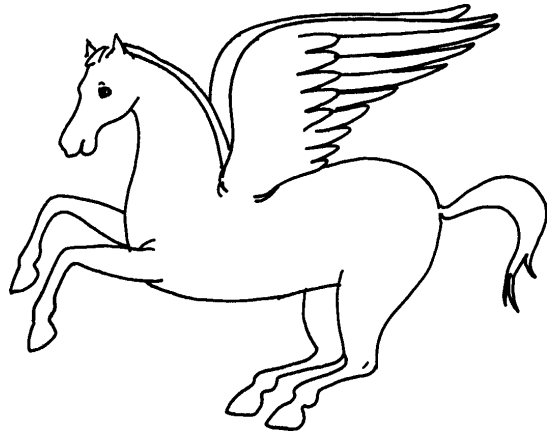
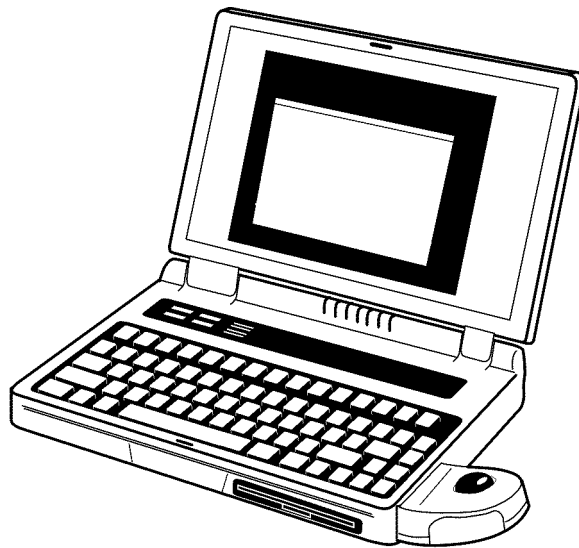
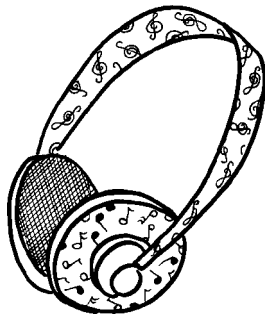


PEGASUS



Programmer's Reference Guide



Contents

<u>Interface Settings</u>	3
<u>The PC to Radio Connection</u>	4
<u>Hardware Vs Software</u>	4
<u>System Overview</u>	5
<u>Power-Up and Restart Sequence</u>	6
<u>Transmitter Safety Features</u>	7
<u>Signal Strength & Power Output Reporting</u>	7
<u>The Pegasus Command Set</u>	9
<u>Radio Command Set</u>	10
<u>Tuning the Radio (general)</u>	11
<u>Operation with Optional 302 Remote Encoder</u>	26

Conventions used in this manual

Numeric Types:

0x0A, 0Ah	Hexadecimal Numbers.
10	Decimal Number.
'A'	ASCII character code. example: ASCII 'A' is 0x41.

Work is underway to create DLL's, VBX's and Active-X tools to support the Pegasus transceiver. This will provide for drop-in Pegasus support to any 32 bit program.

If you would like to participate in the development and testing of these components you may contact Ten-Tec via email at ditsnbits@tentec.com.



Introduction

The Ten-Tec Pegasus DSP HF transceiver is part of a new breed of PC controlled communications equipment that is more software than hardware. The Pegasus contains state of the art RF and Digital Signal Processing (DSP) circuitry which provides all transceiver functions. Unlike conventional transceivers the Pegasus contains no front panel controls. The hardware relies entirely on an external controller to provide the user-friendly, radio-like functionality common in modern communications equipment.

Ten-Tec provides a Windows based Graphical User Interface or GUI (pronounced “gooey”) which allows the Pegasus to be operated from a PC running Windows 3.1, 95 or 98. The radio-like features are created in the PC rather than the radio. Ten-Tec encourages and supports the efforts of other software vendors in adding support for the Pegasus to their programs. You may want to check with your favorite software provider to see if support is already available for the Pegasus. Skilled users may wish to create their own control programs or have a unique application not supported by commercially available programs. In such cases it will be necessary for the developer to communicate with the Pegasus attached to the PC. Ten-Tec has produced this document as a starting point for software developers undertaking the PC-to-Pegasus interface.

Interface Settings

The RS-232 serial interface on the Pegasus is controlled by a 16C550 UART located on the DSP board. The interface parameters are fixed at 57,600 baud, No Parity, 8 Data bits, 1 Stop bit. The UART uses hardware handshaking to control the data flow between the PC and the radio. The host PC should be set to use RTS/CTS signalling.



The PC to Radio Connection

Interfacing an amateur radio transceiver to a personal computer (PC) is not a new idea. Manufacturers have created a variety of hardware & software approaches to support interfacing the RF and DIGITAL worlds. Those knowledgeable in transceiver interfaces will notice that the Pegasus is considerably different from the rest. The Pegasus interface is similar to the Ten-Tec RX-320 interface and contains a superset of our RX-320 command set. The Pegasus and RX-320 command sets are optimized for speed and kept simple for flexibility. Also, the interface takes advantage of the processing power of the host PC. Complex math operations are performed by the PC to lighten the processing load on the radio. This allows more time for signal processing features such as noise reduction and automatic notch.

Hardware Vs Software

The heart of the Pegasus is the DSP processor. It executes instructions contained in firmware much like a typical microprocessor. The DSP processor is designed specifically for high speed signal processing tasks. The DSP processor in the Pegasus handles more than just signal processing. So much of the radio is in software that it may be difficult to understand where software ends and hardware begins. However, for developing Pegasus applications, an understanding of the relationship between hardware and software can be important.

