```
/*-------------------------------------------------------------
   VFO System for ESP32-DevKitC  Ver 1.00
       by T.Uebo / JF3HZB
         Feb. 10, 2019
-------------------------------------------------------------*/

/*------Hard ware Configuration --------------------
<<ESP32-DevKitC>>
pin No.  Connection
  16 :  Rotary Encoder A
  17 :  Rotary Encoder B

  22 :  si5351A SCL
  21 :  si5351A SDA

  18 :  SCK    / ST7735,SEPS525(128x160 display)
  23 :  MOSI   / ST7735,SEPS525(128x160 display)
   5 :  CS     / ST7735,SEPS525(128x160 display)
   2 :  DC(A0) / ST7735,SEPS525(128x160 display)
  15 :  RESET  / ST7735,SEPS525(128x160 display)

<<si5351A>>
CLK0 : Car Signal (I)
CLK1 : Car Signal (Q)
CLK2 : Lo Signal
-----------------------------------------------*/

// Libraries
#include <EEPROM.h>  // https://github.com/espressif/arduino-esp32/tree/master/libraries/EEPROM
//#include <WiFi.h>
//#include <ArduinoOTA.h>
//const char* ssid = "wireless";
//const char* password = "oldmanbernies1942";
/*-------------------------------------------------
```

```
   Frequency settings
-----------------------------------------------------*/
  #define init_freq  14230000    // Initial Frequncy[Hz]

  int32_t offset_frq = 0;    // Offset Frequency[Hz]
  int32_t    car_frq = 0;    // Carrier Frequncy[Hz]
  unsigned char f_carON = 1; // ON/OFF Car signal
  int32_t  freq_step;
  float readParam;

#define fmax 160000000       // Max frequency[160 MHz]
#define fmin   100000        // Min frequency[100 KHz]
#define EEPROM_SIZE 20

/
*-------------------------------------------------------------------------
----
    Control flags
-------------------------------------------------------------------------
---*/
uint8_t f_fchange;  // if frequency changed, set this flag to 1
uint8_t f_cchange;  // if Car frequency and/or "f_carON" changed,
set this flag to 1
uint8_t f_dchange;  // if need to renew display, set this flag to 1

/*-----------------------------------------------------
   pin assign
-----------------------------------------------------*/
//#define LED_BUILTIN 13

//-------------------------------------------------------------------------
#define NAME "VFO System"
#define VERSION "Ver. 1.00"
#define ID "by JF3HZB"
```

```
#include "driver/pcnt.h"
#include "display.h"
#include "graph.h"
#include "dial.h"
#include "si5351.h"

/*---------------------------------------------------------------------------
 *       Global
----------------------------------------------------------------------------*/
float dacc=0.0;
float Maxdacc=500.0;
long frq=init_freq;
int16_t RE_Count = 0;
uint8_t f_redraw;
extern char f_rev;
extern uint32_t cl_BG;
int press;
int cnt=2;
int mem, mem_old;
int cnt_old;
long step;
int address;
int PULSE_INPUT_PIN = 13; // Rotaty Encoder A  //17,0
int PULSE_CTRL_PIN  = 12; // Rotaty Encoder B  //16

/*---------------------------------------------------------------------------
       Timer ISR
----------------------------------------------------------------------------*/
hw_timer_t * timer = NULL;
void IRAM_ATTR onTimer(){}

//--------------------------------------------------------------------------
----------------
void loop() {    // (core1)
//--------------------------------------------------------------------------
```

```
----------------
 // ArduinoOTA.handle();



  press = digitalRead(27);
  if(press==LOW)
  {
  cnt = cnt-1;
  if(cnt<0) { cnt=0 ; }
  Serial.println(cnt);
  delay(300);
  }

 press = digitalRead(26);
  if(press==LOW)
  {
  cnt = cnt+1;
  if(cnt>5) { cnt=5 ; }
  Serial.println(cnt);
  delay(300);
  }

 press = digitalRead(25);
  if(press==LOW)
  {
  mem = mem+1;
  if(mem>9) { mem=0 ; }
//  Serial.println(mem);
  delay(300);
  }

  if(cnt==0) { step=10; freq_step=2.5;         }
  else
  if(cnt==1)  { step=100; freq_step=25;        }
```

```
  else
  if(cnt==2) { step=1000; freq_step=250;       }
  else
  if(cnt==3)  { step=10000; freq_step=2500;    }
