Prosty układ VFO na bazie syntezy DDS 9850
z wizualizacją stanu pracy Trx_a na wyświetlaczu LCD.
Praca konkursowa PUK 2013.

Prezentowany układ posiada budowę modułową z przeznaczeniem do wykorzystania w prostych Trx_ach. W jego skład wchodzi:

- syntezer na układzie AD9850,
- wzmacniacz sygnału generowanego na tranzystorze 2N3904,
- wyświetlacz LCD 2x16 znaków, obrazujący aktualną częstotliwość pracy Trx_a, a także aktualny stan pracy Trx_a, w tym Smetr w formie bargrafu słupkowego;
- całością steruje mikroprocesor Atmega 168P (328P) z oprogramowaniem pisanym w kompilatorze ARDUINO.

Inspiracją projektu były liczne wersje podobnych rozwiązań opisywane na stronach internetowych, lecz żadne z nich nie stanowiło wersji pełnej, mogącej spełniać więcej niż jedną z funkcji podstawowych.

Opis działania:

- ***Modulacja SSB.***

Trybem domyślnym pracy układu jest stan Rx dla SSB i częstotliwość wyświetlana wynosi 3.700.00 (kHz), z uwzględnieniem przesunięcia pilota Generatora Fali Nośnej nadajnika o 1.500 kHz dla modulacji SSB. Częstotliwość generowana DDS jest większa o wartość pośredniej częstotliwości i wynosi 12.688.400 kHz, przy zastosowanym w prototypie Trx_a filtrze o Fśr = 8,989,900 kHz.
Po przejściu na nadawanie informacja o tym stanie jest wyświetlenie na wyświetlaczu trybu pracy „Tx", poprzez podanie na pin „INP Tx" modułu napięcia stałego w granicach 5-12V (z przekaźnika w układzie Trx_a).

- ***Modulacja CW.***.

Po podaniu napięcia na pin „INP CW" w granicach 5-12V z układu Trx_a, wyświetlana częstotliwość uwzględnia tym razem różnicę dla Generatora Fali Nośnej przy CW tylko o 800 Hz, a generowana przez DDS jest większa o wartość pośredniej częstotliwości dla wykorzystanego filtru jw. Wyświetlona jest również informacja na wyświetlaczu LCD o trybie pracy CW i analogicznie o przejściu na nadawanie jak przy SSB.

