Microcomputer QSO Robot

Ever dream of a completely automatic station that would make QSOs while you sit back and watch? Today’s microcomputers make it easier than you might think.

By J. C. Sprott, W9AV

This presentation concerns the use of a BASIC program enabling radio amateurs who are TRS-80 owners to make automated cw QSOs. Little or nothing is required to interface the computer to the station equipment. Because of the simplicity of the technique employed, it does have limitations, but for many, this information could open the way to an exciting new area of amateur communication.

In a few years, a microcomputer is likely to be considered an indispensable part of any well-equipped Amateur Radio station. Although microprocessors are finding their way into many ham products such as programmable scanners and Morse keyboards, only a true microcomputer can be adapted conveniently to the diverse needs of enterprising hams.

One of the more popular microcomputers in use today is the Radio Shack TRS-80 Model I, level II. I hadn’t had mine long before setting out to fulfill a long dream of a completely automated cw station that would make QSOs while I sat back and watched. I expected (at least) a modest hardware interface construction project and a lot of tedious assembly-language programming. After thinking about the problem for a while and playing with the BASIC input and output functions, I realized that my goal could be reached without modification or hardware construction and without resorting to the use of assembly language. A similar approach may be used with the TRS-80 model III.

Keyboard-Generated Morse

My first task was to generate Morse code in response to input from the computer keyboard. That is relatively easy. Table I contains the BASIC transmit program. Each of the 47 code characters is stored as six elements of the array X(I,J). Six elements permit the longest characters (such as the comma) to be stored using a coding of 1 for a dot, 3 for a dash, and 0 to fill in spaces. For example, the letter “O” would be: 3, 3, 1, 3, 0, 0. The characters are generated by the OUT function in a loop, the length of which is controlled by X(I,J). Speed of transmission is adjustable up to about 60 wpm.

When using the program with a real-time keyboard, some limitations exist: lack of a buffer, a variation in weighting, a “Lake Erie” swing and “choke up.” The “choke up” occurs when a key is pressed while a character is being transmitted and is actually caused by a programming error in the ROM interpreter.

Table 1

<table>
<thead>
<tr>
<th>TRS-80 Level II BASIC Program for Generating Morse Code from the Computer Keyboard</th>
</tr>
</thead>
<tbody>
<tr>
<td>410 IF XEK(293)=73 THEN FF=234: F$=&quot;</td>
</tr>
<tr>
<td>410 FOR I=1 TO 47: FOR J=1 TO 6: READ X(I,J): NEXT J,I</td>
</tr>
<tr>
<td>420 CLS: PRINT &quot;KEYBOARD ACTIVE&quot;</td>
</tr>
<tr>
<td>430 X$=INKEY$: IF X$=&quot;&quot; THEN 430</td>
</tr>
<tr>
<td>440 I=ASC(X$)-48</td>
</tr>
<tr>
<td>450 IF I&gt;47 OR I&lt;1 THEN PRINT &quot;&quot;; FOR J=14 TO 7:SI:SI=H74: NEXT: GOTO 430 ELSE PRINT X$;</td>
</tr>
<tr>
<td>460 FOR J=1 TO 6: IF X(I,J) THEN FOR K=2 TO 32:XX(I,J): OUT FF,F$; NEXT: OUT FF,0</td>
</tr>
<tr>
<td>470 NEXT FOR J=46 TO 365: SI: NEXT: GOTO 430</td>
</tr>
<tr>
<td>480 DATA 3,3,1,1,3,1,3,1,3,1,1,1,1,1,0,1,3,1,3,3,3,3,3,3,3,3,3,0,1,3,1,3,3,3,3,3,3,3</td>
</tr>
<tr>
<td>1,1,1,3,3,3,0,1,1,1,3,3,3,0,1,1,1,1,1,1,0,1,1,1,1,1,1,0,1,3,3,1,1,1,0,1,3,3,3,1</td>
</tr>
<tr>
<td>1,0,3,3,3,1,0,3,3,3,1,1,3,1,3,1,3,1,3,1,3,1,1,1,0,3,1,1,3,0</td>
</tr>
<tr>
<td>490 DATA 1,1,1,3,1,1,1,3,3,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,3,3,1,0,0</td>
</tr>
<tr>
<td>3,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0</td>
</tr>
<tr>
<td>500 DATA 3,3,1,3,0,0,1,3,1,0,0,0,0,1,1,1,0,0,1,3,0,0,0,0,0,0,0,0,0,0,0,1,3,0,0,0,1,1,3,0</td>
</tr>
<tr>
<td>0,1,3,3,0,0,0,3,1,1,3,0,0,3,3,0,0,0,3,1,0,0,0,0,0,0,3,2,3,0,0,1,3,3,1,0,0</td>
</tr>
</tbody>
</table>

