I PRESUME YOU ARE FAMILIAR WITH (OR THAT YOU CAN GET HELP ON) THE TOPIC, AND IMPLEMENTATION OF AMATEUR RADIO PACKET NETWORKING PROTOCOL AS IT EXISTS TODAY, IN PARTICULAR, THE X-1J4 AND OTHER DERIVATIVES (THENET, G8BPQ, etc...).

Please review, if necessary, the documentation conventions that are used. A complete description is found in the KPC-9612 Getting Started and Reference Manual beginning on page 23.

Although the high-speed port (port 2) of the KPC-9612 is capable of operating at 4,800 or 9,600 or 19,200 baud, this manual will refer to operation at 9,600 baud. *The KPC-3 is also referenced in this manual. Only those parameters for the KPC-3 apply.*

EQUIPMENT REQUIREMENTS

KPC-9612 OR *KPC3* Terminal Node Controller (AS OF 2008-2009, all KPC-3 are supplied with this EPROM)
        The *KPC-9612 K-Net* PROM MUST be installed in a KPC-9612 Controller.
        It WILL NOT OPERATE if installed in any other TNC.

        The *KPC"3" K-Net PROM* MUST be installed in a KPC*"3"* Controller.
        (AS OF 2008-2009, all KPC-3 are supplied with this EPROM)

TRANSCEIVER(S)
        At least one transceiver is required to enable K-Net operation. Most of the time a second transceiver is used to take advantage of the full potential of K-Net. Usually a 2-meter transceiver is connected to port 1 (1200 baud) of the KPC-9612, and a special "data-ready" radio, capable of operating at 9,600 baud is connected to port 2.

COMPUTER
        A computer running a terminal program must be used to initially configure the KPC-9612 for K-Net operation.

128K STATIC RAM (Optional, but recommended)
        Since K-Net operation requires more RAM than traditional TNC operation, additional RAM is recommended to allow for a larger PBBS. 128K static RAM chips are available from Kantronics.

INTRODUCTION TO BASIC PACKET NETWORKING

A network node is a "collection point" in a packet network. It is at these node locations that packets are collected and routed to other nodes and end-users. Many network nodes operate at high data rates, usually at 9,600 or 19,200 baud. End-user access to the network is usually at a lower baud rate (1200 baud) and on a different frequency. The advantage of a network system is two-fold: automatic routing of connections to other network nodes and increased data throughput. A network node "knows" how to automatically route data through neighboring nodes to distant nodes on the network. They do this by "hearing" other nodes broadcast this information. This means that a user connected to a node does not need to specify the particular route to take in order to reach a distant node as long as the local node contains a listing for that distant node. Using a network node can also dramatically increase the data throughput to distant locations. The packet data and the acknowledgments travel from node to node (rather than from end-to-end), greatly reducing the inevitable data collisions that will occur when attempting to communicate over long distances.

The defacto standard networking protocol, NET/ROM (tm), was developed in the late 1980's by Software 2000, Inc. Since then, other node derivatives such as TheNET, TheNET Plus, TheNET X-1J, and G8BPQ have evolved. Although these variants differ in their features, they all utilize the same basic networking protocol.

Some network nodes have more than one radio port in order to provide a high-speed network "backbone" on one frequency, as well as Local Area Network (LAN) access to the high speed network at a lower data rate on a different frequency. This approach to packet networking makes good sense for a number of reasons. First, the high speed backbone is free to pass large amounts of data without competing with end users for the channel. Generators of large amounts of data are called servers. Many times servers are busy communicating with other servers, so dedicating a channel to this function is desirable. Second, users are "shielded" from the server-to-server activity. Many users could peacefully co-exist with a server if all the server did was to respond to user queries. User input to a server generally consists of very small amounts of data that generate fairly large responses. In a local area, this works quite well since all users can detect the presence of the server's data. Carrier Sense Multiple Access (CSMA) allows the channel to be shared nicely. However, servers, and in particular many Bulletin Board Systems (BBS's), spend much more time forwarding and receiving messages from neighboring bulletin board systems than they do actually serving the local user population! It is not unusual for a BBS to attempt forwarding (AND reverse forwarding) on an HOURLY basis. This situation quickly evolves into quite a paradox - BBS's provide an extremely popular service that has been a boon to the growth and popularity of packet radio. As user numbers increase to take advantage of the services a BBS provides, BBS systems also increase in number in order to provide the service to more and more users. More and more BBS systems must now forward to each other, generating huge amounts of data on the channel, leaving precious little time for the very source of their existence - the user community. User-to-user (keyboarding) QSO's become, at best, a very frustrating proposition. As a result, many users become ex-users, interest declines, and packet usage stagnates. With the K-Net prom installed in your KPC-9612, cost-effective networking can benefit many packet users in your area immediately.

BASIC NETWORKING GUIDELINES

The intent of this section is not to dictate how to configure a packet network. However, certain goals and concepts of design should be carefully considered. The primary goal of any data network should be the efficient transfer of data from point A to point B (high speed), coupled with ease of use (low frustration level). The efficient transfer of data is made possible with the advent of high-speed modems and "data-ready" radios capable of operating at 9,600 and 19,200 baud. These modems and radios are now available from a number of commercial sources. This means that making internal radio modifications for high-speed data operation is quickly becoming a thing of the past.

User access to the high-speed channel is generally accomplished by using a separate, low-speed port at a network node site on a different channel. This protects users from server-to-server activity and also provides a relatively clutter-free Local Area Network for keyboard-to-keyboard activities.

In an ideal packet network, all high-speed links would operate on a dedicated channel (backbone) without contention. This would require that no stations other than network nodes and servers reside on the backbone channel. Each node on the backbone would have a "rock solid" link to the next, preferably on alternating, or full-duplex channels for maximum data transfer efficiency. All users in a community would be grouped into Local Area Networks (LANS) with each LAN assigned a specific operating frequency such that adjacent LANS would not interfere with each other. Each LAN would have a user port at a local network node that would provide access to servers and other LANS in the region. This situation would provide very efficient server-to-server communication, while offering users an enjoyable environment consisting of a useable keyboard-to-keyboard channel and easy access to servers and distant LANS. In reality, much of the above is quite difficult to achieve. Obstacles include finding suitable node sites, available funds for equipment, and dealing with the differing philosphies and interests of other node SYSOPS. There are many differing opinions on how to go about achieving "Network Utopia", but virtually all of them have one goal in common - make packet radio fun!

WHAT YOU CAN DO WITH THE K-Net PROM INSTALLED IN THE KPC-9612 or KPC3

The K-Net PROM adds the function of a DUAL-PORT NETWORK NODE to the operation of your KPC-9612. ALL ORIGINAL FEATURES OF THE KPC9612 or KPC3 ARE RETAINED, including your PBBS, simultaneous dual-port keyboard operations, and remote access. This means that you can still use the KPC-9612 as you do now, and add the K-Net PROM, to provide your local area with a dual-port network node that operates in the background of your normal TNC operations! K-Net is fully compatible with the basic network protocol and provides both a 9,600 baud port and a 1200 baud port that can be added to existing networks. The 9,600 baud port can be used for direct server access and/or a backbone node, while the low-speed (1200 baud) port is available for user access to servers and other LANS in the area. Create A Supernode - 3, 4, or more radio ports! See Node Stacking section of this manual.

