```
// Program til Styring_af_analog_rep_OZ7RES_ASR_15-03-2022_v_1_16_J
//
// Device Arduino NANO.
//
// Dette program er beregnet til at styre analoge FM repeatere.
//
// 20-02-2022: Tilpasses nu til drift sammen med TeamSpeak TS3
// 21-02-2022: pilot_ind, tone_ind og squelch_ind ændret til input1,
input2, input3.
// Pilot ændret til Input1_ , Squelch_ til Input2_ , Tone_ til Input3_
// Enkelt og dobbelt pause dyt styres af flag Input1_Drift
// Pilottonegenerator totalt fjernet *) SE 15-03-2022
// Input1_DelayTimer fjernet
// Ændret Input_StartTimere
// 24-02-2022: Morse stopper straks når der kommer input1 eller input2
// TimeOut 5 min sender enkelt eller dobbelt dyt til allersidst
// 07-03-2022: Call ændret til OZ4RED. Skal bruges på 4 meter med
2xGM900.
// 10-03-2022: Slut- K tilkoblet. Dyt fra TS3 nul-settes. SlutStep
ændret til 70.
// 11-03-2022: (Input2_Ind==1) ændret til (Input2_OK==1) i morsestop
if-sætninger
// Input2_BounceTid ændret fra 10 til 1 (mage til Input1_BounceTid)
// Ændret KoldStart
//
-------------------------------------------------------------------------
// 15-03-2022: ASR version til 2 GM900 ændret til brug i repeater 1 med
TP6000
// Pilottonegenerator geninstalleret
// 03-04-2022: REP 1 70cm ASR med TP6000. K fjernet og dyt-tider ændret
lidt
// 05-04-2022: REP 1 med TP6000 er udgået. REP 4 med 2 GM900 erstatter
REP 1.
// NY REP 4 70cm ASR med 2xGM900. Pilotgenerator fjernet. 1_16_D
// Inverteret D4 tast_til_ts3. Nu aktiv HI. output 5 volt.
// 06-04-2022: ændret div variabelnavne mm. 1-16-E
// 07-04-2022: Input1_Drift nulstilles når Rep_OK = 0. 1-16-F
// digitalWrite(tast_til_ts3) ændret fra Input2_Ind til Input1_Ind (RX
SQ)
// TX_Stop sætter digitalWrite(tast_til_ts3) til LOW ved TimeOut. Luk
TS3.
// Program 1_16_H er nu i både [REP 3 4 meter] og [REP 4 70cm]
repeatere.
//
//
//
//
-------------------------------------------------------------------------
--
// I 2018 skrevet til en 70 cm repeater med TP6000 (REP 1)
// I 2020 tilpasset til 2 meter repeater med 2 stk GM900. (REP 2 med
1750)
// I 2022 tilpasset til 4 meter repeater med 2 stk GM900 med ASR (REP 3)
// I 2022 tilpasset til 70 cm repeater med ASR og TP6000 (REP 1)
(udgået)
// I 2022 tilpasset til 70 cm repeater med ASR og 2 stk GM900(REP 4)
//
// Benforbindelser til NANO er ens i disse repeatere.
// Hardware er lidt forskellig, og der kan derfor være enkelte
```

```
funktioner
// som ikke er mulige i alle repeatere.
//
// Kaldesignal i morsekode, pausedyt med mere genereres i NANO.
//
// Programmet er opbygget med moduler (void), som kaldes efter tur af en
// endeløs sløjfe (void loop). I sløjfen er også flere if-sætninger som
// tester flag og sætter flag som styrer moduler.
// Modulerne indledes med if-sætning, som aflæser flag og bestemmer om
// modulet skal være aktivt eller overspringes.
//
// Beskrivelse på http://www.kjaerbro.dk/arduino_repeater_frame.html
//
// Skrevet af OZ1LN Lasse H.P. Kjærbro 2018. Tilrettet i 2020, 2021 og
2022.
