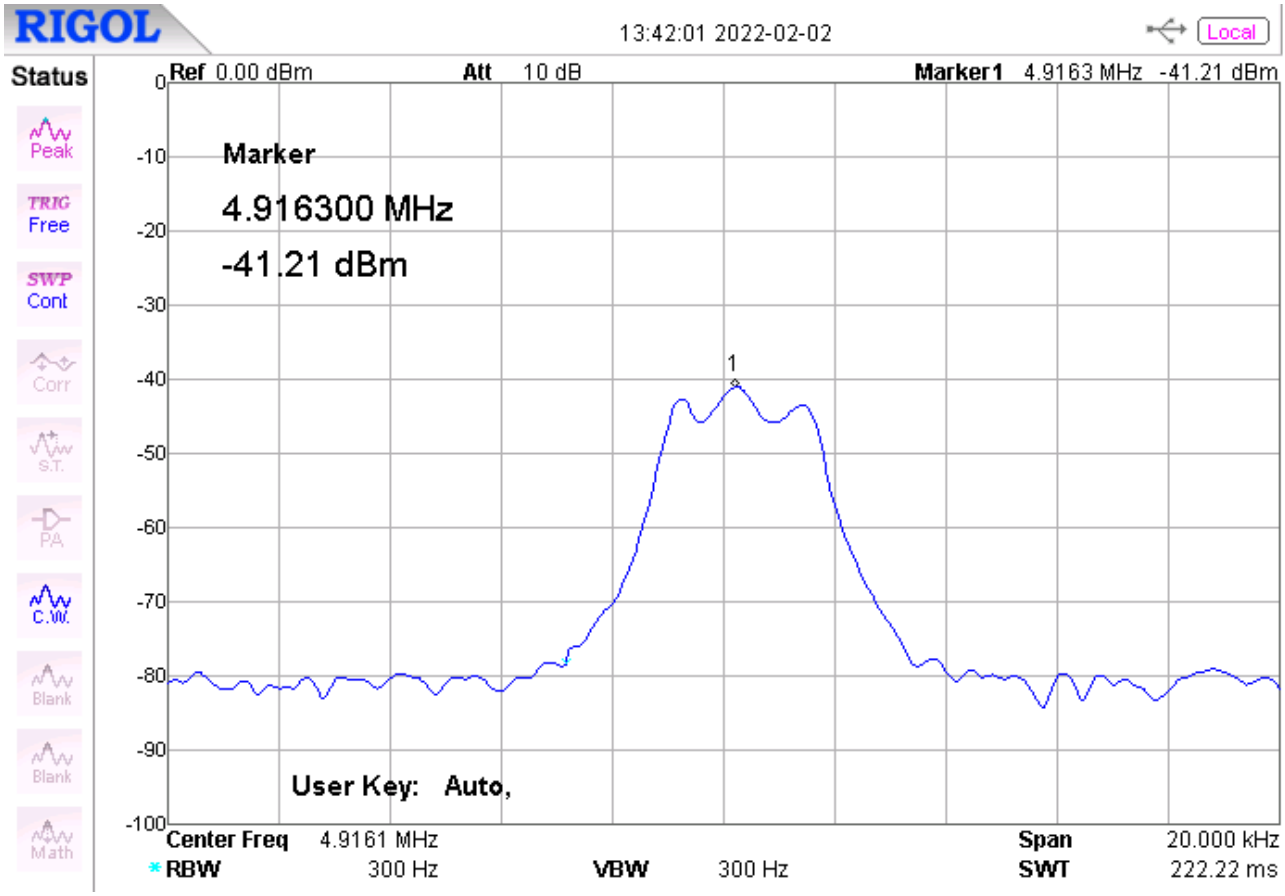
Vier Transistoren und ein Arduino

Ein Superhet für das 80m-Band

Nach dem Direktmischer mit einem quarzstabilen LO, der vom Arduino gesteuert wird, liegt es nahe, das Konzept aufzurüsten. Jetzt kommt ein selbst gebautes Quarzfilter ins Spiel. Mit vier preiswerten PC-Quarzen lässt es sich sehr einfach basteln. Leider hat man dann aber auch Durchlassverluste von 8 dB. Zwei zusätzliche Transistoren kompensieren diesen Leistungsverlust. Nun unterdrückt das Quarzfilter die benachbarten Träger hörbar besser als der einfache LC-Schwingkreis beim Direktmischer. Und die zweite BFO-Frequenz liefert der Si5351 auch noch, denn er hat drei unabhängige PLL- LOs eingebaut. Wir nutzen jetzt CKL 0 und CKL 1.

