```
//--------------------------------------------------------------------------------
#include <Arduino.h>
#include <Wire.h>
#include <U8g2lib.h>

U8G2_SH1106_128X64_NONAME_F_HW_I2C u8g2(U8G2_R0, U8X8_PIN_NONE);

const int einleitung_delay = 1000*(3 /* Sekunde/n */);
const int taster_analog_pin = A0; //Pin
const int display_switch_delay = 1000*(3 /* Sekunde/n */);

void Einleitung();
void Display_Clock();
void Display_Modus();
void Display_Daten();

unsigned long display_getDataTimer;
byte display_layout;
byte display_taster;
byte display_taster_buff = 3;
bool display_layout_switch = false;
//--------------------------------------------------------------------------------
#include <DHT.h>

const int dht_pin = D4; //Pin
const int dht_type = DHT22; //Typ

DHT dht(dht_pin, dht_type);

float temperature = 0;
float humidity = 0;
//--------------------------------------------------------------------------------
#include <MHZ19.h>
#include <SoftwareSerial.h>

const int rx_pin = D7; //Pin
const int tx_pin = D8; //Pin
const int baudrate = 9600; //baud

SoftwareSerial mySerial(rx_pin, tx_pin);

MHZ19 myMHZ19;

int carbondioxide = 0;
//--------------------------------------------------------------------------------
#include <Grove_LED_Bar.h>

const int sda_pin = D6; //Pin
const int scl_pin = D5; //Pin
const int orientierung  = 1;

Grove_LED_Bar bar(scl_pin, sda_pin, orientierung);

const int led_bar_begin = 400; //ppm
const int led_bar_yellow = 1000; //ppm
const int led_bar_red = 1400; //ppm

void LED_Bar();
//--------------------------------------------------------------------------------
const int buzzer_taster_pin = D0; //Pin
const int buzzer_pin = D3; //Pin

const int buzzer_mode_advanced = -100+(1000 /* ppm */);
const int buzzer_timout = 1000*(60 /* Sekunde/n */);
const int buzzer_start = 700; //ppm

void Buzzer();

bool buzzer_taster_buff;
int buzzer_buff;
int buzzer_delay_buff = 0;
bool buzzer_state = false;
unsigned long buzzer_timeout_getDataTimer = 0;
```

```
unsigned long buzzer_delay_getDataTimer = 0;
//-----------------------------------------------------------------------------
#define Feuchtigkeit_leer_width 64
#define Feuchtigkeit_leer_height 64

static unsigned char Feuchtigkeit_leer_bits[] = {
   0x00, 0x00, 0x00, 0x7f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xf0, 0xff,
   0x07, 0x00, 0x00, 0x00, 0x00, 0x00, 0xfe, 0xff, 0x3f, 0x00, 0x00, 0x00,
   0x00, 0x80, 0xff, 0xff, 0xff, 0x00, 0x00, 0x00, 0x00, 0xe0, 0xff, 0x80,
   0xff, 0x03, 0x00, 0x00, 0x00, 0xf8, 0x0f, 0x00, 0xf8, 0x0f, 0x00, 0x00,
   0x00, 0xfc, 0x01, 0x00, 0xc0, 0x1f, 0x00, 0x00, 0x00, 0x7e, 0x00, 0x00,
   0x00, 0x3f, 0x00, 0x00, 0x00, 0x1f, 0x00, 0x00, 0x00, 0x7c, 0x00, 0x00,
   0x80, 0x07, 0x00, 0x00, 0x00, 0xf0, 0x00, 0x00, 0xc0, 0x03, 0x00, 0x7f,
   0x00, 0xe0, 0x01, 0x00, 0xe0, 0x01, 0xe0, 0xff, 0x03, 0xc0, 0x03, 0x00,
   0xe0, 0x01, 0xf8, 0x80, 0x0f, 0xc0, 0x03, 0x00, 0xf0, 0x00, 0x1e, 0x00,
   0x3c, 0x80, 0x07, 0x00, 0xf0, 0x00, 0x07, 0x00, 0x70, 0x80, 0x07, 0x00,
   0x78, 0x80, 0x01, 0x00, 0xc0, 0x00, 0x0f, 0x00, 0x78, 0xc0, 0x00, 0x00,
   0x80, 0x01, 0x0f, 0x00, 0x3c, 0x60, 0x00, 0x00, 0x00, 0x03, 0x1e, 0x00,
   0x3c, 0x60, 0x00, 0x00, 0x00, 0x03, 0x1e, 0x00, 0x3c, 0x30, 0x00, 0x00,
   0x00, 0x06, 0x1e, 0x00, 0x1e, 0x30, 0x00, 0x00, 0x00, 0x06, 0x3c, 0x00,
   0x1e, 0x18, 0x00, 0x00, 0x00, 0x0c, 0x3c, 0x00, 0x1e, 0x18, 0x00, 0x00,
   0x00, 0x0c, 0x3c, 0x00, 0x1e, 0x18, 0x00, 0x00, 0x00, 0x0c, 0x3c, 0x00,
   0x0f, 0x0c, 0x00, 0x00, 0x00, 0x18, 0x78, 0x00, 0x0f, 0x0c, 0x00, 0x00,
   0x00, 0x18, 0x78, 0x00, 0x0f, 0x0c, 0x00, 0x00, 0x00, 0x18, 0x78, 0x00,
   0x0f, 0x0c, 0x00, 0x00, 0x00, 0x18, 0x00, 0x00, 0x0f, 0x0c, 0x00, 0x00,
   0x00, 0x18, 0x00, 0x00, 0x0f, 0x0c, 0x00, 0x00, 0x00, 0x18, 0x00, 0x00,
   0x0f, 0x0c, 0x00, 0x00, 0x00, 0x18, 0x30, 0x00, 0x1e, 0x18, 0x00, 0x00,
   0x00, 0x0c, 0x30, 0x00, 0x1e, 0x18, 0x00, 0x00, 0x00, 0x0c, 0x78, 0x00,
   0x1e, 0x18, 0x00, 0x00, 0x00, 0x0c, 0xfc, 0x00, 0x1e, 0x30, 0x00, 0x00,
   0x00, 0x06, 0xfc, 0x00, 0x3c, 0x30, 0x00, 0x00, 0x00, 0x06, 0xfe, 0x01,
   0x3c, 0x60, 0x00, 0x00, 0x00, 0x03, 0xf7, 0x03, 0x3c, 0x60, 0x00, 0x00,
   0x00, 0x03, 0xf3, 0x03, 0x78, 0xc0, 0x00, 0x00, 0x80, 0x81, 0xf3, 0x07,
   0x78, 0x80, 0x01, 0x00, 0xc0, 0x80, 0xf9, 0x07, 0xf0, 0x00, 0x00, 0x00,
   0x00, 0xc0, 0xf9, 0x0f, 0xf0, 0x00, 0x00, 0x00, 0x00, 0xc0, 0xf8, 0x0f,
   0xe0, 0x01, 0x00, 0x00, 0x00, 0xe0, 0xf8, 0x1f, 0xe0, 0x01, 0x00, 0x00,
   0x00, 0x60, 0xfc, 0x1f, 0xc0, 0x03, 0x00, 0x00, 0x00, 0x70, 0xfc, 0x3f,
   0x80, 0x07, 0x00, 0x00, 0x00, 0x30, 0xfc, 0x3f, 0x00, 0x1f, 0x00, 0x00,
   0x00, 0x38, 0xfe, 0x7f, 0x00, 0x7e, 0x00, 0x00, 0x00, 0x18, 0xfe, 0x7f,
   0x00, 0xfc, 0x01, 0x00, 0x40, 0x18, 0xfe, 0x7f, 0x00, 0xf8, 0x0f, 0x00,
   0x78, 0x1c, 0xfe, 0xff, 0x00, 0xe0, 0xff, 0x80, 0x7f, 0x0c, 0xff, 0xff,
