Prototype of VFO that calibrates frequency using GPS data (using R909-VFO board)

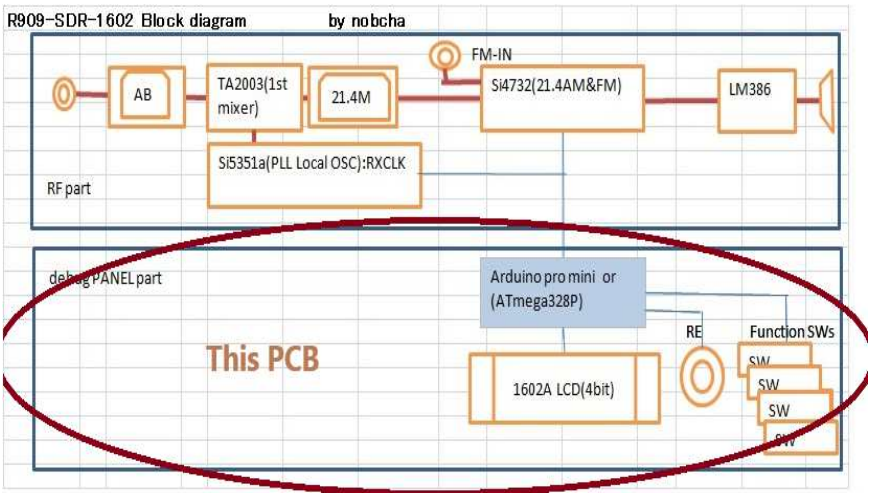
https://nobcha23.hatenablog.com/ ©nobcha

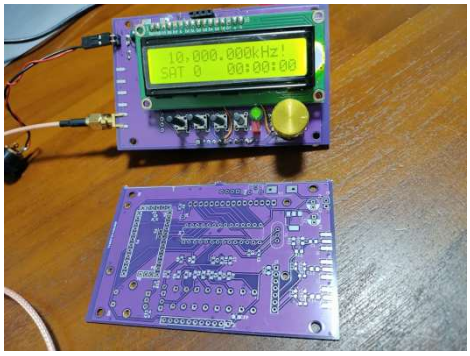
1. Preface

I made a prototype VFO circuit that is calibrated with a high-precision signal received from GPS and improves its accuracy. The frequency error of the GPS signal is said to be -1/10,000,000,000, but with this method, it seems to be about -1/10,000,000 due to the influence of the stability of the crystal oscillator over time. I found and read the relevant QEX article. This method obtains the calibration frequency, so it is also useful for obtaining data for the F-COR (Si5351a reference oscillation frequency correction setting function) of the R909-VFO function for each individual module.

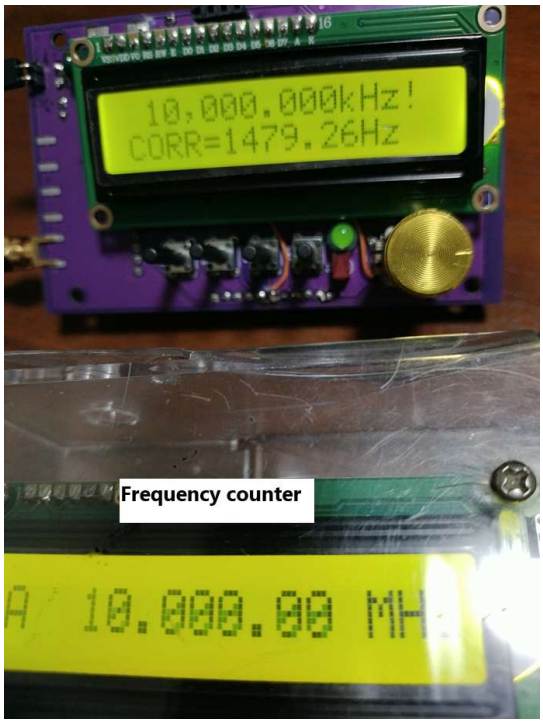
http://www.arrl.org/files/file/QEX_Next_Issue/2015/Jul-Aug_2015/Marcus.pdf

So I modified the R909-VFO I had on hand and diverted this function. The R909-VFO comes from the PANEL control unit (1602LCD display) board, which is the local oscillator part of the R909-DSP radio.





The GPS Corrected VFO works by using the GPS 1 second pulse PPS signal as a gate signal. The Si5351a outputs a frequency of 2.5MHz, counts for 40 seconds, and calculates the difference with 10,000,000. This difference becomes the Si5351a VFO calibration data. The GPS module captures radio waves from the satellite, and when the PPS signal is output, the calibration frequency is obtained 44 seconds later.