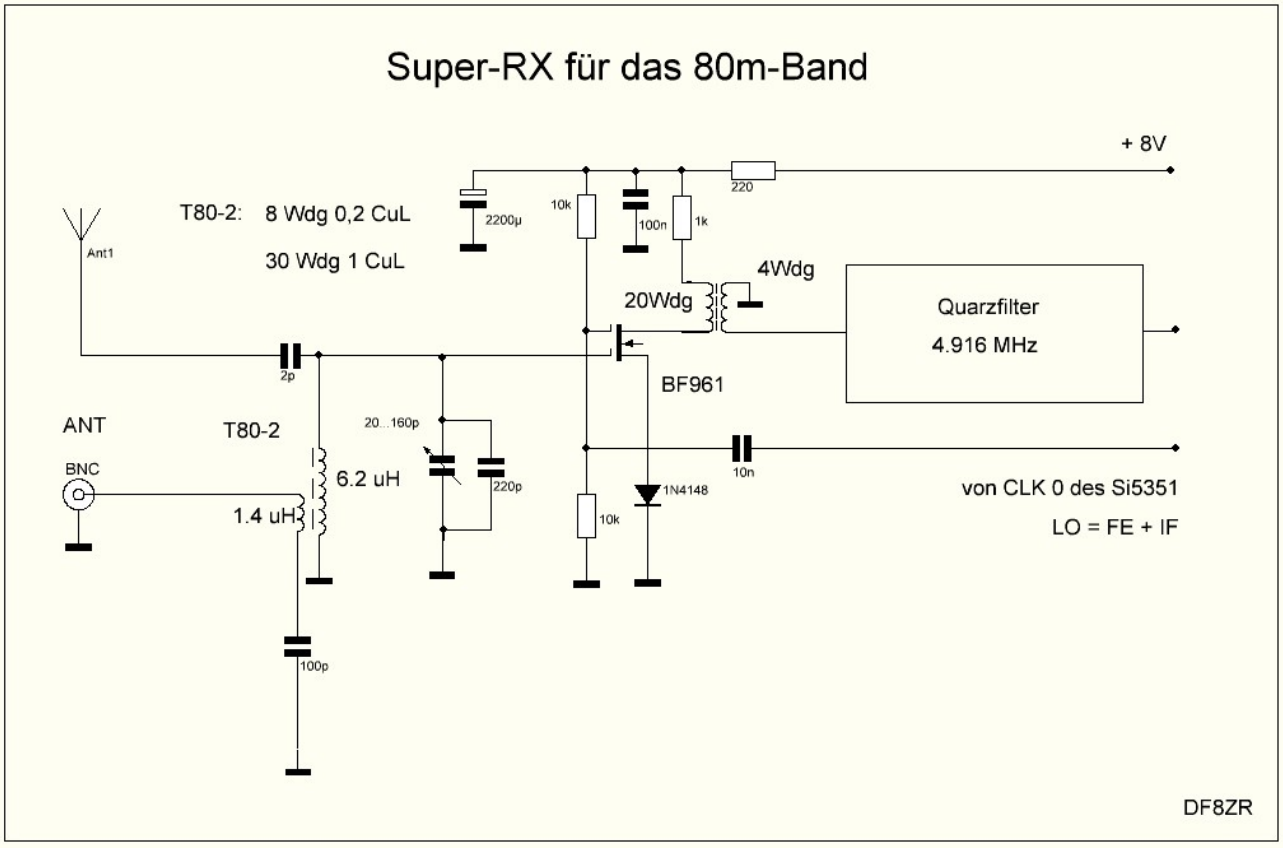
Das ZF-Filter



Es ist knapp 3 kHz breit. 4 Stck nicht ausgesuchte Quarze mit 4.916 MHz. Die Baubeschreibung findet man auf dieser Seite.

Tja, +20dB IF-Verstärkung haben zunächst nicht die gewünschte Lautstärke geliefert, es kam also noch ein Transistor hinzu.

