

```

1 //-----
2 #include <Arduino.h>
3 #include <Wire.h>
4 #include <U8g2lib.h>
5
6 U8G2_SH1106_128X64_NONAME_F_HW_I2C u8g2(U8G2_R0, U8X8_PIN_NONE);
7
8 const int einleitung_delay = 1000*(3 /* Sekunde/n */);
9 const int taster_analog_pin = A0; //Pin
10 const int display_switch_delay = 1000*(3 /* Sekunde/n */);
11
12 void Einleitung();
13 void Display_Clock();
14 void Display_Modus();
15 void Display_Daten();
16
17 unsigned long display_getDataTimer;
18 byte display_layout;
19 byte display_taster;
20 byte display_taster_buff = 3;
21 bool display_layout_switch = false;
22 //-----
23 #include <DHT.h>
24
25 const int dht_pin = D4; //Pin
26 const int dht_type = DHT22; //Typ
27
28 DHT dht(dht_pin, dht_type);
29
30 float temperature = 0;
31 float humidity = 0;
32 //-----
33 #include <MHZ19.h>
34 #include <SoftwareSerial.h>
35
36 const int rx_pin = D7; //Pin
37 const int tx_pin = D8; //Pin
38 const int baudrate = 9600; //baud
39
40 SoftwareSerial mySerial(rx_pin, tx_pin);
41
42 MHZ19 myMHZ19;
43
44 int carbondioxide = 0;
45 //-----
46 #include <Grove_LED_Bar.h>
47
48 const int sda_pin = D6; //Pin
49 const int scl_pin = D5; //Pin
50 const int orientierung = 1;
51
52 Grove_LED_Bar bar(scl_pin, sda_pin, orientierung);
53
54 const int led_bar_begin = 400; //ppm
55 const int led_bar_yellow = 1000; //ppm
56 const int led_bar_red = 1400; //ppm
57
58 void LED_Bar();
59 //-----
60 const int buzzer_taster_pin = D0; //Pin
61 const int buzzer_pin = D3; //Pin
62
63 const int buzzer_mode_advanced = -100+(1000 /* ppm */);
64 const int buzzer_timeout = 1000*(60 /* Sekunde/n */);
65 const int buzzer_start = 700; //ppm
66
67 void Buzzer();
68
69 bool buzzer_taster_buff;
70 int buzzer_buff;
71 int buzzer_delay_buff = 0;
72 bool buzzer_state = false;
73 unsigned long buzzer_timeout_getDataTimer = 0;

```

```

74 unsigned long buzzer_delay_getDataTimer = 0;
75 //-----
76 #define Feuchtigkeit_leer_width 64
77 #define Feuchtigkeit_leer_height 64
78
79 static unsigned char Feuchtigkeit_leer_bits[] = {
80     0x00, 0x00, 0x00, 0x7f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xf0, 0xff,
81     0x07, 0x00, 0x00, 0x00, 0x00, 0x00, 0xfe, 0xff, 0x3f, 0x00, 0x00, 0x00,
82     0x00, 0x80, 0xff, 0xff, 0xff, 0x00, 0x00, 0x00, 0x00, 0xe0, 0xff, 0x80,
83     0xff, 0x03, 0x00, 0x00, 0x00, 0xf8, 0x0f, 0x00, 0xf8, 0x0f, 0x00, 0x00,
84     0x00, 0xfc, 0x01, 0x00, 0xc0, 0x1f, 0x00, 0x00, 0x00, 0x7e, 0x00, 0x00,
85     0x00, 0x3f, 0x00, 0x00, 0x00, 0x1f, 0x00, 0x00, 0x00, 0x7c, 0x00, 0x00,
86     0x80, 0x07, 0x00, 0x00, 0x00, 0xf0, 0x00, 0x00, 0xc0, 0x03, 0x00, 0x7f,
87     0x00, 0xe0, 0x01, 0x00, 0xe0, 0x01, 0xe0, 0xff, 0x03, 0xc0, 0x03, 0x00,
88     0xe0, 0x01, 0xf8, 0x80, 0x0f, 0xc0, 0x03, 0x00, 0xf0, 0x00, 0x1e, 0x00,
89     0x3c, 0x80, 0x07, 0x00, 0xf0, 0x00, 0x07, 0x00, 0x70, 0x80, 0x07, 0x00,
90     0x78, 0x80, 0x01, 0x00, 0xc0, 0x00, 0x0f, 0x00, 0x78, 0xc0, 0x00, 0x00,
91     0x80, 0x01, 0x0f, 0x00, 0x3c, 0x60, 0x00, 0x00, 0x00, 0x03, 0x1e, 0x00,
92     0x3c, 0x60, 0x00, 0x00, 0x00, 0x03, 0x1e, 0x00, 0x3c, 0x30, 0x00, 0x00,
93     0x00, 0x06, 0x1e, 0x00, 0x1e, 0x30, 0x00, 0x00, 0x00, 0x06, 0x3c, 0x00,
94     0x1e, 0x18, 0x00, 0x00, 0x00, 0xc0, 0x3c, 0x00, 0x1e, 0x18, 0x00, 0x00,
95     0x00, 0x0c, 0x3c, 0x00, 0x00, 0x1e, 0x18, 0x00, 0x00, 0x00, 0x0c, 0x3c,
96     0x0f, 0x0c, 0x00, 0x00, 0x00, 0x18, 0x78, 0x00, 0x0f, 0x0c, 0x00, 0x00,
97     0x00, 0x18, 0x78, 0x00, 0x0f, 0x0c, 0x00, 0x00, 0x00, 0x18, 0x78, 0x00,
98     0x0f, 0x0c, 0x00, 0x00, 0x00, 0x18, 0x00, 0x00, 0x0f, 0x0c, 0x00, 0x00,
99     0x00, 0x18, 0x00, 0x00, 0x0f, 0x0c, 0x00, 0x00, 0x00, 0x18, 0x00, 0x00,
100    0x0f, 0x0c, 0x00, 0x00, 0x00, 0x18, 0x30, 0x00, 0x1e, 0x18, 0x00, 0x00,
101    0x00, 0x0c, 0x30, 0x00, 0x1e, 0x18, 0x00, 0x00, 0x00, 0x0c, 0x78, 0x00,
102    0x1e, 0x18, 0x00, 0x00, 0x00, 0x0c, 0xfc, 0x00, 0x1e, 0x30, 0x00, 0x00,
103    0x00, 0x06, 0xfc, 0x00, 0x3c, 0x30, 0x00, 0x00, 0x00, 0x06, 0xfe, 0x01,