This program is suitable for both models I and III.
This anomaly is almost nonexistent in the model III TRS-80, which provides much smoother operation in the keyboard mode, copies a slightly higher (14%) range of speeds and appears to accept a somewhat larger range of input frequencies through the cassette port. The mentioned limitations could be corrected, but at the expense of a much more complicated and obscure machine-language program and a much more exotic hardware interface. This would void the constraints under which the system was designed. I wanted the program to be entirely in BASIC so that others could understand and modify it easily. I also desired that little or nothing be required to interface the computer to the station equipment so that any amateur with a TRS-80 could try it.

Computer To Transmitter

Transmitter keying is accomplished by the contacts of the relay, which the TRS-80 uses to control the cassette recorder motor. Measurements showed that a potential of 6 volts and a current of 100 mA existed at the recorder jack. I assumed that the relay contacts could safely handle any transmitter keying circuit exhibiting similar voltage and current characteristics. My Ten-Tec Century 21 meets this requirement, and no problems have arisen during many months of operation. For a transmitter having high-kilohertz keying voltages or currents, a second relay or a switching transistor could be used, as shown in Fig. 1.

The task of converting received Morse code to a character that can be displayed on the video monitor is a bit more complicated. For hardware simplicity, it would be nice to be able to feed the detected cw audio signal directly from the receiver to the cassette recorder earphone jack on the computer. This can be done, provided the receiver can supply a 1-volt peak-to-peak, 2-kHz signal across the 100-ohm input resistance. (A correspondingly larger voltage is required at lower frequencies.) For a receiver or transceiver with a low-impedance speaker output (8 ohms or less), the circuit shown in Fig. 2 provides impedance matching and a doubling of the audio frequency.

Table 2 contains the BASIC receive program listing. The program has a number of timing loops that measure the duration of each dot, dash and space. Values of I and J are then generated, and the character is lookup in the string table Y$(I,J). Speed of reception is adjusted automatically. A top speed of about 25 wpm is the practical limit of a TRS-80 Morse receive BASIC program. A number of hardware modifications are available that will increase system speed by a factor of two or more. Copy is nearly perfect from machine-sent cw such as W1AW transmissions, but one should not expect miracles in the presence of QRM, QRN or QSB, or when the code is being sent with the wrong foot.

Automated QSOs

For this type of operation, VOX keying or QSK cw operation is used. Messages are stored in a string such as: CO CO CO THIS IS A ROBOT IN WAW WAW K. The computer generates the code characters one at a time, and at the end of the message switches to the receive mode.

During receiving, a string is generated from the incoming characters. Before deciphering the call of the received station, the computer looks for your station call or a portion of it in the first half of the string. This prevents the calling of a station incorrectly when the QRM is heavy. The string is searched for the last occurrence of a numeral, and the characters between the numeral and final character (usually a k or AR) are counted. If there is one such character, the call is a 2 x 1.

NOTES: a) Notes appear on page 32.