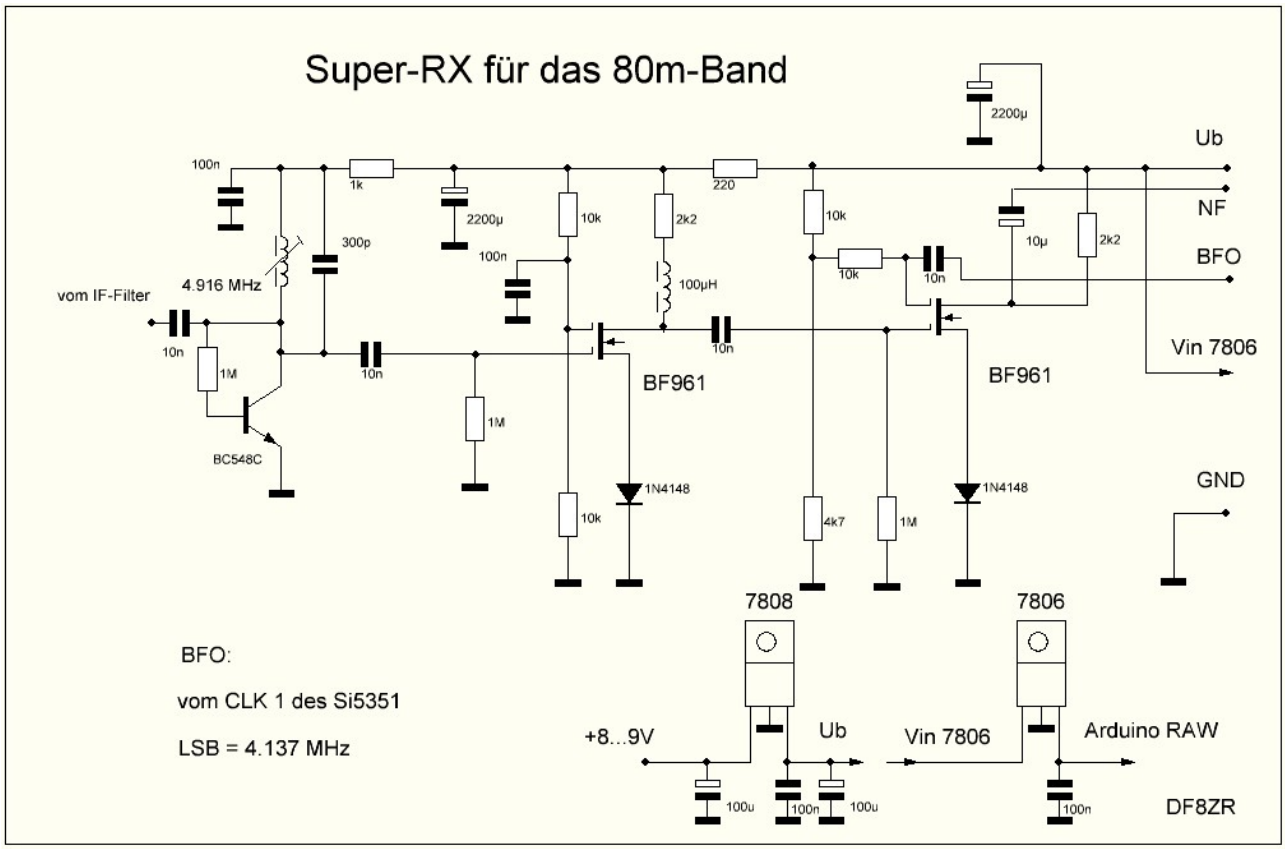
Die Schaltung
Wie beim Direktmischer, allerdings etwas erweitert. Es finden nun zwei Mischvorgänge statt.



Am Eingang ist nur ein Abstimmdrehko. Die Resonanz-überhöhung ist ca. 20 dB. Durch eine Verstimmung lässt sich auch so die Lautstärke herabsetzen.

Die variable Oszillatorfrequenz ist in Steps einstellbar und wird vom OLED angezeigt. Es findet eine Überlagerung statt: LO = EF + IF. Von (3.5 MHz + IF) … (3.8 MHz + IF).

Die hohe Frequenzstabilität erlaubt eine feine Abstimmung beim Empfang in SSB.