104    0x3c, 0x60, 0x00, 0x00, 0x00, 0x03, 0xf7, 0x03, 0x3c, 0x60, 0x00, 0x00,
105    0x00, 0x03, 0xf3, 0x03, 0x78, 0xc0, 0x00, 0x00, 0x80, 0x81, 0xf3, 0x07,
106    0x78, 0x80, 0x01, 0x00, 0xc0, 0x80, 0xf9, 0x07, 0xf0, 0x00, 0x00, 0x00,
107    0x00, 0xc0, 0xf9, 0x0f, 0xf0, 0x00, 0x00, 0x00, 0x00, 0xc0, 0xf8, 0x0f,
108    0xe0, 0x01, 0x00, 0x00, 0x00, 0xe0, 0xf8, 0x1f, 0xe0, 0x01, 0x00, 0x00,
109    0x00, 0x60, 0xfc, 0x1f, 0xc0, 0x03, 0x00, 0x00, 0x00, 0x70, 0xfc, 0x3f,
110    0x80, 0x07, 0x00, 0x00, 0x00, 0x30, 0xfc, 0x3f, 0x00, 0x1f, 0x00, 0x00,
111    0x00, 0x38, 0xfe, 0x7f, 0x00, 0x7e, 0x00, 0x00, 0x00, 0x18, 0xfe, 0x7f,
112    0x00, 0xfc, 0x01, 0x00, 0x40, 0x18, 0xfe, 0x7f, 0x00, 0xf8, 0x0f, 0x00,
113    0x78, 0x1c, 0xfe, 0xff, 0x00, 0xe0, 0xff, 0x80, 0x7f, 0x0c, 0xff, 0xff,
114    0x00, 0x80, 0xff, 0xff, 0x7f, 0x0c, 0xff, 0xff, 0x00, 0x00, 0xfe, 0xff,
115    0x3f, 0x0c, 0xff, 0xff, 0x00, 0x00, 0xf0, 0xff, 0x07, 0x1c, 0xfe, 0xff,
116    0x00, 0x00, 0x00, 0x7f, 0x00, 0x18, 0xfe, 0x7f, 0x00, 0x00, 0x00, 0x00,
117    0x00, 0x18, 0xfe, 0x7f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x18, 0xfc, 0x7f,
118    0x00, 0x00, 0x00, 0x00, 0x00, 0x38, 0xfc, 0x7f, 0x00, 0x00, 0x00, 0x00,
119    0x00, 0x30, 0xfc, 0x3f, 0x00, 0x00, 0x00, 0x00, 0xf0, 0xfc, 0x3f,
120    0x00, 0x00, 0x00, 0x00, 0x00, 0xe0, 0xff, 0x1f, 0x00, 0x00, 0x00, 0x00,
121    0x00, 0x80, 0xff, 0x07, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xfe, 0x01,
122    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x78, 0x00
123 };
124 //-----
125 #define Feuchtigkeit_links_unten_width 15
126 #define Feuchtigkeit_links_unten_height 15
127
128 static unsigned char Feuchtigkeit_links_unten_bits[] = {
129     0x00, 0x1f, 0x80, 0x3f, 0xc0, 0x71, 0xc0, 0x64, 0xe0, 0x6e, 0xe0, 0x64,
130     0xf0, 0x71, 0xf0, 0x3f, 0xf8, 0x1f, 0xf8, 0x07, 0xfc, 0x01, 0x7e, 0x00,
131     0x1f, 0x00, 0x0f, 0x00, 0x07, 0x00
132 };
133 //-----
134 #define Feuchtigkeit_links_mitte_width 19
135 #define Feuchtigkeit_links_mitte_height 9
136
137 static unsigned char Feuchtigkeit_links_mitte_bits[] = {
138     0x00, 0xf0, 0x01, 0x00, 0xfe, 0x03, 0xc0, 0x1f, 0x07, 0xfe, 0x4f, 0x06,
139     0xff, 0xef, 0x06, 0xfe, 0x4f, 0x06, 0xc0, 0x1f, 0x07, 0x00, 0xfe, 0x03,
140     0x00, 0xf0, 0x01
141 };
142 //-----
143 #define Feuchtigkeit_links_oben_width 15
144 #define Feuchtigkeit_links_oben_height 15
145
146 static unsigned char Feuchtigkeit_links_oben_bits[] = {

```

```

147     0x07, 0x00, 0x0f, 0x00, 0x1f, 0x00, 0x7e, 0x00, 0xfc, 0x01, 0xf8, 0x07,
148     0xf8, 0x1f, 0xf0, 0x3f, 0xf0, 0x71, 0xe0, 0x64, 0xe0, 0x6e, 0xc0, 0x64,
149     0xc0, 0x71, 0x80, 0x3f, 0x00, 0x1f
150 };
151 //-----
152 #define Feuchtigkeit_oben_width 9
153 #define Feuchtigkeit_oben_height 19
154
155 static unsigned char Feuchtigkeit_oben_bits[] = {
156     0x10, 0x00, 0x38, 0x00, 0x38, 0x00, 0x38, 0x00, 0x38, 0x00, 0x38, 0x00,
157     0x7c, 0x00, 0x7c, 0x00, 0x7c, 0x00, 0xfe, 0x00, 0xfe, 0x00, 0xfe, 0x00,
158     0xc7, 0x01, 0x93, 0x01, 0xbb, 0x01, 0x93, 0x01, 0xc7, 0x01, 0xfe, 0x00,
159     0x7c, 0x00
160 };
161 //-----
162 #define Feuchtigkeit_rechts_oben_width 15
163 #define Feuchtigkeit_rechts_oben_height 15
164
165 static unsigned char Feuchtigkeit_rechts_oben_bits[] = {
166     0x00, 0x70, 0x00, 0x78, 0x00, 0x7c, 0x00, 0x3f, 0xc0, 0x1f, 0xf0, 0x0f,
167     0xfc, 0x0f, 0xfe, 0x07, 0xc7, 0x07, 0x93, 0x03, 0xbb, 0x03, 0x93, 0x01,
168     0xc7, 0x01, 0xfe, 0x00, 0x7c, 0x00
169 };
170 //-----
171 #define Feuchtigkeit_rechts_mitte_width 19
172 #define Feuchtigkeit_rechts_mitte_height 9
173
174 static unsigned char Feuchtigkeit_rechts_mitte_bits[] = {
175     0x7c, 0x00, 0x00, 0xfe, 0x03, 0x00, 0xc7, 0x1f, 0x00, 0x93, 0xff, 0x03,
176     0xbb, 0xff, 0x07, 0x93, 0xff, 0x03, 0xc7, 0x1f, 0x00, 0xfe, 0x03, 0x00,
177     0x7c, 0x00, 0x00
178 };
179 //-----
180 #define Feuchtigkeit_rechts_unten_width 15
181 #define Feuchtigkeit_rechts_unten_height 15
182
