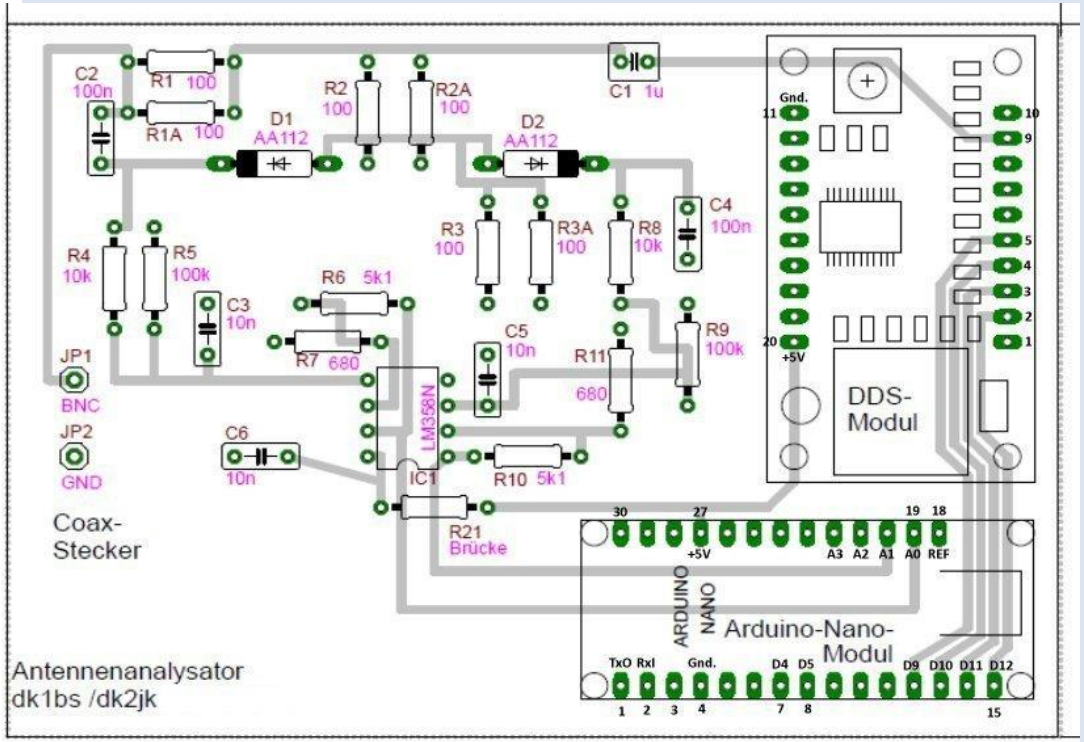
Arduino DDS 9851

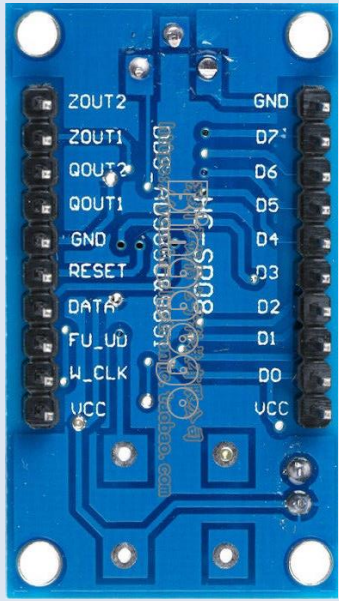
- Inhalt
- Schaltplan von DK1BS / DK2JK
- Arduino Nano
- Schaltplan NanoESP (Pretzel Board)
- Sketch
- <frequency tuning word>
- 40-bit Steuerwort
- AD9851 Programmschritte
- W1 bis W4 finden
- Bits übertragen
- Informationen