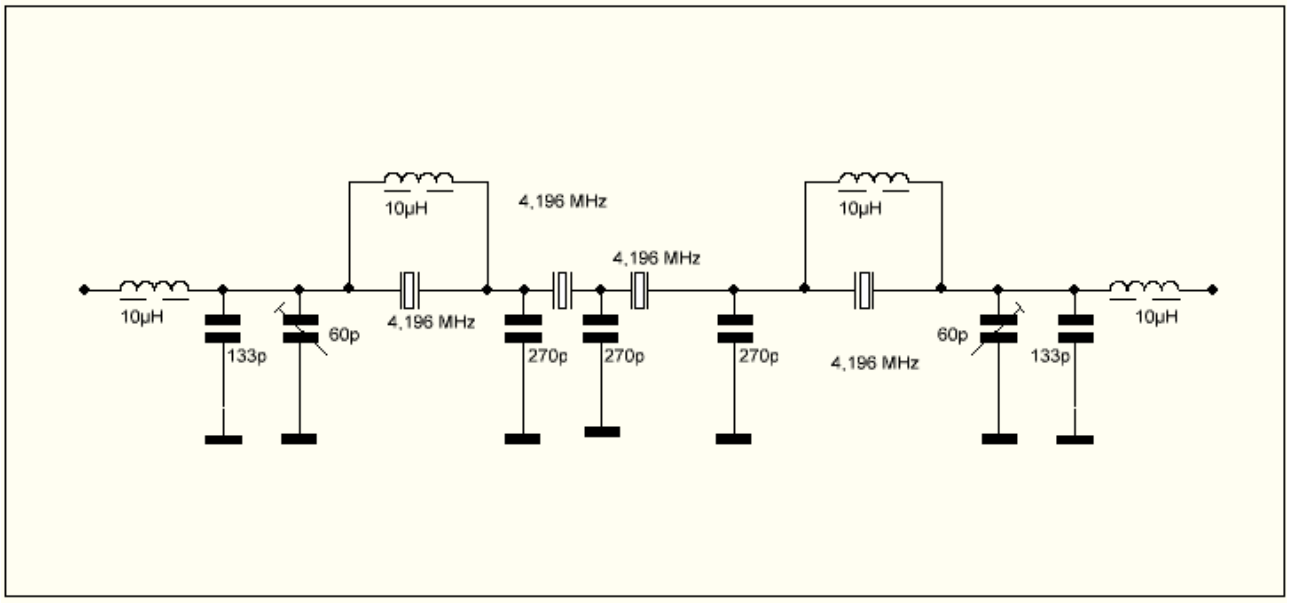
Hier sieht man den zweistufigen IF-Verstärker. Er sorgt für ausreichende Empfindlichkeit. Zur Dekodierung des SSB-Signals wird am Si5351 am CLK 1 das stabile BFO-Signal abgenommen. Es ist nicht veränderlich und liegt etwas tiefer als die IF-Frequenz. Die genaue BFO-Frequenz muss in der Software definiert werden. U.U. Sind dazu einige Versuche erforderlich, da der Si5351 stets geringe Frequenzabweichungen hat, die man durch eine Kalibrierung ausgleichen muss. Hier fand ich im BFO-Test-Betrieb die BFO von 4.137 MHz, bei der sich ein sauberer Ton einstellte.

Wiederum habe ich gleich zwei Stabis für die Stromversorgung vorgesehen, damit nicht wieder wilde Schwingungen entstehen. Das Radio sollte man mit 12...15V DC betreiben. Die Stromaufnahme ist max. 80 mA. Eine gute 9V-Blockbatterie würde bestenfalls zehn Stunden halten.

Quarzfilter

Es ist ein Ladderfilter, wie ich es bereits in meinem Mittelwellen-

RX(auch auf dieser Seite unter „Schaltungen“) gebaut hatte. Aber hier mit 4 statt nur 3 Quarzen.



Man muss event. etwas experimentieren, denn es ist für den 50-Ohm-Betrieb bemessen. Mit dem Ringkern am Eingang wird eine Anpassung vorgenommen. Wenn der Ton nicht verzerrungsfrei sein sollte, liegt es wahrscheinlich an der BFO-Frequenz, die nicht zum Filter(bzw. Quarzfrequenz des Si5351) passt. Hier muss man durch geringe Variation im „BFO-Testbetrieb“ die richtige Frequenz finden. Das Filter ist breit genug.