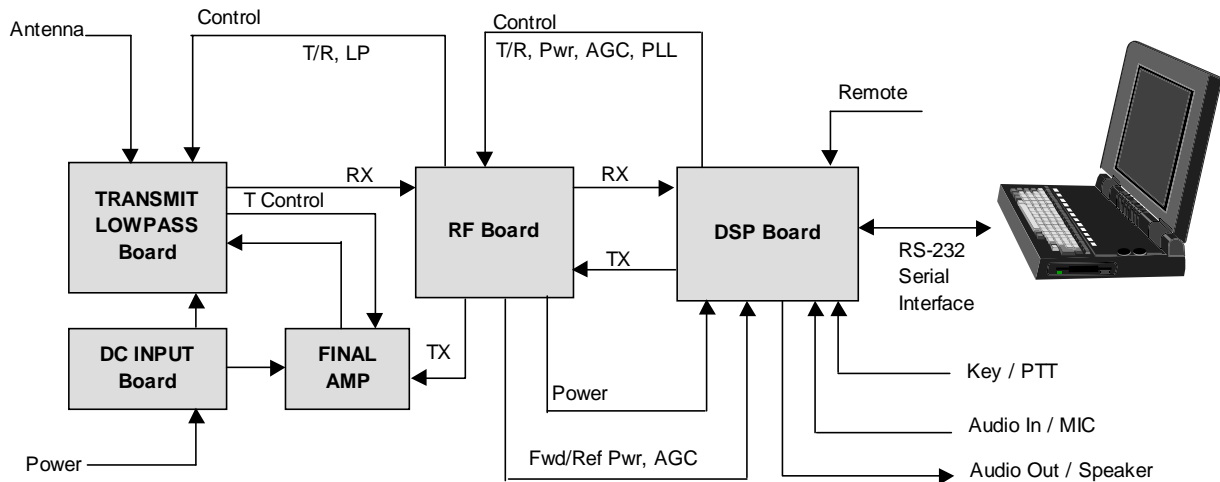
The hardware of the Pegasus consists of 5 system boards, the RF board, DSP board, Transmit Lowpass assembly, the 100 watt final amplifier and the DC input board. The receive/transmit analog functions are handled by circuitry located on the RF board. The RF board contains fixed and tunable mixing stages and provides the initial transceiver selectivity by passing signals through intermediate narrow band stages. In Receive signals pass from the RF board to the DSP board to be converted to audio. In Transmit the DSP output is passed to the RF section to ultimately arrive at the final amplifier.

In addition to signal processing, the DSP board has the responsibility of managing the communications between the radio and the PC. The 9-pin serial port on the Pegasus handles bidirectional RS-232 level communications between the radio and PC. The communication parameters are fixed at 57,600 baud, 8 data bits and no parity. A 16C550 UART manages the data transfer activity between the PC and the DSP processor. The interface uses CTS and RTS handshaking for data flow control to ensure data integrity.

For many functions the RF and DSP boards have to work together. For example, tuning the transceiver to a particular frequency is managed through the proper setting of both the RF board's PLL (Phase Locked Loop) and the DSP processor's internal registers. The DSP board has complete control over the RF board's PLL settings but the program running on the PC must pre-calculate the factors necessary for tuning.

In addition, the DSP interfaces to the Transmit Lowpass board to select the appropriate filter as determined by the transmit frequency. Measurement and control of RF GAIN, AGC, POWER and other RF functions are also managed by the DSP board.





(Figure 1) Pegasus Block Diagram

System Overview

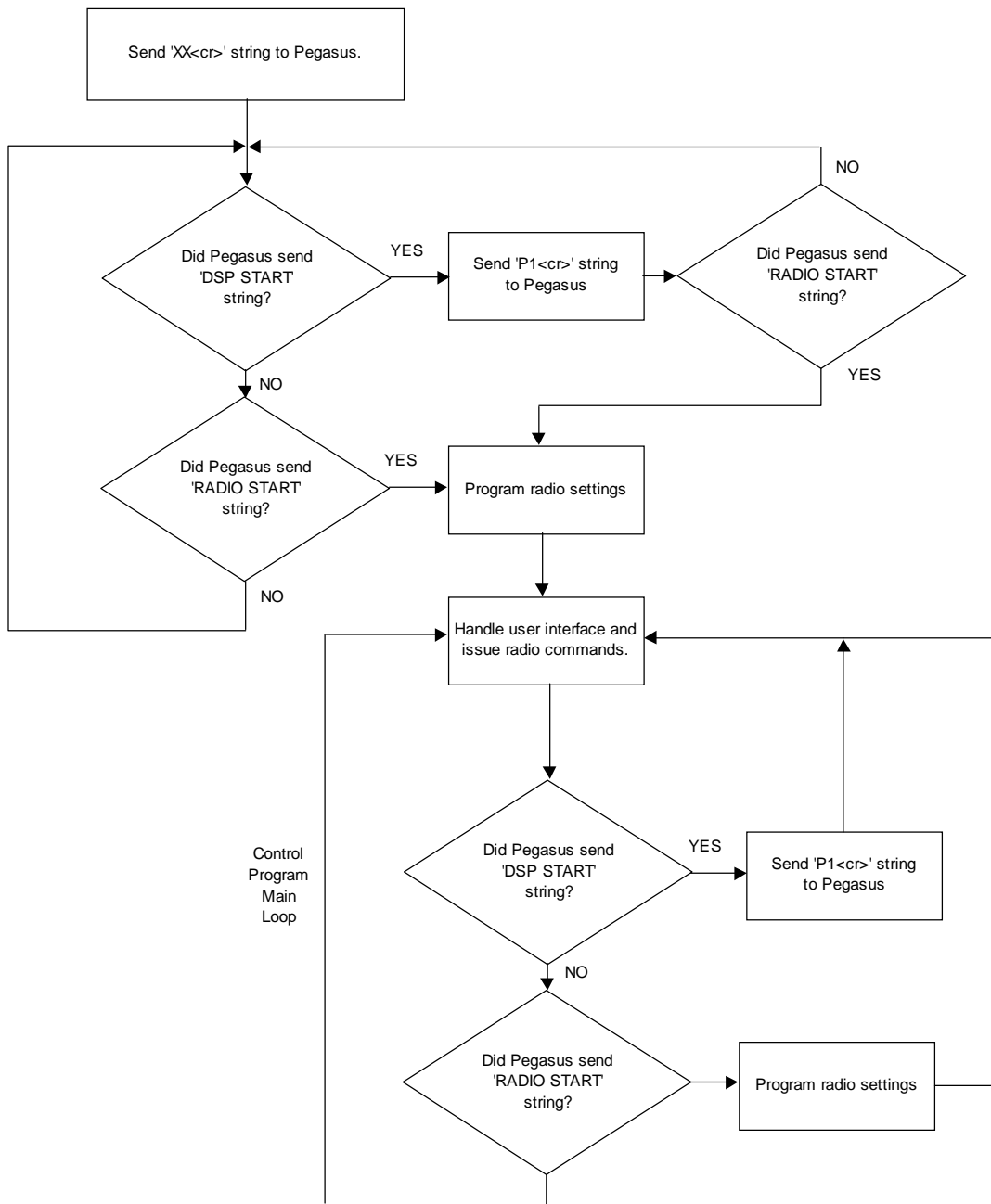
The Pegasus is both a computer and a radio. Strictly speaking the Pegasus is a signal processing computer attached to RF hardware. The signal processing computer executes instructions contained in a ROM (Read Only Memory) chip mounted in a socket on the DSP board. Standard ROMs are programmed using special equipment. The ROM chip used in the Pegasus is a FLASH ROM which can be reprogrammed while still in the board. Using special software provided by Ten-Tec, the ROM inside the Pegasus can be updated through the same serial port used to control the radio. Your Pegasus can be updated in seconds, allowing you to take advantage of the latest enhancements. Also, the Pegasus has the ability to hold multiple DSP programs or radio personalities. Future software offerings will take advantage of this feature to greatly expand the Pegasus.

