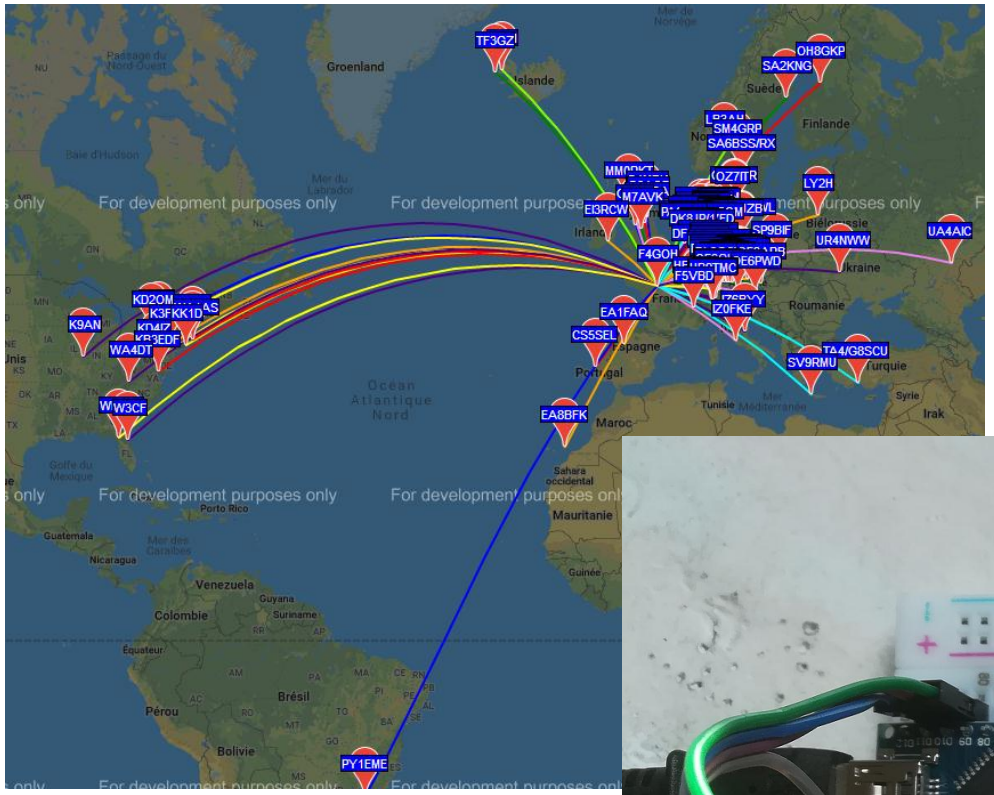
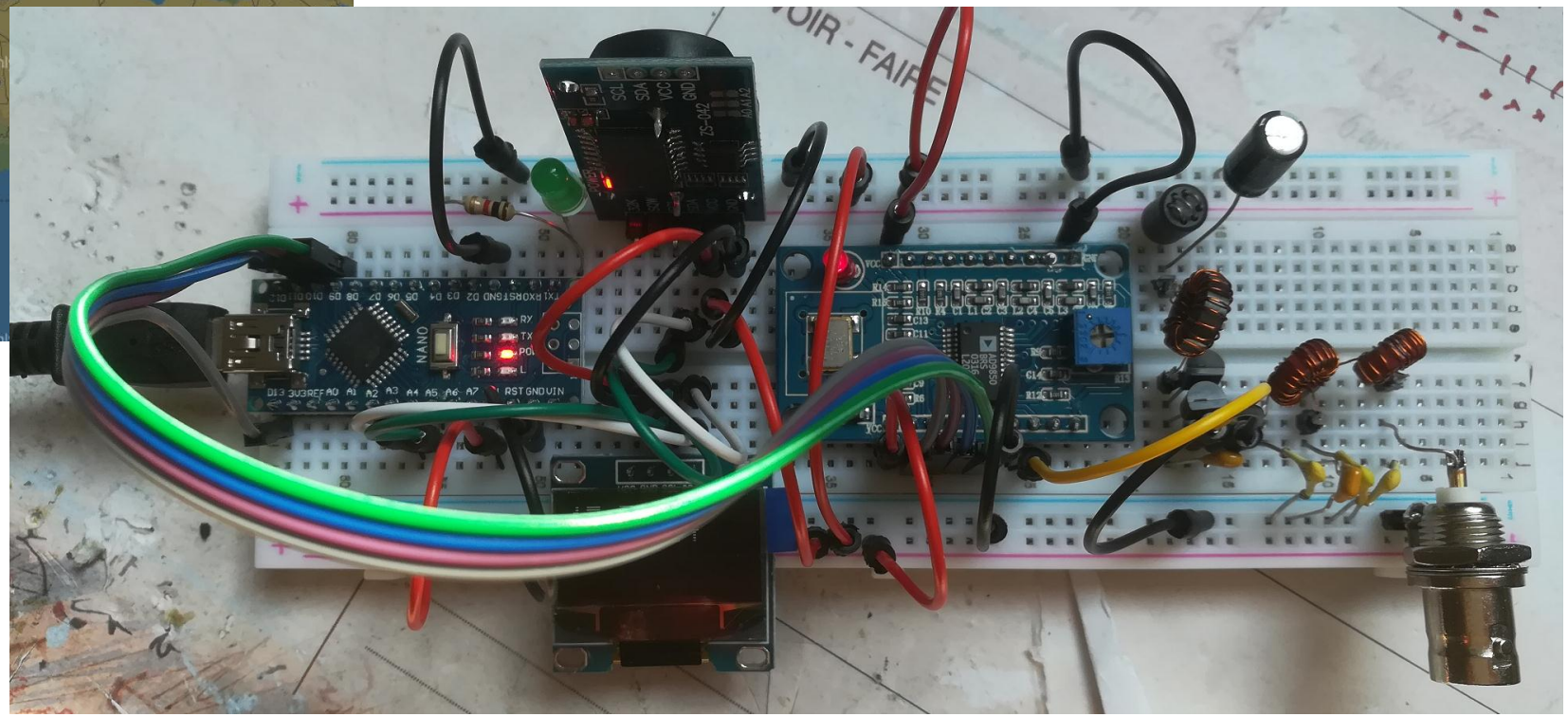
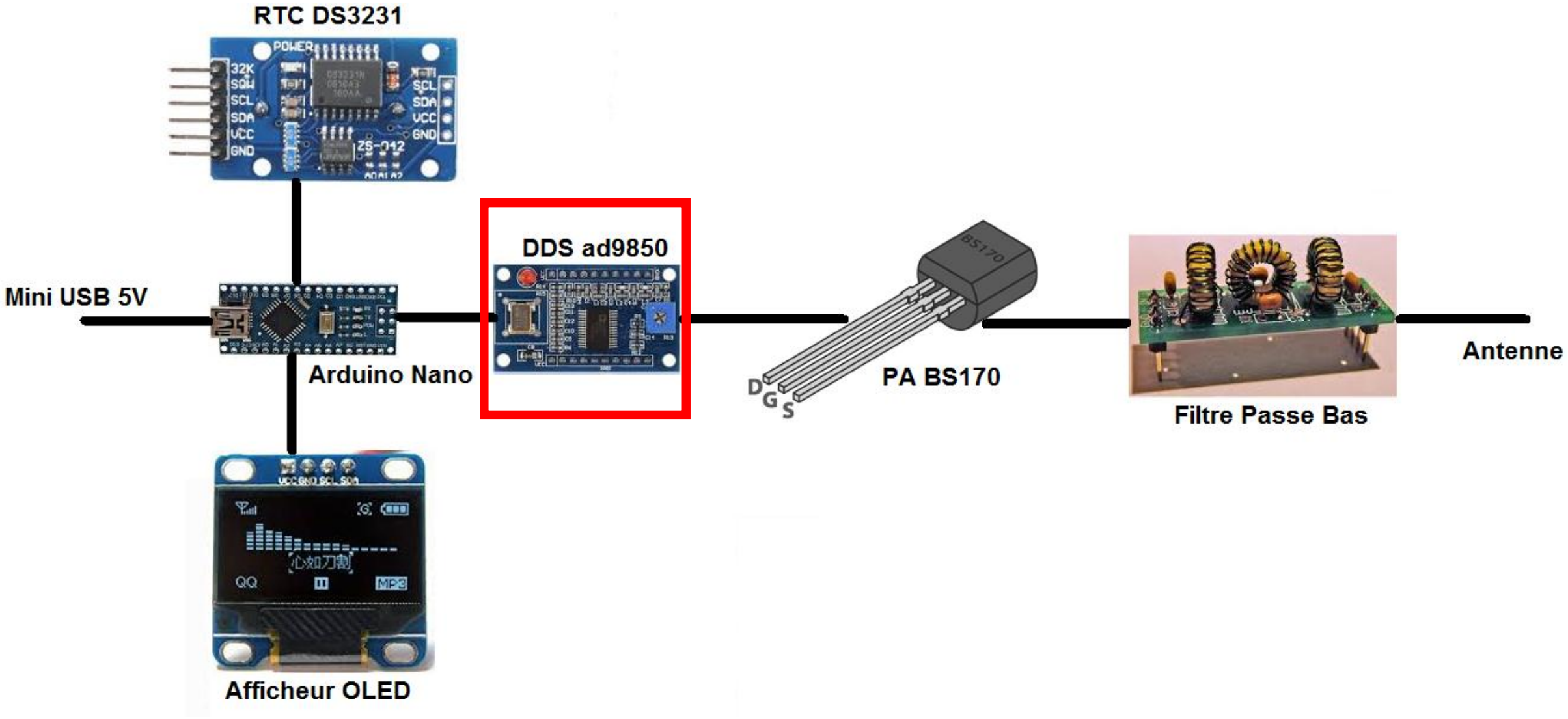
WSPR whisper : Weak Signal Propagation Reporter



