Standalone HAM modulation generator

Anthony LE CREN, F4GOH
47 rue de l'avenir
72220 TELOCHE
FRANCE
f4goh@orange.fr

Abstract

I have always been interested in radio modulations like RTTY, PSK, WSPR etc…and especially to know how it's coded. the asset of these modulations is understanding the DDS (Direct Digital Synthesis) operating principle. Then, you will be able to generate any modulation.

Introduction

The idea is to generate RTTY, Hellschreiber, WSPR, CW, PSK and QPSK 31,63,125 without PC. These audio frequency modulations are delivred by an Arduino. The TRX are controlled by CIV system and an DS3231 RTC synchronize time to transmit WSPR every even minutes. The cost is about 20$. I use a Arduino Nano, but it can be work with an Arduino Uno.

Conclusion

It was a pleasure to study DDS and modulation generation. This project was designed essentially for makers. Adapt it as you want. If you have any questions, send me an email.

Reference

https://hamprojects.wordpress.com/

https://github.com/f4goh/MODULATION

http://www.analog.com/media/en/training-seminars/tutorials/MT-085.pdf

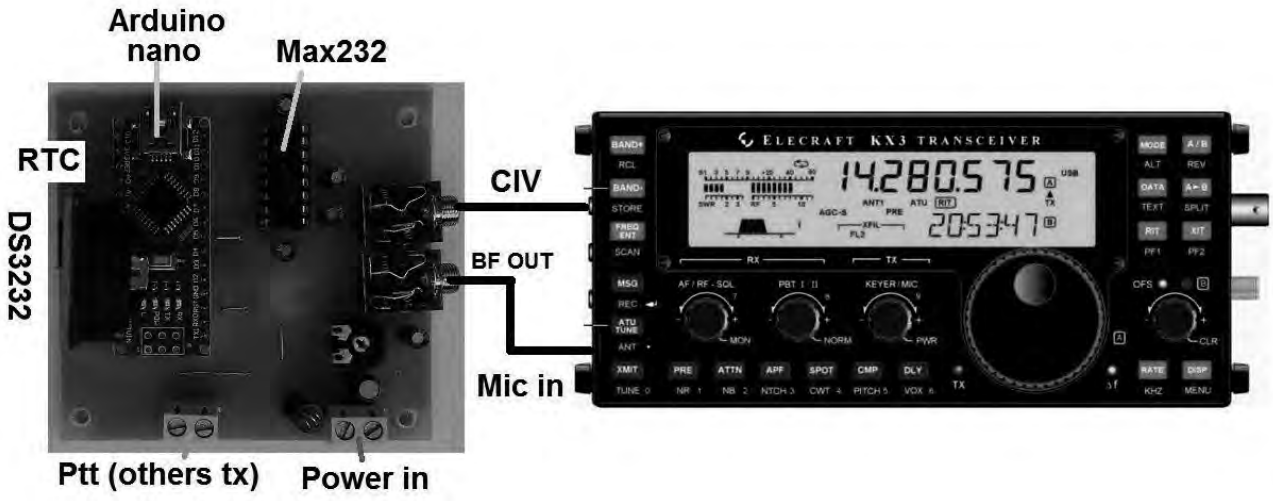
http://www.arrl.org/psk31-spec

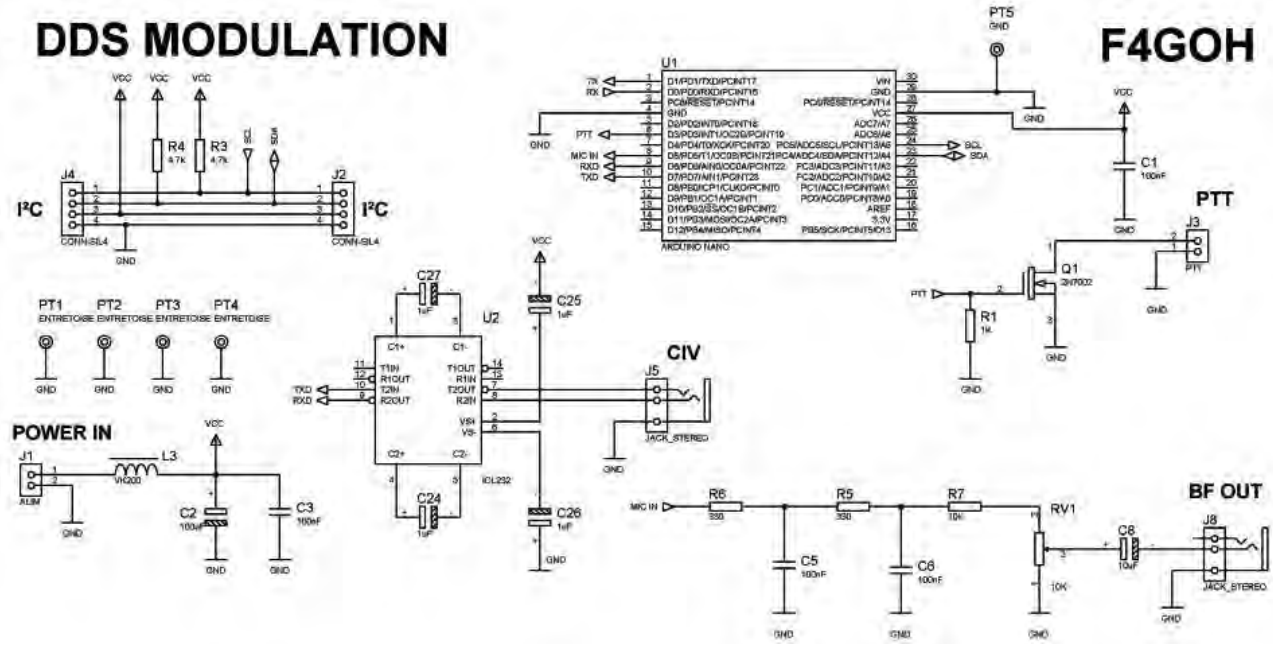
http://www.g4jnt.com/wspr_coding_process.pdf

https://brainwagon.org/2012/01/11/hellduino-sending-hellschreiber-from-an-arduino/

http://www.johnloomis.org/digitallab/audio/audio3/tut_dds.pdf

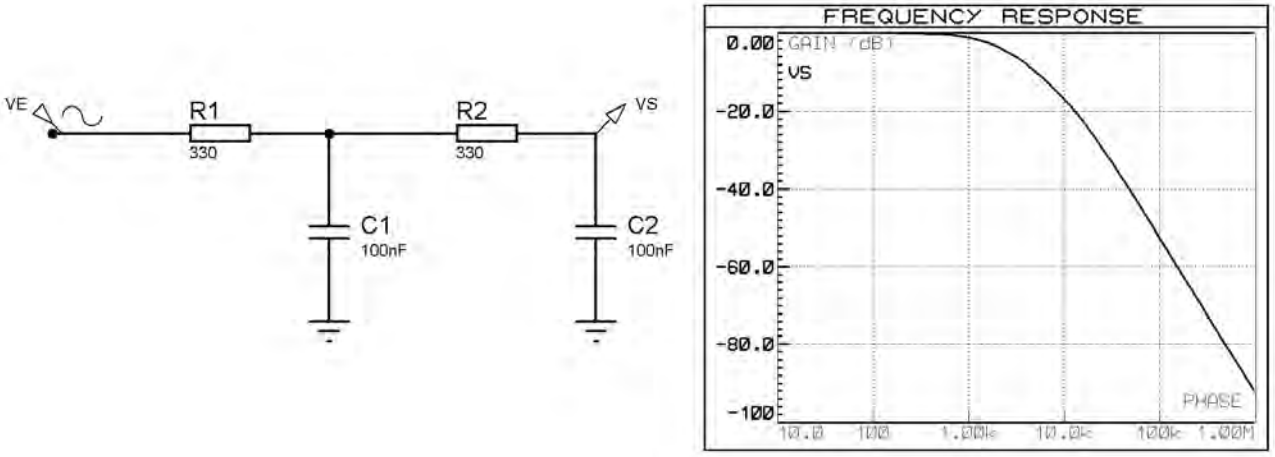
Main board description





The DS3231 is used to synchronize time transmission(J4). Arduino Nano generate Audio frequency around 1500Hz (J3). RV1 adjust Audio level Output. CIV system is a software serial via Max232. That's control VFO and PTT (J5). There is also another PTT output (2N2007 in open Drain J3). Power supply is 5 Volts, there is no DC/DC converter.

