KEYBOARD INPUT MESSAGE WITH AUTOMATIC SWITCHING IN CONNNECTED MODE AND RETURN. FOR PACKET RADIO USING SOFTWARE APPROACH

Robert M. Richardson, W4UCH
22 North Lake Drive
Chautauqua Lake, N.Y. 14722

ABSTRACT:

A number of options open to the programmer are discussed including manual switching, using interrupts, and the psuedo interrupt mode. The objective is to emulate as closely as possible the packet radio hardware approach (VADCG and TAPR TNCs) which utilize their own microprocessor, ROM, and RAM plus a separate host microcomputer. Obviously, a single microcomputer using the software approach does not have the capability of TWO microcomputers (the VADCG and TAPR TNCs are dedicated packet microcomputers), but with judicious programming this problem may be finessed.

INTRODUCTION:

There are many options open to the programmer who is writing subroutines for keyboard message input. A number of approaches that could be used in the connected mode are:

1. While keyboard inputting messages, ignore ALL incoming packets till the operator chooses to switch to receive mode. We will label this approach, IGNORE TILL READY. It is the approach used by the author's Volume 2 - AX.25 Protocol.

2. Use the Model I/III TRS-80's Z-80 microcomputer interrupt mode 1 to switch back and forth between keyboard input and transmit/receive mode data processing. This may be accomplished by splitting both the transmit AND receive 'bit' time delay countdowns between each bit processed into quadrants and using the quadrants for sequentially doing the keyboard processing; i.e, scanning the keyboard psuedo memory locations, converting the keyboard input to ASCII, and stashing the ASCII byte into memory, sequentially during each quadrant's idle countdown time. We will label this approach, INTERRUPT MODE 1. It is a rather fascinating challenge, but not all that difficult. Its major detriments are the amount of memory required and the considerable care that must be exercised to avoid disturbing the software digital phase locked loop while in receive mode or the 'bit' countdown timing in transmit mode.

3. There is yet another approach. While keyboard inputting a packet message to be transmitted in connected mode, have the program test the input from the receiver (via the EXAR 2211 AFSK demodulator and port zero) every millisecond or so and IF a valid change occurs, then automatically switch the program to receive mode where the incoming packet is decoded. If not for your station, then automatically switch back to keyboard input mode. If for your station and it was received correctly, then acknowledge the packet and then automatically switch back to keyboard input mode. This is the approach this paper will discuss with a few enhancements. We will call this the PSUEDO INTERRUPT approach since the change in input in essence brings about an interrupt service subroutine.

GENERAL:

This software approach uses the 1024 byte page of memory beginning at 28672 in memory for keyboard inputting short single frame packets. Long multi-frame packets, up to 12K, are input to low memory beginning at 17408 decimal. During the balance of this discussion we will presume you are already connected to another station. As such, you would press V from the main menu (Figure 1) which takes the program to the edit/modify mode and displays a 1024 byte page of zeros beginning at 28672 decimal in memory.

To activate the modify mode, press M to light the blinking rectangular cursor in the upper left hand corner of the video display. Most any key pressed will insert its ASCII value into the video display AND the corresponding memory location except for the arrow keys which move the cursor on the page rather quickly within the page boundaries, the shift zero keys which insert sequential decimal 128 end of message delimiters;, and the shift @ keys which display the decimal value beneath the cursor.

Figure 2 is the commented source code for that segment of the edit/modify subroutine that performs the blinking cursor operation in modify mode, plus the test for a valid change from the receiver. By valid change we mean:

A. That the EXAR 2211 has noted a change from a mark to space, or vice versa.

B. And that the EXAR 2211 data carrier detect (DCD) has not dropped.

Blink A starting in line 5760 stashes the value in video memory (IX) beneath the blinking cursor in the memory location noted by the label HOLDIT and then loads the same video location with the rectangular cursor character = 143 decimal. TEZRCV in line 5860 is then CALLed.

Lines 5860 - 5980:
First save registers BC, DE, HL, IX, and IY in the stack and then start a few millisecond countdown. Note that lines 5880 and 5910 sample the EXAR 2211 output via port zero, and if any change occurs, then line 5930 jumps off to LOOK in line 5990. If no change occurs during the countdown, line 6100 restores the saved registers from the stack and then returns whence called + 1.

Blink B starting in line 5820 simply replaces the stored character back onto video, CALLS TEZRCV, and then returns. Blink A and B are alternately called frequently enough to comfortably display the blink effect about 30 times a second.

Lines 5990 - 6110:
First test to see if the operator is in the connected mode = CONEK1 set to 1. If not, then line 6010 jumps back to the TEZRCV countdown. If connected, then WATE is tested to see if the operator had previously transmitted a WAIT (RNR) command, and if so, then jumps back to the TEZRCV countdown. The EXAR 2211 DCD is then tested and if dropped (no mark or space tone present), then jumps back to the TEZRCV countdown. If it gets this far, to line 6080, then SIGN10 is set = come back to the modify mode, the receive mode video display restored in 6100, and then the programs jumps off to receive mode to process the incoming packet. When done with the incoming packet, SIGN10 at the beginning of the receive mode subroutine sends the program off to line 6120.