- ***Smetr w formie wizualnej bargrafu.***

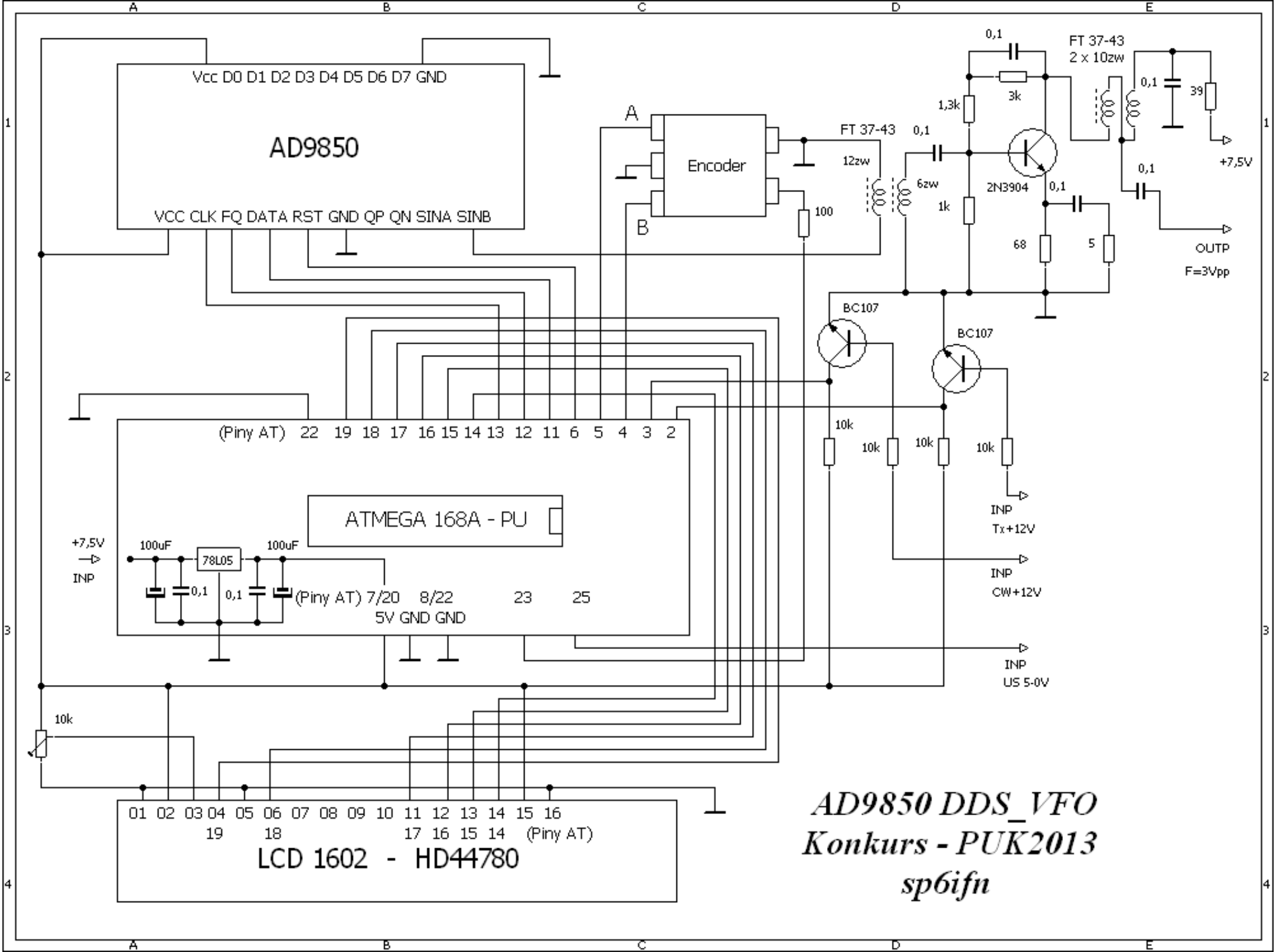
Kolejną realizowaną funkcją układu jest prosty, wizualny wskaźnik w formie bargrafu, który można wykorzystać jako wskaźnik sygnału Smetra, mocy nadawanej własnego nadajnika, lub obie razem, odpowiednio włączając tą funkcje do układu budowanego Trx_a. Realizowane jest to poprzez podanie na pin „US"napięcia z układu ARW odbiornika w granicach 5-0V. Wartości wyświetlane oblicza procesor w zależności od napięcia podanego na jedno z jego wejść analogowych. Wartości napięcia wejściowego mogą być modyfikowane programowo, w zależności od potrzeb konstruktora.

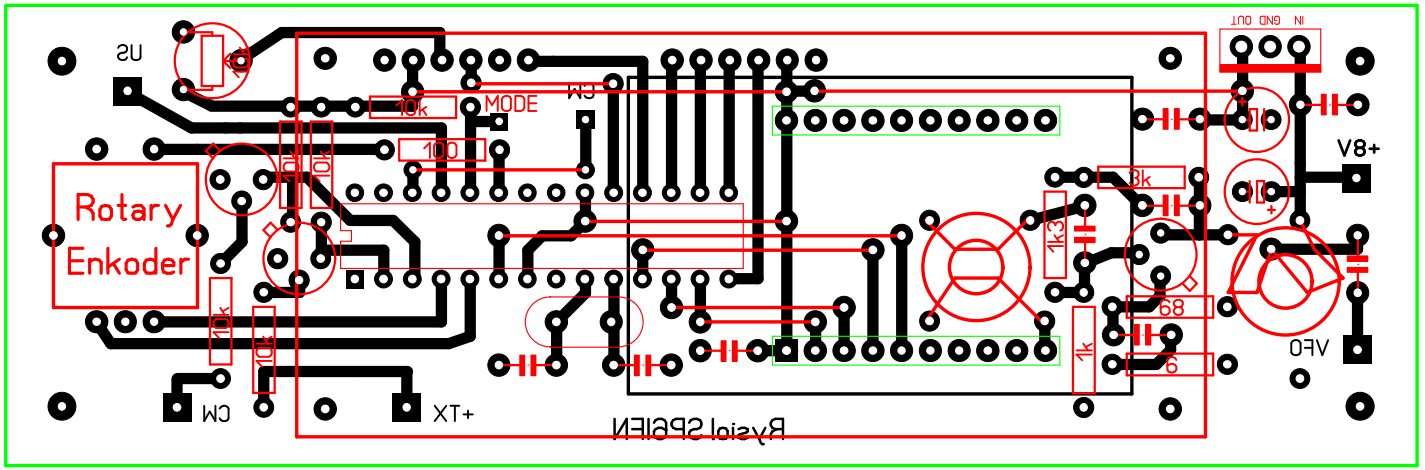
Prezentowany układ charakteryzuje wysoka stabilność częstotliwości, może on zadowolić każdego konstruktora prostych układów nadawczo-odbiorczych, a pisane oprogramowanie w otwartej platformie programistycznej ARDUINO pozwala na wprowadzane dowolnych zmian wg własnego uznania. Poziom sygnału wyjściowego generatora wynosi 3 Vpp/50ohm i wystarcza do wysterowania diodowego mieszacza budowanego odbiornika lub nadajnika.