//
// Deklaration af pins på Arduino NANO Repeater 1 TP6000 70cm ASR
-----------
const int disponibel_RX0  =  0; // RX0 bruges ikke lige nu
const int disponibel_TX1  =  1; // TX1 bruges ikke lige nu
const int dobbelt_dyt     =  2; // D2 ud grøn LED. Dobbelt dyt
const int disponibel_3    =  3; // D3 ud bruges ikke
const int tast_til_ts3    =  4; // D4 ud net tastsignal TS3
const int pilot_ud        =  5; // D5 ud bruges ikke ved GM900
const int tone_ud         =  6; // D6 ud morsetegn og dyt output LF
const int TimeOutLED      =  7; // D7 ud lyser når TimeOut er nået.
const int disponibel_8    =  8; // D8 ud bruges ikke
const int tx_tast         =  9; // D9 ud taster sender
const int disponibel_10   = 10; // D10 bruges ikke lige nu
const int disponibel_11   = 11; // D11 bruges ikke lige nu
const int disponibel_12   = 12; // D12 bruges ikke lige nu
const int blinkled        = 13; // D13 ud Blink-LED på NANO
const int input1          = 14; // A0 ind, Squelch fra radio
const int input2          = 15; // A1 ind, Squelch fra TS3
const int input3          = 16; // A2 ind bruges ikke
const int disponibel_A3   = 17; // A3 bruges ikke lige nu
const int disponibel_A4   = 18; // A4 bruges ikke lige nu
const int disponibel_A5   = 19; // A5 bruges ikke lige nu
const int disponibel_A6   = 20; // A6 bruges ikke lige nu
const int disponibel_A7   = 21; // A7 bruges ikke lige nu

// Deklaration af variable til millis Timere --------------------------

unsigned long Input1_StartMillis = 0;     // Input1_timer
unsigned long Input1_AktuelMillis = 0;    // Input1_timer
unsigned long Input1_Interval = 0;        // Input1_timer

unsigned long Input2_StartMillis = 0;     // Input2_timer
unsigned long Input2_AktuelMillis = 0;    // Input2_timer
unsigned long Input2_Interval = 0;        // Input2_timer

unsigned long Input3_StartMillis = 0;     // Input3_timer
unsigned long Input3_AktuelMillis = 0;    // Input3_timer
unsigned long Input3_Interval = 0;        // Input3_timer

unsigned long MorseStartMillis = 0;       // morse-tegn timer
unsigned long MorseAktuelMillis = 0;      // morse-tegn timer
unsigned long MorseInterval = 0;          // morse-tegn timer
```

```
unsigned long SlutStartMillis = 0;        // slut timer
unsigned long SlutAktuelMillis = 0;       // slut timer
unsigned long SlutInterval = 0;           // slut timer

unsigned long MaxStartMillis = 0;         // max TX timer
unsigned long MaxAktuelMillis = 0;        // max TX timer
unsigned long MaxInterval = 0;            // max TX timer

unsigned long GenStartStartMillis = 0;   // GenStart timer11
unsigned long GenStartAktuelMillis = 0;  // GenStart timer1
unsigned long GenStartInterval = 0;      // GenStart timer1

unsigned long BlinkLastTime;              // a-live timer

int BlinkStatus;                          // a-live LED pin 13
int BlinkDelay = 1000;     // (milliseconds) a-live LED pin 13

// Deklaration af globale variable
-------------------------------------
boolean Input1_Ind = 0;        // Flag for input1 modtaget
boolean Input1_OK = 0;         // Flag for input1 debounced
boolean Input1_Drift = 0;      // Flag for input1 dobbeltdyt

boolean Input2_Ind = 0;        // Flag for input2 modtaget
boolean Input2_OK = 0;         // Flag for input2 debounced

boolean KoldStart_OK = 0;
boolean Rep_OK = 0;             // Flag til start af repeaterfunktioner
boolean TX_Stop = 0;            // Flag for max taletid
boolean Morse_OK = 0;           // Flag for morsegenerator

// Deklaration af konstanter
---------------------------------------------
const int MaxStep = 1000;        // Max_TX_tid step (milliseconds)
(1000)
const int Max_TX_tid = 300;      // Max_TX_tid i sec. default 5 minut
(300)
const int GenStartStep = 1000;   // GenStart_TX_tid step (millisec)
(1000)
const int GenStart_TX_tid = 3;   // Stille tid før GenStart (seconds)
 (3)
const int SendePause = 15;       // Pause for TimeOut reset (i 70 mSec)
(15)
const int SlutStep =70;          // Step i slutgenerator.