   0x00, 0x80, 0xff, 0xff, 0x7f, 0x0c, 0xff, 0xff, 0x00, 0x00, 0xfe, 0xff,
   0x3f, 0x0c, 0xff, 0xff, 0x00, 0x00, 0xf0, 0xff, 0x07, 0x1c, 0xfe, 0xff,
   0x00, 0x00, 0x00, 0x7f, 0x00, 0x18, 0xfe, 0x7f, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x18, 0xfe, 0x7f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x18, 0xfc, 0x7f,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x38, 0xfc, 0x7f, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x30, 0xfc, 0x3f, 0x00, 0x00, 0x00, 0x00, 0x00, 0xf0, 0xfc, 0x3f,
   0x00, 0x00, 0x00, 0x00, 0x00, 0xe0, 0xff, 0x1f, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x80, 0xff, 0x07, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xfe, 0x01,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x78, 0x00
};
//-----------------------------------------------------------------------------
#define Feuchtigkeit_links_unten_width 15
#define Feuchtigkeit_links_unten_height 15

static unsigned char Feuchtigkeit_links_unten_bits[] = {
   0x00, 0x1f, 0x80, 0x3f, 0xc0, 0x71, 0xc0, 0x64, 0xe0, 0x6e, 0xe0, 0x64,
   0xf0, 0x71, 0xf0, 0x3f, 0xf8, 0x1f, 0xf8, 0x07, 0xfc, 0x01, 0x7e, 0x00,
   0x1f, 0x00, 0x0f, 0x00, 0x07, 0x00
};
//-----------------------------------------------------------------------------
#define Feuchtigkeit_links_mitte_width 19
#define Feuchtigkeit_links_mitte_height 9

static unsigned char Feuchtigkeit_links_mitte_bits[] = {
   0x00, 0xf0, 0x01, 0x00, 0xfe, 0x03, 0xc0, 0x1f, 0x07, 0xfe, 0x4f, 0x06,
   0xff, 0xef, 0x06, 0xfe, 0x4f, 0x06, 0xc0, 0x1f, 0x07, 0x00, 0xfe, 0x03,
   0x00, 0xf0, 0x01
};
//-----------------------------------------------------------------------------
#define Feuchtigkeit_links_oben_width 15
#define Feuchtigkeit_links_oben_height 15

static unsigned char Feuchtigkeit_links_oben_bits[] = {
```

```
   0x07, 0x00, 0x0f, 0x00, 0x1f, 0x00, 0x7e, 0x00, 0xfc, 0x01, 0xf8, 0x07,
   0xf8, 0x1f, 0xf0, 0x3f, 0xf0, 0x71, 0xe0, 0x64, 0xe0, 0x6e, 0xc0, 0x64,
   0xc0, 0x71, 0x80, 0x3f, 0x00, 0x1f
};
//-----------------------------------------------------------------------------
#define Feuchtigkeit_oben_width 9
#define Feuchtigkeit_oben_height 19

static unsigned char Feuchtigkeit_oben_bits[] = {
   0x10, 0x00, 0x38, 0x00, 0x38, 0x00, 0x38, 0x00, 0x38, 0x00, 0x38, 0x00,
   0x7c, 0x00, 0x7c, 0x00, 0x7c, 0x00, 0xfe, 0x00, 0xfe, 0x00, 0xfe, 0x00,
   0xc7, 0x01, 0x93, 0x01, 0xbb, 0x01, 0x93, 0x01, 0xc7, 0x01, 0xfe, 0x00,
   0x7c, 0x00
};
//-----------------------------------------------------------------------------
#define Feuchtigkeit_rechts_oben_width 15
#define Feuchtigkeit_rechts_oben_height 15

static unsigned char Feuchtigkeit_rechts_oben_bits[] = {
   0x00, 0x70, 0x00, 0x78, 0x00, 0x7c, 0x00, 0x3f, 0xc0, 0x1f, 0xf0, 0x0f,
   0xfc, 0x0f, 0xfe, 0x07, 0xc7, 0x07, 0x93, 0x03, 0xbb, 0x03, 0x93, 0x01,
   0xc7, 0x01, 0xfe, 0x00, 0x7c, 0x00
};
//-----------------------------------------------------------------------------
#define Feuchtigkeit_rechts_mitte_width 19
#define Feuchtigkeit_rechts_mitte_height 9

static unsigned char Feuchtigkeit_rechts_mitte_bits[] = {
   0x7c, 0x00, 0x00, 0xfe, 0x03, 0x00, 0xc7, 0x1f, 0x00, 0x93, 0xff, 0x03,
   0xbb, 0xff, 0x07, 0x93, 0xff, 0x03, 0xc7, 0x1f, 0x00, 0xfe, 0x03, 0x00,
   0x7c, 0x00, 0x00
   };
//-----------------------------------------------------------------------------
#define Feuchtigkeit_rechts_unten_width 15
#define Feuchtigkeit_rechts_unten_height 15

static unsigned char Feuchtigkeit_rechts_unten_bits[] = {
   0x7c, 0x00, 0xfe, 0x00, 0xc7, 0x01, 0x93, 0x01, 0xbb, 0x03, 0x93, 0x03,
   0xc7, 0x07, 0xfe, 0x07, 0xfc, 0x0f, 0xf0, 0x0f, 0xc0, 0x1f, 0x00, 0x3f,
   0x00, 0x7c, 0x00, 0x78, 0x00, 0x70
};
//-----------------------------------------------------------------------------
#define Grad_width 6
#define Grad_height 6

static unsigned char Grad_bits[] = {
   0x1e, 0x33, 0x21, 0x21, 0x33, 0x1e
};
//-----------------------------------------------------------------------------
#define Temperatur_leer_width 64
#define Temperatur_leer_height 64

static unsigned char Temperatur_leer_bits[] = {
   0x00, 0x00, 0x00, 0x3e, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x80, 0xff,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xc0, 0xe3, 0x01, 0x00, 0x00, 0x00,
   0x00, 0x00, 0xc0, 0x80, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00, 0xe0, 0x80,
   0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x60, 0x00, 0x03, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x60, 0x00, 0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x60, 0x00,
   0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x60, 0x00, 0x83, 0xff, 0x03, 0x00,
   0x00, 0x00, 0x60, 0x00, 0x83, 0xff, 0x03, 0x00, 0x00, 0x00, 0x60, 0x00,
   0x83, 0xff, 0x03, 0x00, 0x00, 0x00, 0x60, 0x00, 0x03, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x60, 0x00, 0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x60, 0x00,