INSTALLATION OF THE K-Net PROM

Please note that after the K-Net PROM is installed, the KPC-9612 will power up in the autobaud mode and reset all commands to the factory defaults. Before you install the K-Net PROM, it's a good idea to use the DISPlay command to make a print-out (or disk file) of your parameters. To install the K-Net PROM, you will need to remove the cover of the KPC-9612, locate and remove the old PROM U19 (*U10 in the KPC3*), insert the new K-Net EPROM, perform a hard reset, and then reattach the cover. The procedure is as follows:

1.   Turn off the KPC-9612 or KPC3
2.   Remove the two screws (one on each side) that secure the top cover to the case and remove the top cover.
3.   Observe proper anti-static precautions and remove the PROM from socket U19 (*U10 in   the   KPC3*). The EPROM has a white Kantronics label on the top.
4.   Carefully insert the new K-Net PROM in socket U19 (*U10 in the KPC3*), insuring that   pin   1   (notch end) of the PROM is located closest to the large notch in the PC board near the battery. Also note that the new KPC-9612 PROM is physically larger than the old one. The K-Net PROM will use all 32 pins
of   the U19 (*U10 in the KPC-3*) socket. When inserting the new PROM, be careful not to bend any of the pins underneath the PROM.
5.   Perform a *HARD RESET* as described on page *105 of the KPC-9612 & KPC3* Reference Manual and reassemble the KPC-9612 (KPC3).

QUICK START

The factory default values in the K-Net PROM have the network node operation disabled. To enable the K-Net node, the NETALIAS and the NETCALL must be entered. The NETALIAS is usually a mnemonic that gives users an idea about where the node is located (other network nodes do not care what the alias is). For example, SUTNE is a good NETALIAS for Sutton, Nebraska. The NETCALL is the callsign of the node. The NETCALL consists of your callsign plus an optional SSID (Example - KA0DNV-1). It MUST be different from any other callsign used in the KPC-9612 (MYPbbs, MYGate, MYAlias, MYNode, etc...). Use the DISPlay Id command to make sure that the NETCALL you select is different from the other callsigns that are in use by the KPC-9612. Many times, local "customs" will serve as guidelines for what NETALIAS and NETCALL to use.

Enter the NETALIAS - cmd:NETALIAS SUTNE

Note that this is a dual-port command, allowing a different NETALIAS on each port.

Set the NETCALL - cmd:NETCALL KA0DNV-1

This command will cause the KPC-9612 (KPC3) to perform a soft reset as memory is allocated for node operation. Your K-Net node is now in operation!

There are six additional "NET" commands that are only available from the command prompt of the KPC-9612. These commands and their current settings can be displayed with the DISP N command.

      Example -   cmd:***DISPlay N***
                      NETBUFFS 40
                      NETCURCS 5
                      NETDESTS 20
                      NETLINKS 4
                      NETROUTE 5
                      NETUSERS 4

The INtface command has also been changed to include a ***NET*** option (see Node Stacking section).

Most other node commands are accessed by connecting to the node either locally with your terminal, or remotely. If connected remotely, SYSOP access (see SYSOP command) is required to list and change SYSOP related parameters. NOTE: RTEXT command in the KPC-9612 MUST be set to allow remote SYSOP access). Following the above example:

                      cmd:***C*** SUTNE <cr>

This will cause the KPC-9612 to do an "internal connect" to the K-Net node alias TT77, much like connecting to your own PBBS. When you connect to your K-Net node in this manner, you will <u>not</u> receive a command prompt (cmd:). Hitting the ENTER key will display the commands available to you.

To see the current setting of any of the commands just enter the command and hit the ENTER key. Whenever you are at this "invisible command prompt" you automatically have access to the entire command set, since you are assumed to be the SYStem OPerator (SYSOP).

A short help description of each command is also available by entering H(elp) or ? followed by the node command (Example - To get on-line Help for the Addnode command, enter H A<cr> or ? A<cr>). The entire HELP contents can be displayed by entering H H<cr>.

TEXT MESSAGES

Now that the node is operating, you will probably want to enter some text messages that will provide users with some helpful information. These messages are entered by connecting to your node (cmd:C TT77) and entering the commands followed by the text (See CTEXT, INFO, and PORTS in the Command section.

COMMAND SECTION

AN ASTERISK (*) PRECEDING A COMMAND INDICATES A SYSOP COMMAND
All other commands are available to users of the node

NOTICE - IN ORDER TO GRACEFULLY INTEGRATE YOUR K-Net NODE INTO AN EXISTING NETWORK, IT IS <u>IMPERATIVE</u> THAT YOU COORDINATE YOUR EFFORTS WITH EXISTING NETWORK SYSOPS. THERE ARE NUMEROUS NODE PARAMETERS THAT CAN SERIOUSLY DEGRADE NETWORK PERFORMANCE IF SET TO CONFLICTING VALUES.

The command section contains some networking terms that MUST be well understood.

NODE -   The term node refers to a network node such as a G8BPQ, X1J, or NET/ROM node. Non-network nodes such as KA-NODE, conference bridges, and digipeaters do not apply to this concept of operation.

USER -   A user is either a real, live person using a TNC to connect to and use your node, or a server (BBS) that is basically doing the same thing under computer control.

NEIGHBOR NODE -   A neighbor node is a node that your node can connect to "direct", without using an intermediate node. Note that digipeater operation is allowed (2 max.). This means that a node can be "out of range", but can still be considered a neighbor since the connect path does not use any other nodes. All known neighbor nodes are displayed with the routes command.

DISTANT NODE -   A distant node is a node that is too far away from you to connect to "direct", requiring the use of one or more network nodes to reach. Distant nodes will not be displayed with the routes command.

DESTINATION NODE -   A destination node is a node that your node knows how to connect to - either directly (ie...neighbor node), or, in the case of a distant node, by using one or more intermediate network nodes. All known destination nodes (neighbor nodes AND distant nodes) are displayed with the Nodes command.

ALL NEIGHBOR NODES ARE DESTINATION NODES, BUT ALL DESTINATION NODES ARE NOT NECESSARILY NEIGHBORS. Destination nodes will automatically appear in the node table by virtue of your node monitoring your neighbor node broadcasts. A neighbor node will automatically be listed in your node table AND routes table. Neighbor node broadcasts also contain information about other nodes that you may or may not be able to hear "direct". These distant nodes will also be listed in your node table if they pass the criteria for being listed (MINQual), and if there is sufficient room for them to be listed (NETDest).

*__Addnode__   [alias:]call port neighbor [via digi1[,digi2]] quality [obscnt]

          This command is used to add a new destination node or to modify an existing destination node in your node table. If the optional obscnt (Obsolescence Count) is set to 0, the node will be PERMANENTLY listed in the node table. Such a route will not be updated or dropped from the nodes table by the K-Net auto-routing logic. If the neighbor node callsign entered is not in the neighbor routes table, it is added with the default quality for the specified port.

As long as MINQUAL is equal to or less than QUALITY, destination nodes (neighbor nodes as well as distant nodes) will be added automatically to your node table as neighbor node broadcasts are received.

EXAMPLE - Addnode KSLVN:WD0EMR-1 2 WK5M-1 200
> K-Net responds: Node added with new route

> This adds the KSLVN node with the netcall of WD0EMR-1 to the node table. KSLVN is available on port 2 (9,600 baud) of the 9612 using the WK5M-1 (netcall) node as your neighbor. An initial quality of 200 has been assigned to the new node. Unless "locked in" with the ADDRoute command, this path quality will most likely change since it is recalculated based on node broadcasts that are received. In this case, WK5M-1 was a previously unknown neighbor and the new route was automatically added.

EXAMPLE - Addnode WLAW:W0XI-1 2 W0XI-1 200
> K-Net responds: Node added with new route

> This would add the WLAW node with the netcall of W0XI-1 to the node table. In this case, W0XI-1 is also the neighbor and is available on port 2 with an initially assigned quality of 200.

