Ultimate2: QRSS Labs Multi-mode QRSS Beacon Kit

1. Introduction

Thank you for purchasing my second generation “Ultimate2” Multi-mode QRSS beacon kit. This kit is capable of automated transmission of a range of weak signal modes that are capable of worldwide HF propagation using a fraction of a watt of RF output power. The DDS module permits accurate, stable operation anywhere on HF whilst plug-in low-pass-filter modules allow easy band changing.

Recommended approach to building the kit: This is a simple design but there are a large number of features which provide a great deal of flexibility. **Read this WHOLE manual and understand it!** Follow the construction section to build the kit. Use a receiver connected to your PC, with a slow-signal decoding program such as Argo, to monitor your signal, experiment and understand the various features before connecting an antenna! **Good performance depends on proper set up:, see the calibration section.**

Pay particular attention to the transistor orientation, which for BS170 transistors should be as per the diagram in section 4.3, NOT the legend on the PCB. Do not follow the PCB legend for Q1-4! Follow the diagram in section 4.3!

Please read the DDS module stability section 11! Do not miss this section, or your output signal may be unstable and look horrible!

The kit supports the following modes:

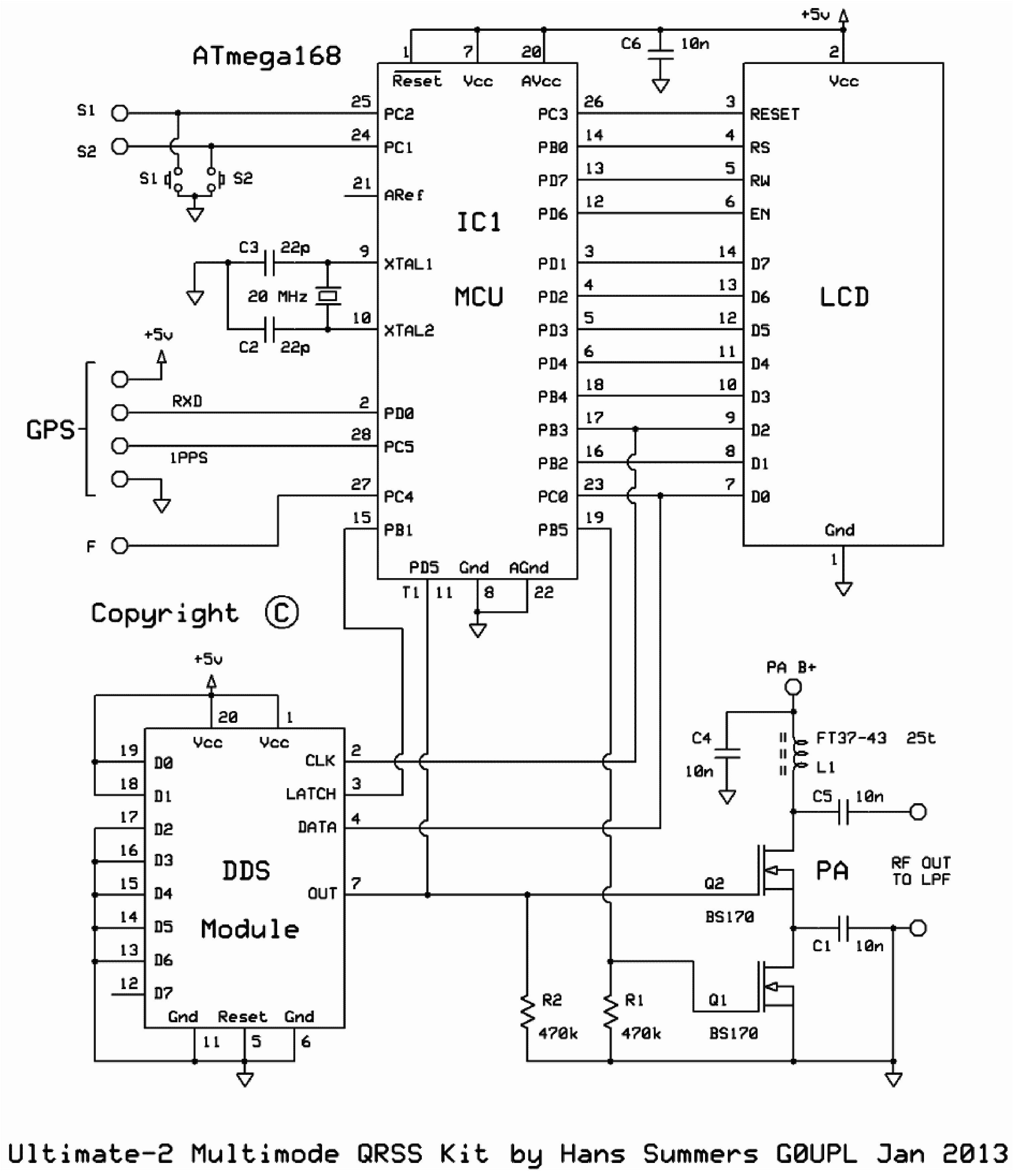
- QRSS mode (plain on/off keyed slow CW)
- FSK/CW mode (frequency shift keyed slow CW)
- DFCW mode (dual frequency CW, dit's and dah's on different frequencies)
- WSPR mode (Weak Signal Propagation Reporter)
- Slow-Hellschreiber (frequency shifted slow Hellschreiber)
- Hellshreiber (full-speed standard Hellschreiber, and half-speed Hellshreiber)
- CW (plain CW)
- Customisable FSK patterns

Other features:

- DDS-controlled output frequency (through-pin DDS module, no SMD soldering required)
- Plug-in low pass filter boards (available for 9 HF bands)
- 24-character LCD + two-button user interface
- User-programmable (callsign, message, speed, FSK, mode, etc.), settings stored in EEPROM
- GPS interface, for locking the frequency, timing and location information
- On-chip generation of WSPR encoded message (no PC required)
- WSPR maidenhead locator can be generated from GPS-derived latitude/longitude
- Selectable “frame” size, for stacked QRSS reception
- Plain CW callsign identifier at selectable interval
- Produces approximately 150mW RF output on 30m (lower output on higher frequency bands)
- Higher output power by fitting additional PA transistors and/or higher PA supply voltage

The transmitter is designed to be powered with 5-6V DC, which could come from a mobile phone charger, wall wart, or even four 1.5V batteries connected in series. Do not use more than 6V: this may kill the microcontroller. Best results will be obtained with a well-regulated and well-smoothed supply.

2. Design



The ATmega168 AVR microcontroller (IC1) is pre-programmed with firmware to control the LCD, buttons, and GPS interface. The AD9850 Direct Digital Synthesis (DDS) module includes its own on-board 125MHz crystal reference oscillator, and reconstruction filter components ready-mounted. On/off keying is provided by the BS170 transistor Q1, and power amplification by another BS170 transistor, Q2, producing over over 150mW from a 5V supply on 30m. Finally standard 7-element low pass filter plug-in modules remove harmonics of the transmission frequency.

3. Parts List

3.1 Resistors

R1, 2	470K (yellow-purple-yellow-gold)
R3	No resistor – space is provided on the PCB only for possible future expansion
R4	No resistor – space is provided on the PCB only for possible future expansion

3.2 Capacitors

C2, 3	22pF (ceramic, marking 22)
C1, 4, 5, 6	10nF (ceramic, marking 103)

3.3 Inductors

L1	25 turns, FT37-43 core (black)

3.4 Semiconductors

Q1, 2, 3, 4	BS170 (note: only two are provided in the kit. The other two may be added for more power).
IC1	Pre-programmed ATmega168 microcontroller
DDS	AD9850 DDS module, including 125MHz reference oscillator and reconstruction filter

3.5 Miscellaneous

Push buttons S1 and S2
1 row, 24 character LCD
Printed circuit board
Wire for winding toroids
Socket for IC1
20MHz quartz crystal
Two 10-pin female header sockets
Two 4-pin female header sockets

4. Construction

4.1 General construction tips

The kit comes as a main board, with pre-assembled DDS module, and a plug-in low-pass-filter module for the desired band. **Since the low-pass filter module is a separate kit in its own right, please refer to the separate instructions for constructing that kit.**

Parts placement is defined by the printed legend on the PCB, so please observe it carefully, paying particular attention to the correct orientation of the semiconductors.

Note that the PCB legend is incorrect for the supplied BS170 transistors, which must be mounted with their flat side the OPPOSITE way around to the PCB legend. The layout diagram below is correct for the BS170 transistors. The PCB legend is only correct for 2N7000 transistors.

The PCB is quite small and the parts are close together. You are recommended to use a low wattage iron with a fine tip, and fine solder e.g. 1mm diameter or less. Take care not to overheat the PCB and risk damaging it. A well-lit area and magnifying glass can assist. Be careful not to bridge solder across closely-packed connections. Some of the joints are very close to each other. I recommend checking with a DVM to make sure no solder bridges have been inadvertently created.

Note that components R3 and R4 are not required or supplied in the kit. Q3 and Q4 are not supplied, additional BS170 transistors may be installed here to increase the output power (see below). IC1 (the microcontroller) has an IC socket, in case in future you wish to change the microcontroller e.g. for a firmware upgrade for new features, etc., or in case you wish to program it yourself. Sockets are provided for the DDS module and the low-pass-filter module.

4.2 LCD and button mounting options

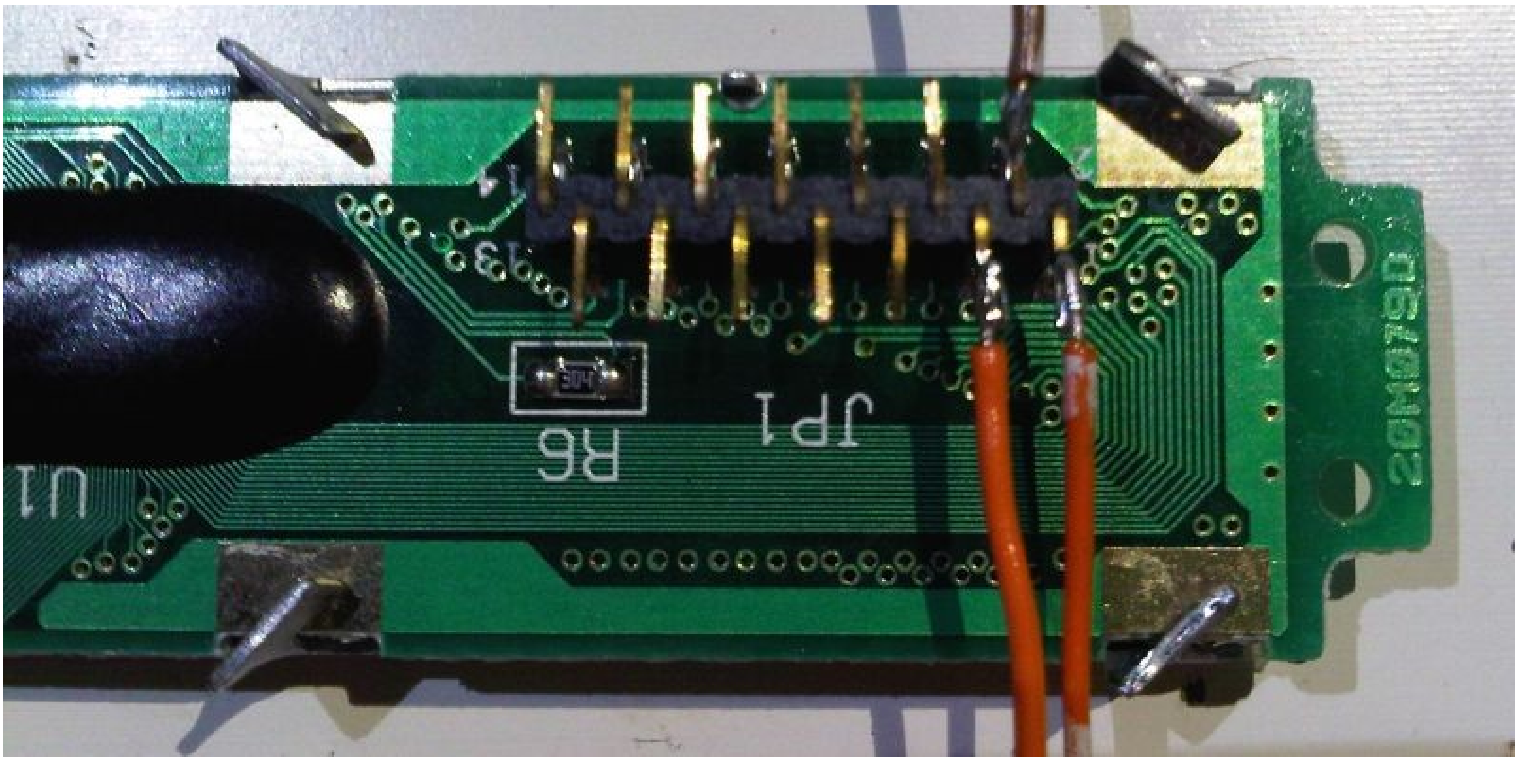
Now before going any further, you need to consider how you want to mount the LCD and buttons. There are two push-buttons in this kit, and they must be mounted on the BOTTOM side of the PCB, so that they do not get in the way of the plug-in modules on the top side. If I was to make space for the buttons on the top side of the PCB, the PCB would have become somewhat larger. A larger PCB costs more to make and that would increase the cost of the kit. I did everything possible to minimise the kit cost! So you have to put up with this potential inconvenience. Please read about the three possible LCD mounting options and decide now, which one you want to use.