Lines 6120 - 6200:
First zero out SIGN10, then save the receive mode video display in memory, restore the modify mode video display exactly the way it was before, restore all the modify mode registers, and lastly return to wherever the program was in the modify mode.

OPERATION:

When connected to another packeteer on an otherwise quiet non-repeater 2 meter frequency, this approach works perfectly. When on a packet digi-peater frequency it works very well as long as activity is low; i.e., not more than 2 pairs of stations on frequency who are NOT sending long files back and forth.

IF on a busy packet digi-peater frequency, one has no choice but to send the other station a WAIT (RNR) command from the main menu by pressing 3, type in the keyboard input message, and then send it from the main menu by pressing V. Your info packet clears your previous WAIT (RNR) command at the other end of the circuit.

Another approach we tried was having the program send an automatic WAIT (RNR) command whenever you entered the modify mode when connected. This of course worked quite well, but seemed a rather needless transmission on a quiet channel, so was removed and left for the operator to perform manually when desired. Press 3 from the main menu.

CONCLUSION:

This simple subroutine hopefully removes the last objection to our software approach by dyed in the wool hardcore hardware approach packeteers who get their jollies by pointing out subliminal differences between the 2 approaches. It is fully implemented in the author's 'Advanced Vancouver Protocol - Software Approach* program.

It may easily be implemented in the author's 'AX.25 Protocol - Software Approach' program if desired. During BETA testing of the Volume 2 - AX.25 Software Approach program, about 1/2 of the BETA testers wanted this function, while the other 1/2 saw no need for it as the program allows the operator to switch to receive mode manually if desired. Since the vote was a draw, we chose to omit it from Vol. 2 - AX.25.

If you would like a disk for the 'Advanced Vancouver Protocol - Software Approach' source & object programs, uncommented, with no instructions whatsoever, you are on your own, then send $29 for the disk (specify Model I or Model III TRS-80) to:

Richcraft Engineering Ltd.
#1 Wahmeda Industrial Park
Chautauqua, New York 14722

Alternatively, call Richcraft at (716) 753-2654 on weekdays during business hours for COD shipment in the U.S.

This program includes automatic disk file access in the AUTO mode by the station connected to your station, as mentioned elsewhere in these conference proceedings.

FIGURE 1

VOLUME 3 MAIN MENU

ENTER OPTION DESIRED ? _

CHANGE ADDRESSEE CALL	= A	VE3NEC CONNECT REQUEST CQ	= B
NOW CONNECTED TOGGLE	= C	VE3NEC DISCONNECT REQUEST	= D
SEND PACKETS FROM LO-MEM	= E	VE3NEC CONNECT ACKNOWLEDGE	= F
INPUT' FRAMES/PACKET LO-MEM	= G	THIS IS VANCOUVER FROTOCOL	= H
BACKOFF DELAY TOGGLE ON	= I	AUTO CONNECT TOGGLE ON	= J
NOW IN UPPER CASE MODIFY	= K	W2EUP - GIL BOELKE MESSAGE	= L
DISPLAY/EDIT MEMORY PAGE	= M	SET INFO FIELD LOMEM PACKS	= N
NOW FORMAT VIDEO TOGGLE	= 0	QUICK BROWN FOX MESSAGE	= P
TRANSMIT EXTERNALLY ONLY	= Q	SET OPENING FLAG LENGTH	= R
TRANSMIT TO HI-MEM ONLY	= S	INPUT/XMIT NORMAL INFO	= V & T
CLEAR NON-PGM MEM 17K-62K	= U	INPUT/XMIT ALL STATION	= V & W
ABORT LOW-MEM PAK SEQUENCE	= X	SET RE-TRY IN CONNECT MODE	= Y
SHIFT MENU	= 1	MOVE HI-MEM TO LOW-MEM	= 2
SEND WAIT REQUEST (RNR)	= 3	SEND CLEAR WAIT (RR)	= 4
not shown:		not shown:	
HI TO LO-MEM INSERT CR/LF <---		MOVE LOW TO HIGH MEMORY	--->