Ich habe in die Software Befehlszeilen eingearbeitet, die die Feinabstimmung der BFO-Frequenz ermöglichen. Sie sind im Kommentar mit „/****BFO-Test“ gekennzeichnet. Die Zeilen mit „/*****Nomalbetrieb“ müssen dann auskommentiert werden und umgekehrt, wenn man dahin zurück geht. Die Taste 100k erhöht die BFO-Frequenz, die Taste 5k erniedrigt sie in 100 Hz-Schritten. Die BFO-Frequenz wird angezeigt. Den Wert notieren und danach für den Normalbetrieb in die Software eintragen.

Film bei YouTube:

DF8ZR, im Jan. 2022
/

```
**********************************************************
**********************************************************
  10kHz to 120MHz VFO / RF Generator with Si5351 and
Arduino Nano, with Intermediate Frequency (IF)
  offset (+ or -). See the schematics for wiring details. By J.
CesarSound - ver 1.0 - Dec/2020.

  Private Anwendung und Anpassung bei DF8ZR im Jan. 2022:
Direktmischer für das 80m-Amateurfunkband.
  Mit Dank an den Autor J. Cesar, bei dem alle Rechte sind.

  BFO-Test: Hier werden die Tasten 100k und 5k für die
Feinabstimmung der BFO-Frequenz verwendet. Im BFO-Betrieb
  ehöht die Taste 100k in 100 Hz-Schritten und die Taste 5k
erniedrigt sie. Sie wird angezeigt. Den Wert notieren
  und nach der Rückstellung in den Normalbetrieb bei IF2
eintragen.
**********************************************************
**********************************************************/

//Libraries
#include <Wire.h>               //IDE Standard
#include <Rotary.h>             //Ben Buxton
https://github.com/brianlow/Rotary
#include <si5351.h>             //Etherkit
https://github.com/etherkit/Si5351Arduino
#include <Adafruit_GFX.h>       //Adafruit GFX
https://github.com/adafruit/Adafruit-GFX-Library
#include <Adafruit_SSD1306.h>   //Adafruit SSD1306
https://github.com/adafruit/Adafruit_SSD1306

//User preferences
//---------------------------------------------------------------------
---------------------------
#define IF  4916                //Enter your IF frequency, ex: 455 =
```

```
455kHz, 10700 = 10.7MHz, 0 = to direct convert receiver or RF generator, + will add and - will subtract IF offfset.
#define FREQ_INIT  3500000 //3600000 für BFO-Test   //Enter your initial frequency at startup, ex: 7000000 = 7MHz, 10000000 = 10MHz, 840000 = 840kHz.
#define XT_CAL_F   33000     //Si5351 calibration factor, adjust to get exatcly 10MHz. Increasing this value will decreases the frequency and vice versa.
#define tunestep   A0        //Change the pin used by encoder push button if you want.

#define IF2 49137             //BFO- Frequenz für LSB

#define FREQ_INIT2 0

#define ZEHN_PIN 9           //D9 Tasten für Steps
#define HUNDERT_PIN 8        //D5
#define KILO_PIN 7           //D6
#define FUENFK_PIN 6         //D7
#define HUNDERTK_PIN 5       //D8
//--------------------------------------------------------------------------------
----------------------------

Rotary r = Rotary(2, 3);
Adafruit_SSD1306 display = Adafruit_SSD1306(128, 64, &Wire);
Si5351 si5351;



unsigned long freq = FREQ_INIT;
unsigned long freqold, fstep;
long interfreq = IF;
unsigned long freqdiff = 1.05;//tatsächliche LO-Frequenz war 1.05 kHz zu hoch
//wird vor dem tunebefehl subtrahiert
```

```
unsigned long freq2 = FREQ_INIT2;  //für LSB

long interfreq2 = IF2;             //für LSB an Clock 2

//long freqbfo = 49135;  // Anfangswert ;
/*****************************BFO-Test = aktuelle BFO-
Frequenz


long cal = XT_CAL_F;
unsigned long long pll_freq = 90000000000ULL;
byte encoder = 1;
byte stp;
unsigned int period = 100;   //millis display active