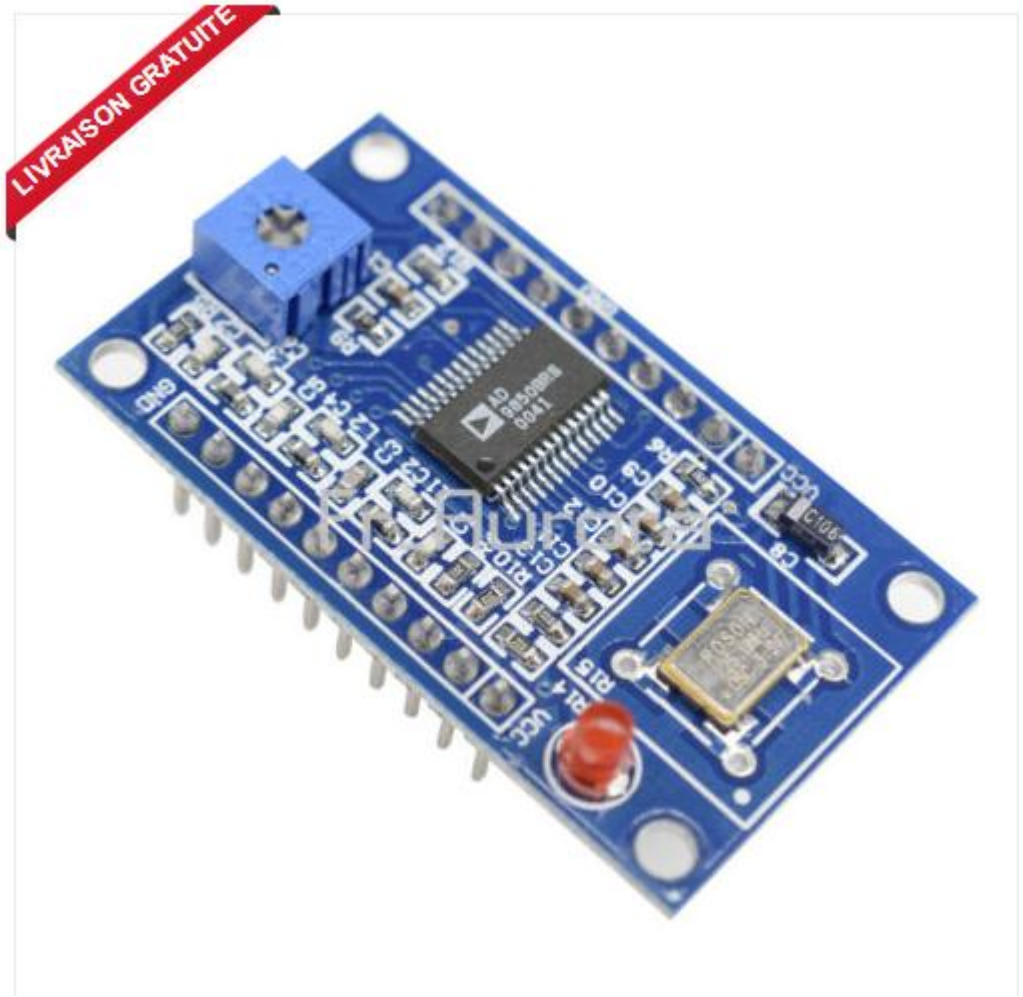
Oscillateur programmable DDS Protocole WSPR



WSPR Synoptique



DDS = Direct Digital Synthesis



AD9850 DDS Signal Generator Module Test Equipment Sine Square Wave

État : **Neuf**

Quantité : 1

5 disponible(s)
34 objets déjà vendus

12,60 EUR

Achat immédiat

Ajouter au panier

Ajouter à votre liste d'Affaires à suivre

Livraison gratuite	Plus de 86 % vendus	34 objets déjà vendus

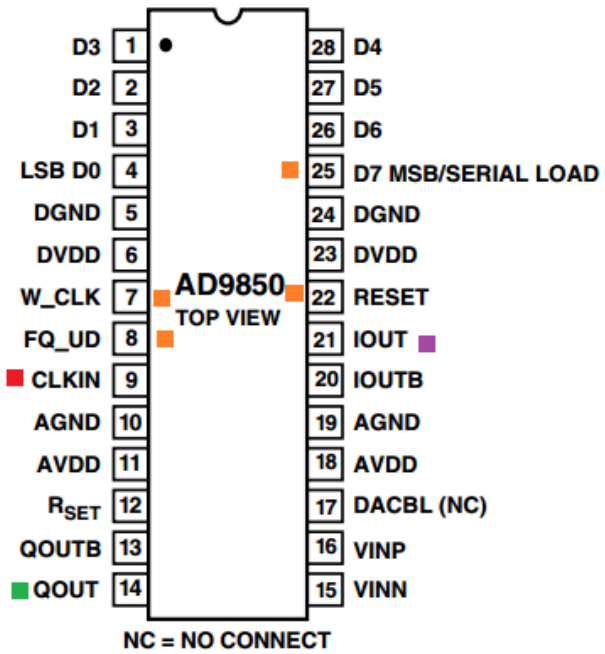
Livraison : **GRATUIT** Autres | Détails
Cliquez ici pour en savoir plus sur la livraison internationale.
Lieu où se trouve l'objet : ShenZhen, Chine
Lieu de livraison : Monde entier Afficher les exclusions

Délai de livraison : Estimé entre le **mer. 24 avr. et le mer. 5 juin**

Paiements : PayPal CB Maestro VISA MasterCard

CMOS, 125 MHz Complete DDS Synthesizer

AD9850

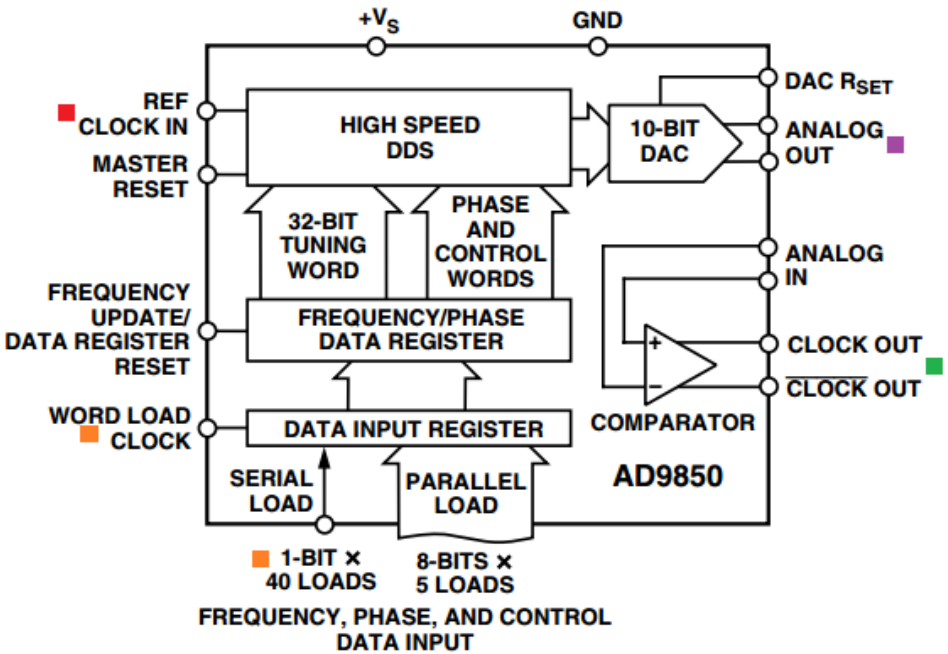


NC = NO CONNECT

FEATURES

- 125 MHz Clock Rate
- On-Chip High Performance DAC and High Speed Comparator
- DAC SFDR > 50 dB @ 40 MHz A_{OUT}
- 32-Bit Frequency Tuning Word
- Simplified Control Interface: Parallel Byte or Serial Loading Format
- Phase Modulation Capability
- 3.3 V or 5 V Single-Supply Operation
- Low Power: 380 mW @ 125 MHz (5 V)
  155 mW @ 110 MHz (3.3 V)
- Power-Down Function
- Ultrasmall 28-Lead SSOP Packaging

