An Arduino Baudot Keyboard and RTTY Transmitter

An article called “Rookie RTTY Contesting” in the January 2019 issue of QST by Jerry Flanders, W4UK, got me interested in trying out RTTY. I already have an inexpensive digital interface for JT65 and WSPR, so I followed the author’s advice and downloaded the free RTTY software called MMTTY from http://hamsoft.cal/pages/mmtty.php . I set it up using suggestions made both by the author of that article, and by Steve Ford, WB8IMY, in his ARRL book *Get on the Air with HF Digital*. I soon found myself eavesdropping on contest QSOs (see the WB7BNM contest calendar), and dabbling in RTTY contesting myself.

I also got interested in the Baudot code, and in the life of Emile Baudot. I was fascinated by the 5-key keyboard Baudot patented in 1874 as part of a multiplexing telegraphy system. Baudot’s system allowed several transmitting stations and receiving stations to share a single wire, with a ground return. You can read about the ingenious electro-mechanical system used to synchronize the transmitting and receiving stations in a 1919 publication of the British Post Office Engineering Department called “Pamphlets for Workmen: Baudot Multiplex Type-printing System” available at http://www.samhallas.co.uk/repository/telegraph/b6_baudot_multiplex.pdf .

The Baudot keyboard, illustrated in Figure 1, was used to send a 5-bit code which Emile Baudot created, based on the earlier work of German mathematician and scientist Carl Friedrich Gauss and his physicist associate Martin Weber. Each of these three gentlemen gave his surname to a standard unit of measurement.

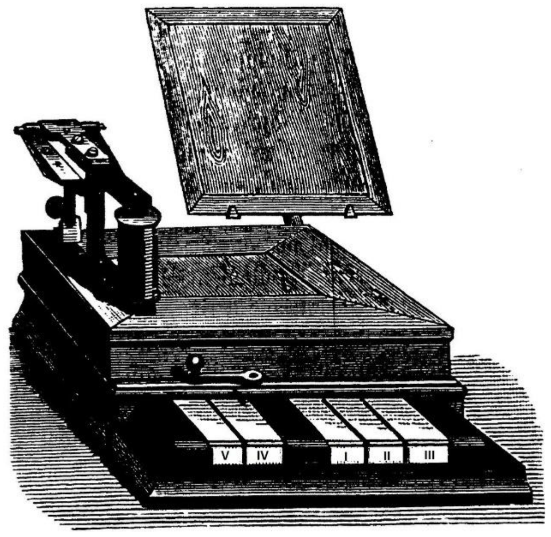
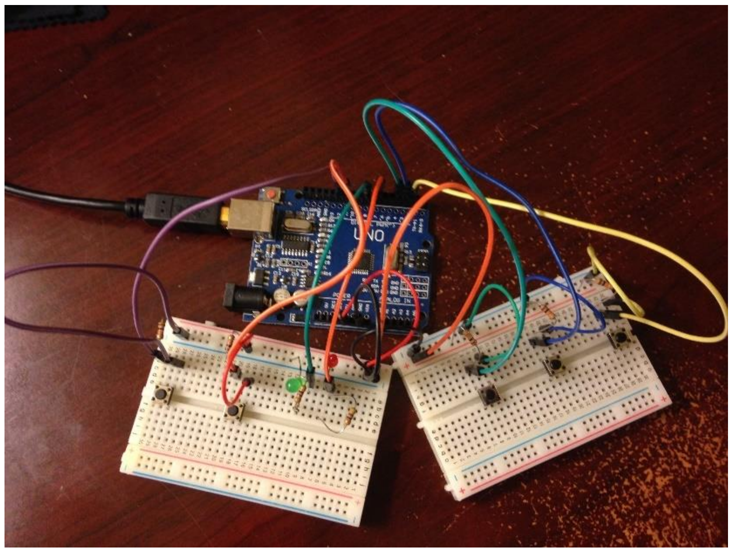


Figure 1

Call me crazy, but this made me want to try out the Baudot keyboard, and maybe learn how to send my callsign and name. I remembered the joy I felt as I began to master Morse code. Could I replicate that joy learning how to use the Baudot keyboard? A little research suggested that these keyboards are rare museum pieces, that I’m not likely to be able to buy or borrow. So I decided to simulate the Baudot keyboard with an Arduino and some pushbuttons, and to try to use that keyboard to make casual RTTY contacts on the ham bands. In Figure 2, you can see the initial version of my Baudot keyboard.