183 static unsigned char Feuchtigkeit_rechts_unten_bits[] = {
184     0x7c, 0x00, 0xfe, 0x00, 0xc7, 0x01, 0x93, 0x01, 0xbb, 0x03, 0x93, 0x03,
185     0xc7, 0x07, 0xfe, 0x07, 0xfc, 0x0f, 0xf0, 0x0f, 0xc0, 0x1f, 0x00, 0x3f,
186     0x00, 0x7c, 0x00, 0x78, 0x00, 0x70
187 };
188 //-----
189 #define Grad_width 6
190 #define Grad_height 6
191
192 static unsigned char Grad_bits[] = {
193     0x1e, 0x33, 0x21, 0x21, 0x33, 0x1e
194 };
195 //-----
196 #define Temperatur_leer_width 64
197 #define Temperatur_leer_height 64
198
199 static unsigned char Temperatur_leer_bits[] = {
200     0x00, 0x00, 0x00, 0x3e, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x80, 0xff,
201     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xc0, 0xe3, 0x01, 0x00, 0x00, 0x00,
202     0x00, 0x00, 0xc0, 0x80, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00, 0xe0, 0x80,
203     0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x60, 0x00, 0x03, 0x00, 0x00, 0x00,
204     0x00, 0x00, 0x60, 0x00, 0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x60, 0x00,
205     0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x60, 0x00, 0x83, 0xff, 0x03, 0x00,
206     0x00, 0x00, 0x60, 0x00, 0x83, 0xff, 0x03, 0x00, 0x00, 0x00, 0x00, 0x60, 0x00,
207     0x83, 0xff, 0x03, 0x00, 0x00, 0x00, 0x60, 0x00, 0x03, 0x00, 0x00, 0x00,
208     0x00, 0x00, 0x60, 0x00, 0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x60, 0x00,
209     0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x60, 0x00, 0x83, 0x1f, 0x00, 0x00,
210     0x00, 0x00, 0x60, 0x00, 0x83, 0x1f, 0x00, 0x00, 0x00, 0x00, 0x60, 0x00,
211     0x83, 0x1f, 0x00, 0x00, 0x00, 0x00, 0x60, 0x00, 0x03, 0x00, 0x00, 0x00,
212     0x00, 0x00, 0x60, 0x00, 0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x60, 0x00,
213     0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x60, 0x00, 0x83, 0xff, 0x03, 0x00,
214     0x00, 0x00, 0x60, 0x00, 0x83, 0xff, 0x03, 0x00, 0x00, 0x00, 0x60, 0x00,
215     0x83, 0xff, 0x03, 0x00, 0x00, 0x00, 0x60, 0x00, 0x03, 0x00, 0x00, 0x00,
216     0x00, 0x00, 0x60, 0x00, 0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x60, 0x00,
217     0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x60, 0x00, 0x83, 0x1f, 0x00, 0x00,
218     0x00, 0x00, 0x60, 0x00, 0x83, 0x1f, 0x00, 0x00, 0x00, 0x00, 0x60, 0x00,
219     0x83, 0x1f, 0x00, 0x00, 0x00, 0x00, 0x60, 0x00, 0x03, 0x00, 0x00, 0x00,

```

```
220 0x00, 0x00, 0x60, 0x00, 0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x60, 0x00,
221 0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x60, 0x00, 0x83, 0xff, 0x03, 0x00,
222 0x00, 0x00, 0x60, 0x00, 0x83, 0xff, 0x03, 0x00, 0x00, 0x00, 0x60, 0x00,
223 0x83, 0xff, 0x03, 0x00, 0x00, 0x00, 0x60, 0x00, 0x03, 0x00, 0x00, 0x00,
224 0x00, 0x00, 0x60, 0x00, 0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x60, 0x00,
225 0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x60, 0x00, 0x03, 0x00, 0x00, 0x00,
226 0x00, 0x00, 0x70, 0x00, 0x07, 0x00, 0x00, 0x00, 0x00, 0x00, 0x38, 0x00,
227 0x0e, 0x00, 0x00, 0x00, 0x00, 0x00, 0x0c, 0x00, 0x18, 0x00, 0x00, 0x00,
228 0x00, 0x00, 0x06, 0x7f, 0x30, 0x00, 0x00, 0x00, 0x00, 0x00, 0xc3, 0xf1,
229 0x61, 0x00, 0x00, 0x00, 0x00, 0x00, 0x63, 0xe0, 0x63, 0x00, 0x00, 0x00,
230 0x00, 0x80, 0x31, 0xe0, 0xc7, 0x00, 0x00, 0x00, 0x00, 0x80, 0x11, 0xf0,
231 0xc7, 0x00, 0x00, 0x00, 0x00, 0x80, 0x19, 0xfc, 0xcf, 0x00, 0x00, 0x00,
232 0x00, 0xc0, 0x08, 0xfe, 0x8f, 0x01, 0x00, 0x00, 0x00, 0xc0, 0x08, 0xff,
233 0x8f, 0x01, 0x00, 0x00, 0x00, 0xc0, 0x08, 0xff, 0x8f, 0x01, 0x00, 0x00,
234 0x00, 0xc0, 0x98, 0xff, 0x8f, 0x01, 0x00, 0x00, 0x00, 0xc0, 0xf8, 0xff,
235 0x8f, 0x01, 0x00, 0x00, 0x00, 0x80, 0xf9, 0xff, 0xcf, 0x00, 0x00, 0x00,
236 0x00, 0x80, 0xf1, 0xff, 0xc7, 0x00, 0x00, 0x00, 0x00, 0x80, 0xf1, 0xff,
237 0xc7, 0x00, 0x00, 0x00, 0x00, 0x00, 0xe3, 0xff, 0x63, 0x00, 0x00, 0x00,
238 0x00, 0x00, 0xc3, 0xff, 0x61, 0x00, 0x00, 0x00, 0x00, 0x00, 0x06, 0x7f,
239 0x30, 0x00, 0x00, 0x00, 0x00, 0x00, 0x0c, 0x00, 0x18, 0x00, 0x00, 0x00,
