

## Experiment „Digitaler Oszillator“

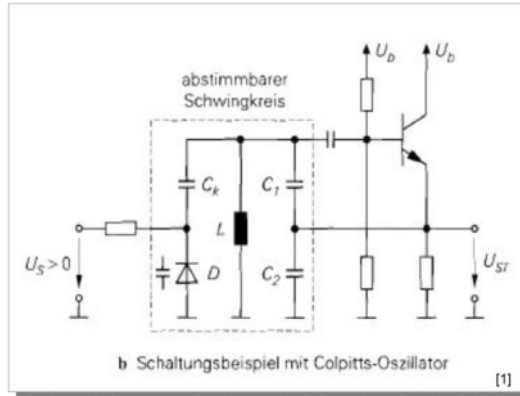
### Kurzfassung.

Es werden zwei Typen von digitalen Oszillatoren vorgestellt:

1. DDS = Digital Direct Synthesis
2. Programmierbarer Clock

An Beispielen werden die Funktion und die Programmierung der speziellen Integrierten Schaltungen gezeigt.

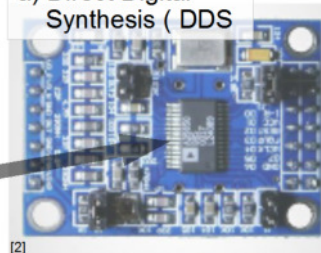
## Analog war gestern! (?)



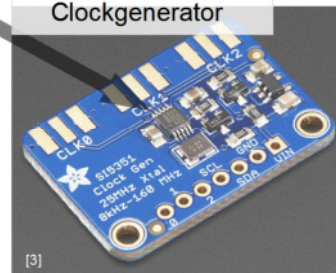
Experiment „Digitaler Oszillator“



a) Direct Digital Synthesis ( DDS



b) Programmierbarer Clockgenerator



1. Okt 2016 S. 2/16

## Einleitung.

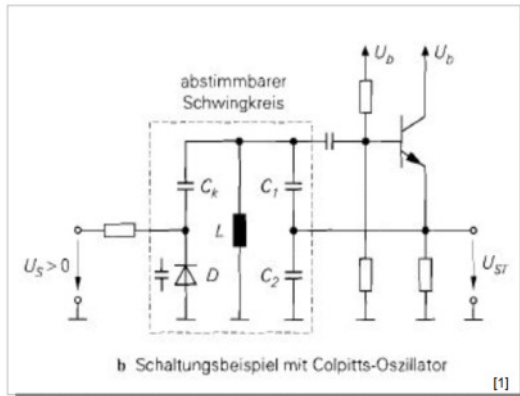
In den letzten 10 Jahren hat sich die Schaltungstechnik der Transceiver verändert, So findet man in vielen TRX statt eines analogen VFOs immer öfter einen digitalen Oszillator.

Hier werden zwei Typen von digitalen Oszillatoren vorgestellt:

- DDS = Digital Direct Synthesis
- Programmierbarer Clock



## Standard VFO



- Bekanntes Design
- Einfach
- Preiswert



- Stabilität
- Frequenzbereich
- Evtl. Drehko

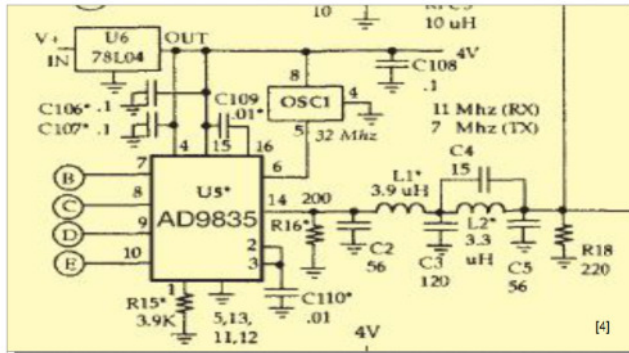
1. Okt 2016 S. 3/16

Der Standard-VFO ist natürlich allen bekannt – auch aus der Amateurfunklizenz-Vorbereitung. Er ist relativ einfach und preiswert.

Er hat jedoch auch Nachteile.

- Frequenz-Stabilität ist eine Herausforderung.  
Mögliche Stabilisierungsmaßnahmen:  
niedrige Frequenz verwenden, DAFC, Huff & Puff, PLL
- Der Frequenzbereich ist eingeschränkt.  
Erweiterung des Frequenzbereichs kann durch Premixer, PLL etc. erreicht werden.
- Evtl. ist ein Drehkondensator schwer zu beschaffen
- Passt nicht in die „Digitale Welt“ ( ? )

