

PS2B Features

- Keyer speed range: 5 - 99 WPM
- Automatic touch sensitivity calibration
- Dynamically allocated message memory
- Keying Modes: Bug, Ultimatic, Iambic A or B
- Speed control Potentiometer with on/off switch
- Adjustable Weighting: 25 to 75 %
- Adjustable Keying Compensation: 0 to 31 mSec
- Paddle swap command
- Continuously adjustable Sidetone frequency
 - Four Push-button user interface
- Two User Configurations each with callsign
- Low Power Consumption: 10 mA idle
- HSCW and QRSS support
- Non-Volatile Message Memory: 240 letters in 12 Slots/dual banked with embedded commands.
- Multiple Message stacking
- Integrated Touch Paddle
- 39 easy to use commands
- Supply Voltage Monitor
- Serial Number Generation
- Adjustable Letter Spacing: 25 to 75%
- Automatic letter space mode (Autospace)
- Beacon: Programmable interval: 1 to 99 seconds
- Sidetone Output: On board mini-speaker
- Key/PTT Output: Solid state relay (350V @120mA)
- Nine volt battery holder
- Backspace supported on message entry
- Dual RCA isolated output jacks
- Rx and Tx Practice Modes
- External auxiliary paddle input
- Weighted Base

PS2B Description

The PS2B is a dual function device that can act as a simple touch paddle or a full function iambic keyer with touch paddle input. Optimal touch paddle sensitivity is maintained by an auto-calibration algorithm. The PS2B is implemented in a Microchip PIC16F1825 microcontroller and utilizes a special version of the K1EL K16 keyer core which provides a wide range of features. A simple command configures the device in paddle or keyer mode. In keyer mode, setup commands are directly entered on the paddles in Morse code. All settings and messages are stored in nonvolatile memory so that they are preserved when the keyer is turned off. The K16 keyer core has many original features not found in other keyers:

• Supply Voltage Monitoring

The PS2B has a unique feature that is useful in battery powered applications. It can accurately monitor its supply voltage and provide an indication when battery voltage is low. Normally the PS2B will respond with an **R** when entering command mode. If the battery voltage is approaching the minimum operating limit, it will respond with an **L** instead. This tells the operator that the batteries need to be replaced. The actual supply voltage can be read out in Morse by using the **V** command in the extended command set. (Page 24). Note that a voltage regulator maintains the keyer supply voltage at five volts until the power supply drops below approx. 5.5 volts.

• Dual User Configuration

The PS2B provides storage for two complete configuration setups. For example, one setup could be used for contesting while the other for casual operating. Or, when two operators share the same keyer, each user can have their own setup profile. The extended command **U** is provided to select one configuration or the other. All PS2B settings are included in each profile including a stored callsign. (Page 24)

• Dual Message Banks

The PS2B has two message banks of six slots plus two callsign slots. Total message storage is 240 bytes. While this does not seem like a lot of message space, since the PS2B dynamic allocates message storage in memory, it turns out to be more than adequate for most users. It is very easy to swap message banks with the **E** command. This is the shortest command sequence and allows you to swap banks quickly. There is also a buffered message command **/E** that swaps message banks within a message. (Page 19)

• Stored Callsigns

Two special memory slots are provided to store a callsign for two users. These callsigns are programmed by using the callsign load command in the extended command set. A callsign is embedded in a message by using the **/M** buffered command. The callsign slot works like any other message slot, you can call other messages and embed commands, (Page 24)

• Wide range of embedded message commands

Please refer to the list on page 28.

- **Sidetone Frequency**

The PS2B sidetone can be set to any frequency between 300 Hz and 2000Hz. (Z command, page 23)

- **Fast Message Interruption**

The PS2B will stop a message immediately upon paddle press, stopping in mid-letter if need be.

- **Practice Mode**

Both send and receive practice are included. The user can select practice content by letter group so that easier letters can be mastered first followed by progressively more difficult groups. A very good random letter generator is provided which generates a varied, ever changing letter order. (Page 21)

- **Message Stacking**

Up to 10 messages can be queued to be sent in the order requested.

- **Simplified Beacon Formatting**

For example, this is all that is required to setup a repeating 15 second beacon: **/B15 K1EL BCON**

- **Easy Beacon**

Any message slot can be turned into a beacon without having to add the special **/B** embedded command. The **B** command allows 'on the fly' beaoning without embedded commands. (Page 19)

- **PTT Lead In and Tail Settings**

The PS2B adopts the Winkeyer scheme for PTT control. Both the lead in delay and tail delay can be specified in milliseconds as well as speed dependent hang delay for paddle operation. (Pages 20 and 22)

- **Dit/Dah Ratio Control**

The timing of dits vs. dahs can be customized. A ratio of 1:3 is standard but this can be altered to suit different tastes. (Page 23)

- **Improved Cut Number Selection**

The PS2B allows serial number cuts to be used for 0, 9, both, or neither. (Page 24)

- **Tuning Duty Cycle Selection**

Tune can be set to generate either a 50% or 100% key down duty cycle. (Page 19)

- **Contest Word Spacing**

The PS2B allows a shorter inter-word spacing to be selected. This is intended to speed up exchanges during contests. Standard word spacing is 7 dits while contest mode word spacing is 6 dits. (Page 23)

- **Keyer Lock**

The PS2B can be locked by command and it will stay locked until the command push button is pressed for eight seconds. When locked, the PS2B will ignore any input. (Page 29)

- **Fixed Speed Setting**

The PS2B supports both a variable speed control with a fixed "favorite" speed setting. It is very easy to switch between them. The fixed speed setting is set by the **S** command. (Page 21)

- **Fast Speed Change**

Fixed speed setting can be incremented or decremented by holding the command pushbutton and then tapping the paddles left or right. (Page 25)

- **Speed Pot Range Setting**

The upper and lower limit of the speed pot is set with a new extended command; **R** for range. Two values are entered the lowest speed pot setting followed by the highest speed pot setting. (Page 24)

- **First Element Extension**

This is another command that is brought over from the Winkeyer command set. It allows the first dit or dah of a transmission to be elongated to allow for receive to transmit relay delay. (Page 23)

• Command Response Time Adjustment

The PS2B will enter command mode when the command pushbutton is pressed for about 2 seconds. This may be too long for some operators. A new extended command **F** has been added that allows the delay to be shortened to about 1.3 seconds. (Page 23)

• Full Time Speed Pot

The PS2B responds to speed pot changes without delay, even while sending messages.

Paddle Stick Functional Block Diagram

Figure 1 is a block diagram of the PaddleStick PS2B. The PS2B can be configured to act as a simple touch paddle or a powerful, self-contained, Morse keyer. The electronics are divided between two PCBs; the PaddleStick and the PSbase. As shown, there are four message pushbuttons, a sidetone speaker, plus two solid state relays that can key practically any transmitter. The two outputs can be configured as a Key/PTT pair or as two separate Key outputs allowing keying of two radios. One of the push buttons is dual purpose; press and hold to enter commands, or a quick press to play a message. Keyer paddles are integrated into the PaddleStick circuit board which allows very compact layout. A weighted base with rubber feet provides a solid, stable base.

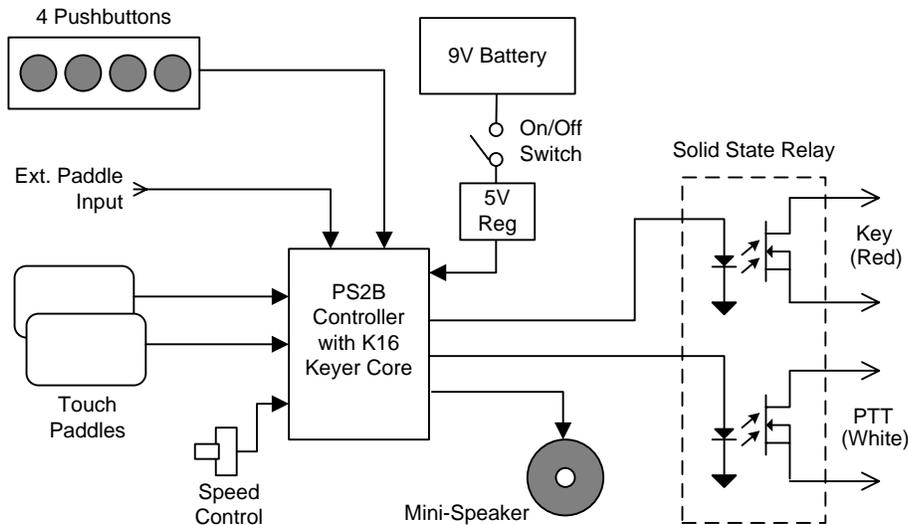


Figure 1 – PaddleStick PS2B Block Diagram

Touch Paddle Adjustment

The PS2B scans and adjusts paddle sensitivity automatically based on current conditions, however, due to process variations in PIC microcontrollers, a baseline calibration must be run once when the kit is complete.

PaddleStick PS2B Kit

The PS2B keyer kit consists of two printed circuit boards and a weighted base. We will cover the kit assembly in three portions; the PaddleStick PCB, the PSbase PCB, and the weighted base kit. Please read all of the assembly instructions before starting and be sure to follow the steps in the order presented for best results. The order in which the steps are called out is very important. Even if you are an experienced builder, we highly recommend reviewing the appendices A, B, and C at the end of this document for helpful assembly hints.

Please Note! Small board mounted components are packaged by board, one section is for the paddle stick board and another section for the PS base board. Open one section at a time and don't mix them up, this will make component identification much easier.

PaddleStick Kit Assembly

We will start with the PaddleStick kit. This consists of a PS2B IC, sidetone speaker, touch paddles, and connector. This board is carried by the PSbase PCB which has keying output drivers, voltage regulator, speed pot interface, a multiple pushbutton interface and a 9 volt battery.

PaddleStick Parts Inventory

U1	- PS2B 14 pin DIP IC (16F1825)
R1	- 2.2K Ω 1/8 watt (red red red)
R2,3	- 470 Ω 1/8 watt (yellow violet brown)
R4	- 47 Ω 1% 1/8 watt (yellow violet black)
C1	- .1 μ F ceramic capacitor 104
C2	- .001 uF ceramic capacitor 102

Q1	- PN2222A NPN transistor
SP1	- Mini speaker
J1	- 8 position right angle header
1 pc	- 14 Pin DIP Socket
1 pc	- PaddleStick Rev C PC board

Paddle Stick PCB Assembly

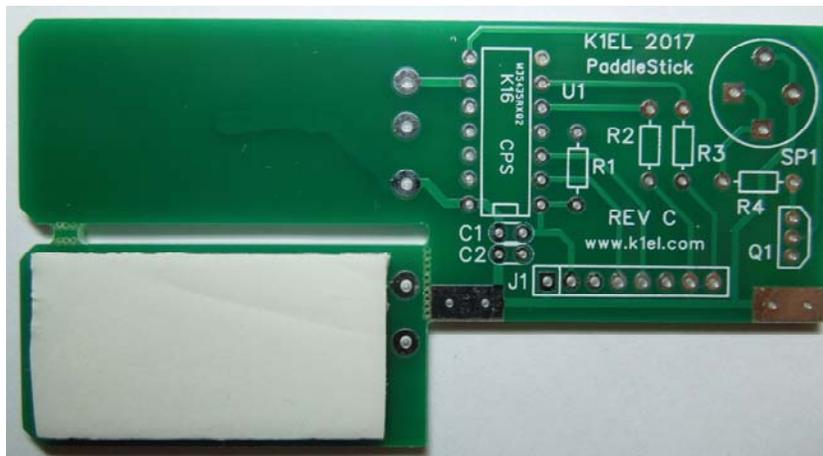


Figure 2 - PaddleStick PC Board before assembly

First step is to separate the paddle half from the main board. Use a pair of wire cutters to cut through the paddle separator holes.

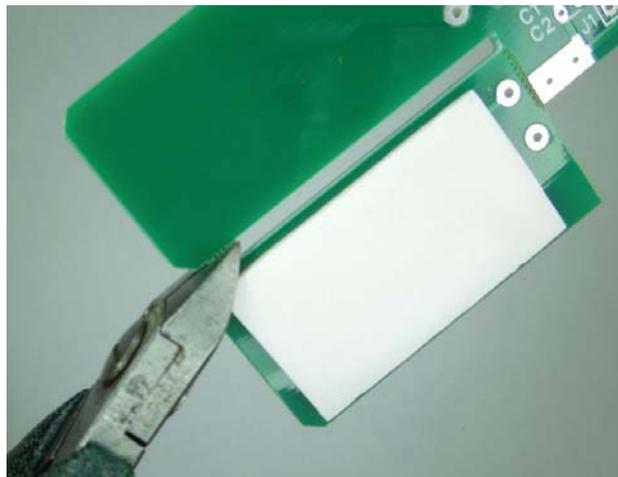


Figure 3 – Cut paddle separator

Bend the paddle section to break it free. Trim off separator remnant and use sandpaper to remove burs.