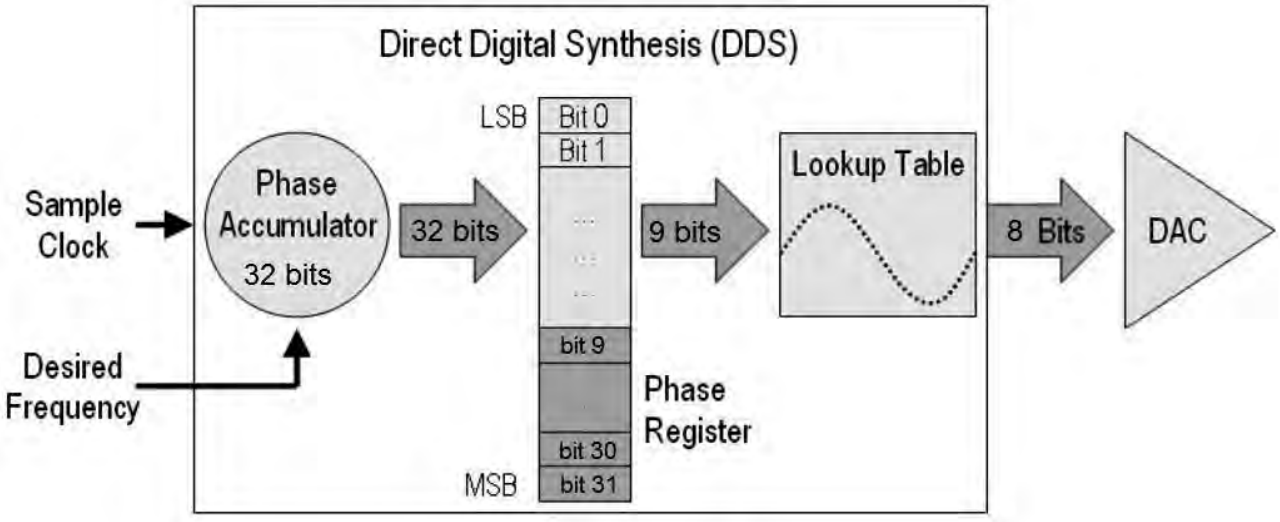
The audio sinus frequency is generated by Arduino PWM, so a low pass filter is required.