To provide this feature, the Pegasus firmware contains a special operating mode called SYSTEM/MONITOR mode. The Pegasus starts-up in SYSTEM/MONITOR mode when power is first applied. The flashing RX L.E.D. on the front panel is an indicator that the unit is in SYSTEM/MONITOR mode. The S/M mode provides a limited number of features but a great amount of utility. In this mode it is possible to reprogram and verify the FLASH ROM, add additional programs and instruct the DSP which program to run. To maintain future compatibility, the radio firmware provided by Ten-Tec will always be the first program in memory. The 'P' command is used to execute a program. So to execute the Ten-Tec radio program inside the Pegasus use the command 'P1' (without quotes) followed by a carriage return. Since the Pegasus always powers up in S/M mode the controlling program will need to tell the Pegasus which DSP program to run. So the controlling program will always issue a 'P' command as part of the radio initialization.



Power-Up and Restart Sequence

When power is applied or when the power switch is set to ON the radio will transmit the string ‘ DSP START <cr>’ over its serial interface. A controlling program can watch for this transmission and will know that the PEGASUS will need to be reprogrammed. Initially, the Pegasus enters SYSTEM/MONITOR mode (see system overview). A command must be sent to the Pegasus to begin executing transceiver firmware. When Pegasus begins executing transceiver operations a ‘RADIO START <cr>’ string will be sent to the PC over the serial interface. The audio will be muted and all parameters need to be programmed before the radio will operate properly. This involves setting the tuning factors, selecting a filter, setting the audio levels, etc.



(Figure 2) Suggested Pegasus to PC Interface Start-up Sequence



Transmitter Safety Features

Because the PC is such a critical component of the Pegasus system the radio firmware contains a safety feature which will disable the transmitter in the event of a PC-to-Pegasus communications failure. This KEEP ALIVE feature operates by resetting a time-out timer each time a serial command is received by the Pegasus. In order to keep the transceiver operating it is necessary to talk to the Pegasus at least once every 2 seconds when using the default time-out settings. The time-out period can be set from less than 1 second to greater than 6.5 seconds through software control. This KEEP ALIVE feature can also be disabled through software control which can be useful during software development activities. It should never be disabled during regular operation unless this safety feature is provided by other means. In a typical operation the PC would be communicating regularly with the Pegasus gathering Signal Strength and Power Output information anyway. This would be sufficient to keep the transceiver operating. The KEEP ALIVE feature and its associated Time-Out has no affect on receiver operations. Should a Time-Out occur while transmitting the Pegasus will terminate transmit operations and return to normal receive operation.

Another useful feature is the ability to disable the transmitter path via software command. The radio will behave normally but the transmitter stages will never be activated. This can keep the Pegasus off-the-air during software development and testing. Other than the lack of transmitter output the Pegasus will be fully functional in this mode. The KEEP ALIVE feature will also be functional in this mode unless disabled.

Signal Strength & Power Output Reporting

The Pegasus can return Signal Strength information upon request. It can also provide Forward and Reflected output power levels upon request. To simplify control software the radio will automatically provide the appropriate information depending upon whether the radio is transmitting or receiving. This is tied to the Signal Strength request command. Should the radio be receiving at the time of the Signal Strength request the Signal Strength will be reported. Should the radio be transmitting at the time the radio will respond with Forward and Reflected power levels. This also provides an indication to the controlling program that the radio has entered transmit mode.



This page intentionally left blank.



The Pegasus Command Set

The Pegasus command set is a superset of the command set developed for the Ten-Tec RX-320 receiver. The command set is extensive, as would be expected on a PC dependent transceiver. However, every effort has been made to keep the individual commands as simple as possible. Although the Pegasus interface operates at 57,600 baud short commands are used to further reduce processing overhead for the radio and the PC.

In a conventional receiver the operator controls the radio by way of front panel controls. The user only has to turn a knob or push a button. Usually a microprocessor hidden inside the radio will react to the user's motions and then manipulate the radio hardware to get a predefined result. The Pegasus lacks a front panel and the DSP processor has very little predefined functionality. It is up to the controlling program to provide the operator's knobs & buttons and manipulate the DSP to create the functionality. In this arrangement the controlling program is responsible for preprocessing the user's request and determining the necessary DSP settings to achieve the end result. Many DSP functions are interrelated and simply tuning the receiver may require adjustment of all DSP settings.

In general a PEGASUS instruction is a single command letter which may be followed by data and then terminated by a carriage return <cr>. Unless otherwise noted the data is in binary format. Using a binary format requires fewer bytes when compared to an ASCII format. By comparison, some radio interfaces use a BCD (Binary Coded Decimal) format. BCD format is quite useful for equipment with a front panel because many display devices will require BCD. The Pegasus operates with binary data and unless otherwise noted all commands that require numeric data will need that data to be in binary format. In binary format a number like 25 decimal is represented as 19 hexadecimal or 19h or sometimes written as 0x19. All refer to the same number.

The DSP provides control over TRANSMIT and RECEIVE MODE, FREQUENCY, FILTER, POWER as well as AGC MODE, SPEAKER LEVEL and many other features. In addition, the DSP can respond to requests for SIGNAL STRENGTH, DSP FIRMWARE REVISION, etc. The Pegasus contains no provision for storing any parameters so the radio must be reprogrammed whenever the radio is turned on and/or when the controlling program is started. Reprogramming the radio at power-up requires setting all appropriate parameters. To prevent unwanted audio output the VOLUME setting should be done last.