## Beispiele DDS

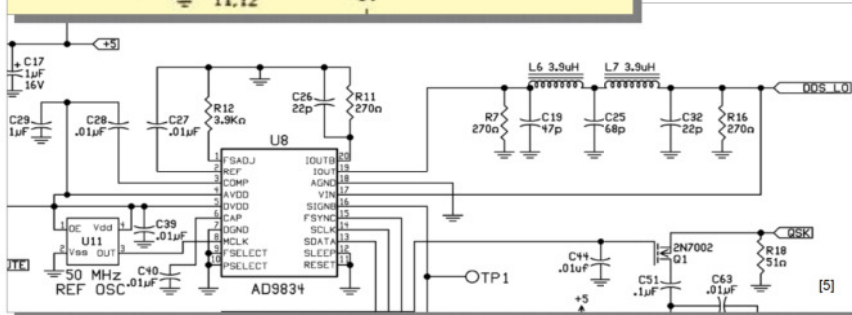


DSW-40



Mountain  
Topper  
MTR-2

Siehe auch :  
• PFR-3

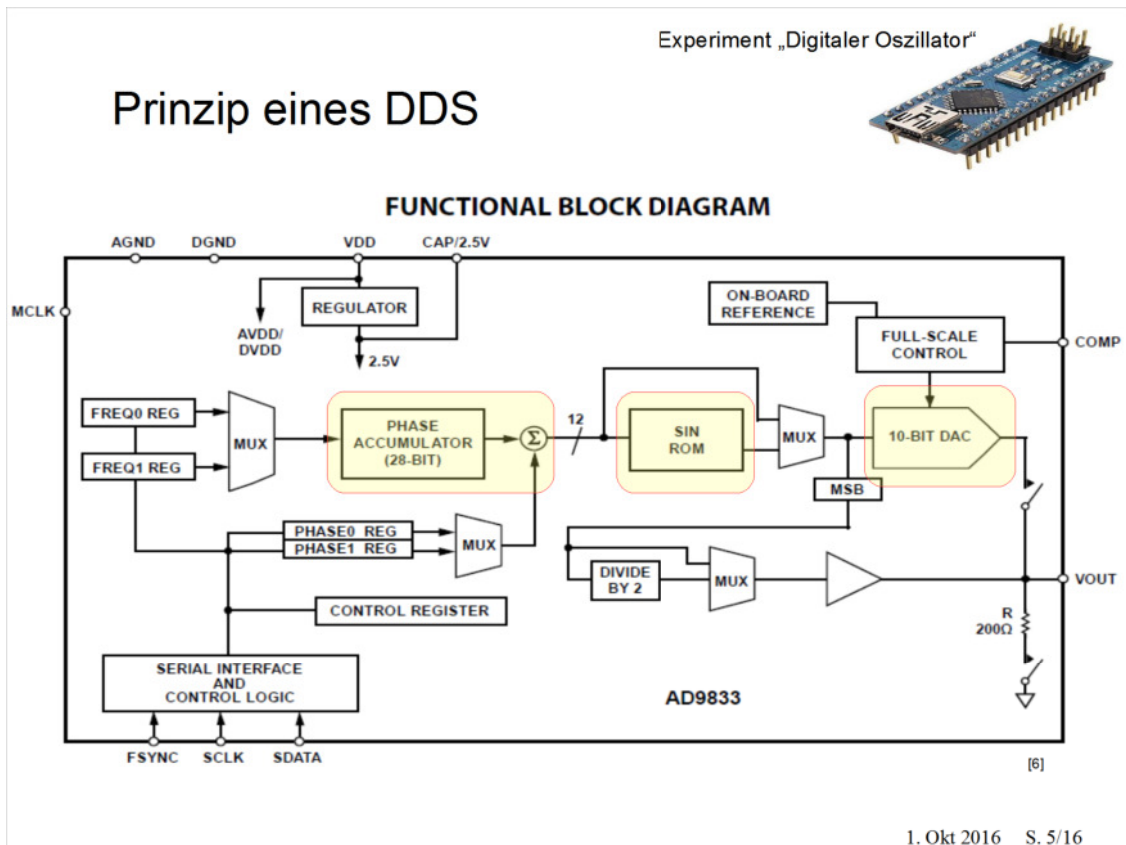


1. Okt 2016 S. 4/16

Im Folgenden werden beispielhaft Geräte aufgeführt, die ein DDS- Chip enthalten:

- DSW-40 ( Small Wonder Labs )
- PFR-3, MTR-2 ( Steve Weber, KD1JV )
- Blue Cool Radio ( DK1HE )
- KX-1 ( Elecraft )
- SM-15 ( DK1HE )

Alle aufgeführten Geräte enthalten die dargestellte oder eine ähnliche Schaltung. Typisch ist der DDS- Chip mit Quarzoszillator und Tiefpassfilter.



