```
//*********************************************************************
****
// cqm5000_si5153_7segm_v.231
// Author  : OZ1LN
// Syntese til CQM5000 med Si5351A, 4 x 7segment display og CP5000
keyboard
// videreudvikling af VFO_Si5351A_0_31W fra 2026-02-27 med 2x16 LCD
display
//*********************************************************************
****
// HUSK AT RETTE VERSIONS NUMMER HERUNDER og i TITLEN HEROVER:
//
int ver1 = 2; int ver2 = 3; int ver3= 1;
int typ1 = 5; int typ2 = 1; int typ3 = 1; int typ4 = 2; // CQM5112
//
//*********************************************************************
****
// 2026-08-07 xtalvalg 10/25 MHz med knap 1 eller 6 ved koldstart
// 2026-08-07 udlæs xtalvalg, tjek om si5153 er tilstede.
// 2026-08-11 updateDisplay() flyttet ind i tastatur-rutine. version 213
// 2026-08-11 Forsøg med Scan()
// 2026-08-13 tilføjet TX_plus og SQinput i void Tjekinput

// 2026-10-22 tilpasset til nyt printlayout, display og tastatur
// 2026-10-24 tilrettet karakter-koder, segmentArray[]
// 2026_10_24 opstarts-status: xtal, programversion, radiotype,
si5351_fejl

// 2026-11-03 v 230 : Ændret til Si output 48 / 44 MHz output, pga at Si
ikke
// kan levere 2 frekvnser på PLLA højere end ca. 112 MHz. Nødvendigt for
at
// kunne lave 2 LO frekvenser med den ene inverteret til PushPull udgang.



//*********************************************************************
****

#include "si5351.h"  // hjælpebibliotek til Si5351
//Etherkit Si5351 by Jason Mildrum 2.2.0
#include "Wire.h"
Si5351 si5351;

#include <EEPROM.h>

// deklaration af pins ----------------------------------------
// OBS disse pin numre skal spejles top til bund, når Easy print
// tages i brug. Byttet i forhold til test version af styreprint
// -----------------------------------------------------------------

int rowPin1 = 2;  // D2
int rowPin2 = 3;  // D3
int rowPin3 = 4;  // D4
int rowPin4 = 5;  // D5
int colPin4 = 6;  // D6
int colPin3 = 7;  // D7
int colPin2 = 8;  // D8
int colPin1 = 9;  // D9
int TX_plus = 10; // D10
int SQinput = 11; // D11
int PtAFout = 12; // D12

int BlinkLED = 13;  // D13 pin 13 er LED på NANO

int clockPin = 15;  // A1
```

```
int dataPin  = 16;  // A2
int latchPin = 17;  // A3



unsigned long freq_min  = 144000000;            // laveste frekvens
144.000 MHz
unsigned long freq_max  = 146000000;            // højeste frekvens
146.000 MHz
unsigned long freq_valg = 145500000;            // Start frekvens 145,500
MHz
unsigned long freq_mf =    10700000;            // mellemfrekvens
10700000 Hz
unsigned long freq_dup =    600000;             // repeaterDUP  00600000
Hz
unsigned long freq_step1 =   12500;              // Kanalstep 12,5 kHz
unsigned long myNum; // bruges ved formattering af frekvens til 4 cifret
display
unsigned long freq_led;                         // Frekvens, som sendes
til formattering (Display)
unsigned long freq;                             // Bruges ved formattering
af frekvens til display
unsigned long freq_dds;                         // Frekvens som sendes til
Si5351
unsigned long last_freq_dds;                    // bruges til test om
opdatering af 5351

int a=0;

unsigned long xtalvalg = 10; // MHz. Vælges med knap 1 eller 6 under
opstart, 1 = 10, 6 = 25.
// krystal valg er kun krævet i testmoduler, til skift af
styreoscillatorfrekvens.

boolean DUP = 0;
boolean TX = 0;
boolean SQ = 0;

// boolean opdater = 1;  // Dette giver 1. opdatering af display
boolean last_TX = 0;
boolean last_DUP = 0;
boolean vis_si_fejl = 0; // Hvis SI5351 mangler sættes "1"
boolean vis_xtalvalg = 1;
boolean vis_version = 1;
boolean vis_radiotype = 1;
boolean scan_ON = 0;

int D0,D1,D2,D3,D4,D5,D6,D7,D8,D9; // display cifre til 7segment
// bruges i Format_freq

int x1; // Visning af xtalvalg 10 MHz
int x2; // Visning af xtalvalg 25 MHz

byte displayData[4];
byte displayArray[4];
byte segmentArray[34];
byte onLEDArray[4];

byte displayNr;

int diode_red = 2;
int diode_gul = 4;
int diode_green = 8;
int dioder_On = 0;   //diode_red + diode_gul + diode_green;

unsigned long BlinkLastTime;              // a-live timer
```

