```
//*************************************************************************
// cqm5000_si5153_7segm_v.218
// 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 =1; int ver3= 8;
//
//*************************************************************************
// 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

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

#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 SQinput = 10; // D10
int TX_plus = 11; // D11

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;
```

```
70  boolean SQ = 0;
71
72  // boolean opdater = 1;  // Dette giver 1. opdatering af display
73  boolean last_TX = 0;
74  boolean last_DUP = 0;
75  boolean Device_not_found = 0; // Hvis SI5351 mangler sættes "1"
76  boolean vis_xtalvalg = 1;
77  boolean scan_ON = 0;
78
79  int D0,D1,D2,D3,D4,D5,D6,D7,D8,D9; // display cifre til 7segment
80
81  int cif1 = 27; // Blank
82  int cif2 = 27;//1;
83  int cif3 = 27;//4;
84  int cif4 = 27;//5;
85
86  int x1; // Visning af xtalvalg
87  int x2; // Visning af xtalvalg
88
89  byte displayData[4];
90  byte displayArray[4];
91  byte segmentArray[34];
92  byte onLEDArray[4];
93
94  byte displayNr;
95
96  int diode_red = 2;
97  int diode_gul = 4;
98  int diode_green = 8;
99  int dioder_On = 0; //diode_red + diode_gul + diode_green;
100
101 unsigned long BlinkLastTime;              // a-live timer
102 int BlinkStatus;                          // a-live LED pin 13
103 int BlinkDelay = 1000;     // (milliseconds) a-live LED pin 13
104
105 unsigned long updateLastTime;              // updata timer
106 int updateStatus;                          // update LED pin 13
107 int updateDelay = 0;    // (milliseconds)  // update tid
108
109 unsigned long debounceLastTime;              // updata timer
110 int debounce = 10;       // (milliseconds)  // debounce af knapper
111
112 unsigned long vis_si_fejl_last;
113 int vis_si_fejl_tid = 1000;
114
115 unsigned long vis_xtal_last;
116 int vis_xtal_tid = 1000;
117
118 unsigned long scan_last;
119 int scan_tid = 500;
120
121 // ------------------------------------------------------------|
122 int32_t cal_factor = 130000;  // 130000 25 MHz  -9000 XO iLN
123 // Si5351 justeres til at give nøjagtig frekvens.              |
124 // Forøgelse vil forminske udgangsfrekvensen og omvendt. 139000 |
125 // ------------------------------------------------------------|
126 //*************************************************************
127 void setup() {
128 //set pins to output so you can control the shift register
129 pinMode(latchPin, OUTPUT);
130 pinMode(clockPin, OUTPUT);
131 pinMode(dataPin, OUTPUT);
132 //set pins to 4x4 keyboard
133 pinMode(rowPin1, OUTPUT);
134 pinMode(rowPin2, OUTPUT);
135 pinMode(rowPin3, OUTPUT);
136 pinMode(rowPin4, OUTPUT);
137 pinMode(colPin1, INPUT_PULLUP);
138 pinMode(colPin2, INPUT_PULLUP);
139 pinMode(colPin3, INPUT_PULLUP);
140 pinMode(colPin4, INPUT_PULLUP);
141
142 pinMode(TX_plus, INPUT_PULLUP);
```

```
143   pinMode(SQinput, INPUT_PULLUP);
144
145   // segmentArray f  g  a  b  c  d  e  dp
146   //     a
147   //   f   b
148   //     g
149   //   e   c
150   //     d
151   // ---------------------------------------------------------------
152   // OBS nedenstående sement definitioner skal redigeres, når Easy print tages
153   // i brug. Segment pins ændret i forhold til test version af styreprint