In general, the Pegasus will accept any provided data as being valid. It is up to the programmer to ensure that the supplied data is correct. Where a command has a limited number of data options, such as the AGC MODE command, failure to provide a valid selection will result in the radio choosing a default setting. There will be no notification that the data is invalid. Where a command is totally unrecognized the radio will send back a response consisting of a single letter 'Z' followed by a carriage return <cr>. Some commands may have data fields that may contain the binary number 0x0d which is also the carriage return character. The firmware in the Pegasus that is responsible for parsing the command strings has the intelligence to ignore carriage returns (0x0d) embedded in the data field.

Because the Pegasus was built to be reprogrammed in-system the command set presented here is subject to change or enhancement. We will make every effort to make the system backward compatible with existing documented commands whenever possible. However, the Pegasus is an HF RADIO PLATFORM that could host a variety of radio services. Persons or companies developing control software for the Pegasus should not assume that a Pegasus is operating original factory firmware but rather should always query the radio about operating modes (SYSTEM/MONITOR, RADIO, etc.) and firmware revisions to ensure compatibility.



Radio Command Set

Restart and Notify

This command will cause the radio to do a software restart. Upon start-up the radio will issue a start message appropriate for the currently running operation. For example, if the radio was operating in SYSTEM/MONITOR mode the radio will respond with “ DSP START”. It will respond with “RADIO START” if the radio was operating in RADIO mode.

Under normal circumstances, the controlling program on the host PC would issue this command when starting to determine which mode the Pegasus may be in. Then the program could take appropriate action. For example, if the Pegasus was in SYSTEM/MONITOR mode and the controlling program wanted RADIO mode it would need to issue the appropriate commands to switch modes (the ‘P’ command, see Dsp Program Execute Command).

WARNING:

SYSTEM/MONITOR MODE IS PROVIDED TO SUPPORT IN-SYSTEM FIRMWARE UPDATES AND DIAGNOSTICS. EXERCISE EXTREME CAUTION WHILE THE RADIO IS IN SYSTEM/MONITOR MODE. UNDER NORMAL CIRCUMSTANCES THE ONLY COMMAND A CONTROLLING PROGRAM WOULD ISSUE IN SYSTEM/MONITOR MODE IS THE COMMAND PLACING THE PEGASUS IN RADIO MODE

format: ‘XX’ <cr>
where: ‘X’= is the ASCII ‘X’ (0x58) character
<cr> = ASCII carriage return (0x0D)
response: “ DSP START” or “ RADIO START”

Dsp Program Execute

The Pegasus firmware and hardware have been developed to support storage and execution of multiple DSP programs. In this way a Pegasus can take on a different personality for a particular, as yet undetermined, task. The initial release of the Pegasus firmware contains only a single DSP program which provides the transceiver behavior. Future firmware could provide more advanced or customized DSP features. The PC control program would determine which program the Pegasus should execute.

format: ‘P’n<cr>
where: ‘P’= is the ASCII ‘P’ (0x50) character
n = program to execute...
0 = SYSTEM/MONITOR program
1 = Radio Program.
2 thru 8 = future assignment.
<cr> = ASCII carriage return (0x0D)
response: “ DSP START” or “ RADIO START” for program 0 or 1.
Future programs would likely generate different responses.



Tuning the Radio (general)

The PEGASUS does not have the ability to accept a frequency command directly. The radio can only accept tuning factors, which are the result of calculations made by the PC. There are 3 factors involved, the coarse-tuning factor, the fine-tuning factor and the BFO factor. The need for the different factors arises from the fact that the RF board provides a portion of the tuning while the remainder is provided by the fine-tuning and BFO mixers inside the DSP processor.

Figure 3 shows a diagram of the PEGASUS's mixing scheme. The dotted vertical line shows the break between Analog (RF board) and DSP processes. Table 1 gives some example frequencies using the mixing scheme of figure 3. As indicated on the diagram, the first local oscillator, LO1, is tunable in 2.5 kHz steps. In General, ignoring correction factors, the coarse tuning factor results from the need to tune LO1 as close as possible to the desired frequency. The fine-tuning factor represents the difference between the desired frequency and the frequency resulting from where LO1 will be tuned.

For example, given a desired frequency of 12.001 MHz and a 2.5 kHz tuning step we could tune to 12.000 MHz. The remaining 0.001 MHz would be passed to the fine-tuning mixer and LO3. The equations shown on page 12 show dividing the frequency given in MHz by 0.0025 (2500 Hz). The integer portion of the result is the basis of the coarse tuning factor while the fractional portion is the basis of the fine-tuning factor. Given the example above the $12.001/0.0025 = 4800.4$. The 4800 applies to the coarse tuning factor and 0.4 applies to the fine-tuning factor. Again, LO1 provides the coarse tuning in 2.5 kHz chunks and the DSP provides the fine-tuning. LO4, also referred to as the BFO, supplies an additional mixer that is used to provide additional post-filtering frequency translation in some detection modes.

The discussion and examples presented thus far have ignored a variety of correction factors that are needed to make the whole thing work. For example, LO1 tunes from 45 – 75 MHz, which is 45 MHz away from the desired frequency. This fact is taken into account by the 18000 ($45.00/0.0025$) which is added to the integer result of the basic calculation. Other adjustments arise from design constraints while still others arise from the selection of detection mode or filter bandwidth. See the Tuning Commands on pages 12 thru 15.



The command documentation applies only to the Pegasus running factory program 1 ('P1' command).

Setting the Receive Tuning Factors

Because the FILTER, MODE and BFO selections all affect the tuning factors they should be adjusted before calculating the new tuning factors. Any change in these parameters will require recalculation of the tuning factors. The command format for programming the PEGASUS's tuning factors is shown below. The controlling program must pre-calculate and format the tuning factors.

format: 'N' Ch Cl Fh Fl Bh Bl <cr>

where: 'N' = is the ASCII 'N' (0x4E) character.
Ch = high byte of the 16 bit coarse tuning factor.
Cl = low byte of the 16 bit coarse tuning factor.
Fh = high byte of the 16 bit fine tuning factor.
Fl = low byte of the 16 bit fine tuning factor.
Bh = high byte of the 16 bit BFO factor.
Bl = low byte of the 16 bit BFO factor.
<cr> = ASCII carriage return (0x0D) character.

response: none.

The tuning factors Ch, Cl, Fh, Fl, Bh and Bl are calculated using the formulas presented here.
input:

Tfreq = Tuned Frequency in MHz.

Mcor = Mode Correction = 0 for AM mode, +1 for USB, -1 for LSB, -1 for CW, 0 for FM.

Fcor = Filter Correction calculated using (Bandwidth/2)+200 in Hz.

