys! Note also that the speed limitation of the K9NG modem is that of the bandwidth of the RF system. There is no reason why much higher speeds could not be attained with specially-designed radios.

There is also no reason why a 9600-baud local port could not be implemented. For instance, a digipeater in the network could have a 9600- baud network port, a 1200-baud "entry-level user" port and a 9600-baud user port for the more ambitious packeteer. The current 2-port digipeater, which is based on a Xerox 820, will support a 1200/9600, 300/1200 or most any combination of two standard speeds.

OTHER USES OF THE MULTI-PORT DIGIPEATER

This isn't really part of the network discussion, but it hasn't been published anywhere else to my knowledge so I thought I would include it.

The flexibility of the routing scheme employed by the multi-port digipeater gives rise to several other possible uses. One of the more interesting is its use as an HF/VHF gateway. This is done by "cross channel" routing, in which everything received on port 0 is retransmitted out port 1 and vice versa. (Naturally, by "everything" I mean all frames addressed to go through the digipeater!) With port 0 on 2 meters and port 1 on HF, you have a simple, cheap gateway which can be used by multiple users simultaneously. Right now, the only operational HF/VHF gateways of which I am aware make use of the W0RLI MailBox software. Such gateways can only be used by one user at a time and require the use of two TAPR TNCs or clones thereof.

Another possibility, albeit one that doesn't excite me, is use as a "closed" packet repeater. By having the addresses of the authorized users in the ROM and "grounding" (throwing away) any received frame from other users, the repeater would be effectively closed. Grounding frames which are not explicitly routed by the tables is accomplised by setting the default retransmission port to route frames to a null I/O driver. Of course, the

EPROM would have to be updated to add a user. This would be a pain, but anyone who runs a closed repeater deserves it. It also requires that someone write the null driver or modify the existing software slightly.

Although a multi-port digipeater with more than two ports has not yet been implemented, the software is designed to support it. What needs to be done is to add more HDLC ports to the Xerox 820, or run the software on a different machine with more ports built in. The FAD board from TAPR is one possible solution to adding ports to the 820, although the software is not set up to drive an 8530, the chip used on this board. I hope to hear shortly from some people who have been working on an 8530 driver and will make that available after testing it.

FINAL WORDS

I realize that this treatise could be viewed as simply grinding my own axe. I have tried to act as a dispassionate observer, but my particular situation, in which I have a weak signal into the nearest digipeater, forces me to have a degree of concern for the problem which is, perhaps, greater that that of more fortunate users. I would be interested to hear any comments, thoughts, complaints or Bronx cheers you may have on this subject.

73, Jon Bloom, KE3Z @ W1AW
17 June, 1985

****

ANOTHER VIEW OF PACKET LINKING AND TWO PORT DIGIPEATERS

Jerald Morris, WB0RUS
Box 329
Hooper NE 68031

Sitting here in the Midwest I have been reading about the problems being encountered on the East coast through the courteous efforts of Mike, NF0N, who has been kind enough to download information from Compuserve and place it on our local PBBS (KC0OJ). It is the result of those efforts that I stay somewhat informed.

I have been busy trying to design a two port digipeater myself and have some facts and views to share. The board itself will measure 5" x 6" and will have one modem on board. The addition of a second modem will make up a two port digipeater. Currently I am about to the beta test point.

One thing that I chose to do differently is the SSID number decoding. Rather then use the SSID number to dictate which port is which I chose to have it determine the function of the digipeater. That is to say that all ports of the digipeater have the same SSID number, after all they will all be on different frequencies (we hope). Then I assigned a different SSID number to direct the digipeater to put a packet to a different port. This eliminates the need for the digipeater to have to change things in the original packet and the return routing remains intact. As an example suppose we have three two port digipeaters known as A, B and C and port 1 of these digipeaters is on 145.010 and port 2 is on 220 if a person wanted to use the two meter side of the network his VIA list would be A-1,B-1,C-1 as normal but if the 220 side was to be used the list would be A-2,B-1,C-2. The "2" being the directive to cross to port 2 and back to port 1 in the case of the last digipeater. As you can see the return path routing would cause the same thing in reverse order. I believe this to be a better way because there is less tampering with each packet and the ability of the individual to remember the path is easier. Further yet digipeaters with the same call could still use the same SSID number for the cross port directive if it could be agreed upon for the entire network, easing the problem of keeping track of different cross directives from digipeater to digipeater.

