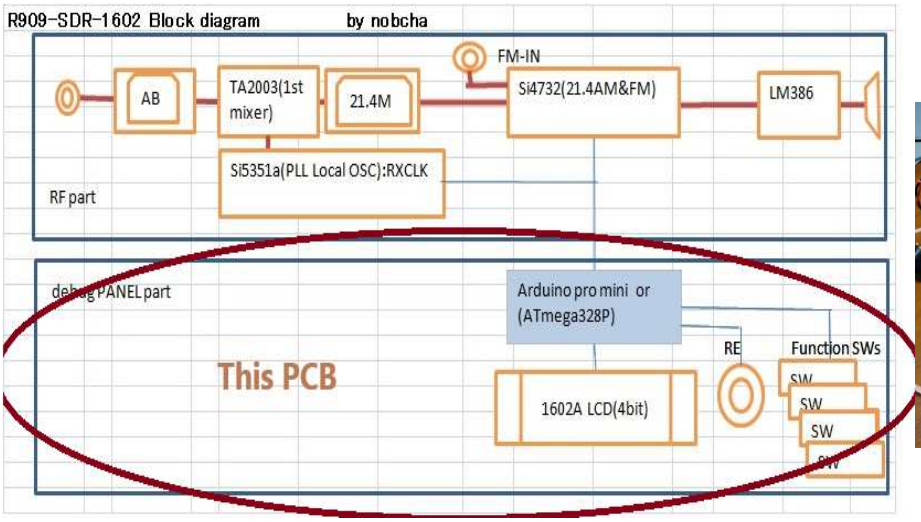
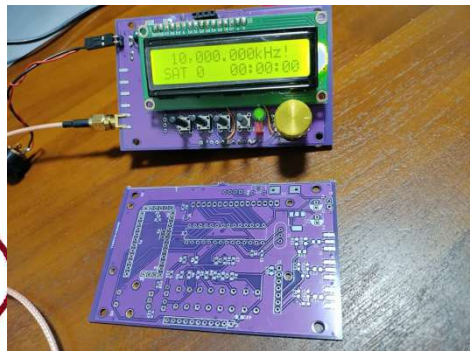
GPS データで周波数較正する VFO の試作(R909-VFO 基板を利用)　nobcha23 の日記

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

WEB を見ていたら GPS Corrected VFO という試作が紹介されていました。GPS が受信した高精度の周波数を使用し Si5351aVFO の基準発振を較正、高精度信号を出す VFO 回路です。引用された該当 QSP 記事を探し出し読んでみました。このやり方で較正周波数を取得できるなら、F-COR（Si5351a の基準発振周波数の補正設定機能）用データ取得にも役立ちそうです。と言うことでこの機能を手元にある R909-VFO に移植してみました。

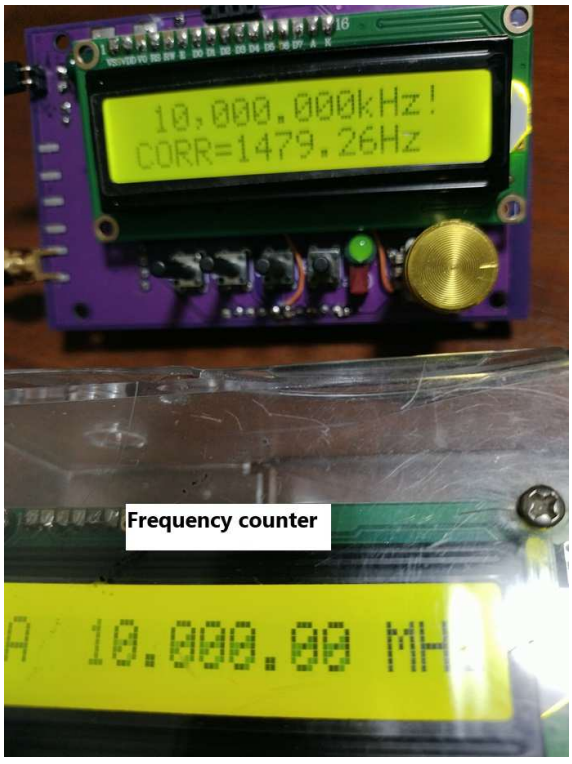
R909-DSP のデバッグ用に開発した PANEL 制御部(1602LCD 表示)基板を使います。





