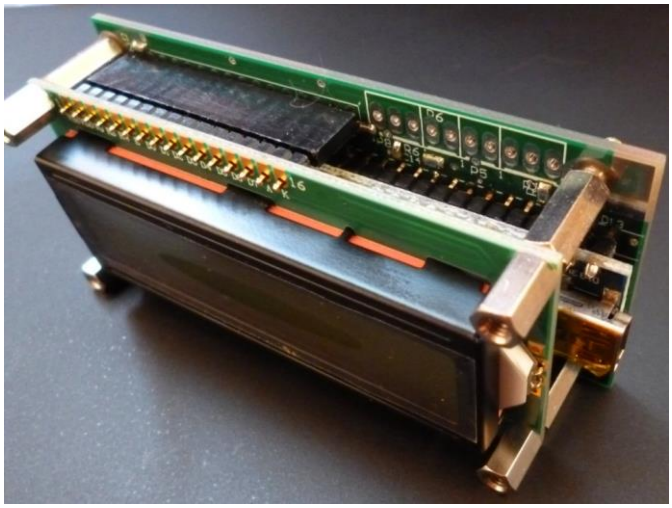
Le VFOduino à Si5351



Le générateur d'horloges programmables Si5351, maintenant connu du monde radioamateur, offre la possibilité de réaliser un VFO ultra miniaturisé. Intégré à un module de la taille d'un afficheur LCD, et piloté par un Arduino Nano, il devient un sous-ensemble complet pour construire un petit transceiver comme le Rocket. Quelques indications en fin d'article, vous expliqueront comment le programmer en toute facilité.

Décidant à me mettre à l'Arduino après des années à programmer en assembleur, j'ai recherché des applications dans le domaine radioamateur. Pour travailler sur un objectif concret, je m'étais intéressé aux transceivers Bitx40 puis µBitx de Ashhar Farhan VU2ESE en Inde [1]. Ce qui m'a séduit avant tout était le Raduino, ce VFO ultra compact et consommant beaucoup moins de courant qu'une DDS d'Analog Device. J'ai acheté le Raduino pour l'évaluer puis l'intégrer à mon projet de transceiver « OM-made ». Puis, en le comparant un jour au module Si5351 d'Adafruit, je m'étais dit qu'il était possible de l'améliorer en incluant quelques options. Par cette version de ce VFO, j'ai recherché à l'améliorer au niveau de sa pureté spectrale : régulateur séparé, découplages d'alimentation et plan de masse sur les deux faces. Pour rentabiliser la fabrication du circuit imprimé je me suis servi de tout l'espace disponible pour y prévoir des implantations multiples (TCXO, régulateur) et des découplages HF sur les ports du Nano. L'article suivant n'entrera pas dans les détails de fonctionnement du Si5351 ou de la programmation de tous ces registres. En revanche, il aidera ceux qui souhaitent construire leur propre VFO à partir de librairies existantes.

Conçu pour être un générateur d'horloges à trois sorties le **Si5351A-B-GT** concurrence bien largement les DDS telles que l'AD9850 (120mA consommé à 125 MHz) ou l'AD9833 limité à 12 MHz. Le Si5351 moins gourmand en courant (25 mA maximum) ne prend qu'un espace de 5x5 mm sur la carte. Son bruit de phase dépend du mode de division et s'améliore en diminuant la fréquence programmée. Il offre un niveau de sortie élevé, plus que suffisant pour commander un mélangeur à diodes 7 dBm. Pour comprendre son fonctionnement interne, on se reportera à sa datasheet et au guide de programmation du fabricant Silicon Labs [2]

Voici les points forts du Si5351 :

- Trois sorties HF paramétrables entre 2,5 kHz et 200 MHz offrent des possibilités pour des applications à conversions multiples.
- Activation et fréquence indépendantes sur chaque sortie.
- Les sorties peuvent être paramétrées individuellement en niveau sur 4 paliers.
- Résolution de 1 Hz avec la librairie de KE7ER.
- Référence commune partant d'un quartz de 25 MHz.
- Calage du quartz par capacités commutables internes par paramétrage.

- Calibration fine en fréquence par une procédure logicielle et un facteur de correction.
- Bruit de phase caractérisé [7]
  Sortie à 100 MHz : < -80 dBc/Hz à 100 Hz, < -110 dBc/Hz à 1 kHz.
  Sortie à 10 MHz : < -103 dBc/Hz à 100 Hz, < -127 dBc/Hz à 1 kHz.

Le concept d'un tel VFO est vraiment séduisant : pour environ 30 € on a un VFO à trois sorties (**figure 1**). Avec l'afficheur LCD et l'Arduino Nano ce module peut gérer bien d'autres fonctions ! Ce VFO se destine principalement à un transceiver CW/SSB, mais on ne pourra que le moduler en fréquence et très lentement par programmation (usage en WSPR).

