

esp32 oszilloskop Uhr

Datum: 2025-12-22

Oszilloskop im XY-Modus als Uhrenanzeige.

Die Spannungen werden mit dem esp32 erzeugt, weil

- der über WLAN die Uhrzeit holen kann und
- zwei Digital zu Analog Wandler hat.

ESP32 Osziclock using DAC1 and DAC2, pin 25 and 26.

Contents

1 Inbetriebnahme	1
2 TODO	2
2.1 Plot different things	2
3 Übernehmen des alten Projekts und testen	2
3.1 Change 01	6
3.2 Change 02	6
3.3 Change 03	7
3.4 Change 04: Sommer/Winterzeit aus NTP	8
3.5 Change 05: Konfiguration	8
4 Erweiterungen/Umbauten	9
4.1 Draw something: xmas mode	9
4.2 Snowfall, NOT YET	10
4.3 Cleanup: remove coordinate duplicates from draw	11
4.3.1 (2025-01-01) Versuch zu Hause, kompiliert und läuft (keine optische Kontrolle)	12
4.3.2 (2025-01-02) Test in Schule	14
4.4 Cleanup: for more drawings	14
4.5 Changes in Nov 2025	16
4.6 Changes in Dec 2025	16
4.7 Extract OszGraphics.h and .cpp	16
4.7.1 Extract class OszGraphics into a .h	16
4.8 Plot commands from web	20
5 Aktueller Code	20
5.1 esp-oszdisplay.ino	20
5.2 wifitime.h	26
5.3 OszGraphics.h	26
5.4 OszGraphics.cpp	27
5.5 wlan.h.in	28

1 Inbetriebnahme

Die Datei wlan.h.in in wlan.h kopieren

- SSID und Passwort anpassen.
- ntpServer anpassen, wenn über DHCP ein ntpServer konfiguriert ist, wird der vom DHCP verwendet.
- time_zone anpassen.

2 TODO

- HW Koax-Kabel, Telefonkabel geht auch :-)
- HW Gehäuse ... is only covering the details :-))
- HW Netzteil
- HW 5V aus dem Oszilloskop
- split .h in .h and .cpp
- Konfiguration via serielle Konsole
- Konfiguration in Datei am ESP speichern.
- refetch time once a day
- digital countdown zu Stundenende
- HTL Schriftzug beim Start. Logo ?
- DMA zum DAC mit dac_contious.h ... mehr Punkte ?

2.1 Plot different things

get from server/serial/dip-switch

save points to draw by skipping the circle

- christmas tree
- burnings candle/s
- different watch hand: candle, xmastree, ...

3 Übernehmen des alten Projekts und testen

```
// Use an oszilloscope as display

// revision log
// 16-dec-2022

// for brownout detection
#include "soc/soc.h"
#include "soc/rtc_cntl_reg.h"

#include <driver/dac.h>

#include "wifitime.h"

int getLocalTime() {
    struct tm timeinfo;
    int t = -1;
    if (getLocalTime(&timeinfo)) {
        int h = timeinfo.tm_hour;
        if (timeinfo.tm_isdst) {
            h++;
        }
        t = h * 100 + timeinfo.tm_min;
    }
    return t;
}

#include <ArduinoGraphics.h>

class OSZIGraphics : public ArduinoGraphics {
```

```

    uint8_t xy[512][2]; // the display ... the dots to display
public:
    int xy_last = 0;

    OSZIGraphics()
        : ArduinoGraphics(256, 256) {}

    void dump() {
        for (int i = 0; i < xy_last; i++) {
            Serial.print(xy[i][0]);
            Serial.print(" ");
            Serial.println(xy[i][1]);
        }
    }

    void set(int x, int y, uint8_t r, uint8_t g, uint8_t b) {
        xy[xy_last][0] = x;
        xy[xy_last][1] = y;
        xy_last++;
    }

    // tick must be called in loop or an isr
    // returns true if new screen starts.
    bool tick() {
        static int xy_index = 0;
        dac_output_voltage(DAC_CHANNEL_1, xy[xy_index][0]);
        dac_output_voltage(DAC_CHANNEL_2, xy[xy_index][1]);
        xy_index++;
        if (xy_index > xy_last) {
            xy_index = 0;
            return true;
        }
        return false;
    }

    void line(int x0, int y0, int x1, int y1) {
        // do not draw horizontal from higher to lower x
        if ((x0 > x1) && (y0 == y1)) {
            int x = x0;
            x0 = x1;
            x1 = x;
        }
        // do not draw vertical from higher to lower x
        if ((x0 == x1) && (y0 > y1)) {
            int y = y0;
            y0 = y1;
            y1 = y;
        }
        ArduinoGraphics::line(x0, y0, x1, y1);
    }
};

OSZIGraphics oszi = OSZIGraphics();

class AnalogClock {
public:
    // index after clock face
    int xy_index_eof = 0;

    int mid[2] = { 100, 100 };

