

Simple Audio Computer to Transmitter Morse Keyer



Have you ever looked at the interfaces available for your PC to send Morse but you didn't want or need all the available features? I did and I decided they weren't for me, so I designed and built my own.

Dennis McCall, KF3AD

Although I was first licensed in 1959, I was only active during the first year. A few years ago I decided to get back into radio, but just long enough to get my Amateur Extra class license. I've listened to CW from time to time to keep up some speed but hardly ever did a QSO. About three years ago I built an OHR 100A low power (QRP) 40 meter transceiver and some equipment to go with it, but never did try very hard to make a contact.¹ Ham radio slipped into the background of my interests. A few months ago my interest increased, so I began to try again. Then came the difficulty.

In the intervening years I had injured my right hand. It had gradually returned to near normal use, but I found the injury was making my code somewhat hard to send and worse to read. I practiced with some PC code readers, and with practice my code improved somewhat but I still had difficulty sending the numeral 3. That normally wouldn't be too much of a problem except that my call was KF3AD.

Bring on the PC

I decided to look at some of the computer-to-CW interfaces currently on the market. My research revealed that what was available was either too expensive for what I wanted, or had more features than I would ever use for the type of CW operating I wanted to do. I wanted to type on the PC and have the PC key the transmitter. That's it — nothing else. I was also only interested in QRP CW and felt that something really simple should be available.

After much searching of the Internet, I didn't find anything that was either simple or inexpensive. Around this time I purchased a used Yaesu FRG-100 receiver for general listening. I added a crystal filter for CW. My intention was to use it as my receiver and use

the transmitters I already had, such as my still great Heathkit HW-8, for transmitting. I also have an old AMECO 40 meter 15 W tube transmitter that I would love to get back on the air. I also had intentions of building some more QRP CW-only transmitters.

Life Gets Complicated

Through the years I had made a number of different interfaces to receive CW and had also developed some receiving software. All of these interfaces used either the serial port or the parallel port to access the interface hardware via toggling specific pins on these ports. When I started looking at interfaces again I found that the serial ports have disappeared and been replaced by USB ports and all my expertise in earlier ports was basically worthless. *BASIC* software had changed so much that I couldn't program the way I used to either.

After taking all this under consideration I decided that I needed to bypass the USB problem and that an audio activated interface

was my best solution. I figured the best way would be to find an available CW software program that played through an external speaker of the PC and have the interface activate a relay that was in parallel with the hand key. This in turn would key the transmitter. I wanted the parallel connection because I didn't want to give complete transmitter control to the computer.

It All Comes Together

My first impulse was that I needed some type of rectifier and filter to convert the tone to an on and off pulse. This approach was successful. I then needed something to activate the relay. A transistor switch looked like it would do exactly what I wanted. What I ended up with was a half wave rectifier consisting of a 1N914 switching diode and a 1.0 μ F non-polarizing capacitor. An audio transformer was used to raise the level to that needed to drive the base of a 2N2222 NPN switching transistor that controlled a small SPDT relay. The relay is in parallel with

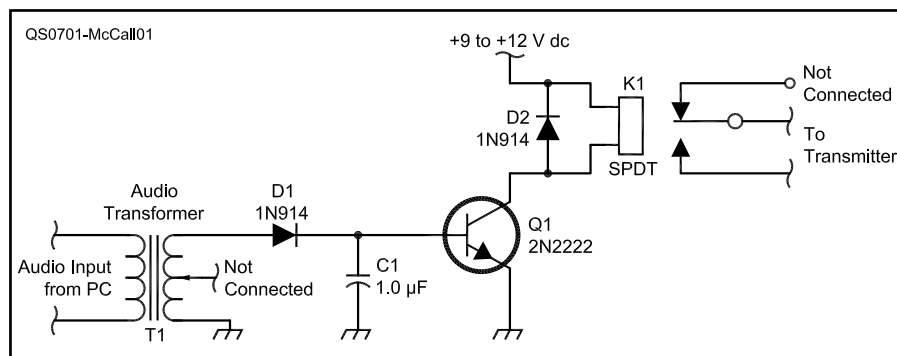


Figure 1 — Audio Morse keyer schematic and parts list.

- C1** — 1.0 μ F, 50 V, non-polarized electrolytic capacitor (RadioShack 272-996).
- D1, D2** — 1N914, or equivalent, silicon diodes (RadioShack 276-1620).
- K1** — Relay, 12 V dc, SPDT, Micromini, contacts rated at 1 A at 120 V ac or 24 V dc (RadioShack 275-241).

- Q1** — 2N2222, or equivalent, NPN switching transistor (RadioShack 276-1617).
- T1** — Miniature audio output transformer 1 k Ω center-tapped to 8 Ω (RadioShack 273-1380).

¹See www.ohr.com/ohr100a.htm.

another 1N914 diode to protect the 2N2222 from the back EMF resulting from the opening and closing of the relay.

The schematic is shown in Figure 1. It is about as simple a project as you can get. This is the one thing that has perplexed me about this project. Why haven't I seen it done before? It's very easy to put together and works really well.