Direct digital synthesis

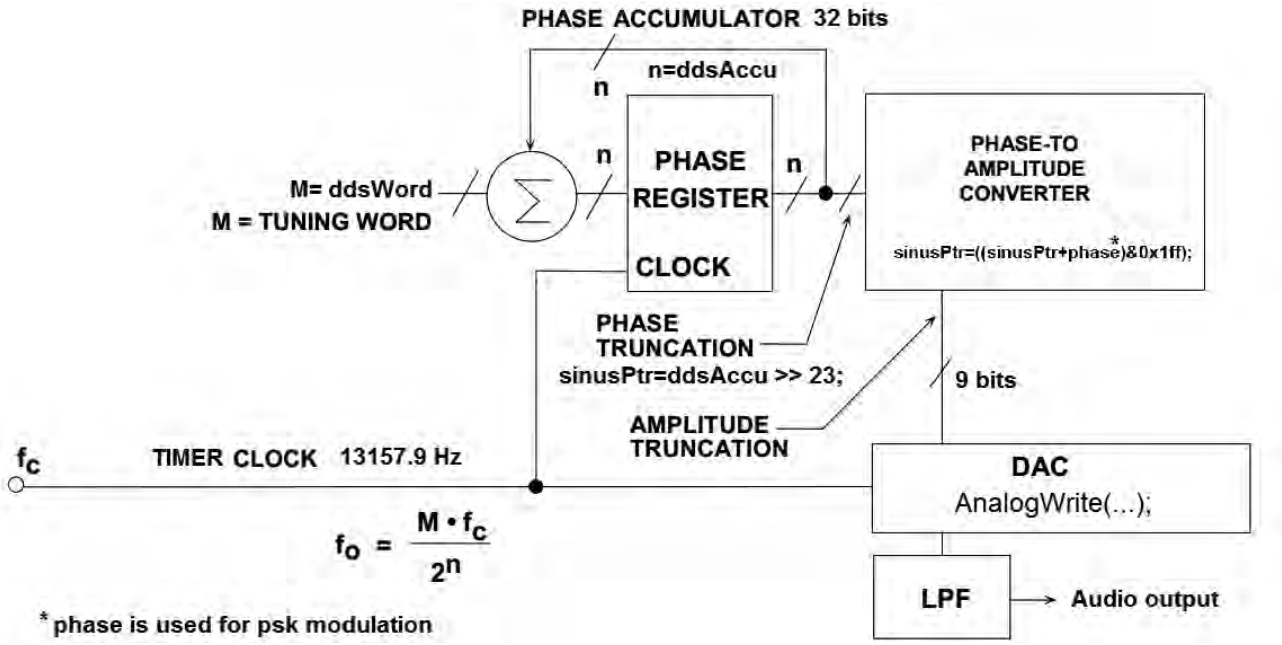
Usually, a sinusoidal table sampled in constant frequency makes it possible to obtain a single frequency. It is not enough, because it is necessary to generate a frequency between 1000 and 1500 hertz with a step of 1.4548 Hertz (WSPR). You can not overload flash memory with as many sinus tables as necessary. So how to generate different frequencies with only one lookup table?

The method is described by Analog Devices or National Instruments.



I just adapt it for an Arduino to generate WSPR, CW, PSK and QPSK 31,63,125 modulations.
I use 32 bits phase accumulator and 9 bits pointer to lookup table. The PWM is used as DAC (Digital Analog Converter)

Let's make the link between the Arduino code and the DDS



I change parameters to use it with 512 values PWM DAC. Sinus table is center to 128 (0, 255 range)

First, compute DDS word as follow:

$$\text{refclk = sample frequency}$$
$$\text{freq : desired frequency}$$
$$\text{DDS word} = \frac{2^{32} \times \text{freq}}{\text{refclk}}$$

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

$$\text{ddsReg[0]} = 489626075{,}89$$

Second, in timer one interrupt (Fc=13157.9 HZ, Tc=76µs)

DDS word is accumulated in ACCU. The result is 23 bits right shifted to retrieve sinus value in the table. Why 23 bits. 32-23=9: there is 512 sinus values, so 2^9=512.

DDS programming

```
void MODULATION::sinus()                    //timer one irq
{

        const static byte sinusTable[512] PROGMEM = {128,129……..};

         ddsAccu=ddsAccu+ddsWord; // soft DDS, phase accu with 32 bits
         sinusPtr=ddsAccu >> 23;
         sinusPtr=((sinusPtr+phase)&0x1ff);          //add phase for psk or qpsk

        analogWrite(bf_pin,pgm_read_byte(&(sinusTable[sinusPtr]))); //DAC
        countPtr++;             //count interrupt for time generation
}

unsigned long MODULATION::computeDdsWord(double freq)
{
        return pow(2,32)*freq/refclk;
}

void MODULATION::send_bit(int tempo)                    //bitrate modulation counter
{
        countPtr=0;
        int countPtrPrec=0;
        while(countPtrPrec<tempo){
                if (countPtrPrec<countPtr) {
                countPtrPrec=countPtr;
        }
}
countPtr=0;
digitalWrite(13,digitalRead(13)^1);                     //scope irq measurement.
}
```

tempo variable is a irq count order. For example if i want to generate a frequency during 0,682 s, the calculation is:
Numbrer irq required = FC*0,682   //13157.9*0.682=8973

Software description

The WSPR, CW, PSK, QPSK 31,63,125 and DDS generator are implemented in MODULATION library on Github.

