W0EB/W2CTX
V4.01R I2C/Digital I/O Firmware instructions for the Micro BITX meter Transceiver

Version 4.01R software instruction manual for an I2C display equipped uBITX transceiver, written by Ron Pfeiffer, W2CTX and Jim Sheldon, W0EB. This software (with minor wiring changes) will operate on a standard Raduino card that has been modified to allow for I2C display operation as well as our new "RadI2Cino" (pronounced Rad ee too CEE no) card which was designed to replace the original uBITX "Raduino" card.

It will NOT work on an original Raduino unless that Raduino has been modified to use an I2C display. If you do use it on a modified original Raduino, you must make sure that you remove any pullup resistors from the display controller itself. The most popular controllers use 4.7K resistors between the SDA and SCL lines and the incoming 5 volts. These are usually labeled R8 and R9 on the controller board. If these are left in, the Si5351 on an original Raduino card will see 5 volts on the I2C bus and most likely will be destroyed as the 5351 is a 3.3 volt device.

On the RadI2Cino board this removing these resistors is not necessary as a TXS0102 Level Translator IC has been included to protect the Si5351a.

INTRO:

uBITX - Necessary information for programming and utilizing this software:

You will need to unzip the ubitxI2C_V4_01R.zip file into your Arduino directory. This Zip file contains ALL the necessary Arduino Nano sketches to allow the program to be compiled and uploaded to the uBITX by the Arduino IDE program. You must also have the **LiquidCrystal_I2C** library

installed in the Arduino IDE's "Libraries" directory or the software will not compile. This is supposedly a "legacy" library but it IS necessary to have this exact one installed or our I2C software will not compile.

Since the I2C display's "backpack" controller will have a unique I2C bus address, and not all have the same address, you will need to know those numbers.

If the display controller supplier does not list the unique address for your controller (or display with an embedded controller) we have help for you.

Included in the zip file is a small Arduino sketch called I2C Scanner. To use it, connect your I2C display to the completed RadI2Cino via the I2C 4 pin header – (watch the polarity Pin 1=ground, 2=+5 volts, 3=SDA and 4=SCL.) Connect the Nano to your computer via the on board "Mini" USB connection, upload and run "I2C Scanner". (The Nano, I2C controller and the display will be powered through the Nano via the USB connection.) In the Arduino IDE, under "Tools", turn on "Serial Monitor", set it's baud rate to 9600 and you should see a "Serial Monitor" window open and start displaying all the I2C addresses it finds on the RadI2Cino's bus. There should be 2. 0x60 (Hexadecimal address 60) which belongs to the Silicon Labs Si5351A Clock/VFO generator IC and the other one will be the address of your display's I2C controller. The most common ones found have been either 0x27 or 0x3F (Hex 27 or Hex 3F). These aren't all of the possible addresses so be sure to write down, save and remember the display address that Scanner finds so you will have it for future software updates as you WILL have to change the address to match yours in EVERY I2C enabled software release. We have absolutely no way of knowing your display controller's address and the software uses the most

common 0x27 address. If yours is different, the display will stay blank until you change the address. How to do that is shown in the following steps.

Once you have the software directory unzipped, either using the Arduino IDE or an external program editor such as Notepad++, Look through the .ino source file until you find a line (around line 128-131 similar to the ones below -

//////////////////////////////// user defines their address of their I2C display
//LiquidCrystal_I2C lcd(0x3F, 2, 1, 0, 4, 5, 6, 7, 3, POSITIVE);
//LiquidCrystal_I2C lcd(0x2F, 2, 1, 0, 4, 5, 6, 7, 3, POSITIVE);
LiquidCrystal_I2C lcd(0x27, 2, 1, 0, 4, 5, 6, 7, 3, POSITIVE);

Notice the first numbers after lcd(are 0x3F, 2F or 27 which means hexadecimal number 3F, 2F or 27. This is the I2C bus unique address for the display controller module. You will need to know that address and if it is not 0x27, but something else such as 0x3F, you will need to change the 0x27 to 0x3F (or whatever your own device's unique address is.

The highlighted line above does not have "//" in front of it because that is the active line in the source code the // indicates a "comment" line and is not acted on by the compiler. For your convenience, the 3 most popular addresses are already set up. If you address is one of the 2 other than 27, put // in front of the line that doesn't have them (0x27) and take out the // in front of the line that contains the true address of your display, compile and upload the modified program to your card. This should make the display operate correctly.