A day of practicing the Baudot code led to the development of five serious cases of pushbutton finger, as shown in Figure 3.

Figure 2



AAA to the rescue! I sliced up an expired AAA card to make keys. Each key is attached to the breadboard by two strips of packing tape. One strip covers about three fourths the length of the key and continues over the rear edge of breadboard, wrapping under the bottom. A second strip is perpendicular to the key, holding the key lightly against the pushbutton, and preventing lateral slippage. See Figure 4.

These keys work reasonably well, but the action is very stiff, requiring considerable force.

Figure 3

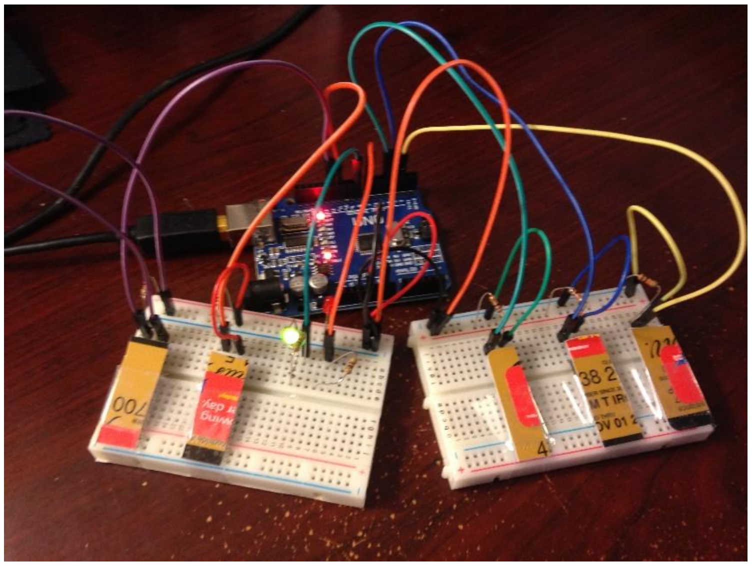


Figure 4

Later, I tried pressure sensitive resistors and touch capacitance sensors, as shown in Figure 5. The pressure sensitive resistors work well, but are extremely expensive. The touch capacitance sensors do not let you rest fingertips on the keys without activating them. So I settled on mechanical keyboard switches. Luckily, it turns out that many keyboard users – from PC gamers to courtroom stenographers – prefer mechanical keyboard switches, so they are readily available. After some experimentation, I settled on a type of keyboard switch called Kailh Speed Silver, which I purchased from www.flashquark.com for $0.32 each. Keycaps are also required. As you might expect, there are plenty of online videos and blogs discussing the advantages and drawbacks of various types of keyboard switches and keycaps.