```

```

int watch_rad = 50;
OSZIGraphics* _oszi;
AnalogClock(OSZIGraphics* disp) {
    _oszi = disp;
}

void drawFace() {
    _oszi->stroke(127, 127, 127); // color
    for (int i = 0; i < 12; i++) {
        float _angle = PI * 2 * i / 12.;
        _oszi->line(mid[0] + watch_rad * cos(_angle),
                    mid[1] + watch_rad * sin(_angle),
                    mid[0] + (watch_rad + 7) * cos(_angle),
                    mid[1] + (watch_rad + 7) * sin(_angle));
        int _sub = 10;
        for (int j = 0; j < _sub; j++) {
            float _angle = PI * 2 * (i + (j / float(_sub))) / float(_sub);
            _oszi->line(mid[0] + (watch_rad + 7) * cos(_angle),
                        mid[1] + (watch_rad + 7) * sin(_angle),
                        mid[0] + (watch_rad + 7) * cos(_angle),
                        mid[1] + (watch_rad + 7) * sin(_angle));
        }
    }
    xy_index_eof = _oszi->xy_last;
}

void drawHand(int len, float angle) {
    int cos_ = len * cos(angle);
    int sin_ = len * sin(angle);
    // BUG line is direction dependend
    _oszi->line(mid[0], mid[1], mid[0] + cos_, mid[1] + sin_);
}

void drawHands(int number) {
    // hhmm
    int mm = number % 100;
    float hh = int(number / 100) + float(mm) / 60;
    _oszi->xy_last = xy_index_eof;

    float mm_angle = PI / 2 + PI * 2 / 60. * (60 - mm);
    drawHand(watch_rad, mm_angle);

    float hh_angle = PI / 2 + PI * 2 / 12. * (12 - hh);
    drawHand(watch_rad - 10, hh_angle);
}

void animation() {
    static int angle = 0;
    _oszi->xy_last = xy_index_eof;
    drawHand(30, angle);
    angle -= 1;
}

void dump() {
    _oszi->dump();
}
};

AnalogClock aClock = AnalogClock(&oszi);

void setup() {

```

```

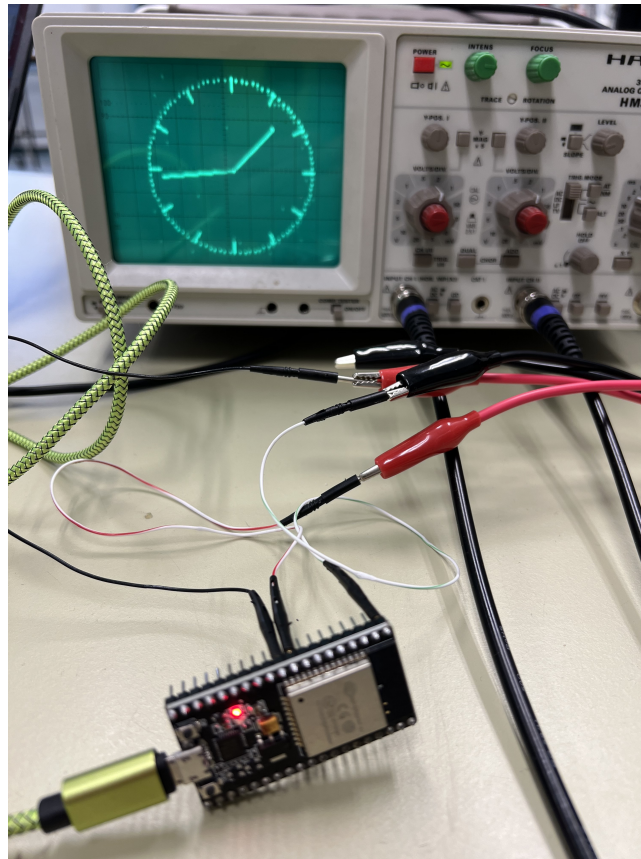
Serial.begin(115200);
// brownout detection off ... at least for wifi connect, when on PC USB
WRITE_PERI_REG(RTC_CNTL_BROWN_OUT_REG, 0); //disable brownout detector

Serial.println("ESP32 Osziclock using DAC1 and DAC2, pin 25 and 26.");
if (!oszi.begin()) {
    Serial.println("Failed to initialize the display!");
    // impossible
}
Serial.print("Draw clock face:");
aClock.drawFace();
Serial.print(aClock.xy_index_eof);
Serial.println("  points");

Serial.println("\nWiFi setup");
wifiSetup();
dac_output_enable(DAC_CHANNEL_1);
dac_output_enable(DAC_CHANNEL_2);
}

void loop() {
    static bool fetch_time = true;
    if (fetch_time) {
        if (getTime_from_network()) {
            fetch_time = false;
        }
    }
    if (oszi.tick()) {
        // one page for a watch is 10ms
        static int cnt = 32000;
        if (cnt > 1000) { // do not fetch time always
            cnt = 0;
            if (fetch_time) {
                aClock.animation();
            } else {
                static int time_ = 0;
                int t = getLocalTime();
                if (t != time_) {
                    time_ = t;
                    Serial.print("localtime ");
                    Serial.println(time_);
                    aClock.drawHands(time_);
                }
            }
        }
        cnt++;
    }
}