154   // ---------------------------------------------------------------
155   segmentArray[0]  = 0b10111110; // 0
156   segmentArray[1]  = 0b00011000; // 1
157   segmentArray[2]  = 0b01110110; // 2
158   segmentArray[3]  = 0b01111100; // 3
159   segmentArray[4]  = 0b11011000; // 4
160   segmentArray[5]  = 0b11101100; // 5
161   segmentArray[6]  = 0b11101110; // 6
162   segmentArray[7]  = 0b00111000; // 7
163   segmentArray[8]  = 0b11111110; // 8
164   segmentArray[9]  = 0b11111100; // 9
165   segmentArray[10] = 0b11111010; // A
166   segmentArray[11] = 0b11001110; // b
167   segmentArray[12] = 0b10100110; // C
168   segmentArray[13] = 0b01011110; // d
169   segmentArray[14] = 0b11100110; // E
170   segmentArray[15] = 0b11100010; // F
171   segmentArray[16] = 0b11011010; // H
172   segmentArray[17] = 0b00001000; // i
173   segmentArray[18] = 0b00011100; // J
174   segmentArray[19] = 0b10000110; // L
175   segmentArray[20] = 0b01001010; // n
176   segmentArray[21] = 0b01001110; // o
177   segmentArray[22] = 0b11110010; // P
178   segmentArray[23] = 0b01000010; // r
179   segmentArray[24] = 0b11000110; // t
180   segmentArray[25] = 0b10011110; // U
181   segmentArray[26] = 0b11011100; // Y
182   segmentArray[27] = 0b00000000; // TOM
183   segmentArray[28] = 0b00000001; // DecimalPunkt
184   segmentArray[29] = 0b11111111; // ALLE
185   segmentArray[30] = 0b00100000; // top streg
186   segmentArray[31] = 0b01000000; // midt streg
187   segmentArray[32] = 0b00000100; // bund streg
188   segmentArray[33] = 0b01100100; // alle streger
189
190   displayArray[0] = 0b10000000;
191   displayArray[1] = 0b01000000;
192   displayArray[2] = 0b00100000;
193   displayArray[3] = 0b00010000;
194
195   displayData[0] = segmentArray[cif1];
196   displayData[1] = segmentArray[cif2];
197   displayData[2] = segmentArray[cif3]; // plus 1 giver decimalpunktum
198   displayData[3] = segmentArray[cif4];
199   // ---------------------------------------------------------------
200   // HENT xtalvalg i EEPROM adresse 0
201   xtalvalg = EEPROM.read(0);
202
203   // Hvis knap 1 eller knap 6 er trykket ved opstart sættes xtalvalg
204   digitalWrite(rowPin1,0);
205   if (!digitalRead(colPin1)) xtalvalg = 10; // Knap 1
206   if (!digitalRead(colPin2)) xtalvalg = 25; // Knap 6
207
208   if (xtalvalg == 10) { x1=1; x2=0; }// til visning i display
209   if (xtalvalg == 25) { x1=2; x2=5; }// til visning i display
210
211   unsigned long freq_xtal = xtalvalg * 1000000;  // Master xtal  MHz
212
213   // Hvis xtalvalg er forskellig fra EEPROM skrives ny værdi til EEPROM adresse 0.
214   EEPROM.update(0, xtalvalg);
215   // ---------------------------------------------------------------
```

```
216
217     // ---------- Test af om Si5351 er til stede ----------------------
218     bool i2c_found;
219     i2c_found = si5351.init(SI5351_CRYSTAL_LOAD_8PF, freq_xtal, 0);
220     if (!i2c_found) { Device_not_found = 1;}// Laver udskrift i display;
221     else { Device_not_found = 0;}
222     // ----------------------------------------------------------------
223
224     si5351.init(SI5351_CRYSTAL_LOAD_8PF, freq_xtal, 0);    // freq_xtal beregnes fra
        xtalgvalg, linje 5
225     si5351.set_correction(cal_factor, SI5351_PLL_INPUT_XO); // Frkv justering
226     si5351.drive_strength(SI5351_CLK2, SI5351_DRIVE_8MA);  // out power. Brug 2MA for
        mindre spur.