EXAMPLE - Addnode YORK:WA0CPS-1 2 WK5M-1 70
> K-Net responds: Node added

> This would add the YORK node with the netcall of WA0CPS-1 to the node table. Since WK5M-1 is an existing route (neighbor) a new route is not added.

EXAMPLE - ADDNODE CRT:KG5DT-1 1 WK5M-1 70 0
> K-Net responds: Node added

> In this case, the optional obscnt (obsolescence count) has been specified as zero. This special case is useful in the event that your node is not able to hear this neighbor's node broadcasts (ie...if a   digipeater(s) or gateway is being used). Normally, if a node is not heard from for a period of time, the obsolescence counter will eventually drop it out of the node table. However, this node (CRT) will remain in the node table permanently, unless removed by the sysop.

*__ADDRoute__ port call [via digi1[,digi2]] quality [!]
This command will add (or modify) the specified netcall of a neighbor node to the routes table. If the optional ! is specified, the route entry is either locked, if unlocked, or unlocked if locked (toggle). An existing locked route does not have to be unlocked before it can be modified. Locking a neighbor's route is used to assure that an assigned quality will always exist for that neighbor. The locked quality will also be used in the automatic quality computation of other distant nodes contained in this neighbor's node broadcasts to determine if they should also be added to your node table. In the event that this locked neighbor disappears from the network, and then returns, it will automatically be assigned the locked quality. During the time that this neighbor is absent, it, and distant nodes associated with this neighbor, may eventually be removed from your node table (even if the route is locked).

This is because the obsolescence count for this neighbor could drop to zero. If this neighbor's route is locked, it will still be listed in the route table, but with a zero count for destination nodes using that route.

> EXAMPLE - ADDRoute 1 W0XI-1 110 !
>   K-Net responds: Route added and locked
>
> This adds the route to the W0XI-1 node to the route table in the K-Net node. The route is "locked in" and assigned to port 1 with a quality of 110.

Locking a route assigns a fixed QUALITY to a neighbor node. This can assure that any connects to a distant node using this neighbor node as a route will take the "hard coded" route, even if the route automatically generated by the K-Net node is different, such as may occur during band openings.

> EXAMPLE - ADDRoute 1 W0XI-1 110 !
>   K-Net responds: Route modified and unlocked
>
> Now the W0XI-1 node is "unlocked" since the ! is a toggle function.

> EXAMPLE - ADDRoute 1 W4NGA-1 0 !
>   K-Net responds: Route added and locked
>
> This adds the W4NGA-1 node to the routes table with a locked quality of 0. This node, and any distant nodes associated with it, will not appear in the node table since the locked in Quality of zero is probably less than MINQual.

NOTE - The W4NGA-1 node can be COMPLETELY ignored by putting its NETCALL (W4NGA-1) in the supcalls list and turning llist ON (from the command prompt - cmd:). This would prohibit the K-Net from connecting to as well as accepting a connect from the W4NGA-1 node.

BBS   {immediate command}   [/S]
This command connects a user of the node to the PBBS of the KPC-9612. If the PBBS is not enabled, this command is not operational and will not appear in the node's HELP list. If the optional S(tay) parameter is used, the user will be returned to the node after sending the B(ye) command to the PBBS.

Bye   {immediate command}
This command causes the node to send a disconnect packet to the user.

Connect   [port] [call|alias] [/s]   {immediate command}
The Connect command is used to establish a connection to another node or to another user/server. If connecting to another node that is listed in the node table (see Nodes command), the KPC-9612 will automatically select the port and path to be used. If connecting to a user (any callsign or alias NOT in the nodes table), you must specify the port and path. If the port is not specified, the connect attempt will be made on the same port the user connected to. A connect using the optional /S will use the STAY option - if the remote station issues a disconnect, you will be returned to the node.

If a user wishes to connect to you, and enters just a C (or C /S to Stay), the node will attempt to "connect" internally to the SYSOP of the KPC-9612. If you have a terminal connected, you can communicate just like a traditional packet session. If you set the CMSG command in the KPC-9612 to PBBS, the user will be routed to your PBBS and will be able to leave you a message.

CQ   [text]   {immediate command}
This command places a user in the CQ mode. The text is transmitted by the node on the port(s) specified by the CQBC command. If another user tries to connect to the callsign contained in the CQ packet during the next 15 minutes, the node will then establish the link back to the user initiating the CQ. Any other text sent by the user that does NOT begin with CQ will cancel the CQ mode.

EXAMPLE - If a user (WA0CPS) connects to the node and gives the command:

CQ This is Ed in York, Nebraska - Please connect to me for a short QSO.<cr>

The node will transmit a packet addressed TO CQ from the node-modified callsign of WA0CPS-15.

WA0CPS-15>CQ: This is Ed in York, Nebraska - Please connect to me for a short QSO.

The node automatically subtracts Ed's SSID (0) from 15 (15 minus 0 = 15) since the callsign WA0CPS is already being used in the circuit. If Ed's callsign was WA0CPS-2, the node would modify it to WA0CPS-13 (15 minus 2 = 13).

If another user tries to connect to WA0CPS-15, the node recognizes the connect request and the two users are connected through the node.

CQBc   ON|OFF   {ON/ON}
This dual-port command controls which port(s) are used to transmit a CQ broadcast from a user. By default, the CQ will be transmitted on both the 1200 baud port and the 9600 baud port. For example, if you do not wish any CQ broadcasts to take place on your 9600 baud port, set the command CQBC ON/OFF.

CText   text   (up to 128 characters) {blank}
This command sets a text message that is sent to users when they connect to your node alias (NETALIAS) but NOT when a user connects to the node callsign (NETCALL).
It is also not sent when the user connects to your node by using another network node.

Example - CText Welcome to the SUTNE K-Net node, operated by KA0DNV, in Sutton, Nebraska

You may clear this text by entering a % as the CText (cmd:CText %)

Delnode   [alias:]call port neighbor [via digi1[,digi2]]
This command is used to delete a destination node from your node table. If the node you are trying to delete has an alias (showing with the Nodes command), you must specify both the alias and the call of the node to be deleted. If this node is the only one using the specified neighbor as a route, and if the neighbor is not locked in the routes table, the neighbor will also be deleted from the route table.

*DELRoute   port call [via digi1[,digi2]] quality

The DELRoute command is used to delete a neighbor from the routes table. A neighbor route cannot be deleted if there are any nodes in your node table using this neighbor as a route.

EXAMPLE - DELROUTE 1 W0XI-1 110
K-Net responds: Route deleted

EXAMPLE - DELROUTE 2 WK5M-1 110
K-Net responds: Route unlocked

In this case, the route was not deleted, and the "Route unlocked" response simply indicates that the route is not locked in the route table. This response will be received if this neighbor is used as a route by other destination nodes in your node table. In order to delete this route, ALL destination nodes that use this neighbor as a route must be deleted first.

Help   {immediate command}

This command displays a list of all commands available to the connected user. Giving the Help command followed by an available node command will display a one-line help message about that command. Entering the Help Help command will give a one-line Help message of all available commands.

Info   text (text up to 128 characters)   {blank}

This command sets a text message that is sent to users when they enter the Info command after connecting to your node.

Example - Info Located 3 miles N.W. of Sutton, NE - 50 watts at 100 feet. Hit ENTER for available commands or C to Connect to me.

You may clear the Info text by entering a % as the info text (Info %).

*IDint n   (n = 0 - 255) {10/10}

The dual-port IDint command sets the time interval (in minutes) between node ID packets (NOT node broadcasts). Setting this command to 0 disables node ID's.