```



funktioniert aber es gibt Warnungen

3.1 Change 01

Der Code ist unter Versionkontrolle, deshalb entfernen wir den manuellen Zeitstempel und lassen das die Versionkontrolle machen.

```
+++ esp-oszidisplay.ino          (Arbeitskopie)
@@ -1,7 +1,6 @@
  // Use an oszilloscope as display

-// revision log
-// 16-dec-2022
+// $Date: 2025-12-22 14:45:28 +0100 (Mo, 22. Dez 2025) $

  // for brownout detection
```

3.2 Change 02

Warning: 'DAC_CHANNEL_1' is deprecated: please use 'DAC_CHAN_0' instead

Warning: 'DAC_CHANNEL_2' is deprecated: please use 'DAC_CHAN_1' instead

sind einfach.

Warning: The legacy DAC driver is deprecated, please use *driver/dac_oneshot.h*, *driver/dac_cosine.h* or *driver/dac_continuous.h* instead"

Wechsel zu *dac_oneshot* ... auf <https://docs.espressif.com/projects/esp-idf/en/stable/esp32/api-reference/peripherals/dac.html> steht

Direct Voltage Output (One-shot/Direct Mode)

The DAC channels in the group can convert an 8-bit digital value into the analog when *dac_oneshot_output_voltage()* is called (it can be called in ISR). The analog voltage is kept on the DAC channel until the next conversion starts. To start the voltage conversion, the DAC channels need to be enabled first through registering by *dac_oneshot_new_channel()*.

```

--- esp-oszidisplay.ino (Revision 528)
+++ esp-oszidisplay.ino (Arbeitskopie)
@@ -7,10 +7,12 @@
#include "soc/soc.h"
#include "soc/rtc_cntl_reg.h"

-#include <driver/dac.h>
+#include <driver/dac_oneshot.h>

#include "wifitime.h"

+dac_oneshot_handle_t DAC[2];
+
int getLocalTime() {
    struct tm timeinfo;
    int t = -1;
@@ -53,8 +55,8 @@
// returns true if new screen starts.
bool tick() {
    static int xy_index = 0;
-    dac_output_voltage(DAC_CHAN_0, xy[xy_index][0]);
-    dac_output_voltage(DAC_CHAN_1, xy[xy_index][1]);
+    dac_oneshot_output_voltage(DAC[0], xy[xy_index][0]);
+    dac_oneshot_output_voltage(DAC[1], xy[xy_index][1]);
    xy_index++;
    if (xy_index > xy_last) {
        xy_index = 0;
@@ -165,8 +167,11 @@

    Serial.println("\nWiFi setup");
    wifiSetup();
-    dac_output_enable(DAC_CHAN_0);
-    dac_output_enable(DAC_CHAN_1);
+    dac_oneshot_config_t dacfg;
+    dacfg.chan_id = DAC_CHAN_0;
+    dac_oneshot_new_channel(&dacfg, &DAC[0]);
+    dacfg.chan_id = DAC_CHAN_1;
+    dac_oneshot_new_channel(&dacfg, &DAC[1]);
}

```

void loop() {

Compiliert und läuft.