Cbfo = Desired center frequency of filter in CW mode (in Hz). Only needed in CW mode.

calculation:

AdjTfreq = Adjusted Tuned Frequency = $Tfreq - 0.00125 + (Mcor * (Fcor + Cbfo) / 1000000)$

Ctf = Coarse tuning factor = $(int) (AdjTfreq / 0.0025) + 18000$

where **(int)** is used to get the integer only portion of the division.

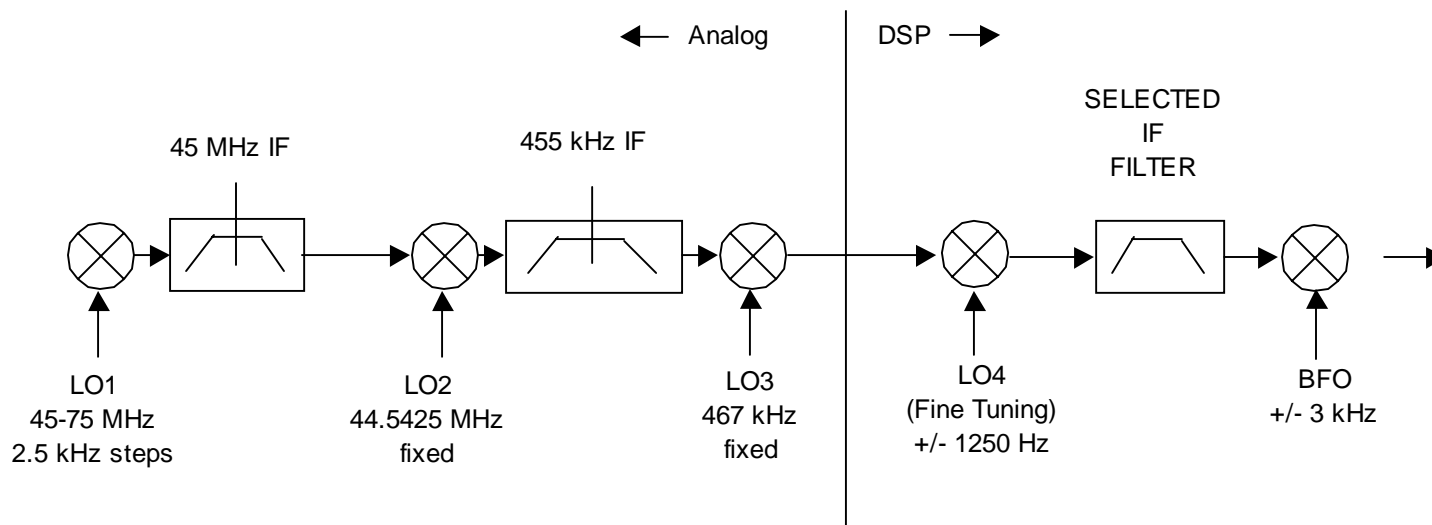
Ftf = Fine tuning factor = $mod(AdjTfreq, 0.0025) * 2500 * 5.46$

where **mod** is used to get the fractional remainder of a division operation.

Btf = Bfo Tuning Factor = $(int) ((fcor + CWBFO + 8000) * 2.73)$

where **(int)** is used to get the integer only portion of the division.





(Figure 3) Pegasus RX Mixing Diagram

Freq	LO1	NVAL	IF1	LO2	IF2	LO3	IF3	LO4
0.100000	45.09750	18039	44.9975	44.5425	0.455	0.467	0.012	0
0.100100	45.09750	18039	44.9975	44.5425	0.455	0.467	0.012	100
2.000000	46.99750	18799	44.9975	44.5425	0.455	0.467	0.012	0
2.005000	47.00250	18801	44.9975	44.5425	0.455	0.467	0.012	0
5.000000	49.99750	19999	44.9975	44.5425	0.455	0.467	0.012	0
10.001500	54.99750	21999	44.9975	44.5425	0.455	0.467	0.012	1500
11.000010	55.99750	22399	44.9975	44.5425	0.455	0.467	0.012	10
15.000000	59.99750	23999	44.9975	44.5425	0.455	0.467	0.012	0
30.000000	74.99750	29999	44.9975	44.5425	0.455	0.467	0.012	0

Mode	Freq Adj	Bfo	
AM	0	n/a	
USB	+filter specific	+filter Specific	
LSB	-filter specific	-filter Specific	LO4 mixer inverted in LSB
CW	-filter specific	-filter specific+CWBfo	LO4 mixer inverted in CW

(Table 1) Pegasus RX Tuning calculations



Setting the Transmit Tuning Factors

Because the FILTER, MODE and BFO selections all affect the tuning factors they should be adjusted before calculating the new tuning factors. The command format for programming the Pegasus Transmit tuning factors is shown below. The controlling program must pre-calculate and format the tuning factors. The details of the command are very similar to the Receive Tuning Factor calculations. However, the MODE and FREQUENCY refer to TRANSMIT MODE and TRANSMIT FREQUENCY.

format: 'T' Ch Cl Fh Fl Bh Bl <cr>

where: 'T' = is the ASCII 'T' (0x54) character
Ch = high byte the 16 bit coarse tuning factor
Cl = low byte of the 16 bit coarse tuning factor
Fh = high byte of the 16 bit fine tuning factor
Fl = low byte of the 16 bit fine tuning factor
Bh = high byte of the 16 bit BFO factor
Bl = low byte of the 16 bit BFP factor
<cr> = ASCII carriage return (0x0D)

response: none.

The tuning factors Ch, Cl, Fh, Fl, Bh and Bl are calculated using the formulas presented here.
input:

Tfreq = Tuned Frequency in MHz.

Mcor = Mode Correction = +1 for USB, -1 for LSB, -1 for CW, 0 for FM.

Fcor = Filter Correction calculated using (Bandwidth/2)+200 in Hz.

Cbfo = Desired center frequency of filter.

In CW mode this is the BFO setting.

In SSB Transmit this value is typically 1500 Hz

calculation:

AdjTfreq = Adjusted Tuned Frequency = $Tfreq - 0.00125 + ((Mcor * Fcor) / 1000000)$

where Fcor is determined as...

if $((FilterBw/2) + 200) > 1500$ then $Fcor = FilterBw/2 + 200$ else $Fcor = 1500$.

if (Mode=CW) $Fcor = 1500 + Cbfo$.