**INtface   NEWUSER|TERM|BBS|KISS|HOST|GPS|NET   {NEWUSER}

The NET option has been added to the INtface command. Setting the INtface command to NET and powering the KPC-9612 off and then back on allows the serial port to operate as a dedicated "virtual radio circuit". This allows multiple, co-located network nodes to communicate with each other via their serial ports. See the Node Stacking section of this manual for more information.

Links {immediate command}

The Links command displays a list of current AX.25 links existing in the node. Links displays the callsign of the two connected stations, the AX.25 State (normally 5), the Port number, the Type of link (U = Uplink, D = Downlink, and X = Crosslink), and the AX.25 Version in use (1 or 2).

EXAMPLE - L
K-Net responds: WK5M KARL S=5 P=1 T=U V=2

This shows that WK5M is connected to KARL, the AX.25 state is 5 (connected), using port 1, it is an Uplink using AX.25 version 2. Note: An UPLINK is an end user/server connected TO the node, a DOWNLINK is FROM the node to an end user/server, and a CROSSLINK is from the node to another node.

Possible State Values (S) are:

1	Disconnected
2	Connect in progress
3	FRMR condition detected
4	Disconnect attempt in progress
5	Connected
6	Waiting acknowledgment
7	Device busy
8	Remote device busy
9	Both devices busy
10	Waiting acknowledgment and device busy
11	Waiting acknowledgment and remote device busy
12	Waiting acknowledgment and both devices busy

THE NEXT SIX COMMANDS DEAL WITH THE ISO OSI (OPEN SYSTEMS INTERCONNECTION) REFERENCE MODEL OF NETWORK STRUCTURE, IN PARTICULAR LEVEL 3 (NETWORK LAYER), AND LEVEL 4 (TRANSPORT LAYER).

***L3ttl** n (n = 0 - 255) {25}

This command sets the limit on the number of times a packet originated by an uplink (normally a user) at this node may be transferred through other nodes. Each time a message is handled by a node, the time-to-live for that message is decremented. If the L3ttl counter reaches zero, the message is discarded. It is possible for network nodes to create circular paths, and therefore a limit is required to prevent a packet from relaying in the node system forever. L3ttl should be standardized network-wide.

***L4delay** n (n = 1 - 60) {5}

This command sets the delay time in seconds used by the node after receiving a Level 4 frame from another node before the ACK is returned. Frequently there will be data ready to be passed in the opposite direction shortly after receiving a frame, and this delay permits the acknowledgement to be "piggy-backed" with the data frame, thus reducing the number of transmissions on the channel.

***L4Limit** n (n = 0 - 65,535) {900}

This command sets a no activity timeout (in seconds) on a node crosslink. When a user connects to another node, the node first establishes a level 2 (AX.25) connect with that node, and then establishes a level 4 circuit on behalf of that user. When the user is finished, the level 4 crosslink is closed, but the level 2 connection remains until the L4Limit is reached, at which point it is terminated. When a user disconnects, the level 2 connection between the nodes remains intact and the L4Limit timer is restarted. This decreases data congention if another crosslink request using the same path occurs before the L4Limit expires since the node will not have to re-establish the level 2 connect.

***L4N2** n (n = 1 - 127) {3}

This command sets the number of retries used to pass data between nodes (Level 4) in the system. After the L4N2 count reaches zero, alternate paths will be tried.

***L4T1** n (n = 5 - 600) {120}

The L4T1 timer sets the time in seconds that the originating node will wait before resending a Level 4 data packet if no acknowledgment is received from the destination node (end-to-end INFO ACK). If L4T1 is set too small, multiple copies of the same packet will exist within the node system even if the first packet was received correctly.

***L4Window** n (n = 1 - 127) {4}

The L4Window sets the maximum number of outstanding frames that may exist in the network for a given circuit. When the node has L4Window un-acknowledged frames, it will not transmit any more frames until a level 4 info-ack is received. Network nodes may allow up to 127 outstanding frames across the system. The L4Window size is negotiated between nodes when a crosslink is requested, therefore the L4Window size should be standardized network-wide.

Mheard [S|L] {immediate command}

This command displays a list of stations recently heard by the node. It displays the callsign followed by a /1 to indicate the station was heard on port 1, or a /2 to indicate the 9600 baud port. A time stamp is also shown (unless the SHORT option is specified). If LONG is specified,

the node displays the TO and FROM callsigns of the stations heard, as well as any digipeaters used.

*MInqual   n   (n = 0 - 255)   {70}
This command sets the minimum Quality required to add a destination node to the node table. Making this value higher will decrease the size of the node table, making this value lower will increase the number of destination nodes listed. When K-Net hears neighbor node (A) transmit a node broadcast, it computes the quality to distant nodes (B, C, D, etc...) contained in that broadcast by using the quality that is assigned to neighbor node (A). If the resultant computation is less than MInqual, the distant nodes (B, C, D, etc...) are not added to the node table (see Quality command). The default settings of Quality = 70/110 and MINQual = 70 means that all nodes that are heard "direct" (neighbors) will be added to the node table. If MInqual is greater than Quality, a new neighbor node will not be automatically added to the node table, nor will it be displayed with the Routes command.

"NET" COMMANDS

The following "NET" commands can only be entered at the command prompt (cmd:) of the KPC-9612 or KPC3. "NET" commands (with the exception of NETAlias and NETCALL which are contained in the ID DISPlay group), and their current settings can be listed by entering DISPlay N at the command prompt (cmd:DISP N).

*NETA*lias - (up to 6 alpha-numeric characters)   {blank/blank}
The NETALIAS is an optional dual-port mnemonic that usually provides information about the location of the node. It exists only for the convenience of users. Network nodes automatically convert the NETAlias to the associated NETCALL whan setting up a circuit for a user. Changing or entering NETAlias will not cause a soft reset.

*NETB*uffs   n   (n = 1 - 255)   {32}
The NETBuffs command sets the number of node buffers available in the node. If the NETBuffs reach zero, node operation may become unstable. Changing NETBuffs will cause a soft reset as memory is reallocated. Each NETBuff allocated uses 320 bytes of RAM.

*NETCA*LL - (up to 6 alpha-numeric characters) [SSID] {blank}
The NETCALL is the callsign of the K-Net node and can have an optional Secondary Station Identifier (SSID) extension of 1 to 15. The NETCALL MUST be entered before the K-Net can be operational and MUST be different from any other callsign used in the KPC-9612.

Entering a % as the NETCALL will clear both the Nodes and Routes tables, but not change any node-related parameters. Entering or changing the NETCALL will cauwe a soft reset as memory is reallocated.

*NETCI*rcs   n   (n = 1 - 64)   {5}
This command limits the number of Network Circuits available in the node. A CIRCUIT is a connection between two nodes. The CIRCUIT will contain an UPLINK (originated by a user or server), a CROSSLINK (the other network node), and usually a DOWNLINK (to another user or server). The DOWNLINK will not exist until requested by the originating UPLINK user. The CIRCUIT will exist until either the destination user or server or the originating user or server disconnects. The CIRCUIT will also be discontinued if one of the nodes determines that data

can no longer be sent due to a retry time-out (L4N2), or if the inactivity timer (L4T1) expires. Changind the NETCIrcs will cause a soft reset as memory is reallocated. Each NETCIrc allocated uses 50 bytes of RAM.