GPS Corrected VFO の仕組みですが、GPS の 1 秒パルス PPS 信号をゲート信号として使い、Si5351a の周波数出力を 2.5MHz を 40 秒間カウント、100,000,000 との差を計算します。この差が Si5351aVFO 較正データになります。　http://www.arrl.org/files/file/QEX_Next_Issue/2015/Jul-Aug_2015/Marcus.pdf

衛星を捕捉しＰＰＳデータが出力されると 44 秒後に較正周波数が得られます。



Si5351a モジュールの 25MHz 水晶発振器が 10Mhz 換算値で GPS 標準周波数より 1479.26Hz ずれています。

更正された 10Mhz の信号出力です。
12.8MHzTCXO で較正した周波数カウンタでの読み取りです。

では試作・実験に取り掛かります。まずは GPS ユニットを買います。PPS 信号端子があり安価だった E108 GN02-D にしました。元ネタでは GPS ユニットと Arduino はハードウェアシリアル接続です。しかし、デバッグでシリアルポートは確保したいと思ったので、手始めの実験ではＧＰＳとの接続は SoftwareSerial を使用することにしました。

試作・実験に使う R909-VFO-Panel 1602 基板には Arduino として ATmega328P あるいは pro mini が載っていて、1602LCD、GPS モジュールを接続ピンヘッダー経由でつなぎます。そして、元々はこの基板と RF 基板をつなぐための J4 コネクタ、IDE 接続用シリアル USB の RXD 端子を GPS モジュールと繋ぐために使います。

1. 基板作成

ユニバーサル基板やブレッドボードで回路を組める程度の構成です。しかし、スイッチ操作や GPS や Si5351a の取り外しなどしても安定動作させたく、プリント基板を作りました。基板のガーバーファイル一式は GITHUB にアップロードしてあります。PCBGOGO の KiCAD プラグインで出力したものです。部品表、回路図も同じディレクトリーに置いてあります。

基板のガーバーファイル一式
https://github.com/Nobcha/R909-VFO-GPS/blob/main/R909-VFO-GPS1602_scm.pdf
部品表　https://github.com/Nobcha/R909-VFO-GPS/blob/main/R909-VFO_GPS_bom.jpg
回路図 https://github.com/Nobcha/R909-VFO-GPS/blob/main/R909-VFO-GPS1602_scm.pdf

2. SoftwareSerial で事前接続実験

GPS 受信動作やデータがどうなっているのかをまずは SoftwareSerial でつないで事前接続実験をしてみました。使用するライブラリの準備も兼ねて行いました。

2..1　ライブラリ TinyGPS++のダウンロード

GitHub mikalhart から TinyGPSPlus ライブラリをダウンロード。
https://github.com/mikalhart/TinyGPSPlus/releases
ダウンロードした、”TinyGPSPlus-1.0.2b.zip”を Arduino IDE へインクルード

2.2.　WEB のページを参考に GPS との接続確認スケッチを作成。

このユニットには EN 端子が付いていて、LOW にすると SLEEP になるので、HIGH にするため、LCD バックライトのポート信号を流用しました。

2.3.1　GPS 時計を移植する

スケッチ　　https://github.com/Nobcha/R909-VFO-GPS/blob/main/GPS_watch_kpa.ino
元の参照した WEB　　よかひより　ざぼんさん　　https://yokahiyori.com/arduino_gps_lcd_clock/

2.3.2.　緯度、経度

続いて、GPS からの NMEA0183 コマンドを解析し緯度、経度などのデータを取得し、LCD に表示してみます。

https://github.com/Nobcha/R909-VFO-GPS/blob/main/GPS_TEST20241022_3.ino

注意：スケッチのデバッグには衛星捕捉が必要です。窓際か庭先に置き *しばらく*待ちます。数分ぐらいするとppsのLEDが光り始める共に経度緯度など取得でき表示できました。