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

unsigned long debounceLastTime;              // updata timer
int debounce = 20;       // (milliseconds)  // debounce af knapper

unsigned long scan_last;
int scan_tid = 200;

// -------------- Opstartstatus begynd -------------------------
int vent1 = 1; int vent2 = 1; int vent3 = 1;

unsigned long vis_xtal_last;  // tider til opstarts statusvisning
int vis_xtal_tid = 500;

unsigned long vis_version_last;
int vis_version_tid = 1000;

unsigned long vis_radiotype_last;
int vis_radiotype_tid = 1500;

unsigned long vis_si_fejl_last;
int vis_si_fejl_tid = 2000; // skal være længere end vis_xtal_last
// -------------- Opstartstatus slut -------------------------

// ---------------------------------------------------------------|
int32_t cal_factor = -14000; //-14000 passer til 10MHz ny mini XO|
// Si5351 justeres til at give nøjagtig frekvens.                |
// Forøgelse vil forminske udgangsfrekvensen og omvendt. 139000  |
// ---------------------------------------------------------------|
//***************************************************************

void setup() {
//set pins to output so you can control the shift register
pinMode(latchPin, OUTPUT);
pinMode(clockPin, OUTPUT);
pinMode(dataPin, OUTPUT);

pinMode(rowPin1, OUTPUT);
pinMode(rowPin2, OUTPUT);
pinMode(rowPin3, OUTPUT);
pinMode(rowPin4, OUTPUT);
pinMode(colPin1, INPUT_PULLUP);
pinMode(colPin2, INPUT_PULLUP);
pinMode(colPin3, INPUT_PULLUP);
pinMode(colPin4, INPUT_PULLUP);

pinMode(TX_plus, INPUT_PULLUP);
pinMode(SQinput, INPUT_PULLUP);

// segmentArray e c a f dp d b g
//     a
//   f   b
//     g
//   e   c
//     d
// ---------------------------------------------------------------
// OBS nedenstående sement definitioner skal redigeres, når Easy print
tages
// i brug. Segment pins ændret i forhold til test version af styreprint
// ---------------------------------------------------------------
segmentArray[0]  = 0b11110110; // 0 OK
segmentArray[1]  = 0b01000010; // 1 OK
segmentArray[2]  = 0b10100111; // 2 OK
segmentArray[3]  = 0b01100111; // 3 OK
segmentArray[4]  = 0b01010011; // 4 OK
```

```
segmentArray[5]  = 0b01110101; // 5 OK
segmentArray[6]  = 0b11110101; // 6 OK
segmentArray[7]  = 0b01100010; // 7 OK
segmentArray[8]  = 0b11110111; // 8 OK
segmentArray[9]  = 0b01110111; // 9 OK
segmentArray[10] = 0b11110110; // A
segmentArray[11] = 0b11010110; // b
segmentArray[12] = 0b10000101; // C
segmentArray[13] = 0b11000111; // d OK
segmentArray[14] = 0b10110101; // E
segmentArray[15] = 0b10110001; // F
segmentArray[16] = 0b11010011; // H
segmentArray[17] = 0b01000000; // i
segmentArray[18] = 0b01000110; // J
segmentArray[19] = 0b10010100; // L
segmentArray[20] = 0b11000100; // n
segmentArray[21] = 0b11000101; // o
segmentArray[22] = 0b10110011; // P
segmentArray[23] = 0b10000001; // r
segmentArray[24] = 0b10010101; // t
segmentArray[25] = 0b11010110; // U
segmentArray[26] = 0b01010111; // Y
segmentArray[27] = 0b00000000; // TOM
segmentArray[28] = 0b00001000; // DecimalPunkt OK
segmentArray[29] = 0b11111111; // ALLE
segmentArray[30] = 0b00100000; // top streg
segmentArray[31] = 0b00000001; // midt streg OK
segmentArray[32] = 0b00000100; // bund streg
segmentArray[33] = 0b00100101; // alle streger

displayArray[0] = 0b10000000; // vælger 7segment blok
displayArray[1] = 0b01000000;
displayArray[2] = 0b00100000;
displayArray[3] = 0b00010000;