   0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x60, 0x00, 0x83, 0x1f, 0x00, 0x00,
   0x00, 0x00, 0x60, 0x00, 0x83, 0x1f, 0x00, 0x00, 0x00, 0x00, 0x60, 0x00,
   0x83, 0x1f, 0x00, 0x00, 0x00, 0x00, 0x60, 0x00, 0x03, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x60, 0x00, 0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x60, 0x00,
   0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x60, 0x00, 0x83, 0xff, 0x03, 0x00,
   0x00, 0x00, 0x60, 0x00, 0x83, 0xff, 0x03, 0x00, 0x00, 0x00, 0x60, 0x00,
   0x83, 0xff, 0x03, 0x00, 0x00, 0x00, 0x60, 0x00, 0x03, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x60, 0x00, 0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x60, 0x00,
   0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x60, 0x00, 0x83, 0x1f, 0x00, 0x00,
   0x00, 0x00, 0x60, 0x00, 0x83, 0x1f, 0x00, 0x00, 0x00, 0x00, 0x60, 0x00,
   0x83, 0x1f, 0x00, 0x00, 0x00, 0x00, 0x60, 0x00, 0x03, 0x00, 0x00, 0x00,
```

```
   0x00, 0x00, 0x60, 0x00, 0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x60, 0x00,
   0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x60, 0x00, 0x83, 0xff, 0x03, 0x00,
   0x00, 0x00, 0x60, 0x00, 0x83, 0xff, 0x03, 0x00, 0x00, 0x00, 0x60, 0x00,
   0x83, 0xff, 0x03, 0x00, 0x00, 0x00, 0x60, 0x00, 0x03, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x60, 0x00, 0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x60, 0x00,
   0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x60, 0x00, 0x03, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x70, 0x00, 0x07, 0x00, 0x00, 0x00, 0x00, 0x00, 0x38, 0x00,
   0x0e, 0x00, 0x00, 0x00, 0x00, 0x00, 0x0c, 0x00, 0x18, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x06, 0x7f, 0x30, 0x00, 0x00, 0x00, 0x00, 0x00, 0xc3, 0xf1,
   0x61, 0x00, 0x00, 0x00, 0x00, 0x00, 0x63, 0xe0, 0x63, 0x00, 0x00, 0x00,
   0x00, 0x80, 0x31, 0xe0, 0xc7, 0x00, 0x00, 0x00, 0x00, 0x80, 0x11, 0xf0,
   0xc7, 0x00, 0x00, 0x00, 0x00, 0x80, 0x19, 0xfc, 0xcf, 0x00, 0x00, 0x00,
   0x00, 0xc0, 0x08, 0xfe, 0x8f, 0x01, 0x00, 0x00, 0x00, 0xc0, 0x08, 0xff,
   0x8f, 0x01, 0x00, 0x00, 0x00, 0xc0, 0x08, 0xff, 0x8f, 0x01, 0x00, 0x00,
   0x00, 0xc0, 0x98, 0xff, 0x8f, 0x01, 0x00, 0x00, 0x00, 0xc0, 0xf8, 0xff,
   0x8f, 0x01, 0x00, 0x00, 0x00, 0x80, 0xf9, 0xff, 0xcf, 0x00, 0x00, 0x00,
   0x00, 0x80, 0xf1, 0xff, 0xc7, 0x00, 0x00, 0x00, 0x00, 0x80, 0xf1, 0xff,
   0xc7, 0x00, 0x00, 0x00, 0x00, 0x00, 0xe3, 0xff, 0x63, 0x00, 0x00, 0x00,
   0x00, 0x00, 0xc3, 0xff, 0x61, 0x00, 0x00, 0x00, 0x00, 0x00, 0x06, 0x7f,
   0x30, 0x00, 0x00, 0x00, 0x00, 0x00, 0x0c, 0x00, 0x18, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x38, 0x00, 0x0e, 0x00, 0x00, 0x00, 0x00, 0x00, 0xf0, 0xc1,
   0x07, 0x00, 0x00, 0x00, 0x00, 0x00, 0xc0, 0xff, 0x01, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x3e, 0x00, 0x00, 0x00, 0x00
};
//--------------------------------------------------------------------------------
#define CO2_width 64
#define CO2_height 64

static unsigned char CO2_bits[] = {
   0x00, 0x00, 0x00, 0x00, 0x00, 0x3c, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x80, 0xff, 0x01, 0x00, 0x00, 0x00, 0x1f, 0xe0, 0xc0, 0xff, 0x03, 0x00,
   0x00, 0xc0, 0x7f, 0xfc, 0xc7, 0xff, 0x03, 0x00, 0x00, 0xe0, 0xff, 0xff,
   0xef, 0xff, 0x07, 0x00, 0x00, 0xf0, 0xff, 0xff, 0xff, 0xff, 0x07, 0x00,
   0x00, 0xf0, 0xff, 0xff, 0xff, 0xff, 0x0f, 0x00, 0x00, 0xf8, 0xff, 0xff,
   0xff, 0xff, 0x3f, 0x00, 0x00, 0xf8, 0xff, 0xff, 0xff, 0xff, 0xff, 0x00,
   0x00, 0xf8, 0x07, 0xfc, 0x0f, 0xf8, 0xff, 0x03, 0x00, 0xf8, 0x01, 0xf0,
   0x03, 0xe0, 0xff, 0x07, 0x00, 0xf8, 0x00, 0xe0, 0x01, 0xc0, 0xff, 0x0f,
   0x00, 0xf0, 0xf0, 0xe1, 0xe1, 0xc3, 0xff, 0x0f, 0x00, 0x70, 0xf8, 0xe3,
   0xf0, 0x87, 0xff, 0x1f, 0x00, 0x78, 0xf8, 0xe3, 0xf0, 0x87, 0xff, 0x1f,
   0x00, 0x7e, 0xfc, 0xf7, 0xf8, 0x8f, 0xff, 0x1f, 0x00, 0x7f, 0xfc, 0xff,
   0xf8, 0x8f, 0xff, 0x1f, 0x80, 0x7f, 0xfc, 0xff, 0xf8, 0x8f, 0xc1, 0x1f,
   0x80, 0x7f, 0xfc, 0xff, 0xf8, 0x8f, 0x80, 0x1f, 0xc0, 0x7f, 0xfc, 0xff,
   0xf8, 0x8f, 0x9c, 0x1f, 0xc0, 0x7f, 0xfc, 0xf7, 0xf8, 0x8f, 0x9f, 0x0f,
   0xc0, 0x7f, 0xf8, 0xe3, 0xf0, 0x87, 0x8f, 0x0f, 0xc0, 0x7f, 0xf8, 0xe3,
   0xf0, 0x87, 0xc7, 0x07, 0x80, 0xff, 0xf0, 0xe1, 0xe1, 0xc3, 0xe3, 0x07,
   0x80, 0xff, 0x00, 0xe0, 0x01, 0xc0, 0xf1, 0x07, 0x00, 0xff, 0x01, 0xf0,
   0x03, 0xe0, 0xf8, 0x0f, 0x80, 0xff, 0x07, 0xfc, 0x0f, 0xf8, 0x80, 0x0f,
   0xc0, 0xff, 0xff, 0xff, 0xff, 0xff, 0x80, 0x0f, 0xc0, 0xff, 0xfe, 0xff,
   0xff, 0xff, 0xff, 0x0f, 0xc0, 0x7f, 0xfc, 0xff, 0xff, 0xff, 0xff, 0x0f,
   0x80, 0x1f, 0xfc, 0xff, 0xff, 0xff, 0xff, 0x07, 0x00, 0x0f, 0xf8, 0xff,
   0xff, 0xff, 0xff, 0x07, 0x00, 0x00, 0xf0, 0xff, 0xff, 0xff, 0xff, 0x03,
   0x00, 0x00, 0xc0, 0xff, 0xff, 0xdf, 0xff, 0x01, 0x00, 0x00, 0x00, 0xf0,