FUNCTIONAL BLOCK DIAGRAM



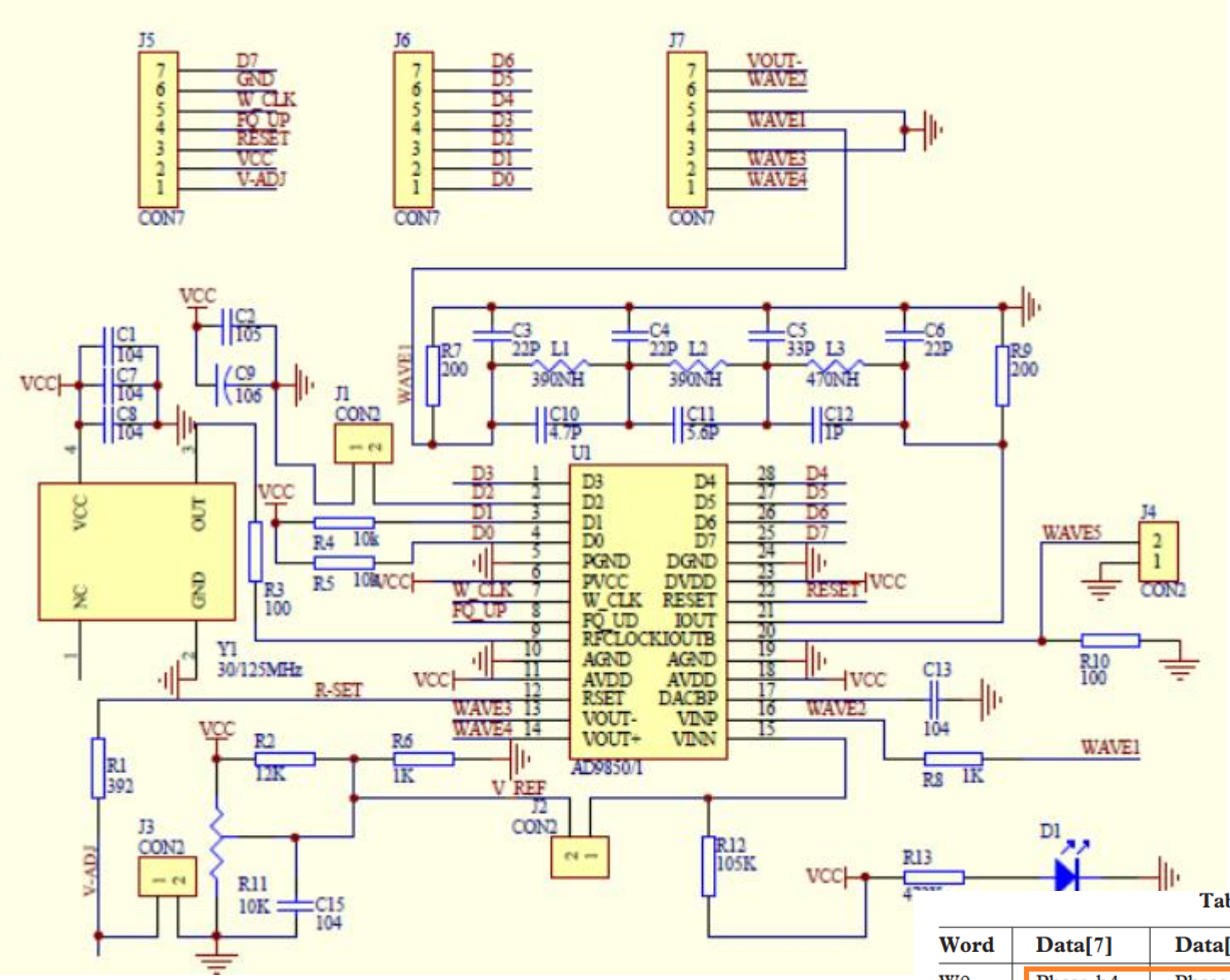


Table III. 8-Bit Parallel Load Data/Control Word Functional Assignment

Word	Data[7]	Data[6]	Data[5]	Data[4]	Data[3]	Data[2]	Data[1]	Data[0]
W0	Phase-b4 (MSB)	Phase-b3	Phase-b2	Phase-b1	Phase-b0 (LSB)	Power-Down	Control	Control
W1	Freq-b31 (MSB)	Freq-b30	Freq-b29	Freq-b28	Freq-b27	Freq-b26	Freq-b25	Freq-b24
W2	Freq-b23	Freq-b22	Freq-b21	Freq-b20	Freq-b19	Freq-b18	Freq-b17	Freq-b16
W3	Freq-b15	Freq-b14	Freq-b13	Freq-b12	Freq-b11	Freq-b10	Freq-b9	Freq-b8
W4	Freq-b7	Freq-b6	Freq-b5	Freq-b4	Freq-b3	Freq-b2	Freq-b1	Freq-b0 (LSB)

$$f_{OUT} = (\Delta\ Phase \times CLKIN)/2^{32}$$

Fréquence de sortie **mot de 32 bits** **fréquence de référence(fixe)**

125mhz

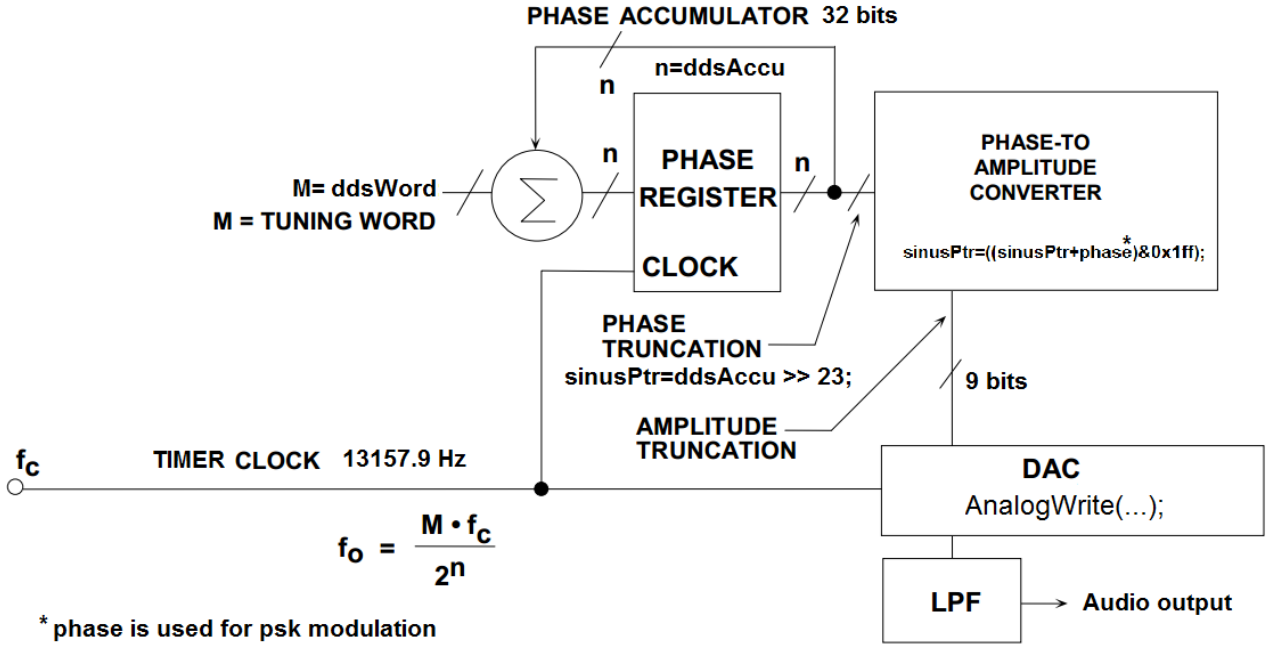
$$\Delta\ Phase = \frac{2^{32} \times f_{OUT}}{CLKIN}$$

```
void setfreq(double f, uint16_t p) {
  uint32_t deltaphase;

  deltaphase = f * 4294967296.0 / (125000000 + factor);
  for (int i = 0; i < 4; i++, deltaphase >>= 8) {
    SPI.transfer(deltaphase & 0xFF);
  }
  SPI.transfer((p << 3) & 0xFF) ;
  pulse(FQ_UD);
}
```