(70)

const int Input1_Step = 100;     // Input1_StartTimer step (millisec)
(100)
const int Input1_BounceTid = 2;  // Debounce Input1_ind (i 100 milliSec)
 (2)
const int Input1_StartTid =  10; // Krævet Input1 (i 100 mSec) koldstart
 10)

const int Input2_Step = 100;     // Debounce Input2_ Timer steps (msec)
(100)
const int Input2_BounceTid = 2;  // Debounce Input2__ind (i 100 mSec)
 (2)
const int Input2_StartTid =  10; // Krævet Input2 (i 100 mSec) koldstart
(10)
```

```
const int BipFrekv = 1000;       // LF frekvensen på Morsetegn (Hz)
(1000)
const int DytFrekv = 800;        // LF frekvensen på pause Dyt (Hz)
(800)
const int BotFrekv = 600;        // LF frekvensen på pause Bot (Hz)
(600)
const int MorseStep = 70;        // Morse step = dot længde (millisec)
(70)
const int DotTid = MorseStep;    // Længde af en prik
const int DashTid = 3 * DotTid;  // Længde af en streg

//
--------------------------------------------------------------------------
void setup()
{
// Serial.begin(9600); // sender data til konsol i IDE for debug
// Linjerne til seriel visning på PC skærm aller-nederst på siden.
// Serial kan ikke bruges under normal drift, indvirker på tiderne
(morse)
//
// Opsætning af pins som INPUT, INPUT_PULLUP eller OUTPUT.
---------------
  pinMode(input1, INPUT_PULLUP);      // LOW modtaget på pin A0. SQ fra
GM900
  pinMode(input2, INPUT_PULLUP);      // LOW modtaget på pin A1. SQ fra
TS3
  pinMode(input3, INPUT_PULLUP);      // LOW modtaget på pin A2. ledig

  pinMode(dobbelt_dyt, OUTPUT);       // LED viser seneste input kilde
  pinMode(tast_til_ts3, OUTPUT);      // Tast til TS3
  pinMode(tone_ud, OUTPUT);           // Morse og dyt, tone()
  pinMode(TimeOutLED, OUTPUT);        // LED viser timeout
  pinMode(tx_tast, OUTPUT);           // Tast af sender
  pinMode(blinkled, OUTPUT);          // Blink LED på Nano

  digitalWrite(TimeOutLED,LOW);       // sluk Timeout LED ved start
}

// ---------------------Her begynder funktioner-------------------------
void aLiveBlink() // blink med LED på NANO, KUN for at vise vi er i live
  // Blink 1000 msec i hvile. 250 msec under drift. Styres af Rep_OK.
 {
  if(millis()- BlinkLastTime >= BlinkDelay)
    {
    BlinkStatus=!BlinkStatus;
    digitalWrite(13,BlinkStatus);
    BlinkLastTime = millis();
    }
 }

// ----------------Input1_Start og Debounce
Timer-------------------------------
void Input1_StartTimer()
// Kører kun når (Input1_Ind == 1).
// Når Input1_BounceTid nås sættes Input1_OK til 1.
// Når Input1_StartTid nås sættes Rep_OK til 1.
// Resetter sig selv, hvis Input1_ forsvinder inden udgangstid opnås.
{
```

```
  Input1_StartMillis = millis();
  if ((Input1_StartMillis - Input1_AktuelMillis >= Input1_Step) &&
(Input1_Ind ==1))
   {
   Input1_Interval = Input1_Interval + 1;
   Input1_AktuelMillis = Input1_StartMillis;
   if ((Input1_Interval == Input1_BounceTid)&&(Input1_Ind == 1))
Input1_OK = 1;
   if ((Input1_Interval == Input1_StartTid)&&(Input1_Ind == 1))  Rep_OK
= 1;
   }
  if (Input1_Ind == 0)
  {
    Input1_OK = 0;
    Input1_Interval=0;
  }
}

// ----------------Input2_Start og Debounce
Timer--------------------------------
void Input2_StartTimer()
// Kører kun når (Input2_Ind == 1).
// Når Input2_BounceTid nås sættes Input2_OK til 1.
// Når Input2_StartTid nås sættes Rep_OK til 1.
// Resetter sig selv, hvis Input2_ forsvinder inden udgangstid opnås.
{
  Input2_StartMillis = millis();
  if ((Input2_StartMillis - Input2_AktuelMillis >= Input2_Step) &&
(Input2_Ind ==1))
   {
   Input2_Interval = Input2_Interval + 1;
   Input2_AktuelMillis = Input2_StartMillis;
   if ((Input2_Interval == Input2_BounceTid)&&(Input2_Ind == 1))
Input2_OK = 1;
   if ((Input2_Interval == Input2_StartTid)&&(Input2_Ind == 1))  Rep_OK
= 1;
   }
  if (Input2_Ind == 0)
  {
    Input2_OK = 0;
    Input2_Interval=0;
  }
}

// ----------------Morse
generator-----------------------------------------
void MorseGenerator()
// Kører kun når MORSE_OK = 1. Kører 1 gennemløb og resetter derefter
sig selv.