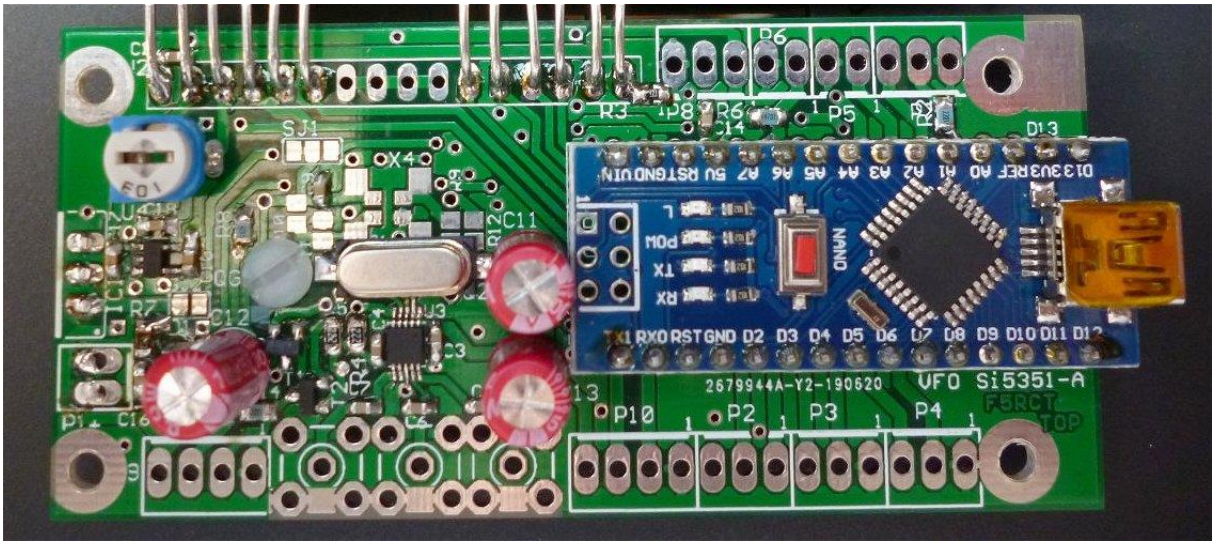


Figure 1 : Le VFOduino à Si5351 : c'est le petit carré noir sous le quartz !

Le Si5351 ne s'alimente qu'en 3,3 V tandis que l'Arduino est en 5 V. VU2ESE ne s'est pas privé d'utiliser le 3,3 V sortant de la carte Arduino Nano. En fait cette tension provient du régulateur interne de l'interface USB/UART FTR232RL ou de son clone (CH340). Cette sortie n'est absolument pas spécifiée en courant disponible, ni en bruit résiduel. De plus, la conception un peu légère du Raduino ne donne pas une bonne pureté spectrale (raies inférieures à -40 dBc entre 2 et 77 MHz). En se donnant les moyens avec un meilleur routage avec un régulateur 3,3 V séparé, le VFOduino montre un spectre bien plus propre (**figure 2**).

Reconnaissons qu'il est difficile d'assurer une bonne séparation HF entre les sorties du Si5351 de par les broches contiguës du boitier TSSOP-10 et la seule broche de masse. Le routage du circuit imprimé oblige à avoir des liaisons de masse très courtes pour limiter le mode commun et l'intermodulation entre les sorties.
Il est possible de diminuer le niveau des raies parasites en diminuant le courant des sorties entre 2 et 6 mA. Quatre niveaux sont paramétrables par configuration. Niveau 0 pour 2 mA, 1 pour 4 mA (+8 dBm), 2 pour 6 mA (+10 dBm), et 3 pour 8 mA (+13 dBm). Pour 4 mA, on a +8 dBm ce qui suffit largement pour un mélangeur à diode +7dBm raccordé via un atténuateur de 1,5 dB.

L'impédance des sorties reste proche de 50 Ω ; il est inutile de vouloir la maîtriser parfaitement car elle dépend du courant de sortie. Les signaux de sortie ne sont pas sinusoïdaux mais proches d'un carré puisque c'est un générateur d'horloges. Pour favoriser l'IP3 en réception, les signaux carrés conviennent bien mieux aux mélangeurs à diodes que du sinus ! Ainsi avec une puissance d'attaque réduite, les diodes commutent plus rapidement et le temps de commutation de la

transition non-linéaire s'en trouve réduit : on est dans des conditions idéales pour un mélangeur à diodes.