Die DDS besteht im wesentlichen aus 3 Blöcken:

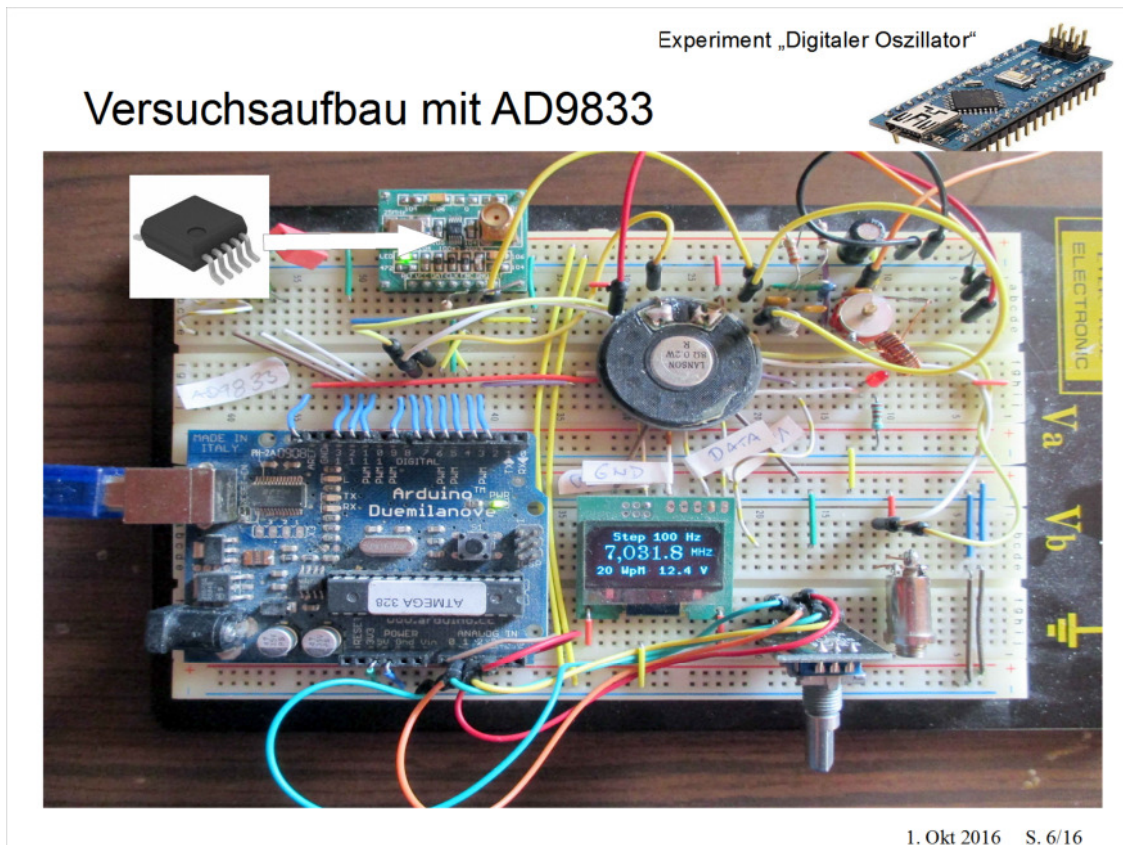
- **Phase-Akkumulator**; hier wird bei jedem Takt ein Stück Phase aufaddiert.
- Aus dem Summenphasenwert wird in der **Sinustabelle** ( SIN-ROM) ein Wert abgelesen.
- Der **Digital-Analog-Wandler** ( DAC) macht daraus einen Analogwert ( Stromquelle ).

In den im folgenden Beispiel wird ein AD9833 verwendet, bei dem der Takt bei 25 MHz liegt. Dies ist eine sehr schlechte Wahl, denn je höher der Takt ist, desto besser ist die Sinusform des Ausgangssignals.

Zum Vergleich:

Der AD9850 hat einen 125 Mhz- Clock- wie z.B. im Antennenanalyser- Projekt von KB6BEZ, welches zum SWT 2015 in modifizierter Form vorgestellt wurde<sup>[12]</sup>.