H39@email.de

Schaltplan von DK1BS / DK2JK

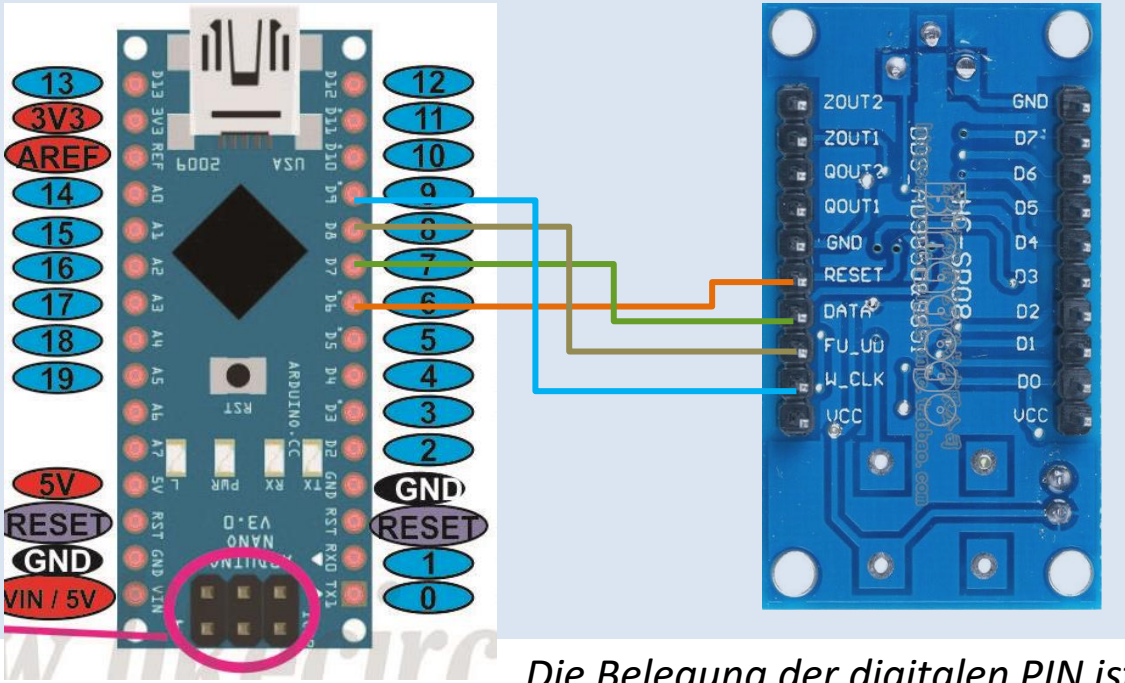


D9	5	RESET
D10	4	DATA
D11	3	FQ_UD
D12	2	W_CLK



Quelle: *http://www.kh-gps.de/ant_analyzer.htm*

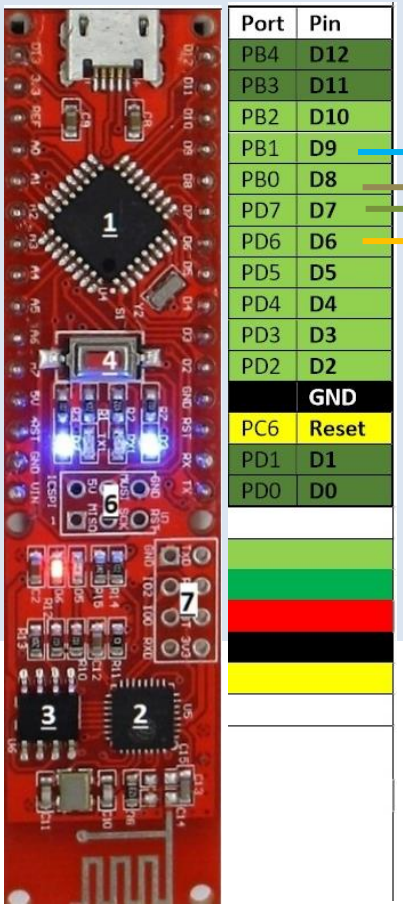
Arduino Nano



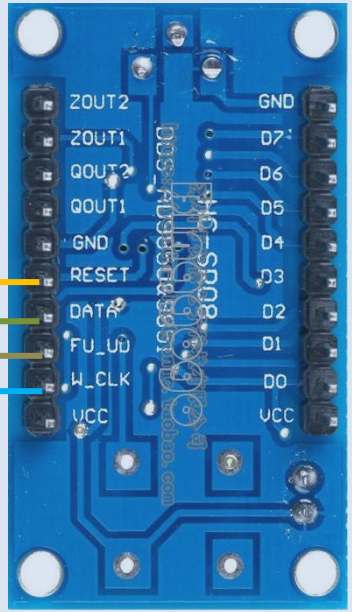
D9	W_CLK	Blau
D8	FQ_UD	Braun
D7	DATA	Grün
D6	RESET	Orange

Die Belegung der digitalen PIN ist frei wählbar!

Schaltplan NanoESP (Pretzel Board)



Port	Pin
PB4	D12
PB3	D11
PB2	D10
PB1	D9
PB0	D8
PD7	D7
PD6	D6
PD5	D5
PD4	D4
PD3	D3
PD2	D2
	GND
PC6	Reset
PD1	D1
PD0	D0



D9	W_CLK	Blau
D8	FQ_UD	Braun
D7	DATA	Grün
D6	RESET	Orange

Die Belegung der digitalen PIN ist frei wählbar!

Sketch

Pin-Zuordnung im Code

```
// Adduino UNO & Nano
const int RESET=6;
const int DATA=7;
const int FQ_UD=8;
const int W_CLK=9;
```

„frequency tuning word“ Anpassung an AD9851 mit 180 MHz:

```
    // Calculate the DDS word - from AD9850 Datasheet
    // int32_t f = Freq_Hz * 4294967296.0/125000000;    // 125 MHz
