The UART Gear Shifter

— *for multi-speed RTTY*

James B. Wilson, Jr. KB8CE
5886 DeMorrow Road
Stevensville MI 49127

I'm one of those amateurs who enjoy shortwave listening. I like to listen to shortwave broadcasts, news, and feature stories. Recently, however, my SWLing took on a new twist. I have been monitoring RTTY press transmissions.

Once I was hooked on receiving press RTTY transmissions, it wasn't long before I discovered that most of the stations weren't using the standard amateur speeds or shifts. I took care of the frequency shift problem by modifying my terminal unit for 425 Hz shift. This covers most of the press transmissions, with a few still using 850 Hz shift. My next problem was the speed. I looked into gears and found that it got kind of sticky trying to change gears for each new station! A Model 28 can be modified for a mechanical gear shift. However, the funds at my QTH indicated that a Model 28 was out of the question at the present time. I'm still saving for one! The only solution then was this thing called a UART.

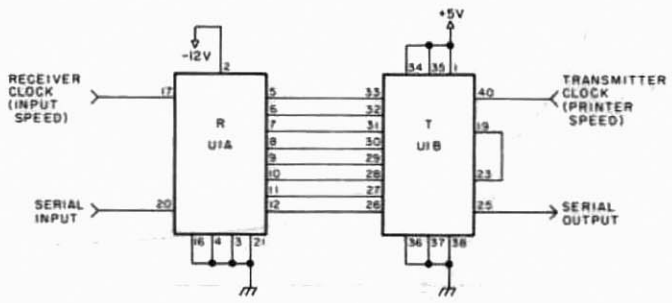

Fig. 1. AY-5-1013 UART.

When I decided to go with the UART, I started gathering all available information on this device. I found out some very interesting things. Firstly, UART stands for Universal Asynchronous Receiver/Transmitter. It seems that this UART takes the serial input data, in this case from the terminal unit, and converts it to parallel data. This parallel data is then applied to the transmitter section, where it is converted back into serial data and then applied to the printer. The speed change occurs due to the clock rates. Both the receiver and transmitter have to be clocked by an external oscillator. The receiver is clocked at the incoming signal speed. The transmitter is then clocked at the *desired* output speed. This particular speed converter

has no provision for converting the speed down. That would require storage of data between the receiver and transmitter. If the receiver was taking in data at a speed of 100 wpm and the transmitter was reading it out at 60 wpm, there would soon be a pileup of data between the two. In the case of press transmissions, the input data would be at a constant speed. This would mean that an extremely large memory would be required to implement a down converter. If a Teletype™ keyboard is the input, as in some previously published articles, the memory could be much less, as the input data would not be at a constant 100 wpm.

Having thought all this through, my next step was to gear my Model 15 for the highest speed possible (at this time 75 wpm, although I have located some 100 wpm gears), and to build a speed converter using a UART. I wanted the speed converter to be a stand-alone unit, mostly because I didn't want to modify any existing gear and also because I had some extra rack space to fill! Making it a stand-alone unit meant that a power supply, Teletype loop, and input interface from the ST-5 loop would have to be included as well as the UART circuitry.

Fig. 1 shows the actual UART circuitry. Not much to it, is there? I'm using an AY-5-1013 UART, as it was available at a reasonable price. U1A represents the receiver section and U1B the transmitter. The connections 5 through 12 on the receiver and pins 26 through 33 on the transmitter are the parallel data lines. This one is shown connected for 8 bits of data. Since Baudot code uses 5 bits of data, it is only necessary to connect pins 8 through 12 to pins 26 through 30. I have mine connected as shown for a possible change to ASCII.

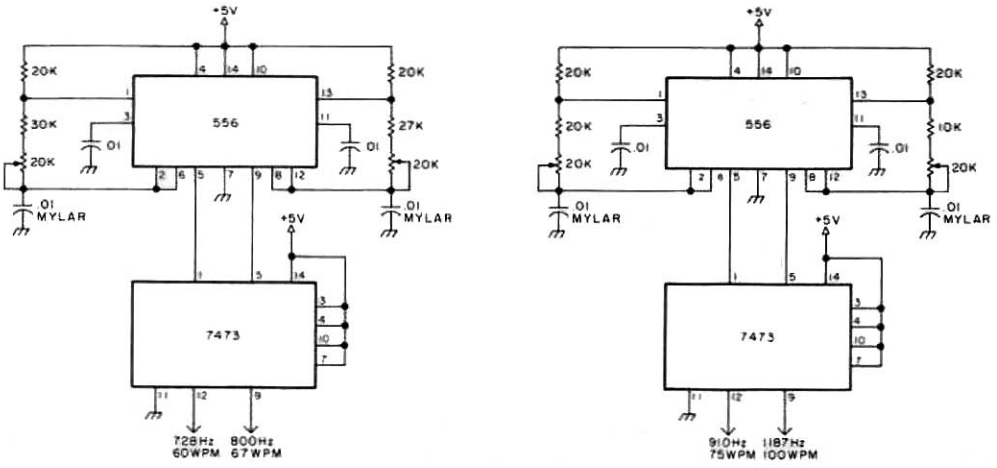

Fig. 2. Clocks. All resistors ½ W.