Method 1: LCD and buttons mounted off-board

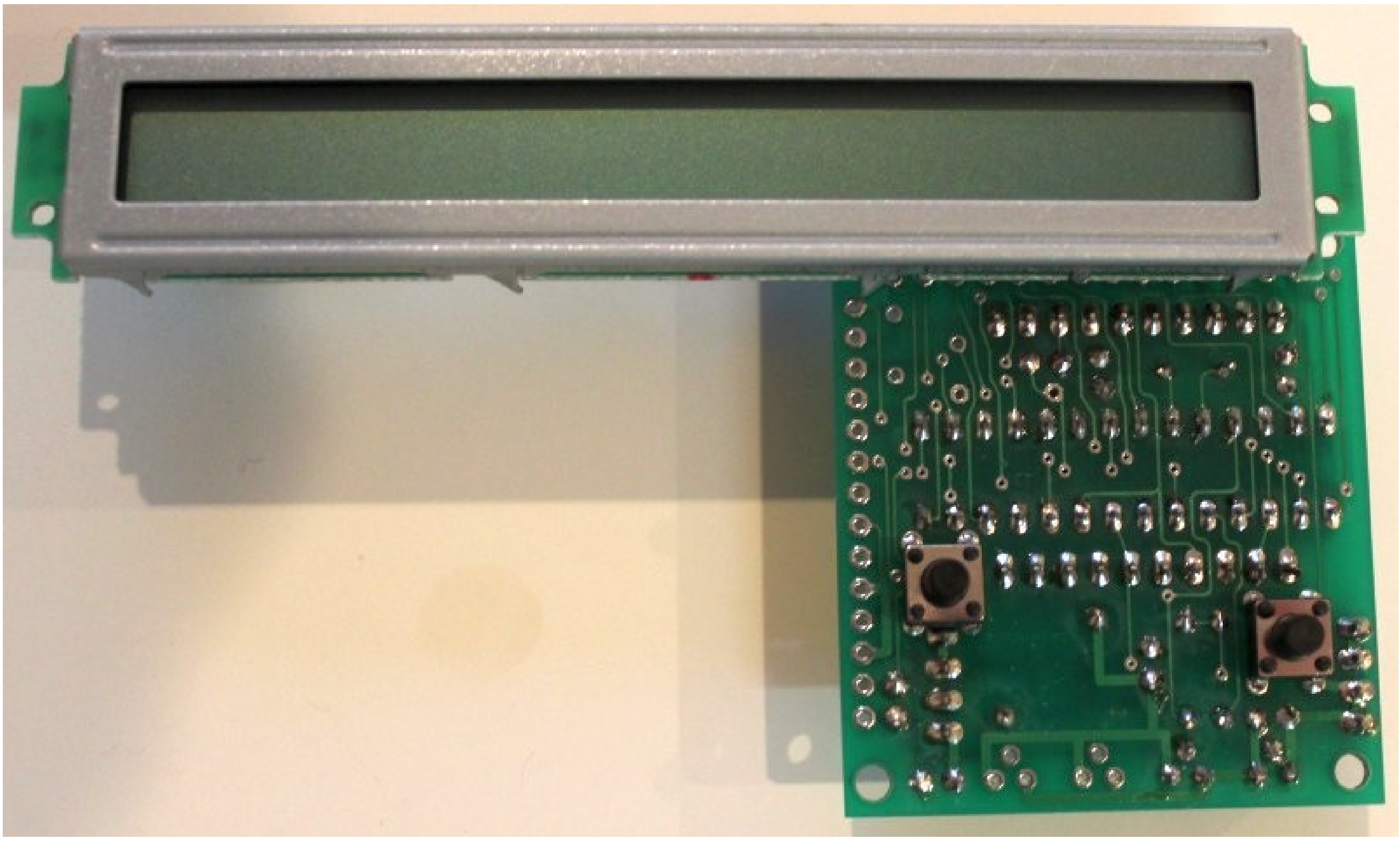
This method removes the requirement to solder the fine-pitch LCD pins. It is also convenient for those of you who wish to install the LCD and kit in an enclosure, for example to mount the LCD in a front panel.

In this case, the pins of the LCD should be bent alternately in different directions. Short lengths of wire can be used to connect the LCD to the PCB, using the row of holes adjacent to the fine-pitched LCD holes. This row of holes are spaced at 0.1-inch pitch so are much easier to solder than the 0.05-inch pitch holes if you were to solder the LCD directly to the PCB. The photograph below shows the LCD pins bent different was and starting to solder some wires to them. Of course, take care to solder the right wires to the right holes.

You may also wish to consider using your own push-buttons installed in a front panel for example. In that case please refer to section 5.3 below, on how to connect external push-buttons.



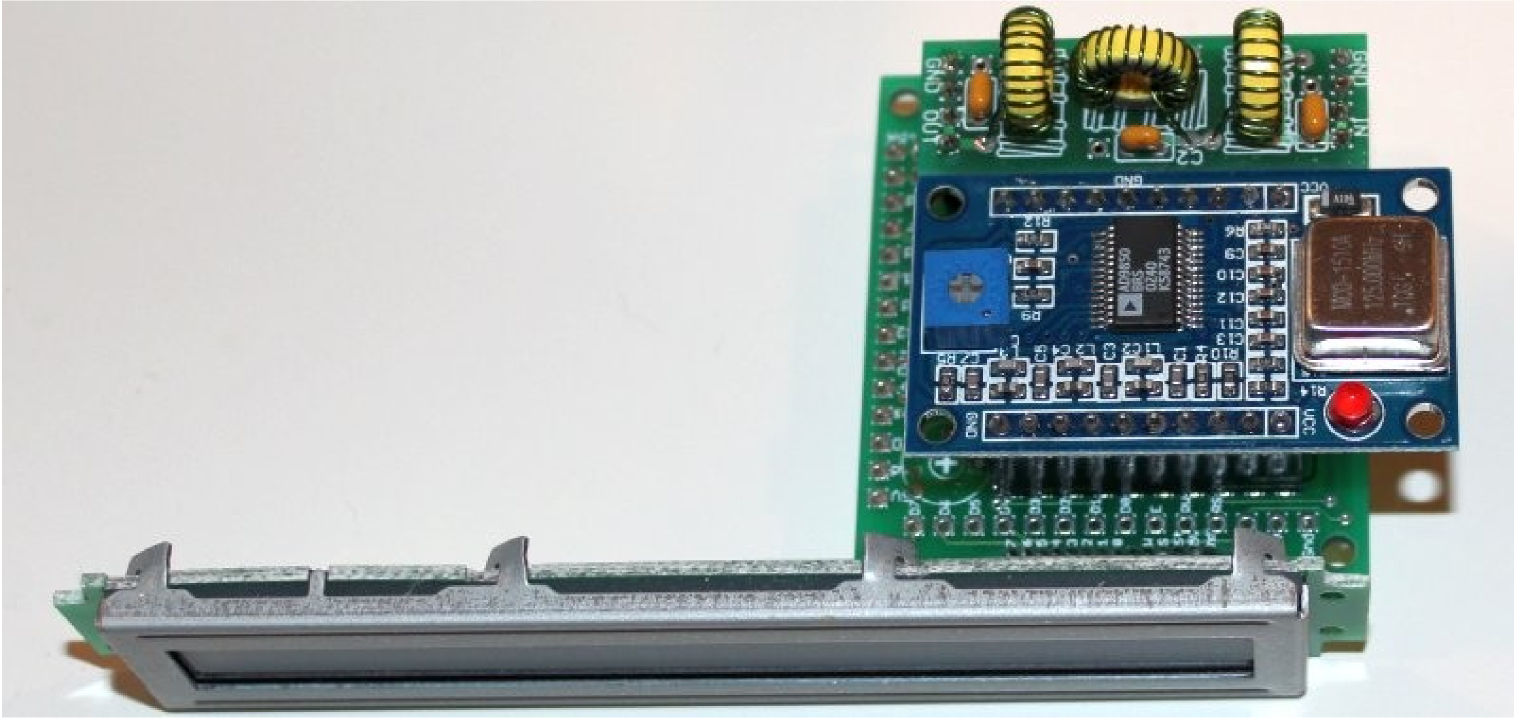
Method 2: LCD mounted directly to the reverse of the board



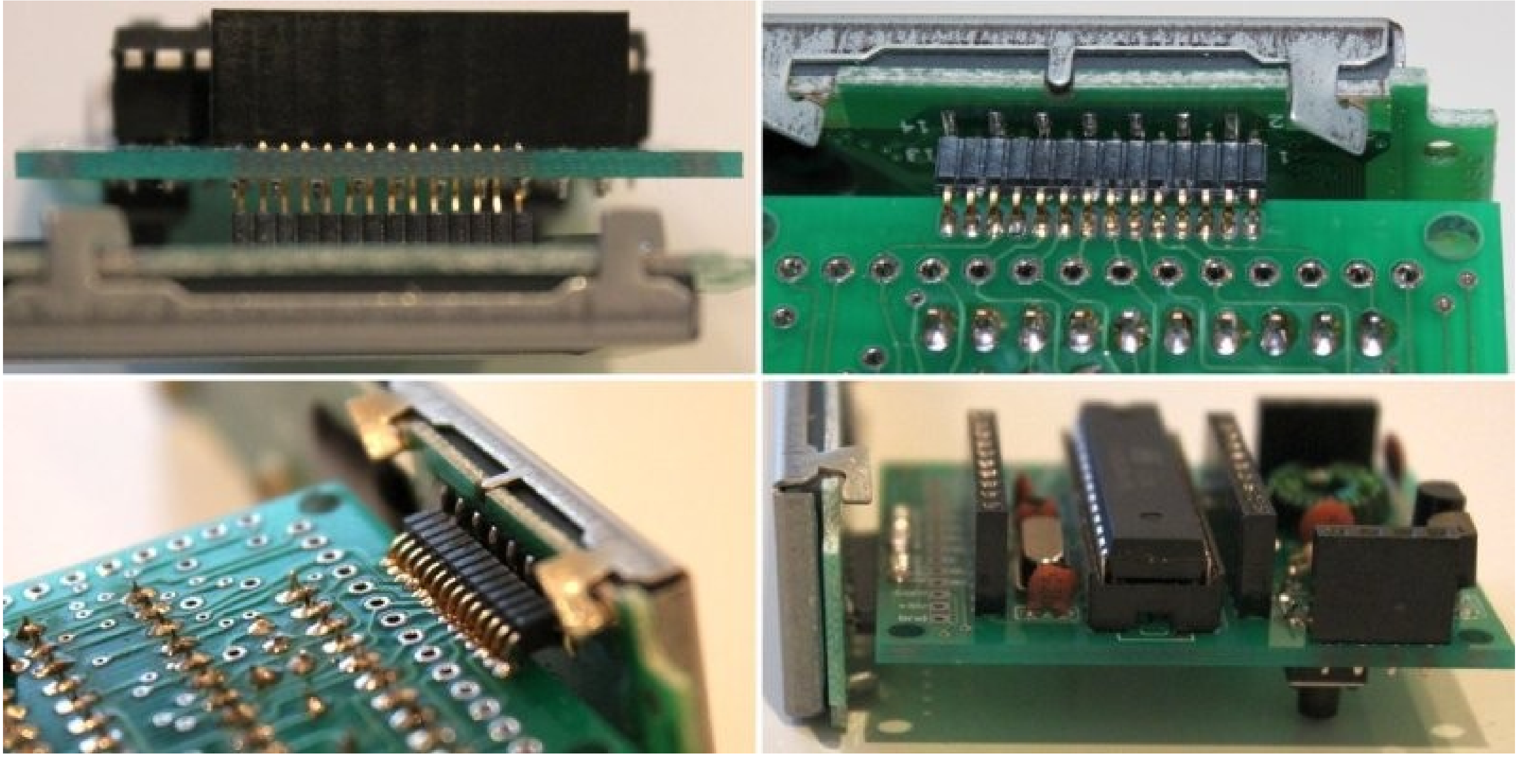
This method is probably the easiest and most direct. The LCD is just soldered in position on the reverse of the board. Make the solder joints on the component side of the PCB. The two buttons are in this case directly accessible. The disadvantage is that the board components are now upside-down! In this configuration you might want to mount some 40mm long bolts in the main PCB corner holes, in order to lift the PCB up, as if on pillars, so that it is not resting on the components underneath.

Note also that in this configuration, you would need to set the “Inv. Btn.” parameter to ON, since the default configuration will have the left and right buttons reversed in this case.

Method 3: LCD at right-angles



The buttons are still underneath the PCB – however it is not too difficult to set up the kit by holding it in one hand, with the LCD facing towards you, and pressing the buttons underneath. My recommendation is still to fit the buttons (or your own buttons) off-board on a length of cable, and mount them in the front panel.



Step 1: Insert the LCD from the bottom of the board. I suggest soldering just the two end pins only, and solder from the top side of the board: this allows un-soldering easily in the event tat However, leave a gap of a few mm, as seen in the top left photo above. That is very important! It allows space for bending the pins at right-angles.

Step 2: Carefully put force on the LCD to bend the pins at right-angles. This has to be done very gently, to not rip off the connector pins from the LCD, which is delicate.

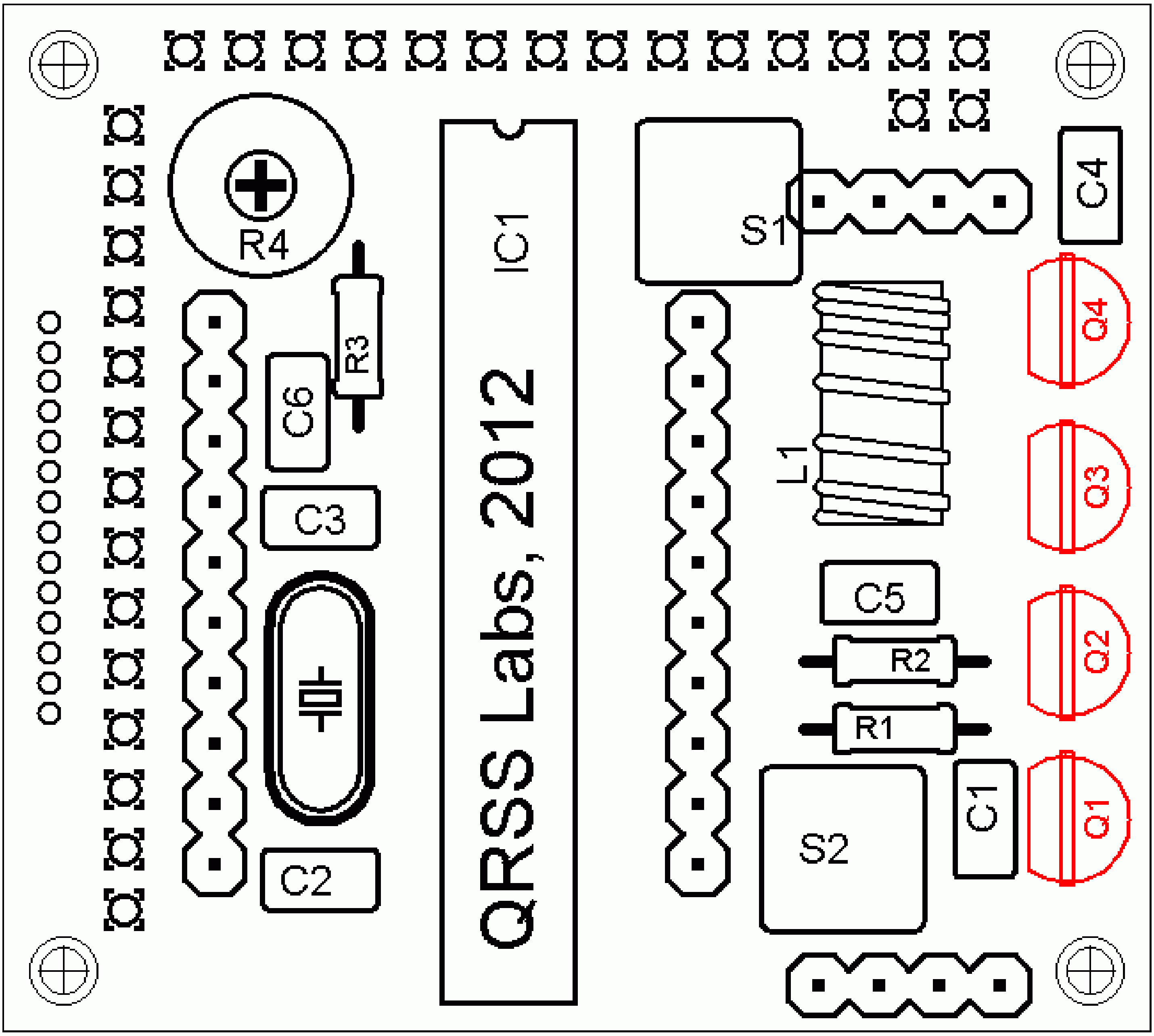
Step 3: Solder the remaining LCD pins, from the top (component side) of the board. The end result should be as shown in the pictures above.