  else
  if(cnt==4) { step=100000; freq_step=25000;   }
  else
  if(cnt==5) { step=1000000; freq_step=250000; }

  if(mem!=mem_old)
  {
  if(mem==0) { frq=init_freq;      }
  else
  if(mem==1) { frq=3730000;   }
  else
  if(mem==2) { frq=3853100;   }
  else
  if(mem==3) { frq=5450000;   }
  else
  if(mem==4) { frq=5505000;   }
  else
  if(mem==5) { frq=6070000;   }
  else
  if(mem==6) { frq=7130000;   }
  else
  if(mem==7) { frq=7878100;   }
  else
  if(mem==8) { frq=8957000;   }
  else
  if(mem==9) { frq=10100000;  }

  set_freq( frq + offset_frq );  // akt wenn frq nicht von Si5351
uebernommen

//  Serial.println(mem);
```

```
//  Serial.println(frq);
  }

      if(frq<2000000) {digitalWrite(4,HIGH); }
  else if(frq>=2000000) {digitalWrite(4,LOW); }

 press = digitalRead(32);
  if(press==LOW)
  {
  EEPROM.begin(EEPROM_SIZE);

  // Write data into eeprom
  address = 0;
  int boardId = 18;
  EEPROM.write(address, boardId);    //EEPROM.put(address,
boardId);
  address += sizeof(boardId);        //update address value
  double param = frq;
  Serial.print("Write to  EEPROM = ");
  Serial.println(param,0);
  EEPROM.writeFloat(address, param); //EEPROM.put(address,
param);
  EEPROM.commit();
  delay(300);
  }

    char str[64];
    if(f_dchange==1||cnt!=cnt_old||mem!=mem_old)
    {
       f_dchange=0;
       //GRAM_clr();
       boxfill(0,0,Nx-1,Ny-1,cl_BG);

       //Display Dial
       Dial(frq);
```

```
        box(7,100,153,126, 0xa0a0a0);
        box(6,99,154,127, 0xa0a0a0);

  if(cnt==0) { sprintf(str, "STEPS 10 Hz"); disp_str8(str,50, 85,
0xffd080); }
  else
  if(cnt==1) { sprintf(str, "STEPS 100 Hz"); disp_str8(str,50, 85,
0xffd080); }
  else
  if(cnt==2) { sprintf(str, "STEPS 1 KHz" ); disp_str8(str,50, 85,
0xffd080); }
  else
  if(cnt==3) { sprintf(str, "STEPS 10 KHz" ); disp_str8(str,50, 85,
0xffd080); }
  else
  if(cnt==4) { sprintf(str, "STEPS 100 KHz" ); disp_str8(str,50, 85,
0xffd080); }
  else
  if(cnt==5) { sprintf(str, "STEPS 1 MHz" ); disp_str8(str,50, 85,
0xffd080); }


  if(mem==0) { sprintf(str, "VFO"); disp_str8(str,5, 85,
0x00ffff); }
  else
  if(mem==1) { sprintf(str, "M1");  disp_str8(str,5, 85,
0xffd080); }
  else
  if(mem==2) { sprintf(str, "M2" ); disp_str8(str,5, 85,
0xffd080); }
  else
  if(mem==3) { sprintf(str, "M3" ); disp_str8(str,5, 85,
0xffd080); }
  else
  if(mem==4) { sprintf(str, "M4" ); disp_str8(str,5, 85,
```

```
0xffd080); }
  else
  if(mem==5) { sprintf(str, "M5" ); disp_str8(str,5, 85,
0xffd080); }
  else
  if(mem==6) { sprintf(str, "M6" ); disp_str8(str,5, 85,
0xffd080); }
  else
  if(mem==7) { sprintf(str, "M7" ); disp_str8(str,5, 85,
0xffd080); }
  else
  if(mem==8) { sprintf(str, "M8" ); disp_str8(str,5, 85,
0xffd080); }
  else
  if(mem==9) { sprintf(str, "M9" ); disp_str8(str,5, 85,
0xffd080); }

        //-------- Display Digital Frquency
-------------------------------------
        sprintf(str, "%3d.%03d,%02d",  frq/1000000,
(frq/1000)%1000, (frq/10)%100 );
        disp_str16(str,17, 105, 0xffd080);
        sprintf(str, "MHz" );

        disp_str12(str,120, 106, 0xffd080);
        if(f_redraw==0){
          trans65k();
          f_redraw=1;
        cnt_old=cnt;
        mem_old=mem;
        }
    }
    //digitalWrite(LED_BUILTIN,
1^digitalRead(LED_BUILTIN) ); //  Toggle LED
}
```

```
/
*-------------------------------------------------------------------------
---------------
      Alternative Loop (core0)
-------------------------------------------------------------------------
---------------*/
void task0(void* arg)
{
    while (1)
    {
       pcnt_get_counter_value(PCNT_UNIT_0, &RE_Count);
       int count=RE_Count;
       pcnt_counter_clear(PCNT_UNIT_0);
       if(f_rev==1) count=-count;
       if(count!=0){
          f_dchange=1;
          f_fchange=1;

          frq+= count * freq_step;
          if(frq>fmax) frq=fmax;
          if(frq<fmin) frq=fmin;
       }

       //-----------------------------------