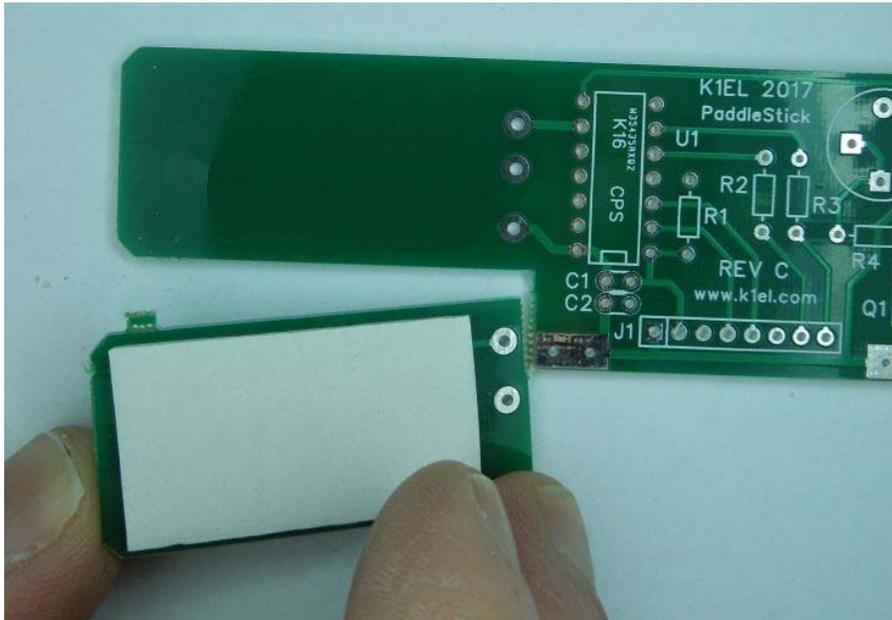


Figure 4 – Free paddle piece and then trim off separator

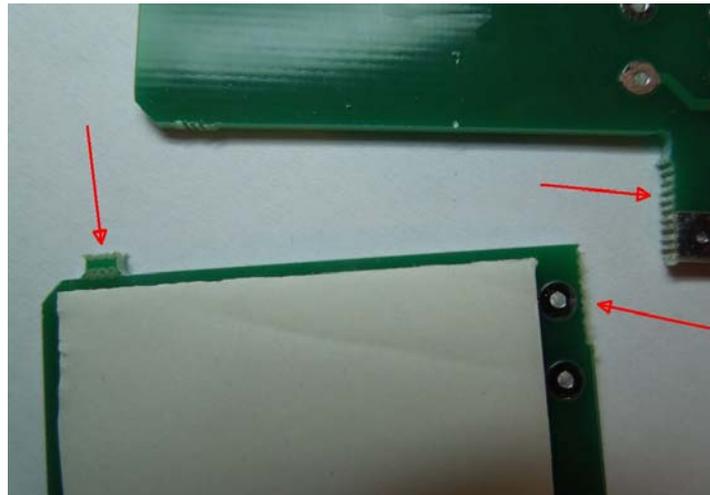


Figure 5 – Sand rough edges smooth and trim off separator remnant

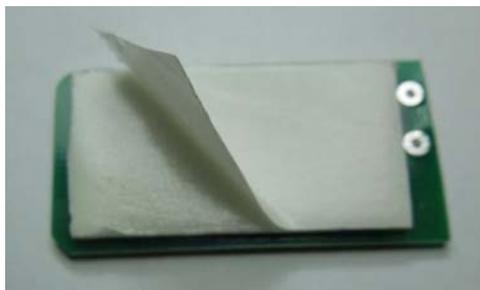


Figure 6 – Remove foam tape paper film

Carefully assemble halves in place as shown, the touch plate surfaces face outside. Make sure that the via holes line up between the two boards and paddle edges align closely. Once the halves are sandwiched together it is nearly impossible to separate them.

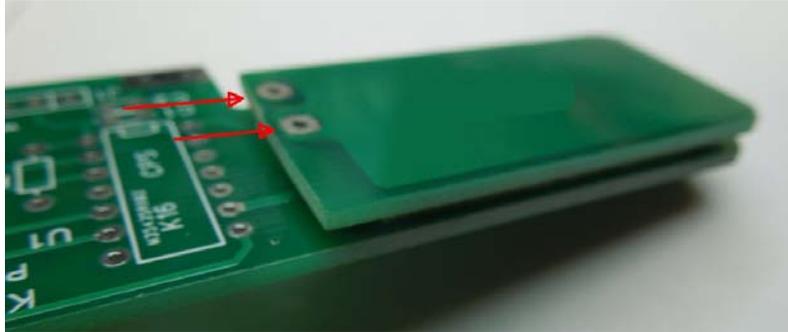


Figure 7 – Assemble paddle halves

Prop up one side of the board. Cut a ½ inch piece of resistor lead off of two resistors in the kit and insert the leads in the holes as shown below. Solder the leads on both boards to electrically connect the two boards together. Trim leads on both sides and fill in the unused via with solder as shown in figure 9.

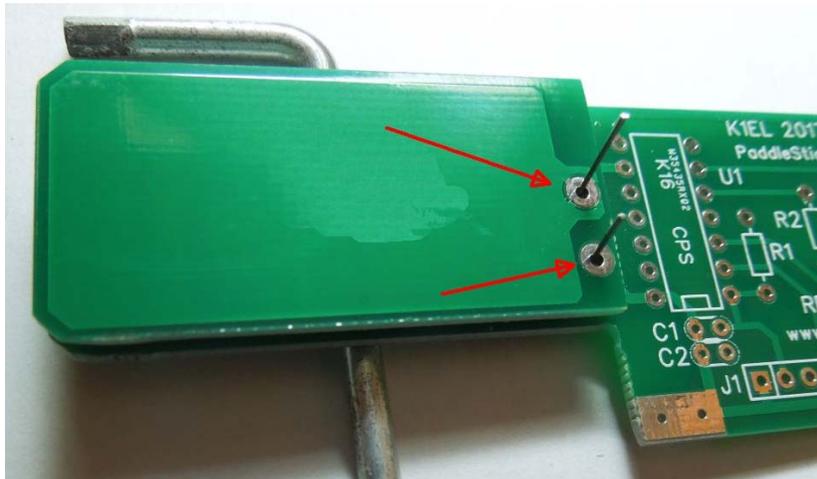


Figure 8 – Insert resistor lead scraps through holes in both boards

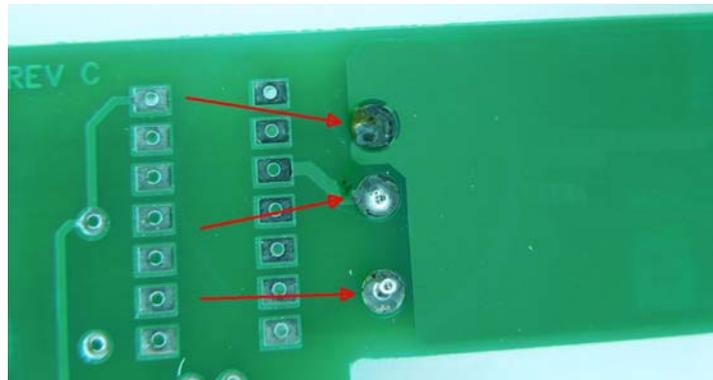


Figure 9 – Don't forget to solder leads on main board side, fill in the third via as shown

Now it's time to install components on the PaddleStick board. Following is a picture of the parts that are on the board to help identify them. The .1uF cap (C1) may be blue instead of the orange color shown.

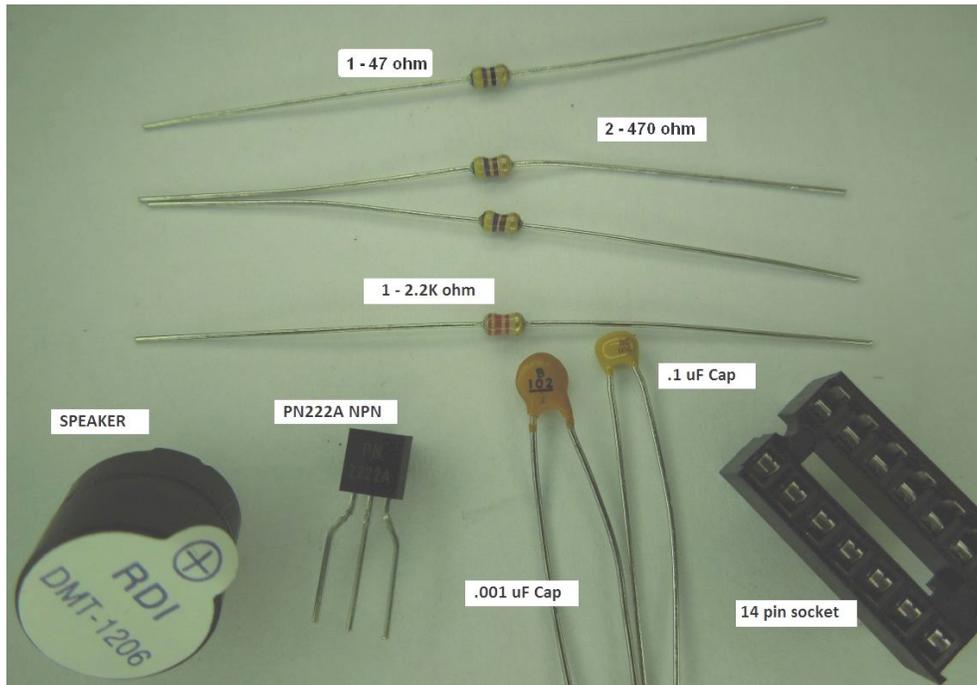


Figure 10 - PaddleStick board components

Solder the 14 pin DIP socket in place, aligning the notched end as indicated by red arrow. Install and solder the four resistors, their color codes are listed in the parts inventory on page 4. We recommend verifying resistor values with an ohmmeter. Now install and solder the two capacitors, be sure to not mix up C1 (104) and C2 (102). Install transistor Q1 being sure to orient it so that the flat side lines up with the silkscreen as shown below. Trim leads after soldering.

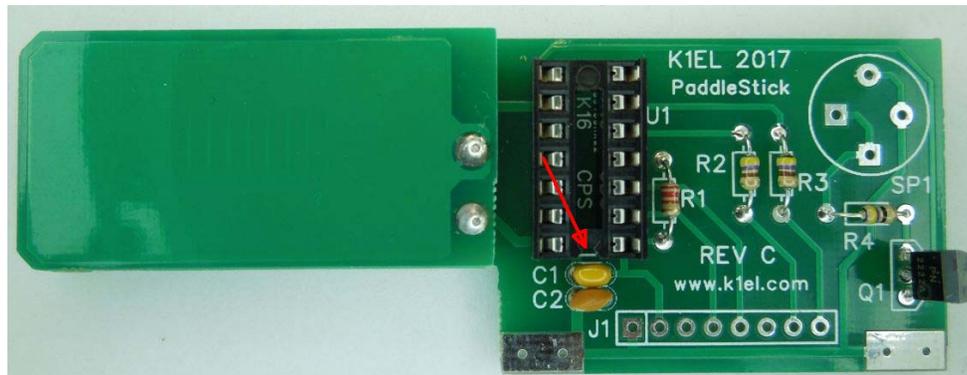


Figure 11 – Install socket, R's, C's, and Q1

It is very important to follow the next steps accurately; if the header connector is not attached correctly it will compromise the entire PS2B assembly. To start, insert the header into place as shown. Before soldering make sure the connector is fully seated into the PC board (see figure 13). Once you are happy with the placement, solder one center pin and then double check that the connector is straight. If it is not, readjust and then solder the remaining 7 pins.

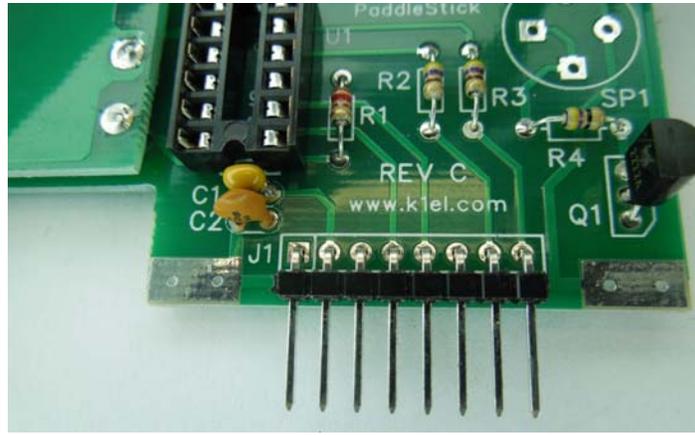


Figure 12 – 8 pin header in place

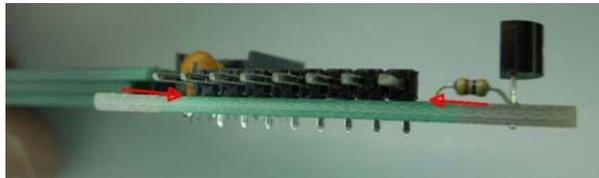


Figure 13 – Header must be flush to the PC board

Now install the speaker, you will see four mounting holes but only one pair will accept the speaker lead spacing. Orient the speaker so that the plus side goes into the square hole. Remove the white protective film from the speaker after soldering it in place. Lastly install U1 noting that pin 1 (the one closest to the circular indentation) goes to the lower right. It's easier to install ICs if the leads are straight, carefully straighten them as shown in the left picture. The silver dot on U1 signifies that the part has been programmed.

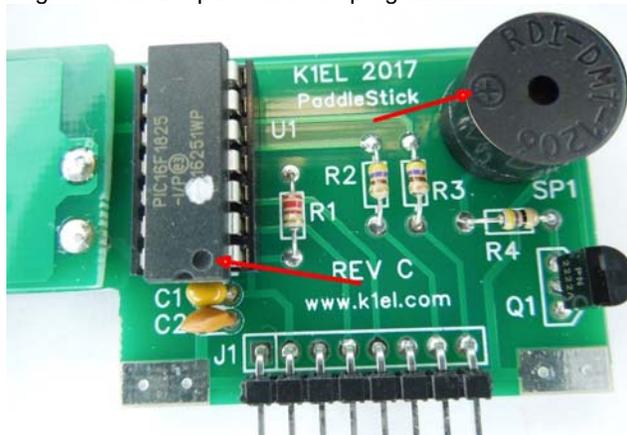
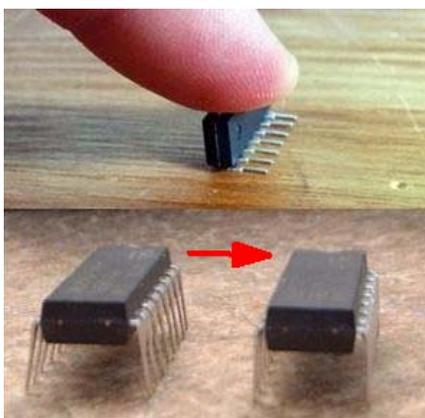


Figure 14 – Straighten leads on U1, then install speaker, transistor Q1, and U1.

Before continuing carefully inspect your work. Make sure that all solder connections have been made and that there are no solder shorts between pads. Double check that Q1 and U1 are installed correctly and that all parts were installed. This completes the PaddleStick assembly, now it's on to the PSbase.