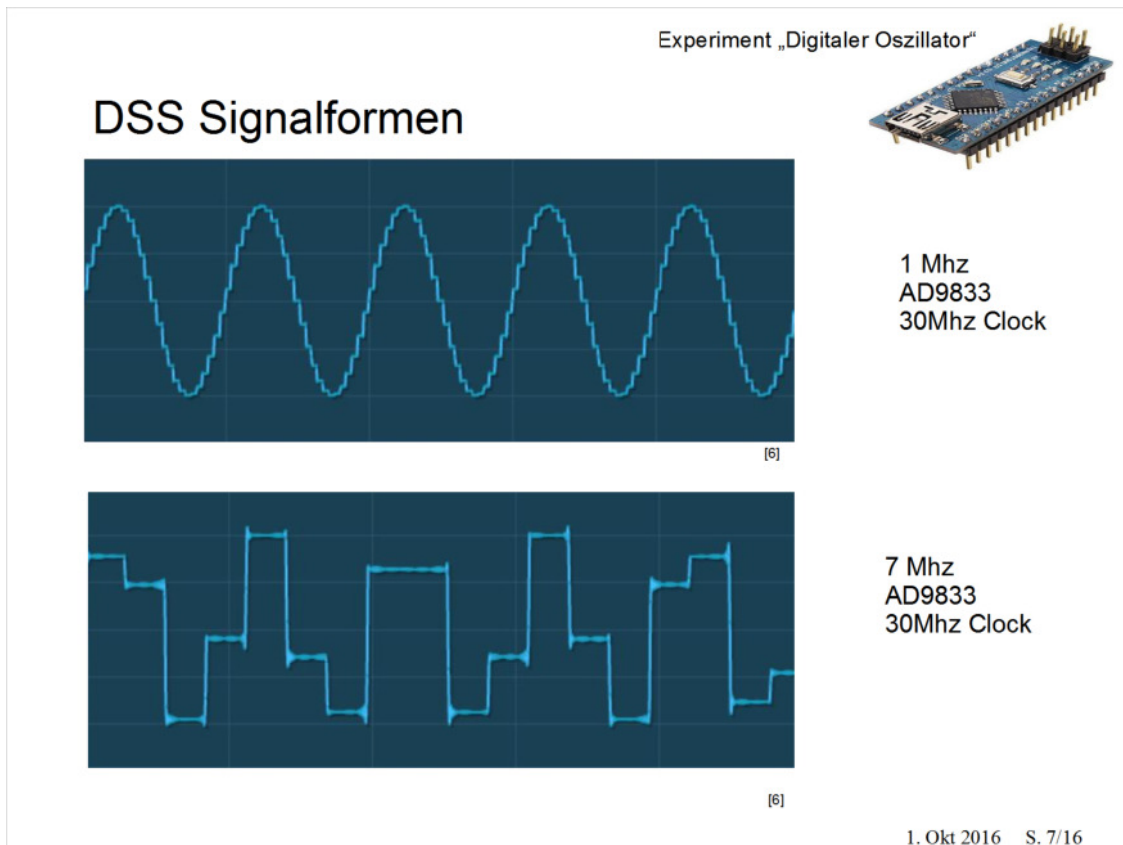


Hier ein Aufbau mit AD9833.

Oben links ist das Modul mit dem DDS- Chip in SMD-Bauform MSOP-10.

Hier ein paar Daten:

- 12.65 mW bei 3 V
- 0 Mhz bis 12.5 MHz
- Typ. 25 MHz Reference Clock
- Sinus, Dreieck, Rechteck
- 2.3 V to 5.5 V Power Supply
- 10-lead MSOP Package



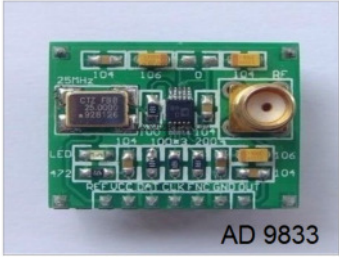
Das Ausgangssignal des DDS ist im Prinzip sinusförmig. Je mehr sich die Ausgangsfrequenz aber der Taktfrequenz nähert, so wird das Signal „eckig“, wie in den Abbildungen gut sichtbar.

Es ist also auf jeden Fall ein Tiefpass am Ausgang erforderlich.


Die Bilder stammen aus der Simulation mit dem Programm 'ADISIMDDS' von Analog Devices [6].

Experiment „Digitaler Oszillator“

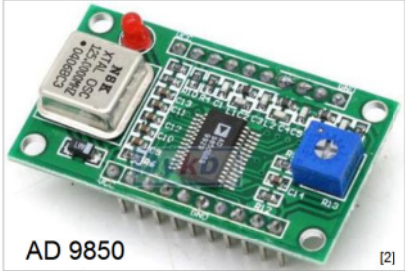
## Fertige DDS- Module



AD 9833 [2]



AD 9850 [2]



AD 9850 [2]

1. Okt 2016 S. 8/16

Es gibt fertige Module zu kaufen:

### **AD9833-Modul**

- 12.65 mW bei 3 V
- 0 Mhz bis 12.5 MHz
- Typ. 25 MHz Reference Clock
- Sinus, Dreieck, Rechteck
- 2.3 V to 5.5 V Power Supply
- 10-lead MSOP Package

### **AD9850-Module ( 2 Varianten)**


- 125 MHz Clock
- On-Chip Comparator ( für Rechtecksignal)
- 32-Bit Frequency Tuning Word
- 3.3 V or 5 V
- Low Power: 380 mW @ 125 MHz (5 V)
- 28-Lead SSOP Packaging

### **AD9851-Modul**

- ähnlich dem mit AD9850, hat jedoch
- Einen 180 MHz Clock ( 30MHz x 6 Multiplier)
- 2.7 V to 5.25 V
- 555 mW @ 180 MHz



Experiment „Digitaler Oszillator“



## Arduino-Programm zu DDS-AD 9833

Stelle Frequenz ein !