Le concept de ce VFO apporte plus de flexibilité que le Raduino et une amélioration de la qualité du signal (**figure 2**) en bénéficiant d'un régulateur séparé pour le 3,3 V. Ce régulateur externe 3,3 V apporte une bien meilleure séparation entre l'Arduino et le Si5351, il contribue aussi à réduire le bruit de phase. Un régulateur en boitier STO23-5 convient très bien pour le courant de 22 mA demandé par le Si5351 ; d'autant plus qu'il existe une multitude de références sur le marché. Le RT9193-33GB est un régulateur faible bruit (100 µV de 10Hz à 100 kHz). A toutes fins utiles de simplification, un strap SJ2 sur le circuit imprimé permet de revenir au 3,3 V (40 mA maxi) du Nano si l'on n'a pas ce régulateur externe.

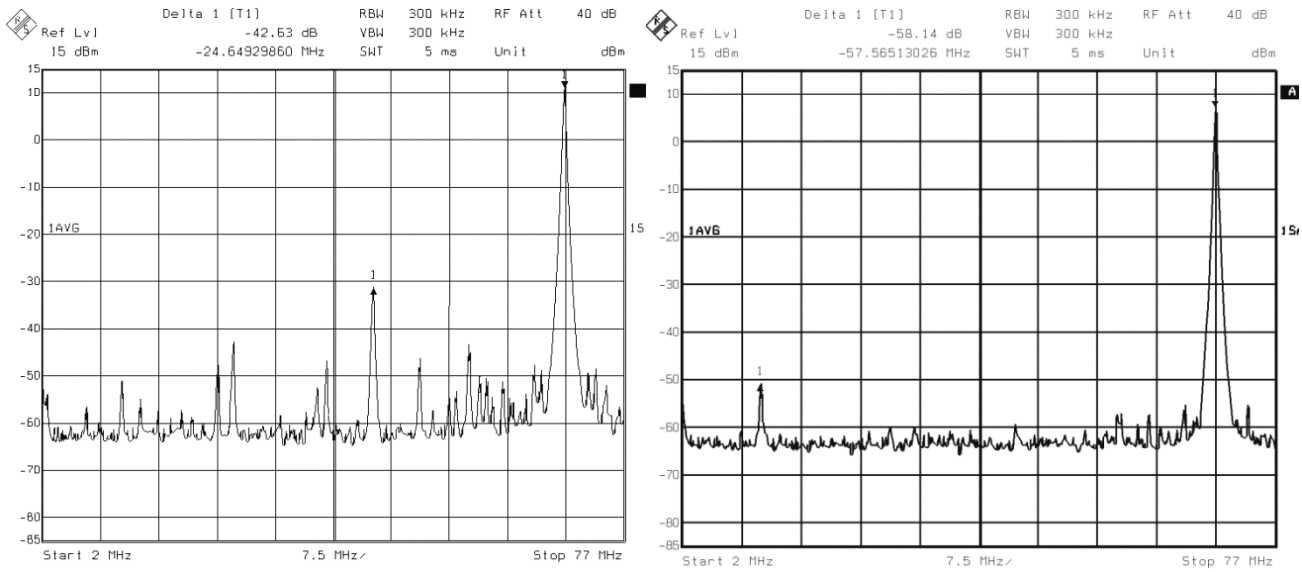


Figure 2 : Spectres de comparaison entre le Raduino de VU2ESE et le VFOduino de F5RCT.

Différents types de références de fréquences sont applicables au Si5351. L'implantation d'un quartz sur le circuit imprimé correspond à un boitier HC49U traversant ou monté en surface (les deux types de boitiers figurent sur le schéma comme deux quartz montés en parallèle, mais un seul sera monté !).
De base, la référence de fréquence est un quartz 25 ou 27 MHz. Certains se sont lancés à essayer ce circuit sur d'autres fréquences de référence : comme à 10 MHz avec un oscillateur externe sur la broche X1 [4]. Cela fonctionne à condition d'adapter le rapport de multiplication de la PLL et de le répercuter sur le diviseur, mais là, ce n'est pas l'objet de cet article.
On pourra aussi y adapter un TCXO CMS de taille 3,2 par 5,5 mm, X2 sur le schéma. Suivant les spécifications du TCXO la broche 1 devra être polarisée à une tension fixe par le jeu de résistances R10 et R11. Dans certains cas un condensateur de découplage pourra se souder au-dessus de R11. La précision d'un quartz ordinaire suffit amplement pour le Rocket qui reste dans une tolérance de +/- 10 Hz à 30 MHz ! Le calibrage s'opère par un paramètre à régler, fini le réglage par un CV au tournevis !

Une diode de protection D1 en CMS protège l'entrée d'alimentation contre l'inversion de polarité. Le diviseur de tension à résistances R7 et R8 connecté à l'entrée analogique A7 servira à afficher la tension d'alimentation.

Un translateur de tension du bus I2C avec le jeu de transistors T1 et T2 offre la possibilité de connecter des périphériques supplémentaires sur le port P9 (extensions entrées/sortie, EEPROM, timer, capteurs, …). L'UART (liaison série asynchrone type RS232) est accessible sur le port P10 pour y connecter un module Bluetooth de type liaison asynchrone.