***NETD**ests n (n = 1 - 255)   {25}

      Setting the NETDests determines the maximum number of destination nodes that can be stored in the K-Net node list. This list is sent to users when the Nodes command is given to the K-Net node and also when the node does a nodes broadcast. Setting NETDests to a very large number (ie...greater than 50 or so) is generally not a good idea since this can create long node broadcasts and long transmission times to users in response to a Nodes command. Setting NETDests to a very low value can result with a desired node getting "bumped" out of the node table.

After your K-Net has been in operation for a while, you should be able to determine how many of the destination nodes that are listed in your nodes table are <u>reliable</u> paths. The value that you enter for NETDests should be greater than this number, allowing room for other nodes as they become available. In any case, never allow the node destination table to become so large that it is filled with unusable and unreliable destination nodes. Very large node tables (>50 or so) are simply baffling to new users, and most of the time unusable to even experienced "node hoppers".

Although setting NETDests will place a "hard limit" on the number of destination nodes that can appear in the nodes table, proper setting of the K-Net QUALITY and MINQUAL commands will allow the number of destination nodes listed in your node table to "float" with changing band conditions and still assure that reliable destination nodes are still listed. Changing NETDests will cause a soft reset as memory is reallocated. Each NETDest allocated requires 31 bytes of RAM.

***NETL**inks   n   (n = 1 - 64)   {10}

      The NETLinks command sets the maximum number of UPLINKS, DOWNLINKS, and CROSSLINKS available to the node. An UPLINK is a user that connects directly TO the K-Net node without using another network node. A DOWNLINK is a connection FROM the K-Net node directly to a user. A CROSSLINK is a connection between two nodes. Changing NETLinks will cause a soft reset as memory is reallocated. Each NETLink allocated requires 130 bytes of RAM.

*NETR*oute   n   (n = 1 - 32)   {5}

      This command sets the maximum number of neighbor nodes that can be listed with the routes command. A neighbor node is a network node that your K-Net can hear "direct", without the benefit of being relayed by another network node. As with the NETDESTS parameter, if the setting of NETRoute is too low, not all of your neighbors will be able to be listed. Setting NETRoutes to a higher number will allow you to see what the local node activity is and still have some room in reserve to handle itinerant nodes, and nodes that will show up during a band opening. Changing NETRoute will cause a soft reset as memory is reallocated. Each NETRoute allocated requires 39 bytes of RAM.

*NETU*sers   n   (n = 0 - 26)   {5/5}

      NETUsers sets the maximum number of UPLINKS and DOWNLINKS from the node. This dual-port command places a limit on how many users can access or be accessed by the node. The NETUsers command will not cause a soft reset when changed or entered.

Nodes [{ * | alias | call}]   {immediate command}

This command displays a list of all known destination nodes. The list displays the ALIAS:CALLSIGN for each destination node. If the optional * is given, hidden nodes (those beginning with #) are also shown. Specifying an alias or call with the Nodes command will display up to three possible routes (neighbor nodes) of descending quality that will be used to establish a link to the specified destination.

EXAMPLE - N YORK
    K-Net responds:
    Routes to YORK:WA0CPS-1
     200 5 2 WA0CPS-1
     170 5 2 KA0DNV-1
     75 3 1 W0XI-1

This list shows the three best routes K-Net will attempt to use (in descending order) to connect to the YORK node. The first number is the route Quality, the second number is the obsolescence count, and the third number is the assigned port.

*NODESInt n (n = 0 - 255) {60}
This command sets the time interval (in minutes) between nodes broadcasts (NOT node ID'S). A new setting will be in effect after the previous interval has expired. This interval should be set to an agreed-upon value for all nodes in the area. This broadcast provides your neighbor nodes with update information which they use to add, modify, and delete nodes automatically from their node and routes tables. Each time your node performs a nodes broadcast, the obsolescence counter for all destination nodes in your node table is decremented by one.

*Obsinit n (n = 0 - 255) {5}
The Obsinit command sets the initial obsolescence count for a destination node. When a nodes broadcast is heard from a neighbor, or when a link is established with a destination node, the obsolescence count is set to Obsinit for that destination node.

Each time your node transmits a nodes broadcast, the obsolescence for each destination node is decremented by one. When the obsolescence value drops below OBSMin for any destination node, that destination node will not be contained in your nodes broadcast. However, it will remain in your node table until the obsolescence count reaches zero, at which time it will be removed unless that node had been added (Addnode) with an initial obsolescence count of zero).

*OBSMin n (n = 1 - 255) {4}
This command sets the minimum obsolescence count which will allow a destination node to be included in your nodes broadcast. The default values of Obsinit 5 and OBSMin 4 allow your node 2 chances to receive a "refreshing" nodes broadcast or a connect from a destination node before that destination node is dropped from your nodes broadcast. When the obsolescence count for a node reaches 0, that node is dropped from the node table unless it had been added or modified (Addnode) previously with an initial obsolescence count of zero.

Ports text/text (text up to 128 total characters) {blank}
This command sets a dual-port text message that is sent to users when they enter the Ports command after connecting to your node.

Example - PORTS 145.01 MHz @ 1200 baud/430.55 MHz @ 9600 baud
This text can be cleared by entering a % as the port text (Ports %).

*Quality   n   (n = 0 - 255)   {255/70/110}

> This is a three-port command (Port 0 = RS-232 port, Port 1 = 1200 baud radio port, Port 2 = 9600 baud radio port) that assigns the Quality that is automatically given to a new neighbor node when it's nodes broadcast is heard. It is also used to calculate the path quality to distant nodes that are listed in the nodes broadcast from a neighbor node. Port 0 Quality defaults to the highest value possible (255) because a back-to-back serial port path is extremely reliable (See Node Stacking section). Also, since it is usually advantageous to route data over the 9600 baud port instead of the 1200 baud port, a higher default quality is given to port 2. The quality value is important for two reasons.

> The quality of a node (either automatically or manually assigned) will determine how far away a node will be visible in the network. Proper use of the quality parameter is necessary in order to maintain a USEABLE node listing by not allowing a node to propagate useless distances over the network.

> The quality command can also be used to assure that known "good" routes are chosen by the auto-routing routines. In the event of a band opening, it is possible for distant nodes to be heard "direct" and be automatically assigned a higher quality than normally calculated by using "good" neighbor routes. If a user attempts to connect to this distant node, the first choice route (direct) will probably fail, and then the second best choice (which may actually be the most RELIABLE route) will be tried.

> The K-Net node will automatically calculate the quality to distant nodes based on the quality that is received from neighbor nodes when neighbor nodes broadcast their known nodes. If your node hears one of your neighbor nodes (A) broadcast a quality of 70 for a distant node (B), your K-Net will automatically determine the quality of the path to distant node (B) that will use your neighbor (A) in the route. This is done by multiplying the quality of the route assigned by you to your neighbor node (A) by the quality of the distant node (B) received within the nodes broadcast from your neighbor (A) and dividing by 256. As an example, if your node has an assigned quality of 70 to your neighbor (A), and receives a nodes broadcast from neighbor (A) containing a quality of 70 to distant node (B), the K-Net will assign a quality of 19 to distant node (B) (70 multiplied by 70 divided by 256 and dropping any fraction of a whole number). If MInqual is set to any value greater than 19, distant node (B) will not be added to your node table. On the other hand, if the route quality to your neighbor had been modified with a higher value, such as 200, a quality of 54 would then be assigned to distant node (B) (70 multiplied by 200 divided by 256). In this case, if MINQual is set to 54, this distant node would be listed in your node table. After you determine a particular quality that works best for a neighbor, "locking" the quality (!) will assure that the desired quality will always exits for that neighbor - even if that neighbor disappears from the network and then returns.

