AMATEUR RADIO APPLICATIONS SOFTWARE FOR THE APPLE II

BY C.H. GALFO - WB4JMD

- SOFTWARE PRODUCTS DESCRIPTION -

1) COMMUNICATIONS PACKAGE

2) SLOW-SCAN TELEVISION PACKAGE

SOFTWARE AVAILABLE EXCLUSIVELY FROM:

Dr. C. H. Galfo
2229 McLaughlin Aye. #4
San Jose, CA 95122

COMMUNICATIONS PACKAGE

This complete package contains the most sophisticated and versatile radio communications software available for any microcomputer system to date. It allows your Apple II to communicate in any of three codes: Morse, Baudot, or ASCII, with a minimum amount of external hardware required. The program resides in less than 8K of memory and consists of a 6K BASIC executive and a 2K set of machine code utilities which handle all real-time tasks such as display, code conversion, and I/O. Some features include:

-> Live keyboard - Text can be prepared for transmission while either receiving or transmitting.

-> 3 field display - Text is placed on the screen in any of 3 fields which all scroll separately. Received text is formatted for a 16 line display. Text being prepared for transmission is shown in a 5 line field. Text being transmitted is shown in a one line field.

-> Variable size text buffer - This means you can prepare a lot of text and not run out of memory. The buffer space available is about 6K (characters) for a 16K system.

-> User-defined text strings (stored messages) - Memory not used for the text buffer is allocated to stored messages. Any number of messages can be defined. Since their definitions appear in the BASIC language program, they are easily modified and saved on tape as part of the program.

-> Stored text can be inserted anywhere in the text you prepare - This feature can eliminate a lot of repetitive typing during a QSO. Let your computer fill in the details while you continue to type!

-> Automatic 72 character line formatting (on RTTY) - You don't have to worry about putting in a CR, LF for the other guy's printer. Word wrap-around is used so that short words are not split at the end of a line.

-> Eight special control functions allow complete control of your transceiver, program modes, transmit speed and aid in minor editing of your text.

-> Code speeds are continuously variable over a wide range - (CW receive is adaptive over a 3 to 1 range)

-> Automatic CW identification - Your station call can be sent in CW automatically at the end of a RTTY transmission.

-> Many new disk-features and buffered receive.

-> Hardware required - Signals received off-the-air must be demodulated using the usual hardware required for RTTY operation. For CW, a sample circuit is included in the documentation. The computer output can be used to directly modulate a CW or FSK keying circuit. I/O is all via the on-board (game) connector. Four separate lines are used for RTTY & CW in and out. There is an output to control the T/R line of your station.

-> Price — (on cassette with documentation) $20.00 On disk: $30.00

April 1978
Revised: Jan. 1979
Revised: March 1980
Digitized version: April 2015

APPLE II AMATEUR RADIO COMMUNICATIONS PACKAGE

Digitized from printed document with
minor edits for readability by Mike Willegal.
C. H. Galfo has granted permission to reproduce and
distribute

-DOCUMENTATION-

This documentation describes the software package provided on cassette or disk which allows an Apple II microcomputer to be used for communications on the amateur radio bands. Selection of the code to be used (Morse, Baudot or ASCII), as well as the speed (baud rate) provides full flexibility for the user's various applications. I/O is all handled through the Apple on-board (game) connector. These lines are used to key the transmitter, receive signals which the user has demodulated, and control the T/R state of the station transmitter.

The program is provided in two parts: a machine language set of subroutines that handle all real-time tasks and code conversion, and a BASIC language executive which provides a conversational interface between you and the computer. The program is loaded from tape in two parts, the machine language section first, then the BASIC:

```
PROGRAM LOADING:      (Begin in the monitor – i.e. RESET first)

    1) *800.FFFR      (return)     (wait for one bell and no ERR)
    2) *CTRL-B        (return)     (CTRL-B to BASIC)
                                   (stop tape if 2 and 3 take you
    3)  >LOAD         (return)          longer than 8 seconds)
                                   (wait for two bells and no ERR)
```