---

**Table 2**

**Level II BASIC Program for Copying Morse Code**

```
400 CLEAR 100: DEFINT A-Z: DIM Y$(6,60): CLS: B=7
610 Y$="ETINAMSORGUKOMHBLFDCP UX O YJ 567": R=-9: 9=3 2 10"
620 N=0: FOR I=1 TO 5: FOR J=0 TO 2I-1: N=N+1: Y$(I,J)=MID$(Y$,N,1): NEXT J,I
630 Y$(5,13)="KN": FOR J=1 TO 63: Y$(6,J)="": NEXT: Y$(6,7)="1"
640 Y$(6,12)="": Y$(6,21)="": Y$(6,40)="": Y$(6,49)="": Y$(6,51)=""
650 FOR I=0 TO 61: J(I)=2I: NEXT: PRINT "RECEIVE ACTIVE"
660 T=0: J=0
670 OUT 255,0: IF INP(255)<128 THEN N=N+1: IF N<26 THEN 670
680 OUT 255,0: N=0: IF INP(255)<128 PRINT "": GOTO 670
690 OUT 255,0: N=N+1: IF INP(255)>128 THEN 690
700 IF N=8 THEN J=J+J(I): B=(9*N+B+20(N+6))/12 ELSE B=(3*B+2*N+2)/4
710 N=0: I=I+1: IF I=6 PRINT "": GOTO 660
720 IF INP(255)<128 THEN N=N+1: IF 2*N>720 THEN 720 ELSE 740
730 OUT 255,0: IF INP(255)>128 THEN N=0: GOTO 670
740 PRINT Y$(I,J): GOTO 660
```
Otherwise, it is assumed it is a $2 \times 2$ or $2 \times 3$ unless the first and last characters are the same, in which case it is probably a $1 \times 2$ or $1 \times 3$. The call of the received station is then stored in a string for future use.

Transmit speed is adjusted automatically to the speed of the received signal (between 10 and 24 wpm) and displayed on the monitor. Wherever 10 spaces in a row are received, the computer assumes that the received station has finished transmitting and the program switches to transmit. This creates a slightly awkward pause that sometimes causes an impatient operator at the other end to begin transmitting again, but it seems to be the only practical solution.

The computer asks a short series of questions such as those in Table 3. Responses can be varied from one QSO to the next to make the computer seem to be more “human.” A considerable amount of program logic is required to extract the relevant information from all the extraneous comments that are inevitably made by those who have never had a computer QSO before. Trial and error, lots of ingenuity and quite a few frustrating QSOs are required before wise cracks about the computer being a “kid” begin to subside!

### Operation

In the first two months of operation, my computer made over 250 QSOs, and the comments received were almost uniformly favorable. Except for a few occasions when I interrupted the computer to answer a particular question, it was on its own. About all I ever had to do was to retune the receiver occasionally when someone called too far off the received frequency. In fact, the system works so well that I was tempted to leave it on all night while I slept, but the FCC requires an appropriately licensed control operator to monitor transmissions.

I prefer to operate on 10 or 15 meters when the band is not crowded and to operate for only an hour or two at a time. Although another computer has yet to answer mine, I’m sure it’s only a matter of time. Upon receipt of an s.a.s.e., I’d be happy to provide further information about this and similar programs that are available at low cost on cassette tapes. I welcome any comments from readers who can improve the operation of the system without greatly increasing its complexity. It would be a contribution we should all welcome.

### Notes

1. Archbold Electronics, 10708 Secovia Way, Rancho Cordova, CA 95670.
2. Simplex, P.O. Box 13887, Tucson, AZ 85732.
3. FCC rules and regulations, §1.91(b)(3): “Automated control” means the use of devices and procedures for control so that a control operator does not have to be present at the control point at all times. (Only rules for automatic control of stations in repeater operation have been adopted.)
4. The ARRL and QST in no way warrant this offer.

### INSTRUCTIONAL UPDATE

I! The Club and Training Department at Hq. has an update sheet available, which can be obtained free for an s.a.s.e., for the 1976 General Class Instructor Guide. The reading assignments match the newest editions of the License Manual and The Radio Amateur’s Handbook and point out additions and deletions. — Maureen Thompson, KA1DYZ, Training Assistant