Routes {immediate command}

> The Routes command will display the list of <u>neighbor</u> nodes. In order for a new neighbor to be contained in the node and routes tables, MINQual MUST be equal to or less than Quality. The Routes table shows the port number, the neighbor callsign, quality value, the number of destination nodes in the node table that use this neighbor as a Route, if the route is in use, and if the route is locked.

For example:

        R <cr>

```
          KARL:WK5M-1}Routes:
          >2 KA5ZTX-1 110 1
              1 KA5ZTX-1 70 1
              1 WD0EMR-1 63 7
          >2 KA0DNV-1 85 15
              2 WA0CPS-1 110 12 !
              2 W4NGA-1 0 0 !
```

This Routes listing shows that the KARL:WK5M-1 node can reach neighbor node (KA5ZTX-1) on port 2 with a quality of 110 and also on port 1 with a quality of 70. On either port, there is only one destination node (itself - KA5ZTX-1) in Karl's node table that can be reached using this neighbor.

WD0EMR-1 is also a neighbor node for WK5M-1. This node has a port 1 quality of 63. Using the WD0EMR-1 node as a route, Karl's node can reach seven of the destination nodes listed in its node table (WD0EMR-1 plus 6 others).

The KA0DNV-1 node has a path that is available to WK5M-1 on port 2 of the K-Net node. The path quality is 85 with 15 destination nodes in Karl's node table able to be reached using this neighbor as a route (KA0DNV-1 plus 14 others). A right arrow (>) indicates that this route is either in use or has been used within the L4Limit time period.

The next neighbor node is WA0CPS-1. It is reachable on port 2 with a quality of 110.

There are 12 destination nodes in Karl's node table that can be reached using this neighbor as a route (WA0CPS-1 and 11 others). The exclamation point (!) indicates that the route as well as the route quality (110) has been "locked in" by the SYSOP of the WK5M-1 node.

The last neighbor node listed is W4NGA-1. Since the Quality has been set and locked to zero, there are no destination nodes (including W4NGA-1) listed in Karl's node table that can be reached using this neighbor as a route. This node will not be listed in Karl's node table, and all node broadcasts sourced from this node will be ignored.

Stats   {immediate command}

This command displays a summary of statistics concerning level 3 and level 4 activity on this node. A typical display:

```
KARL:WK5M-1}
Time active 3986 mins
Buffers: 200 max, 199 available, 169 min, 0 out
Known nodes:  19
L4 Connects: 26 sent, 5 rcvd
L4 Frames: 205 sent, 392 rcvd, 6 resent, 0 resequenced
L3 Frames Relayed: 2303
```

This shows that the node has been active for 3986 minutes, there are 200 buffers in the node, 199 of those buffers are available. In addition, it indicates that, at some time, there were only 169 buffers available, and that the node has never run out of buffers. If min Buffers nears zero, or if the buffers have run out, the NETBuffs parameter should be increased. It further shows that there are 19 nodes listed in the node table, there have been 26 level 4 connects sent and 5 received.

205 level 4 frames have been sent, 392 received, 6 resent (retries) and none have been resequenced. The level 3 activity shows that 2303 Level 3 frames have been relayed.

SYsop {immediate command}

The SYsop command can be used by a remote user to access SYsop commands (marked with *). When a user gives the SYsop command, the node responds with three lines of numbers. The user must then convert one of these three lines into the corresponding characters of the RTEXT in order to be validated as SYsop. This operates the same as the remote access in your KPC-9612 and uses the same RTEXT (See the "Remote Access" section in your KPC-9612 Reference Manual). If connecting to the node with an attached terminal, SYsop validation is automatic.

Users {immediate command}

This command displays a list of Users currently connected to the node. It also displays the number of buffers currently available (in parentheses). Uplinks, downlinks, and crosslinks will be displayed by this command.

FOR EXAMPLE:

Node user enters:
USERS

K-Net responds:
KSLVN:WD0EMR-1} K-Net Packet Switch v1.0 (145)
Uplink 1 (WA0PWS) <--> Circuit(KSBLC:WZ0M-3 WA0PWS)
Circuit(KSLAW:WK5M-1 WA0CPS) <--> Downlink 1 (WA0CPS-15 KB0VA)
Uplink 2 (KA0DNV)

KSLVN:WD0EMR-1} K-net Packet Switch v1.0 (145)

There are currently 145 buffers available (145) at the KSLVN:WD0EMR-1 node.

Uplink 1 (WA0PWS) <--> Circuit(KSBLC:WZ0M-3 WA0PWS)

WA0PWS is a user that has connected (Uplinked) on port 1 to the KSLVN node. WA0PWS has requested, and KSLVN has established, a Circuit (Crosslink) to the KSBLC:WZ0M-3 node on behalf of WA0PWS.

Circuit(KSLAW:WK5M-1 WA0CPS) <--> Downlink 1 (WA0CPS-15 KB0VA)

A Circuit, (Crosslink) from KSLAW to KSLVN has been established by the request of WA0CPS with WA0CPS-15 (node modified callsign) then Downlinking to KB0VA on port 1.

Uplink 2 (KA0DNV)

KA0DNV has Uplinked to KSLVN on port 2, but has not established a Crosslink or Downlink at this time.

FINE-TUNING YOUR K-Net NODE

Although the default parameters of the K-Net node will get you "up and running", there are some guidelines that can enhance the desired operation. Good neighbor relationships are really the key to good networks. This takes the form of proper quality settings for your neighbor nodes, as well as good relationships with your neighbor SYSOPS. If your node table has a consistent listing of destination nodes that have reliable paths, users will not be frustrated, and your node will be perceived as working quite well. On the other hand, if your node table is frequently allowed to accumulate a large number of destination nodes that are not fairly solid links, it becomes difficult to use, non-responsive, and very unfriendly to the network and especially to a new user.

After the K-Net node has been running for a while using the default parameters, some "tweaking" of commands are in order. <u>The Quality settings for your neighbor nodes will have the largest impact on the usefulness of your node</u>.

REMEMBER - THE QUALITY NUMBER DETERMINES HOW FAR AWAY A NODE WILL BE VISIBLE IN THE NETWORK AND THE ORDER OF AUTOMATIC ROUTE SELECTION.

The Routes table contains the information that will help you as you change the default settings. The Quality assigned to your neighbors listed in the Routes table can be changed to achieve the desired node set up. There are two very general philosophies that will help you determine how to "finalize" and maintain your node.

STATIC NODE

The default values for the K-Net node will cause every neighbor node that you hear to be included in both the route table and the node table (Quality = 70/110 and MINQual = 70). Initially, this may be a bit of a problem since weak neighbor nodes and their distant nodes may be saved in your node table. Try to connect to all the neighbor nodes that are listed in the routes table. Be sure that you have set the number of Netroutes such that all neighbors will be included. Make note of which nodes respond with a good signal and modify the route table with the ADDRoute command to set and lock the quality of these nodes to a number higher than either value automatically assigned by the Quality command for either port. For example, if the Quality command is set to 70/110, and all good routes are locked at a Quality of 115, setting MINQual to 115 results in a static node that is only aware of the neighbor nodes that you have "locked in". This node will NEVER have any new neighbor nodes OR distant nodes automatically added to the node table. If distant nodes are to be listed in the node table, they must be manually added to the node table using the Addnode command, listing one of the existing neighbors as the route. Since distant nodes may have multiple paths from your node, the preferred route and alternate route(s) to a distant node can be specified with multiple Addnode commands. When connections to these distant nodes are requested, your node will ALWAYS use the locked route path(s) in spite of changing band conditions.

