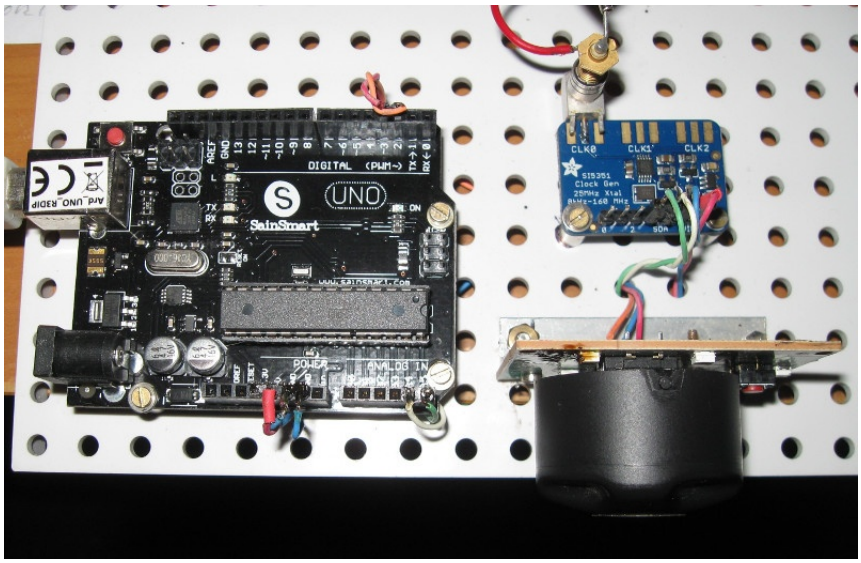
Ein universeller Frequenzerzeugungsbaustein mit einem Si5351A von Adafruit (FA) und einem Digispark (R)

Bei einem Treffen mit OM Heinz DL4AKI kam mir die Idee mit dem programmierbaren Baustein zur Frequenzerzeugung.
Die Anregung dafür hatte ich hier gefunden: http://www.kh-gps.de/si5351.htm

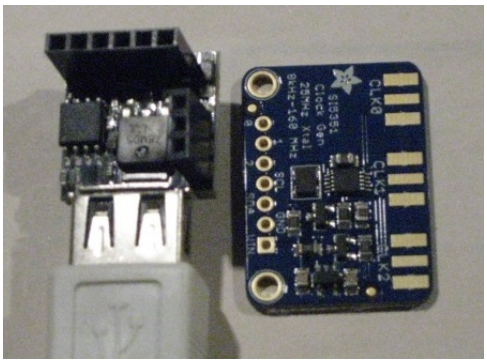
Ein erster Aufbau mit einem Arduino Uno sah dann so aus:



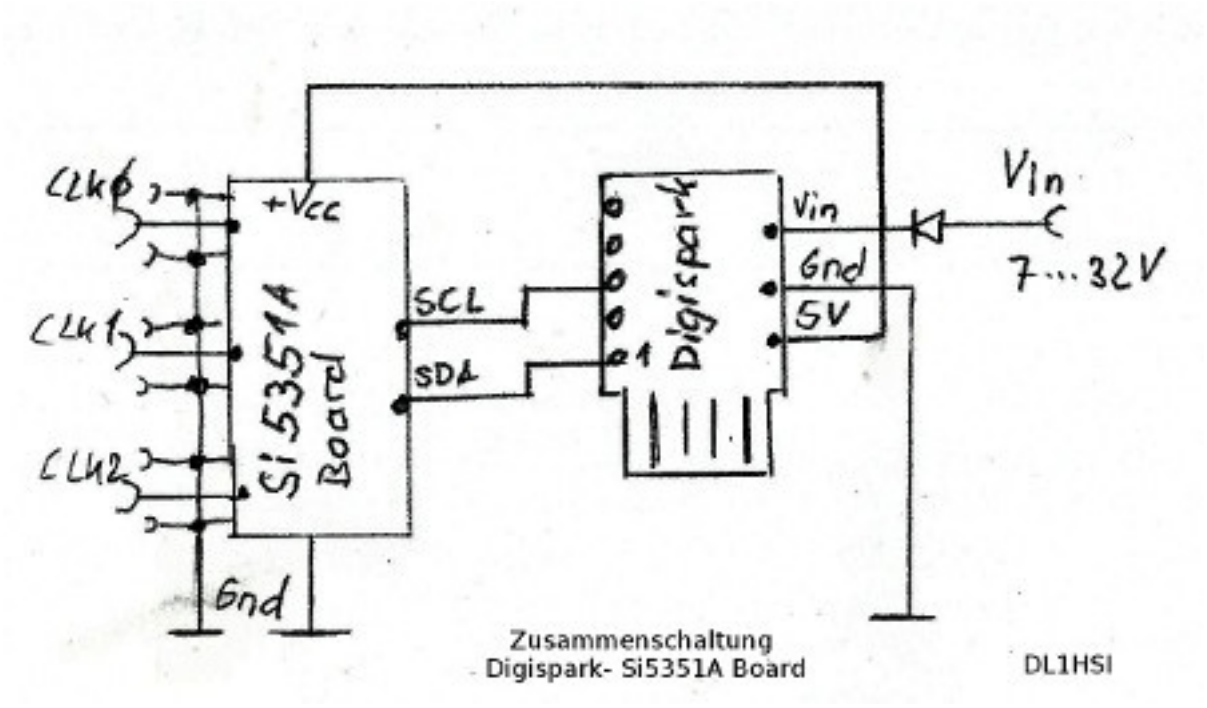
Das war mir aber etwas zu groß! Von OM Carsten bekam ich den Hinweis, dass bei (R) ein wesentlich kleinerer Baustein, der Digispark Mini, angeboten wird.