The 25MHz crystal oscillator of the Si5351a module is 1479.26Hz off from the GPS standard frequency when converted to 10MHz.

This is the calibrated 10MHz signal output. Read with a frequency counter calibrated with a 12.8MHz TCXO.

For the prototype and experimental GPS unit, we adopted the E108 GN02-D, which has a PPS signal terminal and is inexpensive.

The R909-VFO-Panel 1602 board used for prototyping and experiments has an ATmega328P or pro mini mounted as an Arduino, and the 1602LCD and GPS module are connected via the connection pin header. The J4 connector, originally prepared to connect this board to the RF board, is used to connect to the GPS module.

R909-VFO-GPS-1602			2024.12.03
Arduino signal	R909-VFO-LCD-1602	PANEL/MAIN J4	GPS EN108 GN02D
D0/RX	GPS-TX/USB-SERIAL	#7	#4 GPS/TXD via 100R
D1/TX	USB-SERIAL		
D2	GPS 1pps input	#6	#2 GPS/PPS via 100R
D3	Rotary encoder pin A		
D4	Rotary encoder pin B		
D5	2.5 MHz input from Si5351 CLK1 pin		
D6		#8	
D7	LCD/Back light & GPS/EN	#9	#1 GPS/EN
D8	LCD RS		
D9	LCD enable		
D10	LCD DB4		
D11	LCD DB5		
D12	LCD DB6		
D13	LCD DB7		
A0/D14	Rotary encoder SW		
A1/D15	Monitor LED		
A2/D16	Push switches		
A3/D17		#5	
A4/D18	Si5351 SDA	#3	
A5/D19	Si5351 SCL	#4	
VCC	VCC	#1,#11	#6 VCC
GND	GND	#2,#10	#5 GND

2. PCB

The circuit can be built using a universal board or breadboard. However, there are switches to operate, and I created a printed circuit board to ensure stable operation even if the GPS or Si5351a is removed and installed. The complete set of Gerber files for this board has been uploaded to GITHUB. It was output in standard specifications using the PCBGOGO KiCAD plugin. The bill of materials and circuit diagram are also in the same directory. Complete set of Gerber files for the board

https://github.com/Nobcha/R909-VFO-GPS/blob/main/R909-VFO-LCD-GPS_rev.kicad_pcb.zip

3. Assembling

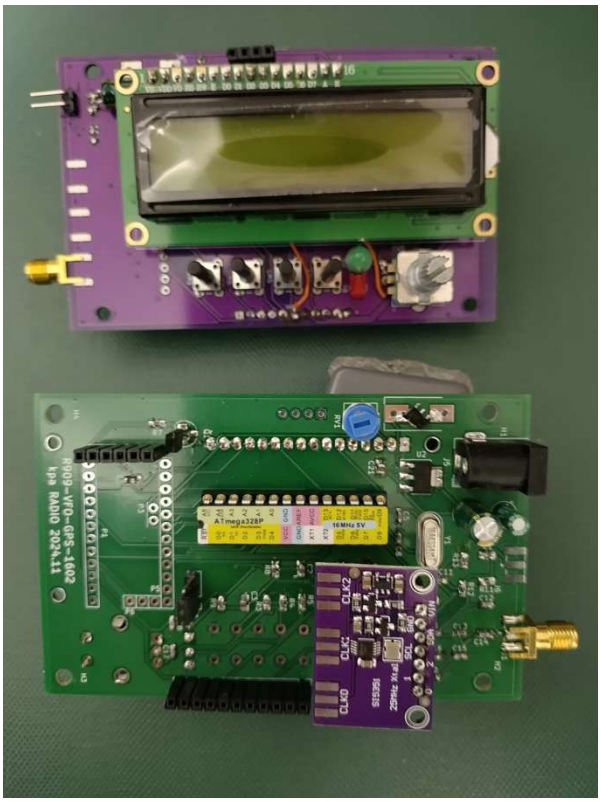
Refer to the parts list and circuit diagram and first solder the surface-mounted L, C, R and IC. Next, solder the DIP components: crystal, electrolytic capacitor, semi-fixed resistor, connector, IC socket, and LED. Finally, the SMA and switches.