Software Makes it Play

Now I needed some software. There is a lot of CW learning, plus send and receive software on the Internet. A good share of it is based on DOS and uses a serial port, but I didn't find any that used a USB port. I needed software that would output the CW to the external speakers. After much searching and testing I found that *CwType*, available from www.dxsoft.com as freeware would more than meet my needs. Figure 2 shows the *CwType* interface screen. It not only provides type-ahead sending and allows correction but also includes many handy macros. One can use a different function key for NAME, QTH, RST, CQ and much more. It has adjustable sending speed and can even send Farnsworth spacing with a little effort by using its INTER-LETTERS SPACE control (ILS).

The audio input for the Morse keyer is plugged into your PC's external speaker jack and is controlled by either your computer's external speaker volume control or another volume control method. I have a separate jack on the right speaker I can plug into and control the volume output with the speaker volume control. You adjust the volume to get a good crisp-sounding contact of your relay. It

shouldn't be underdriven and chattering and not overdriven and stuck.

This Morse keyer should be of use to persons who have either a minor or major impediment to using a key but still have a limited typing ability. *CwType* has a very large macro capability that will keep extensive manual typing to a minimum. Two other


advantages are the speed option and the type-ahead feature. One can adjust the sending speed so as to not exceed one's typing ability. Typing ahead gives you the opportunity to correct any mistakes that were made in typing.

Capacitor C1 appears to have a range of values over which it will function. The lower the capacitance the higher the speed that can be used, limited by the relay no longer functioning properly due to lack of filtering. I've only tested up to 35 WPM but am sure it can go higher. Higher capacitor values will reduce the maximum WPM rate and at some point the relay will stick closed.

Figure 3 shows the breadboard prototype and how simple it is. Figure 3 also shows three RadioShack 276-1388 two position PC board terminals, which I used for audio input, dc power input and keyer connection to transmitter.

Special thanks to Lester Hudgins, N3LH, who not only did the photos for this article but listened to my theories on Morse Code sending and receiving without laughing too loud.

Dennis McCall, KF3AD, is a retired Air Force officer. He is now employed as logistics engineer with a defense contractor. He is a member of the Historic Electronics Museum Amateur

Radio Club (HEMARC) in Linthicum, Maryland. You can reach him at 1215 Nicodemus Rd, Reisterstown, MD 21136 or at dvmccall@comcast.net. Please contact Dennis with a description of your experiences with this project, especially those that involve its use by anyone with disabilities and limitations in using Morse code. 

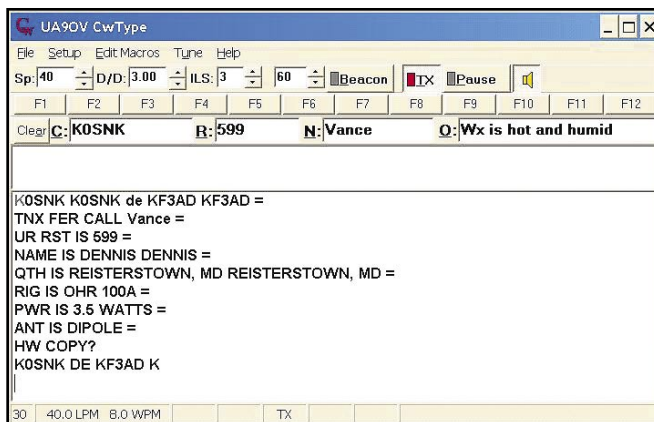


Figure 2 — Screen shot of *CwType* operator interface.

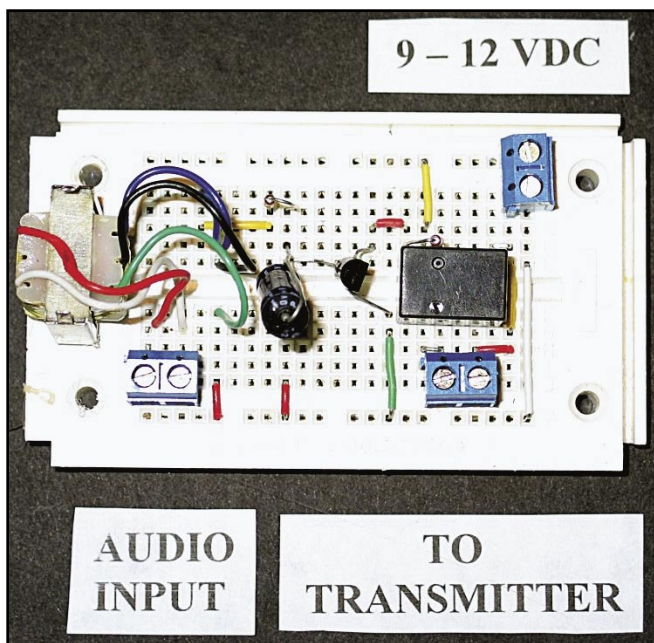


Figure 3 (right)— Prototype of Morse keyer circuit layout.