Das Protokoll für die serielle Übertragung ( Bit für Bit )

Timing der ser. Übertragung

```

void Ad9833::set_f(float f, int channelCode)
{
    long x = (long) (f * 10.73741824);
    int msb = channelCode | (x & 0x3fff);
    int lsb = channelCode | x >> 14;
    dds_spi.send(msb);
    dds_spi.send(lsb);
}

void SoftSpi16::send(int txd16)
{
    signed int bitNr;
    bool level;
    digitalWrite(PIN_DDS_SELECT, LOW);
    for ( bitNr = 15; bitNr >= 0; bitNr--)
    {
        level=bitRead(txd16, bitNr);
        digitalWrite(PIN_DDS_DATA , level);
        digitalWrite(PIN_DDS_CLOCK, LOW);
        digitalWrite(PIN_DDS_CLOCK, HIGH);
    }
    digitalWrite(PIN_DDS_SELECT, HIGH);
}
        
```

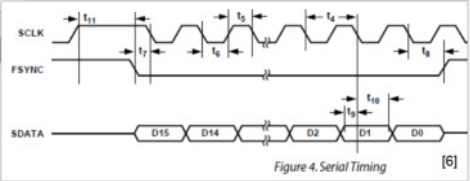


Figure 4. Serial Timing [6]

1. Okt 2016 S. 9/16

Die Programmierung des DDS besteht aus 2 Funktionen:

- Serielle Schnittstelle SPI und
- Frequenzeinstellung.

Die serielle Schnittstelle wird über 3 Port-Pins bedient.

Das Protokoll ist einfach:

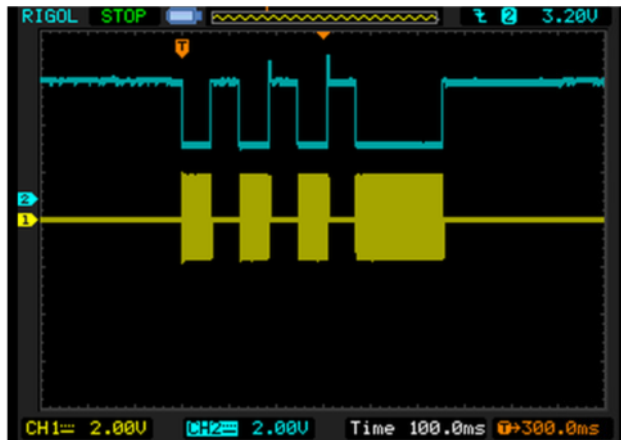
- 1) Chip Select auf Low
- 2) Data auf Pegel von Bit 15
- 3) Clock auf Low
- 4) Clock auf High
- 5) Schritt 2) bis 4) für Bits 14 bis 0 wiederholen
- 6) Chip Select auf High

Die Frequenzeinstellung erfolgt mit der Funktion send (frequenz). Der 28-bit-Frequenzwert wird als High- und Low- Wert an die DDS-Register gesendet.

Die Konstante 10.73741824 ergibt sich aus

$$2^{**}28 / 25 * 10^{**}6 = 10.73741824$$

## DSS Output



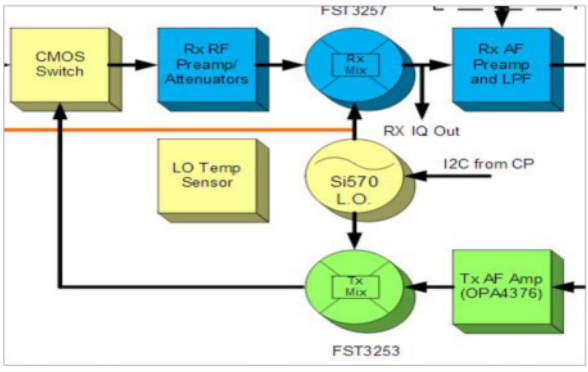
7 Mhz  
AD9833  
Morsezeichen 'v'

1. Okt 2016 S. 10/16

Der DDS AD9833 wird hier vom Softwaremodul „keyer“ direkt getastet durch Einstellung der Frequenz auf 7MHz ( = Signal ein ) bzw. auf Null MHz ( = Signal aus ). Die Verzögerung von ca. 200us am Anfang und am Ende des CW- Signals ist zu vernachlässigen.

Experiment „Digitaler Oszillator“

## Beispiel Programmierbarer Clockgenerator



CMOS Switch

Rx RF Preamp/Attenuators

Rx Mix

Rx AF Preamp and LPF

KX-3

LO Temp Sensor

Si570 L.O.