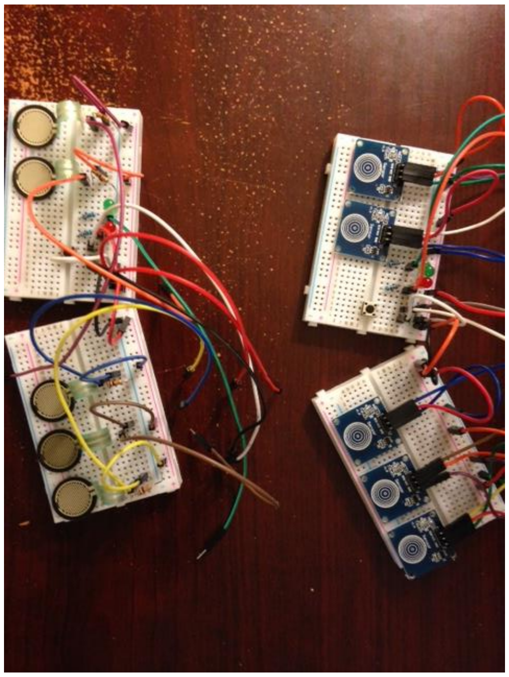


Figure 5

Baudot's original keyboard made a clicking noise (activated by an electromagnet) called a "cadence tap" when it was the telegrapher's turn to send a character. As he pressed down on the keys to form a character, they locked into place until a brush could pass over all 5 contacts, sending a negative voltage pulse down the telegraph line for depressed keys, and a positive pulse for undepressed keys. Then the keys were released. I simulate this process by turning on a green LED when it is time to depress keys. Since my keyboard lacks a locking mechanism, the operator must continue depressing the keys until the red LED completes a flash, signaling that the keys have been read.

The Arduino Uno, with which I started this project, turned out not to be the best choice for an external PC keyboard. While it can be modified to serve as an HID (human interface device) keyboard by flashing it with new firmware (visit http://mitchtech.net/arduino-usb-hid-keyboard/ for a tutorial), I decided to use an Arduino

Leonardo. The Leonardo is designed to be automatically recognized as a USB device by PCs. The Due and the Micro may also be used. There is a Keyboard Library available in the Arduino IDE for these Arduinos, which made coding the keyboard sketch a snap.

A wiring diagram is shown in Figure 6. An ASCII wiring diagram is also included in the comments in the Arduino sketch.