VOLUME 3 SHIFT MENU

SHIFT MENU ? _

XMIT 40960 UP CONTINUOUSLY	= A	BOOT DOS READY	= B
LOAD HI-MEM ASCII UUUUUUU	= C	LOAD HI-MEM LOGIC 1111111	= D
EDIT/MODIFY INSTRUCTIONS	= E	CHANGE RECEIVE DPLL BASE #	= F
LOG ON VE3MHZ REPFSTER	= G	LOG OFF VE3MHZ REPEATER	= H
SEND MORSE I.D.	= I	SEND SEQUENTIAL ACKS	= J
CAUTION ** RESTORE DOS **	= K	DISPLAY LOW MEMORY @ 17408	= L
DISPLAY RECV PACKS @ 53248	= M	RESTORE PROGRAM POINTERS	= N
DISPLAY CALL/ADDRESS LIST	= O	MOVE PROGRAM TO LOW MEMORY	= P
SAVE HI-MEM ON DISK	= Q	LOAD DISK FILE TO HI-MEM	= R
TRANSMIT BAUD RATE SELECT	= S	SEND DISK :1 DIRECTORY	= T
CLEAR HI-MEMORY 53248 +	= U	RECEIVE VANCOUVER PROTOCOL	= V
RECEIVE AX. 25 NOT CONNECT	= W	SEND MORSE FROM KEYBOARD	= X
NORMAL DISPLAY - NOT DPLL	= Y	DISPLAY DPLL LAST QUADRANT	= Z
NOTE: SPACE BAR IN RECEIVE MODE = RESEND LAST PACK			

FIGURE 2

```
05720 ;
05730
05740 ;PSUEDO INTERRUPT AUTOMATIC SWITCH TO RECEIVE SUBROUTINE
05750
05760 BLINKA  LD      A,(IX)          ;VIDEO MEM BYTE VALUE
05770         LD      (HOLDIT), A     ;SAVE IT IN HOLDIT
05780         LD      A, 143          ;RECTANGULAR CURSOR
05790         LD      (IX),A          ;DISPLAY IT ON VIDEO
05800         CALL    TEZRCV          ;TEST INPUT CHANGE
05810         RET                     ;RETURN WHENCE U CAME +1
05820 BLINKB  LD      A,(HOLDIT)      ;PREVIOUS VIDEO CHARACTER
05830         LD      (IX),A          ;DISPLAY IT ON VIDEO
05840         CALL    TEZRCV          ;TEST INPUT CHANGE
05850         RET                     ;RETURN WHENCE U CAME +1
05860 TEZRCV  CALL    SAVE            ;SAVE IY,IX,HL,DE,BC
05870         LD      BC, 100         ;COUNTDOWN VALUE
05880 TZ1     IN      A,(0)           ;TEST PORT ZERO
05890         LD      D,A             ;STASH IT IN 'D' REGISTER
05900         DEC     BC              ;DECREMENT COUNTDOWN
05910         IN      A,(0)           ;TEST PORT ZERO AGAIN
05920         CP      D               ;ANY CHANGE ?
05930         JP      NZ,LOOK         ;IF SO, TEST DCD
05940         LD      A,B             ;TEST COUNTDOWN
05950         OR      C               ;FOR ZERO
05960         JP      NZ,TZ1          ;IF NOT, CONTINUE COUNT
05970         CALL    RSTOR           ;RESTORE IY,IX,HL,DE,BC
05980         RET                     ;RETURN WHENCE U CAME +1
05990 LOOK    LD      A,(CONEK1)      ;CONNECTED POINTER
06000         CP      1               ;1 = CONNECTED
06010         JP      NZ,TEZRCV+3     ;IF NOT CONNECTED, IGNORE
06020         LD      A,(WATE)        ;WAIT RNR POINTER
06030         CP      1               ;1 = WAIT PREVIOUSLY SENT
06040         JP      Z,TEZRCV+3      ;IF SO, THEN IGNORE
06050         IN      A,(0)           ;TEST INCOMING VIA PORT
06060         BIT     0,A             ;DATA CARRIER DETECT ?
06070         JP      Z,TEZRCV+3      ;IF NOT, THEN IGNORE
06080         LD      A,1             ;ELSE SET RETURN TO
06090         LD      (SIGN10),A      ;MODIFY MODE POINTER
06100         CALL    RESRCV          ;RESTORE RECEIVE VIDEO
06110         JP      BEFOR1          ;PROCESS INCOMING PACKET
06120 RESMOD  XOR     A               ;RESTORE MODIFY MODE
06130         LD      (SIGN10),A      ;ZERO OUT MODIFY POINTER
06140         CALL    SAVRCV          ;SAVE RECEIVE VIDEO
06150         LD      HL,28672        ; BEGIN MODIFY MEMORY
06160         LD      DE,15360        ; BEGIN VIDEO MEMORY
06170         LD      BC,1024         ;FULL PAGE TO DISPLAY
06180         LDIR                    ; MOVE THEM TO VIDEO
06190         CALL    RSTOR           ;RESTORE MODIFY REGISTERS
06200         RET                     ;RETURN WHENCE U CAME +1
06210 ;NOTE:
06220 ;(WATE) IS RESET TO ZERO WHEN PACKET IS TRANSMITTED.
```