Pour en savoir plus sur le DDS

http://kudelsko.free.fr/generateur_fonctions/principe_dds.htm
https://www.arrow.com/fr-fr/research-and-events/videos/what-is-direct-digital-synthesis
http://www.johnloomis.org/digitallab/audio/audio3/tut_dds.pdf
Standalone HAM modulation generator (DDS : Structure interne) https://www.tapr.org/
https://github.com/f4goh/WSPR



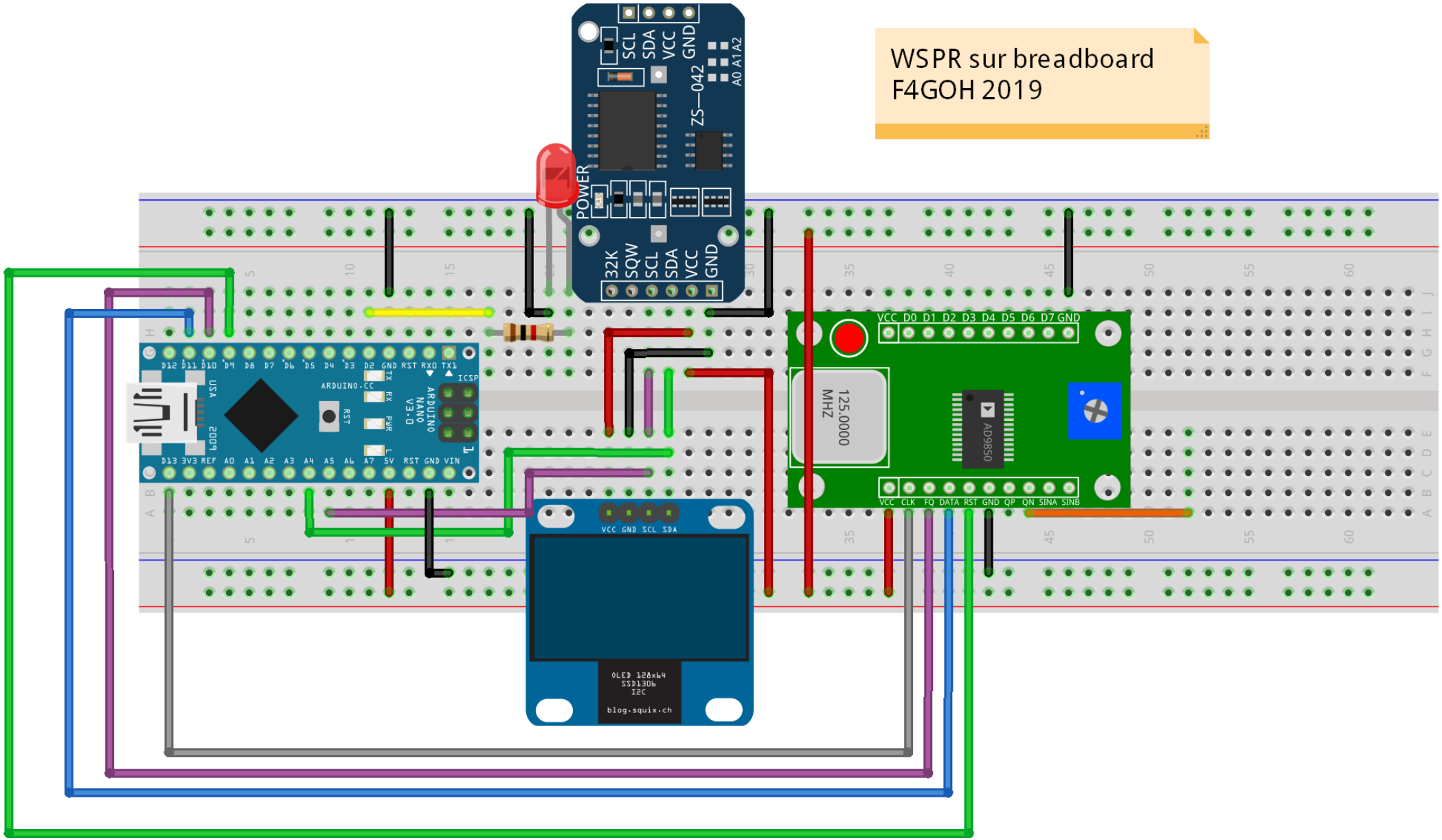
refclk = sample frequency
freq : desired frequency

$$\text{DDS word} = \frac{2^{32} \times \text{freq}}{\text{refclk}}$$

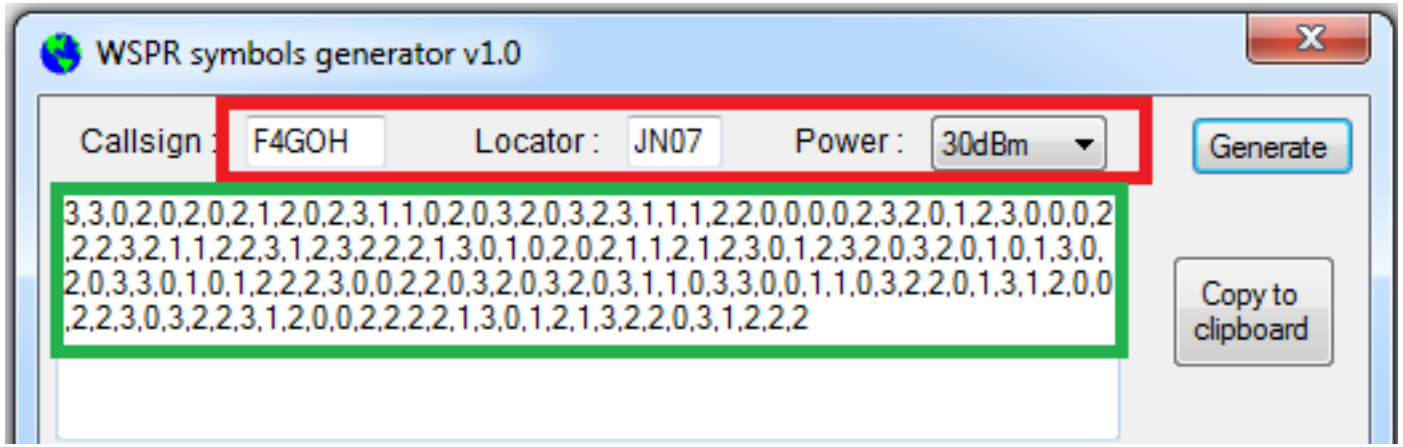
exemple : freq=1500

$$\text{ddsReg[0]} = \frac{2^{32} \times 1500}{13157.9}$$

ddsReg[0]= 489626075,89


WSPR sur breadboard
F4GOH 2019
SCL SDA VCC GND
ZS-042
A0 A1A2
POWER
32K SQW SCL SDA VCC GND
VCC D0 D1 D2 D3 D4 D5 D6 D7 GND
125.0000 MHZ
AD9850
VCC CLK FQ DATA RST GND QP QN SINA SINB
D12 D11 D10 D9 D8 D7 D6 D5 D4 D3 D2 GND RST RX0 TX1
ARDUINO.CC
ARDUINO NANO V3.0
ICSP
D13 3V3 REF A0 A1 A2 A3 A4 A5 A6 A7 5V RST GND VIN
VCC GND SCL SDA
0LED 128x64
SSD1306
I2C
blog.squix.ch
fritzing