I2C from CP



Tx Mix

Tx AF Amp (OPA4376)

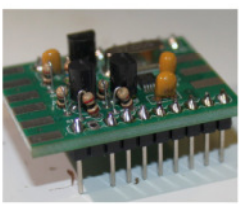
FST3257

FST3253

[7]





[9]



WSPR

[8]



10-MSOP

[9]

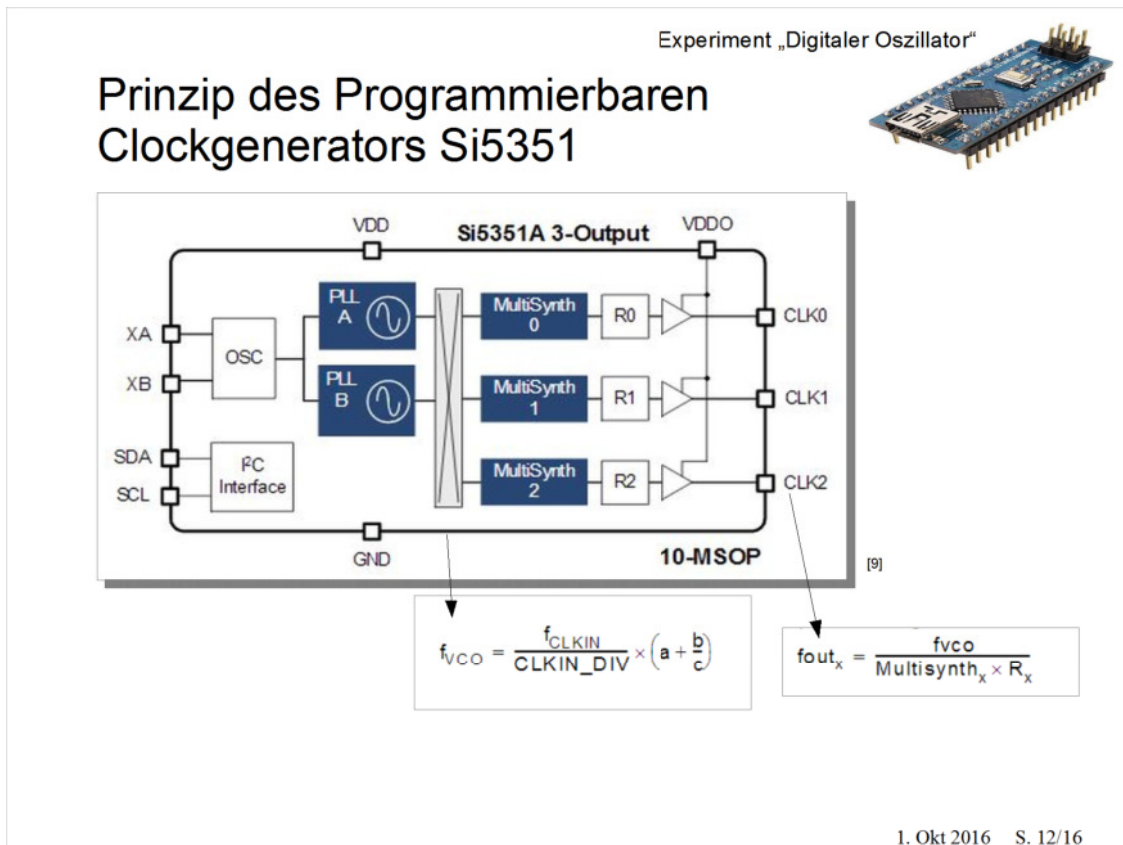
1. Okt 2016 S. 11/16

Eine weitere Gruppe von Integrierten Schaltungen sind die Programmierbaren Clock- Oszillatoren oder PLL- Clock genannt.

Hier einige Anwendungsbeispiele:

- Si570  
KX3, Lima- SDR, Fifi- SDR
- Si5351  
WSPR- Kit, G0UPL , Hans Summers,
- ICS307  
Fernempfangsradio II Harzburg SDR Empfänger

Das Ausgangssignal ist rechteckförmig.



Das Chip Si5351 benötigt einen externen Quarz ( 25 oder 27 Mhz ).