PSbase PCB Kit

PSbase Parts Inventory

U1	–	AQW210 or 280 solid state relay
VR1	–	TL750L05 TO92 5.0V regulator
R1-4	–	470Ω 1/8 watt (yellow violet brown)
C1,2	–	.001 uF ceramic capacitor 102
C3	–	4.7 uF electrolytic capacitor
C4,5	–	.1 μF ceramic capacitor 104
S1-4	–	Pushbutton switches
1 pc	–	PSbase PC board
2 pcs	–	Double sided foam tape squares

1 pc	–	Speed pot/power switch
1 pc	–	Battery holder
J1	–	1/8" stereo jack
J2	–	Two way RCA jack
HW1	–	Five 1/8" 4-40 spacers
HW2	–	Five 4-40 by 1/4" screws
HW3	–	Metal weighted base
HW4	–	Four rubber feet
SK1	–	One DIP8 socket

PSbase PCB Assembly

The first step is to attach the PaddleStick (PS) to the PSbase (PSB). The goal is to mount the PS board so that it is perpendicular to the PSB board and that the bottom of the PS mates as close to the PSB as possible. If this goal is met, the upcoming solder bonding task will be much easier. If you attached the header to the PS board correctly, the PS board will sit perfectly between the four bonding pads on the PSbase. We start by inserting the PS into the PSB as shown below:

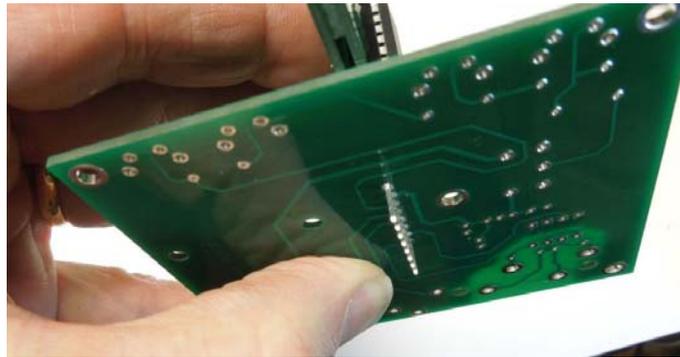


Figure 15 – First step in PS to PSB assembly

Hold the PS firmly in place while you solder one of the middle pins of the connector. Now inspect your work to be sure the board is perpendicular, use a business card to verify this:

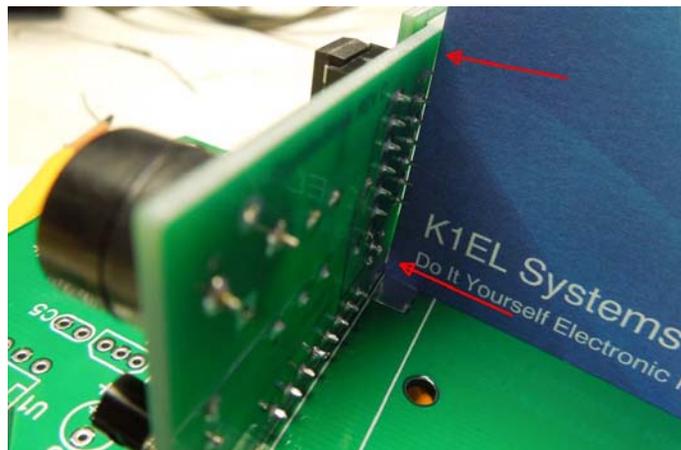


Figure 16 – Verify that the PS is at a right angle to the PSB

Since only one pin is soldered it is possible to bend the PS slightly to correct for misalignment. If it is off too far you may have to reflow the solder connection. Make sure the gap between the PS and PSB is at a minimum since we will be bridging the two boards with solder in the next step. When you are satisfied with your work, solder the remaining 7 pins and trim off the excess leads.

Now we will bond the two boards together with solder bridging. We will flow solder into the areas indicated by the red arrows on both PS and PSB sides. It works best if you place your soldering iron so that it contacts a pad set on both boards, this helps solder flow to both pads at once. It's been reported that pre-tinning (adding a thin layer of solder to) the eight rectangular pads will help the bonding process.

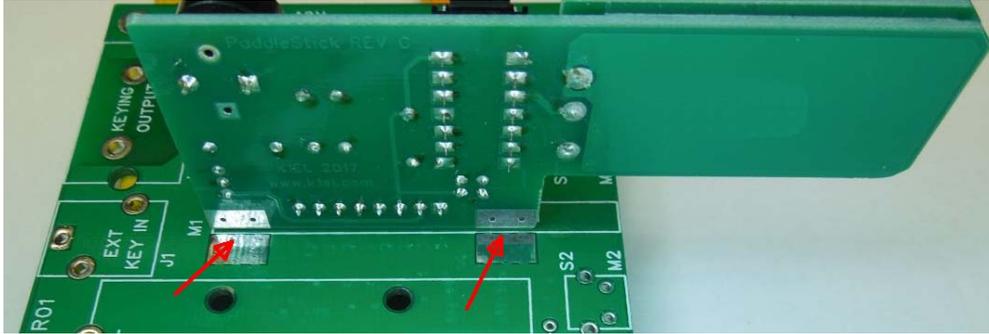


Figure 17 – Red arrow indicate solder bonding zones

Figure 18 is an example of a poor bridging job, it will not have sufficient structural strength. We want nice full, shiny fills as shown in figure 19.

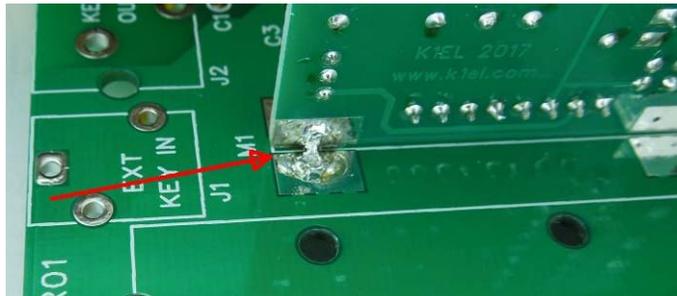


Figure 18 – Poor bridging attempt



Figure 19 – Good bridging, this will be very strong

After all four pad pairs have been bridged we will move on and install components on the PSbase board.

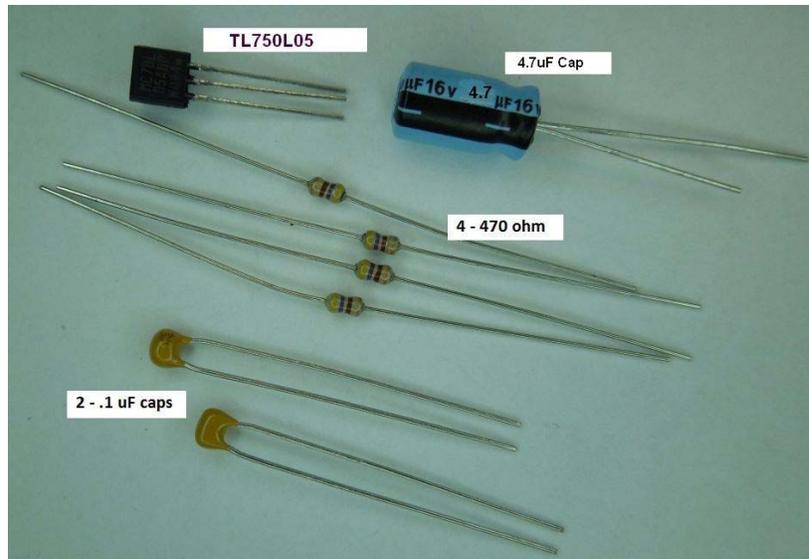


Figure 20 – PSbase components, phase 1

Note that the two .1uF caps may be blue instead of the orange color shown. Don't get these confused with the .001 uF caps which have .2" lead spacing. We start by installing the four 470 ohm resistors. Since they are all the same value, we do not have to worry about mixing them up. It's a still a good idea to check each one with an ohmmeter to be sure the right parts were supplied with your kit. Now install the voltage regulator VR1, align the flat side of the regulator with the flat side of the silkscreen. Next install two .1 uF ceramic capacitors. Finally install the 4.7 uF electrolytic capacitor, this part is polarized and has to be installed correctly. The easiest way to do this is to put the long lead into the square hole. Use Figure 22 as a guide.

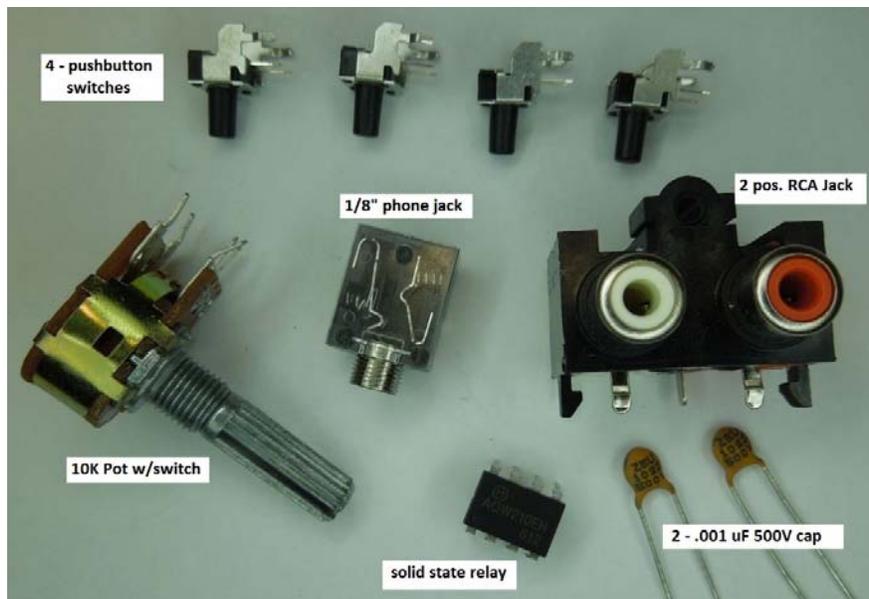


Figure 21 – PSbase components, phase 2 (DIP8 socket not shown)

Install all four pushbutton switches; make sure they are fully seated into the PSB board before soldering. Next install the two .001uF ceramic capacitors which may be blue in color, they have a lead spacing of .2" Now install the 8 pin socket for U1, the solid state relay. Then place U1 into its socket, making sure that pin 1 of U1 is oriented correctly. The red arrow in Figure 22 indicates where pin 1 goes. The last part in this group is the external paddle connector; make sure it is fully seated on the board before soldering.

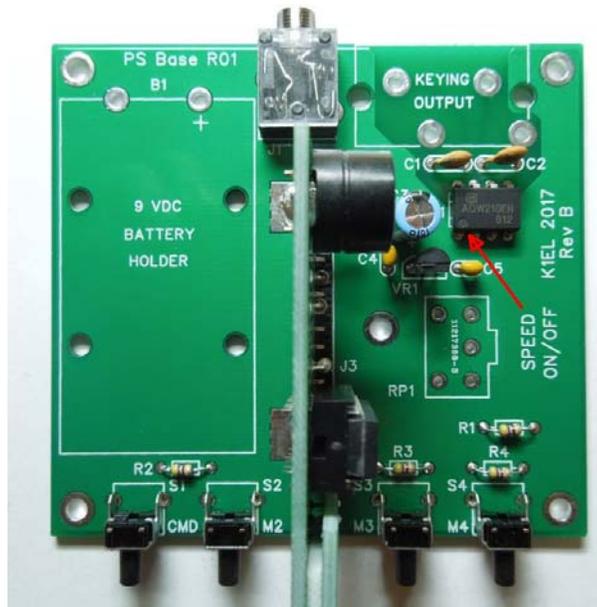


Figure 22 – PSbase components installed

There are only three components left, start with the RCA keying output connector. This part snaps into place but takes a bit of care to get it to fit right. Here's a hint, make sure that all the connector pins thread into their respective holes before trying to snap the connector in place. Now install the speed pot/power switch, like other parts, make sure it is flush to the PSB board. You will probably have to straighten out the five leads before it will go into the board, a pair of long nose pliers is good for this.

Now we have to do a little prep work on the battery holder. Remove the paper film from one side of the double sided foam tape squares and attach them to the bottom of the battery holder as shown below:

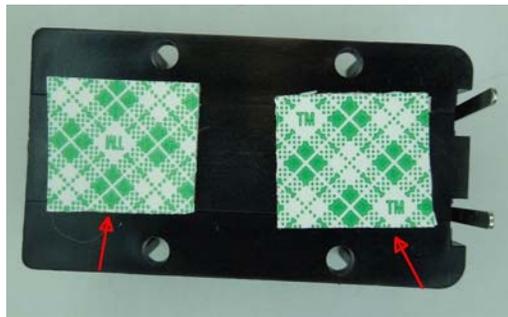


Figure 23 – Foam tape in place on battery holder

Now remove the paper film from the top of the pads and thread the leads into the PSB then board. Carefully align the battery holder with the PSB board silkscreen and press the battery holder in place. Solder the two leads and trim them. Figure 24 shows the completed PS + PSbase assembly.

Before mounting the assembly on to the metal base, run a few quick tests. Make sure that the speed pot/power switch is in the off position, which is fully counter clockwise. Install a nine volt battery into the battery holder by aligning the snaps and pressing it in place. Now turn on the power switch and you should hear an "R" in sidetone. Check that the touch paddles work, dits one way dahs the other. Turning the speed pot should change the Morse sending speed. Press the left most pushbutton until the keyer responds with an "R", release the pushbutton, and after a few seconds the keyer should respond with a "?" A quick press on any of the four pushbuttons will cause the keyer to send "MT", this signifies that all four message slots are empty. That is all the testing we need to do now. If the keyer did not respond with the initial "R" check to make sure that you didn't miss any solder connections on the PSbase. Next check that you have 5 volts between K16 pins 1 and 14. Also make sure that VR1 and C3 are installed properly.