GPS_TEST20241022_3.ino の実行結果

3.　ハードウェアシリアル接続に切り替え ORIGINAL スケッチは SQ1GU 版を参照。

以上の事前実験を経て、GPSの使い方が判ってきました。次に参考スケッチを探します。QEX記事を引用改良したSQ1GUさんのスケッチがありました。　http://sq1gu.tobis.com.pl/pl/syntezery-dds

はじめはこのスケッチを移植し確認実験を進めました。

https://github.com/Nobcha/R909-VFO-GPS/Si5351_GPS_kpa.ino

更に探すと、ロータリエンコーダ接点入力を割り込みにした改良版がGITHUBにありました。このスケッチをダウンロードしポート設定をR909-VFO基板に合わせ、改良修正追加することにしました。

gps-calibration-5351　　https://github.com/csqwdy/gps-calibration-5351/blob/main/README.md

R909 - VFO用に移植、改良したスケッチをGITHUBにアップロードしました。

GITHUB　https://github.com/Nobcha/R909-VFO-GPS/blob/main/Si5351_GPS_kpa.ino

オリジナルの試作、スケッチから修正・改良を行なった点は

① D2にPPSを割り当て割り込み指定し、D3,D4にはロータリエンコーダ接点入力を割り当て信号変化割り込みにした、

② Correction 周波数表示機能を追加、
元ネタでは Si5351a モジュールの 25MHz 発振を TCXO に改良している。この試作では水晶発振器のままであり経時安定度が悪い。なので、EEPROM へ correction 周波数を記憶する閾値 0.2Hz ではなかなか記憶しないので、Correction 周波数表示モード時にロータリエンコーダを時計方向に回すと較正周波数 correction を EEPROM に記憶する機能を追加しました。

4. 使い方

4.1　電源を投入し、GPS ユニットは屋外に出します。衛星を 4 以上安定して捕捉すると PPS 信号がでます。GPS モジュールの PPS－LED が 1 秒間隔で点滅し始めると安定受信状態です。衛星の捕捉が安定しないときは、時々PPS 信号が欠落します。欠落時には周波数表示の後に？を表示します。GPS からデータ更新が円滑に行われているときは！を表示します。

4.2　44 秒間安定して PPS が出ると、較正周波数を算出し correction 値とします。44 秒間 PPS 信号が欠落しないというのは結構な受信安定度が必要です。この correction 値は Si5351a モジュールの固有の較正値(10Mhz 換算)ですので、個別モジュール組み込み時の参考データになります。

4.3　較正周波数が計算され、また、較正周波数の変動幅が F STAB 値として計算されます。F STAB 値が ±0.2Hz 以下になると較正周波数 correction を EEPROM に記憶します。次に電源を入れた時に反映されます。Correction 表示モードの時にロータリエンコーダを時計方向に回すと較正周波数 correction を EEPROM に記憶します。