240 0x00, 0x00, 0x38, 0x00, 0x0e, 0x00, 0x00, 0x00, 0x00, 0x00, 0xf0, 0xc1,
241 0x07, 0x00, 0x00, 0x00, 0x00, 0x00, 0xc0, 0xff, 0x01, 0x00, 0x00, 0x00,
242 0x00, 0x00, 0x00, 0x3e, 0x00, 0x00, 0x00, 0x00
```

```
243 };
```

```
244 //-----
```

```
245 #define CO2_width 64
```

```
246 #define CO2_height 64
```

```
247
```

```
248 static unsigned char CO2_bits[] = {
```

```
249 0x00, 0x00, 0x00, 0x00, 0x00, 0x3c, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
250 0x80, 0xff, 0x01, 0x00, 0x00, 0x00, 0x1f, 0xe0, 0xc0, 0xff, 0x03, 0x00,
251 0x00, 0xc0, 0x7f, 0xfc, 0xc7, 0xff, 0x03, 0x00, 0x00, 0xe0, 0xff, 0xff,
252 0xef, 0xff, 0x07, 0x00, 0x00, 0xf0, 0xff, 0xff, 0xff, 0x07, 0x00,
253 0x00, 0xf0, 0xff, 0xff, 0xff, 0xff, 0xf0, 0x00, 0x00, 0xf8, 0xff, 0xff,
254 0xff, 0xff, 0x3f, 0x00, 0x00, 0xf8, 0xff, 0xff, 0xff, 0xff, 0xff, 0x00,
255 0x00, 0xf8, 0x07, 0xfc, 0x0f, 0xf8, 0xff, 0x03, 0x00, 0xf8, 0x01, 0xf0,
256 0x03, 0xe0, 0xff, 0x07, 0x00, 0xf8, 0x00, 0xe0, 0x01, 0xc0, 0xff, 0x0f,
257 0x00, 0xf0, 0xf0, 0xe1, 0xe1, 0xc3, 0xff, 0x0f, 0x00, 0x70, 0xf8, 0xe3,
258 0xf0, 0x87, 0xff, 0x1f, 0x00, 0x78, 0xf8, 0xe3, 0xf0, 0x87, 0xff, 0x1f,
259 0x00, 0x7e, 0xfc, 0xf7, 0xf8, 0x8f, 0xff, 0x1f, 0x00, 0x7f, 0xfc, 0xff,
260 0xf8, 0x8f, 0xff, 0x1f, 0x80, 0x7f, 0xfc, 0xff, 0xf8, 0x8f, 0xc1, 0x1f,
261 0x80, 0x7f, 0xfc, 0xff, 0xf8, 0x8f, 0x80, 0x1f, 0xc0, 0x7f, 0xfc, 0xff,
262 0xf8, 0x8f, 0x9c, 0x1f, 0xc0, 0x7f, 0xfc, 0xf7, 0xf8, 0x8f, 0x9f, 0x0f,
263 0xc0, 0x7f, 0xf8, 0xe3, 0xf0, 0x87, 0x8f, 0x0f, 0xc0, 0x7f, 0xf8, 0xe3,
264 0xf0, 0x87, 0xc7, 0x07, 0x80, 0xff, 0xf0, 0xe1, 0xe1, 0xc3, 0xe3, 0x07,
265 0x80, 0xff, 0x00, 0xe0, 0x01, 0xc0, 0xf1, 0x07, 0x00, 0xff, 0x01, 0xf0,
266 0x03, 0xe0, 0xf8, 0x0f, 0x80, 0xff, 0x07, 0xfc, 0x0f, 0xf8, 0x80, 0x0f,
267 0xc0, 0xff, 0xff, 0xff, 0xff, 0xff, 0x80, 0x0f, 0xc0, 0xff, 0xfe, 0xff,
268 0xff, 0xff, 0xff, 0x0f, 0xc0, 0x7f, 0xfc, 0xff, 0xff, 0xff, 0xff, 0xff, 0x0f,
269 0x80, 0x1f, 0xfc, 0xff, 0xff, 0xff, 0xff, 0x07, 0x00, 0x0f, 0xf8, 0xff,
270 0xff, 0xff, 0xff, 0x07, 0x00, 0x00, 0xf0, 0xff, 0xff, 0xff, 0xff, 0x03,
271 0x00, 0x00, 0xc0, 0xff, 0xff, 0xdf, 0xff, 0x01, 0x00, 0x00, 0x00, 0xf0,
272 0xff, 0x8f, 0xff, 0x00, 0x80, 0x1f, 0x00, 0xe0, 0xff, 0x07, 0x3e, 0x00,
273 0x80, 0x1f, 0x00, 0x80, 0xff, 0x01, 0x00, 0x00, 0x80, 0x1f, 0x00, 0x00,
274 0x3c, 0x00, 0x00, 0x00, 0x80, 0x1f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
275 0x80, 0x1f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x80, 0x1f, 0x00, 0x00,
276 0x00, 0x00, 0x00, 0x00, 0x80, 0x1f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
277 0x80, 0x1f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x80, 0x1f, 0x00, 0x00,
278 0x00, 0x00, 0x00, 0x00, 0x80, 0x1f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
279 0xc0, 0x3f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xc0, 0x3f, 0x00, 0x00,
280 0x00, 0x00, 0x00, 0x00, 0xc0, 0x3f, 0xff, 0xff, 0xff, 0x00, 0x00, 0xc0, 0x3f, 0xff, 0xff,
281 0xc0, 0x3f, 0xff, 0xff, 0xff, 0x00, 0x00, 0x00, 0xc0, 0x3f, 0xff, 0xff,
282 0xff, 0x00, 0x00, 0x00, 0xc0, 0x3f, 0xff, 0xff, 0xff, 0x00, 0x00, 0x00,
283 0xc0, 0x3f, 0x8f, 0x31, 0xc6, 0x00, 0x00, 0x00, 0xc0, 0x3f, 0x8f, 0x31,
284 0xc6, 0xfc, 0x7f, 0x00, 0xe0, 0x7f, 0x8e, 0x31, 0xc6, 0xfc, 0x7f, 0x00,
285 0xe0, 0x7f, 0x8e, 0x31, 0xc6, 0xfc, 0x7f, 0x00, 0xe0, 0x7f, 0x8e, 0x31,
286 0xc6, 0xfc, 0x7f, 0x00, 0xe0, 0x7f, 0xfe, 0xff, 0xff, 0xfc, 0x7f, 0x00,
287 0xe0, 0x7f, 0xfe, 0xff, 0xff, 0xfc, 0x7f, 0x00, 0xe0, 0x7f, 0xfe, 0xff,
288 0xff, 0xfc, 0x7f, 0x00, 0xf0, 0xff, 0xfc, 0xff, 0xff, 0xfc, 0x7f, 0x00,
289 0xf0, 0xff, 0xfc, 0xff, 0xff, 0xfc, 0x7f, 0x00, 0xf0, 0xff, 0xfc, 0xff,
290 0xff, 0xfc, 0x7f, 0x00, 0xf0, 0xff, 0xfc, 0xff, 0xfc, 0x7f, 0x00,
291 0xf0, 0xff, 0xfc, 0xff, 0xff, 0xfc, 0x7f, 0x00
```

```
292 };
```

```

293 //-----
294 #define Ton_leer_width 32
295 #define Ton_leer_height 32
296
297 static unsigned char Ton_leer_bits[] = {