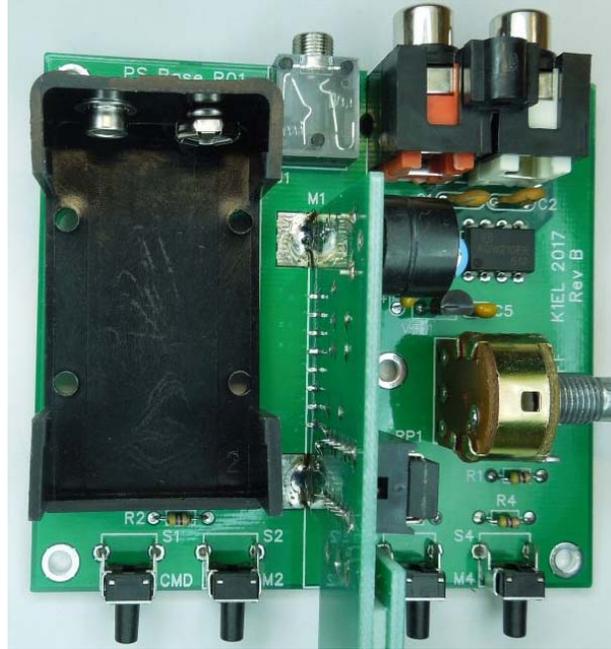


Figure 24 – Completed PSB board assembly

Locate the four rubber feet and install them on the bottom of the base as shown below in figure 26

Figure 25 – Rubber Feet



Figure 26 – Rubber feet installed on bottom of metal bottom

Mounting the PSbase PC board on the metal base is a little tricky. Follow these steps and you will do ok. First put the metal base on a flat surface and place the five spacers over each hole (fig 27).

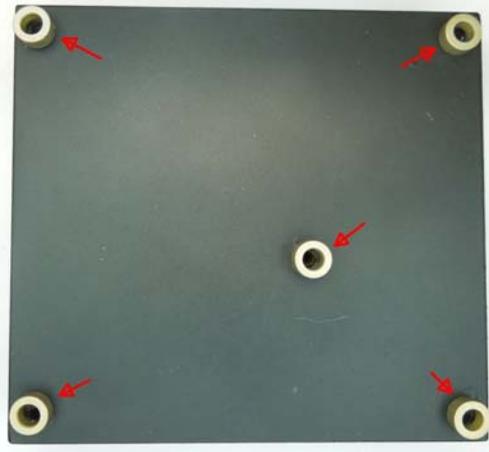


Figure 27 – Spacers placed on top of metal base

Now place one of the 4-40 screws into the center hole of the PSbase. You may need a pair of tweezers or long nose pliers to get the screw in the hole. Now hold the PSbase in one hand and hold the screw in place with a Philips screwdriver. Guide the PSbase down on to the metal base putting the screw through the center spacer and into the threaded hole. Thread the screw a few turns into the metal base. Now carefully install the other four screws in the corners. You might have to shimmy the spacers around to line things up. After all screws have been threaded, tighten them up.

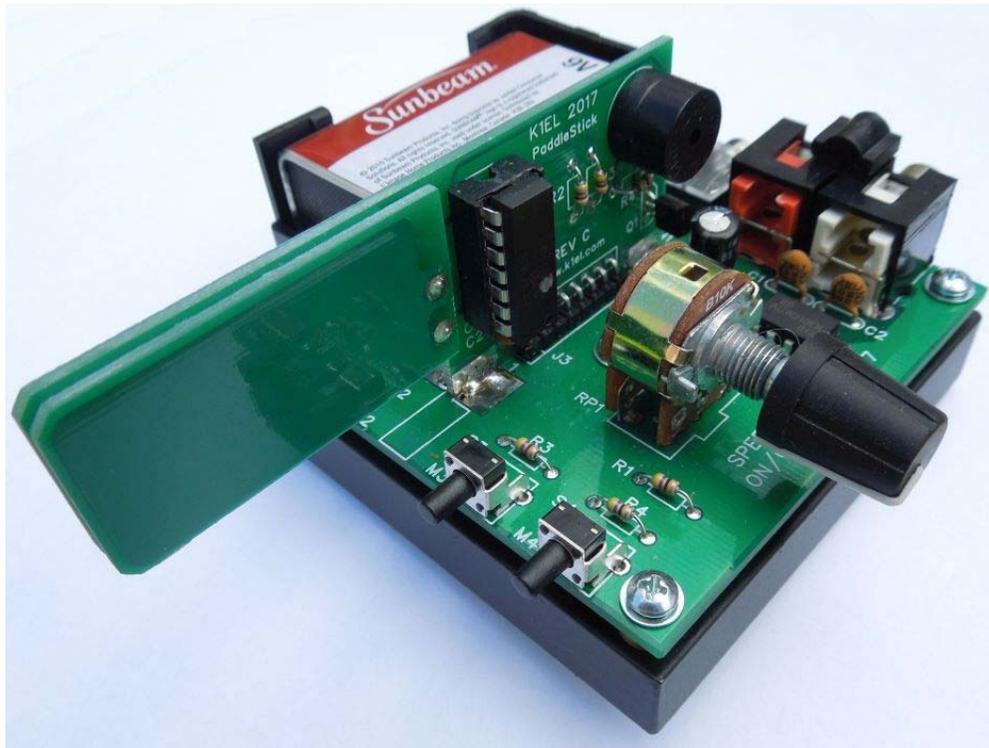


Figure 28 – Completed PS2B

Test Procedure

I suspect you have already started your testing before completing the assembly. Now is the time to go through all of the functions. Let's start by turning the PS2B on. We need to run a baseline calibration to compensate for process variations in the PIC microcontroller. Here is the procedure to do this:

1. Turn PS2B off
2. Press and hold the command PB while you turn the power on.
3. The keyer will send an AR followed by an L, release the cmd PB.
4. Press the left paddle as you normally would and wait for the keyer to send an R
5. Then press the right paddle until the keyer sends an SK followed by a normal startup R.

After the start up "R" press and hold the left most command button until the keyer responds with an R. Release the button and after a few seconds the keyer will respond with a "?" A quick press on all four pushbuttons will cause the keyer to send an MT which signifies that the message slot is eMpTy. Paddle presses will cause dits and dahs to be sent. If you have an RCA cable, plug one end into the red jack and monitor the other end with an ohmmeter. When you press the paddle you should see the meter move with Morse letters. Now move to the white jack and you will see the meter go to a low reading during the time the keyer is actively sending. If you have an iambic paddle set that has an 1/8" connector, plug it into the external key connector and check out the external paddle functionality.

How to connect your PS2B

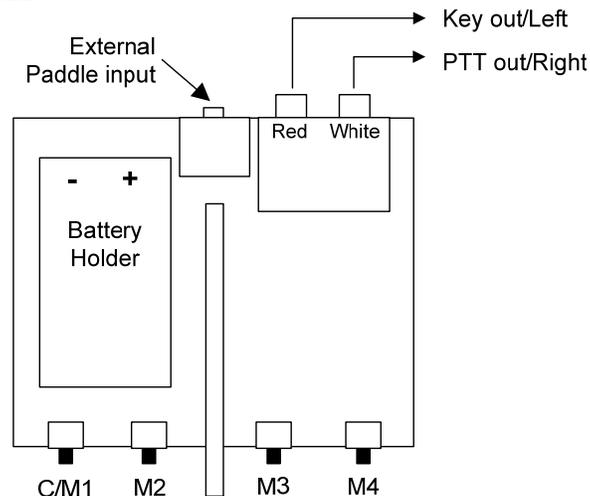


Figure 28 - PaddleStick Connection Diagram

There really isn't too much involved in hooking your PS2B up for use. The RCA jacks are outputs, their function depends on how the keyer is configured. Following is a matrix showing the possibilities:

Configuration	Red RCA Jack	White RCA Jack
Paddle Mode	On when left paddle is pressed	On when right paddle is pressed
Single Output Keyer	Keying Output	PTT Output
Dual Output Keyer	Key output port 1	Key output port 2

The Key and PTT outputs are open collector which means they act like a switch to ground. The outputs are optically isolated from the keyer circuitry for safety. They can safely switch up to 350 V plus or minus and are rated for 120 milliamps max current. Practically any transmitter can be keyed with these outputs. Vacuum tube grid block keying is not usually a problem since the keying voltage is usually -100V or less. Always confirm the keying voltage before connecting PS2B up. Cathode keying is not recommended since the voltages can be much higher especially during keying transients. In addition, exceeding the maximum current rating is another consideration.

Since outputs are open collector, you can wire a straight key across the output. This allows you to quickly switch between the two input keys.

Other safety considerations

The two output connectors are electrically isolated from the keyer's circuitry and they are also isolated from each other. For safety's sake, do not touch or probe the output connectors while connected to a transmitter.

ESD Considerations

Capacitive sensing pads are integrated into the PS2B printed circuit board. While these inputs have ESD protection inside the PIC, the safety margin can be increased by applying a layer of plastic tape to the paddles. You may have to re-run the baseline calibration to compensate for this physical change to the paddles.

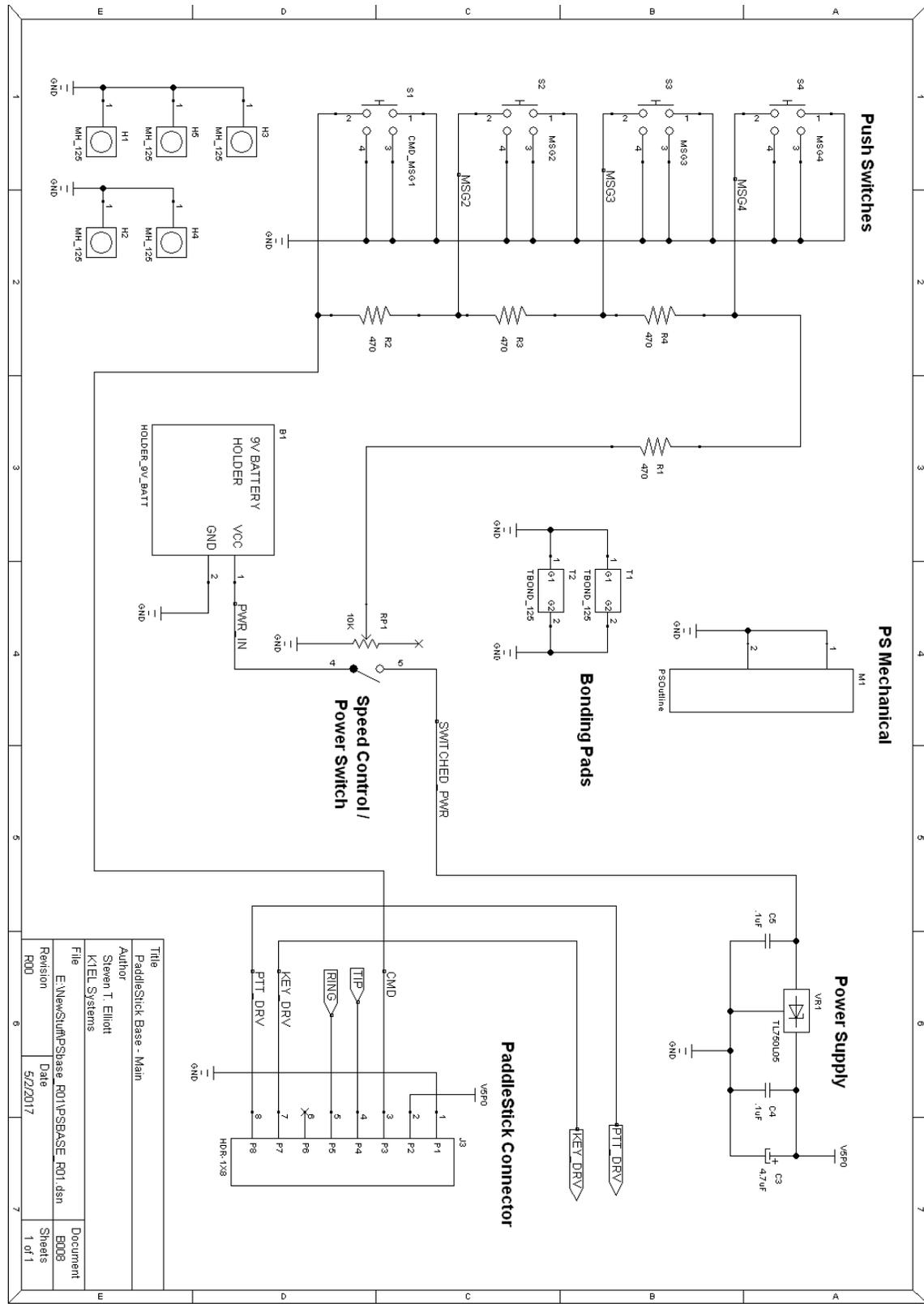


Figure 30 – PSbase Schematic Page 1

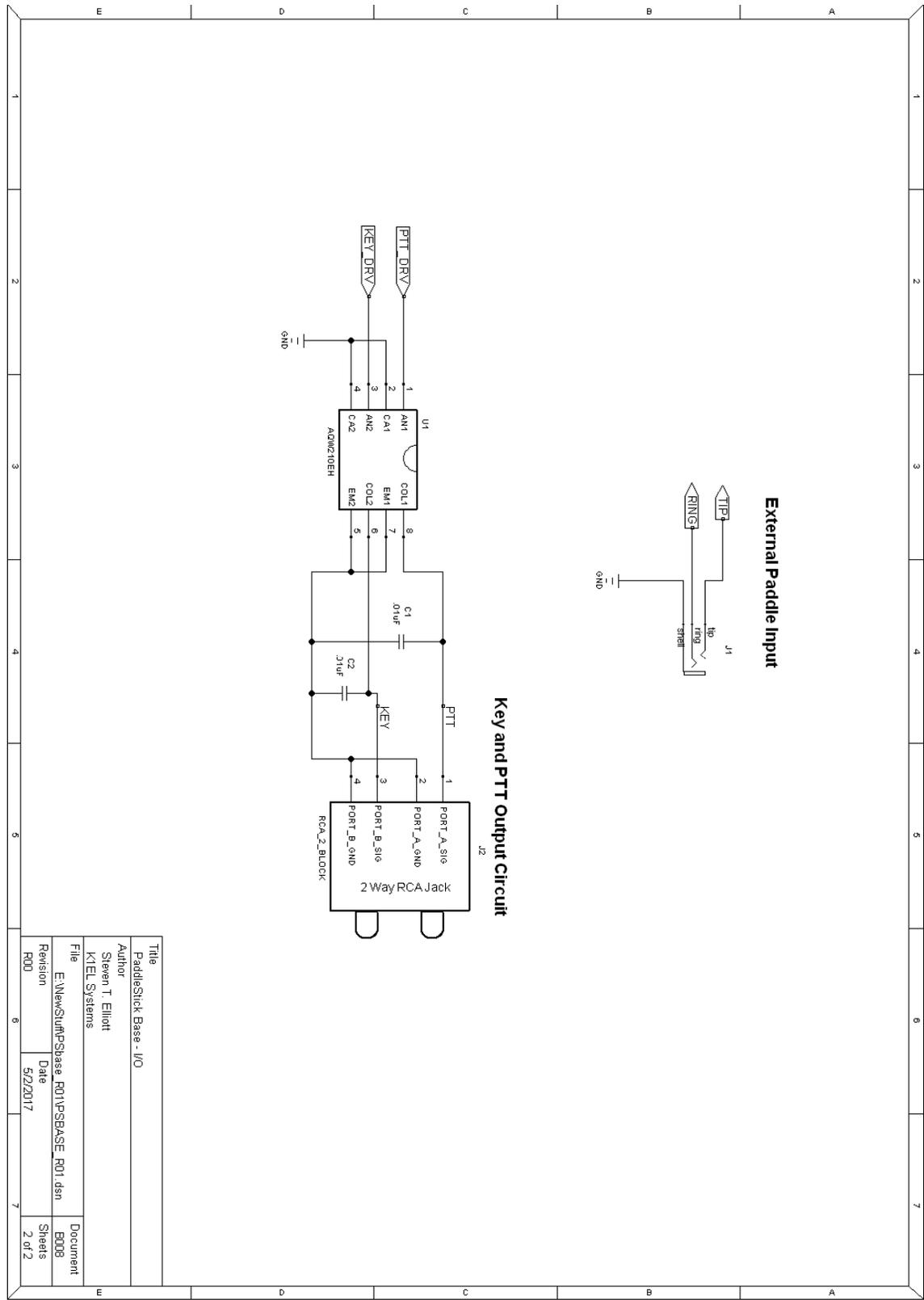


Figure 31 – PSbase Schematic Page 2

Operating the PS2B Keyer

The PS2B incorporates the K16 keyer core so it shares the same functionality as the K16. Presented here is the K16 documentation for reference.