Enfin, tous les ports du Nano sont disponibles et chacun peut être découplé par une capacité à la masse afin de réduire les émissions HF conduites par la carte Nano. Seule l'entrée 14/A0 est protégée contre les surtensions par une résistance série R2 pour la commande d'émission (PTT_IN). Le potentiomètre d'accord en fréquence sera raccordé sur le port P8, et le bouton de fonction (qui fait contact à la masse) sur le port P5.

L'afficheur LCD de type parallèle a été conservé pour la simple raison qu'il est bien plus courant et moins onéreux que le modèle à bus série I2C. Le contraste se règle par R1 et le courant dans le rétroéclairage est limité par R3. Terminons cette description du schéma par les découplages de tension qui ont été renforcés ; notamment par C11 qui améliore grandement la qualité de la conversion analogique / numérique du Nano. La mention « NA » en tant que valeur signifie « Non Applicable », le composant n'est pas monté pour la version de base appliquée au Rocket !

Le circuit imprimé :

Le routage de la carte en figure 4 utilise tout l'espace possible avec les composants sur une seule face sauf pour l'ajustable de contraste et les connecteurs qui se trouvent au dos. Le plan de masse a été maillé autant que possible par des trous métallisés entre les deux faces pour réduire le rayonnement et son impédance. Les trous de 3 mm accueillent des entretoises ou vis M3 pour assembler l'empilement contre la façade. Etant donné le pas très rapproché des broches du Si5351 (0,5 mm), il est inutile de s'aventurer à faire soi-même le circuit imprimé. L'auteur distribue un circuit imprimé de qualité professionnelle prêt à souder.