Parts list: https://github.com/Nobcha/R909-VFO-GPS/blob/main/R909-VFO_GPS_bom.jpg
Circuit diagram: https://github.com/Nobcha/R909-VFO-GPS/blob/main/R909-VFO-GPS1602_scm.pdf

The switches, LCD/OLED sockets, and LEDs are mounted on the back. Use either the Arduino pro mini or ATmega328P. Depending on which one you choose, select and mount the pro mini connector, 28PIC socket, crystal, and 20pF.

R909-VFO-LCD-GPS PCB BOM

#	Reference	Qty	Value	Footprint	Pro mini
1	C1, C11	2	4700p	Capacitor_SMD:C_0805_2012Metric_Pad1.18x1.45mm_HandSolder	
2	C2, C3, C7, C9, C10, C19, C20, C21	8	0.1u	Capacitor_SMD:C_0805_2012Metric_Pad1.18x1.45mm_HandSolder	
3	C4	1	22u	Capacitor_THT:CP_Radial_D5.0mm_P2.50mm	
4	C5, C8	2	20p	Capacitor_SMD:C_0603_1608Metric_Pad1.08x0.95mm_HandSolder	0
5	C6	1	100u	Capacitor_THT:CP_Radial_D5.0mm_P2.50mm	
6	C12, C16	2	5p	Capacitor_SMD:C_0603_1608Metric_Pad1.08x0.95mm_HandSolder	
7	C13, C15	2	3p	Capacitor_SMD:C_0603_1608Metric_Pad1.08x0.95mm_HandSolder	
8	C14	1	100p	Capacitor_SMD:C_0805_2012Metric_Pad1.18x1.45mm_HandSolder	
9	C17, C18	2	0.22u	Capacitor_SMD:C_0603_1608Metric_Pad1.08x0.95mm_HandSolder	
10	D1, D2	2	LED	LED_THT:LED_D3.0mm_Clear	
11	D3	1	1N4001	Diode_SMD:D_SMC-RM10_Universal_Handsoldering	
12	DS1	1	WC1602A	Display:WC1602A Connector_PinHeader_2.54mm:PinHeader_1x16_P2.54mm_Vertical	
13	H1, H2, H3, H4	4	MH	MountingHole:MountingHole_3.2mm_M3	
14	J1	1	LO0	Connector_Coaxial:SMA_Samtec_SMA-J-P-X-ST-EM1_EdgeMount	
15	J2	1	0(RX) select	Connector_PinHeader_2.54mm:PinHeader_1x03_P2.54mm_Vertical	
16	J3	1	Si5351a	Connector_PinHeader_2.54mm:PinHeader_1x07_P2.54mm_Vertical	
17	J4	1	PANEL/MAIN	Connector_PinHeader_2.54mm:PinHeader_1x11_P2.54mm_Vertical	
18	J5	1	DC12 IN	Connector_BarrelJack:BarrelJack_Wuerth_6941xx301002	
19	J6	1	LO1	Connector_Coaxial:SMA_Samtec_SMA-J-P-X-ST-EM1_EdgeMount	
20	J7	1	i2c	Connector_PinHeader_2.54mm:PinHeader_1x04_P2.54mm_Vertical	
21	L1	1	1u	Inductor_SMD:L_0603_1608Metric_Pad1.05x0.95mm_HandSolder	
22	L2	1	1u	Inductor_SMD:L_0603_1608Metric_Pad1.05x0.95mm_HandSolder	
23	P1	0	Digital	Socket_Arduino_Pro_Mini:Socket_Strip_Arduino_1x12	1
24	P2	1	COM	Socket_Arduino_Pro_Mini:Socket_Strip_Arduino_1x06	1
25	P3, P5	0	ADC	Socket_Arduino_Pro_Mini:Socket_Strip_Arduino_1x02	2
26	P4	0	Analog	Socket_Arduino_Pro_Mini:Socket_Strip_Arduino_1x12	1
27	P6	0	GND	Connector_Pin:Pin_D1.1mm_L8.5mm_W2.5mm_FlatFork	1
28	Q1	1	2SC1815	Package_TO_SOT_THT:TO-92_HandSolder	
29	R1	1	2k	Resistor_SMD:R_0805_2012Metric_Pad1.20x1.40mm_HandSolder	
30	R2	1	330R	Resistor_SMD:R_0805_2012Metric_Pad1.20x1.40mm_HandSolder	
31	R3	1	680R	Resistor_SMD:R_0805_2012Metric_Pad1.20x1.40mm_HandSolder	
32	R4	1	1k	Resistor_SMD:R_0805_2012Metric_Pad1.20x1.40mm_HandSolder	
33	R5, R7, R8, R9, R10	5	10k	Resistor_SMD:R_0805_2012Metric_Pad1.20x1.40mm_HandSolder	
34	R6	1	470R	Resistor_SMD:R_0805_2012Metric_Pad1.20x1.40mm_HandSolder	
35	R11, R13	2	160R	Resistor_SMD:R_0805_2012Metric_Pad1.20x1.40mm_HandSolder	
36	R12	1	33R	Resistor_SMD:R_0805_2012Metric_Pad1.20x1.40mm_HandSolder	
37	RV1	1	10k	Potentiometer_THT:Potentiometer_ACP_CA6-H2,5_Horizontal	
38	SW1	1	FREQ	Button_Switch_THT:SW_PUSH_6mm	
39	SW2	1	VOL	Button_Switch_THT:SW_PUSH_6mm	
40	SW3	1	SQL	Button_Switch_THT:SW_PUSH_6mm	
41	SW4	1	SCAN	Button_Switch_THT:SW_PUSH_6mm	
42	SW5	1	RE_Sw	Rotary_Encoder:RotaryEncoder_Alps_EC11E-Switch_Vertical_H20mm_CircularMountingHoles	
43	U1	1	ATmega328P-PU	Package_DIP:DIP-28_W7.62mm IC socket	0
44	U2	1	AMS1117CD-5.0	Package_TO_SOT_SMD:SOT-223-3_TabPin2	
45	Y1	1	16MHz	Crystal:Crystal_HC49-U-3Pin_Vertical	0