Note 1: this kit is NOT compatible with the common Hitachi HD44780-based LCD display modules.

Note 2: the LCD is quite DELICATE! If you use bolts fix the LCD to a front panel, ensure suitable spacers or washers are used so that stress is not put on the LCD, or it may crack.

4.3 Construction steps

Please refer to the parts placement diagram below.



Pay special attention to the orientation of the semiconductors. For IC1, the dimple in the PCB silkscreen must be aligned with the dimple at the top of the IC socket and the IC.

The order of construction is not important but a good principle to follow is to install the smaller components first, so that the larger ones do not prevent easy access. One suggested order of construction is:

1) Install the socket for IC1
2) Install all resistors and capacitors and the quartz crystal
3) Install switches S1 and S2, if not using off-board switches
4) **Install the transistors Q1 and Q2: note that the legend on the PCB is incorrect for the BS170 transistor supplied with the kit. The legend would be correct for a 2N7000 transistor. For the supplied BS170, you must install the transistor the opposite way, with the flat facing inwards. The parts layout diagram above is correct.**

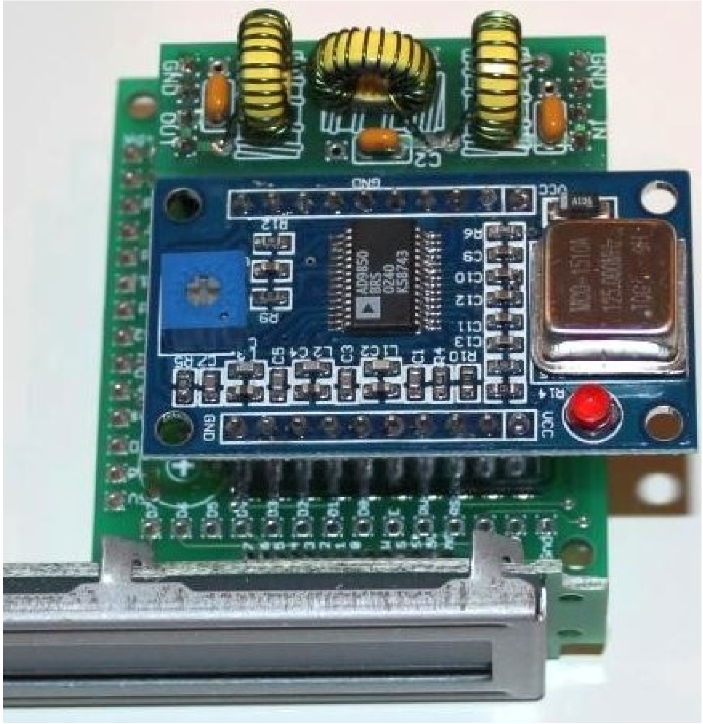
5) Wind and install the toroid – The toroid can be mounted horizontally or vertically. I prefer the horizontal method as there is enough space for it, and it keeps the toroid away from the LPF module which is plugged in above. .

6) Install the sockets for the DDS module and the low-pass-filter board. Some care needs to be taken with the alignment, to ensure that there is a good fit when the plug-in boards are added. One method is to build the LPF kit board first, and use the DDS and LPF boards plugged into the sockets, then solder the pins, to ensure correct alignment.

7) Install the LCD – first consider how you want to box the kit (see LCD mounting options below).

Installation of the LCD requires particular care, because the pin-spacing of the LCD is only 0.05-inches. Use of a fine tip soldering iron is advised, and check carefully for shorts between pins, using a magnifying lens and Digital Voltmeter (DVM). Ensure correct orientation of the LCD as per the photographs in the LCD mounting options section 4.2, above.

When winding the toroid, remember that each time the wire goes through the centre of the toroid counts as one turn. 35cm of wire should be enough for 25 turns. Trim the ends of the wire, scrape the enamel off and tin them with solder. As an alternative to scraping the enamel off, the wire ends may be held in a hot blob of solder on the iron tip for a few seconds, and the enamel will bubble away. Check continuity on the board with a DVM.



4.4 Module assembly

First the microcontroller IC1 should be inserted in its socket.

Next, plug together the three modules as shown in the photograph on the right. Ensure that the DDS module is inserted the correct way round. The Low Pass Filter kit module should be plugged in with the “Out” legend aligned next to the RF Output connector of the main PCB.

4.5 Notes concerning the DDS module

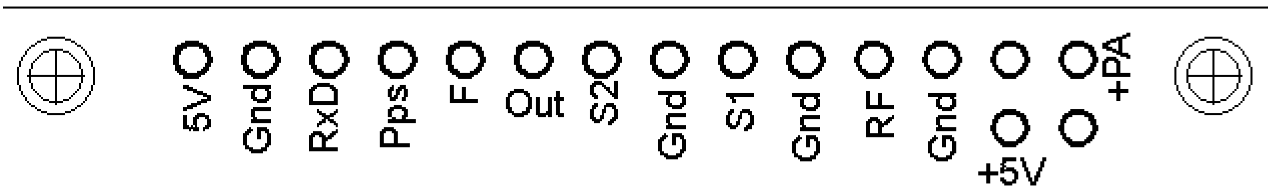
a) You will notice that the DDS module has a square, blue plastic adjustment preset potentiometer. This adjusts the threshold for the DDS chip comparator that turns its 1V p-p sine wave into a square wave to drive the microcontroller. Adjusting it will change the duty cycle of the DDS square wave. You should not need to adjust this potentiometer.

b) The module uses the AD9850 DDS chip. The module is officially rated for output from DC to 40MHz. You may want to test it at higher frequencies than that. In theory the chip could output up to 62.5MHz (half the 125MHz reference clock frequency), but the output amplitude and spectral purity will deteriorate.

c) It is normal for the 125MHz reference oscillator (metal can on the right) to get hot in operation.

5. Hardware Options

5.1 Explanation of connections



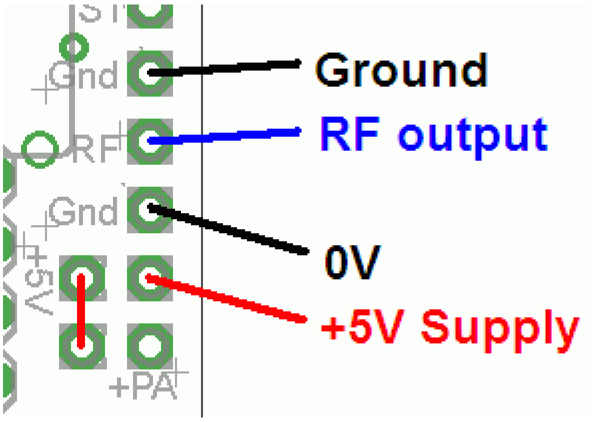
The diagram shows the connection pads along the edge of the board. The pin-spacing is 0.1-inches and a suitable connector could be used if desired. These are further described in the following sections. From left to right:

Label	Group	Description
5V	GPS	+5V supply to GPS module
Gnd	GPS	Ground connection to GPS module
RxD	GPS	Serial data input from GPS module
Pps	GPS	1 pulse per second input from GPS module
F	Future	No connection – for future enhancements
Out	Options	Digitial logic level RF output
S2	Buttons	Optional external button S2, switch to Gnd
Gnd	Buttons	Ground for optional external buttons
S1	Buttons	Optional external button S1, switch to Gnd
Gnd	RF	Ground for RF output
RF	RF	RF output
Gnd	Power	Ground connection to Power supply
+5V	Power	+5V power supply
+PA	Power	PA power supply – may be connected to +5V

5.2 Connection for basic operation

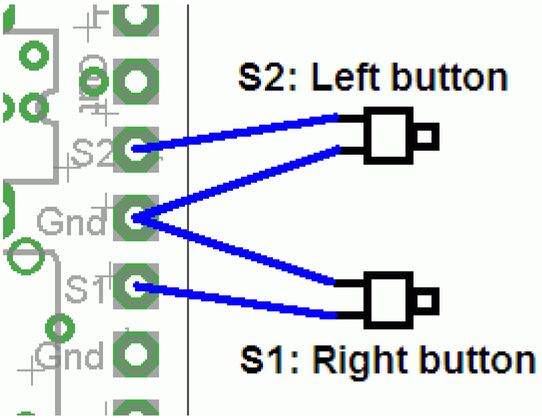
For the most basic operation of this kit as QRSS transmitter using 5V supply, it needs only a power supply and RF output connection.

To allow use of a higher voltage supply for the PA, the PA voltage is not connected to +5V on the PCB. So to run the transmitter using the 5V supply, be sure to connect a wire between the +5V and PA pins, as shown by the short red vertical line in the diagram (right).



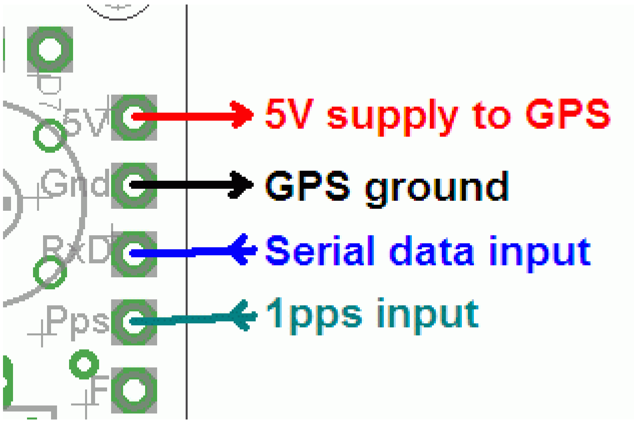
5.3 Alternative button mounting option

If you wish you may use an external pair of buttons to control the kit. The button input signals are available at the edge of the board, labelled S1 (right button) and S2 (left button). The buttons should be of the push-to-make variety. The right button should be connected between the S1 signal and ground; similarly the left button should be connected between the S2 signal and ground.



5.4 Optional connection of GPS module

A GPS module may be connected to the kit, to provide frequency stability, accurate time, and latitude and longitude which can be converted to Maidenhead locator format for encoding in the WSPR message transmission.



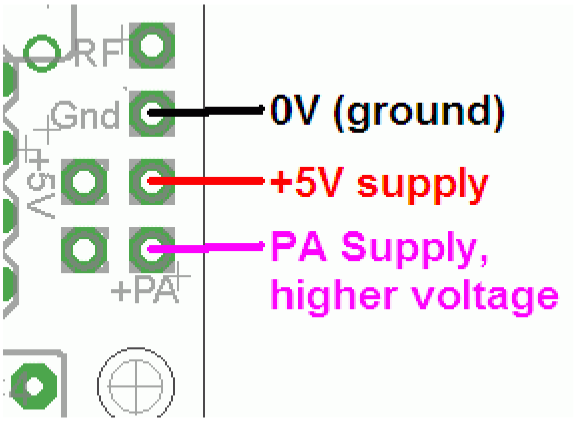
Check that your module is powered from 5V. Many modules specify a 3.3V supply – in this case you will need to provide an external 3.3V voltage regulator. Where a 3.3V GPS module is used, the serial data and 1 pulse-per-second (pps) inputs are not a problem for the 5V microcontroller on the kit PCB – no voltage level conversion is required.

In cases where the location information is not required (modes other than WSPR) or you wish to enter it manually, the serial data input can be left unconnected. The 1 pps input is enough for the frequency lock function. Note that the frequency locking function can only operate if you have selected a wait period between message transmission (i.e. a non-zero “Frame” parameter).

See operation instructions below for details on how to configure the GPS interface.

5.5 Higher power output

The kit provides around 150mW of output power using a 5V supply and a single BS170 transistor (depending on the band). The transistor gets slightly warm to the touch. If higher output powers are desired, this can be achieved by using a higher supply voltage for the Power Amplifier (PA). The microcontroller must still be run at a voltage not exceeding its 5.5V rating (5V is recommended). To facilitate running the PA at higher voltages to provide more power, the PA supply connection is a separate pin.



At higher supply voltages and power outputs, the heat dissipation in the BS170 will increase and its temperature will be higher. Provision is made on the PCB for two more transistors in parallel with the first. This will slightly increase the power output even at 5V supply, but more importantly at higher voltages it should share the heat dissipation between the devices.

It may be necessary to fix a heat sink to the transistors if the temperature rise is excessive. The transistors are sited near the edge to the board to facilitate this. Experiment is needed in this area, with PA voltages up to perhaps 12V.

5.6 Audio frequency output

The kit can be used to generate audio tones for feeding to an SSB transceiver etc. In this case you can simply set the output frequency to the desired audio frequency, and feed the “RF” output to the microphone input. It is very important to note that this output is a just a 5V peak-to-peak square wave – to connect it to a microphone input (for example) you will need to reduce the signal amplitude considerably. In testing, I used a potential divider made of an 18K resistor and a 10-ohm resistor to ground. The mic input was connected across the 10-ohm resistor and provided a suitable signal amplitude.

Another possible way to use the AF output is to rectify it (diode + capacitor) and use that as a plain on/off keying signal to a CW transmitter. This would be suitable for CW and Hellschreiber modes, but not modes where a frequency shift encoding is involved.