298     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
299     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
300     0x00, 0xe0, 0x00, 0x00, 0x00, 0xf0, 0x00, 0x00, 0x00, 0xf8, 0x00, 0x00,
301     0x00, 0xfc, 0x00, 0x00, 0x00, 0xfe, 0x00, 0x00, 0xfe, 0xff, 0x00, 0x00,
302     0xff, 0xff, 0x00, 0x00, 0xff, 0xff, 0x00, 0x00, 0xff, 0xff, 0x00, 0x00,
303     0xff, 0xff, 0x00, 0x00, 0xff, 0xff, 0x00, 0x00, 0xff, 0xff, 0x00, 0x00,
304     0xff, 0xff, 0x00, 0x00, 0xff, 0xff, 0x00, 0x00, 0xfe, 0xff, 0x00, 0x00,
305     0x00, 0xfe, 0x00, 0x00, 0x00, 0xfc, 0x00, 0x00, 0x00, 0xf8, 0x00, 0x00,
306     0x00, 0xf0, 0x00, 0x00, 0x00, 0xe0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
307     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
308     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
309 };
310 //-----
311 #define Ton_welle_1_width 3
312 #define Ton_welle_1_height 4
313
314 static unsigned char Ton_welle_1_bits[] = {
315     0x03, 0x06, 0x06, 0x03
316 };
317 //-----
318 #define Ton_welle_2_width 8
319 #define Ton_welle_2_height 12
320
321 static unsigned char Ton_welle_2_bits[] = {
322     0x18, 0x30, 0x60, 0x60, 0xc3, 0xc6, 0xc6, 0xc3, 0x60, 0x60, 0x30, 0x18
323 };
324 //-----
325 #define Ton_welle_3_width 12
326 #define Ton_welle_3_height 20
327
328 static unsigned char Ton_welle_3_bits[] = {
329     0xc0, 0x00, 0x80, 0x01, 0x00, 0x03, 0x00, 0x03, 0x18, 0x06, 0x30, 0x06,
330     0x60, 0x0c, 0x60, 0x0c, 0xc3, 0x0c, 0xc6, 0x0c, 0xc6, 0x0c, 0xc3, 0x0c,
331     0x60, 0x0c, 0x60, 0x0c, 0x30, 0x06, 0x18, 0x06, 0x00, 0x03, 0x00, 0x03,
332     0x80, 0x01, 0xc0, 0x00
333 };
334 //-----
335 #define Ton_mute_width 32
336 #define Ton_mute_height 32
337
338 static unsigned char Ton_mute_bits[] = {
339     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
340     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
341     0x0c, 0xe0, 0x00, 0x0c, 0x1c, 0xf0, 0x00, 0x18, 0x70, 0xf8, 0x00, 0x30,
342     0xe0, 0xf8, 0x00, 0x30, 0x80, 0xf3, 0x80, 0x61, 0xfe, 0xc7, 0x00, 0x63,
343     0xff, 0x9f, 0x00, 0xc6, 0xff, 0x3f, 0x00, 0xc6, 0xff, 0xff, 0x30, 0xcc,
344     0xff, 0xff, 0x61, 0xcc, 0xff, 0xff, 0x47, 0xcc, 0xff, 0xff, 0x0e, 0xcc,
345     0xff, 0xff, 0x38, 0xc6, 0xff, 0xff, 0x70, 0xc4, 0xfe, 0xff, 0xc0, 0x61,
346     0x00, 0xfe, 0x80, 0x43, 0x00, 0xfc, 0x00, 0x0e, 0x00, 0x00, 0xf8, 0x00, 0x1c,
347     0x00, 0xf0, 0x00, 0x78, 0x00, 0xe0, 0x00, 0x6c, 0x00, 0x00, 0x00, 0x00,
348     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
349     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00
350 };
351 //-----
352 #define Bild_width 32
353 #define Bild_height 32
354
355 static unsigned char Bild_bits[] = {
356     0xe0, 0xff, 0xff, 0x07, 0xf8, 0xff, 0xff, 0x1f, 0xfc, 0xff, 0xff, 0x3f,
357     0x7e, 0x00, 0x00, 0x7e, 0x1e, 0x00, 0x00, 0x78, 0x0f, 0x00, 0x00, 0xf0,
358     0x0f, 0x00, 0x00, 0xf0, 0x07, 0x00, 0x00, 0xe0, 0x07, 0x06, 0x00, 0xe0,
359     0x07, 0x0f, 0x00, 0xe0, 0x07, 0x0f, 0x04, 0xe0, 0x07, 0x06, 0x04, 0xe0,
360     0x07, 0x00, 0x0e, 0xe0, 0x07, 0x00, 0x0e, 0xe0, 0x07, 0x00, 0x1f, 0xe0,
361     0x07, 0x00, 0x1f, 0xe0, 0x07, 0x80, 0x3f, 0xe0, 0x07, 0x84, 0x3f, 0xe0,
362     0x07, 0xc4, 0x7f, 0xe0, 0x07, 0xce, 0x7f, 0xe0, 0x07, 0xee, 0xff, 0xe0,
363     0x07, 0xff, 0xff, 0xe0, 0x07, 0xff, 0xff, 0xe1, 0x87, 0xff, 0xff, 0xe1,
364     0x87, 0xff, 0xff, 0xe1, 0x0f, 0xff, 0xff, 0xf0, 0x0f, 0x00, 0x00, 0xf0,
365     0x1e, 0x00, 0x00, 0x78, 0x7e, 0x00, 0x00, 0x7e, 0xfc, 0xff, 0xff, 0x3f,

```

```

366     0xf8, 0xff, 0xff, 0x1f, 0xe0, 0xff, 0xff, 0x07
367 };
368 //-----
369 #define Text_width 32
370 #define Text_height 32
371
372 static unsigned char Text_bits[] = {
373     0xf0, 0xff, 0x1f, 0x00, 0xf8, 0xff, 0x3f, 0x00, 0x38, 0x00, 0x73, 0x00,
374     0x18, 0x00, 0xe3, 0x00, 0x18, 0x00, 0xc3, 0x01, 0x18, 0x00, 0x83, 0x03,
375     0x18, 0x00, 0x03, 0x07, 0x18, 0x00, 0x03, 0x0e, 0x18, 0x00, 0x03, 0x1c,
376     0x18, 0x00, 0x03, 0x18, 0x18, 0x00, 0x07, 0x18, 0x18, 0x00, 0xff, 0x1f,