Załącznikiem niniejszego opisu jest:

- schemat elektryczny układu,
- rysunek płytki pcb z rozmieszczeniem elementów,
- treść oprogramowania.

Burzenin dnia 14 września 2013r Banasiak Ryszard SP6IFN


Vcc D0 D1 D2 D3 D4 D5 D6 D7 GND
AD9850
VCC CLK FQ DATA RST GND QP QN SINA SINB
A
B
Encoder
100
FT 37-43
12zw
6zw
0,1
1,3k
3k
2N3904
1k
68
5
39
+7,5V
OUTP
F=3Vpp
FT 37-43
2 x 10zw
BC107
BC107
10k
10k
10k
10k
INP
Tx+12V
INP
CW+12V
INP
US 5-0V
(Piny AT) 22 19 18 17 16 15 14 13 12 11 6 5 4 3 2
ATMEGA 168A - PU
+7,5V
INP
100uF
78L05
100uF
0,1
0,1
(Piny AT) 7/20 8/22 23 25
5V GND GND
10k
01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16
19 18 17 16 15 14 (Piny AT)
LCD 1602 - HD44780
AD9850 DDS_VFO
Konkurs - PUK2013
sp6ifn


Rotary
Enkoder
MODE
+8V
VFO
+TX

```
/*Wersja obslugi DDS na podstawie opisu http://www.ad7c.com/
z licznymi modyfikacjami pod wlasne potrzeby
Bargraf zrealizowano na bazie opisu
http://www.rcgroups.com/forums/showthread.php?t=1298192
<sp6ifn> http://www.qsl.net/sp6ifn/
.....*/

#include <LiquidCrystal.h>  //wywolanie biblioteki wyswietlacza
#include <rotary.h>         //wywolanie biblioteki enkodera
                      //Ustawienia niektorych elementow
#define W_CLK 7            // Pin 7 - connect to AD9850 module word load clock pin (CLK)
#define FQ_UD 6            // Pin 6 - connect to freq update pin (FQ)
#define DATA 5             // Pin 5 - connect to serial data load pin (DATA)
#define RESET 4            // Pin 4 - connect to reset pin (RST)
#define pulseHigh(pin) {digitalWrite(pin, HIGH); digitalWrite(pin, LOW); }

Rotary r = Rotary(2,3); // ustawienie pinow enkodera,konieczne sa przerwania impulsow

LiquidCrystal lcd(13, 12, 11, 10, 9, 8); // Wlasna implementacja pinow dla wyswietlacza LCD
int_fast32_t rx=12688400;              // Poczatkowa czestotliwosc VFO
int_fast32_t rx2=1;                   // zmienna do przechowywania zaktualizowanej czestotliwosci
int_fast32_t increment = 10;           // poczatkowy przyrost aktualizacji VFO w Hz
int buttonstate = A0;
String hertz = "10 Hz";
int  hertzPosition = 11;
byte ones,tens,hundreds,thousands,tenthousands,hundredthousands,millions ;  //Symbole zastepcze

 int battPin = A2;                //deklaracja wejscia pomiarowego analog_2
 float d = 0;                    //zmienna pomocnicza
 float oldit = 0;                //stara zmienna pomiarowa
 float rawBatt;                  //zmienna pomiaru
 int PTT = 0;                    //przypisanie PTT do pinu cyfrowego "0"
 int CWlcd = 1;                  //przypisanie CWlcd do pinu cyfrowego "1"
 int SW = A1;
 int CW = A3;


void setup()
{
  pinMode(A0,INPUT);              // Podlaczenie przycisku, ktory idzie do GND przy wcisnieciu
  digitalWrite(A0,HIGH);
  lcd.begin(16, 2);
  PCICR |= (1 << PCIE2);
  PCMSK2 |= (1 << PCINT18) | (1 << PCINT19);
  sei();
  pinMode(FQ_UD, OUTPUT);
  pinMode(W_CLK, OUTPUT);
  pinMode(DATA, OUTPUT);
  pinMode(RESET, OUTPUT);
  pinMode(PTT, INPUT);
  digitalWrite(PTT, HIGH);
  pinMode (CWlcd, INPUT);
  digitalWrite (CWlcd, HIGH);
```

```
  pinMode(SW,INPUT);
  digitalWrite(SW,HIGH);
  pinMode(CW,OUTPUT);
  digitalWrite(CW,LOW);              //stan poczatkowy - domyslny dla SSB
  pulseHigh(RESET);
  pulseHigh(W_CLK);
  pulseHigh(FQ_UD);                  // Impuls ten pozwala na seryjny tryb pracy AD9850
                              //- Dane techniczne str.12 Datasheetu.
  lcd.setCursor(hertzPosition,1);
  lcd.print(hertz);
}