Do not, repeat NOT, use address 0x60 as that is reserved for the Si5351 clock generator on the card and it will get all lost and confused.

Once you have gotten the correct display controller address into the source code, save it and then run the compiler to check the program for errors. If there are none, compile & upload the software to your RadI2Cino. Put the card back into the computer and connect the I2C cable to the display again, (make sure the polarity is right – you don't want to put +5 Volts on the wrong pin of the controller it might be destroyed. Power up the uBITX and make sure your display is working.

uBITX Operational Modes:

There are 2 operational modes available to the uBITX in this version of the software. These modes are selected during power-up via the encoder push button.

The two modes are: Normal and ALIGNMENT. (Alignment is the utility routine which allows you to calibrate the master oscillator and BFO frequencies against a frequency standard station received over the air or an external, well calibrated signal generator.)

SELECTING MODES:

Alignment Mode: The master oscillator and BFO calibration instructions are quite extensive so an entire section has been dedicated to the proper calibration procedure. See APPENDIX B for these alignment instructions.

Normal Mode: General radio operations. Power up your uBITX and do nothing. The normal radio features will be defaulted. This is the mode you will use to operate the uBITX on the air.

NORMAL MODE FEATURES:

Many people expressed the desire for a different method of changing bands and tuning speed as the original method was sometimes difficult to control.

Early on, we removed the accelerated tuning (“faster the knob is turned, the faster the frequency changes”) feature in the original factory software as the timing implemented the speed-up far too quickly for many people and it was hard to set an exact frequency if you happened to slip. The frequency would sometimes jump by 2-300 KHz and then you had to crank forever to get back to where you were. Instead, we changed the tuning to where a short press (tap) of the encoder button would change the position of an underscore cursor and allow you to select the digit of the frequency you wished to change. This allows for more precise quick setting of the frequency.

MENUS:

To enter the Menus, power up your uBITX and wait for it to boot up into the default which is “Normal” mode.

Now, press and hold the encoder button (about a second) until “Menu” appears. Immediately release the button and you should see the menu’s changeable information on the display.
For example:
CW NA St800 Ks25 D-150
XX,xV A 7.040.00

(If you get into the menus accidentally, just press and hold the encoder/function button until Exit Menu appears. Release the button and

you will return to normal operation with nothing changed.)
This means Normal Mode is set to CW.
The sidetone is set to 800 Hertz.
The keyer speed is 25 words per minute.
The return to RX delay (CW mode only) after key release is 150 milliseconds and this can be varied from 50 ms to 3000 ms (3 Seconds).
Power input voltage is XX.x volts, VFO A is active and the frequency is 7.040.00. The XX.xV in the 2nd line of the display is the uBITX input voltage which is optional.

The input voltage display may be turned on or off in the menu and a special voltage divider must be added to the Raduino or RadI2Cino card to even allow reading it. (Instructions for adding the Voltage monitor are covered in APPENDIX C.)

You may then use the encoder to move the underscore cursor to the various items in the menu you would like to change. First up is the operating mode and there are several.

For instance, the preceding example shows the operating mode as CW. To change to one of the other available modes, press & hold the encoder/function button until “Menu” appears – release the button, the cursor will be under the C in CW. Press and hold the encoder/function button for about a second and the word “Change” appears briefly. You can now rotate the encoder right or left to select the mode you want and once you have that mode showing, another short press of the encoder/function button will change to that mode and return you to normal operation in the new mode. The selectable modes are USB, LSB, CW, CWR (receive CW on the opposite sideband), SWU and SWL. SWU and SWL are special

“Short Wave Listener” modes which keep the PTT disabled so while in these 2 modes, you can’t accidentally transmit while using them.

To change the side tone frequency, enter Menu, rotate the encoder to put the cursor under the S in St and press the button. “Change” will briefly appear, and the current tone will sound in the headphones (or speaker if you have one). Rotate the encoder button right or left to adjust the tone to your liking and press the button again. The side tone will be set and you will be returned to the normal operating display. Changing the Keyer speed works exactly the same way.

Selecting the A or B VFO’s and SPLIT operation now requires the addition of an SPST (single pole, single throw), momentary push button switch connected between ground and the D10 digital I/O pin on the Arduino Nano. On the RadI2Cino card, a special header has been added to easily connect this switch.

All the other menu items are selected in the same manner.