If you have purchased this package on Apple II floppy disk, the program can be loaded and run by the disk command: >RUN HAM . The machine code segment is automatically loaded from disk as needed. The file 'HAM.V4.SAV' contains the machine code section. After proper loading, the program package is started with the 'RUN' command from BASIC. When started, two text buffers are allocated automatically by the program; one buffer holds the text that you transmit (Xmit buffer) and the other holds the text that you receive. The size of the transmit (Xmit) buffer determines the maximum number of characters that you can type ahead of the text being prepared for transmission. Memory not allocated for the buffers is free for stored text, which is kept in the BASIC program work-space. The size of these buffers (in bytes) is reported to you when the program starts.

In order to use this package for communications, the proper hardware must be interfaced to the Apple I/O lines. A good TU and FSK circuit are required for RTTY. There are numerous articles and books on radio teletype which can help you get started, if you are not already active on RTTY, since it is impossible for me to do justice to communications techniques in this documentation. For CW operation, a threshold detector and transmitter keying circuit are necessary for on-the-air applications. I have included a sample circuit for CW send and receive with this package.

After the program is started, it goes into an interactive mode of operation where several questions must be answered (only the first letter of a word is necessary for non-numeric answers). After you choose Morse or RTTY, an additional question is asked concerning “Word” or “Character” mode. If you answer “Word” then, when transmitting, the text buffer is allowed to empty to the last complete word typed, which is not released until you have finished typing it. This is useful if you intend to operate at a speed that is greater that your typing speed, since it allows you to edit that word if the transmission catches up with you. In the “Character” mode the xmit buffer can empty up to the last character typed. The transmission speed is determined by your answer to another question. When using Morse code, the answer should be between 2 and 125 wpm and this number not only determines the sending speed but also the optimum receiving speed. The actual receiving routine for Morse will adapt to the received code speed if it is within a factor of 3 of the sending speed. When operating RTTY, the sending and receiving rate can be set anywhere between 32 and 300 baud. When using RTTY on the air, you should note that only Baudot code is currently authorized by the F.C.C. for use on the ham bands, and furthermore, only the standard speeds of 45 baud (60 wpm), 50 baud (67 wpm), 57 baud (75 wpm), and 74 baud (100wpm) are allowed. In addition to setting the RTTY communications speed, you will be asked what fill character you would like to use. A fill character is sent whenever the transmit buffer is empty, and you can choose either of the two non-printable characters or not use a fill character at all, in which case the steady mark condition is maintained when the buffer is empty.

Receive

When all questions are answered, the program will go into its receive mode where it is ready to copy signals present on the input lines, as well as allow you to prepare text for transmission at the same time. The screen is now divided into 3 separate fields, the top 16 lines being the received text display area, a single line field near the center of the screen is active during a transmission and shows the present character being transmitted, as well as the next 38 to be sent, and, the last 5 lines at the bottom of the screen show the text that you are placing into the text buffer for transmission. These 3 fields individually scroll, as text is added to each. The text received is automatically aligned using word “wrap-around” for the 40 character line.

Transmit

A transmission is started and stopped using the ESC key which toggles between receiving and sending. When transmitting, you can continue to enter text on the keyboard, as when receiving. If the buffer empties, the transmission of characters will stop but the program will remain in the transmit state until an ESC is typed which will return it to the receive state. There is another way to end a transmission which does not require you to manually end the transmission. This is accomplished by adding a CTRL-C (end of text) after you have finished preparing your transmission. When a CTRL-C is encountered in the text, the program will go into its receive mode automatically (note: on RTTY, a CW ID will be added before switching to receive). The text that you prepare does not require carriage-returns (CR) or line-feeds (LF) to be entered since they are automatically generated to give a 72 character standard communications line width in a format that does not split short words (word wrap-around) when operating RTTY. Also a CR, LF, LTRS sequence is placed at the beginning of every transmission as is common in RTTY operation. You can also enter or leave the transmit mode via a CTRL-T or a CTRL-Q as explained in the next section.

Control Characters

There are many special control characters which are active when you are entering text. The following control characters give an immediate response when typed:

ESC   Toggles between receive and transmit mode.