// ------------------------------------------------------------
// HENT xtalvalg i EEPROM adresse 0
xtalvalg = EEPROM.read(0);

// Hvis knap 1 eller knap 2 er trykket ved opstart sættes xtalvalg
digitalWrite(rowPin1,0);
if (!digitalRead(colPin1)) xtalvalg = 10; // Knap 1
if (!digitalRead(colPin2)) xtalvalg = 25; // Knap 2

if (xtalvalg == 10) { x1=1; x2=0; }// til visning i display
if (xtalvalg == 25) { x1=2; x2=5; }// til visning i display

unsigned long freq_xtal = xtalvalg * 1000000;  // Master xtal  MHz

// Hvis xtalvalg er forskellig fra EEPROM skrives ny værdi til EEPROM
adresse 0.
EEPROM.update(0, xtalvalg);
// ------------------------------------------------------------

  // ---------- Test af om Si5351 er til stede ----------------------
  bool i2c_found;
  i2c_found = si5351.init(SI5351_CRYSTAL_LOAD_8PF, freq_xtal, 0);
  if (!i2c_found) vis_si_fejl = 1;  // Laver udskrift i display;
  else {vis_si_fejl = 0;}
  // ------------------------------------------------------------

  si5351.init(SI5351_CRYSTAL_LOAD_8PF, freq_xtal, 0);    // freq_xtal
beregnes fra xtalgvalg, linje 5
  si5351.set_correction(cal_factor, SI5351_PLL_INPUT_XO); // Frkv
justering
  si5351.drive_strength(SI5351_CLK2, SI5351_DRIVE_2MA);  // out power.
```

```
Brug 2MA for mindre spur.

} // ************************ SLUT setup()
************************************************
// ----------------------------------------------------------
void beregn_frekvenser() {
  if ((TX == 0) and (DUP == 0))  //  Lyt på valgt frekvens
  {
    freq_dds = ((freq_valg - freq_mf) / 3);
    freq_led = (freq_valg);
  }

  if ((TX == 0) and (DUP == 1))  //  Lyt på indgangen
  {
    freq_dds = (freq_valg - freq_dup - freq_mf);
    freq_led = (freq_valg - freq_dup);
  }

  if ((TX == 1) and (DUP == 0))  //  Send på valgt frekvens
  {
    freq_dds = (freq_valg);
    freq_led = (freq_valg);
  }

  if ((TX == 1) and (DUP == 1))  // REPEATER, send på indgangen
  {
    freq_dds = (freq_valg - freq_dup);
    freq_led = (freq_valg);
  }
}
// ----------------------------------------------------------
void format_freq()
{
// OBS freq MÅ IKKE BEGYNDE MED NUL
// ********************************
// format op til 9 cifret frekvens til 4 stk 7-segmant display
// visning i display tilpasses med D1-D9 herunder

myNum = freq_led;

D0 = myNum % 10;

myNum = myNum/10;
D1 = myNum % 10;

myNum = myNum/10;
D2= myNum%10;

myNum = myNum/10;
D3 = myNum%10;

myNum = myNum/10;
D4 = myNum%10;

myNum = myNum/10;
D5 = myNum % 10;

myNum = myNum/10;
D6= myNum%10;

myNum = myNum/10;
D7 = myNum%10;

myNum = myNum/10;
D8 = myNum%10;
```

```
myNum = myNum/10;
D9 = myNum%10;

// visning i display tilpasses med D1-D9 herunder OBS ÆNDRET PGA NYT
PRINT
displayData[0] =  segmentArray[D3];
displayData[1] =  segmentArray[D4];
displayData[2] =  segmentArray[D5];
displayData[3] = (segmentArray[D6] +8); // plus 8 giver decimalpunktum;
}

