SWEEP9850

```
#define DDS_CLOCK 120000000    // frequentie van de DDS-klok (in Hz)
byte LOAD = 8;                 // I/O 8 is verbonden met FQ_UD van de DDS (frequency update)
byte CLOCK = 9;                // I/O 9 is verbonden met W_CLK van de DDS (word clock)
byte DATA = 10;                // I/O 10 is verbonden met D7 van de DDS (data)

void setup()                   // deze instructies worden eenmaal uitgevoerd
{
// Plaats alle I/O pennen in OUTPUT mode
  pinMode (DATA, OUTPUT);
  pinMode (CLOCK, OUTPUT);
  pinMode (LOAD, OUTPUT);
}

void loop()                     // deze instructies worden doorlopend uitgevoerd
{
  // Doe een frequentiezwaai tussen 1 en 10 kHz in stappen van 1 Hz
  for(unsigned long freq = 1000; freq < 10000; freq++)
  {
    sendFrequency(freq);
    delay(2);
  }
}
void sendFrequency(unsigned long frequency)
{
// Bereken het datawoord
  unsigned long tuning_word = (frequency * pow(2, 32)) / DDS_CLOCK;
// Zet W_CLK laag
  digitalWrite(CLOCK, LOW);
// Zet FQ_UD laag
  digitalWrite(LOAD, LOW);
// Klok de eerste 8 bits in het register van de DDS (W0-W7)
  shiftOut(DATA, CLOCK, LSBFIRST, tuning_word);
// Klok de volgende 8 bits (W8-W15)
  shiftOut(DATA, CLOCK, LSBFIRST, tuning_word >> 8);
// Klok de volgende 8 bits (W16-W23)
  shiftOut(DATA, CLOCK, LSBFIRST, tuning_word >> 16);
// Klok de volgende 8 bits (W24-W31)
  shiftOut(DATA, CLOCK, LSBFIRST, tuning_word >> 24);
// Klok de volgende 8 bits = 0 (W32-W39)
  shiftOut(DATA, CLOCK, LSBFIRST, 0x0);
// Zet FQ_UD hoog om het datawoord door te schuiven in de DDS (= nieuwe frequentie)
  digitalWrite(LOAD, HIGH);
}
```

HOP9850

```
// Frequentiesturing van een AD9850 DDS via een schakelaar met drie standen
#define DDS_CLOCK 120000000    // frequentie van de DDS-klok (in Hz)
byte LOAD = 8;                 // I/O 8 is verbonden met FQ_UD van de DDS
byte CLOCK = 9;                // I/O 9 is verbonden met W_CLK van de DDS
byte DATA = 10;                // I/O 10 is verbonden met D7 van de DDS
byte switchA1 = 11;            // I/O 11 is verbonden met schakelaar A, pen 1
byte switchA2 = 12;            // I/O 12 is verbonden met schakelaar A, pen 2
byte switchA3 = 13;            // I/O 13 is verbonden met schakelaar A, pen 3
unsigned long freq1 = 50040000;
unsigned long freq2 = 50041000;
unsigned long freq3 = 50042000;
int switchNu = 1;
int switchVorig = 0;

void setup()                   // deze instructies worden eenmaal uitgevoerd
{
  pinMode (DATA, OUTPUT);      // I/O 10 is output van het tuning word
  pinMode (CLOCK, OUTPUT);     // I/O 9 is output van de klokpuls
  pinMode (LOAD, OUTPUT);      // I/O 8 is output van de finale puls FQ_UD
  pinMode (switchA1, INPUT);   // via I/O 11 wordt de status van de schakelaar pen 1 gelezen
  pinMode (switchA2, INPUT);   // via I/O 12 wordt de status van de schakelaar pen 2 gelezen
  pinMode (switchA3, INPUT);   // via I/O 13 wordt de status van de schakelaar pen 3 gelezen
}
void loop()                     // deze instructies worden doorlopend uitgevoerd
{
  if (digitalRead(switchA1) != 0)  // schakelaar A1 hoog?