    // Calculate the DDS word - from AD9851 Datasheet
int32_t f = Freq_Hz * 4294967296 / 180e6;               // 180 MHz inserted by EBW Enno
```

„REFCLK“ setzen Anpassung an AD9851:

```
    // 5th byte needs to be zeros, REFCLK not set
    // send_byte(0);
    // 5th byte with REFCLK set
send_byte(0x01);                        // send control byte with setting REFCLK
```

<frequency tuning word>

Quelle	AD9851.pdf
	Im „parallel mode“ kann das 40-bit Steuerwort, aufgeteilt in 5 Bytes mit je 8 Bit, Byte für Byte, d.h. in 5 Schritten übertragen werden.
	Die ersten 4 Bytes enthalten das <frequency tuning word>, also die gegebene Frequenz:
Frequenz	*fout = <System Clock> * <frequency tuning word> / 2^32 in Hz*
fout	Geforderte Frequenz in Hz
<System Clock>	*„180e6 MHz“* Referenzschwingung, hier 30 MHz Quarz multipliziert mit 6. Dann REFCLK setzen.
2^32=	*4294967296*
<frequency tuning word>	= t_freq
	Das <frequency tuning word> „t_freq“ kann berechnet werden mit:
t_freq=	*fout * 4294967296 / 180e6 in Hz*