WSPR : L'encodeur



Resulting in 162 sequential symbols each with a value from 0 to 3

Modulation

Each symbol represents a frequency shift of 12000 / 8192, or approximately 1.46Hz, per Symbol Value giving four-level Multi-FSK modulation. The transmitted symbol length is the reciprocal of the tone spacing, or approximately 0.683 seconds, so the complete message of 162 symbols takes around 110.6 seconds to send and occupies a bandwidth of approximately 6Hz,

```
void sendWspr(long freqWspr) {

  int a = 0;
  for (int element = 0; element < 162; element++) {    // For each element in the message
    a = int(wsprSymb[element]); //   get the numerical ASCII Code
    setfreq((double) freqWspr + (double) a * 1.4548, 0);
    delay(682);
    Serial.print(a);
    digitalWrite(LED, digitalRead(LED) ^1);
  }
  setfreq(0, 0);
  Serial.println("EOT");
}
```

162 x 0,682 = 110,484 soit 1 minute et 50 secondes

WSPR, la théorie

Andy Talbot :
http://www.g4jnt.com/wspr_coding_process.pdf

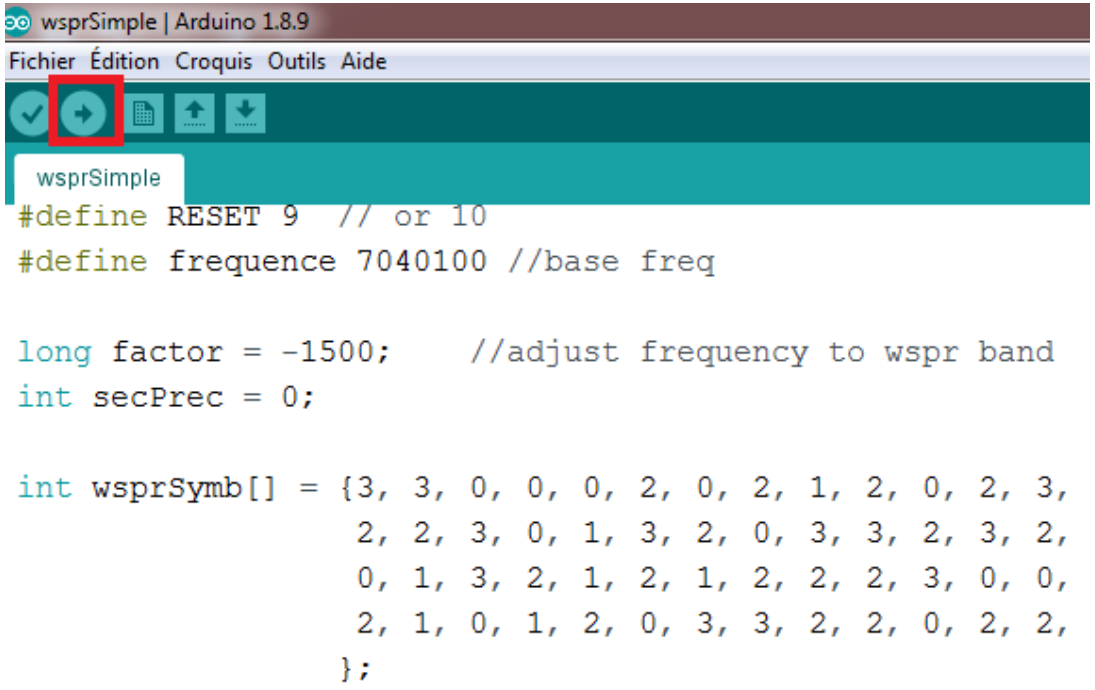
The parity generation process is :

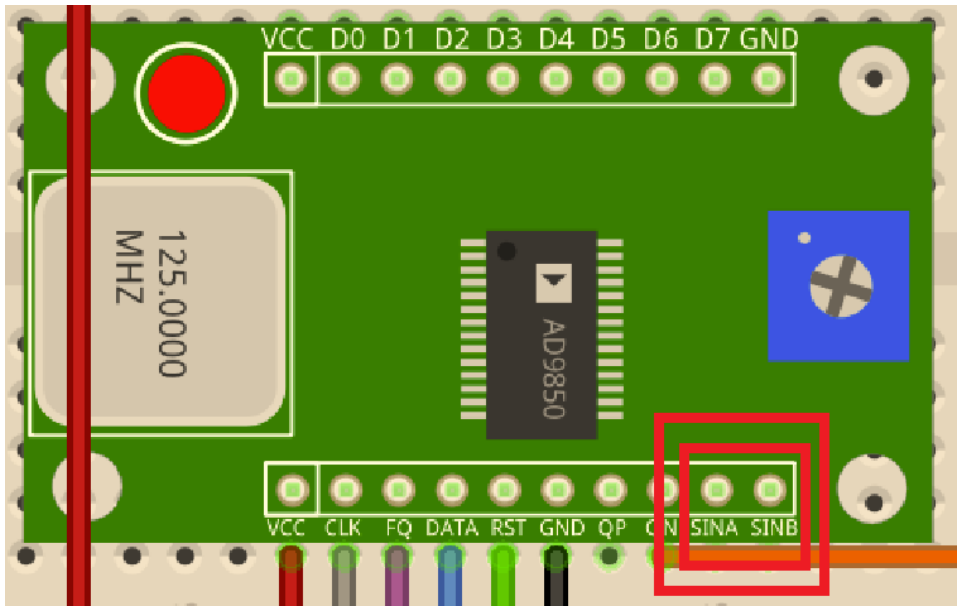
Shift the next source bit into the LSB of both [Reg 0] and [Reg 1],
moving the existing data in each one place left
Take the contents of [Reg 0]
AND with 0xF2D05351
Calculate the single bit parity (XOR) of the resulting sum.
Append to the output data stream
Take the contents of [Reg 1]
AND with 0xE4613C47
Calculate the single bit parity (XOR) of the resulting sum.
Append to the output data stream