Now I'll touch on the auto routing scheme that seems most appealing to me. It seems to me that the other routing systems I've seen talked about require that each digipeater has to look at a lot of addresses not concerning it and that there is a lot of packet changing going on. This scheme only requires that the user know the beginning and end digipeater and a third SSID number. I would hope that this SSID number could be agreed upon in any given network. The SSID number of 15 (QST) is not very useful as far as digipeaters are concerned so I chose to use it to indicate the auto routing function. Here is how I intend it to work. If the first digipeater finds an SSID of 15 in its own address it would look at the next address in the list and compare it to a table. When a match is found it would then insert the path needed to get there. If a match is not there it would then generate a DM packet from the destination station followed by a UI packet saying "PATH UNKNOWN". This only requires that each digipeater keep a table of other digipeaters within its reach. The next digipeater that runs into a 15 in its address would backup to the first address with a 15 and delete all the addresses in between making the routing transparent to the user while at the same time giving extended digipeater range. The last digipeater will have flagged the first 15 in the process of checking for its own address. None of the digipeaters in the network except those two would change any part of a packet greatly reducing the chance of an error. With the falling prices of EEPROMs and their increased density it is possible to use these devices to keep the tables up automatically by simply noting the use of the 220 side of the network. This could be done whenever the cross port function is used. When the first digipeater is instructed to cross it could store all the addresses in the digipeater list to be compared against the table if it don't recognize the last address in the string. Likewise, the last digipeater in the list would do the same, assuming one of them is the new one. During its own idle time the lists could be updated automatically. This would cut down the amount of time the owner of a digipeater would have to spend updating the digipeater lists. Also these lists could be looked at for those who are not knowledgeable of all the paths available.

I have seen a lot of concern about individual users on the 220 backbone. I think that some things could be done to limit the amount of traffic it might see. What comes to mind first is to block BEACONS and UI frames from the backbone. This has become known in Nebraska as the BEACON KILLER and frankly I probably will implement it into the software I write. Another thing that comes to mind is to use one of the spare bits in the control byte as a backbone flag which should discourage most who try to come up by causing framing errors. Also the digipeaters could forget about packets that don't have the bit right.

REVIEW.....

What I have described is a network where all the digipeaters use the same SSID number on both two meters and the 220 backbone and use two additional SSID numbers that could be common to the whole network. One of these SSID numbers directs the digipeaters to put the packet to the other port regardless of which port it came in on. The other number directs the digipeaters to determine routing and cross ports for packets received on two meters and delete routing and cross ports for packets received on the 220 port. If an individual was unfamiliar with the network a simple connect to the local digipeater would reveal all known paths and the SSID number system used on that network. More over all UI frames (including beacons) would be blocked from the 220 backbone of the network and each digipeater would be constantly updating its routing tables.

FUTURE.....

One other thing that I'd like to touch on is LMUs (Link Management Units). I don't think (many of you probably won't agree) that your going to find one of these every 60 miles or so, as funds won't be available. This may not seem like a problem in the more densely populated areas but here in the midwest it would be a big problem. Out here our ham population is very sparse in some areas and some groups would have a difficult enough time raising funds to do their own LMU let alone one more to get half way to the next group. I would like to see the quote "level three" combined with efforts to put up a geostationary satellite where LMUs could be put within 4 or 5 hops. then they could tie directly to other LMUs via the satellite where wide band high speed data transfers could occur. Multifrequency inputs to the satellite could be used and lessen the possibility of collisions. All the hams on packet could (or should) contribute to the building of the satellite and larger groups could fund centrally located LMUs possibly on a state wide basis which I feel would be a lot less burdensome on the individual packeteer and a lot more reliable. This approach would make a geostationary satellite more attractive because the hog syndrome is less likely. This could possibly satisfy both the REAL TIME and the Electronic Mail factions at the same time, while giving all of us without a lot of funds the chance to use "Level Three". I realize that this would cost a lot of money but think that it would still be less then that required to put an LMU every 60 miles

I invite comments on any of the ideas expressed here as I will probably be attempting to implement them in the very near future (except that I can't afford a satellite!!!!).

Please address your responses to the address above or via compuserve to: Mike Nickolaus - NF0N
72716,2116