In der Schule startet der esp immer wieder, ... WLANverbindung ?

3.3 Change 03

Warning: sntp.h in IDF's lwip port folder is deprecated. Please include esp_sntp.h

Warning: void sntp_servermode_dhcp(int) is deprecated: use esp_sntp_servermode_dhcp() instead

```

--- wifitime.h (Revision 530)
@@ -3,3 +3,3 @@
#include "time.h"
-#include "sntp.h" // for ntp timezone and daylight savings
+#include "esp_sntp.h" // for ntp timezone and daylight savings

@@ -21,3 +21,3 @@
    configTzTime(time_zone, ntpServer);
-    sntp_servermode_dhcp(1);
+    esp_sntp_servermode_dhcp(true);
    reconnect_count--;

```

Kompiliert und läuft.

Leider immer noch Winterzeit.

3.4 Change 04: Sommer/Winterzeit aus NTP

Die Uhr geht im Sommer eine Stunde vor.

Die Korrektur ausbauen, dann stimmt die Zeit jetzt.

Warten auf Winterzeit.

```
--- esp-oszidisdisplay.ino          (Revision 537)
@@ -15,3 +15,4 @@

-int getLocalTime() {
+// internal time format 4 digit HHMM
+int getLocalTime_HHMM() {
+    struct tm timeinfo;
@@ -20,5 +21,2 @@
+    int h = timeinfo.tm_hour;
-    if (timeinfo.tm_isdst) {
-        h++;
-    }
+    t = h * 100 + timeinfo.tm_min;
@@ -193,3 +191,3 @@
+    static int time_ = 0;
-    int t = getLocalTime();
+    int t = getLocalTime_HHMM();
+    if (t != time_) {
```

Die Funktion umbenennen ... vielleicht wäre `getTime_in_HHMM` noch klarer.

Tip

Die arduino-IDE hat das rename-refactoring eingebaut, Taste F2. Es gibt auch ein Preview Strg-Enter.

3.5 Change 05: Konfiguration

Wir brauchen einen Platz für die Konfiguration, eine Variable `Config`.

Als erstes bauen wir die Möglichkeit ein Markierungen auf Minuten zu machen.

in C

```
struct {
    bool minute_ticks;
} Config = {
    true
};
```

- Mit `struct` definieren wir einen Datentyp mit verschiedenen benannten Einträgen
- machen gleich eine Variable mit dem Namen `Config` und
- initialisieren mit der Liste `{ true }`.

Wir können jetzt mit `Config.minute_ticks` auf den Wert zugreifen. Davon abhängige Code-Teile fassen wir in

```
if (Config.minute_ticks) {
    // nur wenn true
}
```

ein.