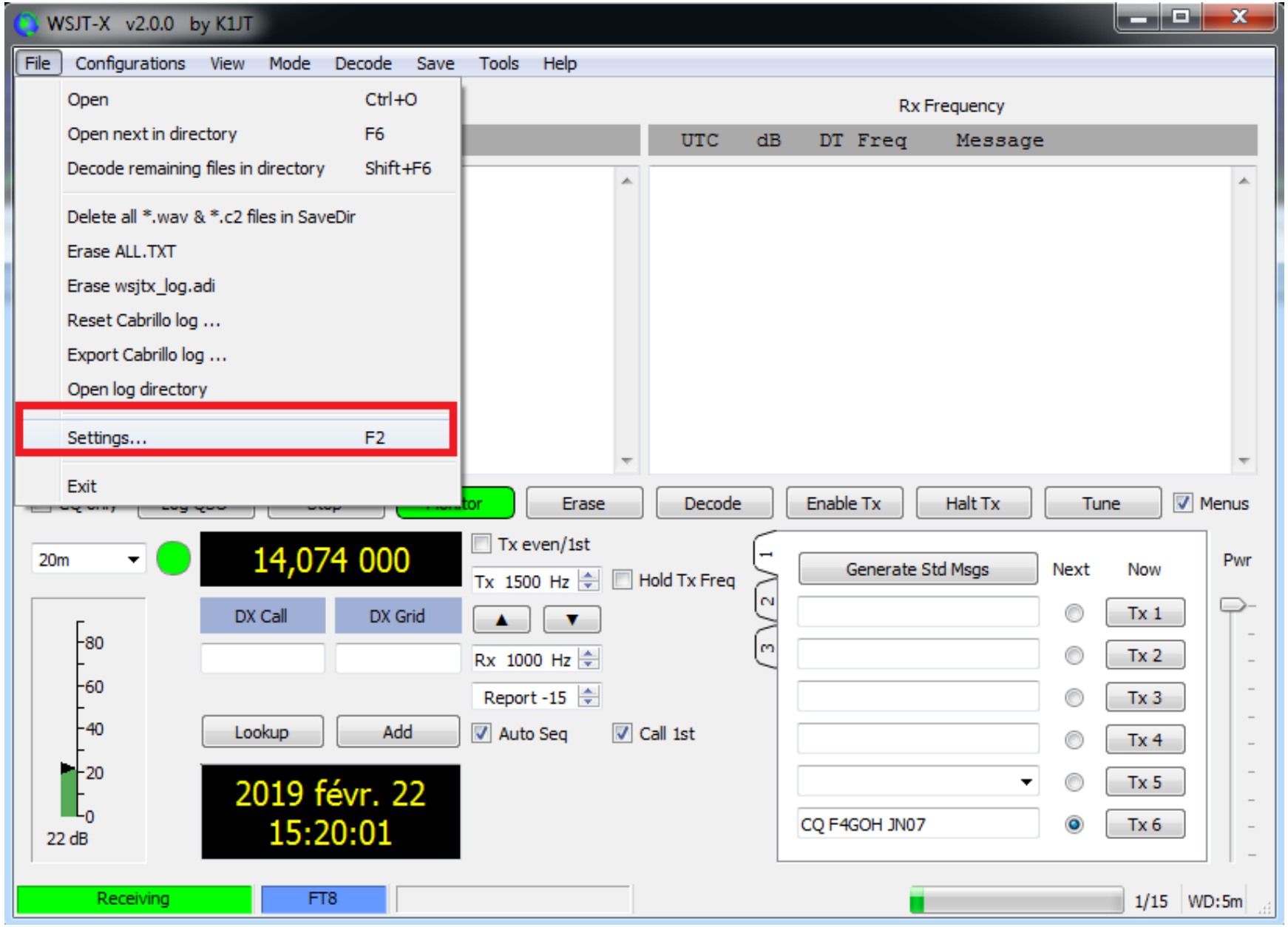
Mise en pratique

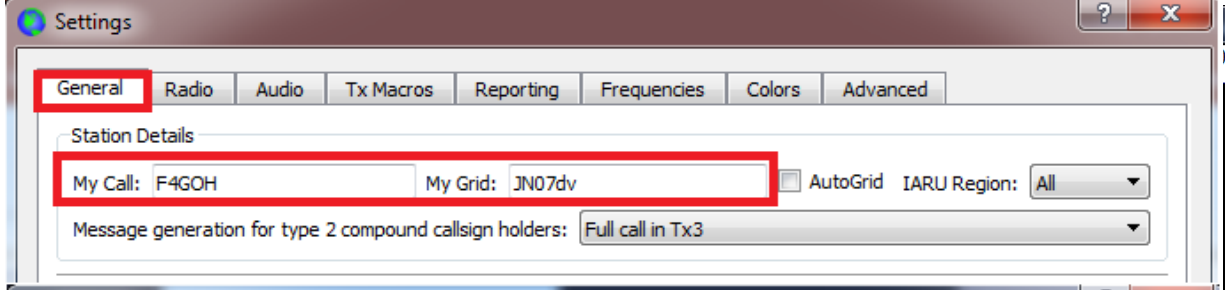
- Générer vos symboles avec WSPR encoder.exe https://github.com/f4goh/WSPR
- Coller les symboles dans le code Arduino wsprSimple.ino
- Téléverser le programme dans l'arduino nano (Attention au port COM)
- Placer un bout de fil en l'air sur la sortie sinus du DDS, SinA ou SinB
- Décoder la trame avec WSJT-X

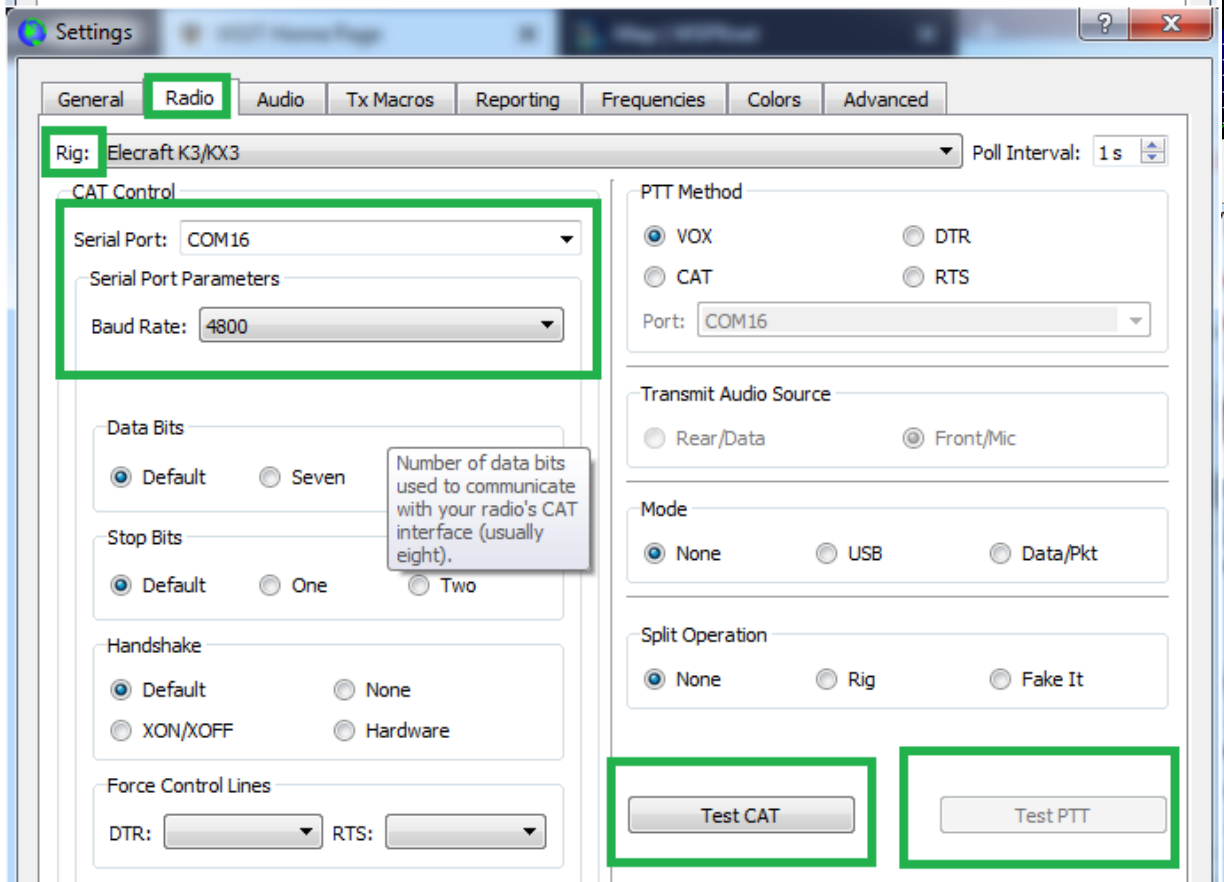
```
int wsprSymb[] = {3, 3, 0, 0, 0, 2, 0, 2, 1, 2, 0, 2, 3, 3, 1, 0, 2, 0, 3, 0, 0, 1, 2, 1, 1, 3, 1, 0, 2, 0, 0, 2, 0, 2, 3, 2, 0, 1, 2, 3, 0, 0, 0, 0,
                  2, 2, 3, 0, 1, 3, 2, 0, 3, 3, 2, 3, 2, 2, 2, 1, 3, 0, 1, 0, 2, 0, 2, 1, 1, 2, 1, 0, 3, 2, 1, 2, 3, 0, 0, 1, 2, 0, 1, 0, 1, 3, 0, 0,
                  0, 1, 3, 2, 1, 2, 1, 2, 2, 2, 3, 0, 0, 2, 2, 2, 3, 2, 0, 1, 2, 0, 3, 3, 1, 2, 3, 3, 0, 2, 1, 3, 0, 3, 2, 2, 0, 3, 3, 1, 2, 0, 0, 0,
                  2, 1, 0, 1, 2, 0, 3, 3, 2, 2, 0, 2, 2, 2, 2, 1, 3, 2, 1, 0, 1, 1, 2, 0, 0, 3, 1, 2, 2, 2
                 };
```