4.4　JST（日本標準時）はグリニッジ標準時に対して 9 時間早いので TIME zone を+9 に設定します。

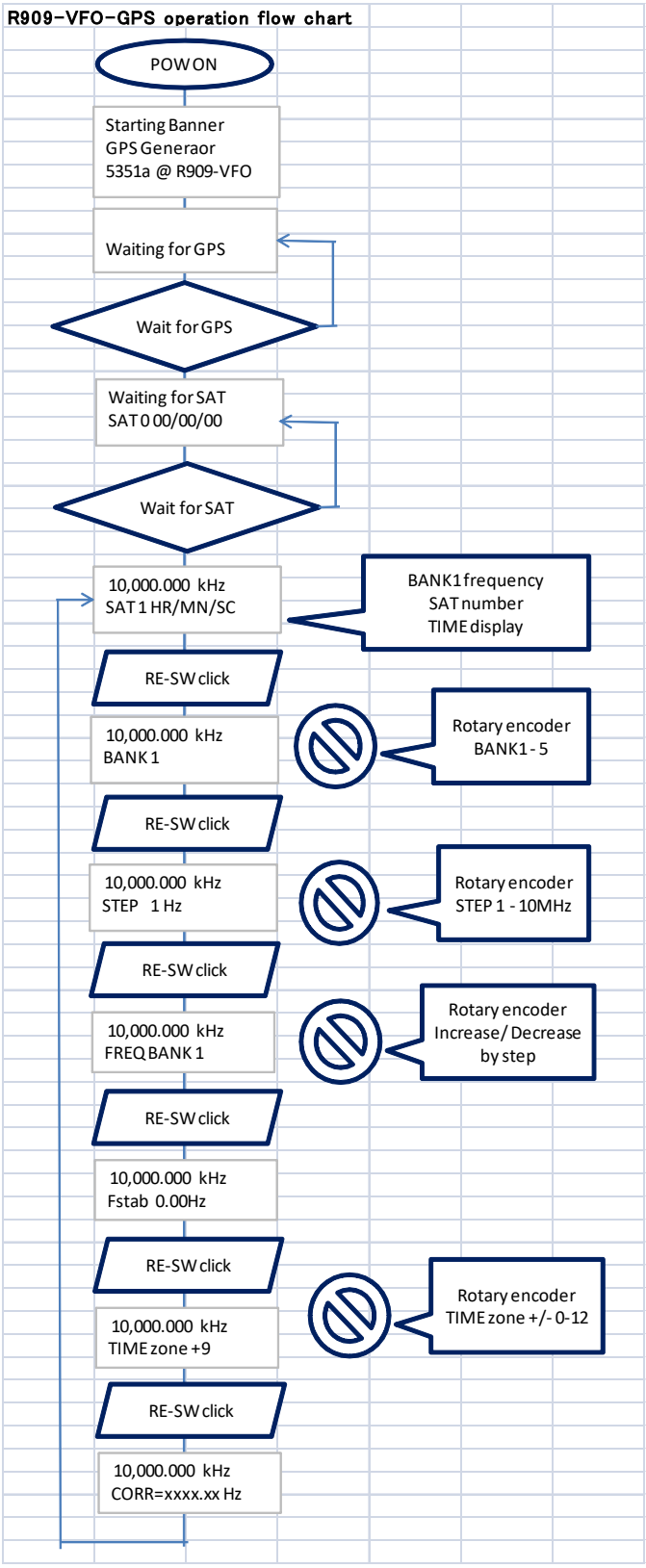
4.5　BANK として 1－5 まであります。1：10MH z 、2：16.7MH z 、3：37.2MH z 、4：64.8MH z 、5：145MH z となっています。設定値は各 BANK 毎に EEPROM に記憶されます。

4.6　STEP 周波数は 1，10，100Hz、1，10，100KHz、1，10MH z の 8 種類が選べます。

4.7　FREQ　Bank モードの時にロータリエンコーダを回すと、STEP 周波数単位で指定 BANK の周波数を増減できます。設定した周波数値は EEPROM に記憶され、次回電源オン時には呼び出されます。

4.8　RE-SW を押しながら電源を投入すると EEPROM を初期化して立ち上がります。

操作の流れをフローチャートにまとめました。



5. 参考情報と謝辞

本試作・実験を行うにあたり、インターネットの各種ホームページから情報を得て参考にさせていただきました。Arduino のライブラリー作者、GITHUB での情報提供者などの皆様に感謝申し上げます。

更新情報は nobcha23 の日記”https://nobcha23.hatenablog.com/”で発信しております。

本試作は nobcha の個人的な興味の範疇で行っております。商用利用を意図したものではありません。なお、記事中にはアフィリエートやプロモーションが含まれていることをご承知願います。また、著作権を維持するため、引用などされる場合は一報いただけると幸いです。

nobcha４８＠gmail.com　（半角にしてお使いください）