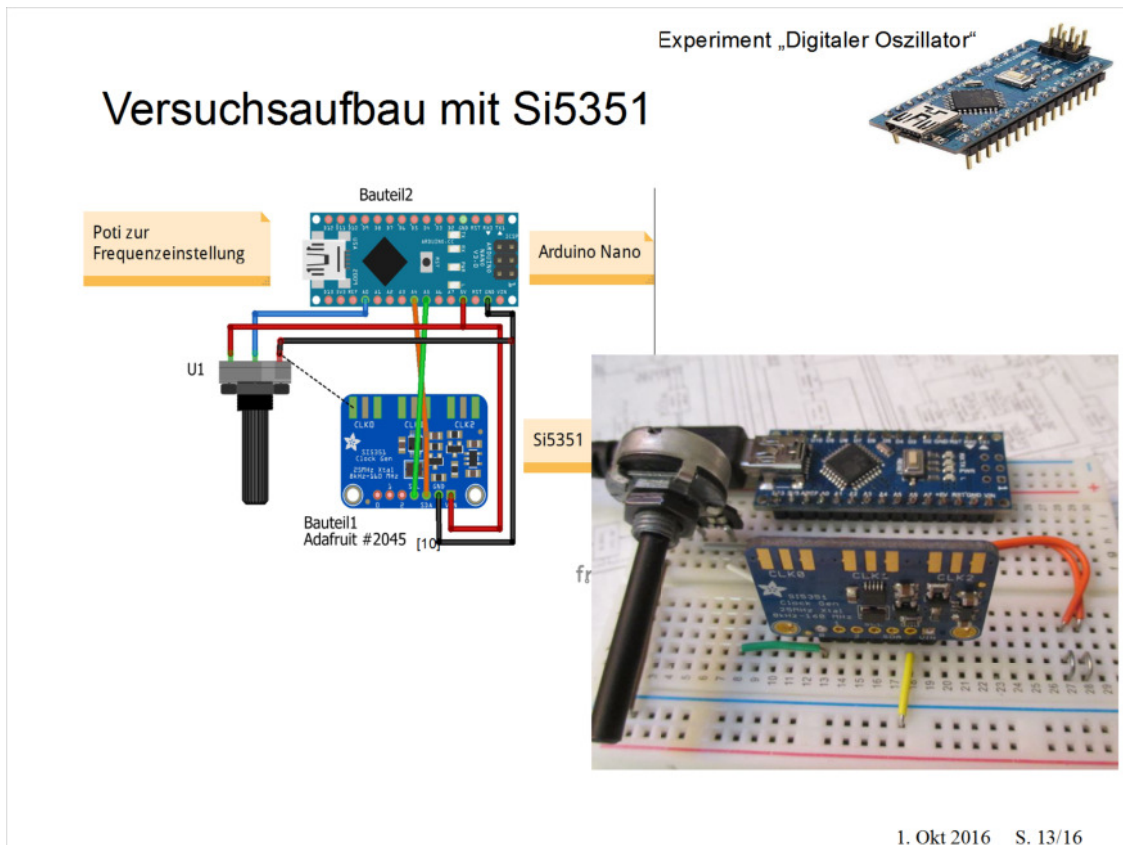
Im Innern befindet sich einige

- **2 x PLL-VFO**, der typisch auf 900 Mhz schwingt , siehe Formel
- **3 x Multisynthesizer**, siehe Formel

Es können 3 Frequenzen gleichzeitig ausgegeben werden.  
Die Programmierung ist umfangreich (100 Register mit vielen Optionen).

Aufgrund der umfangreichen Formeln und einigen zu erfüllenden Randbedingungen ist die Programmierung nicht einfach. Von Silicon Labs gibt es dazu ein Entwurfstool.

Aber es gibt auch eine Library für Arduino, die dem Programmierer die Arbeit vereinfacht.



Die Ansteuerung erfolgt über I2C-Bus ( 2 Leitungen ).  
Der Versuchsaufbau ist ein VFO für 7,000 bis 7,040 Mhz.  
Frequenzeinstellung wird der Einfachheit halber über Poti gemacht. Hier gehört natürlich eigentlich ein Drehgeber hin ! , wie auf Seite 6 zu sehen.





## Arduino-Programm zu Si5351

```

1 // https://github.com/etherkit/Si5351Arduino
2 // http://nt7s.com/2014/06/si5351-libraries-and-breakout-board/
3
4 #include "si5351.h"
5 #define PIN_POTI A0
6 Si5351 si5351;
7
8 void setup()
9 { si5351.init(SI5351_CRYSTAL_LOAD_10PF);
10   si5351.set_pll(SI5351_PLL_FIXED, SI5351_PLLA); // 900 MHz
11   si5351.clock_enable(SI5351_CLK0, 1);          // CLK0 freigeben
12 }
13 void loop()
14 { //frequenz vom Poti ablesen:
15   long frequenz = map( analogRead(PIN_POTI), 0, 1023, 7000000, 7040000);
16   //frequenz neu einstellen:
17   si5351.set_freq(frequenz, SI5351_PLL_FIXED, SI5351_CLK0);
18 }