   0xff, 0x8f, 0xff, 0x00, 0x80, 0x1f, 0x00, 0xe0, 0xff, 0x07, 0x3e, 0x00,
   0x80, 0x1f, 0x00, 0x80, 0xff, 0x01, 0x00, 0x00, 0x80, 0x1f, 0x00, 0x00,
   0x3c, 0x00, 0x00, 0x00, 0x80, 0x1f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x80, 0x1f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x80, 0x1f, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x80, 0x1f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x80, 0x1f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x80, 0x1f, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x80, 0x1f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0xc0, 0x3f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xc0, 0x3f, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0xc0, 0x3f, 0xff, 0xff, 0xff, 0x00, 0x00, 0x00,
   0xc0, 0x3f, 0xff, 0xff, 0xff, 0x00, 0x00, 0x00, 0xc0, 0x3f, 0xff, 0xff,
   0xff, 0x00, 0x00, 0x00, 0xc0, 0x3f, 0xff, 0xff, 0xff, 0x00, 0x00, 0x00,
   0xc0, 0x3f, 0x8f, 0x31, 0xc6, 0x00, 0x00, 0x00, 0xc0, 0x3f, 0x8f, 0x31,
   0xc6, 0xfc, 0x7f, 0x00, 0xe0, 0x7f, 0x8e, 0x31, 0xc6, 0xfc, 0x7f, 0x00,
   0xe0, 0x7f, 0x8e, 0x31, 0xc6, 0xfc, 0x7f, 0x00, 0xe0, 0x7f, 0x8e, 0x31,
   0xc6, 0xfc, 0x7f, 0x00, 0xe0, 0x7f, 0xfe, 0xff, 0xff, 0xfc, 0x7f, 0x00,
   0xe0, 0x7f, 0xfe, 0xff, 0xff, 0xfc, 0x7f, 0x00, 0xe0, 0x7f, 0xfe, 0xff,
   0xff, 0xfc, 0x7f, 0x00, 0xf0, 0xff, 0xfc, 0xff, 0xff, 0xfc, 0x7f, 0x00,
   0xf0, 0xff, 0xfc, 0xff, 0xff, 0xfc, 0x7f, 0x00, 0xf0, 0xff, 0xfc, 0xff,
   0xff, 0xfc, 0x7f, 0x00, 0xf0, 0xff, 0xfc, 0xff, 0xff, 0xfc, 0x7f, 0x00,
   0xf0, 0xff, 0xfc, 0xff, 0xff, 0xfc, 0x7f, 0x00
};
```

```
//-----------------------------------------------------------------------------
#define Ton_leer_width 32
#define Ton_leer_height 32

static unsigned char Ton_leer_bits[] = {
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0xe0, 0x00, 0x00, 0x00, 0xf0, 0x00, 0x00, 0x00, 0xf8, 0x00, 0x00,
   0x00, 0xfc, 0x00, 0x00, 0x00, 0xfe, 0x00, 0x00, 0xfe, 0xff, 0x00, 0x00,
   0xff, 0xff, 0x00, 0x00, 0xff, 0xff, 0x00, 0x00, 0xff, 0xff, 0x00, 0x00,
   0xff, 0xff, 0x00, 0x00, 0xff, 0xff, 0x00, 0x00, 0xff, 0xff, 0x00, 0x00,
   0xff, 0xff, 0x00, 0x00, 0xff, 0xff, 0x00, 0x00, 0xfe, 0xff, 0x00, 0x00,
   0x00, 0xfe, 0x00, 0x00, 0x00, 0xfc, 0x00, 0x00, 0x00, 0xf8, 0x00, 0x00,
   0x00, 0xf0, 0x00, 0x00, 0x00, 0xe0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
};
//-----------------------------------------------------------------------------
#define Ton_welle_1_width 3
#define Ton_welle_1_height 4

static unsigned char Ton_welle_1_bits[] = {
   0x03, 0x06, 0x06, 0x03
};
//-----------------------------------------------------------------------------
#define Ton_welle_2_width 8
#define Ton_welle_2_height 12

static unsigned char Ton_welle_2_bits[] = {
   0x18, 0x30, 0x60, 0x60, 0xc3, 0xc6, 0xc6, 0xc3, 0x60, 0x60, 0x30, 0x18
};
//-----------------------------------------------------------------------------
#define Ton_welle_3_width 12
#define Ton_welle_3_height 20

static unsigned char Ton_welle_3_bits[] = {
   0xc0, 0x00, 0x80, 0x01, 0x00, 0x03, 0x00, 0x03, 0x18, 0x06, 0x30, 0x06,
   0x60, 0x0c, 0x60, 0x0c, 0xc3, 0x0c, 0xc6, 0x0c, 0xc6, 0x0c, 0xc3, 0x0c,
   0x60, 0x0c, 0x60, 0x0c, 0x30, 0x06, 0x18, 0x06, 0x00, 0x03, 0x00, 0x03,
   0x80, 0x01, 0xc0, 0x00
};
//-----------------------------------------------------------------------------
#define Ton_mute_width 32
#define Ton_mute_height 32

static unsigned char Ton_mute_bits[] = {
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x0c, 0xe0, 0x00, 0x0c, 0x1c, 0xf0, 0x00, 0x18, 0x70, 0xf8, 0x00, 0x30,
   0xe0, 0xf8, 0x00, 0x30, 0x80, 0xf3, 0x80, 0x61, 0xfe, 0xc7, 0x00, 0x63,
   0xff, 0x9f, 0x00, 0xc6, 0xff, 0x3f, 0x00, 0xc6, 0xff, 0xff, 0x30, 0xcc,
   0xff, 0xff, 0x61, 0xcc, 0xff, 0xff, 0x47, 0xcc, 0xff, 0xff, 0x0e, 0xcc,
   0xff, 0xff, 0x38, 0xc6, 0xff, 0xff, 0x70, 0xc4, 0xfe, 0xff, 0xc0, 0x61,
   0x00, 0xfe, 0x80, 0x43, 0x00, 0xfc, 0x00, 0x0e, 0x00, 0xf8, 0x00, 0x1c,
   0x00, 0xf0, 0x00, 0x78, 0x00, 0xe0, 0x00, 0x6c, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
};
//-----------------------------------------------------------------------------
#define Bild_width 32
#define Bild_height 32

static unsigned char Bild_bits[] = {
   0xe0, 0xff, 0xff, 0x07, 0xf8, 0xff, 0xff, 0x1f, 0xfc, 0xff, 0xff, 0x3f,
   0x7e, 0x00, 0x00, 0x7e, 0x1e, 0x00, 0x00, 0x78, 0x0f, 0x00, 0x00, 0xf0,
   0x0f, 0x00, 0x00, 0xf0, 0x07, 0x00, 0x00, 0xe0, 0x07, 0x06, 0x00, 0xe0,
   0x07, 0x0f, 0x00, 0xe0, 0x07, 0x0f, 0x04, 0xe0, 0x07, 0x06, 0x04, 0xe0,
   0x07, 0x00, 0x0e, 0xe0, 0x07, 0x00, 0x0e, 0xe0, 0x07, 0x00, 0x1f, 0xe0,
   0x07, 0x00, 0x1f, 0xe0, 0x07, 0x80, 0x3f, 0xe0, 0x07, 0x84, 0x3f, 0xe0,
   0x07, 0xc4, 0x7f, 0xe0, 0x07, 0xce, 0x7f, 0xe0, 0x07, 0xee, 0xff, 0xe0,
   0x07, 0xff, 0xff, 0xe0, 0x07, 0xff, 0xff, 0xe1, 0x87, 0xff, 0xff, 0xe1,
   0x87, 0xff, 0xff, 0xe1, 0x0f, 0xff, 0xff, 0xf0, 0x0f, 0x00, 0x00, 0xf0,