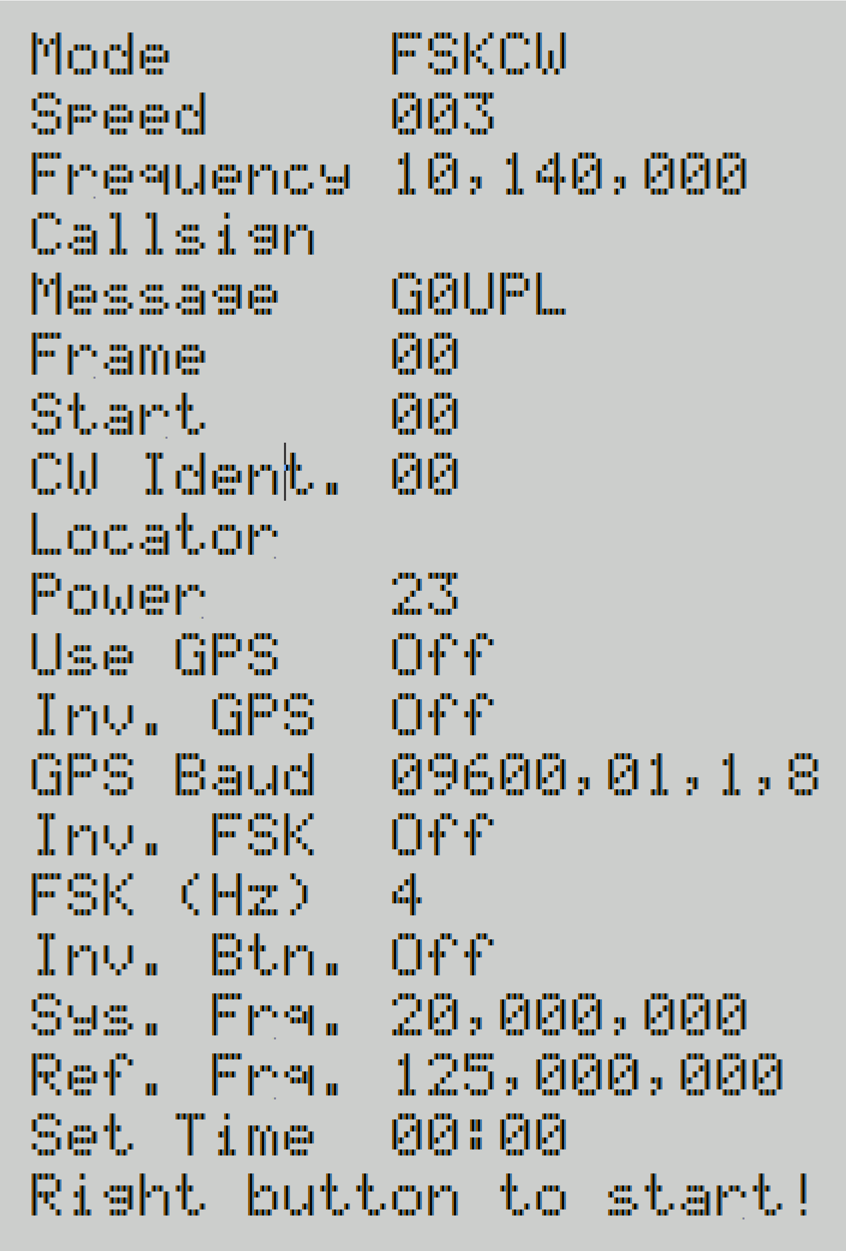
6. Operating instructions

The microcontroller has a 512-byte Electically Erasable Programmable Read Only Memory (EEPROM). This is used for storing your individual settings such as mode, keyer speed, and message. When the kit is first switched on, it automatically enters the “run” mode, in which the message is being transmitted in the selected mode. This is so that in the event of power interruption, the kit recommences the message transmission as soon as power is restored, without user intervention.

When you have just built the kit, the memory will contain only certain default settings, and an empty message. An error will be displayed, saying “Error: Message”, which indicates that the message is empty:

It is now necessary for you to configure the message and settings for your use.

6.1 Menu system



```
Mode        FSKCW
Speed       003
Frequency   10,140,000
Callsign
Message     G0UPL
Frame       00
Start       00
CW Ident.   00
Locator
Power       23
Use GPS     Off
Inv. GPS    Off
GPS Baud    09600,01,1,8
Inv. FSK    Off
FSK (Hz)    4
Inv. Btn.   Off
Sys. Frq.   20,000,000
Ref. Frq.   125,000,000
Set Time    00:00
Right button to start!
```

The user interface consists of the 24-character LCD and two push-buttons. There are 19 user-configurable settings available. The menu system allows all of the settings to be edited. The use of only two buttons to edit so many settings necessarily means that you must scroll through various options by many button presses. Clearly, there is no full computer keyboard in the kit with which to edit your alphanumeric message. However, the use of these two buttons to control the kit is fairly intuitive.

The menu is a list of items, and the LEFT button is used to scroll through these items. When you see the item which you want to edit, press the RIGHT button to start editing it. The list of menu items shown (right) has example settings for sending the callsign “G0UPL” in FSKCW with 3-second dit symbols.

After cycling through the list of menu items using the LEFT button, the system is returned to the “Run” mode, transmitting the configured message.

Note the final item “Right button to start!”. When this final item is reached, you must press the RIGHT button, which then resumes the “Run” mode. The reason for this is that if, for example, you only wanted to change the mode, which is the first setting. It would be tiresome to have to scroll carefully though the remaining items until the “Run” mode was resumed. There are many configurations and you could easily miss the run mode, re-entering the menu system again at the first item. With the right button start feature, you can simply keep the LEFT button pressed, relying on the key autorepeat to quickly cycle through all the remaining menu items and stop at the end. It is also useful for when you want to start the system at an exact time.

6.2 Editing a menu item

There are three types of menu item: alphanumeric (e.g. callsign), numeric (e.g. frequency), and list (e.g. mode). Editing a configuration is slightly different depending on the type. When scrolling through the list of menu items using the LEFT button, you enter editing mode by pressing the RIGHT button.

6.2.1 List editing

If the menu item is a list menu, such as the Mode menu, use the RIGHT button to cycle through the list to find the setting you want. An On/Off menu such as Use GPS is also a list, having only two settings (On and Off). When you have chosen the setting you want, press the LEFT button. This saves the setting into memory and returns to the menu list.

6.2.2 Number editing

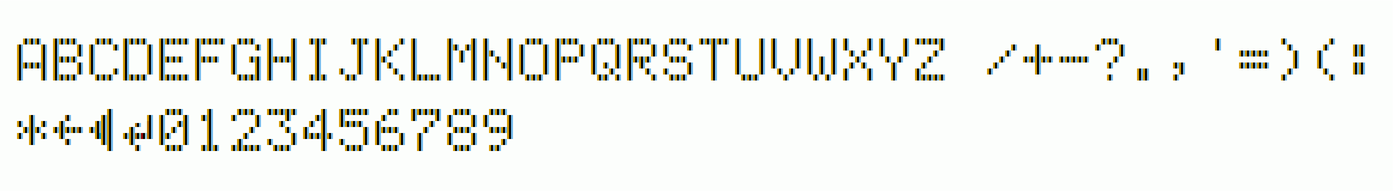
When editing a numeric setting, the RIGHT button is used to cycle through the numbers 0..9 until you find the number you want. The LEFT button moves rightwards one position to the next character to be edited. The current character being edited flashes. When the final (rightmost) digit has been chosen, a LEFT button press saves the setting into memory and returns you to the menu list.

Note that in some cases, there are restrictions on the value which may be chosen. For example, when editing the 10's of minutes digit, the RIGHT button only cycles 0..5 then returns to 0.

6.2.3 Alphanumeric editing

The most complex editing is alphanumeric, which also includes certain punctuation characters supported by the Hellshreiber character set. The principle is the same as for Number editing: the LEFT button moves the cursor one character to the right, and the RIGHT button cycles through the letters, punctuation and digits. However facilities are also provided to delete one character, or the whole message.

The order of the letters, punctuation and numbers is as follows:



ABCDEFGHIJKLMNOPQRSTUVWXYZ /+-?.,'=)(:
*←◀↵0123456789

The following characters have special function:

← **Backspace (delete):** If you select this character as the current flashing character using the RIGHT button, then when you press the LEFT button, the current character is deleted and the flashing cursor moves back left one position.

◀ **Delete all:** If selected as the current character, pressing the LEFT button has the effect of deleting the entire message, starting again at the left of the screen. There is no “undo”, so use with caution!

↵ **Enter (finished):** If selected as the current flashing character, pressing the LEFT button is used to finish editing the setting. The setting is saved, and you return to the main menu list. Note that the text that is saved is only the text to the left of the Enter symbol. If you select this symbol and press the LEFT button when you are not at the furthest right position of the message, then everything right of your position is deleted.

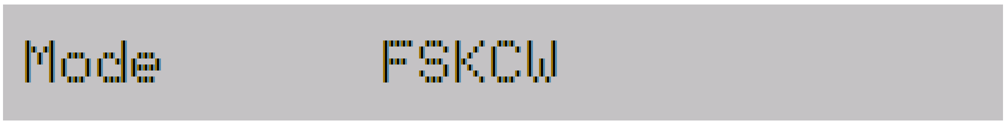
6.2.4 Configuration validation

In some cases, the configuration setting is validated at the time of entry. For example, the system does not allow the entry of an invalid time setting, or entering letters where numbers are required.

Additionally, there are many cases where particular combinations of settings produce an invalid situation for the transmission mode being used. In these cases, the configuration validation is done when the “Run” mode is entered, i.e. transmission started. If there is any reason why the configuration is invalid, an appropriate error message will be displayed, prompting you to return to the configuration setting to correct it.

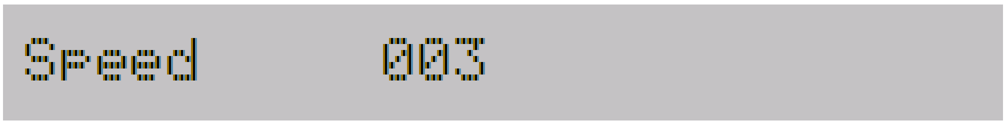
6.3 Menu configuration settings

There are 18 user-configurable settings available in this kit. Not all of them are going to be relevant to the mode you wish to use, or to your hardware configuration. The following sections explain each setting in detail.



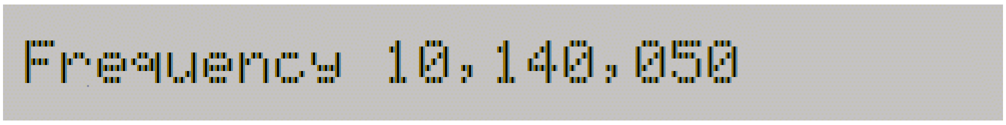
```
Mode      FSKCW
```

The Mode setting selects the transmission mode. The available modes are:
FSKCW, QRSS, DFCW, Slow Hell, Hell, DX Hell, CW, WSPR



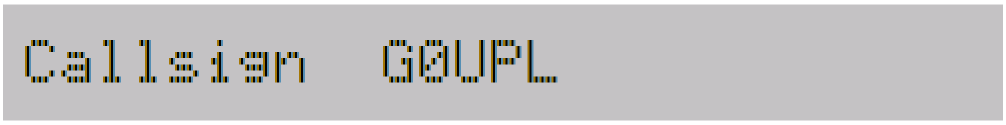
```
Speed     003
```

The Speed setting selects the dit speed, in seconds, for the following modes: FSKCW, QRSS, DFCW. In CW mode, the Speed setting means Words Per Minute. In Slow-Hell mode, the setting is the number of seconds duration of one character (see later section for explanation). The setting has no effect at all, in Hell, DX Hell or WSPR modes, these modes all have a defined timing which cannot be changed.



```
Frequency 10,140,050
```

This frequency sets the baseline RF output frequency (at zero FSK).



```
Callsign  G0UPL
```

The callsign setting is an alphanumeric setting, which may be up to 14 characters long. It can theoretically include punctuation but clearly that isn't normally used in a callsign. The callsign is NOT used for the normal message transmission in most modes. It is used only for TWO purposes:

1) The kit may be configured to automatically send your callsign in standard 12wpm CW at pre-programmed intervals, as a station identifier (read your license conditions)

2) The Callsign is used for generating the WSPR message encoding

Note that when using the WSPR mode, the callsign must satisfy certain rules. Firstly, it can only be a 5 or 6 characters long. The callsign must consist of the following:

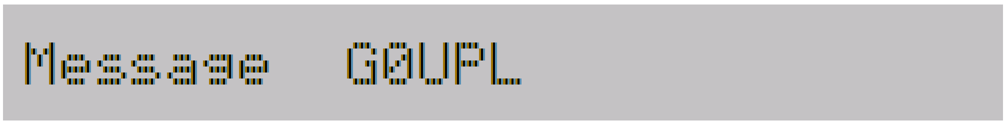
1) One or two characters consisting of A-Z or 0-9

2) One character which must be a number, 0-9

3) Three characters which must be A-Z or space

These restrictions are placed on the callsign in order to satisfy the requirements of the WSPR encoding algorithm. If you are using the WSPR mode, and the callsign violates the above requirements, the program will try to use the contents of your Message setting, to see if that contains a callsign. If the program cannot obtain a valid callsign from either location, then an error will result and be displayed on the LCD (Error: Message).

Note that if your callsign is a 4-character callsign then you need to add spaces at the right, to comply with condition 3 above. If your callsign is of the form AB0CD then you need to add one space at the right, e.g. “AB0CD “.



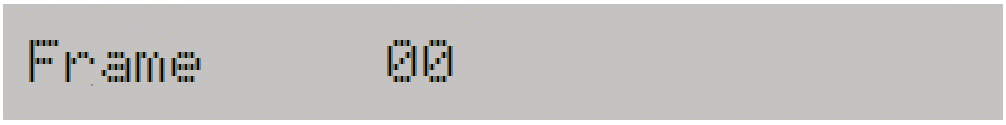
Message G0UPL

The message setting is an alphanumeric parameter which may be up to 250 characters long. For FSKCW, DFCW, CW, QRSS modes, it must be in the range A – Z, 0 – 9, or a space character. Hell modes can use punctuation characters as well, but not the * character.

Any message may be entered, but if the message is invalid for the mode being used, an error will be displayed.

The * character has a special purpose, it is used to signify the start and end of user-customisable pattern generation specification.

IMPORTANT: remember to add a space character at the beginning or end of your message! G0UPLG0UPLG0UPL... etc. is not easy to read, because it has no spaces. G0UPL G0UPL G0UPL is how you want your message sent, with a space between the messages.



Frame 00

This parameter is used to define the message frame length. A technique called “stacking” has been developed by QRSS'ers to improve signal-to-noise ratio even further. The transmitting station transmits messages defined into a frame of a precise length. The receiving station overlays multiple message reception frames on top of each other on the display. Any message which is repeating at that precise frame length gets re-enforced by the averaging of multiple message receptions; any other non-repeating message or QRM gets reduced.

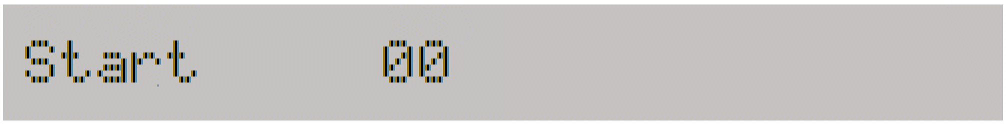
As an example, consider a message which in the chosen QRSS mode and speed happens to take 8 minutes to transmit. You could set the “Frame” setting to 10. This means that after the message transmission has completed, the microcontroller will wait until 10 minutes after the message start, before starting the next transmission. Receiving stations who have are using QRSS decoding software with the stacking feature, and have it set to 10 minutes frame length, will be able to collect six (say) transmissions over one hour and average them together to improve the signal-to-noise ratio.

This parameter is also used for the WSPR mode. In the WSPR mode, the Frame parameter defines how often the WSPR transmission packet is transmitted. WSPR transmissions take just under 2 minutes and are always aligned to the start of an even minute. Therefore in WSPR mode, the Frame parameter must be an even number (a multiple of 2), e.g. 10 minutes (9 minutes is not possible).

Typically, Frame should be set to 10 minutes for WSPR. 10 minutes is also a popular frame length for “stacking” QRSS reception, as long as your message transmission fits within 10 minutes. If it does not, the controller will wait until the next 10 minutes have elapsed.

When using GPS frequency stabilisation, you must choose a non-zero Frame parameter. This is because the frequency stabilisation must take place when there is no transmission from the kit, i.e. during a pause between transmissions. The frequency stabilisation takes 12 seconds and occurs just before the start of a transmission. Therefore you need to ensure that the Frame parameter is sufficiently large that there is a minimum gap of 12 seconds between the completion of your message transmission and the start of the next one.