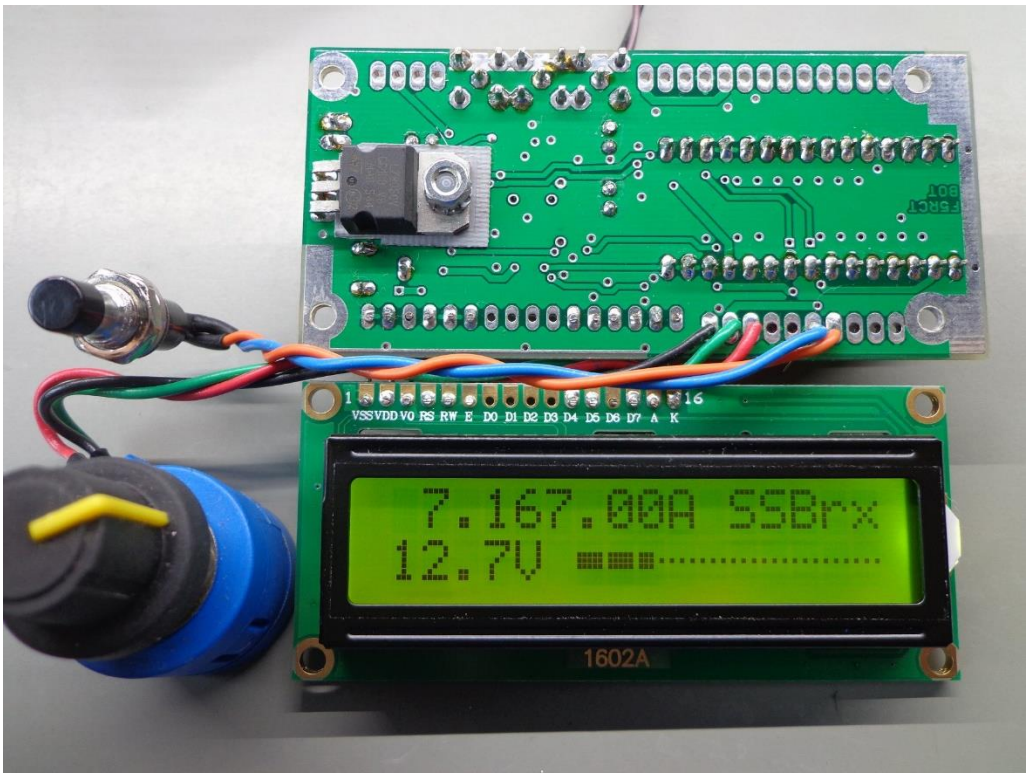
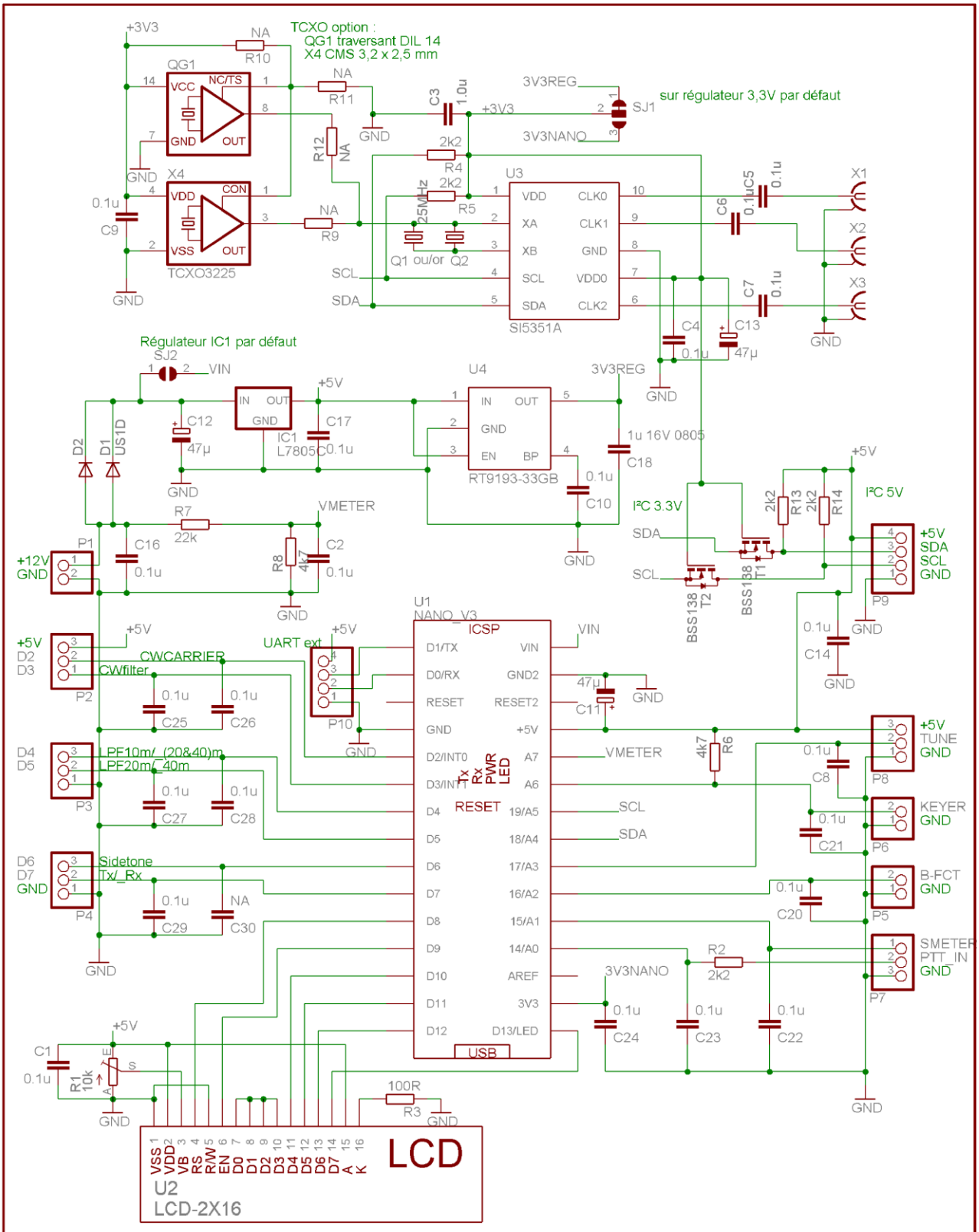


Figure 3 : Le VFOduino monté à plat pour le projet du Rocket.


TCXO option :
QG1 traversant DIL 14
X4 CMS 3,2 x 2,5 mm
sur régulateur 3,3V par défaut
Régulateur IC1 par défaut
TCXO3225
SI5351A
RT9193-33GB
L7805
NANO_V3
ICSP
USB
LCD
U2
LCD-2X16
I²C 3.3V
I²C 5V
CWCARRIER
CWfilter
LPF10m/_(20&40)m
LPF20m_40m
Sidetone
Tx/_Rx
UART ext
N/A : non applicable en version de base
VFOduino Si5351 - F5RCT
TITLE: VFO-SI5351-Nano-A
Document Number:
REV: A
Date: 26/09/2019 21:17:27
Sheet: 1/1

Une notice de montage complète se trouve sur le site « **F5KAV** » dans la rubrique Articles puis dans le dossier Réalisations/Rocket/VFOduino. Vous y retrouverez les fichiers du programme (.ino) de mon projet Rocket de transceiver HF.

La prise USB du Nano reste accessible si le module est placé contre le flanc droit du boitier. Elle peut servir à de multiples usages : reprogrammation, monitoring série pour débugger, et surtout pour le mode « CAT » (Computer Aid Transceiver) pour piloter le VFO par ordinateur.