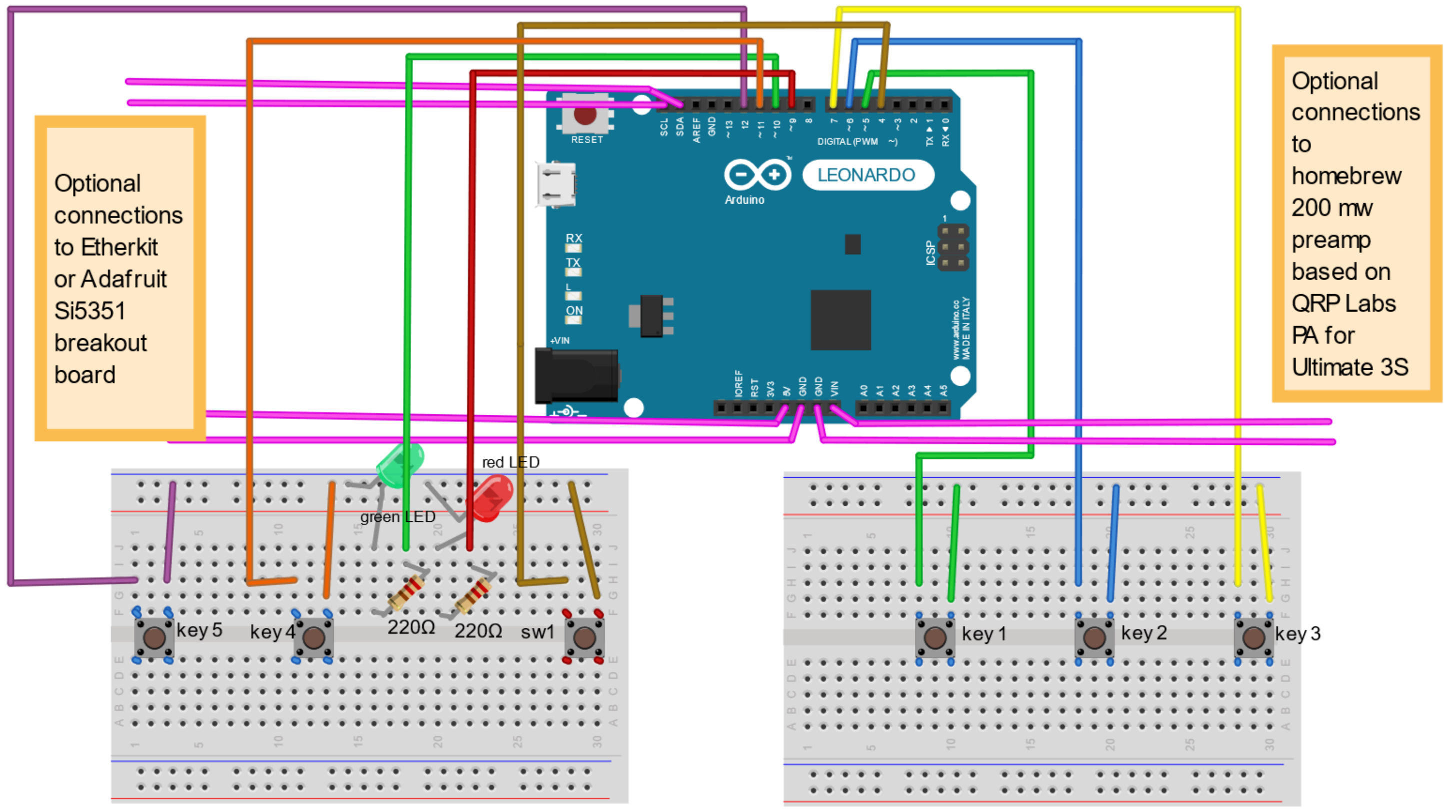

Optional connections to Etherkit or Adafruit Si5351 breakout board
Optional connections to homebrew 200 mw preamp based on QRP Labs PA for Ultimate 3S
LEONARDO
Arduino
RESET
green LED
red LED
220Ω
220Ω
key 5
key 4
sw1
key 1
key 2
key 3

Figure 6

Once you have wired the keyboard, download the .ZIP file called *BaudotKeyboardRTTYTransmitter_V6.zip* from my website at www.mathmage.org/Baudot . Install the Si5351 library included in the .ZIP file, following the directions in the included *Readme.txt* file. Open the Arduino sketch (called *BaudotKeyboardRTTYTransmitter_V6.ino*) in the Arduino IDE, and edit it to enter your own callsign, name, SPC, and beacon text. Then upload the sketch to your Leonardo. Open an application on your PC that permits typing (like Notepad, Word, or MMTTY). Close or minimize the Arduino IDE so you don't inadvertently type into it, corrupting the code! Press key 5 and sw1 to activate the keyboard.

The sketch will type the current transmitter frequency for you to see. Press sw1 to cycle through keyboard speeds, watching the cadence LEDs. The sketch will also type each keyboard speed for you to see. Note that you do not need to connect the transmitter (Si5351, preamp, final, or lowpass filter) to the Arduino to use it as a Baudot Keyboard. If you like, you can simply try out 100+ year old telegraphy technology using the keyboard and a word processer!

With a copy of the modified Baudot Code (see Figure 7) in view, try sending your name and your callsign. (You must press FIGURES key4 to type numbers, then LETTERS key5 to switch back to letters.) Are you starting to feel like a British Post telegrapher?

Modified Baudot Code (ITA1)

LTR	FIG	key 5	key 4		key 1	key 2	key 3
A	1				X		
E	2					X	
I						X	X
O	5				X	X	X
U	4				X		X
Y	3						X
B	8		X				X
C	9		X		X		X
D	0		X		X	X	X
F			X			X	X
G	7		X			X	
H			X		X	X	
J	6		X		X		
K	(	X	X		X		
L		X	X		X	X	
M	)	X	X			X	
N		X	X			X	X
P		X	X		X	X	X