Default: the default value is 00 which means no frame size is defined, and the stacking feature is not used. Frame must be set to a non-zero value for WSPR.

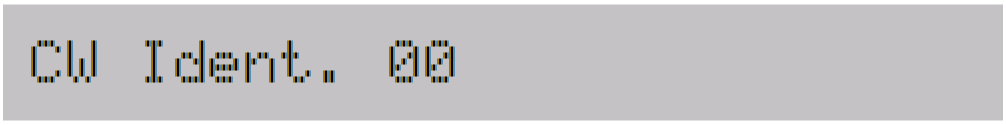


Start 00

This parameter is used in conjunction with the Frame parameter described previously. It defines when the transmission cycle starts. For example if Start is set to 03, and Frame is set to 10, then the message transmission will start on the when time reaches 00:03 (or 3 minutes past any hour), and will repeat at exactly 10 minute intervals.

Again for WSPR, the Start parameter must be even (a multiple of two).

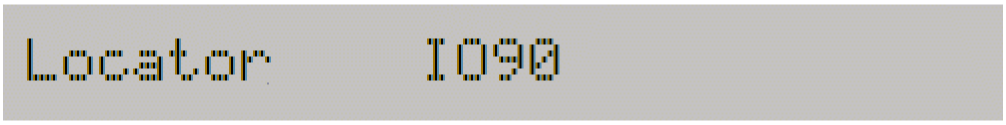
Default: the default value is 00 which means the Start parameter is not used.



CW Ident. 00

If the CW identifier feature is used, then this parameter should be set to a non-zero value indicating the number of minutes interval for sending the CW identifier. The CW identifier sends the callsign in standard 12wpm CW at regular intervals. If the CW Ident parameter is set, then a callsign must be entered in the callsign setting, otherwise an error will be displayed.

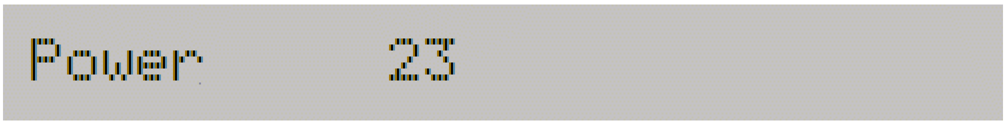
Default: the default value is 00 which means that a CW Identifier is not transmitted.



Locator IO90

The locator setting is only used in WSPR mode, and is one of the inputs to the WSPR encoding algorithm. It is four characters, the first two are always A – R and the second two are numbers in the range 0 – 9. Editing this parameter automatically prevents incorrect entries. The locator is not needed for any other mode than WSPR and may be left blank.

If a GPS module is connected with a correctly configured serial data stream, then the locator is set automatically from the GPS data.



Power 23

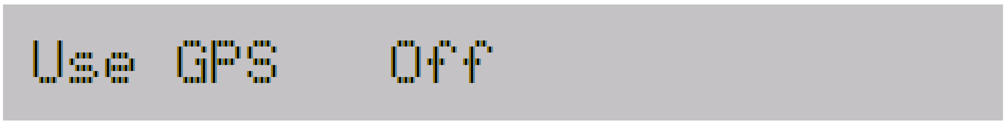
The power setting is another input to the WSPR message. This setting is not needed for any other mode than WSPR. Although the parameter will accept any 2-digit value, only certain values are valid for the

WSPR encoding scheme. The power is the transmitter output in dBm. It must be in the range 0 – 60, and the least significant digit must be 0, 3 or 7. The allowable values, with their corresponding power in watts, are listed here:

00 dBm	=	1mW
03 dBm	=	2mW
07 dBm	=	5mW
10 dBm	=	10mW
13 dBm	=	20mW
17 dBm	=	50mW
20 dBm	=	100mW
23 dBm	=	200mW
27 dBm	=	500mW
30 dBm	=	1W
33 dBm	=	2W
37 dBm	=	5W
40 dBm	=	10W
43 dBm	=	20W
47 dBm	=	50W
50 dBm	=	100W
53 dBm	=	200W
57 dBm	=	500W
60 dBm	=	1000W

The default value in this kit is 23dBm (200mW) which is closest to the measured prototype output power of 180mW.

An illegal value will result in a runtime error display.



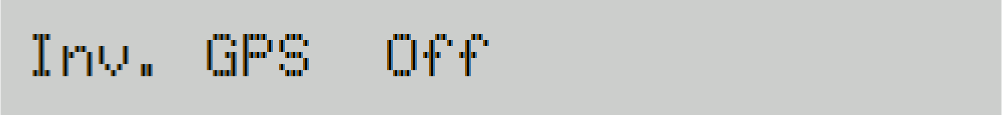
Use GPS   Off

This controls whether GPS is used or not. If set to On, and a GPS module is connected with 1pps signal, then the accurate time pulses from the GPS unit will be used to lock the frequency, and to keep precise time.

If additionally the serial data connection from the GPS unit is valid, the Maidenhead locator will be computed from the latitude and longitude received from the GPS module, and the internal realtime clock will be set using the time from the GPS module.

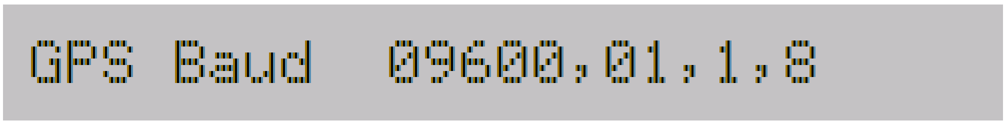
When using the connected GPS module for frequency stabilisation, you must choose a non-zero Frame parameter. This is because the frequency stabilisation must take place when there is no transmission from the kit, i.e. during a pause between transmissions. The frequency stabilisation takes 12 seconds and occurs just before the start of a transmission. Therefore you need to ensure that the Frame parameter is sufficiently large that there is a minimum gap of 12 seconds between the completion of your message transmission and the start of the next one.

Frequency stabilisation can also be provided by a 1pps signal derived from an accurate frequency reference. In this case, connect the 1pps signal to the 1pps connection, and set Use GPS to “On”. With no serial data, the kit will not decode the Maidenhead locator or time, however the 1pps signal **will** be sufficient for frequency stabilisation.



```
Inv. GPS  Off
```

Normally the 1pps signal triggers on the leading positive-going edge of the signal. If your GPS module has an inverted 1pps signal, then set Inv. GPS to “On”, in order to trigger on the trailing, negative-going edge.



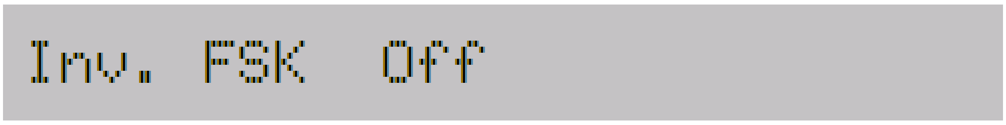
```
GPS Baud  09600,01,1,8
```

These settings determine the communication parameters for receiving a data input from the GPS module. They must be set to match the data output from the GPS module otherwise no data can be decoded. You will need to refer to your GPS module documentation to determine the correct speed and other settings.

There are four settings in the configuration string:

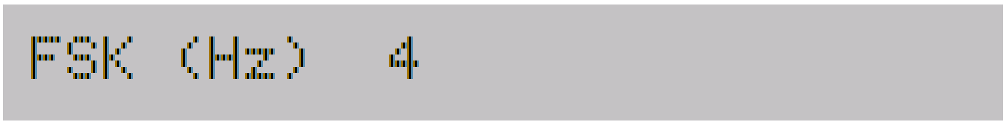
1) Baud rate: any baud rate can be entered here. 9,600 is common
2) Parity: the first digit is 1 to enable parity checking, 0 to disable it. The second digit is 0 for even parity, 1 for odd parity. Therefore the 01 setting shown above means disabled parity, odd parity.
3) Number of stop bits: this must be 1 or 2. In the example shown above, it is 1.
4) Character size, which technically may be in the range 5 – 9 though anything other than 8 is very unlikely where GPS is concerned.

The default setting shown above, is for 9,600 baud no parity, 1 stop bit and 8-bit data. This is a common setting.



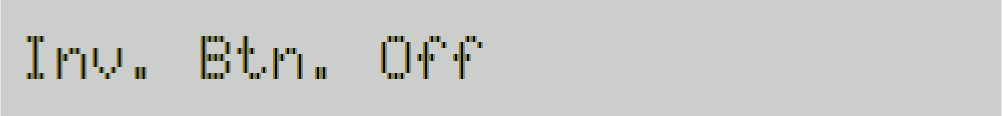
```
Inv. FSK  Off
```

This setting allows you to invert the direction of the frequency shift. It isn't normally used and should be left Off ordinarily.



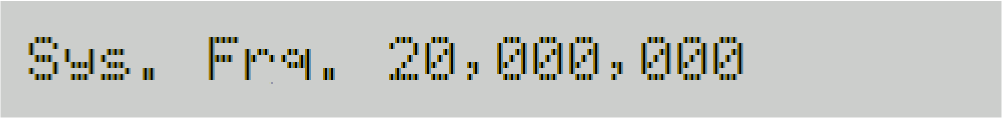
```
FSK (Hz)  4
```

FSK size is determined by this parameter. For FSKCW and DFCW modes, the specified FSK is the height of the symbol shift. For Slow-Hell mode, it is the height of the text. The parameter does not apply for the other modes.



```
Inv. Btn. Off
```

When this parameter is switched on, the left and right buttons are reversed. Refer also to the section above concerning LCD mounting options.



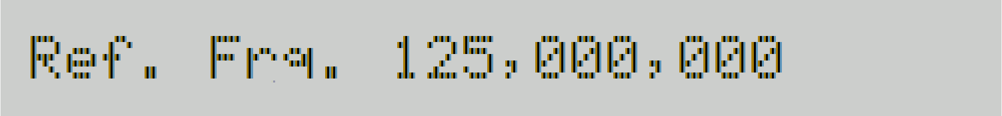
Sys. Frq. 20,000,000

The system clock frequency is the crystal oscillator frequency which clocks the microcontroller. In this kit, a 20MHz crystal is used, and there is no facility for adjusting its frequency – the trimmer capacitor sometimes used in such circuits is omitted here in order to keep the kit as simple and inexpensive as possible.

The system clock is used for timing purposes when there is no GPS module connected to lock the frequency. Where the frequency is slightly different to 20.000000MHz the time-keeping of the real-time clock will become inaccurate. Ordinarily this will not matter for most modes, the exception may be WSPR where the frames must coincide with the start of even minutes, within a minute or two. Over some days, an inaccuracy of a second or two per day in the real-time clock can cause the WSPR signal decoding to become impossible.

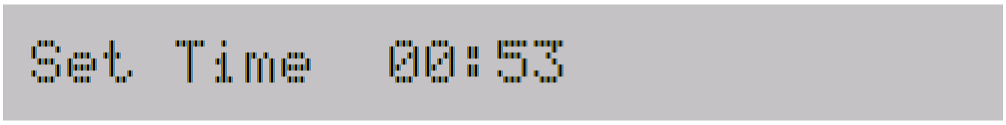
Accordingly this parameter has been provided so that if you wish, you can set the frequency to the actual exact frequency of your 20MHz crystal. This will ensure that time-keeping is precise. The best way to measure the oscillation frequency will be to use a general coverage receiver having an accurate digital display, and listen for the 20MHz signal on the receiver, then tune the receiver to zero-beat, i.e. until the tone of the audio decreases to 0Hz. Then enter this frequency into the Sys. Frq. configuration setting.

An alternative method may be to measure the time gain or loss over a period of several days, and work backwards to determine the required value to use for the Sys. Frq. Parameter to correct it.



Ref. Frq. 125,000,000

The Reference Frequency setting is the actual frequency of the on-board 125MHz canned crystal oscillator module on the pre-assembled DDS module board. It is important to set it accurately, otherwise the actual output frequency of the kit transmission will not match the value set in the Frequency parameter (see above). Calibration of this parameter is discussed in section 9, see below.



Set Time 00:53

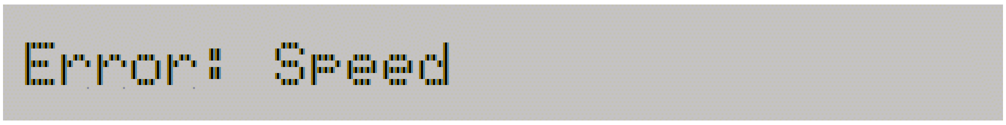
The kit has an internal real time clock which is used for the Frame (QRSS stacking), CW Ident, and for timing WSPR messages. This configuration allows you to set the real time clock. The clock setting takes effect, and the seconds (internally counted but not displayed) are set to zero, when the LEFT button is pressed after setting the minutes. **Accurate timing is critical for the WSPR mode.**

Note that if using a GPS module with "Use GPS" switched on, and if there is a valid serial data stream properly decoded (by correct choice of "GPS Baud" parameters), then there is no need to set the time, because it is automatically decoded from the GPS data.

An accurate real time clock is not required other than for the WSPR mode. Even if the real time clock is not set, it will start from 00:00 when the system is powered up. This is sufficient for the production of CW identification (if enabled) and stacking if enabled (i.e. if Frame is non-zero).

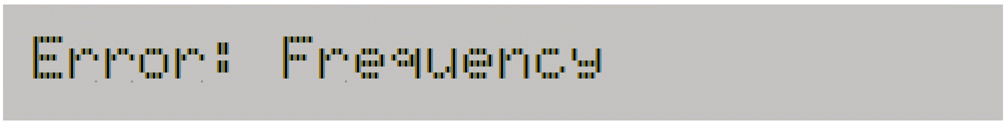
6.4 Understanding error messages, and how to fix them

In the event of inconsistent parameter setting, an error message may be displayed on the LCD. The cause of these messages is well-defined. The following lists the possible error messages and their causes.



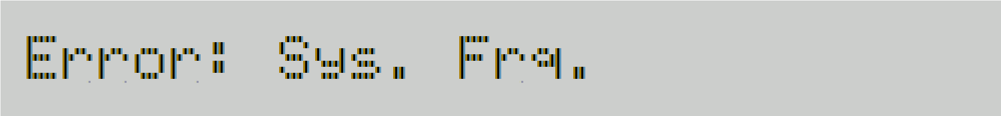
Error: Speed

This error results from having a zero Speed parameter, or Speed > 200 seconds, in any mode other than Hell, DX Hell and WSPR. These three modes have specific defined speeds which are not adjustable, and so the Speed setting is ignored for them.



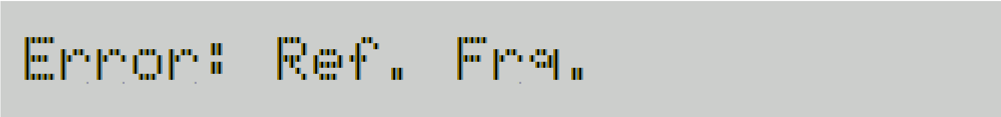
Error: Frequency

The Frequency parameter is set to a value less than 500. You should enter a valid value. The purpose of this minimum is to prevent excessive heat dissipation in the PA transistors in the event that the output frequency is zero or too low.