CTRL-B   Typing a CTRL-B allows you to execute any of six sub-commands. These are activated by typing their first letter.

```
CLEAR – Clears a screen or buffer.
GET   – Gets text from a disk file and places it in a buffer.
PUT   – Puts text from a buffer into a disk file.
DIR   – Displays a disk directory (catalog).
SYS   - Displays current system status.
EXIT  – Exits the program to BASIC.  You can reenter
        properly by using: >”GOTO 0”.
```

CTRL-E   Returns to interactive mode and asks for a definition of the stored message keyword MSG. Enter the text (<128 characters) and terminate with a RETURN. See the next section on the use of stored text keywords.

CTRL-H (<- on the Apple keyboard) Backspaces cursor and deletes a character from the text buffer. Those characters that can’t be deleted, (i.e. have already been sent) result in a “?” on the screen and 8 dits sent on CW.

CTRL-L Typing a CTRL-L (only active when in receive) calls in the contact logging subsystem from disk. All buffers are preserved.

CTRL-Q Similar to the ESC key except that your main Xmit buffer is not used as the source of text when you enter the transmit mode. Text is taken directly from the keyboard which is buffered by an invisible 512 byte “Quick answer” buffer.

CTRL-R See note 1 of communications package notes.

CTRL-S Allows selection of any of five sub-commands:

SPD - Use to change your code speed.
CODE - Use to select Morse, Baudot or ASCII.
MODE - Use to set Word or Character xmit modes.
FILL - Use to change your RTTY fill character.
INVERT - Allows you to receive inverted RTTY signals. Toggles between Normal and Invert as shown by the (TUNE indicator).

CTRL-T Similar to ESC except that your transmission starts at the beginning of your last transmission instead of at the end as with ESC. This is useful in repeating a transmission. Caution: Do not use inserted text keywords in CTRL-T repeated text.

CTRL-U (-> on Apple keyboard) Moves the cursor forward and copies characters to the text buffer.

In addition to the control characters that are listed above, the following control characters are added to the text buffer when typed and the action of each is explained:

CTRL-A Used as delimiter for an inserted text keyword as explained in the next section.

CTRL-C (ETX) Acts as a end-of-text marker in the text buffer. When encountered in text, the transmission is terminated. Also adds a CW ID and CR,LF,LTRS sequence when operating RTTY.

CTRL-I On RTTY only, a CTRL-I will send a CW ID followed by a pair of CR, LF, LTRS sequences and then continue with the transmission.

In Morse and Baudot there are special characters and sequences of characters that do not directly translate into ASCII. The following code conversions are made by the software:

MORSE:	#	-	END OF MESSAGE .-.-.
	$	-	END OF QSO ...-.-
	%	-	END OF WORK ..-.-
	&	-	GO AHEAD -.--.
	*	-	END OF LINE .-.-
	+	-	WAIT .-...
	!	-	BREAK -...-.-

BAUDOT:	#	->	SHIFT-H (STOP)
	CTRL-F	->	CR ONLY
	CTRL-J	->	LF, LTRS SEQUENCE
	CTRL-M	->	CR, LF, LTRS SEQUENCE
	CTRL-B	->	LTRS SHIFT
	CTRL-W	->	FIGS SHIFT

Inserting Stored Text into your Transmission

The flexibility in inserting predefined text “messages” into your transmission is probably the most unusual feature offered by this software package. A stored message (a sentence or phrase) is defined in the BASIC language section of the program as a string of up to 255 characters (a limitation imposed by BASIC). The name of each string, e.g. QTH$, NAME$, ETC$, is used by the software as its corresponding mnemonic “keyword”. The keyword that you will use to reference the text in these strings is the same as the BASIC string variable name without the $ on the end. Thus, the string QTH$ is referred to by simply the keyword QTH. When a keyword is encountered in the transmit text, the stored text in the corresponding string is substituted. To prevent the program from interpreting all your words as possible keywords for stored text, the keywords that you enter into the transmit buffer must be surrounded by a pair of CNTL-A’s. The example that follows on the next page should make this clear:

Define the keyword TNX: (note: should be after line 5000)

```
    5500 DIM TNX$(20)         (Strings must be dimensioned in BASIC)
    5510 TNX$ = “THANKS FOR THE CALL,”     (Define the text)
```

In your transmission, you want to refer to the TNX$ message, so you type:

```
        OK, (^A)THX(^A) OM.   (^A = CTRL-A key)
```

The screen will echo

```
            OK, TNX OM.  (TNX is echoed in inverted video)
```

Now, when the sentence is transmitted it is expanded:

```
            OK, THANKS FOR THE CALL, OM.
```

There are two special keywords used by the software. They are ID and MSG. ID should be defined with your call letters so that a proper CW identification will be generated by a CTRL-C or CTRL-I on RTTY. The text in the keyword MSG can be changed by a CTRL-E without having to stop the program completely. You should change ID to your own call-sign and start adding your own definitions for your QTH, NAME, etc. I have included definitions of the keywords: ID, QTH, RY, & BRAG, the latter two being defined with the string concatenation feature available in Apple’s integer BASIC. To make your changes permanent, the program package should recorded on disk (or tape). This is accomplished:

Tape: (start in monitor – i.e. reset first.)

```
    1) *800.FFFW (return)     (saves machine code segment)
    2) *CTRL-C   (return)     (back to BASIC – no scratch)
    3) >SAVE     (return)     (saves BASIC & your stored messages)
```

Disk: (Apple DOS booted already.)

```
    1)  Exit the program (CTRl-B E)
    2)  >SAVE HAM    (return)
```

-Program Specifications-

Morse:	Dot-Dash Ratio:	1:3 XMIT – 1:2 to 1:4 RCV
	Character Spacing:	3 dots XMIT – 2 to 4.5 dots RCV
	Word Spacing:	5 dots XMIT – greater than 4.5 dots RCV
	Speed:	2 - 125 wpm
RTTY:	Num. of stop bits:	2 XMIT – 1 or greater RCV
	Num of data bits:	5 (Baudot) or 8 (ASCII)
	Carriage shift:	“Unshift on space” (Baudot RCV only)
	Non Print-over:	CR ignored on RCV. (new line of LF only)
	Speed:	32-300 baud (continuous)

.-.-.   .-.-.   .-.-.   .-.-.   .-.-.   .-.-.   .-.-.

- AMATEUR RADIO COMMUNICATIONS PACKAGE -

- MEMORY MAP & I/O CONVENTIONS -

Address (Hex)	Description
0000 - 0049	Page zero storage (shared with Apple monitor).
004A – 00FF	Page zero used by BASIC.
0100 – 01FF	6502 Stack.
0400 – 07FF	Video screen - character map.
0800 – 0AFF	Morse receive/send and keyboard input routines.
0B00 – 0B5F	Constant data and small buffer.
0B60 – 0BBF	Baudot conversion table.
0BC0 – 0BFF	Morse conversion table.
0C00 – 0FFF	RTTY receive/send and disk I/O routines.
1000 – LOMEM	All text buffers.
LOMEM – HIMEM	BASIC program workspace and stored messages.

INPUT / OUTPUT

SW0 – RTTY input signal (mark = High).

SW2 – CW (Morse) input signal (key-up = High).

AN0 – RTTY output signal (mark = High).

AN1 – T/R control line (goes low when transmitting).

(setting variable “DELAY” to a number N in BASIC program will cause the transmission to be delayed N ms after this line goes low).