377     0x18, 0x00, 0xfe, 0x1f, 0x18, 0x00, 0x00, 0x18, 0x18, 0x00, 0x00, 0x18,
378     0x18, 0x00, 0x00, 0x18, 0x18, 0xff, 0xff, 0x18, 0x18, 0xff, 0xff, 0x18,
379     0x18, 0xff, 0xff, 0x18, 0x18, 0x00, 0x00, 0x18, 0x18, 0x00, 0x00, 0x18,
380     0x18, 0x00, 0x00, 0x18, 0x18, 0xff, 0xff, 0x18, 0x18, 0xff, 0xff, 0x18,
381     0x18, 0xff, 0xff, 0x18, 0x18, 0x00, 0x00, 0x18, 0x18, 0x00, 0x00, 0x18,
382     0x18, 0x00, 0x00, 0x18, 0x18, 0x00, 0x00, 0x18, 0x38, 0x00, 0x00, 0x1c,
383     0xf8, 0xff, 0xff, 0x1f, 0xf0, 0xff, 0xff, 0x0f
384 };
385 //-----
386 #define HTL_Wien_West_Logo_128x64_width 128
387 #define HTL_Wien_West_Logo_128x64_height 64
388
389 static unsigned char HTL_Wien_West_Logo_128x64_bits[] = {
390     0xff, 0xff, 0x7f, 0x00, 0x00, 0x00, 0xf8, 0xff, 0xff, 0x03, 0x1c, 0x70,
391     0xf0, 0xff, 0xc1, 0x01, 0x07, 0x00, 0x80, 0x01, 0x00, 0x00, 0x06, 0x00,
392     0x80, 0x03, 0x1c, 0x70, 0xf0, 0xff, 0xc1, 0x01, 0x19, 0x00, 0x00, 0x06,
393     0x00, 0x80, 0x01, 0x00, 0x60, 0x02, 0x1c, 0x70, 0xf0, 0xff, 0xc1, 0x01,
394     0x61, 0x00, 0x00, 0x18, 0x00, 0x60, 0x00, 0x00, 0x18, 0x02, 0x1c, 0x70,
395     0x00, 0x0e, 0xc0, 0x01, 0x81, 0x01, 0x00, 0x60, 0x00, 0x18, 0x00, 0x00,
396     0x06, 0x02, 0x1c, 0x70, 0x00, 0x0e, 0xc0, 0x01, 0x01, 0x06, 0x00, 0x80,
397     0x01, 0x06, 0x00, 0x80, 0x01, 0x02, 0x1c, 0x70, 0x00, 0x0e, 0xc0, 0x01,
398     0x01, 0xf8, 0xff, 0xff, 0x01, 0xfe, 0xff, 0x7f, 0x00, 0x02, 0x1c, 0x70,
399     0x00, 0x0e, 0xc0, 0x01, 0x01, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40,
400     0x00, 0x02, 0xfc, 0x7f, 0x00, 0x0e, 0xc0, 0x01, 0x01, 0x08, 0x00, 0x00,
401     0x01, 0x02, 0x00, 0x40, 0x00, 0x02, 0xfc, 0x7f, 0x00, 0x0e, 0xc0, 0x01,
402     0x01, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40, 0x00, 0x02, 0xfc, 0x7f,
403     0x00, 0x0e, 0xc0, 0x01, 0x01, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40,
404     0x00, 0x02, 0x1c, 0x70, 0x00, 0x0e, 0xc0, 0x01, 0x01, 0x08, 0x00, 0x00,
405     0x01, 0x02, 0x00, 0x40, 0x00, 0x02, 0x1c, 0x70, 0x00, 0x0e, 0xc0, 0x01,
406     0x01, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40, 0x00, 0x02, 0x1c, 0x70,
407     0x00, 0x0e, 0xc0, 0x01, 0x01, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40,
408     0x00, 0x02, 0x1c, 0x70, 0x00, 0x0e, 0xc0, 0x01, 0x01, 0x08, 0x00, 0x00,
409     0x01, 0x02, 0x00, 0x40, 0x00, 0x02, 0x1c, 0x70, 0x00, 0x0e, 0xc0, 0x7f,
410     0x01, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40, 0x00, 0x02, 0x1c, 0x70,
411     0x00, 0x0e, 0xc0, 0x7f, 0x01, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40,
412     0x00, 0x02, 0x1c, 0x70, 0x00, 0x0e, 0xc0, 0x7f, 0x01, 0x08, 0x00, 0x00,
413     0x01, 0x02, 0x00, 0x40, 0x00, 0x02, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
414     0x01, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40, 0x00, 0x02, 0x00, 0x00,
415     0x00, 0x00, 0x00, 0x00, 0x01, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40,
416     0x00, 0x02, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x01, 0x08, 0x00, 0x00,
417     0x01, 0x02, 0x00, 0x40, 0x00, 0x02, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
418     0x01, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40, 0x00, 0x02, 0x00, 0x00,
419     0x00, 0x00, 0x00, 0x00, 0x03, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40,
420     0x00, 0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x0c, 0x08, 0x00, 0x00,
421     0x01, 0x02, 0x00, 0x40, 0xc0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
422     0x30, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40, 0x30, 0x00, 0x00, 0x00,
423     0x00, 0x00, 0x00, 0x00, 0xc0, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40,
424     0xc0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x0b, 0x00, 0x00,
425     0x01, 0x02, 0x00, 0x40, 0x03, 0x00, 0x06, 0x83, 0x31, 0xfc, 0xc7, 0xc0,
426     0x00, 0xfc, 0xff, 0xff, 0x01, 0xfe, 0xff, 0xff, 0x00, 0x00, 0x06, 0x83,