LTR	FIG	key 5	key 4		key 1	key 2	key 3
Q	/	X	X		X		X
R	-	X	X				X
S	;	X					X
T	!	X			X		X
V	'	X			X	X	X
W	?	X				X	X
X	,	X				X	
Z	:	X			X	X	
BS	.	X			X		
FIG	FIG		X				
LTR	LTR	X					
CR LF	CR LF	X	X				
SP	SP				X	X	

X = key depressed

Figure 7

I developed a PCB board (an Arduino Shield) which plugs into the Arduino Leonardo, and holds six Kailh Speed Silver mechanical keyboard switches – five for the Baudot code, and one additional key for the Speed Switch. Any Cherry MX compatible keyboard switch will work with the Shield. The Arduino needs to be mounted to a base larger than the shield, so the whole thing doesn't tip over while keying. You can see an early prototype of the shield in Figures 8 and 9. The Eagle PCB file, as well as the gerber files, for this shield are available on my website at www.mathmage.org/Baudot .

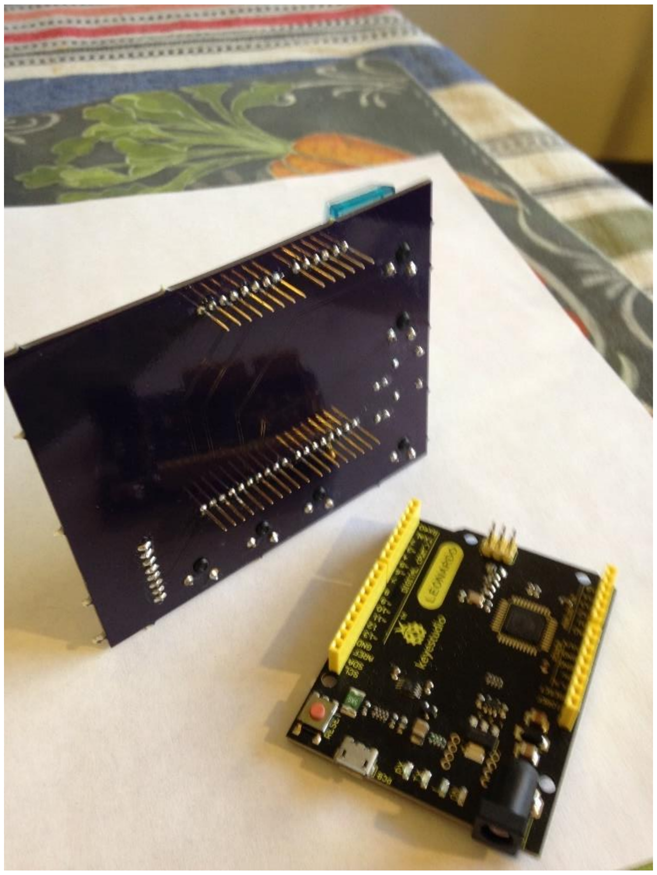


Figure 8

Naturally, I also wanted to incorporate the keyboard into an RTTY transmitter. I had been using an Arduino and an Si5351 digital vfo chip, together with a 200 mw amplifier, for WSPR, so why not RTTY? Thinking 200 mw might be a little anemic for RTTY, I decided to go full bore (QRP style) by adding a QRP Labs 5 watt amp into the rf chain.

Once the keyboard had been incorporated into an RTTY transmitter, it was time to program new key combinations to perform functions like turning transmit on and off, changing frequency, beaconing, contesting, and cq/qsoing. So with the Speed Key also pressed, all 2^5 key combinations of the Baudot code now also perform other RTTY operating functions (See Figure 7 for a list.)

The simplest version of the keyboard and transmitter project, using the Leonardo and with the additional Speed Button, is shown in Figure 10. It used 2 breadboards, 6 very inexpensive breadboard pushbuttons with craft stick lever arms, 2 LEDs and their current-limiting resistors, and 16 Dupont male-male and 2 female-male jumper wires. The rf stages were the Si5351 breakout board (from Etherkit or Adafruit) and a homebrew version of the simple seven component 200 mw amplifier designed by QRP Labs for the PA in their Ultimate 3S transmitter (see their website at https://www.qrp-labs.com/images/ultimate3s/assembly_u3s_r3_lt.pdf for a schematic). This is followed by either a 20 or 40 meter low pass filter, which I built from kits from QRP Labs. A five watt amplifier kit, also from QRP Labs, was added later. The low pass filters and 5 watt amp are not shown in Figure 10.