{
  MorseStartMillis = millis();
  if ((MorseStartMillis - MorseAktuelMillis >= MorseStep) && (Morse_OK
== 1))
    {
     MorseInterval = MorseInterval + 1;
     MorseAktuelMillis = MorseStartMillis;

    if (MorseInterval ==  1) tone(tone_ud, BipFrekv, DashTid);  // O
    if (MorseInterval ==  5) tone(tone_ud, BipFrekv, DashTid);
```

```
    if (MorseInterval ==  9) tone(tone_ud, BipFrekv, DashTid);

    if (MorseInterval == 15) tone(tone_ud, BipFrekv, DashTid);  // Z
    if (MorseInterval == 19) tone(tone_ud, BipFrekv, DashTid);
    if (MorseInterval == 23) tone(tone_ud, BipFrekv, DotTid);
    if (MorseInterval == 25) tone(tone_ud, BipFrekv, DotTid);

    if (MorseInterval == 29) tone(tone_ud, BipFrekv, DotTid);   // 4
    if (MorseInterval == 31) tone(tone_ud, BipFrekv, DotTid);
    if (MorseInterval == 33) tone(tone_ud, BipFrekv, DotTid);
    if (MorseInterval == 35) tone(tone_ud, BipFrekv, DotTid);
    if (MorseInterval == 37) tone(tone_ud, BipFrekv, DashTid);

    if (MorseInterval == 43) tone(tone_ud, BipFrekv, DotTid);   // R
    if (MorseInterval == 45) tone(tone_ud, BipFrekv, DashTid);
    if (MorseInterval == 49) tone(tone_ud, BipFrekv, DotTid);

    if (MorseInterval == 53) tone(tone_ud, BipFrekv, DotTid);   // E

    if (MorseInterval == 57) tone(tone_ud, BipFrekv, DashTid);  // D
    if (MorseInterval == 61) tone(tone_ud, BipFrekv, DotTid);
    if (MorseInterval == 63) tone(tone_ud, BipFrekv, DotTid);

    if (MorseInterval >= 70)
       {
       Morse_OK = 0;
       MorseInterval = 0;
       }
    }
}
//-------------------------------------------------------------------------
-------
void GenStart_Max_TX_timer() //
// Kører kun, når max taletid er overskredet. (TX_Stop = 1).
// Venter på at der er pause længere end GenStart_TX_tid og sætter så
TX_Stop=0.
// Derved kommer repeateren tilbage i normal drift.
{
  GenStartStartMillis = millis();
  if ((GenStartStartMillis-GenStartAktuelMillis >=GenStartStep)&&
     (TX_Stop == 1))
    {
    GenStartInterval = GenStartInterval + 1;
    GenStartAktuelMillis = GenStartStartMillis;

    if (GenStartInterval >= GenStart_TX_tid)
    {
      TX_Stop = 0;
      digitalWrite(TimeOutLED,LOW);
      GenStartInterval = 0;
    }
  }
}

//-------------------------------------------------------------------------
----
void Max_TX_timer()
// Kører når repeateren er i normal drift. Resettes når man har holdt en
pause
```

```
// Hvis taletid uden K når Max_TX_tid sættes TX_Stop=1, og repeateren
stopper.
// Lige før stop udsendes 9 toner med vekslende tonefrekvenser og 2 (2x)
bot.
// Hvis Max Time Out forårsages fra nettet sendes 2 dobbelt-bot.
// enkelt-dot hvis Input1_Drift == 0, dobbelt-dot hvis Input1_Drift ==
1.