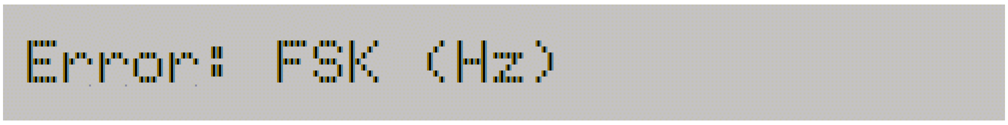
Error: Sys. Frq.

The system frequency is below 1,000,000. There should not generally be any reason why you would need to set a system frequency to any other value than close to 20,000,000.



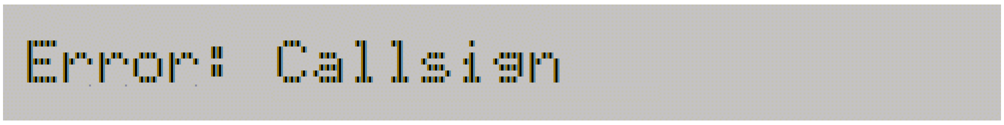
Error: Ref. Frq.

The reference frequency is below 1,000,000. There should not generally be any reason why you would need to set a reference frequency to any other value than close to 125,000,000.



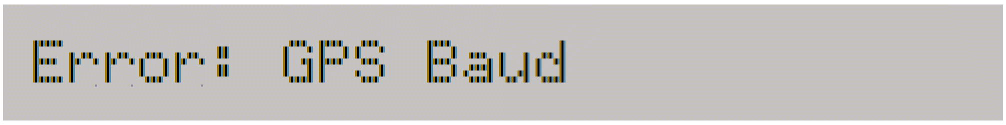
Error: FSK (Hz)

In modes FSKCW, DFCW, Slow Hell and TEST, the FSK size must be set. This error message indicates zero FSK (Hz) parameter setting.



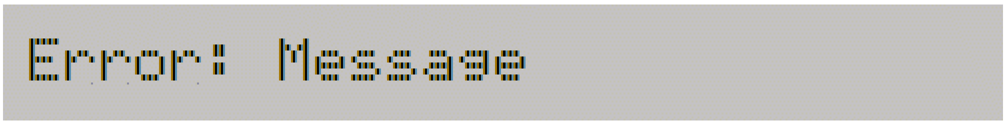
Error: Callsign

This error arises when the CW Ident parameter is non-zero, which means you have requested a CW Identifier transmission in plain 12wpm CW at a periodic interval, but the Callsign has not been set. The resolution to this error is to enter your callsign in the Callsign parameter.



Error: GPS Baud

This error can only occur if the “Use GPS” configuration is On and if the configured Baud rate is less than 1,000. The solution is to correctly configure the Baud rate setting.



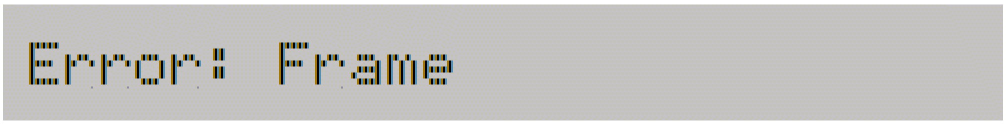
Error: Message

This error concerning the transmission message can occur for three reasons:

1) The message is empty, and the mode is set to anything except WSPR. In WSPR mode, the message encoding is generated automatically from the callsign, locator and power settings. In all other cases, the message must be set, otherwise the kit has nothing to transmit.

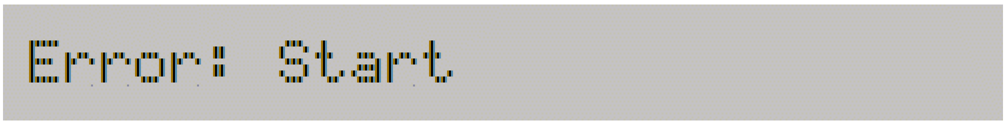
2) In WSPR mode, if the Callsign is empty or invalid, then the WSPR encoding algorithm will try to use the main message. If the Message parameter also does not contain a valid callsign for WSPR, then this error display will occur. To resolve this: study the requirements for a valid callsign in the configuration settings section above, and ensure that the Callsign parameter is set with a valid WSPR callsign.

3) In modes QRSS, CW, FSKCW, DFCW, there is an invalid character in the message. Valid characters are A – Z, 0 – 9, space and /. in the case of FSKCW the message may also contain an asterisk * which enables the user-customised pattern mode.



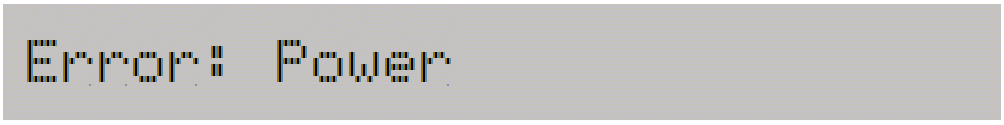
Error: Frame

In WSPR mode, Frame (number of minutes between transmissions) must be even (a multiple of two), and it must be non-zero. 10 [minutes] is the recommended value.



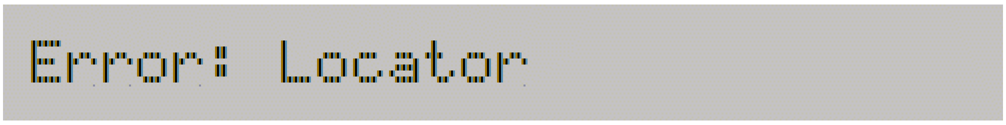
Error: Start

In WSPR mode, the Start parameter must be an even minute (power of two). In all modes, the Start parameter must be less than 60 (as there are only 60 minutes in an hour).



Error: Power

This error can only arise during WSPR mode. The Power setting must be in the range 00 – 60 and the last digit must end in 0, 3 or 7. i.e. 00, 03, 07. 10, 13, 17 are all Ok.



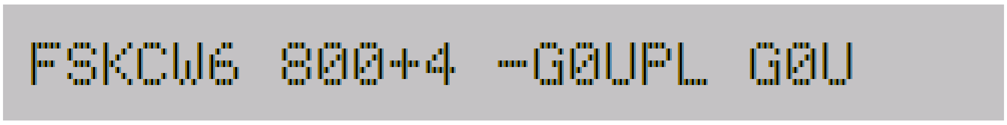
```
Error: Locator
```

Again, an error which can only arise in WSPR mode. The locator must be a standard 4-digit Maidenhead locator (e.g. IO90). If you are getting this error, it means you have not properly set the Locator configuration parameter – it should consist of two letters in the range A – R followed by two digits in the range 0 – 9.

7. QRSS Modes

The following is a short description of each kind of QRSS mode, and what to expect.

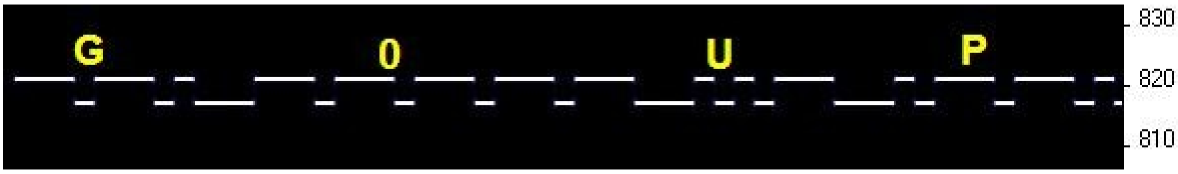
7.1 FSK/CW



```
FSKCW6 800+4 -G0UPL G0U
```

The display will show FSKCW followed by the symbol length – in this example 6 seconds. Next comes the base frequency. In AF Output mode, that is Hz. In RF Output mode, it is the last three digits of the frequency – e.g. if the Frequency parameter is set to 10,140,050 then the display will show 050. The FSK shift in this case, from the "FSK (Hz)" setting, is 4Hz – indicated by the +4. Finally the message will show. The leftmost character is the one currently being sent, and the – before it indicates a dash is being sent. As the message is sent, the message will scroll to the left.

In the following image, the decoded letters have been drawn above the received signal in yellow, to show clearly how the mode is to be decoded.

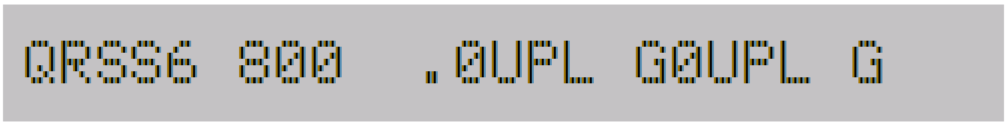


In Frequency Shift Keyed CW (FSK/CW) the dahs and dits have the same timing as traditional (slow) morse i.e. QRSS, but the carrier is always on and "key down" is indicated by a slight upwards shift of a few Hz. FSK/CW is the most popular of the slow-signal modes.

Good for: The transmitter being continuously on, avoids any "chirp" (frequency pulling) of the oscillator, that can be a problem, particularly in more simple circuits. The signal is very readable in the presence of QRM or in weak conditions, because of the continuously present carrier. Decoding the signal is intuitive, just by reading the top line like ordinary CW.

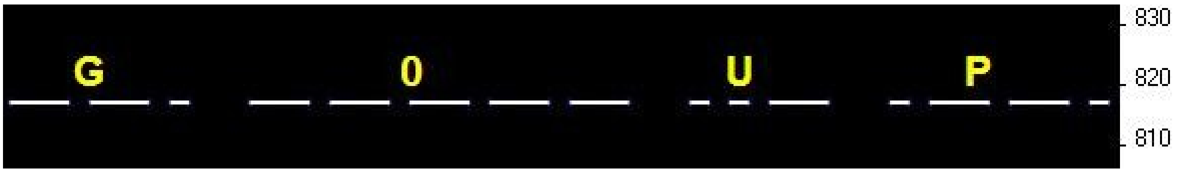
Bad for: Because the carrier is continuously on, it will take more battery power (if that's a concern), than an on/off keyed mode like plain CW. It also takes more bandwidth than the single sub-Hz bandwidth of plain QRSS.

7.2 QRSS



```
QRSS6 800  .0UPL G0UPL G
```

Here the display shows the scrolling message, with a – or a . (for dah and dit), to show which character and symbol is currently being transmitted. The dit speed follows the mode (QRSS), in this example 6 second dits. The frequency is as for FSK/CW (see above) – in this case 800Hz audio output.

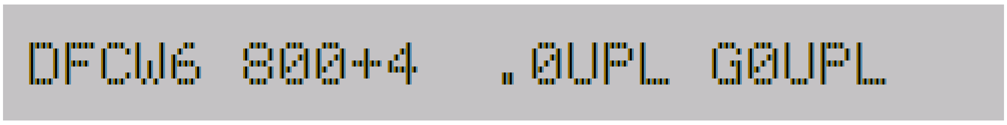


The original plain QRSS mode is simply plain CW, but massively slowed down, typically with dit speeds from 3 seconds or slower, even all the way to 120 seconds in some cases.

Good for: Simple to decode by eye, since it is the same was what we can imagine hearing. Very narrow bandwidth and band occupancy, sub-1Hz.

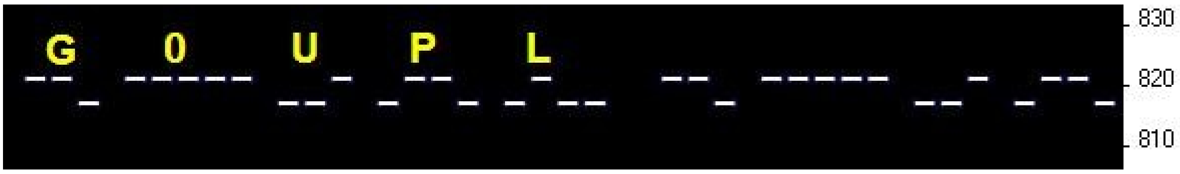
Bad for: In weak signal conditions or where there is a lot of QRM, an interfering weak carrier may also look like an intermittent horizontal line, so the wanted signal can be hard to distinguish from the QRM. Fading (QSB) can cause interfering carriers to look like QRSS too. In simpler circuits, “chirp” (oscillator pulling) can be a problem.

7.3 DFCW



DFCW6 800+4 .0UPL G0UPL

The display will show DFCW followed by the symbol length – in this example 6 seconds. This is followed by the base frequency (see FSK/CW section above) and the FSK size. The scrolling message is shown as usual, with the – or . to show the current symbol of the current character.

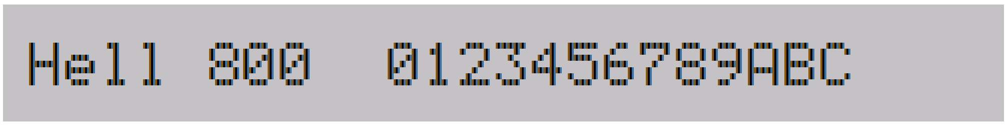


DFCW means dual-frequency CW. The dits and dahs are the same length, but dahs are shifted upwards by the FSK amount. A 1/3 dit-length gap inserted between symbols improves the readability.

Good for: The mode is faster than QRSS, with the same signal-to-noise ratio improvement, because the dah's are the same length as the dits. In QRSS dah's are three times longer. So the message transmission is faster.

Bad for: Not very intuitively easy to read as QRSS is. If the signal is a bit weak and hard to decode, the eye/brain may have more trouble “filling in the gaps”, because it is not so intuitive.

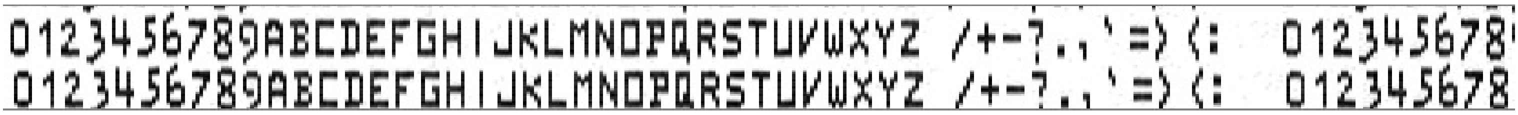
7.4 Hellschreiber (also called FeldHell).



Hell 800 0123456789ABC

As usual, the display shows the mode, frequency and the message scrolls to the left as it is sent.

The screenshot fragment below is from IZ8BLY's excellent Hellschreiber receiving (and sending) software.



0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ /+-?.,'=)(: 012345678

0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ /+-?.,'=)(: 012345678

Hellschreiber is a fax-like mode designed in the 1920's by Rudolf Hell in Germany. It was later used in WWII. The literal translation into English would be “Light Writer” and of course rhymes with its inventor's name. Each character is made up of a 7-row, 5-column grid. Technically there are 14 half-rows, but there are no lonely half-pixels in the characters, half-pixels are only ever seen in groups of at least two. This was an ingenious idea to minimise transmission bandwidth yet improve readability. The baud rate of a standard Hell transmission is 122.5. Most of the characters fit into a 10-half-row, 5-column grid, but the numbers are extended above and below this in some cases.