Le régulateur 5 V est plaqué contre le plan de masse au verso avec une languette de couplage thermique et une vis nylon. Il ne chauffe pas avec la charge du VFO seul qui consomme 45 mA et un afficheur haute luminosité.
Le module processeur Nano peut être pris entre la carte et le LCD, ou derrière la carte, mais on ne verra plus ses LED de statut. L'afficheur LCD est prévu pour être empilé ou relié par des fils, dans ce cas on pourra se passer des liaisons D0 à D3.

La soudure des composants sur le circuit imprimé devra se faire en premier par le S15351, le régulateur 3,3 V et le reste des composants CMS, et enfin les composants traversants. Pour bien se matérialiser l'empilement, on pourra faire un assemblage avec les connecteurs sans les souder.
Notez que l'Arduino Nano n'a que deux fois 15 broches ce qui n'est pas le standard des connecteurs 16 broches courants. On pourra soit couper une broche aux connecteurs ou bien profiter des deux trous dans le plan de masse en bordure du circuit imprimé pour souder directement des connecteurs 16 broches. Puis on bouchera les deux trous non utilisés dans la partie femelle pour détromper le raccordement. Pour ma part, j'achète l'Arduino Nano en Asie (**V3.0 Mini USB ATmega328P CH340**) en version « DIY » (connecteurs à souder soi-même) pas seulement pour le prix mais pour permettre de souder les deux barrettes dans la configuration désirée, ceci sans le connecteur ICSP puisque je le programme par le câble USB. Les versions asiatiques se programment avec la configuration *Outils/Processeur/**Atmega328P (old bootloader)***. **Ce projet n'est pas compatible avec d'autres versions de l'Arduino Nano comme en ATMEGA168P ou MH-Tiny ATTiny88.**

Sur les sorties HF, on peut souder des embases SMA ou SMC, coudées ou droites. Le plan de masse en bordure n'est pas recouvert de vernis pour y souder directement des câbles coaxiaux.

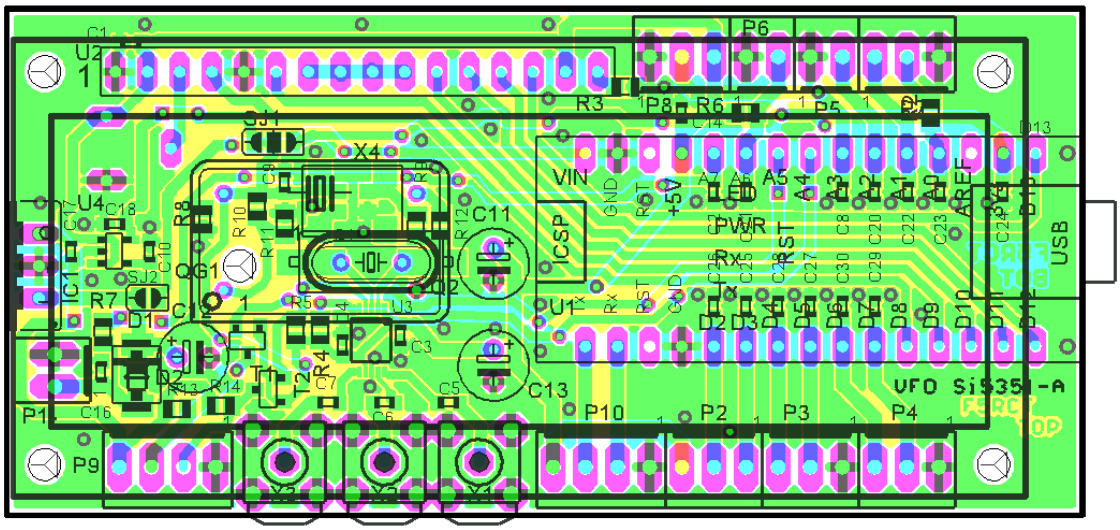


Figure 4 : Routage ultra-compact de la carte VFOduino.

Comment piloter le Si5351 ?

Une librairie complète a été développée et partagée sur Github [5]. Cette librairie a le mérite de pouvoir accéder à des configurations avancées mais elle prend 8 ko de mémoire programme, soit un quart de la capacité du Nano (Atmega 328) !
Pour nos applications, Jerry Gaffke KE7ER a développé des fonctions bien moins gourmandes qui ne demandent que 600 octets à la place d'une librairie. Le temps de réponse pour changer de fréquence est bien plus rapide tout en conservant une résolution en fréquence de 1 Hz ! Ces routines se trouvent au début de mon programme ou du BiTx40 [6]. (ctrl F : KE7ER) dans le fichier de la dernière version *raduino_v1.29.ino,* ou *ubitx_v5.1_code.ino*. J'en profite pour vous faire remarquer que ma carte de VFO diffère du Raduino, et que le programme uBiTx pourrait être utilisé à condition de modifier l'affectation des entrées/sorties.