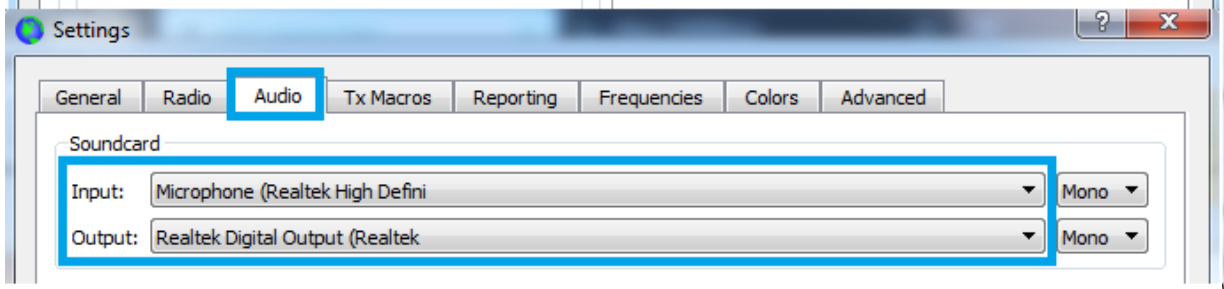


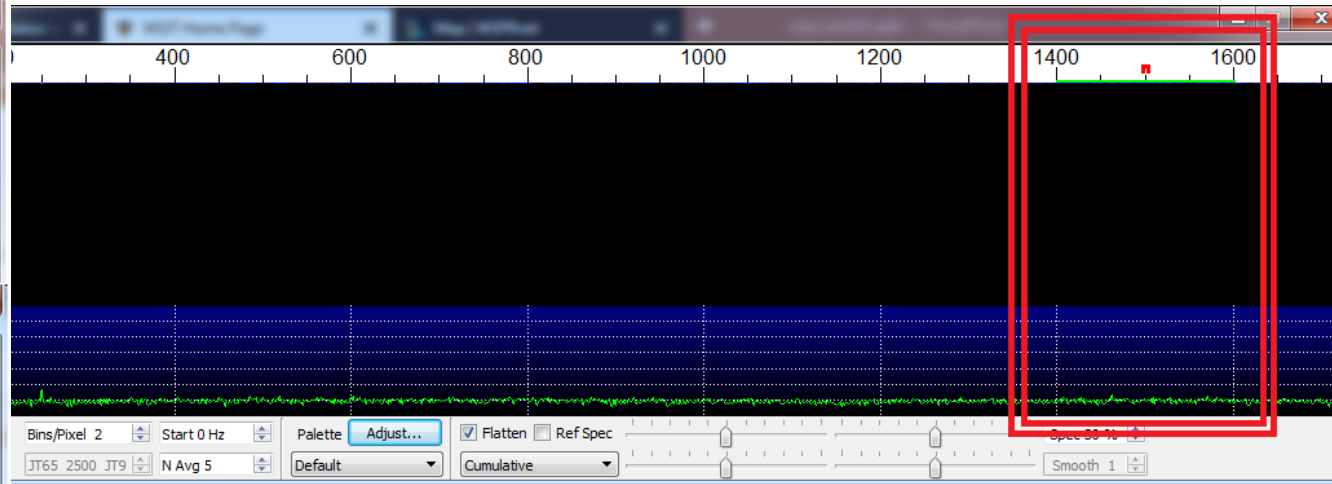


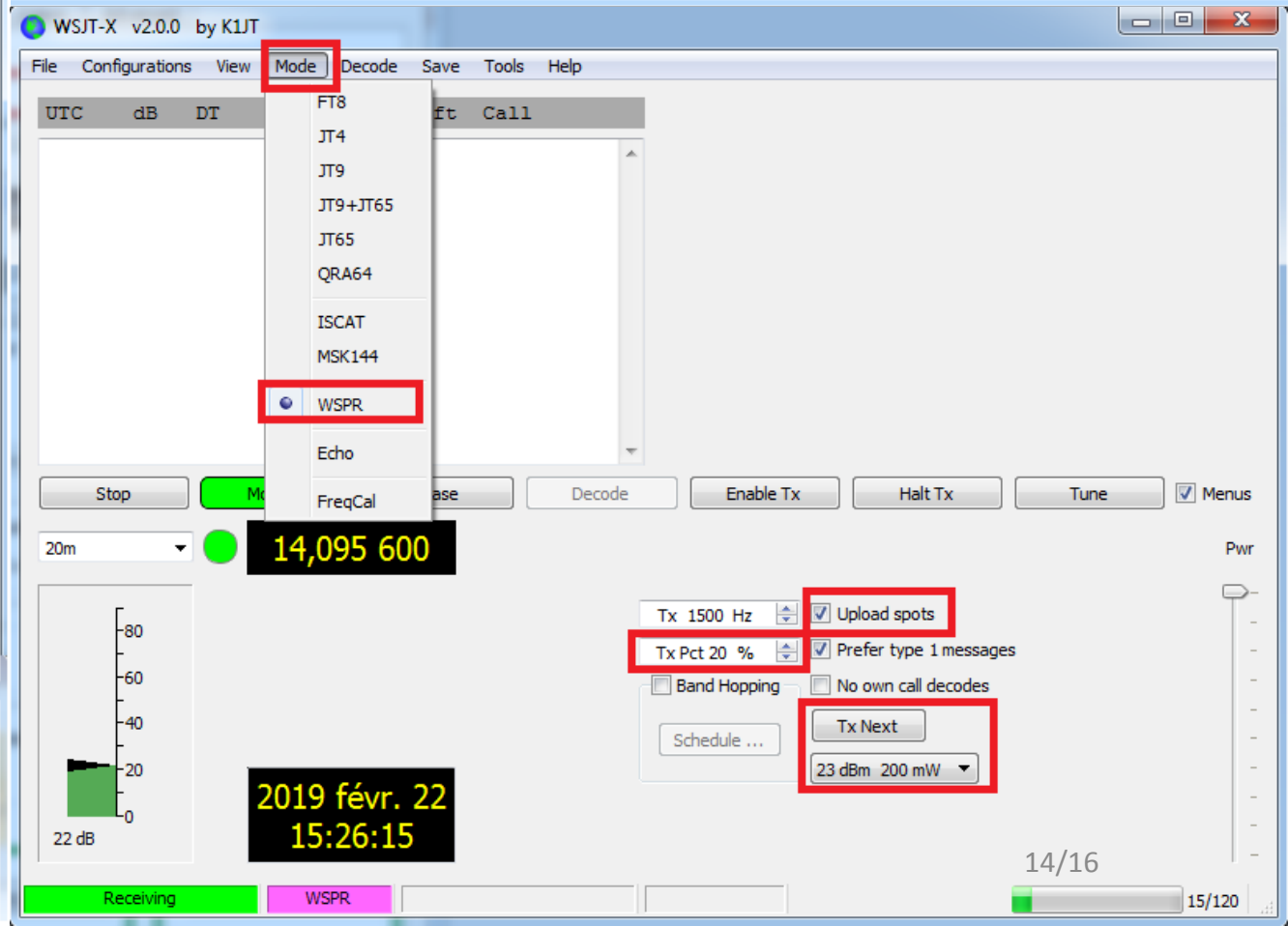

WSJT-X v2.0.0 by K1JT
File
Configurations
View
Mode
Decode
Save
Tools
Help
Open Ctrl+O
Open next in directory F6
Decode remaining files in directory Shift+F6
Delete all *.wav & *.c2 files in SaveDir
Erase ALL.TXT
Erase wsjtx_log.adi
Reset Cabrillo log ...
Export Cabrillo log ...
Open log directory
Settings... F2
Exit
Rx Frequency
UTC dB DT Freq Message
Erase
Decode
Enable Tx
Halt Tx
Tune
Menus
20m
14,074 000
Tx even/1st
Tx 1500 Hz
Hold Tx Freq
DX Call
DX Grid
Rx 1000 Hz
Report -15
Lookup
Add
Auto Seq
Call 1st
2019 févr. 22
15:20:01
80
60
40
20
0
22 dB
Generate Std Msgs
Next
Now
Tx 1
Tx 2
Tx 3
Tx 4
Tx 5
Tx 6
CQ F4GOH JN07
Pwr
Receiving
FT8
1/15
WD:5m


Settings
General
Radio
Audio
Tx Macros
Reporting
Frequencies
Colors
Advanced
Station Details
My Call: F4GOH
My Grid: JN07dv
AutoGrid
IARU Region: All
Message generation for type 2 compound callsign holders: Full call in Tx3


Settings
General
Radio
Audio
Tx Macros
Reporting
Frequencies
Colors
Advanced
Rig: Elecraft K3/KX3
Poll Interval: 1 s
CAT Control
Serial Port: COM16
Serial Port Parameters
Baud Rate: 4800
Data Bits
Default
Seven
Number of data bits used to communicate with your radio's CAT interface (usually eight).
Stop Bits
Default
One
Two
Handshake
Default
None
XON/XOFF
Hardware
Force Control Lines
DTR:
RTS:
PTT Method
VOX
DTR
CAT
RTS
Port: COM16
Transmit Audio Source
Rear/Data
Front/Mic
Mode
None
USB
Data/Pkt
Split Operation
None
Rig
Fake It
Test CAT
Test PTT