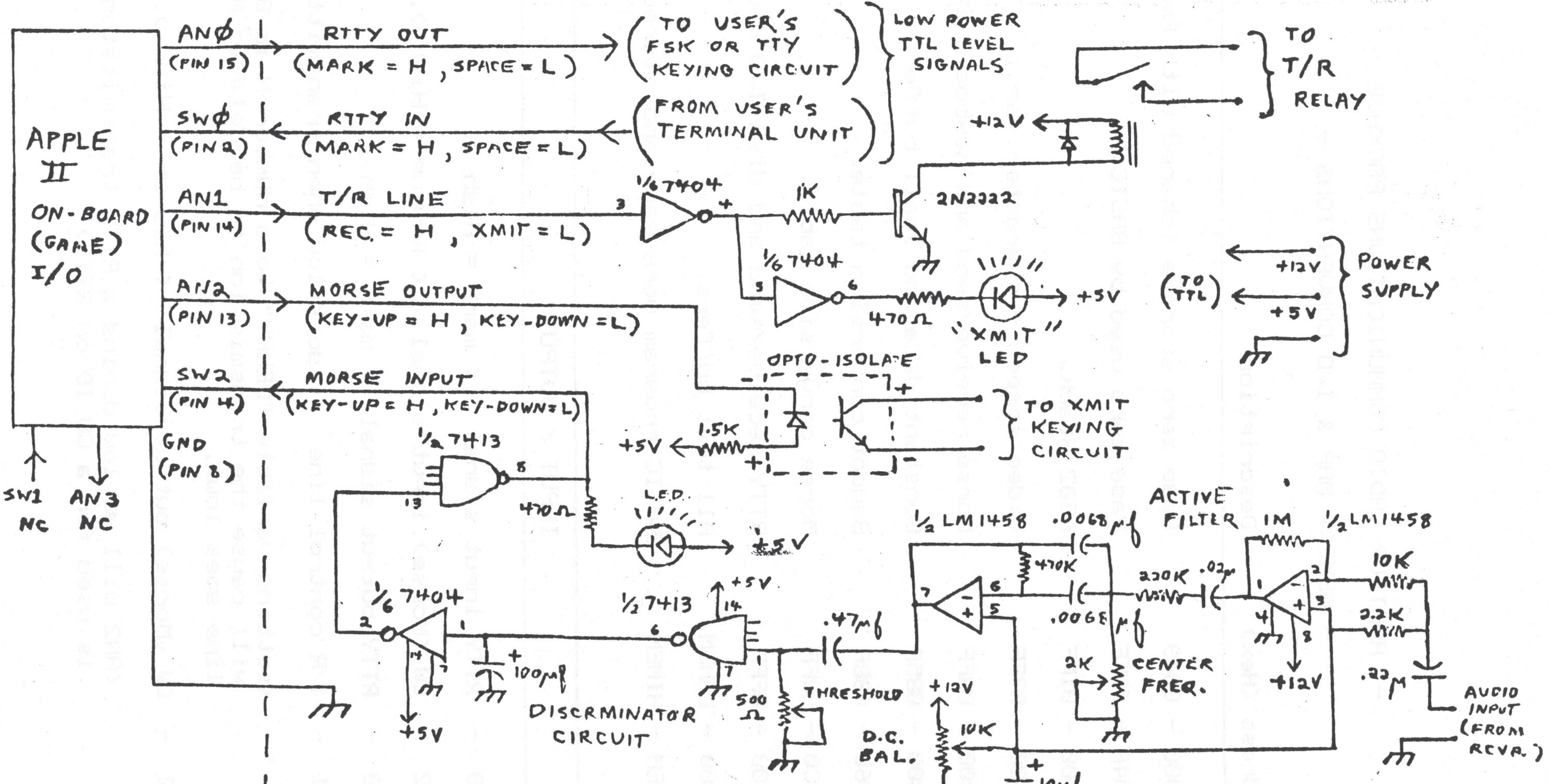
AN2 – CW (Morse) output signal (key-up = High).

(AN2 will go low during a RTTY transmission – This output is used for a CW ID on RTTY)

EXAMPLE INTERFACE FOR THE APPLE II

AMATEUR RADIO COMMUNICATIONS PACKAGE

BY C. H. GALFO - WB4JMD



NOTE: BY-PASS ALL LINES GOING TO THE COMPUTER USING SMALL (~200pf) CAPACITORS TO REDUCE R.F.I.