   0x1e, 0x00, 0x00, 0x78, 0x7e, 0x00, 0x00, 0x7e, 0xfc, 0xff, 0xff, 0x3f,
```

```
   0xf8, 0xff, 0xff, 0x1f, 0xe0, 0xff, 0xff, 0x07
};
//-----------------------------------------------------------------------------
#define Text_width 32
#define Text_height 32

static unsigned char Text_bits[] = {
   0xf0, 0xff, 0x1f, 0x00, 0xf8, 0xff, 0x3f, 0x00, 0x38, 0x00, 0x73, 0x00,
   0x18, 0x00, 0xe3, 0x00, 0x18, 0x00, 0xc3, 0x01, 0x18, 0x00, 0x83, 0x03,
   0x18, 0x00, 0x03, 0x07, 0x18, 0x00, 0x03, 0x0e, 0x18, 0x00, 0x03, 0x1c,
   0x18, 0x00, 0x03, 0x18, 0x18, 0x00, 0x07, 0x18, 0x18, 0x00, 0xff, 0x1f,
   0x18, 0x00, 0xfe, 0x1f, 0x18, 0x00, 0x00, 0x18, 0x18, 0x00, 0x00, 0x18,
   0x18, 0x00, 0x00, 0x18, 0x18, 0xff, 0xff, 0x18, 0x18, 0xff, 0xff, 0x18,
   0x18, 0xff, 0xff, 0x18, 0x18, 0x00, 0x00, 0x18, 0x18, 0x00, 0x00, 0x18,
   0x18, 0x00, 0x00, 0x18, 0x18, 0xff, 0xff, 0x18, 0x18, 0xff, 0xff, 0x18,
   0x18, 0xff, 0xff, 0x18, 0x18, 0x00, 0x00, 0x18, 0x18, 0x00, 0x00, 0x18,
   0x18, 0x00, 0x00, 0x18, 0x18, 0x00, 0x00, 0x18, 0x38, 0x00, 0x00, 0x1c,
   0xf8, 0xff, 0xff, 0x1f, 0xf0, 0xff, 0xff, 0x0f
};
//-----------------------------------------------------------------------------
#define HTL_Wien_West_Logo_128x64_width 128
#define HTL_Wien_West_Logo_128x64_height 64

static unsigned char HTL_Wien_West_Logo_128x64_bits[] = {
   0xff, 0xff, 0x7f, 0x00, 0x00, 0x00, 0xf8, 0xff, 0xff, 0x03, 0x1c, 0x70,
   0xf0, 0xff, 0xc1, 0x01, 0x07, 0x00, 0x80, 0x01, 0x00, 0x00, 0x06, 0x00,
   0x80, 0x03, 0x1c, 0x70, 0xf0, 0xff, 0xc1, 0x01, 0x19, 0x00, 0x00, 0x06,
   0x00, 0x80, 0x01, 0x00, 0x60, 0x02, 0x1c, 0x70, 0xf0, 0xff, 0xc1, 0x01,
   0x61, 0x00, 0x00, 0x18, 0x00, 0x60, 0x00, 0x00, 0x18, 0x02, 0x1c, 0x70,
   0x00, 0x0e, 0xc0, 0x01, 0x81, 0x01, 0x00, 0x60, 0x00, 0x18, 0x00, 0x00,
   0x06, 0x02, 0x1c, 0x70, 0x00, 0x0e, 0xc0, 0x01, 0x01, 0x06, 0x00, 0x80,
   0x01, 0x06, 0x00, 0x80, 0x01, 0x02, 0x1c, 0x70, 0x00, 0x0e, 0xc0, 0x01,
   0x01, 0xf8, 0xff, 0xff, 0x01, 0xfe, 0xff, 0x7f, 0x00, 0x02, 0x1c, 0x70,
   0x00, 0x0e, 0xc0, 0x01, 0x01, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40,
   0x00, 0x02, 0xfc, 0x7f, 0x00, 0x0e, 0xc0, 0x01, 0x01, 0x08, 0x00, 0x00,
   0x01, 0x02, 0x00, 0x40, 0x00, 0x02, 0xfc, 0x7f, 0x00, 0x0e, 0xc0, 0x01,
   0x01, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40, 0x00, 0x02, 0xfc, 0x7f,
   0x00, 0x0e, 0xc0, 0x01, 0x01, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40,
   0x00, 0x02, 0x1c, 0x70, 0x00, 0x0e, 0xc0, 0x01, 0x01, 0x08, 0x00, 0x00,
   0x01, 0x02, 0x00, 0x40, 0x00, 0x02, 0x1c, 0x70, 0x00, 0x0e, 0xc0, 0x01,
   0x01, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40, 0x00, 0x02, 0x1c, 0x70,
   0x00, 0x0e, 0xc0, 0x01, 0x01, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40,
   0x00, 0x02, 0x1c, 0x70, 0x00, 0x0e, 0xc0, 0x01, 0x01, 0x08, 0x00, 0x00,
   0x01, 0x02, 0x00, 0x40, 0x00, 0x02, 0x1c, 0x70, 0x00, 0x0e, 0xc0, 0x7f,
   0x01, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40, 0x00, 0x02, 0x1c, 0x70,
   0x00, 0x0e, 0xc0, 0x7f, 0x01, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40,
   0x00, 0x02, 0x1c, 0x70, 0x00, 0x0e, 0xc0, 0x7f, 0x01, 0x08, 0x00, 0x00,
   0x01, 0x02, 0x00, 0x40, 0x00, 0x02, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x01, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40, 0x00, 0x02, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x01, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40,
   0x00, 0x02, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x01, 0x08, 0x00, 0x00,
   0x01, 0x02, 0x00, 0x40, 0x00, 0x02, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x01, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40, 0x00, 0x02, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x03, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40,
   0x00, 0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x0c, 0x08, 0x00, 0x00,
   0x01, 0x02, 0x00, 0x40, 0xc0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x30, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40, 0x30, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0xc0, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40,
   0x0c, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x0b, 0x00, 0x00,
   0x01, 0x02, 0x00, 0x40, 0x03, 0x00, 0x06, 0x83, 0x31, 0xfc, 0xc7, 0xc0,
   0x00, 0xfc, 0xff, 0xff, 0x01, 0xfe, 0xff, 0xff, 0x00, 0x00, 0x06, 0x83,
   0x31, 0xfc, 0xc7, 0xc1, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x06, 0x83, 0x31, 0x0c, 0xc0, 0xc3, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x04, 0x83, 0x30, 0x0c, 0xc0, 0xc3,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x8c, 0xc7,
   0x30, 0x0c, 0xc0, 0xc6, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x8c, 0xc7, 0x30, 0x0c, 0xc0, 0xc6, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x8c, 0xc4, 0x30, 0xfc, 0xc3, 0xcc,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xc8, 0x4c,