If you are interested in the whys and wherefores of the other pin connections, I shall refer you to the references at the end of this article or, better yet, to the UART data sheet. The UART shown in Fig. 1 is connected for one stop bit, five data bits per character (Baudot), and no parity bit.

The clocks for the UART, shown in Fig. 2, were of particular concern to me. Originally I wanted to use a crystal-controlled clock. Nothing but the best for my project! But when I started to look for parts, I found them very difficult to obtain, if not downright impossible. A colleague convinced me that the NE555 timer was very stable and could be used in this application. So I decided to try the NE556 dual timer chip for my clock. I must report completely satisfactory results with these chips. They were very easy to design and set up. I added 7473 dual JK flip-flops as a buffer and to make the square waves symmetrical. However, this may not have been necessary. The frequencies shown are the frequencies at the output of the JK flip-flops. On initial setup of the speed converter, I used a frequency counter to set the frequency and have not had to adjust it since, even after a 600-mile move and two months in cold storage! So I would say that the NE556 timers have worked exceptionally well.

The switching arrangement is shown in Fig. 3. It is fairly self-explanatory: one switch to set up the receive or input speed, and one to set up the transmitter or output speed. It is only necessary to set the transmitter speed once, as the printer is only geared for one speed. In an updated version, the transmitter or printer speed switch could be eliminated and its speed hard-wired.

The frequencies for the clocks were calculated using the formula: freq. = baud rate × 16. The factor of 16 comes from the UART itself. The baud rates for 60 wpm = 45.5, 67 wpm = 50.0, 75 wpm = 56.9, and for 100 wpm = 74.2.

The input is designed to interface with a standard 60 mA loop, in my case directly from the ST-5. It uses an optoisolator, in my case a GE 4N35 which I found at my local supply store, much to my surprise. Most any optoisolator should work in this configuration. The optoisolator takes the high voltage current loop and converts it down to TTL levels so that the signal will be compatible with the UART. Getting the RTTY signal down to TTL levels has proven to be quite useful. I have already added a TTL RY generator to the speed converter; it was extremely easy since I only needed to patch it into the input of the UART. This would be true for a lot of the other interesting RTTY projects that have been published. The circuitry for the input interface is quite simple and straightforward, as you can see from Fig. 4.

The output interface is shown in Fig. 5. It uses a 7437 as an output buffer for the UART. This is required because the UART output can only stand one TTL load on it or it will

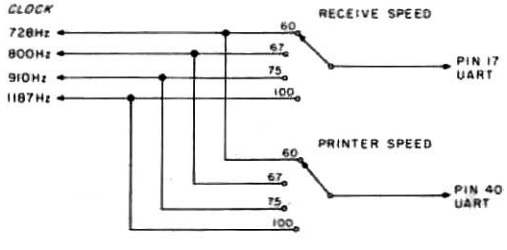

Fig. 3. Switching.

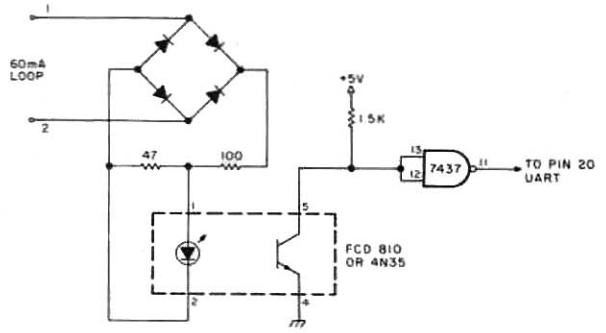

Fig. 4. Input interface. All resistors ½ W.

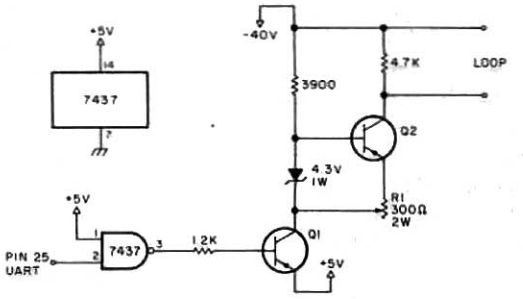

Fig. 5. Output interface.

cease to function. The loop keying transistors, Q1 and Q2, are 2N2528s. These are some TO-3 case transistors I found in my junk box and they work quite well; they do require a sufficient heat sink. I used the chassis as a heat sink in this project. Other suitable transistors are 2N174s, 2N277s, 2N251s, and 2N1501s.