ISR(PCINT2_vect)                     //funkcja na ustalenie zakresu przestrajania DDS
{
  unsigned char result = r.process();
  if (result)
  {
    if (result == DIR_CW){rx=rx+increment;}
    else {rx=rx-increment;};
      if (rx >=12800000){rx=rx2;};       // Gorna granica pracy VFO
      if (rx <=12480000){rx=rx2;};       // Dolna granica pracy VFO
  }
}

// frequency calc from datasheet page 8 = <sys clock * frequency tuning word>/2^32
void sendFrequency(double frequency)               //funkcja na wyliczenie czestotliwosci AD9850
{
  int32_t freq = frequency * 4294967295/125005000;  //Uwaga!, 125 MHz zegar jest na AD9850.
                                       //Mozesz tutaj zrobic "drobne" zmiany strojenia
                                       //wpisujac inna czestotliwosc zegara tzw."korekta".
  for (int b=0; b<4; b++, freq>>=8)
  {
    tfr_byte(freq & 0xFF);
  }
  tfr_byte(0x000);                   // Koncowy bajt kontrolny, wszystkie "0" dla 9850
  pulseHigh(FQ_UD);                  // Gotowe!
}
// transfer danych w czasie, pierwszy LSB do 9850 przez port szeregowy DANYCH
void tfr_byte(byte data)
{
  for (int i=0; i<8; i++, data>>=1)
  {
    digitalWrite(DATA, data & 0x01);
    pulseHigh(W_CLK);                //po kazdym wysłanym bit, CLK jest impulsowe wysokie
  }
}
void setincrement()       //funkcja na ustawianie kroku strojenia enkoderem i wyswietlania na LCD
{
  if(increment == 10){increment = 50; hertz = "50 Hz"; hertzPosition=11;}
  else if (increment == 50){increment = 100;  hertz = "100 Hz"; hertzPosition=10;}
  else if (increment == 100){increment = 500; hertz="500 Hz"; hertzPosition=10;}
  else if (increment == 500){increment = 1000; hertz="1 Khz"; hertzPosition=11;}
```

```
  else if (increment == 1000){increment = 10000; hertz="10 Khz"; hertzPosition=10;}
  else{increment = 10; hertz = "10 Hz"; hertzPosition=11;};
   lcd.setCursor(0,1);
   lcd.print("               ");
   lcd.setCursor(hertzPosition,1);
   lcd.print(hertz);
   delay(250);             // opoznienie przyspieszenia / spowolnienia przycisku predkosci przewijania.
};

void showFreq(int_fast32_t rxssb) //funkcja wyswietlania czestotliwosci na LCD z uwzglednieniem BFO dla emisji ssb
{
    rxssb = rx - 8988400;          //czestotliwosc wyswietlana pomniejszona o czestotliwosc BFO dla SSB
   // rxcw = rx - 8989200;         //czestotliwosc wyswietlana pomniejszona o czestotliwosc BFO dla CW
    millions = int(rxssb/1000000);
    hundredthousands = ((rxssb/100000)%10);
    tenthousands = ((rxssb/10000)%10);
    thousands = ((rxssb/1000)%10);
    hundreds = ((rxssb/100)%10);
    tens = ((rxssb/10)%10);
    ones = ((rxssb/1)%10);
    lcd.setCursor(0,0);
    lcd.print("               ");
   if (millions > 9){lcd.setCursor(2,0);}
   else{lcd.setCursor(8,0);}
    lcd.print(millions);
    lcd.print(".");
    lcd.print(hundredthousands);
    lcd.print(tenthousands);
    lcd.print(thousands);
    lcd.print(".");
    lcd.print(hundreds);
    lcd.print(tens);

};

void showFreq2(int_fast32_t rxcw)  //funkcja wyswietlania czestotliwosci na LCD z uwzglednieniem BFO dla emisji cw
{
   // rxssb = rx - 8988400;        //czestotliwosc wyswietlana pomniejszona o czestotliwosc BFO dla SSB
    rxcw = rx - 8989200;          //czestotliwosc wyswietlana pomniejszona o czestotliwosc BFO dla CW
    millions = int(rxcw/1000000);
    hundredthousands = ((rxcw/100000)%10);
    tenthousands = ((rxcw/10000)%10);
    thousands = ((rxcw/1000)%10);