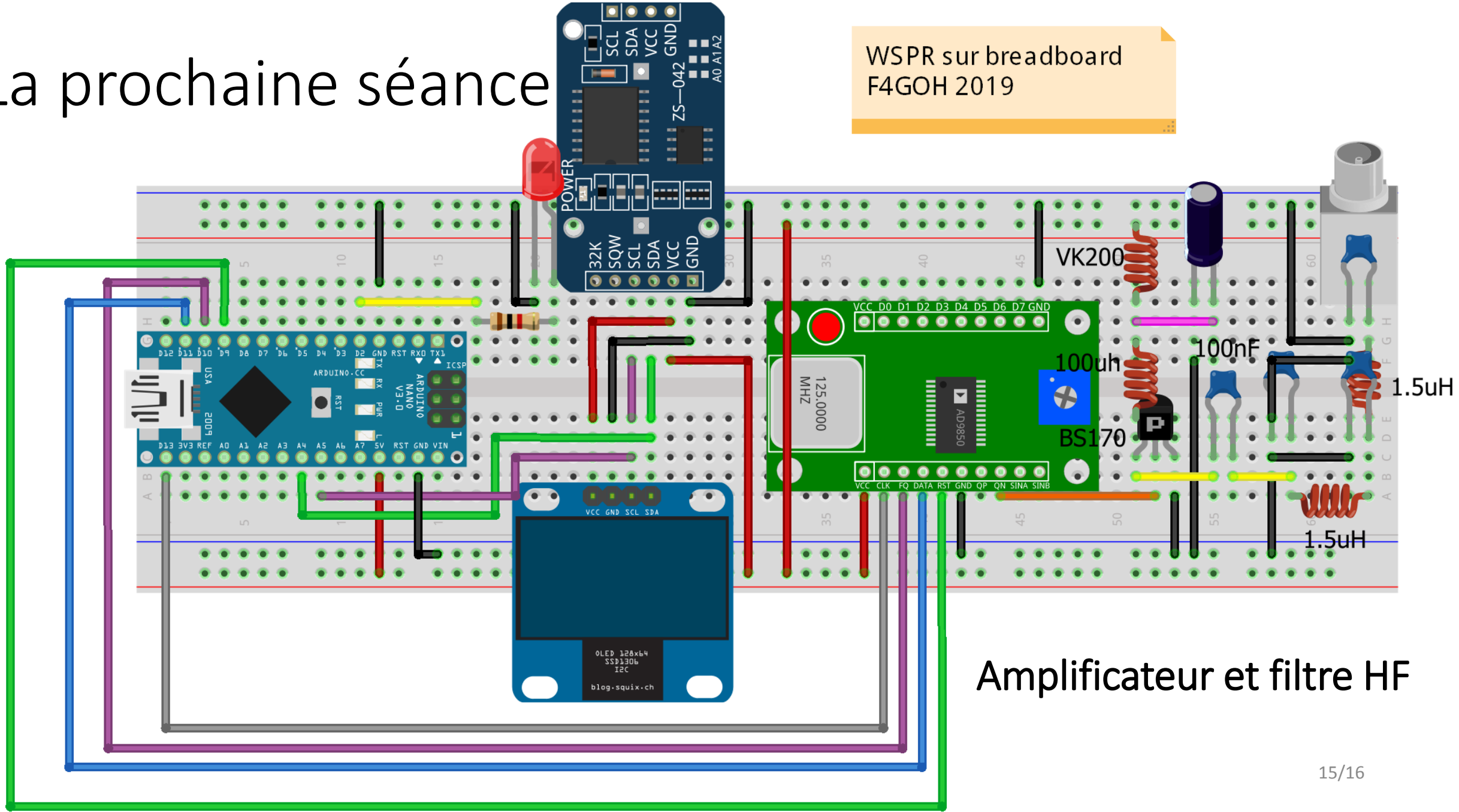

Settings
General
Radio
Audio
Tx Macros
Reporting
Frequencies
Colors
Advanced
Soundcard
Input: Microphone (Realtek High Defini
Mono
Output: Realtek Digital Output (Realtek
Mono


400
600
800
1000
1200
1400
1600
Bins/Pixel 2
Start 0 Hz
Palette
Adjust...
Flatten
Ref Spec
JT65 2500 JT9
N Avg 5
Default
Cumulative
Smooth 1

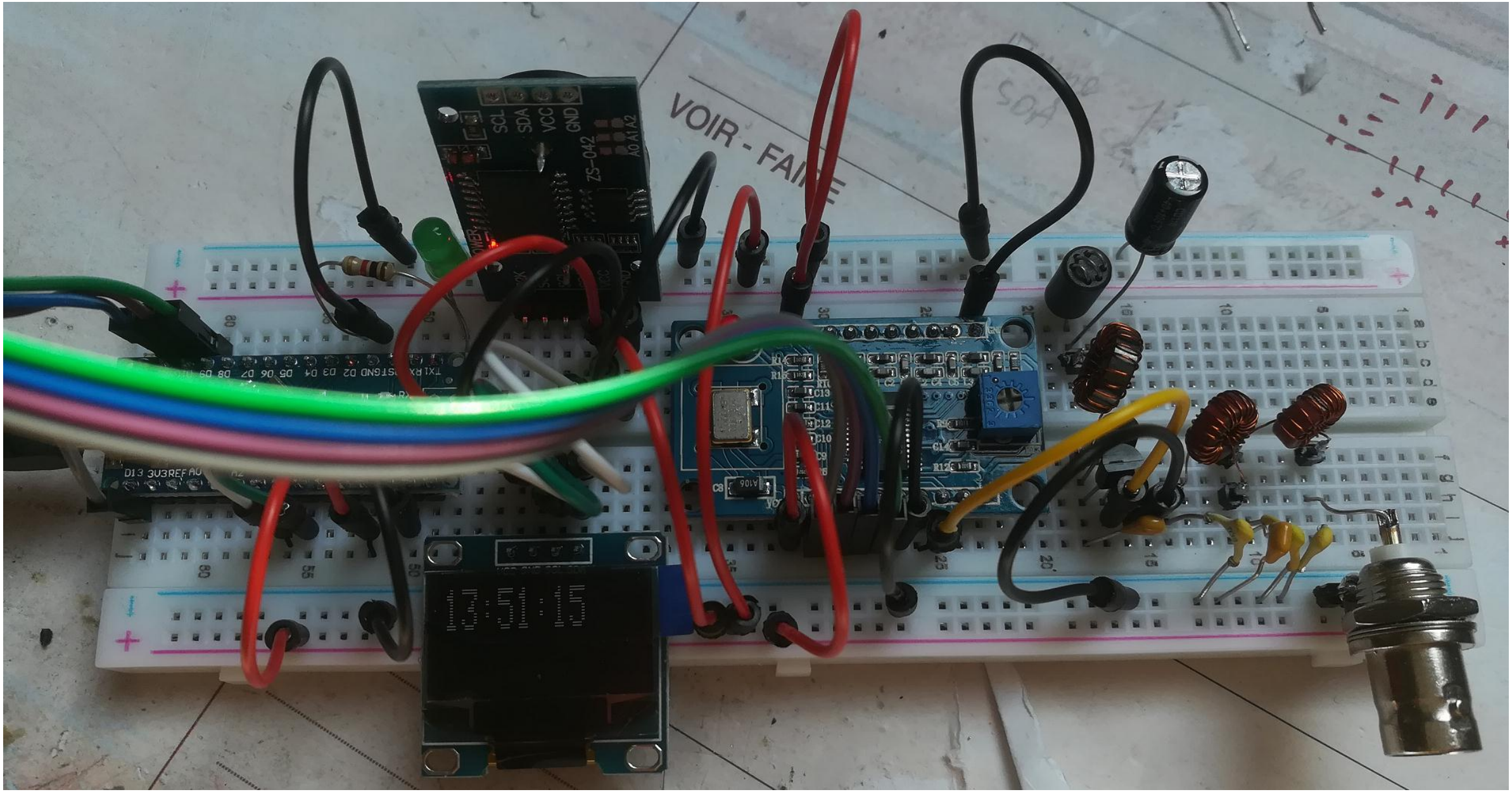

WSJT-X v2.0.0 by K1JT
File
Configurations
View
Mode
Decode
Save
Tools
Help
UTC dB DT ft Call
FT8
JT4
JT9
JT9+JT65
JT65
QRA64
ISCAT
MSK144
WSPR
Echo
FreqCal
Stop
Decode
Enable Tx
Halt Tx
Tune
Menus
20m
14,095 600
Pwr
Tx 1500 Hz
Upload spots
Tx Pct 20 %
Prefer type 1 messages
Band Hopping
No own call decodes
Schedule ...
Tx Next
23 dBm 200 mW
22 dB
2019 févr. 22
15:26:15
Receiving
WSPR
15/120

La prochaine séance

WSPR sur breadboard
F4GOH 2019



Amplificateur et filtre HF


SCL
SDA
VCC
GND
ZS-042
VOIR - FAIRE