The unusual thing about this output interface is that it uses only a 40-volt supply. I did this to take advantage of the power supply transformer I already had in the unit to power the TTL components and the UART. To get away with using a 40-volt loop supply, I used a special transistor keying circuit and wired the printer magnets in parallel. Wiring the printer magnets in parallel meant that the loop current would have to go up to 120 mA. There are several advantages to be gained by this. When you place the printer magnets in parallel, the total inductance is cut by one-fourth! Let me explain. The magnets are originally in series. This means that the total inductance is the sum of both inductances. When we place them in parallel, the total inductance is reduced to half of one inductor's inductance, if both magnets are equal. This gives us a better keying waveshape on the loop. It also allows us to use a smaller, more convenient power supply. See reference 1(b) for a very interesting discussion of this circuit.

The power supply, shown in Fig. 6, is relatively straightforward. I try to make power supplies as simple as possible, primarily because it seems they are always the one to give me trouble. The UART requires -12 V dc and +5 V dc. These are both supplied by IC regulators. I use the hefty K series here, which may be just a little bit of overkill. The 40-volt loop supply comes off a capacitor which charges to the peak of the incoming voltage. In practice, the output loop voltage does drop somewhat, but this has not been a problem with this particular unit. If there was a larger current drain on this supply, then it would be necessary to use a voltage doubler or separate transformer.

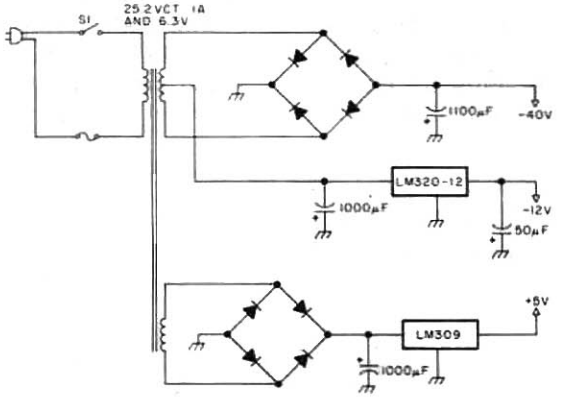

Fig. 6. Power supply.

This unit was constructed using perfboard and point-to-point wiring. As I usually construct only one each of my projects, it has been much more convenient to use perfboard. It is also a lot easier to modify the unit when getting the bugs out, or when adding on or improving. The layout is not critical and any convenient method is acceptable.

Operation of the unit is relatively simple. On installation, check all power supply voltages to be sure they are correct. I did have a little trouble with the -12 V dc regulator; it oscillated! The addition of the 50 uF capacitor on the output terminal took care of that. Next you will want to connect your terminal unit's output loop to the input of the speed converter, and the printer to the speed converter's loop.

Wire an ammeter in series with the printer magnets. At this time you may also want to wire your printer magnets in parallel. This is what I have done to take full advantage of the low-voltage aspects of this particular loop keyer. I have used the keyer with the magnets in series and it did work; however, it is recommended that you take the time to wire your magnets in parallel for optimum performance. Set your terminal unit to standby and turn on the speed converter. The loop should have current flowing at this time. Now adjust the 300-Ohm potentiometer, R1, for a loop current of 120 mA. If your magnets are still wired in series, set the loop current to 60 mA. No other adjustments are necessary, although you may need to do some fine adjustments to the timer frequencies for optimum reception at the particular speed.

That's all there is to it. Just set the printer speed to the speed your printer is geared for, ideally 100 wpm, and the receiver speed to the speed of the incoming RTTY signal. Then you can sit back and watch UPI, Tass, Ceteka, Reuters, or any other press service. It's a great way to get the news! ■

References

1. *The New RTTY Handbook*, 73, Inc., 1977. (a) "TU2 Terminal Unit," pp. 29-33; (b) "Watching Waveforms," pp. 39-41; (c) "UART Speed Changer," pp. 153-155; (d) "Speed Converter and Processor Using the UART and FIFO ICs," pp. 113-116.
2. *RTTY Journal*. (a) "UART," I.M. Hoff and H.L. Nurse, April, 1974; (b) "Using the UART," I. M. Hoff, H. L. Nurse, P. Satterlee, Jr., May, 1974; (c) "The Mainline UT-2," I. M. Hoff, Feb., 1975; (d) "The Mainline UT-4," I. M. Hoff, Mar., 1975.
3. ARRL Specialized Communications Manual.
4. Data Sheet, AY-5-1013/AY-5-1013A, General Instrument Corporation, March, 1974.