427     0x31, 0xfc, 0xc7, 0xc1, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
428     0x00, 0x00, 0x06, 0x83, 0x31, 0xc0, 0xc0, 0xc3, 0x00, 0x00, 0x00, 0x00,
429     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x04, 0x83, 0x30, 0xc0, 0xc0, 0xc3,
430     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x8c, 0xc7,
431     0x30, 0xc0, 0xc0, 0xc6, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
432     0x00, 0x00, 0x8c, 0xc7, 0x30, 0xc0, 0xc0, 0xc6, 0x00, 0x00, 0x00, 0x00,
433     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x8c, 0xc4, 0x30, 0xfc, 0xc3, 0xcc,
434     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xc8, 0x4c,
435     0x30, 0xfc, 0xc3, 0xcc, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
436     0x00, 0x00, 0xd8, 0x6c, 0x30, 0xc0, 0xc0, 0xd8, 0x00, 0x00, 0x00, 0x00,
437     0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xd8, 0x6c, 0x30, 0xc0, 0xc0, 0xd8,
438     0x00, 0xfc, 0xff, 0xff, 0x01, 0xfe, 0xff, 0xff, 0x00, 0x00, 0x58, 0x68,

```

```

439 0x30, 0x0c, 0xc0, 0xf0, 0x00, 0x0b, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40,
440 0x03, 0x00, 0x70, 0x38, 0x30, 0x0c, 0xc0, 0xf0, 0xc0, 0x08, 0x00, 0x00,
441 0x01, 0x02, 0x00, 0x40, 0x0c, 0x00, 0x70, 0x38, 0x30, 0xfc, 0xc7, 0xe0,
442 0x30, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40, 0x30, 0x00, 0x20, 0x10,
443 0x30, 0xfc, 0xc7, 0xc0, 0x0c, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40,
444 0xc0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x03, 0x08, 0x00, 0x00,
445 0x01, 0x02, 0x00, 0x40, 0x00, 0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
446 0x01, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40, 0x00, 0x02, 0x00, 0x00,
447 0x00, 0x00, 0x00, 0x00, 0x01, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40,
448 0x00, 0x02, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x01, 0x08, 0x00, 0x00,
449 0x01, 0x02, 0x00, 0x40, 0x00, 0x02, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
450 0x01, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40, 0x00, 0x02, 0x00, 0x00,
451 0x00, 0x00, 0x00, 0x00, 0x01, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40,
452 0x00, 0x02, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x01, 0x08, 0x00, 0x00,
453 0x01, 0x02, 0x00, 0x40, 0x00, 0x02, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
454 0x01, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40, 0x00, 0x02, 0x00, 0x00,
455 0x00, 0x00, 0x00, 0x00, 0x01, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40,
456 0x00, 0x02, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x01, 0x08, 0x00, 0x00,
457 0x01, 0x02, 0x00, 0x40, 0x00, 0x02, 0x06, 0x83, 0xfd, 0xc7, 0xcf, 0xff,
458 0x01, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40, 0x00, 0x02, 0x06, 0x83,
459 0xfd, 0xe7, 0xdf, 0xff, 0x01, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40,
460 0x00, 0x02, 0x06, 0x83, 0x0d, 0x70, 0x18, 0x0c, 0x01, 0x08, 0x00, 0x00,
461 0x01, 0x02, 0x00, 0x40, 0x00, 0x02, 0x04, 0x83, 0x0c, 0x30, 0x00, 0x0c,
462 0x01, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40, 0x00, 0x02, 0x8c, 0xc7,
463 0x0c, 0x30, 0x00, 0x0c, 0x01, 0x08, 0x00, 0x00, 0x01, 0x02, 0x00, 0x40,
464 0x00, 0x02, 0x8c, 0xc7, 0x0c, 0x70, 0x00, 0x0c, 0x01, 0x08, 0x00, 0x00,
465 0x01, 0x02, 0x00, 0x40, 0x00, 0x02, 0x8c, 0xc4, 0xfc, 0xe3, 0x07, 0x0c,
466 0x01, 0xf8, 0xff, 0xff, 0x01, 0xfe, 0xff, 0x7f, 0x00, 0x02, 0xc8, 0x4c,
467 0xfc, 0xc3, 0x0f, 0x0c, 0x01, 0x06, 0x00, 0x80, 0x01, 0x06, 0x00, 0x80,
468 0x01, 0x02, 0xd8, 0x6c, 0x0c, 0x00, 0x1c, 0x0c, 0x81, 0x01, 0x00, 0x60,
469 0x00, 0x18, 0x00, 0x00, 0x06, 0x02, 0xd8, 0x6c, 0x0c, 0x00, 0x18, 0x0c,
470 0x61, 0x00, 0x00, 0x18, 0x00, 0x60, 0x00, 0x00, 0x18, 0x02, 0x58, 0x68,
471 0x0c, 0x00, 0x18, 0x0c, 0x19, 0x00, 0x00, 0x06, 0x00, 0x80, 0x01, 0x00,
472 0x60, 0x02, 0x70, 0x38, 0x0c, 0x30, 0x1c, 0x0c, 0x07, 0x00, 0x80, 0x01,