{
  MaxStartMillis = millis();
  if ((MaxStartMillis - MaxAktuelMillis >= MaxStep)&&(TX_Stop == 0))
    {
    MaxInterval = MaxInterval + 1;
    MaxAktuelMillis = MaxStartMillis;
    if (MaxInterval == (Max_TX_tid +  0)) tone(tone_ud, BipFrekv,
DashTid);
    if (MaxInterval == (Max_TX_tid +  1)) tone(tone_ud, DytFrekv,
DashTid);
    if (MaxInterval == (Max_TX_tid +  2)) tone(tone_ud, BotFrekv,
DashTid);
    if (MaxInterval == (Max_TX_tid +  4)) tone(tone_ud, BipFrekv,
DashTid);
    if (MaxInterval == (Max_TX_tid +  5)) tone(tone_ud, DytFrekv,
DashTid);
    if (MaxInterval == (Max_TX_tid +  6)) tone(tone_ud, BotFrekv,
DashTid);

    if (MaxInterval == (Max_TX_tid + 8)) tone(tone_ud, BotFrekv,
DashTid);
    if ((MaxInterval == (Max_TX_tid +9))&(Input1_Drift == 1))
tone(tone_ud, BotFrekv, DashTid);

    if (MaxInterval == (Max_TX_tid + 11)) tone(tone_ud, BotFrekv,
DashTid);
    if ((MaxInterval == (Max_TX_tid +12))&(Input1_Drift == 1))
tone(tone_ud, BotFrekv, DashTid);

    if (MaxInterval >= (Max_TX_tid + 14))
      {
      TX_Stop = 1;
      digitalWrite(TimeOutLED,HIGH);
      MaxInterval = 0;
      }
   }
}
//------------------------------------------------------------
void SlutGenerator()
// Kører når repeateren er i gang, og der ikke modtages signal.
// Der udsendes 2 enkelt-dot eller 2 dobbelt-dot.
// enkelt-dot hvis Input1_Drift == 0, dobbelt-dot hvis Input1_Drift ==
1.
// Dyt fra ASR styringen er de-aktiveret.
// Funktionen nulstiller tilsidst diverse timere og slukker bærebølgen.
{
  SlutStartMillis = millis();
  if ((SlutStartMillis - SlutAktuelMillis >= SlutStep)&&(TX_Stop == 0)
        && (Input1_OK == 0) && (Input2_OK == 0) && (Morse_OK == 0))
    {
    SlutInterval = SlutInterval + 1;
    SlutAktuelMillis = SlutStartMillis;
```

```
    if ((SlutInterval== 10)&(Input1_Drift == 1)) tone(tone_ud, DytFrekv,
DotTid); //33
    if (SlutInterval == 13) tone(tone_ud, DytFrekv, DotTid); // 30

    if (SlutInterval == SendePause) MaxInterval = 0; // Reset
Max_TX_timer SendePause
//  reset sker når sendepause = 15 x 70 (1050 ms) er nået, dvs at dyt er
hørt ved 13.

    if ((SlutInterval== 50)&(Input1_Drift == 1)) tone(tone_ud, DytFrekv,
DotTid); //63
    if (SlutInterval == 53) tone(tone_ud, DytFrekv, DotTid); // 60

    if (SlutInterval == 90) Morse_OK = 1; // Morsegenerator startes
       // Timeren kører ikke mens der morses, fordi (Morse_OK == 1).
    if (SlutInterval >= 100)
      {
      Rep_OK = 0;
      Morse_OK = 0;

      SlutInterval = 0;
      MaxInterval = 0;
      }
   }
}
//-------------------------------------------------------------

//*************************************************************
void loop()
// HOVEDPROGRAM. Her står den kode, som kører i uendelig sløjfe.
// Alle funktioner bliver kaldt efter tur og udfører deres job.
// if-sætningerne udfører de nødvendige logiske funktioner.
{
  aLiveBlink(); // blinker med LED på Nano for at vise at programmet
kører.

  Input1_Ind = (!digitalRead(input1)); // input inverteres
  Input2_Ind = (!digitalRead(input2)); // input inverteres

  Input1_StartTimer(); // Kører når input1. Sætter Input1_OK=1 og
Rep_OK=1.
  Input2_StartTimer(); // Kører når input2. Sætter Input2_OK=1 og
Rep_OK=1.
  Max_TX_timer(); // Hvis tid uden pause større end Max_TX_tid sættes
TX_Stop=1
  SlutGenerator(); // Efter dyt og call sættes Rep_OK til 0. Repeater
sover.
  MorseGenerator(); // Kører når Morse_OK=1. Resetter sig selv efter 1
gennemløb.