LOGGING SYSTEM

This brief documentation describes the BASIC program called ‘HAM.LOG’ which can be used as a stand-alone program or as a subsystem when operating with the communications package. When called as a subsystem (with a CTRL-L) all your operating parameters are preserved (i.e. the contents of both buffers, mode of operation, code speed, etc.). You should find that the logging system is very easy to use, since it is written as a menu driven, conversational program. It basically allows you to keep your station log on the disk. Each disk file is referred to as a log book. You can have as many log books as you wish (book numbers 1 to 32767) and each log book can have as many entries in it as YOU wish (the limit is the size of the disk - about 1250 entries). For the purpose of printing, each log book can be thought of as having pages, with 50 log entries on each page. The log entries are kept in a format that is very similar to the standard ARRL log form. A sample print-out of a log is shown below:

```
                 LOGGING SYSTEM            LOG BOOK 999     PAGE 1

----------------------------------------------------------------------------------
START  STATION  HIS UR  FREQ. EMS. PWR. END
TIME CALL SIGN  RST RST MHZ.  TYP. WATT TIME QTH AND OTHER DATA     NAME
----------------------------------------------------------------------------------
<<<< SEPT. 17, ‘76 >>>>
2207 WA4AJF     59  59  3.947  A3J 90   2235 N. VA.                 JAY
2225 WA9NEW/4   59+ 59  3.947  A3J 90   2232 NEWPORT HEWS, VA.      BRUCE
2225 K4KDJ      --  --  3.947  A3J 90   2245 BLACKSBURG, VA.        VA. TECH
2225 WB4DRB     57  58  3.947  A3J 90   2310 ARLINGTON, VA.         MARC
2225 K4VWK      59  59  3.947  A3J 90   2310 -                      SCOTT
<<<<SEPT. 22 >>>>
0146 W8VV       589 589 3.628  Fl  150  0212 GRAND RAPIDS, MICH     CAL
110S WA4RIV     56  56  7.220  A3J 90   1120 MIAMI, FLA.            DICK
112S W0LMP      58  58  3.894  A3J 90   1205 BOONE, IOWA            HARROLD
<<<<NOV. 14 >>>>
1540 PJ3AR      589 579 14.090 Fl  180  1535 ARUBA ISLAND           ROY
1536 VE5BX      599 599 14.090 Fl  180  1600 REGINA
```

For best results, the printing of the log should be done on a line printer with at least 80 columns that interfaces to the Apple II via one of the peripheral slots 1 through 7.

- NOTES ON THE COMMUNICATIONS PACKAGE SOFTWARE -

The following notes are included to help answer some questions that you may have concerning the software.

1) When receiving Morse code, the self-adapting algorithm used, sometimes gets locked into a very slow or high speed when copying dead carriers or noise. To clear the routine, type a CTRL-R on the keyboard. This can also be used to force an “unshift” when receiving Baudot RTTY.

2) You can recover form all known BASIC or DOS errors by typing: “GO TO 0(return)” to the BASIC prompt: “>”. All important information will be preserved by reentering either the ‘HAM’ or ‘HAM. LOG’ program, in this manner.

3) For those who wish to change the communications line width to something other than 72, add these lines:

```
3315 INPUT “OUTPUT LINE LENGTH = “,LL
3316 POKE 3308, LL-2 : POKE 3315, LL-8
```

4) The files that are written to disk do not necessarily contain CR’s (carriage returns) in the text. For example, received RTTY would have CR’s, Morse would not, etc. Each file is written with a CTRL-Z at the end of the file. These files can be edited by Apple II text editors that do not demand text containing CR’s.

5) You can convert program listings to text files so that they can be transmitted on the air. Here’s how to do it:

   1) Load the program to be converted.
   2) Add this line to it:

```
32767    PRINT “@OPEN X”:PRINT “@WRITE X”
         LIST 0,32766:PRINT “#”:PRINT “@CLOSE”:END
```

   (note: @ = CTRL-D and # = CTRL-Z)

   3) Now type: RUN 32767

This causes a disk file called X to be created with a listing of lines 0 to 32766 in it. This file can be loaded for transmission by using the ‘GET’ command to place the text in the Xmit buffer.

End of notes.