    switchNu = 1;
  if (digitalRead(switchA2) != 0)  // schakelaar A2 hoog?
    switchNu = 2;
  if (digitalRead(switchA3) != 0)  // schakelaar A3 hoog?
    switchNu = 3;
  if (switchNu != switchVorig)     // als de schakelstand gelijk is: niets uitvoeren
  {
    switch (switchNu)
    {
      case 1:                      // A1 is gesloten
        sendFrequency(freq1);
        break;
      case 2:                      // A2 is gesloten
        sendFrequency(freq2);
        break;
      case 3:
        sendFrequency(freq3);      // A3 is gesloten
        break;
    }
    switchVorig = switchNu;
    delay (1000);
  }
}

void sendFrequency(unsigned long frequency) → zie SWEEP9850
```

FSKCW9850

```
// DDS FSKCW generator
// fragmenten afkomstig uit Morse.h (Erik Linder SM0RVV and Mark VandeWettering K6HX)
// morsetabel aangevuld door ON4LP

#define DDS_CLOCK 120000000    // frequentie van de DDS-klok (in Hz)
byte LOAD = 8;                 // I/O 8 is verbonden met FQ_UD van de DDS (frequency update)
byte CLOCK = 9;                // I/O 9 is verbonden met W_CLK van de DDS (word clock)
byte DATA = 10;                // I/O 10 is verbonden met D7 van de DDS (data)
byte speedWpm = 8;                   // seinsnelheid in wpm
int lenDah;                  // lengte dah
int lenDit;                  // lengte dit
unsigned long int freqLo = 28350000;      // DDS lage frequentie
unsigned long int freqHi = 28350170;      // DDS hoge frequentie
byte morseTab[] = {          // 1 = geen morseteken beschikbaar
    107, //ASCII 33 !
    82,  //ASCII 34 "
    1,   //ASCII 35 #
    137, //ASCII 36 $
    1,   //ASCII 37 %
    40,  //ASCII 38 &
    94,  //ASCII 39 '
    109, //ASCII 40 (
    109, //ASCII 41 )
    1,   //ASCII 42 *
    42,  //ASCII 43 +
    115, //ASCII 44 ,
    97,  //ASCII 45 -
    106,  //ASCII 46 .
    41,  //ASCII 47 /
    63,  //ASCII 48 0
    62,  //ASCII 49 1
    60,  //ASCII 50 2
    56,  //ASCII 51 3
    48,  //ASCII 52 4
    32,  //ASCII 53 5
    33,  //ASCII 54 6
    35,  //ASCII 55 7
    39,  //ASCII 56 8
    47,  //ASCII 57 9
    120, //ASCII 58 :
    53,  //ASCII 59 ;
    1,   //ASCII 60 <
    49,  //ASCII 61 =
    1,   //ASCII 62 >
    76,  //ASCII 63 ?