It took me about a month to learn the modified version of the Baudot code used in this keyboard program. In Figure 7 and in comments contained in the Arduino sketch, I display the code in small groups. These groups were suggested for ease of learning by the British Post Office publication mentioned in the second paragraph above. I eliminated the characters which have fallen into disuse (like " Nº ", and " st ") and added a RETURN, a SPACE character, and a BACKSPACE character.

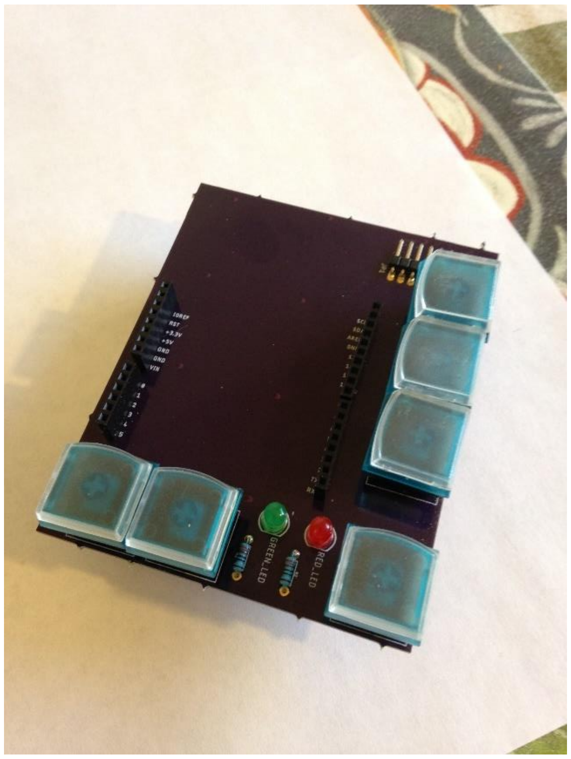


Figure 9

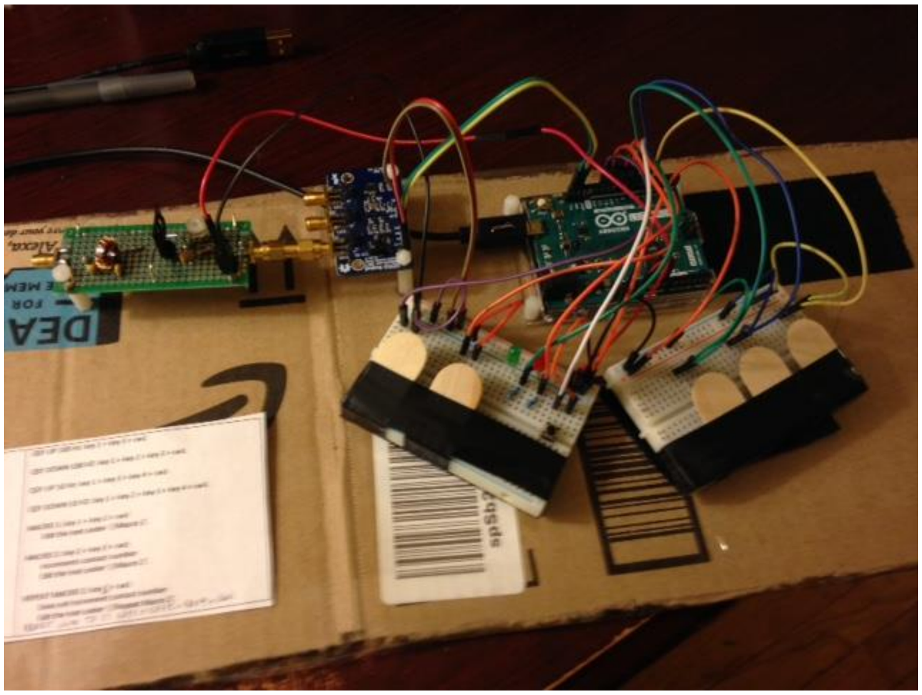


Figure 10

By default, the keyboard operates at approximately 2.1 baud, or about 5 WPM. As you master the Baudot code, you will want to try higher speeds. You may cycle through speeds of 5, 10, 15, and 20 WPM using the Speed Button. Keeping up with the "cadence tap" quickly becomes a challenge. Welcome to the world of telegraphy in the late 19th and early 20th centuries!

Using the keyboard to transmit directly I call "naked mode." All of my blemishes show, mistakes and pauses while I try to recall the Baudot code. Remember those painful pauses during your first few CW contacts? I don't suggest using this keyboard in "naked mode" for contesting, or trying to work rare DX that way, at least not right away. Those pursuits call for lightning fast exchanges. Instead, try “Search and Pounce” contesting, using the built-in contest macros, activated by keystrokes shown in Figure 11. I had a blast working the CQ WPX RTTY and the NA QSO PARTY contests that way. There are also QSO macros. Try opening the Arduino IDE’s serial monitor, and use either the Baudot keyboard or your PC keyboard to enter text into the buffer. Then key the “TRANSMIT BUFFERED TEXT” macro, and hit enter.