4 Erweiterungen/Umbauten

4.1 Draw something: xmas mode

Die Einstellung `bool Config.xmas` hinzufügen.

```
+++ esp-oszidisplay.ino (Arbeitskopie)
@@ -12,8 +12,9 @@

    struct {
        bool minute_ticks;
+   bool xmas;
    } Config = {
-   false
+   false, true
    };

    // digital to analog channel handles
@@ -102,6 +103,47 @@

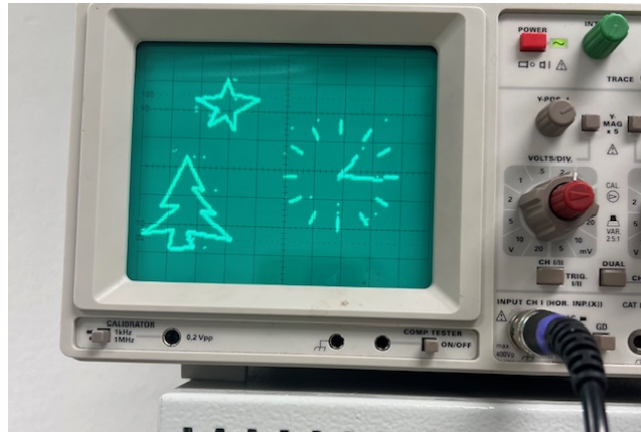
    void drawFace() {
        _oszi->stroke(127, 127, 127); // color
+   if (Config.xmas) {
+       // draw the tree
+       int lines[] = {
+           // x y
+           30, 10, 32, 20,
+           32, 20, 15, 18,
+           16, 18, 34, 34,
+           34, 34, 25, 34,
+           25, 34, 38, 63, // tip
+           38, 63, 48, 44,
+           48, 44, 42, 45,
+           42, 45, 55, 30,
+           55, 30, 47, 31,
+           47, 31, 64, 15,
+           64, 15, 40, 19,
+           40, 19, 43, 10,
+           43, 10, 31, 9,
+           // star
+           50, 82, 54, 92,
+           55, 90, 42, 97, // tip2
+           42, 97, 56, 99,
+           56, 99, 60, 109, // tip 3
+           60, 109, 63, 100,
+           64, 100, 77, 98, // tip 4
+           77, 98, 64, 90,
+           64, 90, 64, 80, // tip 5
+           64, 80, 59, 88,
+           59, 88, 50, 82 // tip 1
+       };
+       int MAX_LINE = sizeof(lines) / sizeof(lines[0]);
+       for (int i = 0; i < MAX_LINE; i += 4) {
+           _oszi->line(lines[i],
+                       lines[i + 1],
+                       lines[i + 2],
+                       lines[i + 3]);
+       }
+       // shift the watch and scale down
+       mid[0] = 125;
+       mid[1] = 55;
```

```

+     watch_rad = 25;
+ }
+   for (int i = 0; i < 12; i++) {
+       float _angle = PI * 2 * i / 12.;
+       _oszi->line(mid[0] + watch_rad * cos(_angle),

```

Wir geben Baum und Stern als Linien mit Anfangs- und Endpunkt ein.



4.2 Snowfall, NOT YET

Schneeflocken (nicht immer ... vielleicht jede Stunde für 15 Minuten ?)

- Ein Ding Schneefall
- mit 40 Flocken
- die nach unten torkeln (random bevorzugt nach unten aber vielleicht auch schräg)

Cancelled: ... upload works only sometimes and loop is aggressively optimized WHY?

```

+++ esp-oszidisplay.ino (Arbeitskopie)
@@ -13,8 +13,9 @@
+   struct {
+       bool minute_ticks;
+       bool xmas;
+   } Config = {
+       false, true
+   };

// digital to analog channel handles
@@ -89,7 +90,10 @@

OSZIGraphics oszi = OSZIGraphics();

+const int MAX_FLAKE = 40;
+
+class AnalogClock {
+   int flakes[2][MAX_FLAKE];
+public:
+   // index after clock face
+   int xy_index_eof = 0;
@@ -99,8 +103,38 @@
+   OSZIGraphics* _oszi;
+   AnalogClock(OSZIGraphics* disp) {
+       _oszi = disp;
+       randomSeed(micros());
+   }

```

```

+ void snowFall() {
+   int top = 140;
+   int width = 170;
+   // x, y
+   for (int i = 0; i < MAX_FLAKE; i++) {
+     if ((flakes[i][0] <= 0) || (flakes[i][0] >= width)) {
+       flakes[i][0] = int(random(width));
+     }
+     if ((flakes[i][1] <= 0) || (flakes[i][1] > (top + 20))) {
+       flakes[i][1] = int(random(top, top + 20));
+     }
+     _oszi->point(flakes[i][0], flakes[i][1]);
+     // TODO slower
+     switch (random(5)) {
+       case 5: break;
+       case 4: flakes[i][0] = -2;
+         // fall through
+       case 3: flakes[i][0] = +1;
+         // fall through
+       default:
+         // TODO slowdown at bottom
+         flakes[i][1] = -1;
+         if (flakes[i][1] < 0) {
+           flakes[i][1] = top;
+         }
+     }
+   }
+ }
+
+ void drawFace() {
+   _oszi->stroke(127, 127, 127); // color
+   if (Config.xmas) {
@@ -240,9 +280,11 @@ # in loop()
+     int t = getLocalTime_HHMM();
+     if (t != time_) {
+       time_ = t;
+       Serial.print("localtime ");
+       Serial.println(time_);
+       aClock.drawHands(time_);
+     } else if (Config.snow_falling) {
+       aClock.snowFall();
+     }
+   }
+   cnt++;
+ }