Part of the responsibility of being a node SYSOP is maintenance. It's probably a good idea to set MINQual equal to or less than Quality from time to time in order to see if any new neighbor nodes appear in your route table. If a good path to a new neighbor shows up, lock the new node route with your "good" neighbor Quality and then reset MINQual to it's previous value.

DYNAMIC NODE

The default values for Quality and MINQuality will allow ALL neighbor nodes that are heard to be listed in the node table. This means that both good and marginal neighbors will be listed in the node table. Of course, the node table will be larger than with the static node settings, but the benefit obtained is that the node will know when new neighbors appear and then automatically include them in the node table. Although distant nodes that may have good routes from your node will not be listed, some SYSOPS feel that this is one method to control the propagation of marginal nodes. Another effect of dynamic node settings is that users can take advantage of band openings that could allow the use of nodes not normally visible on the network. This can lead to some frantic DXing when band conditions permit. The dynamic approach allows the node table to be elastic. It will grow as previously unknown neighbors appear, and shrink as the obsolescence counters for each route decrement.

After the K-Net node has been in operation for a while, and you have noted your "good" and "not-so-good" neighbors, a combination of static and dynamic settings can be used. By locking in "good" neighbors at high quality values, you can force the routing logic to take a preferred route in spite of changing band conditions. As long as MINQual is equal to or less than Quality, new neighbor nodes

and nodes heard by virtue of a band opening will also come and go as the obsolescence counter is either reset or decremented. In all probability, your neighbor nodes will have "good" nodes that your node should know about, and your neighbor's neighbor will also have "good" nodes, as well as your neighbor's neighbor's neighbor, and so on. Once "good" neighbors have been established, the Quality to assign to each neighbor will have to be determined experimentally in order to control the listing of marginal destination nodes that will appear as a result of receiving neighbor node broadcasts.

As a practical point, once a distant node is 3 or 4 RF hops away from your node, having it listed in your node table is not a good practice unless the path is extremely reliable. The chances of a user being able to access a node this distant diminishes greatly as the number of RF hops increases beyond this point. Also, allowing destination nodes to appear in your node table that are more than 3 or 4 hops away will result in VERY large node tables being generated. When your node receives a nodes broadcast from a neighbor node, K-Net will automatically calculate a quality to all the nodes contained in that broadcast. If the computed quality is less than MINQual, that distant node is not added to your node table. Therefore, it is desirable to have a combination of Route quality and MINQual that will only allow nodes that are 3 or 4 node hops away to appear in your node table.

As an example, if your "good" neighbor node (A) has a "good" neighbor node (B), and node (B) has a "good" neighbor node (C), your node has a good chance of reliably reaching node (C). The computed quality at your node for distant node (C) must be greater than MINQual in order for it to be listed in your node table. If a quality of 70 is assigned to all neighbors in this example, your node would assign a quality of 5 for distant node (C), computed as follows (it helps to draw this out on paper):
The quality from (B) to (C) is 70, when (B) broadcasts (C)'s quality as 70, (A) multiplies 70 by 70 and divides by 256 to arrive at a quality of 19 for (C). When your neighbor (A) does a node broadcast, his broadcasted quality of 19 for (C) is multiplied by your path quality for (A) and divided by 256, resulting in a quality of 5 for distant node (C) (19 times 70 divided by 256). Since 5 is probably below your MINQual, distant node (C) will not be listed in your node table even though it is probably useable. What to do? You could change the Quality to your neighbor (A) to 255. This will result with your node assigning a quality of 18 for distant node (C), but it does not leave much room for a special case route that may be 5 or 6 hops away. This will also cause your neighbor (A) to propagate further than desired in the network; creating useless routing. As a compromise, lock the quality of (A) to 140. If your neighbor node (A) would also lock in a quality of 140 for his neighbor (B) and node (B) would do the same for node (C), then your node would now assign a quality of 41 for distant node (C), leaving some "wiggle room" for other nodes as they join the network. Now a MINQual setting of 41 will allow these "good" distant nodes to be listed in your node table.

If you work through the above exercise a few times using different quality values, it becomes apparent that there are many settings of quality that will produce the desired effect as long as all SYSOPS adhere to a common guideline. However, the quality values used in the above example were not just "picked out of the air":

140 is a <u>reasonable</u> quality to assign to a good 1200 baud neighbor, since this allows good distant nodes to appear in your node table while also limiting node propagation. It also leaves room in the quality range (0 - 255) to assign higher qualities to preferred 9600 baud neighbors.

Generally speaking, 9600 baud routing should be used whenever possible since this has the effect of "unloading" the Local Area Network. Locking in a quality of 200 to "good: 9600 baud neighbors will have the same general benefits of the 140 setting on the 1200 baud port - reliable paths to good destination nodes, limiting of node propagation, and still leaving room in the quality range for ultra-reliable routes and even higher node speed preferences (remember - the KPC-9612 can operate up to 19,200 baud).

This is where good neighbor relationships with other SYSOPS in your area become important. Neighbor node quality must be continually examined in order to arrive at a useable network configuration that makes good use of all possible paths. Coordination of efforts and philosophies should result in an expanding network that is low in frustration and fun to use.

NODE STACKING

Although the K-Net firmware in the KPC-9612 provides a quick and easy method to increase network usage and efficiency, as the network grows there will be an increasing demand to provide an even higher level of service to the community. This can take several

forms: Adding more user ports to the network as existing user ports become heavily loaded, adding dedicated server access ports to "unload" activity from existing user/server shared channels, and the creation of point-to-point dedicated backbone channels to increase overall network performance.

Since knowledge of how to properly configure the K-Net node is assumed, this section will deal only with the physical connections and parameters that are required to create a node stack.

Before deciding to "help" the community, you should be aware of the impact that adding more ports of any nature will have on the existing network. Generally speaking, more available radio ports will provide a more efficient network PROVIDED THAT this effort is well co-ordinated with the other node SYSOPS in the area that will be affected by your actions. SYSOP coordination and cooperation must exist in order for the network to provide the maximum benefit for all users.

The concept of a node stack is practically self-explanatory. A node stack is formed when two or three or more network nodes are "hard-wired" together (stacked) and allowed to communicate with each other. Since the radio ports are already in use, the serial port of the TNC is used to pass the networking information and data between the nodes in the stack. Setting the INTFACE command (from the cmd: prompt) to NET forces the K-Net node to use the serial port as well as the radio ports for passing network information. Because of this, the serial port can no longer be used as an ASCII terminal port since all communication is now "net-ese" and used by the other nodes in the stack for routing and other network functions.

If you need to change a node parameter, the terminal mode can be regained by connecting a terminal to the serial port and using the standard packet transparency escape sequence (three control C method), or the node can be remotely accessed via a radio port if the RTEXT has been set, and the proper response to the SYSOP command is sent to the K-Net node.

Again, the Quality parameter plays a major role when node stacking is used. The default Quality settings are 255/70/110 (255 for port 0 - serial port, 70 for port 1 - 1200 baud, and 110 for port 2 - 9600 baud). The Quality setting for the serial port (port 0) is defaulted to the maximum value allowable because the serial link between nodes in a stack is very reliable. Of course this value can be changed, but doing so may cause undesirable node propogation and routing.

The most basic form of a node stack would be to add another 1200 baud port to your KPC-9612 node. The additional hardware required is another complete 1200 baud network node station (transceiver, feedline, antenna, TNC to radio cable, TNC with network node firmware such as a KPC-3 with version X.XX N firmware or other TNC with NET/ROM or TheNET firmware), and a special serial cable to connect the KPC-9612 to the other TNC.

Construction of the serial cable is straightforward. Using two male DB-25 connectors, connect pin 1 to pin 1, pin 2 to pin 3, pin 3 to pin 2, and pin 7 to pin 7.