You should experiment with changing each of the available items until you become completely familiar with how the system works. It was designed to be as intuitive as possible so we’ll leave it up to you, the user to play with it as an exercise in familiarization.

CW keying has been changed significantly from earlier versions of our software. The keyer module has been rewritten to make the CW mode completely “interrupt” driven as makes further enhancements to the software much easier to implement

Also, both the CW hand key/external keyer and push to talk functions now utilize the same digital I/O line. This has several advantages including

needing only one common connection for push to talk and the hand key/external keyer input.

Since we don't normally send CW in the SSB mode, the input is used for the push to talk line and grounding it puts the uBITX into transmit. When in CW mode, The same line becomes the input for the hand key or external keyer and grounding it causes the uBITX to enter transmit mode and puts out a CW carrier for as long as the line is grounded. Once released, after the normal short delay, the uBITX returns to receive until the line is grounded again by the key. The microphone jack's PTT line is already connected here.

There are two additional ways you can utilize this input. First, you can add a separate, dedicated "Hand Key" jack and connect both the "Tip" connection of this jack AND the "Ring" connection of the microphone jack (PTT) in parallel to the PTT input (Orange wire on P1 (The digital input plug).

The second option is to wire your hand key or external keyer so they utilize the "Ring" contact of the plug for keying rather than the normal connection to the "Tip" contact and then you can plug your key or external keyer directly into the microphone jack.

If you don't want to modify your available key(s) cord(s) to plug into the microphone jack and you also don't want to add the extra panel jack, you could make up a short adapter cable with a plug on one end that fits whatever microphone jack you've chosen to use on your uBITX, and a standard jack on the other end matching whatever plug you use on your hand key or external keyer. The keying connections should go from the "Tip" contact on the jack the key plugs into to the "PTT" (push to talk)

contact on the microphone connector. The shield or common wire of this cable should go from the “Sleeve” connection on the key jack to whichever contact is connected to the uBITX “ground” on the radio end.

ADDITIONAL MENU ITEMS:

In this version of the software, a “Quick change” item has been added.

RIT (receiver incremental tuning) can be turned on by a bit longer press on the encoder button. Keep it pressed until after “Menu” appears, disappears, and RIT appears. Immediately release the button and RIT will be on.

The display will look something like this.

CW Rit 00.0
XX.xV A 07.040.00

This indicates that RIT is now on. The transmitter will stay on 7.040.00 MHz but, using the encoder, you may tune the receiver independently higher (+) in frequency by turning the encoder to the right or lower in frequency (-) by turning it to the left. The tuning for the RIT is plus or minus 5 KHz from the center frequency which is the main VFO A (or B) display shown on the screen.

To turn RIT off, press and hold the button past where Menu appears and you will see RIT again. Immediately release the button and RIT will be turned off.

A/B, SPLIT operation requires that a separate push button switch be installed to use it. As stated earlier, the switch should be wired between ground and the D10 digital I/O line on the Arduino Nano. The RadI2Cino

card already has provisions for this switch but to use this feature on a standard Raduino card, the user is responsible for carefully connecting the switch between the Nano's digital I/O D10 and ground.

A short press of the A/B switch will toggle between the A and B VFO's (just like the big factory rigs. A longer press of the switch will turn SPLIT on. When SPLIT is first turned on, VFO B is set to the same frequency as VFO A and you will see a display similar to the following:

CW t 07.040.00
XX.xV R 07.040.00

VFO A is for receive and VFO B is for transmit. Whichever of the two is selected for tuning by the encoder is indicated by an upper case R or T. The VFO which is NOT being tuned by the encoder is indicated by a lower case r or t. Tapping the A/B button toggles the encoder between the two VFO's.

Your selected frequency and mode will be saved to EEPROM after a short period of no frequency changes so that if you shut the uBITX off, when you turn it on again it will start in the previously saved mode and on the saved frequency.

Jim Sheldon, W0EB w0eb@cox.net
Ron Pfeiffer, W2CTX w2ctx@yahoo.com

APPENDIX A:

W0EB/W2CTX uBITX software – programmer's reference guide.

PC control of the uBITX transceiver was added, starting with the release of Version 2.01R of the W0EB/W2CTX software. Listed here are the current commands we use. They are pertinent to both the I2C and non-I2C display versions of our software.