unsigned long time_now = 0;  //millis display active

int tastenklick;
int hundert;//für frequency
int eink;//für frequency
int st5k;//für frequency
int hundertk;//für frequency
int zehn;//für frequency


ISR(PCINT2_vect) {
  char result = r.process();
  if (result == DIR_CW) set_frequency(1);
  else if (result == DIR_CCW) set_frequency(-1);
}

void set_frequency(short dir) {
  if (encoder == 1) {                    //Up/Down frequency
    if (dir == 1) freq = freq + fstep;
    if (freq >= 120000000) freq = 120000000;
    if (dir == -1) freq = freq - fstep;
```

```
    if (fstep == 1000000 && freq <= 1000000) freq = 1000000;
    else if (freq < 10000) freq = 10000;
  }
}

//**********************************************************
**********SETUP**********************

void statup_text() {
  display.setTextSize(1);
  display.setCursor(4, 5);
  display.print("Si5351");
  display.setCursor(4, 20);
  display.print(" CW/SSB-Superhet ");
  display.setCursor(4, 35);
  display.print("Vers. DF8ZR");
  display.setCursor(4, 50);
  display.print(">> 80m-Super <<");
  display.display();
  delay(3500);//3000
  display.clearDisplay();
}


void setup() {


  /*Serial.begin(9600); // Öffnet die serielle Schnittstelle bei 9600
Bit/s:
  Serial.print("BEGINN");
  Serial.print("\t");
  */
pinMode(ZEHN_PIN, INPUT_PULLUP);
pinMode(HUNDERT_PIN, INPUT_PULLUP);
```

```
pinMode(KILO_PIN, INPUT_PULLUP);
pinMode(FUENFK_PIN, INPUT_PULLUP);
pinMode(HUNDERTK_PIN, INPUT_PULLUP);


zehn=0;
st5k=0;
eink=0;
hundert=0;
hundertk=0;
tastenklick=0;//keine Taste betätigt

  Wire.begin();
  display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
  display.clearDisplay();
  display.setTextColor(WHITE);
  display.display();

  pinMode(2, INPUT_PULLUP);
  pinMode(3, INPUT_PULLUP);
  pinMode(tunestep, INPUT_PULLUP);

  pinMode(st5k, INPUT_PULLUP);///

  statup_text();

  si5351.init(SI5351_CRYSTAL_LOAD_8PF, 0, cal);
  si5351.output_enable(SI5351_CLK0, 1);                 //1 - Enable /
0 - Disable CLK
  si5351.output_enable(SI5351_CLK1, 0  );
  si5351.output_enable(SI5351_CLK2, 1);                 //aktiviert

  si5351.drive_strength(SI5351_CLK0,
SI5351_DRIVE_2MA);  //Output current 2MA, 4MA, 6MA or
8MA
```

```
  si5351.drive_strength(SI5351_CLK2,
SI5351_DRIVE_2MA);   // 2 mA for HB mixers///eingefügt für
Superhetbetrieb

  si5351.set_freq_manual((freq2 + (interfreq2 * 100ULL)) *
100ULL, pll_freq, SI5351_CLK1);//LSB = BFO

  PCICR |= (1 << PCIE2);
  PCMSK2 |= (1 << PCINT18) | (1 << PCINT19);
  sei();

  stp = 3;
  setstep();
  layout();
  displayfreq();
}



//##################################################################
##########LOOP
void loop() {

 if (tastenklick == 0){

  if (freqold != freq) {
    time_now = millis();
   tunegen();
    freqold = freq;
    }


  if (digitalRead(tunestep) == LOW) {
    time_now = (millis() + 300);
```

```
    setstep();
    delay(300);
  }

}

if (digitalRead(ZEHN_PIN) == LOW) {
    time_now = (millis() + 300);

 zehn = 1;
 tastenklick=1;

 tasten();

}

if (digitalRead(HUNDERT_PIN) == LOW) {
    time_now = (millis() + 300);

 hundert = 1;
 tastenklick=1;

 tasten();

}

if (digitalRead(KILO_PIN) == LOW) {
    time_now = (millis() + 300);
tastenklick=1;
 eink = 1;
 tasten();

}

if (digitalRead(FUENFK_PIN) == LOW) {
```

```
    time_now = (millis() + 300);
tastenklick=1;  //Normalbetrieb
//tastenklick=0;//**********************BFO-Test
 st5k = 1;
 tasten();

}

if (digitalRead(HUNDERTK_PIN) == LOW) {
    time_now = (millis() + 300);
tastenklick=1;//Normalbetrieb

//tastenklick=0;//**************************BFO-Test
 hundertk = 1;
 tasten();

}