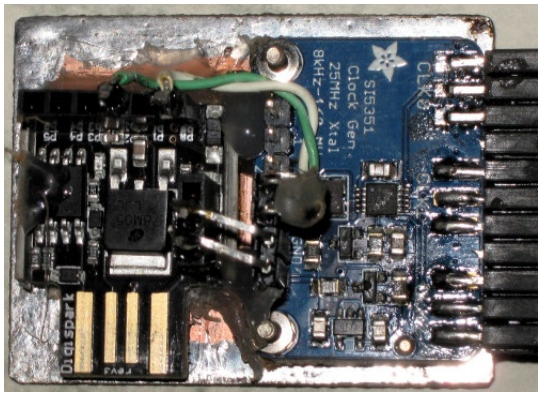
Eine sehr gute Anleitung für den Digispark gibt es ebenfalls bei der Fa. Reichelt als PDF:
https://www.reichelt.de/index.html?ACTION=3;ARTICLE=192128;SEARCH=digispark#av_tabdata

Einen Digispark beschafft und alles mal zusammengebracht:



Das war schon etwas kleiner und gefiel mir.
Also beide LPs auf eine kleine Platine montiert und verdrahtet.



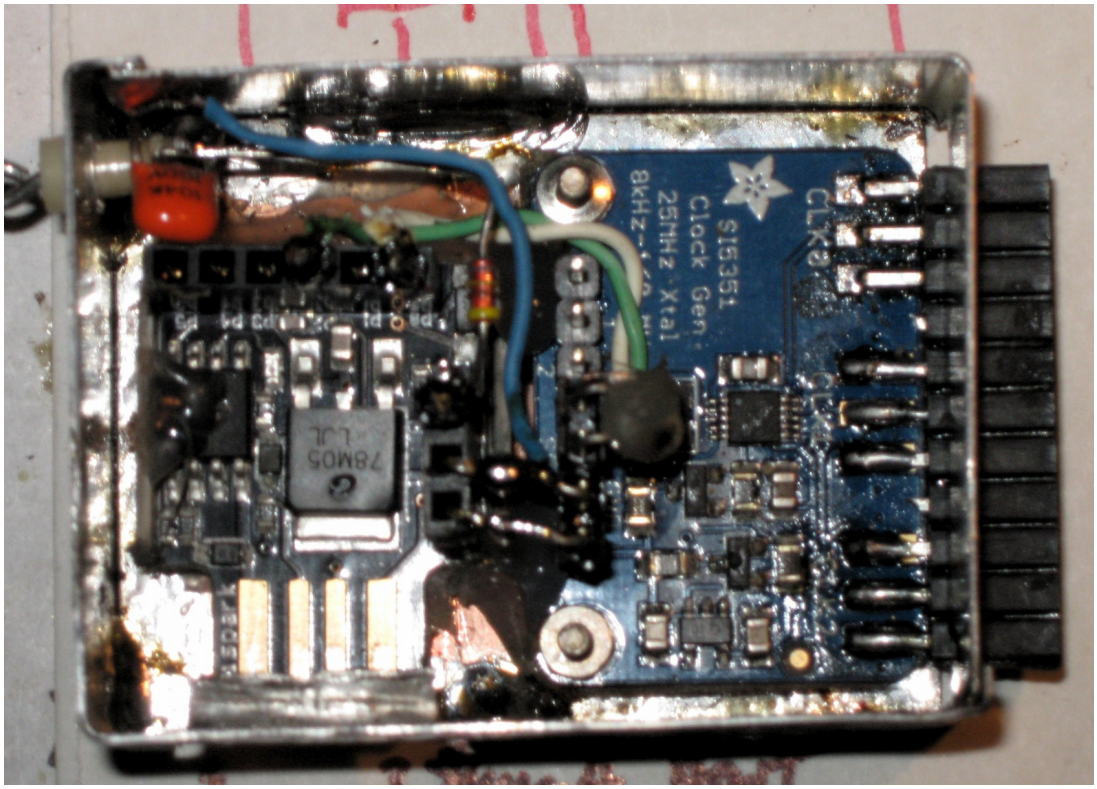


Noch eine kleine Abschirmung darum und die Spannungsversorgung herangeführt, dabei für die USB-Buchse natürlich einen Ausschnitt gelassen und es kann mit der Programmierung losgehen.

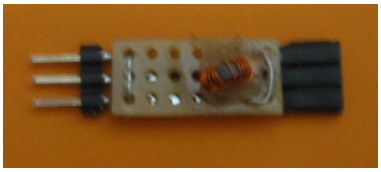
In meinem Fall (für den Pionier 6 Konverter) wurde auf CLK0 die 3,400 MHz programmiert, auf CLK2 die BFO-Frequenz (für das K455J Filter) auf 451,86 kHz programmiert.
Die auf CLK1 programmierten 17,5 MHz sind für einen eventuell zukünftigen 21 MHz Konverter vorgesehen.

Das soll lediglich ein Beispiel sein, es kann da jeder seine gewünschte Frequenz programmieren! Ohne jemals langwierig einen Quarz zu suchen oder irgendwo anfertigen zu lassen.

Mit geringfügigen Änderungen an der Platine (mehr dazu in einem weiteren Artikel) erzeugt das Adafruit Si5351A-Board ein relativ nebenwellenarmes Ausgangssignal, welches für viele Anwendungen völlig akzeptabel ist.



Sollte am Ausgang noch ein Band- oder Tiefpass gewünscht sein, ist eine Berechnung dafür z.B. hier zu finden: http://www.electronicdeveloper.de



Ansicht der Filter als Beispiel für ca. 15 MHz.

Bezugsquellen:
(FA) www.funkamateur.de
(R) www.reichelt.de