In order for PC control to work properly, the device controlling the uBITX must have the USB serial port parameters set to one of the following baud rates: 1200, 2400, 4800, 9600, 19200, 38400, 57600 or 115200 baud, N-8-1 (no handshake, 8 data bits and 1 stop bit). The N-8-1 is implied for all baud rates.

This command list is provided so that anyone wishing to write external control software compatible with our versions of software for the uBITX will have an understanding of what we used and how the commands operate. All commands, must terminate with a semi-colon or they will not take effect and will generate errors. All frequency numbers must be exactly 11 digits long (after the command and before the semi-colon. All non used frequency digits must be zeros).

uBITX CAT Commands.

uSA; - Select vfoA

uSB; - Select vfoB

uLA; - Return vfoA

uLAxxxxxxxxxxx; - Load vfoA

uLB; - Return vfoB

uLBxxxxxxxxxxx; - Load vfoB

uCV; - Return current active vfo

uCU; - Current VFO up increment

uCD; - Current VFO down increment

uFIx; - Return/Set Frequency Increment

uDTIxxx; - Display Text I = 1-4,xxx = up to 14 characters

uRVx; - Return Version where x = text string of current version

Dx; - Return Version Date where x = text string of date

uFLx; - Lock uBITX x = 0-unlock, 1-Lock=encoder active + freq incr actve.

uMD; - Return Mode

uMDx; - Set Mode x=-> 0=USB - 1=LSB – etc

uRS; - RIT Start

uROxxxxx; - RIT Offset -5000 to + 5000

uRE; - RIT End

uSC; - Return 64 words at beginning of EEPROM

uSS; - SPLIT Start

uSE; - SPLIT End

uKT; - Keyer Request

uKTxxx; - Set Sidetone 100 – 990

uKSxx; - Keyer Request/Set Speed 5 – 50

uKIA; - Keyer Iambic A

uKIB; - Keyer Iambic B

uKICx; - Return Keyer Iambic selected

uKPN; - Keyer Paddle Normal

uKPR; - Keyer Paddle Reverse

uVS; - Volts Start

uVRxxx; - Return Volts volts X 10

uVE; - Volts End

APPENDIX B:

New uBITX calibration procedure as of this release (will be included in all subsequent release versions).

Turn the uBITX on and as soon as the Splash Screen (Banner) appears, press and hold the Encoder/Function button until “Alignment” appears. Immediately release the button. XTAL (6, 8 or 10pF) will appear along with "push to set". Using the encoder select the load capacitance of the master crystal on your Raduino. The default is usually 8 pF if it’s a factory Raduino card. If you don’t know the master crystal’s load capacitance, assume 8 pF and just press the encoder button to set.

If you do know it and it’s different from 8 pF, rotate the encoder to select 6, 8 or 10 pF and push the function button to set it. If it’s none of the 3 values, select the closest one and use that. It won’t affect the accuracy of the final calibration enough to worry about unless you operate right on the band edges.

Now, “Exit” will appear. Rotate the encoder clockwise to the next selection. The display will now read “Tune station?”. Press the encoder button and tune to a known accurate frequency such as WWV, CHU, GBR on 10 MHz or 5.0 MHz (whichever you can hear better) The frequency display and underscore cursor works just like for normal operation so select the digit you want to change and use the encoder to change it.

Once you have the frequency standard station tuned in, press the encoder (takes a longer hold) until “Exit Tune” is displayed and let go of the encoder button. This gets you back to the alignment menu and “Tune station?” will appear again.

Turn the encoder to the right for the next item which will be “Set Calibration?” Continue past this for right now and continue to the last item which will be “Set the BFO”. Press the encoder/function key to enter this mode. The frequency of 11.995.0 will appear on the top line and you will probably hear high pitched noise in the speaker or headphones. The nominal frequency of the 2nd IF’s crystal filter is 12.000 MHz but due to variations in the crystals used to create this filter, and the fact that it is a lower sideband filter, the correct BFO frequency will be lower than 12 MHz. On mine it came out to 11.997.6. Tune the BFO until you to zero beat the frequency standard station’s carrier. If you continue past the first zero beat and find a second one, ignore the second one and tune back to the first zero beat. If you use the second one you will have your BFO set to the wrong sideband and receive sensitivity may be adversely affected, not to mention the menu items for USB and LSB will be reversed. If this happens, the only cure is to re-do the calibration via the Alignment routine.

Once you are satisfied you are as close to the exact zero beat as you can get, PRESS THE PTT SWITCH ON THE MICROPHONE, not the encoder button, to save the value temporarily.