//-----------------------------------------------
void updateDisplay() // Der sendes data til registre
// 1. byte vælger display blok 1-4 (1.nipple), 3 lysdioder og 1 output
(2.nipple)
// 2. byte vælger display visning i 7 segment, 8 bit fra segmentArray
 {
  int dioder_On = (diode_red + diode_gul + diode_green); //diode_red +
diode_gul + diode_green;
  for (displayNr = 0; displayNr < 4; displayNr ++)

    {
     digitalWrite(latchPin, LOW);
     shiftOut(dataPin, clockPin, MSBFIRST, ((displayArray[displayNr]) +
dioder_On));
     // digitalWrite(latchPin, HIGH); // Må ikke være aktive, fordi
     // digitalWrite(latchPin, LOW);  // det gør displaypulser smallere
     shiftOut(dataPin, clockPin, LSBFIRST, displayData[displayNr]);
     digitalWrite(latchPin, HIGH);
    }
 }
//-----------------------------------------------
void aLiveBlink() // blink med LED på NANO, KUN for at vise vi er i live
 {
  if(millis()- BlinkLastTime >= BlinkDelay)
    {
    BlinkStatus=!BlinkStatus;
    digitalWrite(BlinkLED,BlinkStatus);
    BlinkLastTime = millis();
    }
 }
//-----------------------------------------------
void tastatur()
{
// ---------- Første runde ----------
digitalWrite(rowPin1,0); // pin 1 LOW
digitalWrite(rowPin2,1);
digitalWrite(rowPin3,1);
digitalWrite(rowPin4,1);

if (!digitalRead(colPin1)) freq_valg = 144950000;  // kNAP 1  sæt
frekvens til 145,500 MHz
if (!digitalRead(colPin2)) freq_valg = 145312500;  // kNAP 2  sæt
frekvens til 145,500 MHz
if (!digitalRead(colPin3)) freq_valg = 145500000;  // kNAP 3  sæt
frekvens til 145,500 MHz
if (!digitalRead(colPin4)) freq_valg = 145550000;  // kNAP 4
updateDisplay();

//------------- Anden runde ----------
digitalWrite(rowPin1,1);
digitalWrite(rowPin2,0); // pin 2 LOW
digitalWrite(rowPin3,1);
digitalWrite(rowPin4,1);

if (!digitalRead(colPin1)) freq_valg = 145650000;  // kNAP 6
```

```
if (!digitalRead(colPin2)) freq_valg = 145725000;  // kNAP 7
if (!digitalRead(colPin3)) freq_valg = 145750000;  // KNAP 8
if (!digitalRead(colPin4)) freq_valg = 145775000;  // KNAP 9
updateDisplay();

//-------------- Tredje runde ----------
digitalWrite(rowPin1,1);
digitalWrite(rowPin2,1);
digitalWrite(rowPin3,0); // pin 3 LOW
digitalWrite(rowPin4,1);

if (!digitalRead(colPin1)) a = 3;  // Knap 12 (B)
if (!digitalRead(colPin2)) a = 8;  // Knap 13 (blank)
if (!digitalRead(colPin3)) {
  if (scan_ON == HIGH)            //   SCAN  KNAP 14  (T)
      scan_ON = LOW;
    else
      scan_ON = HIGH;
  }
if (!digitalRead(colPin4)) a = 9;

updateDisplay();

//---------------- Fjerde runde ----------
digitalWrite(rowPin1,1);
digitalWrite(rowPin2,1);
digitalWrite(rowPin3,1);
digitalWrite(rowPin4,0); // pin 4 LOW

if (!digitalRead(colPin1)) a = 4;
if (!digitalRead(colPin2)) a = 9;
if (!digitalRead(colPin3)) { // Knap
  if(millis()- debounceLastTime >= debounce)
    freq_valg = freq_valg + freq_step1;   //   OP  KNAP 10 (0)
    debounceLastTime = millis();
  }

if (!digitalRead(colPin4)) {
  if (millis()- debounceLastTime >= debounce)
    freq_valg = freq_valg - freq_step1;   //     NED  KNAp  #
    debounceLastTime = millis();
  }
updateDisplay();