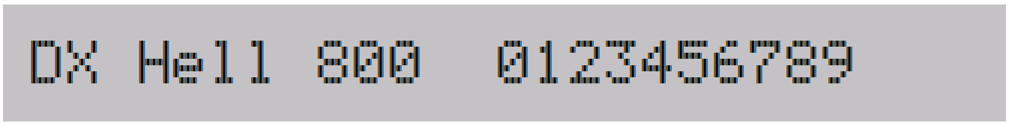
Good for: Fun, history, and easy readability.

Bad for: Bandwidth. Hell is not a slow-signal mode; the bandwidth is said to be around 400Hz.

I am proud to be using the original Hell font in this kit as carefully designed by Rudolf Hell. I know this because Pierre ON5SL, who owns a real antique FeldHell machine, kindly sent me a scan of one page of the operator's manual. This page shows the drum pattern of the machine.

Various software is freely available that can send and receive Hellschreiber, using any Windows font. Specially designed fonts are available. One called “FeldReal.fon” claims to use the original Hellschreiber character design but I am not convinced – if you are interested, look for example, at the K. FeldReal.fon's K actually looks nicer, but it doesn't quite match the one in the machine, look at the top right of the K.

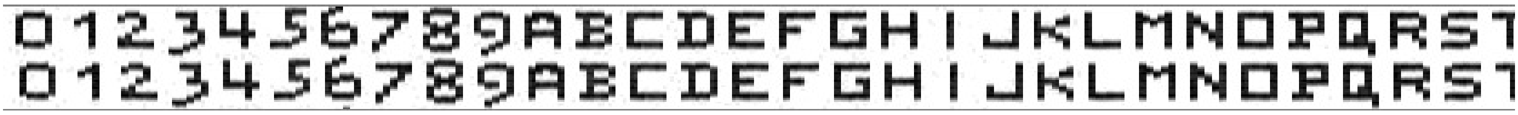
7.5 DX Hellshreiber



DX Hell 800 0123456789

As usual, the display shows the mode, frequency and the message scrolls to the left as it is sent.

The screenshot fragment below is from IZ8BLY's excellent Hellschreiber receiving (and sending) software.

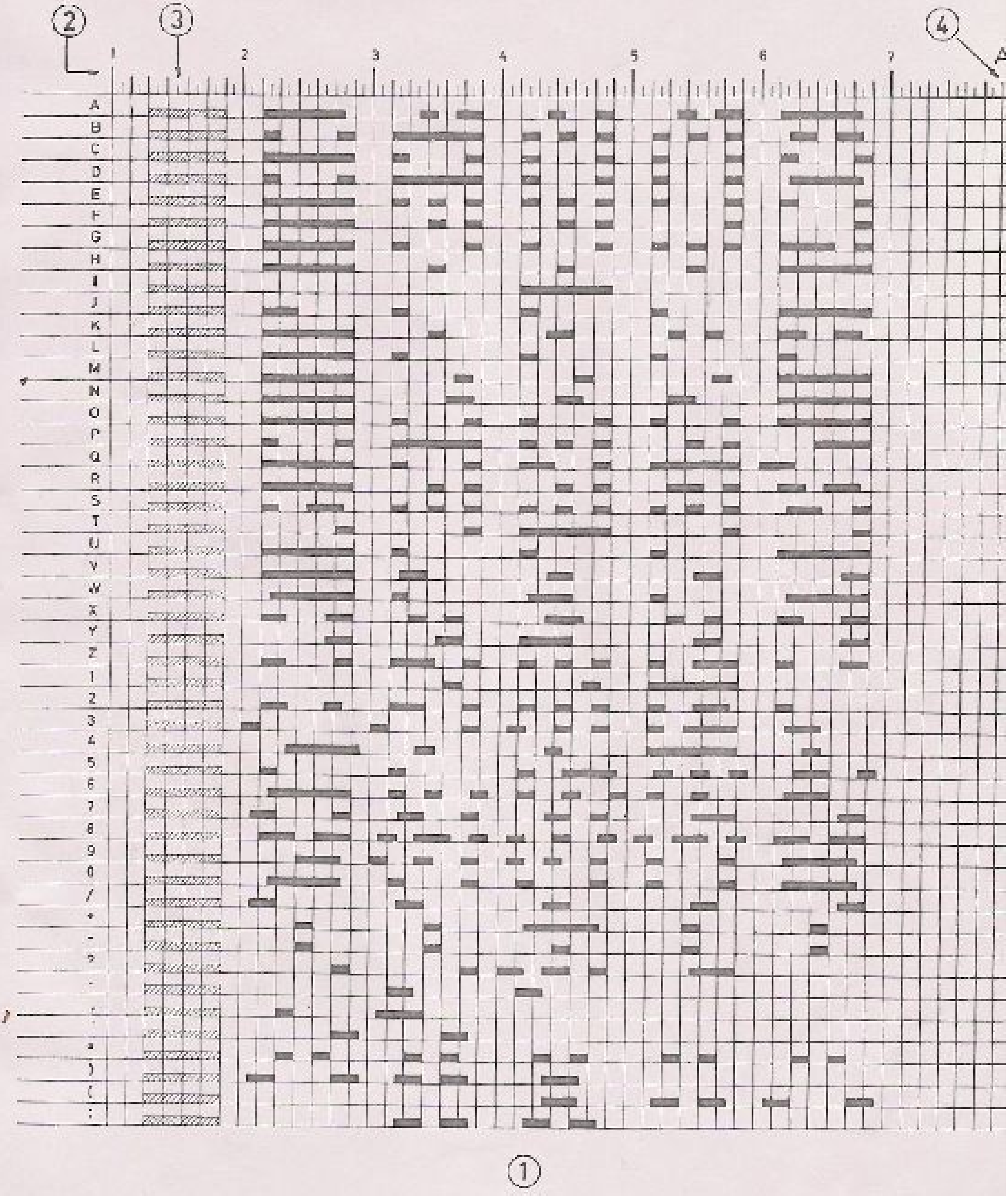


0123456789ABCDEFGHIJKLMNOPQRST

0123456789ABCDEFGHIJKLMNOPQRST

So-called DX Hellschreiber is the same as ordinary Hellschreiber described above, except that every column is sent twice. The characters are therefore twice as wide, the message takes twice as long to send, but it has the advantage of potentially being more readable in weak conditions.

Page from a real Hellschreiber machine owned by Pierre ON5SL, showing the character drum.



15

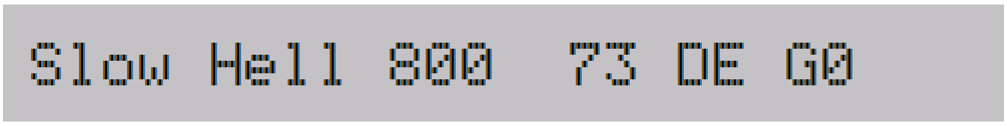
① Abwicklung der Sendenocken

② Einteilung der Meßscheibe

③ Anlaufschritt

④ Nullstellung der Senderachse

7.6 Slow Hellshreiber



Slow Hell 800 73 DE G0

As usual, the display shows the mode, frequency and the message scrolls to the left as it is sent.

Slow Hellschreiber is the slow-signal Hellschreiber equivalent. Each Hellschreiber letter row is shifted slightly in frequency. Typically the single pixel rate is very slow, for example 1 second per pixel (0.5 seconds per half-pixel). The character pattern is “scanned” one column at a time from bottom to top, shifting the frequency as the row increases up the character. In this way an image of the character is built up at the receiving station.

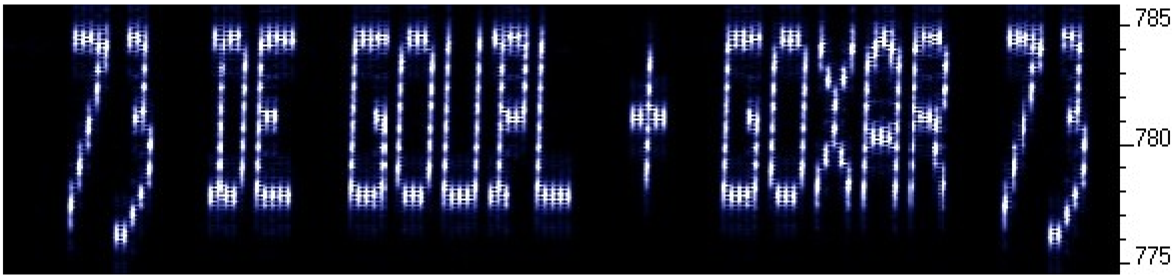
In Slow Hellshreiber mode, certain configuration settings in the kit require special attention:

Speed: The speed setting is the number of seconds required to transmit one whole character. Recall that a Hell character is composed of 7 rows (14 half-rows) and 5 columns; there is also one empty column as an inter-character space. Therefore each character is composed of 42 “pixels”. Therefore if you want the transmission time to be 1 second per pixel, set Speed to 042. If you wanted 1.5 seconds per pixel, you would set Speed to 063, and so on.

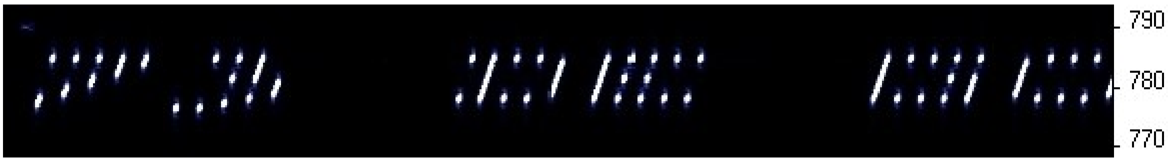
FSK (Hz): The FSK size configuration specifies the height of each character, it is the number of Hz for 5 rows of the character. Recall that most Hell characters, such as the letters, fit on a 7-column, 5-row grid. FSK (Hz) specifies the size of that grid; some of the numbers extend above or below the grid. For example an FSK (Hz) setting of 05 would result in characters 5Hz high, i.e. 1Hz per row (pixel).

Be aware of fundamental limitations of information theory. If the pixel rate is 1 second (Speed = 042), the minimum theoretical bandwidth is 1Hz. Therefore choosing a pixel size of less than 1Hz, i.e. FSK (Hz) = 05, does not make any sense. Faster transmissions require larger bandwidth, and you will have to set the FSK size higher to match. If you do not, the image at the receiving end will be blurred.

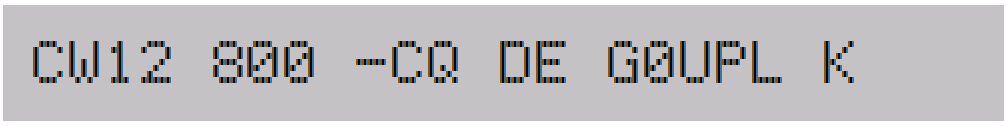
This example image was taken in Argo, in its 10 second “slow” mode. The message is “73 DE G0UPL + G0XAR”, the Speed is 042, and FSK (Hz) is 08. Note that even here, the characters start to look slightly blurred – a consequence of the fourier transform and minimum bandwidth, because the bucket size in Argo's 10 second mode is well under 1Hz so the 1Hz.



The following image shows part of the same thing in Argo's 3 second “slow” mode. The larger fourier transform bucket size means the image is more clear, but the faster scroll rate makes the letter's height to width ratio somewhat less attractive. Other slow-speed fourier transform software present more configuration options and may be more suited for slow-Hellshreiber reception than Argo.



7.7 CW



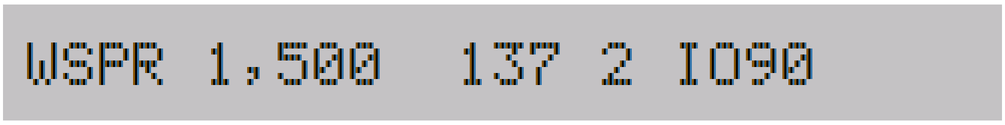
```
CW12 800 -CQ DE G0UPL K
```

CW mode is just plain, ordinary on/off keyed CW (morse code) at ordinary speeds. The display will show CW followed by the speed – in this example, 12 words-per-minute (wpm), followed by the message which scrolls to the left as it is sent. The character position right before the message shows – or . depending on whether a dah or dit is being sent.

In CW mode, the Speed setting is the words-per-minute speed, e.g. 012 for 12 wpm.

Have you ever spent a lazy Sunday afternoon calling CQ on 80m CW with QRP power, with no luck? Why not use the kit as a CQ machine, and take the hard work out of calling CQ! In AF output mode, the AF output could be rectified and used to key a CW transmitter, with the Frame setting used to control how often the CQ message would be transmitted.

7.8 WSPR



```
WSPR 1,500  137 2 IO90
```

The WSPR mode display is composed of several elements. Given the example display above:

WSPR	The mode being used!
1,500	Baseline output frequency (last few digits only)
137	Symbol number currently being sent; WSPR messages are made of 162 symbols
2	Tone currently being sent – WSPR messages are made of four tones, 0 – 3.
IO90	Maidenhead locator setting coded into the WSPR message

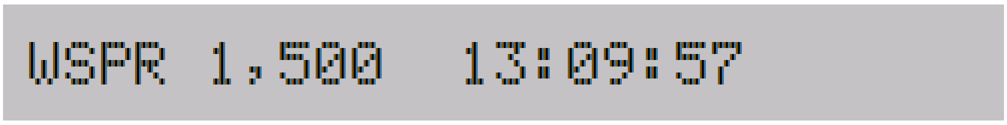
WSPR stands for Weak Signal Propagation Reporter. The message consists of three parts: the operator's callsign, Maidenhead locator (4-character, e.g. IO90) and two digits specifying the power. At the receiving station, messages are decoded and uploaded to a central internet database.

The WSPR message is encoded into a set of 162 symbols, each may be 0, 1, 2 or 3, using a compressed data format with forward error correction. The symbols are transmitted as tones, each tone separated by 12,000 / 8,192 Hz i.e. approximately 1.46Hz. The duration of each symbol is the reciprocal of the tone spacing, which is approximately 0.683 seconds. WSPR messages take about 110.6 seconds to transmit, and always start at even minutes past the hour.

In WSPR, timing is critical, so when using WSPR you must set the time configuration parameter as exactly as possible. Be sure to keep the rightmost flashing 1-minute digit, watch your clock until the seconds turn over to 00, and then press the left button. This will ensure the seconds are in sync with your real clock time. If careful attention is given to setting the frequency and the real time clock, then successful WSPR reports will be obtained. Of course these things are easier if you are using a GPS module: the Maidenhead locator will be calculated from the received latitude and longitude, and the time decoded nicely from the GPS serial data stream.

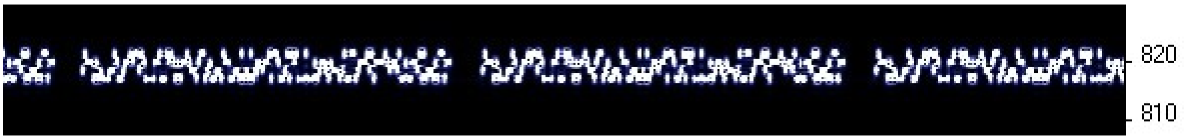
The microcontroller in this kit takes care of the WSPR message encoding algorithm, without any assistance from a PC host computer. It also calculates the tone spacing and symbol duration.