****

▼A = AUDIO GROUND IN "MODEM" AREA

RADIO SHACK PARTS LIST

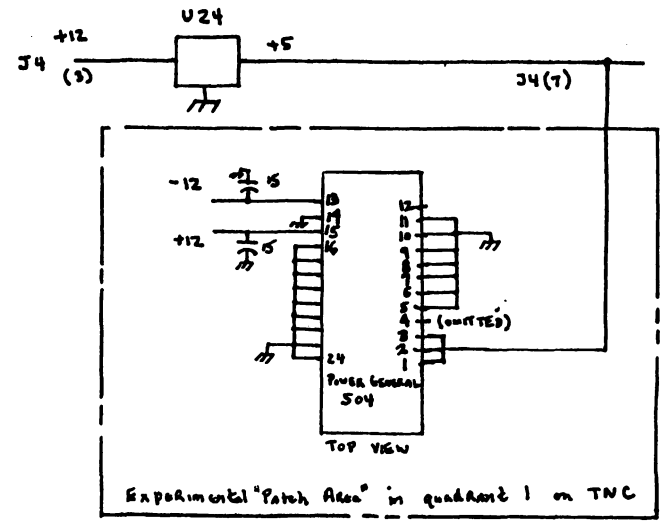
SWITCH	3PDT	275-661	$3.39
R-36a	POT.	271-343	$1.49
R-38a	POT.	271-340	$1.49
R-46a	RES.	271-1350	$ .39

TAPR TNC MODIFICATIONS FOR 12v USE

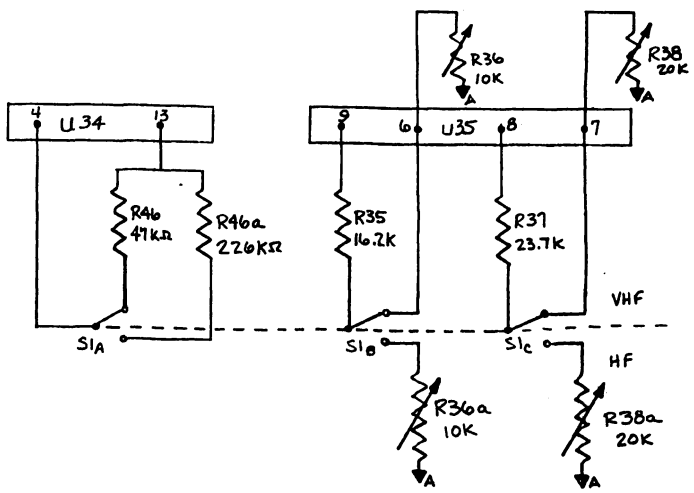
Robert Ball, WB8WGA
(Reprinted from QEX, June 1985)

I have modified my TAPR TNC for 12-v portable operation by installing a Power-General dc to dc converter chip to generate the + or - 12v source for the EIA and analog circuitry. The circuit, as shown on the data sheet, can be referenced below.

I removed U22 and U23 from the board since thdy are no longer needed. Note the power dissipation in U24 increases about 4 watts, but the supplied heatsink is adequate if mounted properly on the chassis. The board will still operate properly from 12 volts. If you desire to leave the 12v connected while operating from the AC supply, a diode should be provided in the +12v lead connection to J4. The TNC draws about 1 amp from a 13.5 v power source.



****



PARTS LIST AND SCHEMATIC FOR TNC 1 HF OPERATION

(Article on page 8)

The Tucson Anateur Packet Radio Corporation is a nonprotit scientific research and development corporation. The Corporation is licensed in the State of Arizona for the purpose of designing and developing new systems for packet radio communication in the Amateur Radio Service, and for freely disseminating information acquired during and obtained from such research.

The officers of the Tucson Amateur Packet Radio Corporation are:

Lyle Johnson ..... WA7GXD ... President
Pete Eaton ..... WB9FLW ... Executive VP
Pat Snyder ..... WA0TTW ... Secretary
Dan Morrison ..... KV7B ..... Treasurer

The Packet Status Register is the official publication of the Tucson Amateur Packet Radio Corporation. Explicit permission is granted to reproduce any material appearing herein, providing credit is geven to the author and TAPR.

TAPR membership and PSR subscription mailing address:

Tucson Amateur Packet Radio Corp.
P.O. Box 22888
Tucson, AZ 85734
(602) 746-1166

PSR editorial submission address:

PSR Editor
812 Childers Loop
Brandon, FL 33511
(813) 689-3355

MEMBERSHIP APPLICATION

Tucson Amateur Packet Radio Corporation
P. O. Box 22888, Tucson, AZ 85734

Name:_____

Call Sign:_____ License Class:_____

Address:_____

City & State:_____ Zip Code:_____

Home Phone:_____ Work Phone:_____

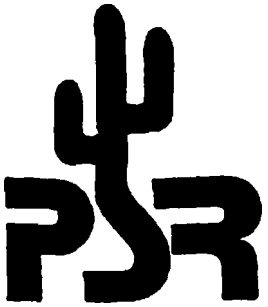
If you wish not to have any of the above information published in a membership list, indicate the items you wish supressed here:

_____

I hereby apply for membership in TAPR. I enclose $12.00 dues for one year.

Signature:_____ Date:_____

TUCSON AMATEUR PACKET RADIO CORPORATION
P. O. BOX 22888
TUCSON, AZ 85734



Check your address label for membership expiration date.