Command Mode

Changes to the PS2B's configuration are made by entering values on the paddles. Before doing this, the PS2B must be put into command mode.

If the command push-button is pressed and held, the K16 will respond after about two seconds with the letter **R** in sidetone. This means the K16 is ready to accept a command. Simply enter the command letter in Morse on the paddles and the command will be executed. Some commands require additional parameters. In this case, the PS2B will prompt you with the letter **E** (for enter). When the PS2B is in command mode, the KEY and PTT outputs are de-activated. All commands provide some sort of feedback to tell you if the command was understood and executed properly. If an illegal command or parameter is entered, the PS2B will respond with a question mark.

Important Note ! In command mode, transmitter keying is disabled and replies are sent in sidetone only. If sidetone had been disabled with the **A** command, it will be re-enabled temporarily during command mode.

Command Toggles – Many settings such as Paddle Swap and Transmit Mute have just two states, on or off. These settings are turned on or off like a toggle switch. Issuing the command will change the state back and forth between off and on. When a toggle command is turned off, the PS2B will echo an **N** for “Not enabled” Likewise when a toggle command is turned on an **A** is echoed for en**A**bled.

K16 Command List

In the following command descriptions, the letter displayed in **BOLD** is the command and **BOLD ITALIC** is PS2B response. [**pb**] means that the PS2B will wait for you to press one of the message pushbuttons.

A - Sidetone enable is a toggle command. Sidetone should be disabled when using a transceiver's built in sidetone. The PS2B will acknowledge this command by responding with an **A or N**. Note: If sidetone is disabled, it will be temporarily re-enabled when entering command mode.

B [n,dd] - Easy Beacon: Any message in the current bank can be repeated at a specified rate. After entering **B**, the user is prompted to enter a single digit message number **n** (1-6) and then prompted to enter a two digit beacon cycle time **dd** in seconds (01-99). A beacon can be interrupted by hitting either paddle. The beacon is timed from start of beacon to start of next beacon. For example if you set a delay time of 10 seconds, the beacon will start every 10 seconds regardless how long the message is. If the message is longer than delay time, then there will be no gap between messages.

C [nn] – Command WPM: The PS2B uses different speeds for command transactions and keyed transmit. Changes in transmit speed will not affect command speed. After the **C** command is issued enter the speed **nn** in WPM (05-99). If the speed is valid the PS2B responds with an **R**, otherwise a **?** See the **S** command for details on setting transmit Morse speed.

D - Decrement serial number by 1, PS2B responds with an **E**

E - Swap message banks: The PS2B has two separate message banks, six messages each. In response to this command, the PS2B will respond with an **E** for bank one or an **I** for bank two.

F [nn] - Set Farnsworth Speed: This is used primarily for code practice. Letters are sent at the Farnsworth speed while maintaining the default code speed. For example, if Farnsworth is set to 25 WPM and the operating speed is set to 7 WPM, individual letters will be sent at 25 WPM while the spacing between letters remains at a 7 WPM rate. To disable Farnsworth mode, set it to zero.

G - Toggle 50% tune duty cycle: The default tune duty cycle is 100% key down. This command allows either 100% or 50% duty cycle to be selected. Some folks prefer a 50% tuning duty cycle to reduce power dissipation thereby protecting transmitter finals and/or a linear amplifier. An **A** is echoed when 50% is enabled, an **N** is echoed when set to 100%

H [n] - Set Transmit PTT Hang Delay Time: Sets a PTT delay that is proportional to sending speed. The delay begins after paddle sending stops. You can select one of four delays:

HangTime = 0: wait 1 wordspace + 1 dit before ending PTT
 HangTime = 1: wait 1 wordspace + 2 dits before ending PTT
 HangTime = 2: wait 1 wordspace + 4 dits before ending PTT
 HangTime = 3: wait 1 wordspace + 8 dits before ending PTT

After entering the command letter, you will be prompted with an **E** to enter the desired hang time as a number 0 to 3 as indicated in the table. Hang delay is different than PTT tail delay in that it is proportional to code speed while PTT delay is a primarily a fixed delay time. This means you don't have to change the PTT delay every time you change sending speed.

I [nn] - Set Letterspace Adjustment: **nn** is a value 0 to 31 that specifies an additional letterspace delay to be applied between letters. Multiply **nn** by two to arrive at the actual adjustment percentage. For example a value of 7 applies 14% additional letterspace between letters. The maximum adjustment is 62%.

J [nn] - Paddle Sample Delay: normally the PS2B waits one dit time after a paddle press has been sensed before latching a second paddle press. The **J** command allows this delay time to be adjusted longer or shorter than one dit. If the delay is set too short, the keyer may send unwanted dits or dahs, if there is too much delay it can make sending a bit more tedious. From the formula provided below, delay time can be set to be greater or smaller than one dit time. The default value is 50 which specifies one dit time. Value of 99 doubles the delay time while a value of 25 halves the delay. A value of 55-60 gives a more relaxed feel. **If the paddle sensitivity is set to zero, both dit and dah paddle memories are disabled.** The delay, which tracks sending speed, is calculated with this formula:

$DELAY_TIME = (nn \times DIT_TIME) / 50$ where Switchpoint is a value between 01 and 99.

K - Set Keying Mode: There are six different keying modes supported by the PS2B: Iambic mode A, Iambic mode B, Straight Key/Bug, Ultimatic, Dit priority mode, and Dah priority mode. In either Iambic mode, alternating dits and dahs are sent while both paddles are held closed. In mode B an extra alternate dit or dah is sent after both paddles are released. In straight key/bug mode a dah paddle press will key the transmitter for as long as the paddle is pressed and dits will be generated automatically when the dit paddle is pressed. In Ultimatic mode when both paddles are pressed the keyer will send a continuous stream of *whichever paddle was last pressed. Hold dah then press dit->sends dits. Hold dit then press dah -> dahs are sent.* Ultimatic dit and dah priority mode will generate dits and dahs automatically in response to single paddle presses, but when both paddles are pressed either dit or dah has priority.

After the **K** command is issued the current mode is set by entering a single letter:

Iambic B:	Enter B
Iambic A:	Enter A
Ultimatic:	Enter U
Straight Key:	Enter S (This is also the Bug setting)
Dit Priority:	Enter E (Ultimatic with dits always taking priority when both pressed)
Dah Priority:	Enter T (Ultimatic with dahs always taking priority when both pressed)

L [nn] - Set PTT Lead In Time to a value between 0 and 99 milliseconds. See the Set PTT Lead/Tail description on page 21 for more information.

M – Mute Transmit: This is a toggle command which turns keying output on and off. Mute transmit when you want to use the PS2B as a Code Practice Oscillator (CPO) When muted, the PS2B will send CW in sidetone only. In response to this command, the PS2B will echo an **A** when mute is turned on and an **N** when mute is turned off.

N [nnnn] - Load 4 Digit Serial Number: All four digits must be entered including leading zeroes. The serial number is played by inserting a play message token **/N** into a message. The serial number is automatically incremented after playing. See *Embedded Command* section for more details.

O - Swap Key Output Port: In dual output mode, this command will swap keying outputs. In other words, each time the **O** command is issued, the key port will toggle between key port 1 and key port 2. Command response is one dit for port 1 and two dits for port 2. In single output mode, the K16 will respond with an **X**. Refer to extended command **O** (page 23) to learn about output options. The ability to change keying outputs

if useful if you have two transmitters, it frees you from moving cables around when you want to switch the PS2B keyer from one radio to another.

P [m,d] - Start Practice Mode: A dual mode multi-level code practice program is built into the PS2B. There are two styles of practice, receive only and echo (receive/respond) practice. There are four levels of practice organized by easy to difficult letter groups. The four levels are:

Level 1: E T A N I M W S G D U K O R
 Level 2: C Q P J F B V Y H X Z L including level 1
 Level 3: 1 2 3 4 5 6 7 8 9 0 including level 1 & 2
 Level 4: ? / , . AR SK BT AS including level 1 & 2 & 3

The syntax for entering a practice mode is:

P m d where m is **R** for receive or **E** for echo practice and d is a single digit **1** to **4** to select difficulty level.

P R 2 selects Level 2 receive practice

P E 4 selects Level 4 echo practice

Receive Practice Description: Random characters from the selected level are sent in groups of five. Practice will continue until the command pushbutton is pressed.

Echo Receive/Transmit Practice Description: The PS2B will send a random character from the selected level and you must respond by echoing the character back on the paddles. If you get it right the PS2B will repeat the first character followed by a new character. Now you must echo back both characters. The PS2B will continue to add characters until a set of five characters has been completed successfully. After that it will start over with a new set. If you miss a character the PS2B will respond with an **X** and start over with a new sequence of characters.

When you want to end practice, either respond with di-dah-di-dah or press the command pushbutton. By default practice will be sent on the active output key port, enable transmit mute (**M** command) to inhibit this.

Q - Query PS2B Current Settings: PS2B will respond with current settings sent in the following format:

WPM is sent first
N followed by Serial Number-1
M followed by free msg memory space in letters available
C followed by command WPM
W followed by weight
L followed by lead time
T followed by tail time
E followed by 1st extension (this parameter described in host mode section)
V followed by key compensation
F followed by Farnsworth WPM
I followed by Letterspace adjustment
J followed by Paddle Sample Adjust
Y followed by dit/dah ratio
B followed by speed pot min WPM
T followed by speed pot max WPM
REV followed by K16-PS firmware revision denoted by a single letter; A or B or C... etc.

You can abort this command at any time after the first parameter is sent by pressing the Command and PB4 pushbuttons together or holding either the left or right paddle until the listing stops.

R [pb] - Review a message without transmitting: After the **R** command is entered the PS2B will respond with an **E**. Press the message button of the message you wish to play. The message will be sent in sidetone only. If you try to play an empty slot, the PS2B will respond with **MT**. Embedded commands will be sent without expansion. In other words /S10TEST will be sent as: **DAH-DI-DI-DAH-DIT S10TEST**

S [nn] - Set Fixed Speed in WPM: One value **nn** is entered in WPM (05-99). This speed is activated when the speed pot is turned fully counterclockwise. The speed pot will act normally above this setting. Enter a zero to disable fixed speed which will allow full speed pot range.

T [nn] - Set PTT Tail Time: The PS2B provides a transmitter PTT output that can be used to switch a transmitter or linear amplifier over to transmit mode in advance of actual CW keying. You have control over the time delay between when PTT is asserted and when CW keying will start, this is lead-in. You also have control over how long the transmitter will stay in transmit after keying has stopped this is tail delay. The tail delay is handled differently for CW sent by paddle and CW sent by message. Paddle delay is controlled by the Hang Time setting while message PTT delay is controlled by the Tail setting. The formula to calculate tail time is:

$$\text{Tail Delay} = \text{Three Dit Times} + (\text{Tail Setting} \times 10 \text{ milliseconds})$$

Examples:

At 20 WPM, Tail set to 7, Tail Delay = $(3 \times 60) + (7 \times 10) = 250$ mSec

At 40 WPM, Tail set to 7, Tail Delay = $(3 \times 30) + (7 \times 10) = 160$ mSec

At 20 WPM, Tail set to 0, Tail Delay = $(3 \times 60) + (0 \times 10) = 180$ mSec

At 15 WPM, Tail set to 55, Tail Delay = $(3 \times 80) + (55 \times 10) = 790$ mSec

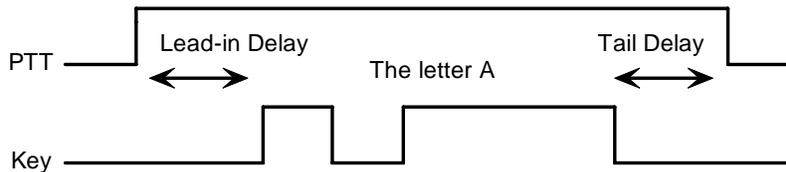


Figure 32 – PTT Lead-in and Tail Example

In general we want a very short tail time when sending messages and we want PTT to stay asserted between letters while sending with a paddle set. That's the reason the delay is dictated by two mechanisms.

U - Toggle Autospace Mode Off and On: When autospace is enabled, the PS2B will automatically force correct inter-letter space between letters. When the **U** command is issued, the PS2B will respond with an **A** for autospace enabled or an **N** for autospace disabled.