Rotate the encoder to the left to find “Set Calibration” and press the encoder/function button to select it. Turn the encoder to the right or left until you have the carrier of the station zero beat and again, PRESS PTT, not the encoder button. This saves the calibration offset temporarily.

Now go back and reset the BFO, PRESS PTT and go back to Calibration. Do this several times until the settings quit interacting with each other. Finally, PRESS PTT to save the Cal setting temporarily and turn the encoder to the left until you see “Exit”. Press the encoder button and the

phrase “Power Cycle Radio” will appear. At this point you MUST turn power off and back on to save the temporary calibration figures permanently in EEPROM. You shouldn’t have to do this again for quite a while though the values will change slightly over time due to normal aging of the master oscillator crystal on the Raduino card.

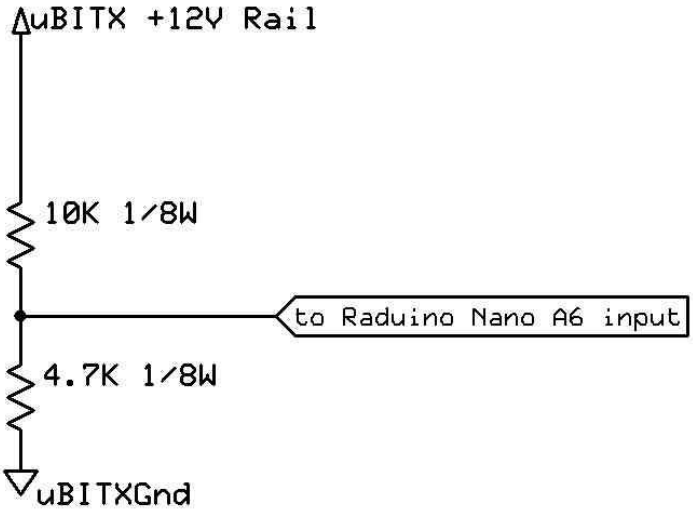
Points to remember - You may have to switch back and forth several times between setting the BFO frequency and setting the Calibration to achieve the most accuracy. In both the Set BFO and Calibration modes, DO NOT USE THE ENCODER/FUNCTION SWITCH to store the results. You MUST use the Push-To-Talk switch or the values will not be saved in the temporary location and you will need to do the entire procedure over again. After you are satisfied that your calibration is right, select “Exit”, press the encoder button and cycle the power (turn it off and back on) to save your calibration values in EEPROM so you don’t have to calibrate the uBITX each time you load new software.

APPENDIX C:

Adding a Voltage input monitor:

This simple modification might be very useful especially if you are operating the uBITX from a battery and you need to keep an eye on it so you don't discharge the battery too far.

Make up a simple voltage divider using a 10K 1/8 watt 1% and a 4.7K 1/8 watt 1% resistor in series with each other. One end of the 10K resistor connects to the uBITX +12 volt rail and the other end goes in series with the 4.7K resistor to ground. The center junction of the two resistors connects to the A6 analog input on the Raduino (or RadI2Cino) Nano's board. This allows the software to monitor the power supply voltage to the uBITX by using the analog to digital converter built into the Arduino Nano. If you use 1 percent resistors, the readout will be quite accurate, usually within 1 or two tenths of a volt. The schematic is shown below.



The 4.7K 1/8W resistor may be 2.2K 1/8W in later versions.

A voltage monitor calibrate function has been added to the alignment routine. To get to this function, Turn the uBITX off and back on again. Immediately after turning it on, press and hold the encoder/function button until “Alignment” appears on the display. Release the button.

XTAL 10pf (might be 6pf or 8pf depending on the crystal on your card)
Push to set
will appear. Push the encoder button and Exit will appear. Rotate the encoder to the right until the last item “Calibrate Volts” appears. Push the encoder button.

Match your VCC
Encoder to exit will appear for a second or so and then

Encoder XXX (voltage calibration constant)
XX.XV (a voltage value)
will appear. At this point, rotate the encoder in either direction as required until the voltage value is equal to the measured DC voltage from the battery or power supply connected to your uBITX and when they match, press the encoder/function button to temporarily save the value. Calibrate Volts will again appear. Rotate the encoder to the left until you get to Exit and press the button.

Power Cycle
Radio

Will appear – turn the uBITX off and back on again – this will save the calibration constant in the EEPROM so you won’t have to do it again.