       if(f_fchange==1){
          f_fchange=0;
          // Output Lo freq
          set_freq( frq + offset_frq );
       }

       //-----------------------------------
       if(f_cchange==1){
          f_cchange=0;
          // Output Car freq
```

```
        set_car_freq(car_frq, f_carON, 0);
      }

      if(f_redraw==1){
        Transfer_Image();
        f_redraw=0;
      }
      delay(1);
    }
}

//-------------------------------------------------------------------------
----------------
void setup() {
//-------------------------------------------------------------------------
----------------

  //  WiFi.begin(ssid, password);
 //ArduinoOTA.setHostname("test");
 //ArduinoOTA.setPassword("1234");
 //ArduinoOTA.begin();




   Serial.begin(9600);
   Serial.println("Start");
   delay(100);
   pinMode(25,INPUT_PULLUP);
   pinMode(26,INPUT_PULLUP);
   pinMode(27,INPUT_PULLUP);
   pinMode(32,INPUT_PULLUP);
   pinMode(4,OUTPUT);
   pinMode(19,INPUT_PULLUP);
```

```
// Init EEPROM
  EEPROM.begin(EEPROM_SIZE);

// Read data from eeprom
  address = 0;
  int readId;
  readId = EEPROM.read(address);  //
EEPROM.get(address,readId);
  address += sizeof(readId);      // update address value
  EEPROM.get(address, readParam); //
readParam=EEPROM.readFloat(address);
  Serial.print("Read from EEPROM = ");
  Serial.println(readParam,0);
  EEPROM.end();
  if(readParam!=0) { frq=readParam; }

    char str[64];
    //--------- create tasks on core0 ------------------------------
    xTaskCreatePinnedToCore(task0, "Task0", 4096, NULL, 1,
NULL, 0);

    //--------- Set up Interrupt Timer -----------------------------
    timer = timerBegin(0, 80, true); //use Timer0, div80 for 1us
clock
    timerAttachInterrupt(timer, &onTimer, true);
    timerAlarmWrite(timer, 10000, true); // T=10000us
    timerAlarmEnable(timer); // Start Timer

    //pinMode(LED_BUILTIN, OUTPUT);   //LED

    bool direct = digitalRead(19);
        if(direct==1) { PULSE_INPUT_PIN=12 ;
PULSE_CTRL_PIN=13 ; }
    else if(direct==0) { PULSE_INPUT_PIN=12 ;
PULSE_CTRL_PIN=13 ; }
```

```
//    Serial.println(direct);
//    Serial.println(PULSE_INPUT_PIN);
//    Serial.println(PULSE_CTRL_PIN);

    //--- Counter setup for Rotary Encoder ----------------------
    pcnt_config_t pcnt_config_A;// structure for A
    pcnt_config_t pcnt_config_B;// structure for B
    //
    pcnt_config_A.pulse_gpio_num = PULSE_INPUT_PIN;
    pcnt_config_A.ctrl_gpio_num = PULSE_CTRL_PIN;
    pcnt_config_A.lctrl_mode = PCNT_MODE_REVERSE;
    pcnt_config_A.hctrl_mode = PCNT_MODE_KEEP;
    pcnt_config_A.channel = PCNT_CHANNEL_0;
    pcnt_config_A.unit = PCNT_UNIT_0;
    pcnt_config_A.pos_mode = PCNT_COUNT_INC;
    pcnt_config_A.neg_mode = PCNT_COUNT_DEC;
    pcnt_config_A.counter_h_lim = 10000;
    pcnt_config_A.counter_l_lim = -10000;
    //
    pcnt_config_B.pulse_gpio_num = PULSE_CTRL_PIN;
    pcnt_config_B.ctrl_gpio_num = PULSE_INPUT_PIN;
    pcnt_config_B.lctrl_mode = PCNT_MODE_KEEP;
    pcnt_config_B.hctrl_mode = PCNT_MODE_REVERSE;
    pcnt_config_B.channel = PCNT_CHANNEL_1;
    pcnt_config_B.unit = PCNT_UNIT_0;
    pcnt_config_B.pos_mode = PCNT_COUNT_INC;
    pcnt_config_B.neg_mode = PCNT_COUNT_DEC;
    pcnt_config_B.counter_h_lim = 10000;
    pcnt_config_B.counter_l_lim = -10000;
    //
    pcnt_unit_config(&pcnt_config_A);//Initialize A
    pcnt_unit_config(&pcnt_config_B);//Initialize B
    pcnt_counter_pause(PCNT_UNIT_0);
    pcnt_counter_clear(PCNT_UNIT_0);
```

```
    pcnt_counter_resume(PCNT_UNIT_0); //Start

    display_init();
    GRAM_clr();
    sprintf(str,  NAME  ); disp_str16(str,20, 90, 0x00ffff);
    sprintf(str,    ID  ); disp_str12(str,40, 70, 0x00ffff);
    sprintf(str, "extended by DJ7OO"); disp_str8(str,30, 5,
0xffd080);
    trans65k();
    f_redraw=1;
    delay(4000);
    init_Dial();
    GRAM_clr();
    si5351_init();
    f_fchange = 1;
    f_cchange = 1;
    f_dchange = 1;
    f_redraw=0;
}
```