Here is how autospace works: If you pause for more than one dit time between a dit or dah PS2B will interpret this as a letter-space and will not allow the next the next dit or dah to be started until the proper letter-space time has been met. The normal letter-space is 3 dits however this can be increased by using the **I** command. PS2B has a paddle event memory so that you can enter dits, dahs, or squeeze both during the inter-letter space and PS2B will send them as they were entered. With a little practice, autospace will help you to send near perfect Morse.

V [nn] - Keying Compensation: specifies a fixed amount of time to be added to the length of all dits and dahs. QSK keying on modern transceivers can cause shortening of these elements which is especially noticeable at high speeds. The PS2B allows the length of the elements to be increased uniformly to compensate for this. The adjustments can be made in one-millisecond steps. The maximum adjustment is 31 mSecs. Key compensation is very similar to Weighting in that any adjustment added to the dits and dahs is subtracted from the spacing so the resulting speed is not changed. The difference between weighting and keying compensation is that compensation is independent of speed, so if 10 mSec of key compensation is selected, 10 mSec will be always be added regardless of speed. So be careful, using large values of keying compensation at high speeds may result in dits and dahs being run together with no spacing at all.

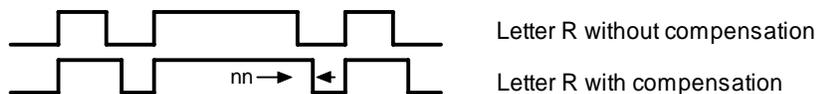


Figure 33 – Key Compensation

W [nn] - Key Weighting: can be adjusted in percentage from 25% to 75%. When set to 50 % the dit time is equal to the inter-element time, which is normal. Values less than 50 reduce weighting while values greater than 50 increase weighting. Note that weighting does not affect sending speed because any increase in keyed time is subtracted from spacing time. Reduction in weighting results in a thinner sound while increased weighting results in a heavier sound. Since weighting tracks speed, a given weighting will sound the same at all speeds.

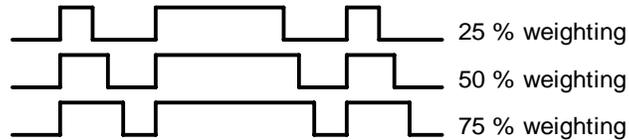


Figure 34 – Key Weighting

X – Extended Keyer Commands

An additional set of commands are located in a sub menu which control settings that not changed very often. Extended commands require two entries, an **X** followed by a sub command. Here is the procedure in detail:

Press and hold the command pushbutton and the PS2B will respond with an **R**
 Enter an **X** and the PS2B will respond with an **E** (command request)
 Enter desired Extended Command with additional parameters, if required.

A list of extended commands is provided in the next section.

Y [nn] - Set Dit/Dah Ratio: nn ranges from 33 to 66. Entering **Y 50** sets the standard 1:3 ratio. For example a value of **33** selects a dit/dah ratio of 1:2 while a value of **66** selects 1:4. The ratio formula is:

$$\text{Ratio of } 1:N \text{ where } N = (nn * 3) / 50 \text{ example } nn = 40 \text{ give a ratio of } 1:((40*3)/50) = 1:2.4$$

Z - Change Sidetone Frequency: After this command is entered the sidetone oscillator will be keyed at a steady rate with transmit muted. Pressing the paddles will raise or lower the frequency. The range varies nearly continuously from 300 Hz at the low end to 2000Hz on the high end. It takes a while to sweep through the whole range. Pressing the command pushbutton will end this command and store the new sidetone frequency. Like all settings, use the Preserve Settings extended command to save the new sidetone setting in EEPROM.

Extended Command List

An additional set of commands are located in a sub menu which control settings that not changed very often. Extended commands require two entries, an **X** followed by a sub command.

A – Pushbutton Input Diagnostic: is used to verify the correct operation of the switch network connected to PS2B input pin 7. This design allows four switches and a potentiometer to share a single pin on the PS2B. The state of the network generates a unique numeric value which is then handled appropriately, either as a pushbutton press or speed pot change.

After the Extended **A** command is issued, the numeric network state will be sent in Morse sidetone. For proper operation the input values must fall within the following ranges:

Command PB	0 to 11
PB 2	12 to 31
PB 3	32 to 57
PB 4 5	8 to 89
Speed Pot	90 to 205 (approx)

The PS2B will remain in this diagnostic state until power is cycled.

C - Toggle Contest Spacing: When contest spacing is enabled, word space is set to 6 dit times as opposed to the normal 7 dit times. An **A** is sent when CT space is enabled, an **N** is sent when it is turned off.

E[nn] - First Element Extension: Forces the first dit or dah of a transmission to be elongated to allow for receive to transmit delay. Enter a value between 0 and 99 mSecs for nn.

F - Toggle Fast Command Response Time: Normally the PS2B will enter command mode when the command pushbutton is pressed for 2 seconds. This may be too long for some operators. When fast response is enabled, the delay time is reduced to 1.3 seconds.

M - Load Callsign: Use this command to load a stored callsign. It works just like a message entry and the destination is the currently enabled user slot. Each user slot has one unique callsign. Since the callsign is treated like any other message, embedded commands can be included if desired. The only way to play back a callsign is by embedding the **/M** command in a message.

N - Select Cut Number for number 9: When enabled, an N will be substituted for the number nine when sending a serial number. Command response: an A is sent when enabled, an N is sent when disabled.

O – Set Output Mode: This is used to set the keyer output configuration. After the command is entered the K16 will prompt you for a configuration setting:

Enter **E** to select single output mode with key output locked to port 1 and PTT locked to port 2.

Enter **I** to select dual output mode. Keying can be routed to port 1 or 2 with the **O** command (page 20)

Enter **P** for paddle pass-through mode to use PS2B as a touch paddle only with an external keyer.

If the toggle port command **O** is issued in single output mode, PS2B will respond with an **X** to remind you that it is not allowed. **Known Bug:** When the **I** command is first issued, both keying outputs are enabled at the same time, please use the **O** command to select a single output 1 or 2. In pass-through mode, PS2B's internal keyer is disabled and outputs are asserted in the same manner that the paddles are pressed. Use this feature if you want to use the PSB as a touch paddle only with your radio's internal keyer.

R [nn nn] - Set Speed Pot minimum and maximum: After the **R** is entered the PS2B will prompt for two values, the minimum WPM and the maximum WPM. An error will be flagged if the minimum value entered is greater than the maximum, if the minimum value is less than 5 WPM. Maximum WPM allowed is **99**.

S - Preserve Settings in EEPROM: will respond with a letter **R** to signify that settings were saved successfully.

T - Select Cut Number for Zero: When enabled, a T will be substituted for the number zero when sending a serial number. An **A** response means this cut is enabled, an **N** response means it is disabled.

U - Swap Users: The PS2B provides two complete user configurations, each with a unique callsign. The two message banks are shared between users. This allows a universal set of messages to be created that will work with either user. For example a message like CQ CQ CQ DE /M /M K will send the selected user's callsign. In response to this command a single dit is echoed when user 1 is selected and two dits when user 2 is selected.

V - Voltage Readout: The PS2B will measure the current supply voltage and send it in Morse sidetone. For example a voltage of 4.52 will be sent as **4R52** with the r indicating the decimal point.

X - Toggle paddle swap (Exchange dit/dah inputs): PS2B will respond with a letter **A** to signify when paddle swap is enabled and an **N** when turned off.

PTT Functionality

PS2B's PTT output is normally used to control an accessory device in addition to normal CW transmitter keying. In most cases this device is a linear amplifier but it could be an antenna relay or the PTT input of a transmitter. In all of these cases there are delay requirements that must be met to insure that the accessory device is switched on before transmission begins and is held on until transmission completes. This prevents damage to the accessory device due to hot switching. The PS2B provides three independent PTT delays to meet this requirement; Lead In, Tail, and Hang Delay.

Lead-In delay initiates a keying event. PTT will be asserted first and then, after Lead-In expires, the key output will be asserted. Lead-In can be set to a value from 0 to 990 milliseconds in 10 mSec steps (0 to 99).

Tail Delay specifies the amount of time PTT will be released after Key is released. Like Lead-In delay, it is adjustable from zero to 990 milliseconds in 10 millisecond steps. Tail delay is the sum of two delays, Tail setting times 10 milliseconds plus three dit times.

Tail Delay = Three Dit Times + (Tail Setting *times* 10 milliseconds)
note: one dit time = (1200/WPM) mSec

Examples:

At 20 WPM, Tail set to 7, Tail Delay = $(3 \times 60) + (7 \times 10) = 250$ mSec
 At 40 WPM, Tail set to 7, Tail Delay = $(3 \times 30) + (7 \times 10) = 160$ mSec
 At 20 WPM, Tail set to 0, Tail Delay = $(3 \times 60) + (0 \times 10) = 180$ mSec
 At 15 WPM, Tail set to 55, Tail Delay = $(3 \times 80) + (55 \times 10) = 790$ mSec

In setting tail delay it is desirable to set the delay long enough to prevent hot switching. The PS2B will hold PTT between letters for messages and then PTT will timeout after the last letter of a message is sent.

For paddle sending, most ops want PTT to be held just long enough to prevent drop out between letters. An issue arises in that Tail delay is not solely proportional to sending speed. This is problematic when someone sets a comfortable tail delay at a slow speed and then increases the speed to a much faster rate. At the faster rate, the tail delay will hold too long after keying stops. Alternatively, if a comfortable delay is set at a faster WPM rate, PTT will then drop out between letters at a slower speed. Since it is very time consuming to constantly adjust the tail delay with sending speed, a different delay method is used for paddle sending, namely Hang Time.

Hang Time is adjustable in four steps and is measured only in wordspace and dit times rather than fixed milliseconds. This means that the delay will track sending speed. In other words it will decrease automatically as sending speed increases and vice versa. The four settings for Hang Time are:

HangTime = 0: wait 1 wordspace + 1 dit before ending paddle insertion
 HangTime = 1: wait 1 wordspace + 2 dits before ending paddle insertion
 HangTime = 2: wait 1 wordspace + 4 dits before ending paddle insertion
 HangTime = 3: wait 1 wordspace + 8 dits before ending paddle insertion

To sum up, Tail delay is set to provide just enough delay to prevent hot switching for machine sent CW while hang delay is designed to hold PTT between letters independent of sending speed. PTT will automatically be held between machine sent letters and words but will drop out quickly after a message has been sent as long as there is no trailing wordspace. When entering a message be sure to end the message with the di-dah-di-dah (**AA**) character to prevent the PS2B from automatically appending a wordspace.

Shortcuts and Speed Ups

The PS2B design goal was to streamline command entry, here are some notable improvements:

Fast Command responses – PS2B uses **A** for positive responses instead of an **R** or **Y**

Cut Numbers on Command Entry – When entering numeric values use cut numbers to save time. For example when changing the command speed to 19 WPM, use **C A N** instead of **C 1 9**

These are the command cuts used in the PS2B: T=0, A=1, U=2, V=3, 4=4, 5=5, 6=6, B=7, D=8, N=9

Single digit number entry – If you are entering a parameter that usually requires two digits but you only need to enter one, just enter the single digit and the PS2B will figure out that there is only one. For example instead of entering **T 0 7** you can simply enter **T 7**.

PS2B will respond with an **E** or **I** when swapping users, message banks, or output ports. Think of it as a single dit for 1 and a double dit for 2. This is much faster than responding with a Morse 1 or 2.

Fast Message Entry – Just press the command button till you get an R then simply press the message button you want to load. To stop load mode, either press the command button or enter di-dah-di-dah.

Fast Tune Mode – Press and hold pushbutton 4 and then press the command button to start tune. Press the command button to end tune.

Fast Fixed Speed Change – Press the command button and then press either paddle to increment or decrement the fixed speed setting. The PS2B echoes a single dit after each speed change.

Fast Command Response – If you think the PS2B takes too long to respond to the command button, enable fast command reply mode which halves the delay time. This is the extended command **F**.

Command Prompts – The PS2B will respond to commands that require additional input with an **E**. The **E** stands for “Entry Required” and is fast and efficient. For example if you enter the Weight command **W**, the PS2B will respond with an **E** to let you know it is waiting for the value.

Speed Potentiometer Configuration

Turning the speed control will change the speed and update the WPM rate with minimal lag. The entire sweep of the speed pot is called the speed pot range and it can be modified with the extended **R** command which sets the upper and lower speed limits. This allows you to tailor the speed control to an area that you prefer. The minimum acceptable value for speed bottom is 5 WPM. If you want to set an exact speed, the easiest way to do this is to turn the speed control fully counter clockwise and set the fixed speed to whatever you want using the **S** command. You can determine the current speed control setting with the **Q** (query) command. Turning the speed pot fully counterclockwise will select the fixed “favorite” speed.

Message Functionality

Messages are loaded by holding the command button until PS2B responds with an **R**, and then pressing the message pushbutton of the memory slot you wish to enter. When PS2B is ready to accept a new message it will respond with an **E**. If you wait too long, PS2B will respond with a **?** and you will have to start over. Since only four pushbuttons are provided, messages 5 and 6 are accessed by pressing two pushbuttons in the following sequence after the **R** is echoed:

Press and hold either message button 2 or 3. (for message 5 or 6 respectively)
Press the command pushbutton.
Release both push buttons.

Message 5 and 6 can be played in a similar manner by first pressing and holding pushbutton 2 or 3, then pressing the command pushbutton, and finally releasing both. Don't forget that there are also two separate message banks each containing six slots bringing the total number of unique messages to 12.

After PS2B responds with an **E**, a new message is entered directly on the paddles at a steady rate, making sure to leave proper space between letters. To insert a word space simply pause for longer than a word space and PS2B will respond with an **E** to signify a word space insertion. You can force a wordspace insertion by entering di-di-dah-dah (**IM**). This allows you to put a wordspace at the beginning of a message or insert more than one wordspace in a row. A ½ letterspace pad character can be inserted by entering di-di-dah-dah-dit (**IG**).

If a mistake is made while entering a message, press and hold the command pushbutton and PS2B will backspace through the letters that have been entered. When you reach the position you want, release the button and new letters can then be added starting at that position. If the message memory becomes full while entering a message, PS2B will stop further loading, respond with an **F**, and then return PS2B back to non-command mode. When a new message has been completely entered, press the command pushbutton, or enter di-dah-di-dah (**AA**), and PS2B will respond with an **R** to signify that the message was accepted and stored. There are 232 letters in message memory that can be distributed in any way between 14 message slots. The length of the individual message slots is not fixed. This means, for example, you could have one message of 80 characters, one message with 5 characters, and a third with 10 characters and still have 141 locations left to split among the remaining three slots. Keep in mind that each word space occupies one memory location.