  GenStart_Max_TX_timer(); // Efter TimeOUT og pause på GenStart_TX_tid:
TX_Stop=0

// Morsegenerator stop og nulstilling hvis der modtages input1 eller
input2 --
if ((Input2_OK==1)) MorseInterval = 0;
if ((Input2_OK==1)) Morse_OK = 0;

if ((Input1_OK==1)) MorseInterval = 0;
```

```
if ((Input1_OK==1)) Morse_OK = 0;

// KoldStart betingelser (for at give repeat straks ved signal ind)
-------------
  KoldStart_OK = 0;
  if ((Input1_Ind == 1)&&(TX_Stop == 0) && (Rep_OK == 0)) KoldStart_OK =
1;
  if ((Input2_Ind == 1)&&(TX_Stop == 0) && (Rep_OK == 0)) KoldStart_OK =
1;
  if (KoldStart_OK == 1) digitalWrite(tx_tast,HIGH); // Tast ON

// Drift betingelser ------------------------------------------------

  // Rep_OK sættes af input-timere, når input_starttid er opnået
  if ((Rep_OK == 1) && (TX_Stop == 0)) digitalWrite(tx_tast,HIGH); //
Tast ON


  if ((TX_Stop == 0) && (Input1_Ind == 1))
digitalWrite(tast_til_ts3,HIGH);
  // åbner tast_til_ts3
  if (Input1_Ind == 0) digitalWrite(tast_til_ts3,LOW); // lukker
tast_til_ts3

// GenStart betingelser (Nulstil ved signal fra input1 eller input2
-----------
  if ((Input2_OK == 1) || (Input1_OK  == 1)) //
  {
    GenStartInterval = 0; // nulstil GenStart 1 timer
    SlutInterval = 0; // nulstil SlutGenerator
  }

// Slut betingelser ----------------------------
  if ((Rep_OK == 0) && (KoldStart_OK == 0)) digitalWrite(tx_tast,LOW);
// TX OFF
  if (Rep_OK == 0) MaxInterval = 0; // nulstil MaxStart timer
  if (Rep_OK == 0) SlutInterval = 0; // nulstil SlutGenerator

// Stop betingelser
-----------------------------------------------------------
  if (TX_Stop == 1) digitalWrite(tx_tast,LOW); // TX blokeres pga
Max_TX_timer
  if (TX_Stop == 1) digitalWrite(tast_til_ts3,LOW); // TS3 blokeres pga
Max_TX
  if (TX_Stop == 1) MaxInterval = 0; // nulstil MaxStart timer
  if (TX_Stop == 1) SlutInterval = 0; // nulstil SlutGenerator

// A-Live blink på Arduino LED. Hurtig blink ved aktivitet.
  if (Rep_OK == 1) BlinkDelay = 250; else BlinkDelay = 1000; // BlinkLED
på Nano.

// if-sætninger herunder bruges kun til at skifte til pause dobbelt-dyt.
// Hvis der er input1 (Squelch) tændes LED og flag dobbelt_dyt
// Derved kan skelnes om senest modtagne transmission kom fra radio
eller TS3.

  if (Input1_OK == 1) Input1_Drift = 0; // squelchsignal fra radio
  if (Input2_OK == 1) Input1_Drift = 1; // tastsignal fra TS3
  if (Rep_OK == 0) Input1_Drift = 0; // nulstil Input1_Drift
```

```
  if (Input1_Drift == 0) digitalWrite(dobbelt_dyt,LOW);
  if (Input1_Drift == 1) digitalWrite(dobbelt_dyt,HIGH);
  if (Rep_OK == 0) digitalWrite(dobbelt_dyt,LOW);
// ---------------------------------------------------

// Efterfølgende kan bruges til dbug til monitor, og bør ud-blankes ved
drift
// Brug af monitor i denne mængde nedsætter blandt andet
morse-hastigheden
/*
  Serial.print(" | P-Ind "); Serial.print(Input1_Ind);
  Serial.print(" | P-int "); Serial.print(Input1_Interval);

  Serial.print(" | P-delay "); Serial.print(Input1_Delay);
  Serial.print(" | P-OK "); Serial.print(Input1_OK);
  Serial.print(" | Rep-OK "); Serial.print(Rep_OK);
  Serial.print(" | 1750 "); Serial.print(Input3_Ind);
  Serial.print(" | SQ "); Serial.print(Input2_OK);
  Serial.print(" | T-int "); Serial.print(Input3_Interval);
  Serial.print(" | SQ-PLT "); Serial.print(Input1_Drift);
  Serial.print(" | Rep "); Serial.print(Rep_OK);
  Serial.print(" | Max "); Serial.print(MaxInterval);
  Serial.print(" | Slut "); Serial.print(SlutInterval);
  Serial.print(" | Morse "); Serial.print(Morse_OK);
  Serial.print(" | M-int "); Serial.print(MorseInterval);
  Serial.print(" | Gen1 "); Serial.print(GenStartInterval);
  Serial.print(" | Gen2 "); Serial.print(GenStart2Interval);
  Serial.print(" | Stop "); Serial.print(TX_Stop);
  Serial.println();
*/
}
//*************************************************************
```