Manual and Description of IZ4KBS's version of DL4YHF's QRP-PIC-Keyer

Speed Control

CW Speed is controlled by the **E** and **T** commands (see **Command Mode** further down), and the selected speed also applies during playback. The speed range is about from 6 to 45 words per minute (WPM), adjustable in 31 steps. A word about speed adjustment is in order here. In the original keyer by DL4YHF the keying speed could be adjusted by means of an external pot-based control. The adjustment was based on measuring the voltage across a capacitor in an RC circuit, with R being the pot. Now, the discharge in such a circuit is non linear but rather it decays asymptotically over time, so the speed adjustment wasn't linear either. In the modified keyer I removed the pot speed control in favour of an all-software, paddle-based control, but I decided to retain the original non-linear behaviour, partly because if was already there but also because it is more convenient in my view. The keyer defaults to about 15 WPM at power-up, and it takes about 17 taps on the **DOT** paddle (**E** command) to take it down to the minimum speed of 6 WPM, while it takes only about 14 taps on the **DASH** paddle (**T** command) to get from the default 15 WPM up to the top speed of 45 WPM. If the control were linear, i.e. one WPM of increment at a time, it would take as many as 30 taps to get from 15 to 45 WPM, with a level of “gradation” that I believe would be somewhat inconvenient at high speeds and do more harm than good. Said differently, one tap on the **DASH** paddle while at 20 WPM will get you say at 21 WPM, while one tap while at 42 WPM will make the keyer jump to 45 WPM. I believe this is convenient, as I see very little point in being able to adjust high speeds 1 WPM at a time, as if one could really tell the difference between 40 and 41 WPM. On the same note, retaining a higher level of gradation below about 20 WPM makes possibly more sense to better suit novice operators. I know all this is subjective, but this is how it works now anyway :-) Please note that the above speed values are referred to a power supply voltage of 3.6 Vdc, and they may vary slightly if the keyer is supplied with a different voltage, within the allowed range of 3-5 Vdc. I personally power my unit with 3 x 1.2V LR44 button cells, which are expected to last a very long time, almost shelf-life, given the minimal current requirements of the keyer, unless of course you add your own bells and whistles, like a LED indicator and so on. If you do add stuff, then 3 x 1.2-1.5 AAA batteries may be more appropriate.

Buttons

Two buttons control recording and playback of messages.

Button **Message 1** controls a message of about 63 characters stored in the PIC's internal EEPROM. This message will not be deleted if you disconnect the battery.

Button **Message 2** controls a message of about 55 characters stored in the PIC's internal RAM. This message will be deleted if the battery is disconnected. There is normally no need to disconnect the battery and thus no power switch.

If both buttons are pressed simultaneously the keyer switches to **Command Mode** as explained below. This can more easily be accomplished by providing a third dedicated button, along the lines of the Four State QRP Group Keyer, which schematic is available on the Web at the following link at the date of this writing:
http://www.wa0itp.com/aa0zzkeyerschematicv3.pdf

Playing a stored message

A short press of a **Message** button initiates the corresponding message. If a message is "partitioned" (see below), use repeated short presses of a button to select the part of the message you want to play.

Recording a message

To record a new message press the corresponding button for about one-half second or until the keyer signals **M**, which means **Message Record**.

To stop recording a message tap the message button a second time. The keyer will signal **S** for **Stored**.

If the keyer runs out of memory during recording, it signals **F** (**Full**) and stops recording.

This QRP-Keyer also measures the gaps between WORDS. The "Pause-Code" and the length of the pause are recorded as an individual character in memory and, therefore, occupies one memory location. The Pause-Length is recorded as a "multiple" of a dot-

length, so if you replay a message with a higher speed than when recording it, "long" pauses will be shorter.

Partitioning a Message

Instead of one long message you may also record multiple shorter messages in a single message memory. I call this a partitioned message.

To separate all sub-messages in a memory you have to insert a special character called **End Of Message** (**EOM**). All messages are recorded as one long message, with an EOM character at the end of each sub-message. The **EOM** character is entered into the memory by running together the letters **E+O+M** (".-----").

Playing a segment of a Partitioned Message

To recall (play) the second part of a recorded message press the **Message** button two times, with only a short delay between to button "clicks."

To play the third sub-message press the button three times and so on.

If you press a message button quickly more often than there are sub-messages, the keyer will not play anything at all.