key5	key4	sw1	key1	key2	key3	Macros
		GENERAL MACROS				
		x	x			XMIT ON
		x		x		XMIT OFF
		x				SPEED
	x	x				BEACON
		VFO MACROS				
x	x	x	x		x	QSY UP 1000 HZ
x	x	x	x	x	x	QSY DOWN 1000 HZ
	x	x	x		x	QSY UP 100 HZ
	x	x	x	x	x	QSY DOWN 100 HZ
		x	x		x	QSY UP 10 HZ
		x	x	x	x	QSY DOWN 10 HZ
x	x	x				CHANGE BAND
	x	x	x			ENTER FQY FROM PC
		CONTEST MACROS				
		x	x	x		CONTEST MACRO 1
		x		x	x	CONTEST MACRO 2
		x			x	REPEAT MACRO 2
	x	x	x	x		RESET NUM

key5	key4	sw1	key1	key2	key3	Macros
		QSO MACROS				
x	x	x	x			2X2 CQ
x	x	x	x	x		3X3 CQ
	x	x			x	QSO
x		x			x	AGN
x	x	x		x		QRZ
	x	x		x		BUFFERED TEXT
	x	x		x	x	73
		EXTRA MACROS				
x		x	x			MEM1
x		x		x		MEM2
x		x	x		x	MEM3
x		x	x	x	x	MEM4
x		x		x	x	MEM5
x		x	x	x		MEM6
x	x	x		x	x	MEM7
x	x	x			x	MEM8

FIGURE 11

Or forget the whole QRP transmitter thing, and use the Baudot keyboard to type directly into the MMTTY buffer, and use that program to transmit with your commercial transceiver.

I suggest building this keyboard and working your friends using “naked mode” during a time of day when your favorite band is relatively inactive. Demonstrate the keyboard for them. Then twist their arms into learning a few characters (their own call sign and name) and nag them into getting on the air with you. It could be a fun club project, under the heading of “Historical Curiosities of Telegraphy.” That might make Gauss, Weber, and Baudot smile.

If you hear a slow CQ on 20m or 40m RTTY, it might be someone using a Baudot keyboard! It’s not because they don’t know about macros. It’s because they’re channeling a British Post telegrapher from 100 years ago. “I say old bean, after we wear away our fingertips here, shall we pop into the pub for a pint?”