    hundreds = ((rxcw/100)%10);
    tens = ((rxcw/10)%10);
    ones = ((rxcw/1)%10);
```

```
   lcd.setCursor(0,0);
   lcd.print("               ");
  if (millions > 9){lcd.setCursor(2,0);}
  else{lcd.setCursor(8,0);}
   lcd.print(millions);
   lcd.print(".");
   lcd.print(hundredthousands);
   lcd.print(tenthousands);
   lcd.print(thousands);
   lcd.print(".");
   lcd.print(hundreds);
   lcd.print(tens);

};

void loop()
{
 buttonstate = digitalRead(A0);
 if(buttonstate == LOW)
 {
      setincrement();
   }
 if (digitalRead(SW) == LOW)       //sprawdzenie czy wlaczono prace CW, jesli tak...
 {
  digitalWrite(CW,HIGH);           //zalaczony tryb pracy TRx_a dla CW
  while(digitalRead(SW)==LOW);     //oczekiwanie na zalaczenie SW
  }else                            //w przeciwnym razie wroc do....
   {
    digitalWrite(CW,LOW);          //stanu poczatkowego - domyslny dla SSB
 }

 if (digitalRead(CWlcd) ==LOW)     //sprawdzenie czy wlaczony jest tryb pracy CW
 {
   if (rx != rx2)
  {  showFreq2(rx);                //wykonanie funkcji wyswietlania - rxcw
     sendFrequency(rx);
     rx2 = rx;
   }
  lcd.setCursor(4,0);              //ustawienie kursora w czwartm  rzedzie i pierwszej kolumnie
  lcd.print("CW");                 //wyswietlenie stanu pracy TRx_a
  lcd.setCursor(0,0);              //ustawienie kursora w pierwszym rzedzie i pierwszej kolumnie
  lcd.print("Rx");                //wyswietlenie stanu pracy TRx_a
  while(digitalRead(CWlcd) ==LOW);   //oczekiwanie na zalaczenie trybu pracy CW

 }  else                           //w przeciwnym razie
 {
   if (rx != rx2)
  {  showFreq(rx);                 //wykonanie funkcji wyswietlania - rxssb
     sendFrequency(rx);
     rx2 = rx;
   }
  lcd.setCursor(4,0);              //ustawienie kursora w czwartym rzedzie i pierwszej kolumnie
```

```
 lcd.print(">>");                //wyswietlenie stanu pracy odbiornika dla emisji SSB
 lcd.setCursor(0,0);              //ustawienie kursora w pierwszym rzedzie i pierwszej kolumnie
 lcd.print("Rx");                //wyswietlenie stanu pracy TRx_a
}
   if (digitalRead(PTT) ==LOW)     //sprawdzenie czy wlaczone jest PTT
{
lcd.setCursor(0,0);               //ustawienie kursora w pierwszym rzedzie i pierwszej kolumnie
lcd.print("Tx");                  //wyswietlenie stanu pracy TRx_a
while(digitalRead(PTT) ==LOW);      //oczekiwanie na zalaczenie PTT
}

rawBatt = analogRead(battPin);          // odczyt wejscia pomiarowego dla sygnału US

// RSSI subroutine
float num = analogRead(battPin);        // czytanie wejscia
float it = map(num, 0, 1023, 9, 0);     // odczyt danych, odwrocenie i podzial na 9 czesci,
                                // <== TU JEST KALIBRACJA WYSWIETLANIA

if (it > oldit) {               // sprawdzenie zwiekszania liczby w odniesieniu do starego pomiaru
  for (d = 0; it >= d; d++)    // zliczanie od 0-9
  {
    lcd.setCursor(d, 1);       // poczatek wyswietlania od lewej strony
    lcd.write(1023);           // wyswietlenie do konca zakresu pomiarowego
  }
}
if (it <= oldit) {              // sprawdzenie zmniejszania liczby w odniesieniu do starego pomiaru
  for (d = 9; it <= d; d--)    //odliczanie od 9-0
    {
    lcd.setCursor(d, 1);       // poczatek wyswietlania z prawej dolnej lini wyswietlacza
    lcd.write(1022);           // wyswietlenie pustej linii na LCD
  }
}
oldit = it;
  delay(50);                   //wprowadzenie opoznienia aby uniknac migotania na LCD

}
```