  if ((time_now + period) > millis()) {
    displayfreq();
    layout();
  }
  delay(100);//für tastensetzen, sonst Dauersetzen


}

//#########################################################
#########



void tunegen() {
  freq = freq-freqdiff;//Abweichung von der tatsächlichen
Frequenz, hier gemessen = +1,05 kHz
```

```
  si5351.set_freq_manual((freq + (interfreq * 1000ULL)) *
100ULL, pll_freq, SI5351_CLK0);
}



void displayfreq() {
  unsigned int m = freq / 1000000;
  unsigned int k = (freq % 1000000) / 1000;
  unsigned int h = (freq % 1000) / 1;

  display.clearDisplay();
  display.setTextSize(2);

  char buffer[15] = "";
  if (m < 1) {
    display.setCursor(41, 1); sprintf(buffer, "%003d.%003d", k, h);
  }
  else if (m < 100) {
    display.setCursor(5, 1); sprintf(buffer, "%2d.%003d.%003d", m,
k, h);
  }
  else if (m >= 100) {
    unsigned int h = (freq % 1000) / 10;
    display.setCursor(5, 1); sprintf(buffer, "%2d.%003d.%02d", m,
k, h);
  }
  display.print(buffer);
}

void setstep() {


  switch (stp) {
    case 1:
```

```
      stp = 2;
      fstep = 10;
      break;
    case 2:
      stp = 3;
      fstep = 100;
      break;
    case 3:
      stp = 4;
      fstep = 1000;
      break;
    case 4:
      stp = 5;
      fstep = 5000;
      break;
    case 5:
      stp = 6;
      fstep = 100000;
      break;
    case 6:
      stp = 1;
      fstep = 1000000;
      break;
  }
}
```

```
void tasten(){

if (zehn == 1) {

stp = 2;
      fstep = 10; // statt 10kHz-steps werden 5kHZ-steps eingestellt!
```

```
}


if (st5k == 1) {

stp = 5;
     fstep = 5000; // statt 10kHz-steps werden 5kHZ-steps
eingestellt!

    // bfo();//***************
}


if (hundert == 1) {

stp = 3;
     fstep = 100;
}

if (eink == 1) {

stp = 4;
     fstep = 1000;
}

if (hundertk == 1) {

stp = 6;
    fstep = 100000;

//bfo();       //**********************************BFO-Test

}
```

```
zehn=0;
hundert = 0;
eink=0;
st5k=0;
hundertk=0;
tastenklick=0;

tunegen();
freqold = freq;




void layout();

digitalWrite (ZEHN_PIN,HIGH);
digitalWrite (HUNDERT_PIN,HIGH);
digitalWrite (KILO_PIN,HIGH);
digitalWrite (FUENFK_PIN,HIGH);
digitalWrite (HUNDERTK_PIN,HIGH);

}

void layout() {
  display.setTextColor(WHITE);
  display.drawLine(0, 20, 127, 20, WHITE);
  display.drawLine(0, 43, 127, 43, WHITE);
  display.drawLine(105, 24, 105, 39, WHITE);
  display.setTextSize(2);
  display.setCursor(2, 25);
  display.print("ST:");
  if (stp == 2) display.print("10Hz"); if (stp == 3)
display.print("100Hz"); if (stp == 4) display.print("1k");
```

```
  if (stp == 5) display.print("5k"); if (stp == 6)
display.print("100k"); if (stp == 1) display.print("1M");
  display.setCursor(2, 48);
 //
display.print("BFO:");//************************BFO_Test
 // display.print("IF;");
 //display.print(interfreq);

 //display.print(freqbfo);//**********************BFO-Test


  display.print("80m-Band");
  //display.print("k");
  display.setTextSize(1);
  display.setCursor(110, 23);
  if (freq < 1000000) display.print("kHz");
  if (freq >= 1000000) display.print("MHz");
  display.setCursor(110, 33);
  if (interfreq == 0) display.print("VFO");
  if (interfreq != 0) display.print("LO");
  display.display();
}
```

//hier folgt Code, der nur für die Optimierung der BFO=-Frequenz unkommentiert werden soll /**********BFO-Test

```
/*void bfo(){




if (stp==6) {
```

```
freqbfo = freqbfo + 1;

si5351.set_freq_manual((0 + (freqbfo * 100ULL)) * 100ULL,
pll_freq, SI5351_CLK1);

}


if (stp==5) {

freqbfo = freqbfo - 1;

si5351.set_freq_manual((0 + (freqbfo * 100ULL)) * 100ULL,
pll_freq, SI5351_CLK1);


}*/
```