Ctf = Coarse tuning factor = $(int) (AdjTfreq / 0.0025) + 18000$

where (int) is the integer function to get the integer only portion of the division

Ftf = Fine tuning factor = $mod(AdjTfreq, 0.0025) * 2500 * 5.46$

where the mod function is used to get the fractional remainder of a division operation.

Btf = Bfo Tuning Factor = $(int) ((fcor + CWBFO + 8000) * 2.73)$

where (int) is the integer function to get the integer only portion of the division



Receive & Transmit Modes

The Pegasus supports LSB,USB,CW and FM modes in both transmit and receive and AM mode in receive only. The selected mode must be considered in calculations for the FREQUENCY tuning factors so any change in receive or transmit modes will require a recalculation of the frequency tuning factors.

- format:** 'M' Rx Tx <cr>
where: 'M' = the ASCII M (0x4d) character.
 Rx = receive mode (See Tx).
 Tx = receive mode.
 ASCII '0' (0x30) for AM mode
 ASCII '1' (0x31) for USB mode
 ASCII '2' (0x32) for LSB mode
 ASCII '3' (0x33) for CW mode
 ASCII '4' (0x34) for FM mode
 <cr> = ASCII carriage return (0x0D) character.
- response:** none.
- example:** M11<cr> for USB receive, USB transmit.
 M23<cr> for LSB receive, CW transmit.

Receive Filter

The Pegasus contains a large number of selectable filters that can be used in AM, LSB, USB and CW detection modes. There is no default filter selected and the appropriate filter should be set for the selected detection mode. The selected filter must be considered in calculations for the FREQUENCY tuning factors, so any change in filter selection will require a recalculation of the frequency tuning factors. In general the filter selection will also affect the BFO setting factor. The exception is AM mode, where the BFO setting has no meaning and is ignored.

- format:** 'W' fn <cr>
where: 'W' = the ASCII W (0x57) character.
 fn = a filter number in the range 0 thru 33 (binary). From Table 1 below.
 <cr> = ASCII carriage return (0x0D) character.
- response:** none.
- example:** W 0x10 <cr> for filter number 16, 2100 Hz bandwidth.

Filter #	Bandwidth
0	6000
1	5700
2	5400
3	5100
4	4800
5	4500
6	4200
7	3900
8	3600
9	3300
10	3000
11	2850

Filter #	Bandwidth
12	2700
13	2550
14	2400
15	2250
16	2100
17	1950
18	1800
19	1650
20	1500
21	1350
22	1200
23	1050

Filter #	Bandwidth
24	900
25	750
26	675
27	600
28	525
29	450
30	375
31	330
32	300
33	8000

(Table 3) Pegasus Receive Filters



Transmit Filter

The Pegasus contains a large number of filters that can be used in SSB transmit modes. The SSB transmit filters are taken from the same filter table as the receive filters. However, transmit filter selections should be limited to those shown below. There is no default filter so a filter should be selected before allowing the radio to transmit. The selected filter must be considered in calculations for the Transmit Frequency tuning factors so any change in filter selection will require a recalculation of the frequency tuning factors. Please see the Tuning Factor section for additional information.

- format:** 'C' fn <cr>
where: 'C' = the ASCII 'C' (0x43) character.
fn = a filter number in the range 7 thru 24 (binary). From Table below.
<cr> = ASCII carriage return (0x0D) character.
response: none.
example: C 0x10 0x0D for filter number 16

Filter #	Bandwidth
7	3900
8	3600
9	3300
10	3000
11	2850

Filter #	Bandwidth
12	2700
13	2550
14	2400
15	2250
16	2100
17	1950
18	1800
19	1650
20	1500
21	1350
22	1200
23	1050

Filter #	Bandwidth
24	900

(Table 4) Pegasus Transmit Filters



Line Level Audio Output Control

The Pegasus has two audio outputs and while they can be controlled independently each is affected by the other's settings. The Line Output Control sets the low level output out of the DSP processor. This output is fed to the Line Output and the input to the Speaker Amplifier. There is a separate control for setting the output level of the audio amplifier. Under normal circumstances it is not necessary to set the Line Out level. The default setting will set the DSP output to MAX so it can be ignored unless there is a reason to change it.

The Line Output Level command refers to a volume level but really represents an attenuation code that is passed to the DSP processor. The range is 0 thru 63 where 0 represents the loudest setting and 63 represents the lowest setting or greatest degree of attenuation. Each step is equal to 1.5 dB of attenuation for a total control range of 96 dB.

format: 'L' nn <cr>
where: L = 'L' (Ascii 0x4C) character
nn = volume attenuation number 0-63 (binary)
<cr>=ASCII carriage return (0x0D)
response: none
example: L 0x20 0x0D sets the Line Level to 32.
default: 0 (full output)

Speaker Output Control

The speaker volume control sets the gain of the audio amplifier. The control range is off (0x00) to full on (0xff). The input to the audio amplifier is set by the Line Level Audio Control. Refer to the Line Level Audio command for further information.

format: 'V' nn <cr>
where: V = 'V' (Ascii 0x56) character
nn = speaker volume 0 to 255 (0x00 - 0xff)
<cr>=ASCII carriage return (0x0D)
response: none
example: V 0x20 0x0D sets the speaker to level 32
default: 0 (no output)



AGC Mode Control

The PEGASUS can operate in SLOW, MEDIUM or FAST AGC mode. At power-up the radio will default to MEDIUM AGC mode which will provide the most versatile response. The AGC mode may be changed at any time to take advantage of changing band conditions.

format: 'G' cc <cr>
where: G = 'G' (Ascii 0x47) character
cc = Control Character
ASCII '1' (0x31) for SLOW
ASCII '2' (0x32) for MEDIUM (default)
ASCII '3' (0x33) for FAST
response: none.
example: G1 0x0D for SLOW AGC Mode.
default: MEDIUM AGC

RF GAIN CONTROL

The relative RF gain can be controlled over a range of 0-255. A setting of 0 represents full RF gain whereas a setting of 255 (0xff) represents the minimum RF gain level. Because this control directly affects the RF hardware this will directly affect S-Unit responses and Squelch settings.

format: 'A' nn <cr>
where: A = 'A' (Ascii 0x41) character
nn = 8 Bit Attenuation level (0x00 to 0xff)
<cr> = ASCII carriage return (0x0D)
response: none.
default: 0 (minimum gain)

RF ATTENUATOR CONTROL

An RF Attenuator may be switched in or out under software control. The Attenuator applies approximately 15 db of attenuation. Because this control directly affects the RF hardware this will directly affect S-Unit responses and Squelch settings.