WSPR_TRX.ino is an experimental program and very short.

Update RTC before begin transmission :
update time: format yy,m,d,h,m,s
example: 2016,6,18,16,32,30,

Program send audio signal every minute according to modulation table

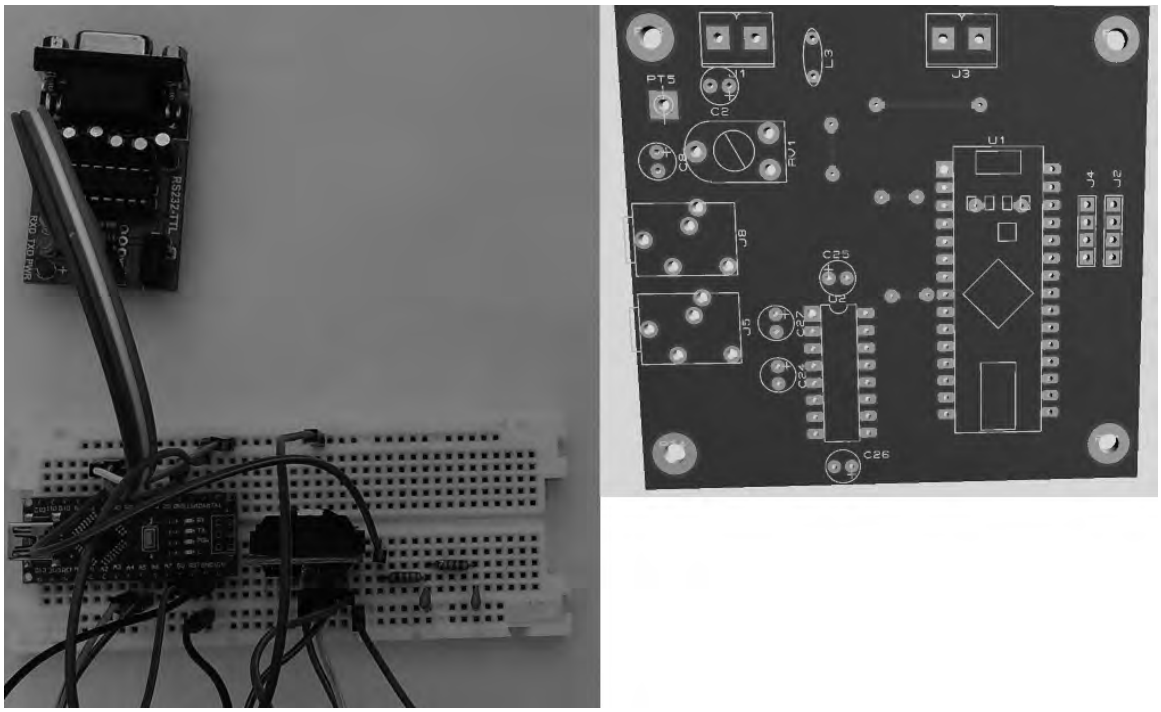
1	RTTY
2	Hellschreiber
3	WSPR
4	CW
5	PSK and QPSK 31,63,125

txing(3); // send WSPR modulation

CIV is used for KX3 TRX. Change sentence according to your TRX.

Realization

There are two ways to test it. Go on Github and made the PCB or use a breadboard.



Results on **wsprnet.org** with a KX3, 1W, 40 meters on a dipole.

Anthony Le Cren F4GOH has been licensed since 2010 and loves to experiment with Arduino applied to the radio. He is a professor of Computer Science at Gabriel Touchard High School in Le Mans, France. Anthony has written numerous articles in different countries (France, GB, USA, Belgium, Italy, Spain, Germany, Poland) and maintains a web page of Amateur Radio Projects at: https://hamprojects.wordpress.com/

You can reach Anthony at: f4goh@orange.fr