4. Sketch

I searched for a sketch to use as a reference and found a sketch by SQ1GU that was modified from a QEX article. http://sq1gu.tobis.com.pl/pl/syntezery-dds
Initially, I ported this sketch and carried out confirmation experiments. The following sketch is modified for the R909-VFO. https://github.com/Nobcha/R909-VFO-GPS/Si5351_GPS_kpa.ino

With this sketch, the rotary encoder operation did not go smoothly. After further searching, I found an improved version on GITHUB that made the rotary encoder contact input an interrupt. I downloaded this sketch, improved it to match the port settings to the R909-VFO board, and decided to modify and add functions.
gps-calibration-5351 https://github.com/csqwdy/gps-calibration-5351/blob/main/README.md
I have uploaded the sketch ported and improved for the R909-VFO to GITHUB.
GITHUB https://github.com/Nobcha/R909-VFO-GPS/blob/main/Si5351_GPS_kpa.ino

The points that were modified and improved from the original prototype sketch are:

① PPS was assigned to D2 and specified as an interrupt, and rotary encoder contact input was assigned to D3 and D4 and made into signal change interrupts.

② Correction frequency display function was added. In the original, the 25MHz oscillation of the Si5351a module was improved to TCXO. In this prototype, it remains a crystal oscillator and has poor stability over time. Therefore, since it is difficult to store the correction frequency in EEPROM at the threshold of 0.2Hz, a function was added to store the correction frequency

in EEPROM when the rotary encoder is turned clockwise in Correction frequency display mode.

5. Usage

5.1 When the power is turned on, a banner with the name and version is displayed. Then it waits for a signal from the GPS unit. The GPS unit then waits for satellite capture. When a satellite is captured, the time is displayed. The number of satellites captured is displayed. When 4 or more satellites are captured stably, a PPS signal is output. When the PPS-LED on the GPS module starts flashing at 1-second intervals, it is in a stable reception state. When satellite capture is unstable, the PPS signal is sometimes lost. When it is lost, a ? is displayed after the frequency display. When data is updated smoothly from the GPS, a ! is displayed. It seems that stable reception cannot be achieved unless the GPS unit or antenna is taken outdoors.

5.2 When it is stable for 44 consecutive seconds and a PPS is output, the calibration frequency is calculated and used as the correction value. A considerable level of reception stability is required to not lose a PPS signal for 44 seconds. This correction value is a calibration value specific to the Si5351a module (10 MHz conversion), so it can be used as reference data when incorporating individual modules.

5.3 The calibration frequency is calculated, and the fluctuation range of the calibration frequency is calculated as the F STAB value. When the F STAB value becomes ±0.2Hz or less, the calibration frequency correction is stored in the EEPROM. It will be reflected the next time the power is turned on. When the rotary encoder is turned clockwise in the correction display mode, the calibration frequency correction is stored in the EEPROM.