    69,  //ASCII 64 @
    6,   //ASCII 65 A
    17,  //ASCII 66 B
    21,  //ASCII 67 C
```

```
9,   //ASCII 68 D
2,   //ASCII 69 E
20,  //ASCII 70 F
11,  //ASCII 71 G
16,  //ASCII 72 H
4,   //ASCII 73 I
30,  //ASCII 74 J
13,  //ASCII 75 K
18,  //ASCII 76 L
7,   //ASCII 77 M
5,   //ASCII 78 N
15,  //ASCII 79 O
22,  //ASCII 80 P
27,  //ASCII 81 Q
10,  //ASCII 82 R
8,   //ASCII 83 S
3,   //ASCII 84 T
12,  //ASCII 85 U
24,  //ASCII 86 V
14,  //ASCII 87 W
25,  //ASCII 88 X
29,  //ASCII 89 Y
19,  //ASCII 90 Z
1,   //ASCII 91 [
1,   //ASCII 92 \
1,   //ASCII 93 ]
1,   //ASCII 94 ^
77,  //ASCII 95 _
94,  //ASCII 96 `
6,   //ASCII 97 a
17,  //ASCII 98 b
21,  //ASCII 99 c
9,   //ASCII 100 d
2,   //ASCII 101 e
20,  //ASCII 102 f
11,  //ASCII 103 g
16,  //ASCII 104 h
4,   //ASCII 105 i
30,  //ASCII 106 j
13,  //ASCII 107 k
18,  //ASCII 108 l
7,   //ASCII 109 m
5,   //ASCII 110 n
15,  //ASCII 111 o
22,  //ASCII 112 p
27,  //ASCII 113 q
10,  //ASCII 114 r
8,   //ASCII 115 s
3,   //ASCII 116 t
12,  //ASCII 117 u
24,  //ASCII 118 v
14,  //ASCII 119 w
25,  //ASCII 120 x
29,  //ASCII 121 y
```

```
    19   //ASCII 122 z
    };

void setup()
{
// Plaats alle I/O pennen naar de DDS in OUTPUT mode
  pinMode (DATA, OUTPUT);
  pinMode (CLOCK, OUTPUT);
  pinMode (LOAD, OUTPUT);
  lenDit = (1200/speedWpm);
  lenDah = (3*lenDit);
}

void loop()                     // deze instructies worden doorlopend uitgevoerd
{
  sendMsg("ON6MS JO10UX");
  sendFrequency(freqLo);
  delay(5000);
}


void sendMsg(char *str)
  {
   while (*str) {
     send(*str++);
  }
}

void dah()
{
  sendFrequency(freqHi);
  delay(lenDah);
  sendFrequency(freqLo);
  delay(lenDit);
}

void dit()
{
  sendFrequency(freqHi);
  delay(lenDit);
  sendFrequency(freqLo);
  delay(lenDit);
}


void send(char c)
{
  byte i;
  byte p;

  if (c == ' ') {
    delay(7*lenDit) ;
    return ;
  }
```

```
// Morsewaarde opzoeken in de tabel
  else {
    i = ((byte) c) - 33;
    p = morseTab[i];
  }

// Main algoritm for each morse sign
  while (p != 1) {
    if (p & 1)
      dah();
    else
      dit();
    p = p / 2;
  }
  // Tekenspatie
  delay(2*lenDit);
}


void sendFrequency(unsigned long frequency)
{
// Bereken het datawoord
  unsigned long tuning_word = (frequency * pow(2, 32)) / DDS_CLOCK;
// Zet W_CLK laag
  digitalWrite(CLOCK, LOW);
// Zet FQ_UD laag
  digitalWrite(LOAD, LOW);
// Klok de eerste 8 bits in het register van de DDS (W0-W7)
  shiftOut(DATA, CLOCK, LSBFIRST, tuning_word);
// Klok de volgende 8 bits (W8-W15)
  shiftOut(DATA, CLOCK, LSBFIRST, tuning_word >> 8);
// Klok de volgende 8 bits (W16-W23)
  shiftOut(DATA, CLOCK, LSBFIRST, tuning_word >> 16);
// Klok de volgende 8 bits (W24-W31)
  shiftOut(DATA, CLOCK, LSBFIRST, tuning_word >> 24);
// Klok de volgende 8 bits = 0 (W32-W39)
  shiftOut(DATA, CLOCK, LSBFIRST, 0x0);
// Zet FQ_UD hoog om het datawoord door te schuiven in de DDS (= nieuwe frequentie)
  digitalWrite(LOAD, HIGH);
}
```

Nuttige links:

www.arduino.cc – de enige echte Arduino site
arduino.cc/en/Reference/HomePage De Engelstalige Arduino referentiemanual
www.kompanje.nl/arduino/Arduino%20manual%201_0%20NL.pdf – Arduino programmeer manual Nederlands
www.analog.com/static/imported-files/data_sheets/AD9850.pdf Datasheet AD9850
www.electricalelectronics.jotoexplorer.com/electrical-engineering/digital-frequency-synthesis-demystified-free-pdf/ - Digital Frequency Synthesis Demystified – Free PDF