473 0x00, 0x00, 0x06, 0x00, 0x80, 0x03, 0x70, 0x38, 0xfc, 0xf7, 0x0f, 0x0c,
474 0xff, 0xff, 0x7f, 0x00, 0x00, 0x00, 0xf8, 0xff, 0xff, 0x03, 0x20, 0x10,
475 0xfc, 0xe7, 0x07, 0x0c

```

```

476 };
477 //-----
478 bool buff = false;
479 unsigned long getDataTimer = 0;
480 int buff_count = 0;
481 byte timer = display_switch_delay/(32 /* >4 */);
482 //-----
483 void setup() {
484   pinMode(buzzer_pin, OUTPUT);
485   pinMode(buzzer_taster_pin, INPUT);
486   Serial.begin(115200);
487   u8g2.begin();
488   bar.begin();
489   dht.begin();
490   mySerial.begin(baudrate);
491   myMHZ19.begin(mySerial);
492   myMHZ19.autoCalibration(true, 24);
493   myMHZ19.setRange(5000);
494   buzzer_buff = myMHZ19.getCO2()/100*100;
495   Einleitung();
496 }
497
498 void loop() {
499   if(myMHZ19.getCO2() > 0) carbondioxide = myMHZ19.getCO2();
500   if(dht.readTemperature() > 0) temperature = dht.readTemperature();
501   if(dht.readHumidity() > 0) humidity = dht.readHumidity();
502
503   Display_Clock();
504   LED_Bar();
505   Buzzer();
506 }
507
508 void Display_Clock() {
509   if(analogRead(taster_analog_pin) < 256) display_taster = 1;
510   if(analogRead(taster_analog_pin) > 256 & analogRead(taster_analog_pin) < 768)
511     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.drawLine(26, 42-round(temperature), round(temperature));
567                         u8g2.drawLine(27, 42-round(temperature), round(temperature));
568                         u8g2.drawLine(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,

```





```

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_timeout & buzzer_state == false
    & carbondioxide >= buzzer_start) {
682         buzzer_state = true;
683     }
684     Serial.print("Time: " + String(millis() - buzzer_timeout_getDataTimer) +
        " // buzzer_state: " + String(buzzer_state) +
685         " // CO2: " + String(carbondioxide));
686     if(carbondioxide >= buzzer_buff+100) {
687         if(buzzer_state == true & digitalRead(buzzer_taster_pin) == HIGH) {
688             if(buzzer_delay_buff == 0 or (millis() - buzzer_delay_getDataTimer >= 50 &
        (buzzer_delay_buff == 2 or buzzer_delay_buff == 4))) {
689                 digitalWrite(buzzer_pin, HIGH);
690                 if(buzzer_buff <= buzzer_mode_advanced) buzzer_delay_buff = 5;
691                 else buzzer_delay_buff++;
692                 buzzer_delay_getDataTimer = millis();
693             }
694             if(millis() - buzzer_delay_getDataTimer >= 200 & (buzzer_delay_buff == 1 or
        buzzer_delay_buff == 3 or buzzer_delay_buff == 5)) {
695                 digitalWrite(buzzer_pin, LOW);
696                 if(buzzer_delay_buff == 5) {
697                     buzzer_state = false;
698                     buzzer_delay_buff = 0;
699                     buzzer_timeout_getDataTimer = millis();
700                 }
701                 else {
702                     buzzer_delay_buff++;
703                     buzzer_delay_getDataTimer = millis();
704                 }
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,
730                                 HTL_Wien_West_Logo_128x64_height, HTL_Wien_West_Logo_128x64_bits);
731                     break;
732                 case 1:
733                     u8g2.setFont(u8g2_font_ncenB10_tr);
734                     u8g2.setCursor(10,21);
735                     u8g2.print("Entwickelt von");
736                     u8g2.setCursor(42,42);
737                     u8g2.print("Jakob");
738                     u8g2.setCursor(22,63);
739                     u8g2.print("Estermann");
740                     break;
741                 case 2:
742                     u8g2.drawXBMP(0, 0, HTL_Wien_West_Logo_128x64_width,
743                                 HTL_Wien_West_Logo_128x64_height, HTL_Wien_West_Logo_128x64_bits);
744                     break;
745             }
746         } while(u8g2.nextPage());
747     }
748     delay(einleitung_delay);
749 }

```