227   } // SLUT setup()
228   // ---------------------------------------------------------
229   void beregn_frekvenser() {
230     if ((TX == 0) and (DUP == 0))  //  Lyt på valgt frekvens
231     {
232       freq_dds = (freq_valg - freq_mf);
233       freq_led = (freq_valg);
234     }
235
236     if ((TX == 0) and (DUP == 1))  //  Lyt på indgangen
237     {
238       freq_dds = (freq_valg - freq_dup - freq_mf);
239       freq_led = (freq_valg - freq_dup);
240     }
241
242     if ((TX == 1) and (DUP == 0))  //  Send på valgt frekvens
243     {
244       freq_dds = (freq_valg);
245       freq_led = (freq_valg);
246     }
247
248     if ((TX == 1) and (DUP == 1))  // REPEATER, send på indgangen
249     {
250       freq_dds = (freq_valg - freq_dup);
251       freq_led = (freq_valg);
252     }
253   }
254   // ---------------------------------------------------------
255   void format_freq()
256   {
257   // OBS freq MÅ IKKE BEGYNDE MED NUL
258   // ********************************
259   // format op til 9 cifret frekvens til 4 stk 7-segmant display
260   // visning i display tilpasses med D1-D9 herunder
261
262   myNum = freq_led;
263
264   D0 = myNum % 10;
265
266   myNum = myNum/10;
267   D1 = myNum % 10;
268
269   myNum = myNum/10;
270   D2= myNum%10;
271
272   myNum = myNum/10;
273   D3 = myNum%10;
274
275   myNum = myNum/10;
276   D4 = myNum%10;
277
278   myNum = myNum/10;
279   D5 = myNum % 10;
280
281   myNum = myNum/10;
282   D6= myNum%10;
283
284   myNum = myNum/10;
285   D7 = myNum%10;
286
```

```
287   myNum = myNum/10;
288   D8 = myNum%10;
289
290   myNum = myNum/10;
291   D9 = myNum%10;
292
293   // visning i display tilpasses med D1-D9 herunder
294   displayData[0] = (segmentArray[D6] + 1);
295   displayData[1] =  segmentArray[D5];
296   displayData[2] =  segmentArray[D4]; // plus 1 giver decimalpunktum
297   displayData[3] =  segmentArray[D3];
298   }
299
300   //-----------------------------------------------------
301   void updateDisplay() // Der sendes data til registre
302   // 1. byte vælger display blok 1-4 (1.nipple), 3 lysdioder og 1 output (2.nipple)
303   // 2. byte vælger display visning i 7 segment, 8 bit fra segmentArray
304    {
305     int dioder_On = (diode_red + diode_gul + diode_green); //diode_red + diode_gul +
       diode_green;
306     for (displayNr = 0; displayNr < 4; displayNr ++)
307
308       {
309        digitalWrite(latchPin, LOW);
310        shiftOut(dataPin, clockPin, MSBFIRST, ((displayArray[displayNr]) + dioder_On));
311        // digitalWrite(latchPin, HIGH); // Må ikke være aktive, fordi
312        // digitalWrite(latchPin, LOW);  // det gør displaypulser smallere
313        shiftOut(dataPin, clockPin, LSBFIRST, displayData[displayNr]);
314        digitalWrite(latchPin, HIGH);
315       }
316    }
317   //-----------------------------------------------------
318   void aLiveBlink() // blink med LED på NANO, KUN for at vise vi er i live
319    {
320     if(millis()- BlinkLastTime >= BlinkDelay)
321       {
322       BlinkStatus=!BlinkStatus;
323       digitalWrite(BlinkLED,BlinkStatus);
324       BlinkLastTime = millis();
325       }
326    }
327   //-----------------------------------------------------
328   void tastatur()
329   {
330   digitalWrite(rowPin1,0); // pin 1 LOW
331   digitalWrite(rowPin2,1);
332   digitalWrite(rowPin3,1);
333   digitalWrite(rowPin4,1);
334
335   if (!digitalRead(colPin1)) a = 1;
336   if (!digitalRead(colPin2)) a = 6;
337   if (!digitalRead(colPin3)) freq_valg = 145500000;  // sæt frekvens til 145,500 MHz
338   //if (!digitalRead(colPin4)) a = 0;