5.4 JST (Japan Standard Time) is 9 hours earlier than Greenwich Mean Time, so set the TIME zone to +9.

5.5 There are 5 banks. The initial values are 1: 10MHz, 2: 16.7MHz, 3: 37.2MHz, 4: 64.8MHz, and 5: 145MHz. The setting value is stored in the EEPROM for each bank.

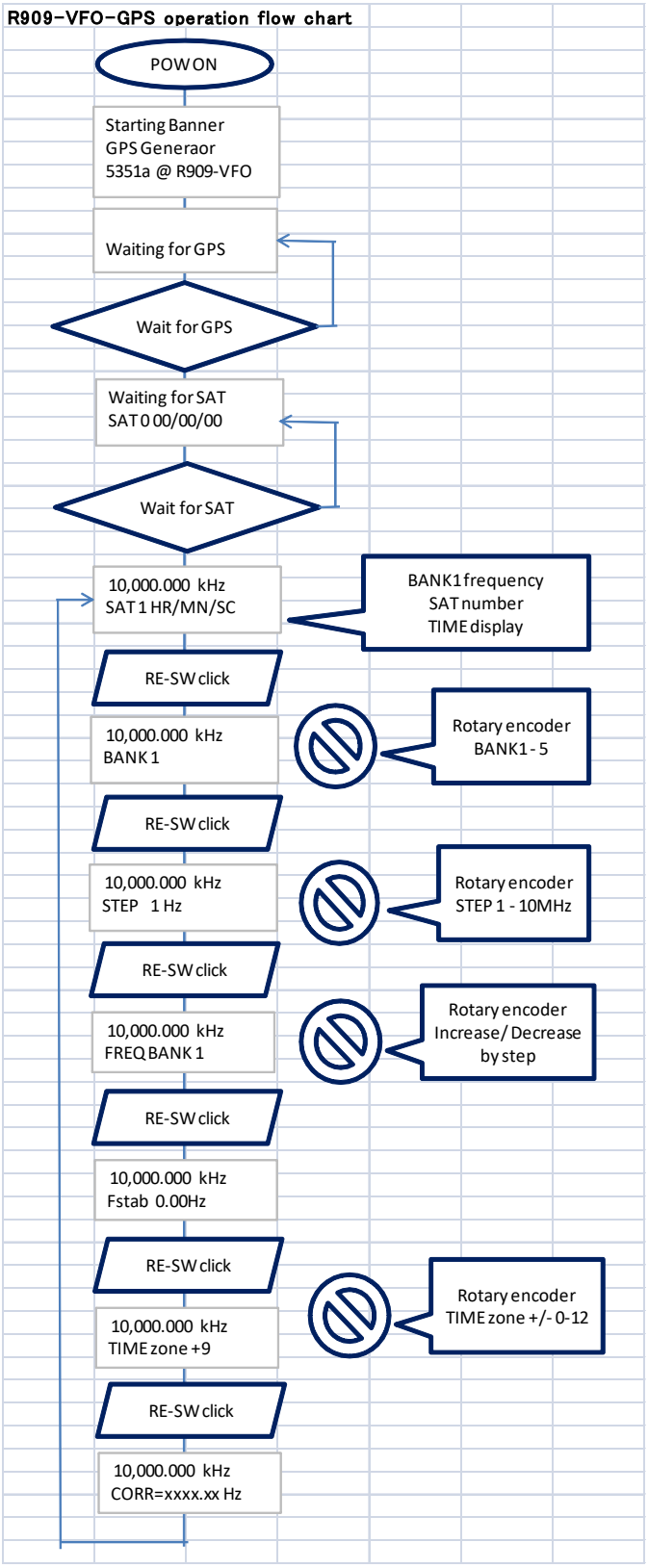
5.6 Eight types of step frequencies can be selected: 1, 10, 100Hz, 1, 10, 100KHz, and 1, 10MHz.

5.7 When in Bank mode, turning the rotary encoder increases or decreases the frequency of the specified bank in STEP frequency units. The set frequency value is stored in the EEPROM for each bank and will be recalled the next time the power is turned on.

5.8 Press and hold RE-SW while turning on the power to initialize the EEPROM and start up.

YOUTUBE https://www.youtube.com/shorts/6HuBzLwyKoM

The operation flow is summarized in a flowchart below.



6. References and Acknowledgements

In carrying out this prototype and experiment, we obtained and used as reference information from various websites on the Internet. We would like to thank the Arduino library authors, the information providers on GITHUB, and others. (The sources are listed in the text.)

The prototype board was also created with the cooperation of PCBGOGO.