   0x30, 0xfc, 0xc3, 0xcc, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0xd8, 0x6c, 0x30, 0x0c, 0xc0, 0xd8, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xd8, 0x6c, 0x30, 0x0c, 0xc0, 0xd8,
   0x00, 0xfc, 0xff, 0xff, 0x01, 0xfe, 0xff, 0xff, 0x00, 0x00, 0x58, 0x68,
```

```
    0x30, 0x0c, 0xc0, 0xf0, 0x00, 0x0b, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40,
    0x03, 0x00, 0x70, 0x38, 0x30, 0x0c, 0xc0, 0xf0, 0xc0, 0x08, 0x00, 0x00,
    0x01, 0x02, 0x00, 0x40, 0x0c, 0x00, 0x70, 0x38, 0x30, 0xfc, 0xc7, 0xe0,
    0x30, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40, 0x30, 0x00, 0x20, 0x10,
    0x30, 0xfc, 0xc7, 0xc0, 0x0c, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40,
    0xc0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x03, 0x08, 0x00, 0x00,
    0x01, 0x02, 0x00, 0x40, 0x00, 0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    0x01, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40, 0x00, 0x02, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x01, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40,
    0x00, 0x02, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x01, 0x08, 0x00, 0x00,
    0x01, 0x02, 0x00, 0x40, 0x00, 0x02, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    0x01, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40, 0x00, 0x02, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x01, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40,
    0x00, 0x02, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x01, 0x08, 0x00, 0x00,
    0x01, 0x02, 0x00, 0x40, 0x00, 0x02, 0x06, 0x83, 0xfd, 0xc7, 0xcf, 0xff,
    0x01, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40, 0x00, 0x02, 0x06, 0x83,
    0xfd, 0xe7, 0xdf, 0xff, 0x01, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40,
    0x00, 0x02, 0x06, 0x83, 0x0d, 0x70, 0x18, 0x0c, 0x01, 0x08, 0x00, 0x00,
    0x01, 0x02, 0x00, 0x40, 0x00, 0x02, 0x04, 0x83, 0x0c, 0x30, 0x00, 0x0c,
    0x01, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40, 0x00, 0x02, 0x8c, 0xc7,
    0x0c, 0x30, 0x00, 0x0c, 0x01, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40,
    0x00, 0x02, 0x8c, 0xc7, 0x0c, 0x70, 0x00, 0x0c, 0x01, 0x08, 0x00, 0x00,
    0x01, 0x02, 0x00, 0x40, 0x00, 0x02, 0x8c, 0xc4, 0xfc, 0xe3, 0x07, 0x0c,
    0x01, 0xf8, 0xff, 0xff, 0x01, 0xfe, 0xff, 0x7f, 0x00, 0x02, 0xc8, 0x4c,
    0xfc, 0xc3, 0x0f, 0x0c, 0x01, 0x06, 0x00, 0x80, 0x01, 0x06, 0x00, 0x80,
    0x01, 0x02, 0xd8, 0x6c, 0x0c, 0x00, 0x1c, 0x0c, 0x81, 0x01, 0x00, 0x60,
    0x00, 0x18, 0x00, 0x00, 0x06, 0x02, 0xd8, 0x6c, 0x0c, 0x00, 0x18, 0x0c,
    0x61, 0x00, 0x00, 0x18, 0x00, 0x60, 0x00, 0x00, 0x18, 0x02, 0x58, 0x68,
    0x0c, 0x00, 0x18, 0x0c, 0x19, 0x00, 0x00, 0x06, 0x00, 0x80, 0x01, 0x00,
    0x60, 0x02, 0x70, 0x38, 0x0c, 0x30, 0x1c, 0x0c, 0x07, 0x00, 0x80, 0x01,
    0x00, 0x00, 0x06, 0x00, 0x80, 0x03, 0x70, 0x38, 0xfc, 0xf7, 0x0f, 0x0c,
    0xff, 0xff, 0x7f, 0x00, 0x00, 0x00, 0xf8, 0xff, 0xff, 0x03, 0x20, 0x10,
    0xfc, 0xe7, 0x07, 0x0c
};
//---------------------------------------------------------------------------
bool buff = false;
unsigned long getDataTimer = 0;
int buff_count = 0;
byte timer = display_switch_delay/(32 /* >4 */);
//---------------------------------------------------------------------------
void setup() {
  pinMode(buzzer_pin, OUTPUT);
  pinMode(buzzer_taster_pin, INPUT);
  Serial.begin(115200);
  u8g2.begin();
  bar.begin();
  dht.begin();
  mySerial.begin(baudrate);
  myMHZ19.begin(mySerial);
  myMHZ19.autoCalibration(true, 24);
  myMHZ19.setRange(5000);
  buzzer_buff = myMHZ19.getCO2()/100*100;
  Einleitung();
}

void loop() {
  if(myMHZ19.getCO2() > 0) carbondioxide = myMHZ19.getCO2();
  if(dht.readTemperature() > 0) temperature = dht.readTemperature();
  if(dht.readHumidity() > 0) humidity = dht.readHumidity();

  Display_Clock();
  LED_Bar();
  Buzzer();
}

void Display_Clock() {
  if(analogRead(taster_analog_pin) < 256) display_taster = 1;
  if(analogRead(taster_analog_pin) > 256 & analogRead(taster_analog_pin) < 768)
  display_taster = 2;
```

```
511     if(analogRead(taster_analog_pin) > 768 & analogRead(taster_analog_pin) <= 1024)
        display_taster = 0;
512
513     if(display_taster != display_taster_buff) Display_Modus();
514     display_taster_buff = display_taster;
515
516     if(millis() - display_getDataTimer >= display_switch_delay & display_layout_switch
        == true) {
517       switch(display_taster) {
518         case 0:
519           display_layout = 0;
520         break;
521         case 1:
522           if(display_layout < 3) display_layout++;
523           else display_layout = 0;
524         break;
525         case 2:
526           if(display_layout < 3) display_layout++;