format: 'B' ch <cr>
where: B = is the ASCII 'B' (0x42) character
ch = Setting
'1' (0x31) = ON
'0' (0x30) = OFF
<cr> = ASCII carriage return (0x0D)
response: none.
default: Off



Keyer Timing

The Pegasus contains a built-in iambic keyer. Keyer DIT and DAH timing are programmed separately to provide a means of controlling keyer weighting. The key jack located on the front of the Pegasus is a dual purpose jack. It will be a key jack (straight key) if the Keyer is disabled or a keyer jack (dit/dah) if the keyer is enabled. See Transmitter Controls on page 24 for a description of how to enable/disable the built-in keyer.

format: 'E' dith ditl dahh dahl spch spcl <cr>
where: 'E' = is the ASCII 'E' (0x45) character
dith = high byte of 16 bit dah factor
ditl = low byte of 16 bit dah factor
dahh = high byte of 16 bit dit factor
dahl = low byte of 16 bit dit factor
spch = high byte of 16 bit space factor
spcl = low byte of 16 bit space factor
<cr> = ASCII carriage return (0x0D)

calculation:

```
ditfactor=(int)((double)0.50/speed*(double)0.4166*(double)0.0001667);  
spcfactor=(int)((double)0.50/speed*(double)0.4166*(double)0.0001667);  
dahfactor=ditfactor*3;
```

response: none.
default: approximately 8 wpm.

TX Audio Monitor Volume

You can listen to the Tx Audio of the Pegasus using Monitor Volume Control. The range is 00 (off) to (0x3f) full on. The speaker volume setting also affects this output. The monitored output is prior to any tx filtering.

format: 'H' nn <cr>
where: H= (ASCII 'H' 0x48) character
nn = 6 bit monitor setting (00-3f)
<cr> = ASCII carriage return (0x0D)
response: none.
default: 0 (off).

CW Sidetone Volume

You can adjust the CW Sidetone Volume over the range 00 (off) to 255 (full on). The speaker volume and line output settings also affect this output.

format: 'J' nn <cr>
where: 'J' = is the ASCII 'J' (0x4A) character
nn = 8 bit monitor setting (0 - 255)
<cr> = ASCII carriage return (0x0D)
response: none.
default: 255 (full scale)



Noise Reduction and Automatic Notch

The Noise reduction (NR) and Automatic Notch (AN) are available for use in SBB and CW receive modes. They can be used individually or together.

format: 'K' nr an <cr>
where: 'K' = is the ASCII 'K' (0x4B) character
nr = Noise Reduction State (1=on 0 =off)
nr = Auto Notch State (1=on 0 =off)
<cr> = ASCII carriage return (0x0D)
response: none.
default: both off.

Transmit Audio Source Select and Mic Gain Settings

In SSB and FM transmit modes there are two possible input sources for transmit audio. The microphone or auxiliary audio input can be selected via software command.

format: 'O' src gn <cr>
where: 'O' = is the ASCII 'O' (0x4F) character
src = transmit audio source
0 = microphone
1 = auxiliary input
gn = 4 bit audio input gain factor
<cr> = ASCII carriage return (0x0D)
response: none.
default: microphone is the transmit audio source.

Output Power

Transmitter output power can be set from 0 -255 which represent a power of approximately 5 - 100 watts.

format: 'P' pwr <cr>
where: 'P' = is the ASCII 'P' (0x50) character
pwr = output power setting 0 -255
<cr> = ASCII carriage return (0x0D)
response: none.
default: 0 (minimum power output)



Software Key Down

The radio may be placed in transmit mode using a software command. This operates as if the PTT or Key had been activated. KEEP ALIVE and TX ENABLE setting will also affect this mode. By placing the radio in CW mode first, this command can be used to simulate a 'TUNE' feature on the host PC. This can also be used to create a 'MANUAL TRANSMIT' feature.

format: 'Q' ch <cr>
where: 'Q' = is the ASCII 'Q' (0x51) character
ch = set transmit state
0 = key up
1 = key down
<cr> = ASCII carriage return (0x0D)
response: none.
default: off.

Squelch (all mode)

The squelch is active in all modes and is programmed in 'S' or signal strength units. The signal strength range is 0 to about 19. A setting of 0 would essentially turn the squelch off. An 'S' unit represents 6 dB of receiver sensitivity.

format: 'S' su <cr>
where: 'S' = is the ASCII 'S' (0x53) character
su = squelch setting in s units
<cr> = ASCII carriage return (0x0D)
response: none.
default: 0 (off).

VOX On/Off

VOX mode is available in SSB and FM modes. The VOX can be enabled or disabled under software controls. Settings for VOX parameters (GAIN, ANTIVOX and VOX HANG) will immediately take effect when the VOX is enabled.

format: 'U' ch <cr>
where: 'U' = is the ASCII 'U' (0x55) character
ch = Vox state
'0' = off
'1' = on
<cr> = ASCII carriage return (0x0D)
response: none.
default: off.



VOX Gain

This controls the VOX sensitivity to the audio input. The higher the number the more easily the VOX system will trigger. The MIC GAIN setting will affect this control and should be done before setting the VOX settings.

format: 'UG' gn <cr>
where: 'U' = is the ASCII 'U' (0x55) character
'G' = is the ASCII 'G' (0x47) character
gn = 8 bit vox gain setting
<cr> = ASCII carriage return (0x0D)
response: none.
default: none.

VOX AntiVox

This controls the sensitivity of the VOX system to speaker audio that may enter the microphone input. The higher the number the more the VOX system ignore speaker audio.

format: 'UA' gn <cr>
where: 'U' = is the ASCII 'U' (0x55) character
'A' = is the ASCII 'A' (0x41) character
gn = 8 bit anti-vox gain setting
<cr> = ASCII carriage return (0x0D)
response: none.
default: none.

VOX HANG

This controls how long the transceiver will remain keyed following VOX trigger. This should be adjusted for operator comfort and can be controlled from 0 to more than 5 seconds.

format: 'UH' tm <cr>
where: 'U' = is the ASCII 'U' (0x55) character
'H' = is the ASCII 'H' (0x48) character
tm = 8 bit hang time setting. Range 0 to 255.
Time delay = (tm*0.0214) seconds.
<cr> = ASCII carriage return (0x0D)
response: none.
default: none.