```

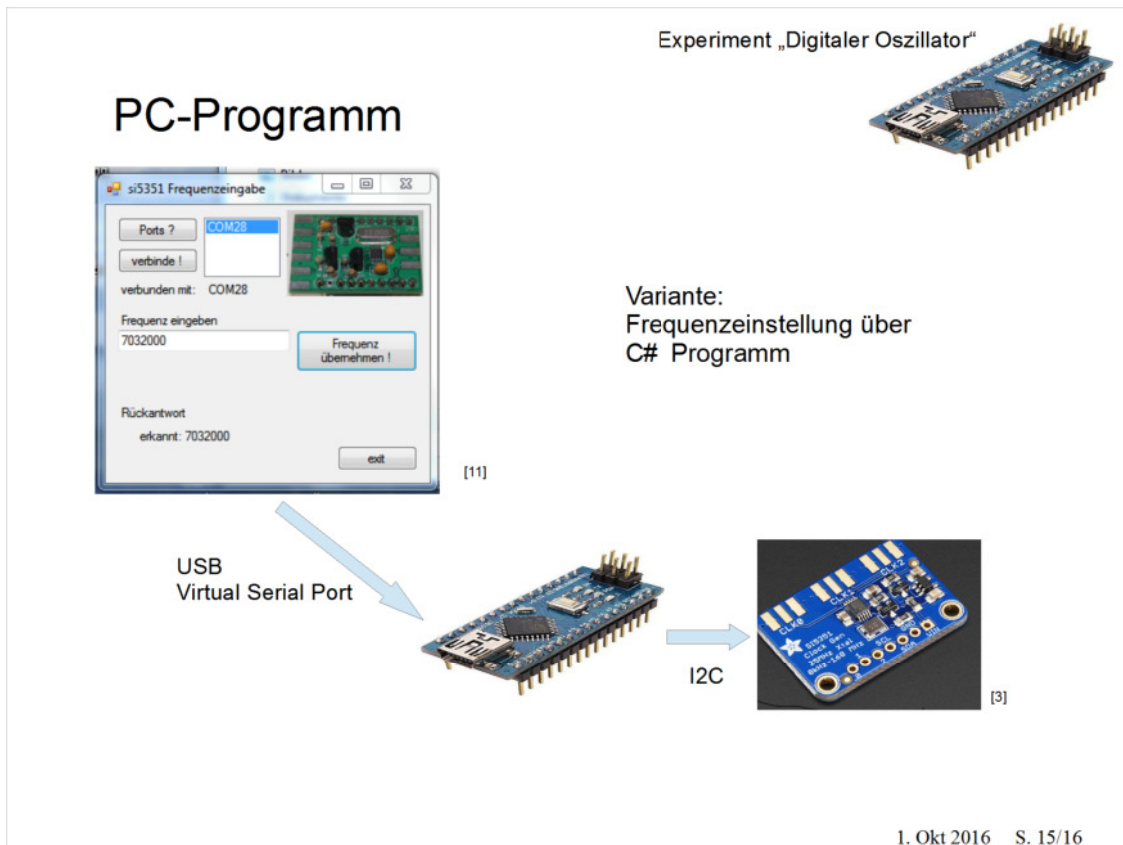
1. Okt 2016 S. 14/16

Die Library vereinfacht uns die Arbeit !

- Das Objekt heißt natürlich „si5351“
- PLLA wird auf 900 Mhz eingestellt.
- Es wird nur CLK0 verwendet.

In einer Schleife ( Loop) wird

- die Soll- Frequenz über Analog-Digital-Wandler  
eingelesen ( Poti ). ( Map() !! )
- die Frequenz neu eingestellt.



Eine Windows App. Macht das Modul von PC aus programmierbar.

Nach Auswahl der Com- Port- Nr. wird die serielle COM-Schnittstelle geöffnet.

Die Frequenzeingabe erfolgt in der Eingabemaske.

Durch Anklicken von „Frequenz übernehmen“ wird die Frequenz als Textstring zum Arduino übertragen; dieser stellt die gewünschte Frequenz ein.

Da nur ASCII-Zeichen verwendet werden, kann die Programmierung auch über Terminal erfolgen.

Anmerkung:

Eine ähnliche Applikation wird in CQDL 10/2016 S.26 vorgestellt, wobei dort die Pins der seriellen COM-Schnittstelle direkt den SPI- Bus eines AD9851 steuern.



## Quellenangaben

- [ 1] Tietze-Schenk, Halbleiter Schaltungstechnik
- [ 2] Ebay
- [ 3] Adafruit <https://www.adafruit.com/>
- [ 4] Small Wonder Labs [http://smallwonderlabs.qrpradio.org/docs/DSW40\\_Manual.pdf](http://smallwonderlabs.qrpradio.org/docs/DSW40_Manual.pdf)
- [ 5] Steve Weber, KD1JV <http://www.qrpkits.com/>
- [ 6] Analog Devices <http://www.analog.com/en/index.html>  
<http://beta-tools.analog.com/adisimdds>
- [ 7] Elecraft <http://www.elecraft.com/>
- [ 8] Hans Summers G0UPL <http://www.hanssummers.com/>
- [ 9] Silicon Labs <http://www.silabs.com/>
- [ 10] Fritzing <http://fritzing.org>
- [ 11] Microsoft "Visual C# 2010 Express Edition"  
<http://www.chip.de/downloads/>
- [ 12] Antennenanalyser nach KB6BEZ ( modifiziert )  
[http://dk2jk.darc.de/arduino/antennenanalyser/prototyp\\_m4/beschreibung.pdf](http://dk2jk.darc.de/arduino/antennenanalyser/prototyp_m4/beschreibung.pdf)