527           else display_layout = 1;
528         break;
529       }
530       display_getDataTimer = millis();
531     }
532     Display_Daten();
533   }
534
535   void Display_Daten() {
536     if(millis() - display_getDataTimer >= display_switch_delay) {
537       display_layout_switch = true;
538       display_getDataTimer = millis();
539     }
540     if(display_layout_switch == true) {
541       u8g2.firstPage();
542       do {
543         switch(display_layout) {
544           case 0:
545             u8g2.setFont(u8g2_font_ncenB08_tr);
546             u8g2.setCursor(0,10);
547             u8g2.print("Temperatur:");
548             u8g2.setCursor(80,10);
549             u8g2.print(temperature);
550             u8g2.print("C");
551             u8g2.setCursor(0,35);
552             u8g2.print("Feuchtigkeit:");
553             u8g2.setCursor(80,35);
554             u8g2.print(humidity);
555             u8g2.print("%");
556             u8g2.setCursor(0,60);
557             u8g2.print("CO2:");
558             u8g2.setCursor(80,60);
559             u8g2.print(carbondioxide);
560             u8g2.print("ppm");
561           break;
562           case 1:
563             u8g2.setFont(u8g2_font_ncenB18_tr);
564             u8g2.drawXBMP(0, 0, Temperatur_leer_width, Temperatur_leer_height,
              Temperatur_leer_bits);
565             if(temperature >= 0 & temperature <= 37) {
566               u8g2.drawVLine(26, 42-round(temperature), round(temperature));
567               u8g2.drawVLine(27, 42-round(temperature), round(temperature));
568               u8g2.drawVLine(28, 42-round(temperature), round(temperature));
569             }
570             u8g2.drawXBMP(81, 35, Grad_width, Grad_height, Grad_bits);
571             u8g2.setCursor(65,26);
572             u8g2.print(temperature);
573             u8g2.setCursor(88,54);
574             u8g2.print("C");
575           break;
576           case 2:
577             u8g2.setFont(u8g2_font_ncenB18_tr);
578             u8g2.drawXBMP(0, 0, Feuchtigkeit_leer_width, Feuchtigkeit_leer_height,
              Feuchtigkeit_leer_bits);
579             if(humidity <= 15) u8g2.drawXBMP(17, 23, Feuchtigkeit_links_unten_width,
```

```
            Feuchtigkeit_links_unten_height, Feuchtigkeit_links_unten_bits);
            if(humidity > 15 & humidity <= 30) u8g2.drawXBMP(13, 23,
            Feuchtigkeit_links_mitte_width, Feuchtigkeit_links_mitte_height,
            Feuchtigkeit_links_mitte_bits);
            if(humidity > 30 & humidity <= 45) u8g2.drawXBMP(17, 17,
            Feuchtigkeit_links_oben_width, Feuchtigkeit_links_oben_height,
            Feuchtigkeit_links_oben_bits);
            if(humidity > 45 & humidity <= 55) u8g2.drawXBMP(23, 13,
            Feuchtigkeit_oben_width, Feuchtigkeit_oben_height, Feuchtigkeit_oben_bits);
            if(humidity > 55 & humidity <= 70) u8g2.drawXBMP(23, 17,
            Feuchtigkeit_rechts_oben_width, Feuchtigkeit_rechts_oben_height,
            Feuchtigkeit_rechts_oben_bits);
            if(humidity > 70 & humidity <= 85) u8g2.drawXBMP(23, 23,
            Feuchtigkeit_rechts_mitte_width, Feuchtigkeit_rechts_mitte_height,
            Feuchtigkeit_rechts_mitte_bits);
            if(humidity > 85) u8g2.drawXBMP(23, 23, Feuchtigkeit_rechts_unten_width,
            Feuchtigkeit_rechts_unten_height, Feuchtigkeit_rechts_unten_bits);
            u8g2.setCursor(65,26);
            u8g2.print(humidity);
            u8g2.setCursor(86,54);
            u8g2.print("%");
          break;
          case 3:
            u8g2.setFont(u8g2_font_ncenB18_tr);
            u8g2.drawXBMP(0, 0, CO2_width, CO2_height, CO2_bits);
            u8g2.setCursor(68,26);
            u8g2.print(carbondioxide);
            u8g2.setCursor(68,54);
            u8g2.print("ppm");
          break;
        }
      } while(u8g2.nextPage());
    }
  }

  void Display_Modus() {
    u8g2.firstPage();
    do {
      switch(display_taster) {
        case 0:
          u8g2.drawXBMP(0, 16, Text_width, Text_height, Text_bits);
          display_layout = 0;
        break;
        case 1:
          u8g2.drawXBMP(0, 16, Text_width, Text_height, Text_bits);
          u8g2.drawXBMP(48, 16, Bild_width, Bild_height, Bild_bits);
          display_layout = 0;
        break;
        case 2:
          u8g2.drawXBMP(48, 16, Bild_width, Bild_height, Bild_bits);
          display_layout = 1;
        break;
      }
      if(digitalRead(buzzer_taster_pin) == HIGH) {
        if(millis() - getDataTimer >= timer & buff == true) {
          buff_count++;
          getDataTimer = millis();
        }
        switch(buff_count) {
          case 0:
            u8g2.drawXBMP(96, 16, Ton_leer_width, Ton_leer_height, Ton_leer_bits);
          break;
          case 1:
            u8g2.drawXBMP(96, 16, Ton_leer_width, Ton_leer_height, Ton_leer_bits);
            u8g2.drawXBMP(116, 30, Ton_welle_1_width, Ton_welle_1_height,
            Ton_welle_1_bits);
          break;
          case 2:
            u8g2.drawXBMP(96, 16, Ton_leer_width, Ton_leer_height, Ton_leer_bits);
            u8g2.drawXBMP(116, 26, Ton_welle_2_width, Ton_welle_2_height,
            Ton_welle_2_bits);
          break;
          case 3:
```

```
640             u8g2.drawXBMP(96, 16, Ton_leer_width, Ton_leer_height, Ton_leer_bits);
641             u8g2.drawXBMP(116, 22, Ton_welle_3_width, Ton_welle_3_height,
                Ton_welle_3_bits);
642             buff = false;
643           break;
644         }
645       }
646       else {
647         u8g2.drawXBMP(96, 16, Ton_mute_width, Ton_mute_height, Ton_mute_bits);
648         buff = false;
649       }
650     } while(u8g2.nextPage());
651     display_getDataTimer = millis();
652     display_layout_switch = false;
653   }
654
655   void LED_Bar() {
656     if(carbondioxide <= led_bar_begin) bar.setLevel(1);
657     else {
658       if(carbondioxide >= led_bar_red) bar.setLevel(10);
659       else {
660         if(carbondioxide > led_bar_begin & carbondioxide <= led_bar_yellow)
          bar.setLevel(map(carbondioxide, led_bar_begin, led_bar_yellow, 1, 9));
661         else bar.setLevel(9);
662       }
663     }
664   }
665
666   void Buzzer() {
667     if(digitalRead(buzzer_taster_pin) != buzzer_taster_buff) {
668       if(digitalRead(buzzer_taster_pin) == HIGH) {
669         buff = true;
670         buff_count = 0;
671         getDataTimer = millis();
672       }
673       else {
674         buff = false;
675         Display_Modus();
676       }
677     }
678     if(buff == true) Display_Modus();
679     buzzer_taster_buff = digitalRead(buzzer_taster_pin);
680
681     if(millis() - buzzer_timeout_getDataTimer >= buzzer_timout & buzzer_state == false
      & carbondioxide >= buzzer_start) {
682       buzzer_state = true;
683     }
684     Serial.print("Time: " + String(millis() - buzzer_timeout_getDataTimer) +
685                  " // buzzer_state: " + String(buzzer_state) +
686                  " // CO2: " + String(carbondioxide));
687     if(carbondioxide >= buzzer_buff+100) {
688       if(buzzer_state == true & digitalRead(buzzer_taster_pin) == HIGH) {
689         if(buzzer_delay_buff == 0 or (millis() - buzzer_delay_getDataTimer >= 50 &
          (buzzer_delay_buff == 2 or buzzer_delay_buff == 4))) {
690           digitalWrite(buzzer_pin, HIGH);
691           if(buzzer_buff <= buzzer_mode_advanced) buzzer_delay_buff = 5;
692           else buzzer_delay_buff++;
693           buzzer_delay_getDataTimer = millis();
694         }
695         if(millis() - buzzer_delay_getDataTimer >= 200 & (buzzer_delay_buff == 1 or
          buzzer_delay_buff == 3 or buzzer_delay_buff == 5)) {
696           digitalWrite(buzzer_pin, LOW);
697           if(buzzer_delay_buff == 5) {
698             buzzer_state = false;
699             buzzer_delay_buff = 0;
700             buzzer_timeout_getDataTimer = millis();
701           }
702           else {
703             buzzer_delay_buff++;
704             buzzer_delay_getDataTimer = millis();
705           }
706         }
707       }
```

```
708       if(buzzer_state == false or digitalRead(buzzer_taster_pin) == LOW){
709         buzzer_buff = buzzer_buff+100;
710       }
711     }
712     if(carbondioxide <= buzzer_buff-100) buzzer_buff = buzzer_buff-100;
713     Serial.println(" // Buffer: " + String(buzzer_buff));
714     if(millis() - buzzer_delay_getDataTimer >= 200) digitalWrite(buzzer_pin, LOW);
715     if(digitalRead(buzzer_taster_pin) == LOW){
716       digitalWrite(buzzer_pin, LOW);
717       buzzer_state = false;
718       buzzer_timeout_getDataTimer = millis();
719     }
720   }
721
722   void Einleitung() {
723     bar.setLevel(0);
724     for(int index1 = 0; index1 <= 2; index1++) {
725       u8g2.firstPage();
726       do {
727         switch(index1) {
728           case 0:
729             u8g2.drawXBMP(0, 0, HTL_Wien_West_Logo_128x64_width,
              HTL_Wien_West_Logo_128x64_height, HTL_Wien_West_Logo_128x64_bits);
730           break;
731           case 1:
732             u8g2.setFont(u8g2_font_ncenB10_tr);
733             u8g2.setCursor(10,21);
734             u8g2.print("Entwickelt von");
735             u8g2.setCursor(42,42);
736             u8g2.print("Jakob");
737             u8g2.setCursor(22,63);
738             u8g2.print("Estermann");
739           break;
740           case 2:
741             u8g2.drawXBMP(0, 0, HTL_Wien_West_Logo_128x64_width,
              HTL_Wien_West_Logo_128x64_height, HTL_Wien_West_Logo_128x64_bits);
742           break;
743         }
744       } while(u8g2.nextPage());
745       delay(einleitung_delay);
746     }
747   }
748
```