Les exemples de code ci-après sont extraits du programme du VFOduino pour le Rocket. Dans la déclaration des constantes des routines de KE7ER, on paramètre l'une ou l'autre fréquence de quartz avec le rapport de multiplication correspondant. Dans cet exemple un quartz de 25 MHz est utilisé avec un ratio 35.

```
// If using 27mhz crystal, set XTAL=27000000, MSA=33.  Then vco=891mhz
#define SI5351BX_XTAL 25000000          // Crystal freq in Hz
#define SI5351BX_MSA  35
```

La précision relative d'un quartz suffit amplement pour la bande HF, mais comment le caler en fréquence puisqu'il n'y a pas de capacité d'ajustage ?

Il y a deux modes d'ajustages à prendre en compte :

En premier lieu, le calage par la capacité de charge spécifiée par le constructeur du quartz. Il se fait par le paramétrage du Si5351 à la mise sous tension. Toujours dans les routines de KE7ER, il est possible de paramétrer la capacité de charge du quartz pour s'approcher au plus près d'une fréquence connue que l'on mesure. Je vous conseille de laisser la valeur à 3 pour un quartz HC49U. Les capacités de charge faibles sont destinées aux boîtiers CMS très petits. Dans la déclaration des constantes on définit ce paramètre :

```
#define SI5351BX_XTALPF  3              // 1:6pF,  2:8pF,  3:10pF
```

Un peu plus bas on trouve un jeu de variables qui servent entre autre à définir **le niveau de chaque sortie**.

```
uint8_t  si5351bx_drive[3] = {1, 1, 1}; // 0=2mA, 1=4mA, 2=6mA, 3=8mA for CLK 0,1,2
```

Pour limiter les raies parasites il est conseillé de ne pas dépasser le niveau 2.

A l'initialisation dans le `setup ()` il suffit d'appeler en premier la fonction `si5351bx_init();` pour initialiser le circuit.

L'étalonnage en fréquence se fait par une procédure de calibrage dans l'application même. On paramètre une des trois sorties à une fréquence « ronde » que l'on va mesurer et compenser par le réglage d'une variable. Par exemple, on paramètre la sortie 0 à 10 MHz sur laquelle on raccorde un fréquencemètre.

```
si5351bx_setfreq(0, 10000000L);
```

Puis dans une boucle on met à jour la variable correction (entier signé type int) à chaque ajustement (codeur, boutons +/-,…). La figure 5 montre l'affichage à cette étape du Rocket. La valeur 800 est l'image de la variable correction que l'on règle pour mesurer précisément 10 MHz sur la sortie CLK0. On mémorise le réglage par la commande d'émission ou bien l'on sort par le bouton de fonction.

```
si5351bx_vcoa = (SI5351BX_XTAL * SI5351BX_MSA) + correction; // modifie la base
si5351bx_setfreq(0, 10000000L); // et on réactualise la fréquence
```

La plage de la variable correction s'étend sur +/- 131072 (entier signé), ce qui correspond à +/- 146 ppm avec une résolution de 0,3 ppm, soit un étalonnage à 1 Hz près pour 30 MHz !
Cette variable correction est mémorisée en EEPROM et rappelée à la mise sous tension dans la variable cal. La ligne ci-dessous est calculée dans le setup () avant de programmer une fréquence dans le reste de l'application loop ().

```
si5351bx_vcoa = (SI5351BX_XTAL * SI5351BX_MSA) + cal; // applique le facteur de correction mémorisé en EEPROM
```

Figure 5 : Affichage pendant l'étalonnage du VFOduino de F5RCT.

La programmation d'une fréquence s'effectue en une ligne ! Dans le programme principal, il suffit de programmer les sorties à la fréquence désirée par un entier long (int long). Tout le programme devra travailler sur des nombres entiers pour la fréquence et non pas des nombres flottants qui ne sont pas acceptés par les routines de programmation en fréquence.

```
si5351bx_setfreq(0, Freq_sortie0);
si5351bx_setfreq(1, Freq_sortie1);
si5351bx_setfreq(2, Freq_sortie2);
```

Pour couper une sortie il suffit de mettre la fréquence en question à 0 :

```
si5351bx_setfreq(1, 0); //coupe la sortie 1
```

Une fois configuré, les bases de programmation de ce VFO sont finalement simples. Le reste du temps le programme ne demande qu'à s'occuper de l'interface utilisateur.