//------------- Runder slut, nulstil rows
digitalWrite(rowPin1,1);
digitalWrite(rowPin2,1);
digitalWrite(rowPin3,1);
digitalWrite(rowPin4,1);
}
// ------------------------------------------------------------
void update_Si5351()
// data freq_dds * 100 sendes til Si5351
{
if (freq_dds != last_freq_dds) // Ny frekvens ? JA, opdatering
  {
    si5351.set_freq(freq_dds * 100ULL, SI5351_CLK2);

  last_freq_dds = freq_dds; // Ny frekvns er nu last_freq_dds
  }
}

// ------------------------------------------------------------
void Vis_xtalvalg()
{
   if ((vis_xtalvalg == 1))
```

```
 {
   if (millis() - vis_xtal_last >= vis_xtal_tid)
   {
    vis_xtalvalg = 0;
    vis_xtal_last = millis();
    vent1 = 0;
   }
   if (vis_xtalvalg == 1)
   {
    displayData[0] = segmentArray[31]; // -
    displayData[1] = segmentArray[x2]; // 1  2
    displayData[2] = segmentArray[x1]; // 0  5
    displayData[3] = segmentArray[31];  // -
   }
 }
}
// ---------------------------------------------------------------
void Vis_version()
{
 if ((vis_version == 1) & (vent1 == 0))
 {
   if (millis()- vis_version_last >= vis_version_tid) {
    vis_version = 0;
    vis_version_last = millis();
    vent1 = 0;  vent2 = 0;
   }
   if (vis_version == 1)
   {
  displayData[3] = segmentArray[22]; // P
  displayData[2] = (segmentArray[ver1] +8); //
  displayData[1] = segmentArray[ver2]; //
  displayData[0] = segmentArray[ver3];  // -
  }
 }
}
// ---------------------------------------------------------------
void Vis_radiotype()
 {
 if ((vis_radiotype == 1) & (vent1 == 0) & (vent2 == 0))
   {
   if (millis()- vis_radiotype_last >= vis_radiotype_tid)
    {
    vis_radiotype = 0;
    vis_radiotype_last = millis();
     vent1 = 0;  vent2 = 0; vent3 = 0;
    }
  displayData[3] = segmentArray[typ1]; // -
  displayData[2] = segmentArray[typ2]; //
  displayData[1] = segmentArray[typ3]; //
  displayData[0] = segmentArray[typ4];  // -
  }
 }
// ---------------------------------------------------------------
 void Vis_si_fejl()
{
  if ((vis_si_fejl == 1) & (vent1 == 0) & (vent2 == 0) & (vent3 == 0))
 {
   if (millis()- vis_si_fejl_last >= vis_si_fejl_tid)
    {
    vis_si_fejl = 0;
    vis_si_fejl_last = millis();
    vent1 = 0;
    }
   if (vis_si_fejl == 1)
  {
  displayData[0] = segmentArray[5];  // d
```

```
  displayData[1] = segmentArray[13]; // d
  displayData[2] = segmentArray[13]; // S
  displayData[3] = segmentArray[31]; // -
  }
 }

}
// ----------------------------------------------------------------
void Scan()
{
  if (scan_ON) {
   if (millis() - scan_last >= scan_tid) {

   freq_valg = freq_valg + freq_step1;

   if (freq_valg == 146000000) freq_valg = 144000000;

   // if (freq_valg <= freq_max); freq_valg = freq_min;
    scan_last = millis();
   }
  }
}
// ----------------------------------------------------------------
void Tjek_input()
{
if (!digitalRead(TX_plus)) TX = 1; else TX = 0;

if (!digitalRead(SQinput)) SQ = 1; else SQ = 0;

}
//****************************************************************
// HOVEDPROGRAM. Her står den kode, som kører i uendelig sløjfe.
//****************************************************************
void loop() {

  aLiveBlink();

  Vis_xtalvalg();
  Vis_si_fejl();
  Vis_version();
  Vis_radiotype();

  tastatur();
  beregn_frekvenser();
  format_freq();
  update_Si5351();
  Tjek_input();

  Scan();

  //updateDisplay(); // flyttet ind i tastatur rutine 20260811 213
 }
//-------------------------------------------------
```