CW SPOT Level

This set the audio level of the CW SPOT tone. The tone will only be audible in CW mode. To turn the SPOT function OFF the level should be set to zero.

format: 'F' b <cr>
where: 'F' = is the ASCII 'F' (0x46) character
b = 8 bit volume setting
<cr> = ASCII carriage return (0x0D)
response: none
default: 0 (off).



Transmitter Controls

This group contains settings which control how the transmitter will operate. Through these controls the transmitter can be enabled or disabled and the T-Loop (for amplifier control) can be activated. There is also control of the Keep Alive Timer.

format: '#n<cr>'
where: '#' = is the ASCII '#' (0x23) character
n = control code....
 '0' - Disable Transmitter.
 '1' - Enable Transmitter.
 '2' - Disable 'T' loop (amplifier loop).
 '3' - Enable 'T' loop.
 '6' - Enable Keyer.
 '7' - Disable Keyer.
 '8' - Disable Keep Alive.
 '9' - Enable Keep Alive.
<cr> = ASCII carriage return (0x0D)

response: none.



Query

The Pegasus also provides a limited number of radio query operations. Some, such as SIGNAL STRENGTH QUERY are used often. Other, such as READ AUX PORT RAW are use for testing purposes and documentation is provided here for completeness.

format: '?' ch <cr>
where: '?'= is the ASCII '?' (0x3F) character
ch =
 'V' = Version Query
 'X' = DSP Signal level
 'Y' = Analog Agc Level
 '!' = Read Auxiliary Port (RAW)
 'F' = Read forward Power
 'R' = Reflected Power
 'S' = Signal Strength in 'S' units
<cr> = ASCII carriage return (0x0D)

response:

'V' (Version Query)

String response in the form 'VER 1134' which represents version 1.134.

'X' (DSP Signal Level)

Responds with the letter 'X' followed by the signal strength relative to full scale DSP input. The numeric format is a two byte hex number.

'Y' (Analog AGC Level)

Responds with the Analog AGC level as a single ASCII character. (range 0-255)

'!' (Read RAW Auxiliary status)

Returns a single byte representing the status of the 8 bits on the Auxiliary port. Intended as a testing aid.

'F' (Forward Power Reading)

Responds with 'F' followed by a single byte representing the forward voltage from the final amplifier.

'R' (Reflected Power Reading)

Responds with 'R' followed with a single byte representing the Reflected voltage from the final amplifier.

'S' (Signal Level)

Responds with the letter 'S' followed by the signal strength in 'S' units. The numeric format is a two byte ASCII sequence with the first byte containing the whole 'S' units and the lower byte containing the fractional 'S' units.



Operation with Optional 302 Remote Encoder

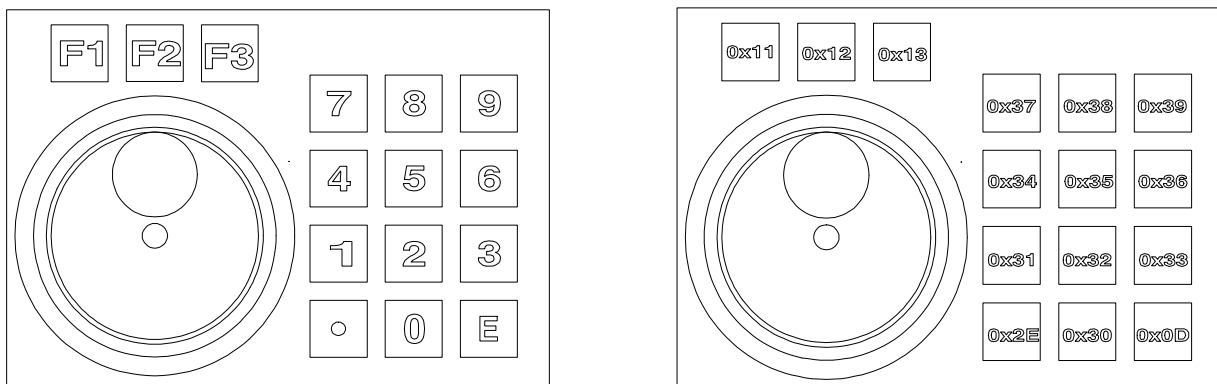
The optional 302 Remote Control provides the Pegasus with a remote tuning encoder, direct frequency keypad and auxiliary function keys. The 302 connects to the REMOTE jack on the front of the Pegasus. The additional features provided by the model 302 are actually contained in the Pegasus. The 302 provides the physical knob and switches while firmware in the radio processes the input and passes the information to the host PC. The host PC can determine what action to take when a 302 notification is received.

KEYPAD OPERATION

When a key is pressed on the remote box the DSP interprets the action and sends a notification to the host PC. The notification indicates the active key and if the key was being pressed or released. The diagram below shows the mapping of the keys on the model 302. Note that the numeric keys are mapped to standard ASCII characters while the function keys are not.

When a notification is received by the host PC the direction of key travel can be determined by the high bit of the keycode. If d7 is low the key was depressed; if d7 is high the key was released. The design of the 302 prevents the activation of multiple keys and keys are non repeating. The host PC could simulate a repeating key by using the key-down/key-up sequence to start and stop a key repeat routine.

format: 'U'k<cr>
where: 'U' = is the ASCII 'U' (0x55) character
 k = keycode....
 See Diagram Below.
 if d7 is low key was depressed.
 if d7 is high key was released.
 <cr> = ASCII carriage return (0x0D)



(Figure 5) 302 Remote Encoder/Keypad and keypress assignments.



ENCODER OPERATION

The optical encoder on the 302 is connected to the DSP inside the Pegasus. A routine running inside the radio interprets the encoder movement and notifies the host PC. The notification indicates direction, step count and key press status. The program running on the PC could then choose how to use the information. For example, if the encoder is for tuning frequency, the program would then make the appropriate change in frequency and send the new tuning factors to the radio. Once encoder movements are passed to the host PC the movement registers inside the Pegasus are emptied.

The notification from the Pegasus contains a numeric value that indicates the encoder movement. The value is a 16 bit signed 2's complement format. There is also a byte which indicates if a key is being pressed. Again, the host can decide what to do with the notification.

format: '!'hlk<cr>

where: '!' = is the ASCII '!' (0x21) character
h = high byte of 16 bit movement register.
l = low byte of 16 bit movement register.
k = keycode....

See model 302 Diagram.

<cr> = ASCII carriage return (0x0D)

example: '!(0x00)(0x02)(0x00)(0x0d)'

Notification that the encoder moved 2 steps in the positive direction.
And that no key was pressed at the time.