Sur le schéma, le jeu de résistances R7 et R8 envoie une fraction de la tension d'alimentation sur une entrée analogique pour une fonction voltmètre. La calibration de ce voltmètre consiste à ajuster le facteur VmeterCal autour de la valeur nominale 146. Dans la fonction ci-dessous le décalage à droite de 7 réalise une division sur un entier non signé à virgule fixe. Le résultat à afficher est en dixièmes de Volts. La fonction sprintf formate l'entier en une chaîne de caractères qui sera affichée. On peut aussi utiliser des nombres flottants au détriment de la place en mémoire et du temps de calcul.

```
#define VmeterCal     146 // calibration factor for Voltmeter
char meter[17];
unsigned int  Vmeter;
(…)
Vmeter = analogRead(ANALOG_VMETER) >> 2; // 1023/4=255.75 for 25.6 Vmax
Vmeter = Vmeter * VmeterCal; //apply correction factor
Vmeter = Vmeter >>7 ;        // divide as fixed point integer
sprintf(b, "%3d", Vmeter);   // convert to char string
sprintf(meter, "%.2s.%.1s", b, b+2); // formate the string
strcat(meter, "V ");         // add symbol Volt
```

En début d'article j'évoquais le bruit de phase qui est le gage de qualité d'un oscillateur. Rappelons que ce circuit se constitue en premier d'une PLL qui multiplie la fréquence du quartz de 25 MHz par 35 pour obtenir 875 MHz. Les sorties sont obtenues par division entière ou fractionnaire du VCO calé à 875 MHz. Plus on divise par un facteur entier, plus le bruit de phase devient meilleur. Des mesures avaient été effectuées par KE5FX relatées dans le blog de NT7S [7]. Le niveau de bruit de phase est tout à fait exploitable dans nos applications pour ce type de circuit très abordable ! En revanche la densité de bruit proche de la porteuse pour un offset inférieur à 100 Hz pourrait être liée à la stabilité à court terme de l'oscillateur à quartz et sa dérive en température, ou bien lié la méthode de mesure qui n'est pas réellement précisée. Le Si5351 appartient à la même famille que le Si570 employé dans le transceiver KX3 d'Elecraft.

Ce VFO a été conçu pour le Rocket, un transceiver SSB et CW de poche qui couvre toute la bande HF avec une FI à double conversion (45 MHz et 12 MHz)et qui comprend un PA large bande de 10 W.
Olivier F4HTB s'est servi de cette carte pour le générateur d'horloge du websdr F4KIJ. Avec un PA de 1W, il a également réalisé une balise 50 MHz.

Nous venons de réaliser un *shield* (bouclier, écran). Dans ce jargon, les shields pour sont des cartes qui se raccordent aux cartes des processeurs Arduino. Elles sont particulièrement adaptées pour le monde des *Makers* et du DIY (*Do It Yourself*). Les shields Arduino conservent l'esprit original d'Arduino : compatible avec une carte processeur Arduino, facile à reproduire et à utiliser. Le circuit imprimé de ce shield sans le Si5351 a également servi pour piloter le PA 100 W du Rocket. On peut aussi imaginer en faire un wattmètre HF numérique, un TOSmètre, un décodeur CW, un thermomètre ou une horloge UTC. Tant d'applications pour une toute petite platine !

F5RCT Jean-Matthieu Stricker
contact : f5rct.jm(@)gmail.com

La notice de montage, le manuel d'utilisation et l'application Arduino se trouvent sur "F5KAV.fr": https://www.f5kav.fr/articles/articles-f5rct/ , puis dossier *Réalisations/Rocket/VFOduino.*

Liens utiles en rapport avec le VFOduino :

[1] Bitx40 transceiver à chaîne d'amplification bi-directionnelle. Raduino : module VFO à SI5351 et Arduino : http://www.hfsignals.com/

[2] Datasheet : https://www.silabs.com/documents/public/data-sheets/Si5351-B.pdf
Note d'application : https://www.silabs.com/documents/public/application-notes/AN619.pdf

[3] Module Si5351 : https://learn.adafruit.com/adafruit-si5351-clock-generator-breakout/overview

[4] Référence 10 MHz : https://www.zachtek.com/post/2019/02/14/using-the-si5351-directly-with-a-10mhz-ocxo-for-increased-stability

[5] Librairie complète Si5351 : https://github.com/etherkit/Si5351Arduino

[6] Routines de KE7ER pour le Si5351 : https://github.com/amunters/bitx40

[7] Mesures de bruit de phase : http://soldersmoke.blogspot.com/2015/08/going-through-phase-on-phase-noise.html

Où trouver les composants ?

Sur Ebay.com effectuer les recherches : « arduino nano atmega328P »; « RT9193-33GB » ; « BSS138 MOS » ; « crystal 25 MHz HC-49 » s'assurer quand même des tolérances et de la plage de température, « LCD 2x16 », « 16 Pin 2.54mm Pitch Single Row Female Straight », « 16 Pin 2.54mm Single Row male header », « 2.54mm Male 40 Pin Single Row Straight Round Pin Header Strip ».
Radiospares ou Digikey : SI5351A-B-GT