What if you want to insert one of the message controls (**IM**, **IG**, **AA**) into a message ? Simply precede them with a / (**DN**) and it will not be acted on as a control code.

Usually when you end a message, PS2B will append a wordspace before you have a chance to press the command pushbutton. There are cases when you do not want a wordspace at the end, especially if you are using PTT to key an amplifier. This is because PS2B will hold PTT during that added wordspace. The best way around this is to end a message with the di-dah-di-dah (**AA**) control code inserted immediately after the last letter in the message. This will terminate the message without a trailing word space.

If you are having problems loading messages into PS2B, make sure you leave adequate space between letters and are not sending much faster or slower than current command speed. If, for example, you enter an **A** followed by a **T** and end up with a **W**, you are not allowing enough space between letters. It's a fine line though because if you allow too much space PS2B will interpret that as an intentional pause and insert a

word space. Temporarily lowering the command speed (see command **C**) can help while you learn how the process works.

To play a message back, simply press the desired message button, release, and that message will be sent. If you press a pushbutton that does not have a message loaded, the PS2B will respond with an **MT**, short for eMpTy. If you want to review the message without keying the transmitter, use the **R** (review) command. Note that review will ignore control codes and send them as entered. To abort a message, press the command and PB4 pushbuttons together or press and hold one of the paddles and PS2B will stop transmission immediately.

Gap (Extra Space) Insertion

In messages, PS2B interprets the **IG** prosign (di-di-dah-dah-dit) as a ½ dit delay time. The **IG** character can be included in a text string to add extra emphasis to similar sounding sequences. An example is W1OMO, sending it as W1**IG**O**IG**M**IG**O makes it easier to copy. To insert an **IG** prosign in a message without translating it to a gap, enter it as **/IG**.

Word Space Insertion

In messages, PS2B interprets the **IM** prosign (di-di-dah-dah) as a 7 dit wordspace delay time. The **IM** character can be included to add a small amount of delay and is easier to use than a delay command. To insert an **IM** prosign in a message without translating it to a wordspace, enter it as **/IM**.

'Two Press' Message Button Functionality

Two Press Message Button Functionality

As previously mentioned above, you can trigger message 5 and 6 directly with the following sequence:

- 1) Press and hold PB2 for message 5 or PB3 for message 6
- 2) Press the command PB (you now have two PBs pressed)
- 3) Now release both pushbuttons and selected message will play.

Quick Tune Command

If you use the sequence outlined above but start with PB4 instead, tune mode is turned on. This keys your transmitter until you press one of the paddles or two press PB1+PB4. Use the **G** command to select either a 100% or 50% tune duty cycle.

Quick Serial Number Decrement

Sometimes during contest operation, a serial number has to be reissued. Since the serial number is automatically incremented when it's played, we need a way to decrement the serial number. This can be done by using the **D** paddle command. After the serial number is decremented, PS2B echoes a single dit.

There is a second way to decrement a serial number, that is with a **/D** command embedded in a message. A message can be built that pre-decrements the serial number before sending it. For example the following two messages can be created:

Message 1: 5NN /N QSL ?

Message 2: /D/N QSL ?

Message 3: /N/D /N

Message 1 is the initial exchange, if the serial number needs to be resent, message 2 would be sent. The second message pre-decrements the serial number before sending it. Both of these messages leave the serial number incremented after it is sent. Message 3 will send the serial number twice in one message.

Embedded Message Command List

It is an easy procedure to embed commands in a message. The format is the fraction bar **DN** (D and N sent together as one letter) followed by the desired command letter. If you want to insert the DN prosign into a message but don't want it to be interpreted as a command simply enter DN twice.

Example: **K1EL/1** would be entered as **K1EL//1**

Embedded message command table

/Bnn	Set a beacon cycle time of nn seconds (nn=00 to 99). Put this at the beginning of a message to set the beacon period.
/Cn	Call message in slot n (1-6), return and resume current message.
/D	Decrement serial number.
/E	Toggle message banks
/Hn	Set HSCW speed. See table below for determining n.
/Inn	Increase letterspacing within a message, nn is a value from 0 to 31 percent times 2.
/Knn	Key transmitter for nn seconds. (nn=00 to 99)
/M	Play user callsign.
/N	Play Serial Number with auto post increment.
/On	Select key output port; 1 for port 1 or 2 for port 2.
/P	Pause and wait for paddle entry and then continue after one word space time. The pause is ended three ways 1) paddle entry 2) Press a msg PB (2-6) or 3) Press the cmd PB to cancel.
/Q	Set QRSS speed. See table below for determining n.
/Snn	Set a speed change within a message. (nn=WPM, 5 to 99)
/Un	Turn PTT on or off. PTT is turned on when n = 1 and off when n=0.
/V	Send the current voltage in Morse, useful for beacons.
/Wnn	Wait for nn seconds. (nn=00 to 99)
/Xn	Cancel speed override, for example cancel HSCW, QRSS, or /Snn speed.
/Yn	Force a relative speed change up. Add n to the current WPM. n=(0-9)
/Zn	Force a relative speed change down. Subtract n from the current WPM. n=(0-9)
/1	Jump to message 1
/2	Jump to message 2
/3	Jump to message 3
/4	Jump to message 4
/5	Jump to message 5
/6	Jump to message 6
AA	End message load immediately
IG	Insert ½ letterspace pad
IM	Insert wordspace pad
	(DI-DAH-DI-DAH)
	(DI-DI-DAH-DAH-DIT)
	(DI-DI-DAH-DAH)

Rate Table

n	HSCW Rate	QRSS Rate
0	1000 lpm (200 wpm)	3 sec dit
1	1500 lpm (300 wpm)	6 sec dit
2	2000 lpm (400 wpm)	10 sec dit
3	3000 lpm (600 wpm)	12 sec dit
4	4000 lpm (800 wpm)	30 sec dit
5	6000 lpm (1200 wpm)	60 sec dit

Embedded Command Examples:

/B60BCON DE K1EL BEDFORD NH will send BCON DE K1EL BEDFORD NH every 60 seconds
UR RST IS /P QSL will pause to allow the user to enter the RST then resume automatically
/K05 /W10 VVV DE K1EL will key down for 5 secs, wait 10 secs, and then send VVV DE K1EL
CQ CQ CQ DE /M /M /M will send a 3x3 CQ using the user callsign
/H2CQ CQ DE K1EL K1EL K1EL/S15 DE K1EL will send 1st part at 1500 lpm and the 2nd at 15 WPM
CQ CQ CQ DE K1//10TMT//100 K will send message with 20% extra space in TMT. A more manageable way of doing this would be to load **K1//10TMT//100** in the callsign slot to get this spacing any time /M is used.
CQ CQ CQ DE /Z4K1EL K1EL K1EL/Y4 K send the callsigns 4 WPM slower then return to normal WPM.
QTH IS /E/C1/E NAME IS STEVE Will play the QTH from message bank 1 and then return for the name.
/Q2EL /1 will continuously send EL at QRSS10 speed (this message is in slot 1). Avoid inserting a space between the QRSS command and the start of text: /Q2 EL unless you want a long delay at start of message.
/B10K1EL BCON/W2/VVOLTS sends K1EL BCON, wait 2 secs, send XrXX VOLTS repeat every 10 secs
SOM/E/C3/E/U1/W5/U0/S50K1EL//1/XEOM Send SOM, swap to msg bank 2, call msg 3, swap back to msg bank 1, turn PTT on for 5 seconds, change speed to 50 WPM and send K1EL/1, cancel 50WPM send EOM.

Preserve Settings

PS2B setting changes are not automatically copied into permanent storage. That means the settings will be lost if power is cycled. To save the settings permanently, use extended **S** command. Press and hold the command PB until the **R** and enter an **X** followed by a **S**. This will save current settings in nonvolatile memory so that all settings will be retained on power cycling. Note that messages are always saved directly into nonvolatile memory so the **S** command is not required to preserve messages.

Restore User Defaults (Warm Restart)

It's possible that a command could get entered by accident and put the PS2B into an undesired state. An easy way to restore user default settings is to power off and on.

Restore Factory Defaults

If you want to restore the PS2B to original "factory" settings, turn the PS2B off, squeeze both paddles, and then turn the PS2B on. After about 2 seconds, release both paddles and the PS2B will send a **C** signifying a cold reboot. This erases all settings and messages and then restores factory settings. Baseline paddle calibration is not changed by a cold reboot.

Factory Defaults are:

Fixed WPM: 15	Command WPM: 15	Sidetone: 800Hz	Weight: No adjustment
KeyComp: 0	Interchar Spacing: Normal	SampleAdjust: None	KeyMode: Iambic B
Sidetone: On	Autospace: Off	OutputMode: KEY/PTT	Serial Number: 0001
TX Mute: Off	User: 1	Message Bank: 1	Messages: All Erased
Speed Pot Min: 5	Speed Pot Max: 35	First Extension: 0	Farnsworth: Off
Dit/Dah Ratio: 1:3	Sidetone: On	Tune Duty Cycle: 100%	Extra Letterspace: None
Output: Key0/PTT			

Keyer Lock

A lock feature is provided to disable paddle input and message pushbuttons. This is useful when you want to pack up a battery powered keyer and insure that it stays off with batteries still connected. It is also handy to lock the keyer paddles to keep little hands from sending 'messages'. To lock the keyer, press the command pushbutton, wait for the **R**, and then enter a period (di-dah-di-dah-di-dah). The PS2B will respond with an **L** to let you know it is now in a locked state. To unlock the keyer, press and hold the command pushbutton for about 8 seconds and the PS2B will wake up and send an **R**.

Power On/Off Switch

Since the PS2B has a power on/off switch it does not go into a low power sleep mode when idle like other K1EL keyer ICs do. It uses minimal power when idle but if you leave the power on, the batteries will drain in a few days. So always remember to turn power off when not in use.

External Paddle and Straight Key Connections

If you want to take a break from the touch paddle and use a conventional iambic paddle, simply plug it into J1 and it will operate in parallel with the touch paddles. As far as a straight key goes, it's recommended to not feed this through the PS2B. Instead, wire the straight key directly across the PS2B keying output. Since the PS2B outputs are open drain, the two devices happily coexist.

PS2B Command Cheat sheet

Immediate Command List:

A	- Toggle sidetone on/off	N	- Load 4 digit serial number
B	- Easy Beacon	O	- Output port configuration
C	- Set Command Speed	P	- Start Practice Session
D	- Decrement Serial Number	Q	- Query: Report current settings
E	- Swap Message Banks	R	- Review message with transmit muted
F	- Set Farnsworth Speed	S	- Set Fixed Speed
G	- Toggle Tune Duty Cycle	T	- Set PTT Tail Delay
H	- Set PTT Hang Delay	U	- Toggle Autospace On/Off
I	- Set Letterspacing	V	- Set Keying Compensation
J	- Set paddle sensitivity	W	- Set Keying Weight
K	- Select keyer mode	X	- Enter Extended Command Mode
L	- Set PTT Lead-In Delay	Y	- Set Dit/Dah Ratio
M	- Toggle Transmit Mute	Z	- Select Sidetone Frequency
<u>AA</u>	- Set tune on	.	- Lock paddles (period)

Extended Command List

A	- Analog Input Diagnostic	R	- Set Speed Pot Min and Max WPM
C	- Toggle Contest Spacing	S	- Preserve settings in EEPROM
E	- Set 1 st Element Extension	T	- Toggle Number 0 cut
F	- Toggle Fast Command Response	U	- Swap Users
M	- Load Callsign	V	- Voltage Readout in Morse
N	- Toggle Number 9 Cut	X	- Swap Paddle Inputs (left to right)
O	- Set output Mode		

Embedded Message Command List

/Bnn	- Set a beacon cycle time	/Un	- Turn PTT On/Off (n=0 or 1)
/Cn	- Call message (n=1-6)	/V	- Send Voltage reading in Morse
/D	- Decrement Serial Number	/Wnn	- Wait for nn seconds (n=0-99)
/E	- Swap message banks	/X	- Cancel speed override
/Hn	- Set HSCW Speed (n=0-5)	/Yn	- Relative WPM change up (n=0-9)
/Ann	- Set Alternate Letterspacing (n=0-31)	/Zn	- Relative WPM change down (n=0-9)
/Knn	- Key down for nn seconds (n=0-99)	/1	- Jump to message 1
/M	- Play user callsign	/2	- Jump to message 2
/N	- Play Serial Number	/3	- Jump to message 3
/On	- Select Key Output Port (n=0 or 1)	/4	- Jump to message 4
/P	- Pause and Wait for Paddle	/5	- Jump to message 5
/Qn	- Set QRSS Speed (n=0-5)	/6	- Jump to message 6
/Snn	- Set Sending speed in WPM (n=0-99)	<u>AA</u>	- End message load
<u>IM</u>	- Insert Wordspace (DI-DI-DAH-DAH)	<u>IG</u>	- Insert Gap (DI-DI-DAH-DAH-DIT)

Rate Table for H and Q Commands

n	HSCW Rate	QRSS Rate
0	1000 lpm (200 wpm)	3 sec dit
1	1500 lpm (300 wpm)	6 sec dit
2	2000 lpm (400 wpm)	10 sec dit
3	3000 lpm (600 wpm)	12 sec dit
4	4000 lpm (800 wpm)	30 sec dit
5	6000 lpm (1200 wpm)	60 sec dit

PS2B Tutorial

Start with power on:

After power is applied, the PS2B will output the letter **R** to let you know it's ready. Touching the paddle will generate dits and dahs both in sidetone and keyed output. If you press any of the message pushbuttons, the PS2B will send an **MT** meaning the message slot is eMpTy. Even though the keyer uses a small amount of power when idle, be sure to turn the keyer off when not in use to prolong battery life.

Command Entry:

A command cheat sheet can be found on page 30. There are three command types, immediate, extended, and embedded message. The first commands to look at are the immediate commands.

Immediate Command Entry:

The leftmost pushbutton is the command pushbutton (CMD). Press and hold this until the PS2B answers with an **R**. Then without hesitation. Release CMD, and then enter an immediate command letter on the paddles. Try the **Z** command which allows the sidetone frequency to be changed. The PS2B will output a continuous tone and you can adjust the frequency by touching the paddles. When you are finished, press the command button to return to normal operation.