When playing a recorded message, the keyer will stop when it reaches an **EOM** character in the message memory, with one exception: in the **Memory-List** mode all codes stored in a memory are played "without conversion."

In **Memory-list** mode you will also hear the recorded **EOM** characters. More on **List Mode** further down.

Sidetone and Signal-tone

You may connect a small (passive) piezo speaker to pin 2 of the processor (this pin is called RA3 or Audio Out).

The piezo speaker will generate a sidetone for CW transmission and a signal-tone to

indicate special conditions, or "feedback," that may be useful for some operations.

During normal telegraph operation the sidetone may be useful if you want to build the keyer into your homebrew QRP transceiver.

In **Command Mode**, for *Warnings* and other *Signals* the keyer will generate a lower pitched SIGNAL TONE. This simplifies the keyer's operation, especially if you often use the keyer's special functions. The signal tone produces Morse code at a FIXED speed; not the operator's preset operating speed, but 12-16 words per minute.

If you want to use the sidetone, you may find a suitable piezo speaker in an old electronic birthday greeting card.

If you don't want to use a piezo because you have a transceiver with built-in sidetone, you may use optical signals from a signal LED. Connect a low current LED from Pin 8 (RB2 or Signal LED) via a resistor to ground. This LED will only be driven by the keyer, when a SIGNAL TONE is generated. Therefore the LED will NOT be on during normal CW transmission.

Command Mode

Special commands to the keyer are entered in **Command Mode** with the paddles. To enter **Command Mode**, press both buttons simultaneously, or add a third button as previously explained. The keyer will answer with the signal-tone **C** (**Command Mode**).

As long as the command-mode is active, the keyer tries to interpret all characters you enter with your paddles as a command. Commands usually consist of a single Morse letter.

Most valid commands are acknowledged with the signal-tone **R** (**Roger**), commands that are not recognized are answered with a question mark "**?**" (..--..).
For convenience, the commands **E**, **T** and **U** (see below) are not confirmed with an **R**.

You may exit **Command Mode** either by a second simultaneous press of both message

buttons or by entering the **D** command (**Done**) with the paddles.

The following commands are recognized:

Command A: Turn **DOT/DASH memory OFF**. With the DOT/DASH-memory turned off the keyer behaves like this: If you release the paddles while a dot or dash is being sent, the dot or dash will be completed, and nothing else is sent. In other words: If you tap the DOT, while a dash is being sent, and release the DOT before the transmission of the dash is over, no dot be sent after the dash. This is called Iambic mode A.

Command B: Turn **DOT/DASH memory ON** (BETTER Mode). This is my preferred keying mode, where the keyer stores the "opposite" element while transmitting one element. For example, if you first tap the DASH and while the dash is being sent tap the DOT, the keyer will send the DOT after the DASH is finished (no matter if you release the DOT before the end of the DASH transmission). This is called Iambic mode B.

Command C: **BeaCon Mode** (Endless replay without time limit.)
The **BeaCon Mode** (implemented in early March 2000) is similar to "endless loop" mode, but the **BeaCon Mode** does not have the 255-repeats-limit. Only beacons (or ARDF-transmitters) may use **BeaCon Mode**. To transmit repeated (almost "endless") CQ loops, **Loop Mode** is preferred, see **F** command below.

Command D: **Done** Exits from **Command Mode** and returns to normal keyer operation.
Command E: Decreases CW speed. If it gets below the minimum allowed it wraps-around to the maximum speed.

Command F: **Play Forever**, switches the memory playback mode to an (almost) endless loop. This allows playing a stored message (almost) endlessly, while the operator sits in the background drinking a cold beer. After 255 repeats the keyer terminates the "endless" replay loop; which prevents real "endless" transmissions, if the operator forgets to stop the transmission.
Switching to endless loop mode does not automatically begin playing a message, you still have to start playing by pressing the message button.

To terminate the endless loop mode, enter a second **F** command.

To terminate playing a message ("endless" or "normal") just tap either paddle.

Tip: Append a long pause at the end of your CQ call message, before you stop recording the CQ call; this provides additional listening time.

If someone answers your call, tap the paddle and the CQ loop stops.

BeaCon Mode (implemented in early March 2000) is pretty much like endless loop mode, but without the 255-repeats limit. See description of the **C** command above.

Command L: **List Mode**, used to check the content of a complete message buffer with special functions.