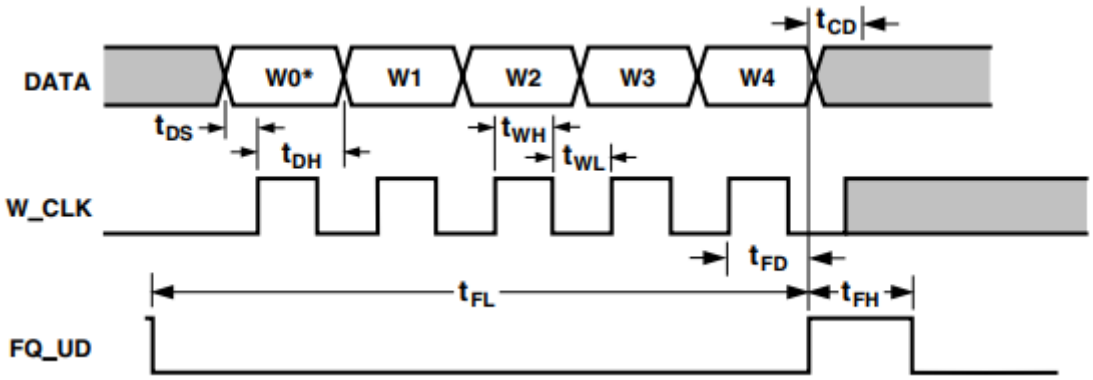
40-bit Steuerwort

Beispiel	fout = *10e6* Hz				
Dezimal	t_freq = *238609296*				
Binär	t_freq = B *00001110 00111000 11100011 10010000* (Leerzeichen wegdenken)				
40-bit Steuerwort bestehend aus: 8 bit Steuerbyte und 32 bit Frequenz	W0	W1	W2	W3	W4
	s. unten	*00001110*	*00111000*	*11100011*	*10010000*
Steuerbyte	W0				
REFCLK setzen	*0x01 oder B00000001*				
	Beginnend mit W4 bis W0, muss das <frequency tuning word> Byte für Byte übertragen werden.				
W_CLK	Das HIGH/LOW-schreiben des W_CLK-PINs trennt die Bytes voneinander.				
FQ_UD	Das HIGH/LOW-schreiben des FQ_UD-PINs sendet das 40-bit-Register				

AD9851 Programmschritte

Quelle AD9851.pdf

1. RESET
2. Byte W4 übertragen
3. W_CLK HIGH/LOW
4. Byte W3 übertagen
5. W_CLK HIGH/LOW
6. Byte W2 übertagen
7. W_CLK HIGH/LOW
8. Byte W1 übertagen
9. W_CLK HIGH/LOW
10. Byte WO übertragen
11. W_CLK HIGH/LOW
12. FQ_UD HIGH/LOW



Sketch: W1 bis W4 finden

Sichtbarer dezimaler Wert:	t_freq=238609296	
Binär im Speicher:	00001110001110001110001110010000	
Aufteilen in 4 Byte:	for (int b=0; b<4; b++, t_freq>>=8) { … } mit „>>“-Operator	
für b=0:	00001110001110001110001**10010000**	
für b=1:	00001110001110001100011	
für b=2:	0000111000111000	
für b=3:	00001110	
Die ersten 8 Bit selektieren:	(t_freq & **0xFF**) mit binärem „&-Operator“ maskieren	
z.B. für b=0:	00001110001110001110001110010000	
	00000000000000000000000011111111	Hex „0xFF“ mit „&“
	00000000000000000000000**10010000**	Ergibt erstes zu übertragenes Byte

Sketch: Bits übertragen

für b=0:	00000000000000000000000**10010000** data_to_send =10010000	
Aufteilen in Bits	for (int i=0; i<8; i++, data_to_send>>=1) {...} mit „>>“-Operator	
i=0	10010000	
i=4	00001001	(4 * >>)
…	…	
i=7	1	(7 * >>)
Das erste Bit selektieren:	(… data_to_send & 0x01) mit binärem „&-Operator maskieren	
für i=4	00001001	
	00000001	Hex „0x01“ mit „&“
	00000001	Zu übertragenes Bit rechts

Informationen

http://www.dk2jk.darc.de/vna_dk2jk/
http://www.dk2jk.darc.de/vna_dk2jk/dokumentation/13dez2014/antennen_analyser_Baumappe%20v2.pdf

http://www.kh-gps.de/ant_analyzer.htm

https://www.electrodragon.com/w/AD9850_Module_DDS_Signal_Generator_V2

http://elektronikbasteln.pl7.de/ad9851.html