339   updateDisplay();
340
341   //----------------------
342   digitalWrite(rowPin1,1);
343   digitalWrite(rowPin2,0); // pin 2 LOW
344   digitalWrite(rowPin3,1);
345   digitalWrite(rowPin4,1);
346
347   if (!digitalRead(colPin1)) a = 2;
348   if (!digitalRead(colPin2)) a = 7;
349   if (!digitalRead(colPin3)) freq_valg = 145550000;  // sæt frekvens til 145,500 MHz
350   if (!digitalRead(colPin4)) a = 5;
351   updateDisplay();
352
353   //----------------------
354   digitalWrite(rowPin1,1);
355   digitalWrite(rowPin2,1);
356   digitalWrite(rowPin3,0); // pin 3 LOW
357   digitalWrite(rowPin4,1);
358
```

```
359   if (!digitalRead(colPin1)) a = 3;
360   if (!digitalRead(colPin2)) a = 8;
361   if (!digitalRead(colPin3)) {
362   if (scan_ON == HIGH)
363         scan_ON = LOW;
364       else
365         scan_ON = HIGH;
366   }
367   if (!digitalRead(colPin4)) {
368     if(millis()- debounceLastTime >= debounce)
369       freq_valg = freq_valg + freq_step1;
370       debounceLastTime = millis();
371     }
372   updateDisplay();
373
374   //----------------------
375   digitalWrite(rowPin1,1);
376   digitalWrite(rowPin2,1);
377   digitalWrite(rowPin3,1);
378   digitalWrite(rowPin4,0); // pin 4 LOW
379
380   if (!digitalRead(colPin1)) a = 4;
381   if (!digitalRead(colPin2)) a = 9;
382   if (!digitalRead(colPin3)) freq_valg = 144950000;  // sæt frekvens til 145,500 MHz
383   if (!digitalRead(colPin4)) {
384     if (millis()- debounceLastTime >= debounce)
385       freq_valg = freq_valg - freq_step1;
386       debounceLastTime = millis();
387     }
388   updateDisplay();
389
390   //----------------------
391   digitalWrite(rowPin1,1);
392   digitalWrite(rowPin2,1);
393   digitalWrite(rowPin3,1);
394   digitalWrite(rowPin4,1);
395   }
396   // ---------------------------------------------------------------
397   void update_Si5351()
398   // data freq_dds * 100 sendes til Si5351
399   {
400   if (freq_dds != last_freq_dds) // Ny frekvens ? JA, opdatering
401     {
402       si5351.set_freq(freq_dds * 100ULL, SI5351_CLK2);
403
404     last_freq_dds = freq_dds; // Ny frekvns er nu last_freq_dds
405     }
406   }
407   // ---------------------------------------------------------------
408   void Tjek_si_fejl()
409   {
410    if (millis()- vis_si_fejl_last >= vis_si_fejl_tid) {
411       Device_not_found = 0;
412       vis_si_fejl_last = millis();
413     }
414     if (Device_not_found == 1){
415        displayData[0] = segmentArray[31]; // -
416        displayData[1] = segmentArray[13]; // d
417        displayData[2] = segmentArray[13]; // d
418        displayData[3] = segmentArray[5];  // S
419      }
420   }
421   // ---------------------------------------------------------------
422   void Tjek_xtalvalg()
423   {
424      if (millis()- vis_xtal_last >= vis_xtal_tid) {
425       vis_xtalvalg = 0;
426       vis_xtal_last = millis();
427     }
428     if (vis_xtalvalg == 1){
429        displayData[0] = segmentArray[31]; // -
430        displayData[1] = segmentArray[x1]; // 1  2
431        displayData[2] = segmentArray[x2]; // 0  5
```

```
      displayData[3] = segmentArray[31];  // -
    }
}
// ---------------------------------------------------------------
void Scan()
{
  if (scan_ON) {
   if (millis() - scan_last >= scan_tid) {
   freq_valg = freq_valg + freq_step1;
   // 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();
  tastatur();
  beregn_frekvenser();
  format_freq();
  update_Si5351();
  Tjek_input();
  Tjek_xtalvalg();
  Tjek_si_fejl();
  Scan();

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