Im Anhang die Arduino .ino für die Ausgabe obiger Frequenzen.
Um die Stromaufnahme zusätzlich zu reduzieren, wird der Digispark am Ende der Initialisierung des Si5351A schlafen gelegt. Da es sich hier um Festfrequenzen handelt, müssen diese nur einmalig bei Programmstart (anlegen der Versorgungsspannung) in den Si5351A geschrieben werden.
Da der Flash im Digispark knapp ist, wird mit einer älteren Version der Si5351-Bibliothek von NT7S gearbeitet.
Eine Frequenzabweichung des Si5351-Referenzoszillators kann in der Zeile:

```
si5351.set_correction(220);  // Frequenzkorrektur in Hz bei 10 MHz
```

korrigiert werden.

S. Stengel, DL1HSI
Radisleben 28.10.2018

```
=====================================================================
// Board: Digispark (Default - 16.5 MHz)
#include "si5351_ds.h"   // NT7S alter Stand
#include "Wire.h"
#include <avr/sleep.h>

Si5351 si5351;

void setup()
{
  // Start serial and initialize the Si5351
  si5351.init(SI5351_CRYSTAL_LOAD_8PF);
  si5351.set_correction(220);   // Frequenzkorrektur in Hz bei 10 MHz

  // PLLA-Frequenz setzen
  si5351.set_pll(SI5351_PLL_FIXED, SI5351_PLLA);

  // CLK0 auf 3,4 MHz
  si5351.set_freq(3400000, SI5351_PLL_FIXED, SI5351_CLK0);

  // CLK1 auf 17,5 MHz
  si5351.set_freq(17500000, 0, SI5351_CLK1);

  // CLK2 auf 451,86 kHZ(BFO-USB-K455J)
  si5351.set_freq(451860, 0, SI5351_CLK2);

}

void loop()
{
  delay(500);
  set_sleep_mode(SLEEP_MODE_PWR_DOWN); // sleep mode setzen
  sleep_enable();
  sleep_mode();                        // sleep mode aktivieren
}
=====================================================================
```