If the PS2B does not understand a command, or you are too late in entering a command, the PS2B will respond with a question mark.

Some commands require additional parameters; a good example is setting the command speed. Press CMD, wait for the **R**, and then enter **C**. The PS2B will respond with an **E** telling you it's waiting for you to enter the new speed in WPM. Enter a 1 followed by a 0. You have changed the command entry speed to 10 WPM. Try 10 again but this time use a **T** for the zero. This is a handy shortcut. If this command speed is too slow, repeat the command with a more comfortable speed.

Change Keyer Mode:

If you enter the **K** command, the PS2B will prompt you for a keying mode. **A** sets lambic A while **B** sets lambic B, there are many other choices, all covered on page 20. No matter what mode you set, the PS2B will respond with an **A** to let you know it acknowledged the command or **?** if an illegal mode was entered.

Other keying parameter commands:

The sample adjust command **J** adjusts the paddle timing to respond the way you prefer. (a value of 55 will come close to another well known keyer). Setting sample adjust to zero will disable the dit and dah paddle memories. Weighting, Keying compensation, and letterspace are a few other ways to adjust the way Morse is generated. These commands are covered in detail on pages 19 through 22.

Entering Extended Commands:

Since we have more commands than letters in the alphabet, we added extended commands. They work just like immediate commands with the exception that you have to enter two letters. An often used extended command is the 'save settings' command **S**. Press CMD, wait for the **R**, and in response enter an **X** for extended command, wait for the **E**, then enter **S** for save. The PS2B will save the current settings to nonvolatile memory so that when you power off and on your custom settings are preserved. Messages are automatically saved in memory when entered but settings have to be saved by the Save command. It works this way since most users make temporary changes settings but always want to go back to their favorite settings when they are done. It's easy to restore default settings, just turn the PS2B off and then back on again. Another interesting extended command is **V** which tells the PS2B to report the current keyer supply voltage. It plays it in this form: **4r35** which in this case is 4.35 volts.

Messages:

Now let's play with messages. Review the procedure for message loading on page 26. The PS2B has two unique features associated with message loading. The first is backspace, if you make a mistake while entering a message, just hold the command button down and the PS2B will back up letter by letter. Just release CMD when you are done backing up. The second feature is a variable message slot size; if you only use two bytes in slot one, only two bytes of message memory are used up, not an entire slot.

To start, we will load a message into slot 1. Press and hold CMD, wait for the **R**, release CMD, and then momentarily press CMD again. This tells the PS2B you want to load a message into slot 1. The PS2B responds with an **E** to let you know it is ready to accept the message. On the paddles enter **CQ CQ TEST**. Pause after entering the first CQ and a word space will be inserted. You will hear a dit in sidetone when this

happens. To end the message entry, either enter **AA** (DI-DAH-DI-DAH) or press the CMD PB. Now when you press CMD your message will be sent. All 6 message slots are loaded in the same manner except slots 5 and 6. The procedure for those are described on Page 27 (*Two Press ... Functionality*)

Embedded Message Commands

Follow the procedure you learned in the previous paragraphs and enter the following message in slot 2:

/S10SLOW /S25FAST where **/** is the DN prosign (DAH-DI-DI-DAH-DIT)

The letter(s) following a **DN** are the embedded command. This message has two embedded speed change commands which happen while the message is playing: **SLOW** will be sent at 10 WPM and **FAST** at 25 WPM. (make sure to not allow a space to be inserted between **DN** and **S**) After this message ends, the operating speed will be returned to what it was before the message was sent.

Embedded Beacon Command: It's easy to compose a beacon command. First, we will load msg slot 2.

Enter: **/B60/K05 BCON DE K1EL NH**

The beacon starts when you press MSG2, first a key down for 5 seconds, then **BCON DE K1EL NH** is sent. **/B60** specifies that the beacon will be repeated every 60 seconds. To cancel a beacon simply press CMD and PS2B will stop the loop and respond with an **X** to let you know the beacon was cancelled.

More embedded commands with Serial Numbers:

Next we will test out serial numbering. First enter a starting serial number using the immediate **N** command. You need to enter all four digits including leading zeroes. A serial number is sent by inserting the **/N** embedded command in a message. You may want to select the way PS2B will send 0s and 9s in a serial number through the use of the extended commands **N** and **T**. (see page 24).

Here is an example of a message that will play a serial number incorporating the **/P** pause command:

In slot 1 enter: **CQ DE K1EL/P UR NR /N QSL?/P**.

In slot 2 enter: **UR NR /D/N QSL ?**

PB1 will send CQ and then pause to let you listen for a reply. If there is no reply, hit PB1 to repeat the CQ. If there was a reply, enter the station's callsign and the PS2B will send the serial number and pause again. If the station needs a repeat of the callsign, press PB2. Since the serial number is incremented after an **/N** command you need to pre-decrement it with **/D** to send the correct serial number.

In software terms, the **/P** command is a three-way branch: First branch: *paddle something to continue*, Second branch: *hit a msg button (other than CMD PB)*, Third branch: *hit the CMD PB to cancel the message*. Since MSG1 = CMD PB you can't use slot one as a 2nd branch choice.

HSCW and QRSS: The PS2B supports two alternate sending rates. They are selected by putting embedded commands in a message. QRSS is extremely slow CW for VLF operation, while HSCW is extremely fast CW typically used for QSOs via meteor scatter. Here are examples of each:

QRSS: **/K10 /Q2EL/2** Keydown for 10 seconds followed by **EL** at QRSS6 rate, repeat.

HSCW: **/H3K1EL K1EL K1EL K1EL K1EL K1EL** the callsign K1EL is repeated six times at 3000 LPM

Manual Revision History

A.1 Initial Release	4/4/2017
A.2 Minor Corrections	5/2/2017
A.3 Clarified paddle assembly	4/15/2018
A.4 Added baseline calibration, appendices	5/7/2018

K16-PS F/W Revisions

Rev A – Initial First Release

Rev B – Fixed message stacking bug

Rev C – Added baseline cal, favorite speed disable, allow output during echo practice

Watch the K1EL Website for latest updates and new product offerings: <http://www.k1el.com>

Appendix A - Kit Construction Hints

1. Find a good workspace.

It is essential that you have a good place to work on your kit,

You will need room to spread out your parts and have access to tools. Good lighting and ventilation is essential. A magnifying glass or visor is highly recommended.

2. Have the proper tools.

At a bare minimum you will need:

Small side cutters, flush cutters are a plus.

Small needle nosed pliers

Small flat blade & Philips head screw drivers

A good quality, 40-60Watt, temperature controlled Soldering Iron. The price has come down on these lately, look for a [Weller WLC100 40W](#) soldering station it has adjustable power control for under \$40.

3. Read the Instructions First.

Read through the assembly instructions completely and have everything on hand before you start. Inventory the kit parts and make sure you have ALL of them.

4. Follow the assembly instructions in order.

Although not always obvious, the order in which parts are added to a board is important and should be followed. Sometimes sections are installed and tested in order or there could be a mechanical consideration.

5. Keep your Workplace Clean and Orderly.

Nothing spoils a kit building experience more than lost parts. Second to that is stray bits of dirt and metal that get into a printed circuit board assembly. Our PC boards are nicely plated and accept solder easily. There is no need to clean the board with steel wool before starting. A good rosin core solder will work fine, avoid organic core unless you have the capability to clean the board after assembly. Lead free solder is recommended for obvious health reasons.

6. Take your time.

There is no need to rush, enjoy the process and the end result will be much better. Moving too quickly or working when you are tired often leads to big mistakes which could be difficult if not impossible to fix.

Appendix B - A Note About Safety

Burns to your skin can be very painful and can lead to serious injury.

Burns to your eyes can be catastrophic.

Toxic fumes can cause serious harm.

Flying objects such as wire ends etc. can cause painful and serious injuries.

When building your kit please remember that Soldering Irons and Solder are used at High Temperatures !

Soldering Irons can remain hot for many minutes after being turned off. Never touch the tip to see if it is hot. Touch the tip to a wet pad to test for temperature.

Wear safety glasses to protect your eyes from flying objects.

Appendix C - Soldering Basics

1. Insert component leads into PCB holes and bend them back slightly to hold the part in place. You can either trim the lead now or wait till after the joint is soldered. I usually install several parts at one time and then solder and trim multiple leads.
2. Place a hot and clean iron tip against both the lead and pad as in Fig. C1.

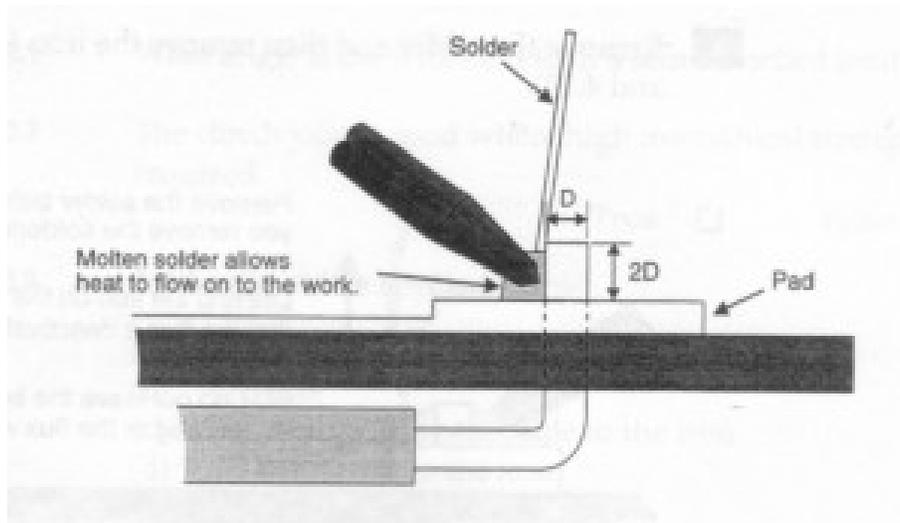


Fig. C1 - Form a heat bridge

3. Create a heat bridge between the lead, the PCB pad and the iron by placing a small amount of solder on the tip.
4. Apply solder around the outside edge of the pad as in Fig. C2. If the pad and lead are at the correct temperature, the solder will flow around the connection.

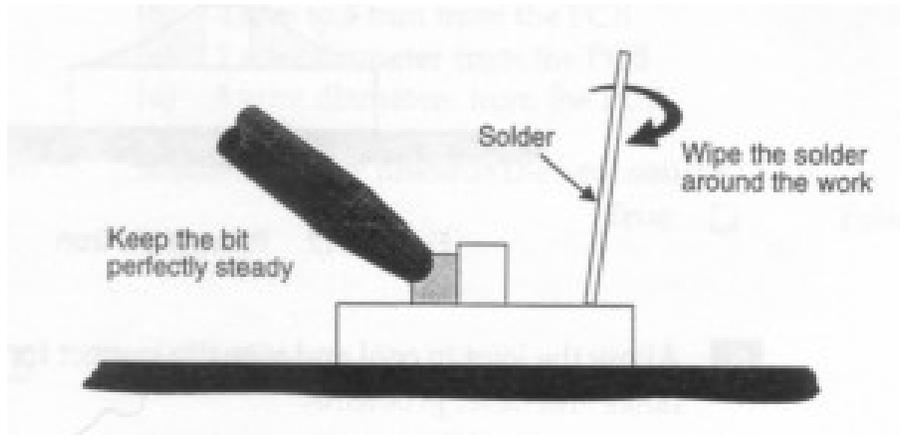


Fig. C2 - Spread solder around the work

5. Remove the solder and then remove the iron.

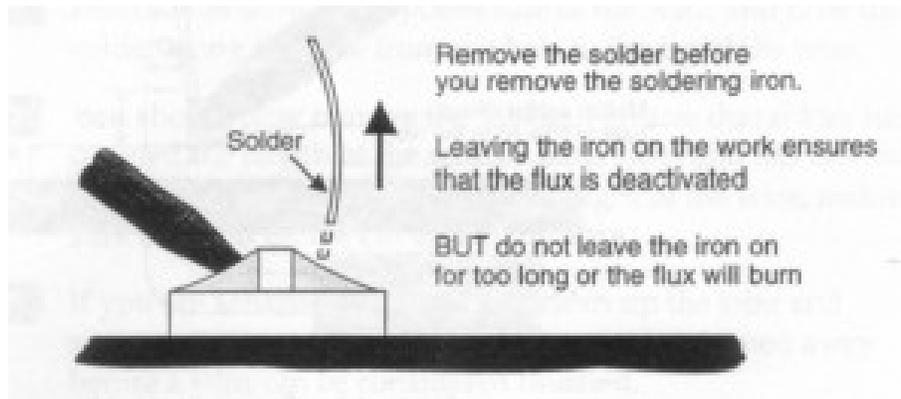


Fig C3 - Remove the solder

6. Allow the joint to cool and visually inspect for defects or other problems. You should have a solder joint with a bright shiny finish and a profile like that shown in the middle picture below. Make sure you use enough heat so that solder flows around both the lead and pad. A good connection will always look like a tent, if it looks like a round ball, then odds are, the solder did not bond to the PCB pad.

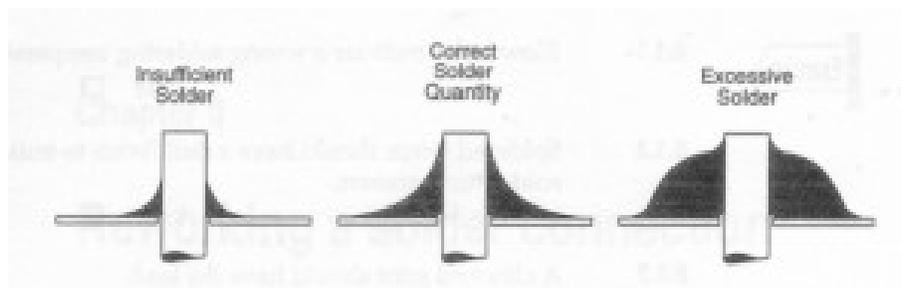


Fig. C4 - Solder quantity comparison

7. To avoid cold solder joints, do not move the board or component lead while the solder is cooling. Keep your iron tip clean, contamination can cause poor solder joints.