If you play a recorded message in **List Mode**, there will be no conversion of special codes like **EOM**, **NNN** and **ANN**.

Return from **List Mode** to normal operation with the **M** command.

Command M: **Macro Mode,** if this message playback mode is active, special characters like **EOM**, **NNN** and **ANN** are treated in a special way.

The sequence **NNN** is expanded into three digits. This is the default mode, it is complementary to **List Mode** (see **L** command above).

Command N: Set **Number** for contest-operation; see chapter Contest Operation. This command is used to initialize the contest serial number to any value between 000 and 999. The keyer answers with the **Signal Tone** “NR” to tell you it expects the entry of a three-digit number using the paddles.
Following the **N** command you must enter three digits, after inputting the third digit the keyer signals "R."

Command Q: **Quick Digits**, switches the digit output mode to quick digit mode.
In this mode the keyer will generate digits as follows:

Digit	0	1	2	3	4	5	6	7	8	9
Output	T	A	U	V	4	5	6	7	8	N

You may always enter numbers using "normal" digits or "quick" digits, no matter if the quick digit mode is active or not. The keyer understands both formats as it waits for numeric input.

Note: Quick digit mode does not apply to normal recorded numbers. At the time of this writing this only applies to the expansion of the macro "NNN."

Command S: **Standard Digits** or **Slow Digits**; switches the digit output mode back to the standard mode. In this mode all digits generated by the keyer are played as ordinary five-element Morse code.

Command T: Increases CW speed. If it gets above the maximum allowed it wraps-around to the minimum speed.

Command U: **tUne**, turns the transmitter continuously ON for about 30 seconds. To stop tuning tap either paddle or any message button. If you stop tuning manually, the keyer will also exit from command mode. If the keyer stops tuning after the 30-second timeout, it will NOT exit from command mode. You may then start a new 30-second tune interval with another **U**.

Contest-Operation
In combination with the two message memories you may use the keyer as a contest keyer with automatic contest number generation. All you need is a little “smart programming" of both message memories in combination with two macros you record along with your contest messages.

All you have to do during the contest is to repeat the other station's callsign and recall the proper message from memory.

Beside contests, this feature may also be useful in other situations.

The following macros are used to generate numbers and to increment a counter:

Macro NNN: Number Number Number**,** Is entered into a memory as "-.-.-." .
Function: when playing a message, this macro will be expanded into a three-digit contest number. This function can be called as often as needed, it does NOT affect the contest counter!

Macro ANN: Advance Number, Is entered into a memory as ".--.-." .

Another way to remember this macro is:
".--.-." = **PN** = **Plus Number** or
".--.-." = **AC** = **Add Counter**

Function: Increments the current contest number by one. It does NOT generate a transmitted character when played from a message memory.

You will only see this macro in a message memory when you activate **List Mode, L Command** (see **Command Mode** above).

As you can see, the macros **NNN** and **ANN** have to be combined in a contest. The message you use to give a contest report, will use the **NNN** macro (possibly multiple times). An example for a very simple contest report: "599/<NNN> 599/<NNN> BK <EOM>"

When playing this message, the keyer replaces the <NNN> by the current contest number. For example, it will transmit 599/123 599/123 BK

After a contest QSO is complete, you will play a different message from another message memory (or at least from another partition of a message memory), which contains the macro **ANN**.

It is OK to use a short message like: "73 gl <ANN> qrz ?"

When playing this message the keyer only transmits this: 73 GL QRZ ? because the macro <**ANN**> only increments the contest number, but does not transmit a character.

Next time you play the contest report from the first example, the keyer will transmit "599/124 599/124 BK."

Technical Information

Processor: PIC16F628A: Data sheet and development system available at www.microchip.com.

Power-saving sleep-modes with "wake-up" on change of any input signal on Port B
Power consumption (supply voltage = 3.6 V): - operational, no sidetone active: 60 microamperes
Operational, with piezo-sidetone beeping: 200 microamperes
During standby: less than 1 microampere (typical)
The keyer enters standby-mode automatically after a number of seconds of "no activity."

Revision history:
07/1999: First release of keyer hardware, firmware and manuals.
03/2000: Implemented **Beacon** mode (useful also for ARDF transmitters)
08/2011: Modified Four State QRP Group EZ-Keyer hardware and added paddle-based speed control

Last Revision: 2011-09-03 15:21z