KPC-9612				KPC-3 (Ver. 6.0N)
Frame ground	1	----------------------------	1	**Frame ground**
Transmit data	2	----------------------------	3	**Receive data**
Receive data	3	----------------------------	2	**Transmit data**
Signal ground	7	----------------------------	7	**Signal ground**

With the above cable assembled, the KPC-9612 (K-Net installed) and the KPC-3 (K-Net installed) are free to exchange network data over their respective serial ports.

Operation is probably very close to what you would imagine it to be. Each TNC must be given a unique **NETALIAS** and **NETCALL** because they are separate nodes.

They should also be configured with the node parameters that are recommended in your area. The serial cable functions as a "virtual radio circuit" for node-to-node communication in the stack. *REMEMBER*! THE SERIAL PORTS ARE TALKING TO EACH OTHER - *THE SERIAL PORT MODE PARAMETER MUST BE SET THE SAME IN BOTH NODES* (e.g....9600, N, 8, 1).

The final step in order to activate this 3 port stack is to set the INTFACE command in each of the TNCs to NET and to power them off and then back on.

FOUR PORTS
(TWO HIGH-SPEED AND TWO 1200 BAUD PORTS)

A four port node stack consisting of two high-speed ports and two 1200 baud ports can be created by using two KPC-9612 modems with the K-Net PROM installed. A pair of KPC-9612s with the K-Net can be utilized to great benefit. Since the high-speed port of the KPC-9612 is capable of 4800, 9600 or 19200 baud operation, this set-up offers tremendous flexibility in network configuration. As above, each of the KPC-9612 K-Net nodes must be programmed with a different NETALIAS and NETCALL and use the serial port cable described above to tie them together. Now set the INTFACE command in each KPC-9612 to NET.
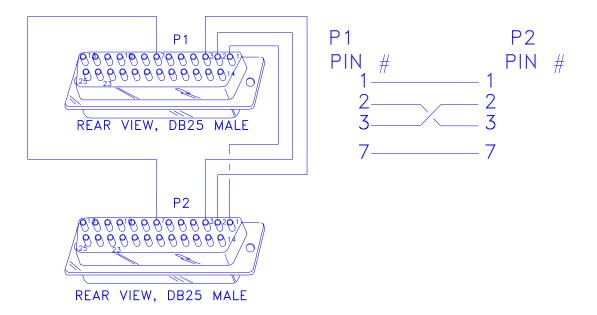*Power them OFF and back ON to complete the installation.*
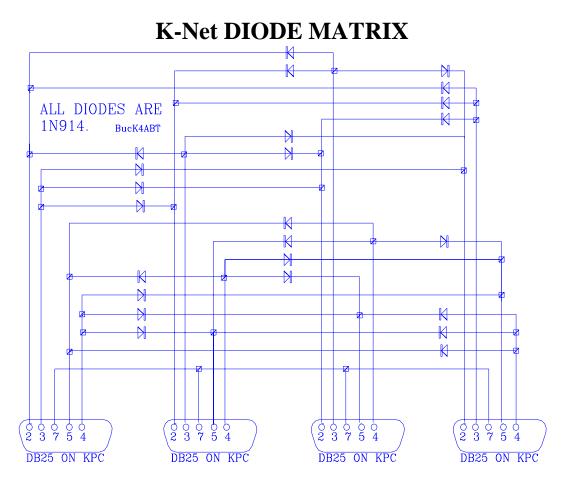
MORE THAN FOUR PORTS

The three and four port stacks discussed above are relatively simple to cable together since there is no contention for the serial port. Each node is free to broadcast and pass data over the serial port "radio". Adding another node to the stack means that now there needs to be a method to control access to the common serial port cable in order to avoid data collisions. The K-Net node uses the RTS (Request To Send - pin 4) and CTS (Clear To Send - pin 5) control lines of the serial port for this purpose.

When a K-Net node has data to send to the other nodes in the stack, it first checks the RTS line for a true (logic high) state. If RTS is true, then the node drops CTS and the data is sent to the other nodes in the stack. The dropping of CTS by any node in the stack will inhibit all the other nodes from using the serial port by forcing RTS false (logic low), thus avoiding data collisions within the node stack. The additional hardware required for this operation is commonly refered to as a Diode-Matrix.

KNET TWO PORT INTERCONNECT
NOTE the lines between pins 2 & 3 are rolled!

REAR VIEW, DB25 MALE

REAR VIEW, DB25 MALE

P1 PIN #		P2 PIN #
1	—	1
2	⤬	2
3	⤬	3
7	—	7

The configuration shown here will enable two KPC/KNET nodes to gateway between two or more frequencies or baud rates. If more baud rates or frequencies are needed for gateway connectivity, the diode matrix shown below may be utilized (Drawings by BucK4ABT). For more info, visit: **www.SEDAN.org**

K-Net DIODE MATRIX

ALL DIODES ARE 1N914.   BucK4ABT

DB25 ON KPC   DB25 ON KPC   DB25 ON KPC   DB25 ON KPC

A diode matrix is used to "stack" more than two KPC KNET nodes. A five port Diode Matrix follows the same logic as above, with six ports probably being about the limit of practicality. The number of diodes required for a Diode Matrix with *N* serial ports is 2*N*(N-1) (Drawings by BucK4ABT).

TheNET X1-J / K-Net CROSS-REFERENCE GUIDE

This cross-reference guide is included to assist in the coordination of node parameters if an X1-J node is located in your area. The basic X1-J PARMS can be listed by connecting to an X1-J node and giving the P command.

This should precipitate meaningful discussion among node SYSOPS in an attempt to arrive at a "Network Standard" of node parameters that benefit all users.

X1-J4 PARM   K-Net COMMAND

1.	Max. Destination Node Size	NETDest
2.	Min. Auto update quality	MINqual
3.	Neighbor default quality	Quality
4.	RS-232 default quality	N/A
5.	Initial obsolescence count	Obsinit
6.	Min. Obs. count to broadcast	OBSMin
7.	Node broadcast interval	NODESInt
8.	Initial Time-to-Live	L3ttl
9.	Transport Frack timeout (sec)	L4T1
10.	Transport Retry counter	L4N2
11.	Transport Ack Delay (sec)	L4delay
12.	Transport Busy Delay (sec)	N/A
13.	Transport Window Size (frames)	L4Window
14.	Transport Overfill Limit (frames)	N/A
15.	No Activity Time Out	L4Limit
16.	Persistence	cmd: PERsist
17.	Slot	cmd: Slottime
18.	Link Frack (T1)	cmd: FRrack
19.	AX.25 Maxframe	cmd: MAXframe
20.	AX.25 Retries	cmd: RETry
21.	Link Response Time (T2)	N/A
22.	Active Check (T3)	cmd: CHeck
23.	AX.25 Digipeat	cmd: DIGipeat
24.	Validate Callsigns	N/A
25.	ID Beacon	cmd: IDINT
26.	CQ Broadcasts	CQBc

ACL (Access Control List)   cmd: SUPCalls/LLIst
To COMPLETELY ignore uplinks from AND downlinks to specific user callsigns and neighbor node <u>callsigns</u> (not alias)

cmd: BUDCalls/CONList
To allow uplinks from and downlinks to ONLY specific user callsigns and node <u>callsigns</u> (not alias)

Having Fun KNET'n on the SEDAN

De **Buck Rogers K4ABT**

For more KNET information; See the "***PACKET USERS NOTEBOOK***" in *CQ* Magazine Page 100 August 95 and *CQ* Magazine Page 64 October 95.