In between message transmissions, the display will show instead just a clock (see below), while we wait patiently for the next WSPR transmission to begin, according to the settings of the configuration parameters Frame and Start. This is useful for checking that the time on your kit is accurately set:

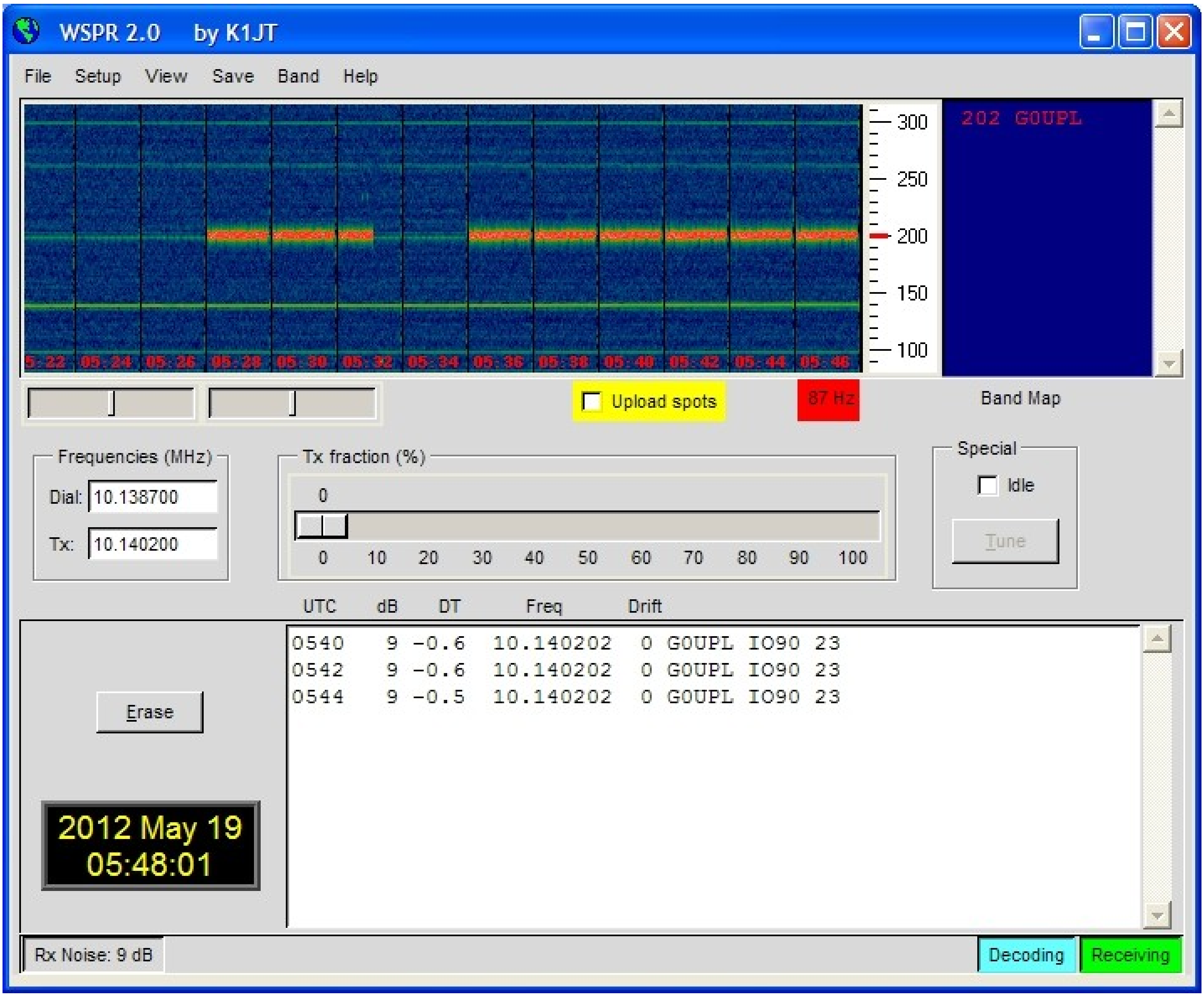


WSPR 1,500 13:09:57

On Argo, WSPR messages look ugly, like this:



WSPR decoding takes place in the WSPR program by K1JT (see http://physics.princeton.edu/pulsar/K1JT/wspr.html). This is a screenshot showing the WSPR 2.0 screen following reception of a few transmissions (output frequency = 1,500Hz, Frame = 02, Start = 00).



Frequencies (MHz)

Dial: 10.138700

Tx: 10.140200

Tx fraction (%): 0

```
UTC   dB   DT      Freq      Drift
0540   9  -0.6  10.140202   0  G0UPL IO90 23
0542   9  -0.6  10.140202   0  G0UPL IO90 23
0544   9  -0.5  10.140202   0  G0UPL IO90 23
```

2012 May 19
05:48:01

Rx Noise: 9 dB

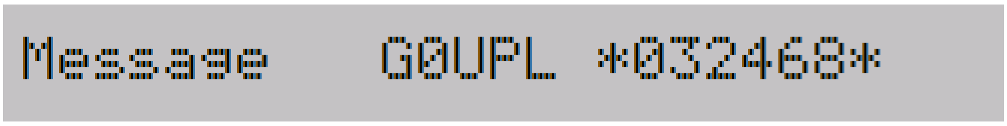
Choose a transmission frequency somewhere in the 200Hz-wide standard HF WSPR bands:

160m:	1.838000 – 1.838200	20m:	14.097000 – 14.097200
80m:	3.594000 – 3.594200	17m:	18.106000 – 18.106200
60m:	5.288600 – 5.288800	15m:	21.096000 – 21.096200
40m:	7.040000 – 7.040200	12m:	24.926000 – 24.926200
30m:	10.140100 – 10.140300	10m:	28.126000 – 28.126200

8. Customised message patterns

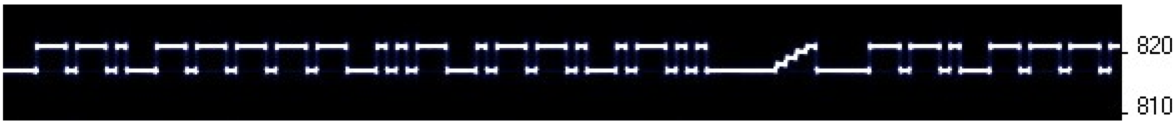
The kit has the ability to produce customised message patterns in the FSKCW mode. To enter or leave custom mode, a * character is inserted in the message. The first two characters after the * character specify the symbol duration of each symbol, in seconds. Subsequent characters specify the frequency shift of each symbol one by one, in units of 0.5Hz. For example, 5 means a shift of 2.5Hz. After the numbers, you can continue with letters. So A would mean a shift of 5Hz, B a shift of 5.5Hz, C a shift of 6Hz, and so on.

For example consider the following message:

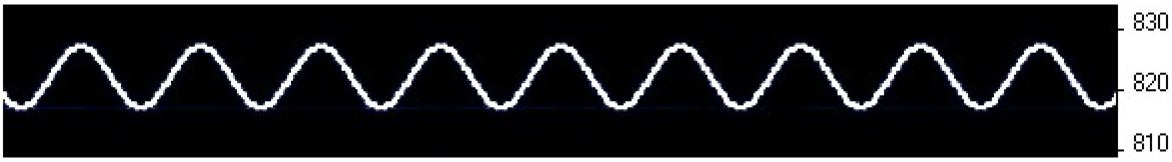


Message   G0UPL *032468*

The callsign G0UPL is sent in ordinary FSK/CW with height and speed according to the configuration settings. Next comes a custom message, the duration of each symbol is 03 seconds. The shifts are 1, 2, 3, 4Hz respectively. On Argo, this message produces (with kit configuration FSK (Hz) = 04, Speed = 003):



As an example of what kinds of patterns can be produced, the pattern below is created using message *01ACDFGIJJKKKJJIGFDCA87542110001124578*



9. Calibration

This section is REALLY IMPORTANT! It is very important to realise that by far the hardest part of any successful QRSS beacon operation, is tuning the oscillator to the correct frequency. For example, most 30m QRSS stations monitor a narrow 100Hz-wide band from 10,140,000 to 10,140,100. If you are much outside this, the chances are that nobody will see your signal. It is therefore essential either to adjust your output frequency using an accurately calibrated frequency counter, or an accurately calibrated receiver.

If you are going to use a GPS receiver module to provide a 1pps signal to the kit, no calibration is required: the kit will self-calibrate from the 1pps signal.

In the case that you will not be using a GPS signal, you need to calibrate the kit by entering the correct value of the 125MHz reference oscillator frequency into the “Ref. Frq.” configuration setting.

It is possible that you have an accurate means to measure the 125MHz reference oscillator frequency itself by probing the correct point on the DDS module. However most kit builders will not have the necessary equipment, and such a measurement at VHF is not easy to make without itself risking disturbing the frequency.

The easiest method is to set the output frequency to something convenient such as 12.500000 MHz, and measure it. Measurement can either be by using an accurately calibrated frequency counter, or by setting up an accurately calibrated receiver with Argo and monitoring the output signal frequency that way.

Once you have measured the actual output frequency, you can calculate the required correction to the 125MHz reference frequency and enter it in the “Ref. Frq.” configuration setting. For example, suppose you set the output frequency to 12.500000 MHz but you actually measure 12.500075. Your output frequency is 75 Hz too high. Since your output frequency setting is 10 times less than the 125MHz reference oscillator frequency, this means that the actual reference oscillator value is 750Hz too high. So enter 125,000,750 in the “Ref. Frq.” setting.

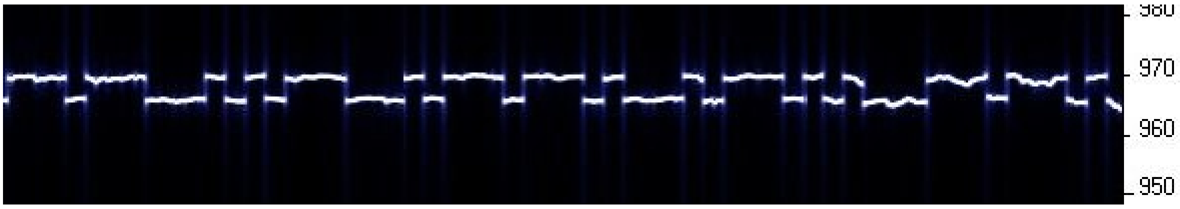
10. Factory Reset

On powering up the kit, a message is shown for a few seconds on the LCD, which indicates the version number. If the Right button is pressed during the display of this splash screen message, then the chip EEPROM contents are erased and returned to the original factory settings.

The effect of this feature is to entirely return the microcontroller chip to its original settings, it is the same as buying a new chip and inserting it. All calibration settings, message, frequency, etc. are erased and returned to the default settings.

11. DDS Module Stability

Here's an extremely important tip regarding the stability of the DDS module. If you just wire everything up and go on air, you would surely be ashamed to see how horrible your signal will probably look:



Obviously the DDS module was designed for something else, not just our QRSS passion, and the stability at Hz-level wasn't important. My own theory is that there are temperature sensitive components within the 125MHz crystal reference oscillator, which take time to all reach the same temperature. When they do, the temperature coefficients largely cancel each other out. But in the short term, while the different components are experiencing slightly different temperatures, the frequency can shift considerably. I believe that even small air currents over the oscillator case are enough to alter the local temperature and cause frequency instability.

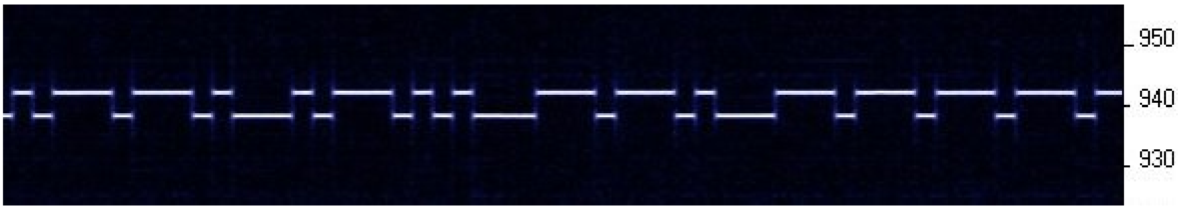
I have found that in practice there are two easy ways to solve this problem. Preventing the instability is very important before putting the transmitter on air.

1) If you put the kit into an enclosure, such that you've removed any air currents around the crystal oscillator, the frequency instability disappears.

2) Attaching (e.g. with glue) a small heatsink to the crystal oscillator can also cures the frequency instability. I found that almost any heatsink will do. Even take the smallest most pathetic coin in your country and place it on top of the crystal oscillator, and the frequency instability is cured. In fact even non-metal will do it – a piece of plastic, for example – which also adds weight to the air current theory, since anything you put on top of the crystal oscillator stops air currents altering the temperature so easily.

Arguably a heatsink may be a good idea in any case, to slow down temperature changes of the crystal oscillator anyway. Either way: just don't leave the crystal oscillator in the open air.

Below see the result of placing a coin on the crystal oscillator.



12. Signal-generator mode

Another use of this kit, is as an accurate, stable signal generator. If you set the kit in the FSKCW mode, and set the FSK to 0 Hz, then whatever message you've entered, the output will always just be a single un-modulated signal at the chosen frequency.

13. Resources

Please see the kit page http://www.hanssummers.com/qrssultimate2 for any information on latest updates and issues.

Use the QRSS Knights mailing list: http://cnts.be/mailman/listinfo/knightsqrss_cnts.be to announce your operation.

You should get reception reports by email and see your signals on the various online “grabbers” (see http://digilander.libero.it/i2ndt/grabber/grabber-compendium.htm for a set of links to “grabbers” worldwide).

Please join the Yahoo group http://groups.yahoo.com/group/qrplabs/ to for new kit announcements, to discuss any problems with the kit, enhancements you’ve made, or just to tell everyone how much fun you’re having.

For general information and more QRSS projects, links to other QRSS resources etc., see http://www.hanssummers.com.

Argo QRSS decoding software by Alberto I2PHD: http://www.weaksignals.com/

WSPR network home page: http://wsprnet.org

Analog Devices DDS information: http://www.analog.com/en/rfif-components/direct-digital-synthesis-dds/products/index.html

14. Version History

1 24-Mar-2013

- Initial draft version, for firmware version v2.00 and beta testers

2 20-Apr-2013

- First official version, accompanies firmware version v2.01
- Bug fix: after frame wait, frequency didn't return to correct frequency until tone changed
- Bug fix: WSPR did not work with any system clock frequency > 16,777,216 Hz
- New menu item "Inv. GPS" for triggering on a negative edge 1pps signal

- Impose limit of 200 seconds on the speed setting, to prevent arithmetic overflow
- Pressing right button during Run resets frequency and forces key-up