```

4.3 Cleanup: remove coordinate duplicates from draw

Die Linien im lines-Array haben Start- und Endpunkt, es würde ausreichen

- den nächsten Punkt anzugeben
- und eine Möglichkeit keine Linie zu zeichnen.

```

for (int i = 2; i < MAX_LINE; i += 2) {
  // no line into negative space
  if (lines[i] >= 0) {
    // line from previous to current point
    _oszi->line(lines[i - 2],
               lines[i - 1],

```

```

        lines[i],
        lines[i + 1]);
    }
}

```

Ergibt (meistens)

:Guru Meditation Error: Core 1 panic'ed (LoadProhibited). Exception was unhandled.

LoadProhibited means the CPU was trying to load from an illegal address in memory.

- Letzte Log-Meldung Zeitausgabe:
Versuch wifitime.h umzubauen, um weniger Speicher in Serial.print(time) zu benötigen.
Keine Änderung.
- Die lines ausserhalb der Funktion deklarieren hilft auch nicht.
- TODO i += 2 killt (meistens)

4.3.1 (2025-01-01) Versuch zu Hause, kompiliert und läuft (keine optische Kontrolle)

Die Liste von int auf uint8_t umstellen, das sind nur positive Zahlen kleiner 255.:

```

static uint8_t lines[] = {
static weil ... halt.

```

Sie enthält nur noch *Punkte*, XY-Koordinaten. Aus

```

-      30, 10, 32, 20,
-      32, 20, 15, 18,

```

wird

```

+ 30, 10,
+ 32, 20,
+ 15, 18,

```

Es wird eine Linie zum vorigen Punkt gezogen. Zum *vorigen* deshalb fangen wir mit 2 an:

```

-      for (int i = 2; i < MAX_LINE; i += 4) {
+      for (int i = 2; i < MAX_LINE; i += 2) {

```

Die Linie wird nur gezogen, wenn beide X-Koordinaten kleiner 255 sind, damit können wir Unterbrechungen machen:

```

        for (int i = 2; i < MAX_LINE; i += 4) {
-      if (lines[i] < 255) {
        for (int i = 2; i < MAX_LINE; i += 2) {
+      if ((lines[i - 2] < 255) && (lines[i] < 255)) {
            _oszi->line(lines[i - 2],
                        lines[i - 1],
                        lines[i],

```

Zum Beispiel hier: vorher – nachher +:

```

+ 30, 10,
+ 32, 20,
+ 15, 18, 255, 255,
+ 16, 18,
+ 34, 34,
-      30, 10, 32, 20,
-      32, 20, 15, 18,
-      16, 18, 34, 34,

```

```

OSZIGraphics oszi = OSZIGraphics();

```

```

static uint8_t lines[] = {
    // x y
+ 30, 10,

```

```

+ 32, 20,
+ 15, 18, 255, 255,
+ 16, 18,
+ 34, 34,
+ 25, 34,
+ 38, 63, // tip
+ 48, 44,
+ 42, 45,
+ 55, 30,
+ 47, 31,
+ 64, 15,
+ 40, 19,
+ 43, 10, 31, 9,
+ // no line
+ 255, 255,
+ // star
+ 50, 82,
+ 54, 92, 255, 255,
+ 55, 90,
+ 42, 97, // tip2
+ 56, 99,
+ 60, 109, // tip 3
+ 63, 100, 255, 255,
+ 64, 100,
+ 77, 98, // tip 4
+ 64, 90,
+ 64, 80, // tip 5
+ 59, 88, 50, 82 // tip 1
- 30, 10, 32, 20,
- 32, 20, 15, 18,
- 16, 18, 34, 34,
- 34, 34, 25, 34,
- 25, 34, 38, 63, // tip
- 38, 63, 48, 44,
- 48, 44, 42, 45,
- 42, 45, 55, 30,
- 55, 30, 47, 31,
- 47, 31, 64, 15,
- 64, 15, 40, 19,
- 40, 19, 43, 10,
- 43, 10, 31, 9,
- // star
- 50, 82, 54, 92,
- 55, 90, 42, 97, // tip2
- 42, 97, 56, 99,
- 56, 99, 60, 109, // tip 3
- 60, 109, 63, 100,
- 64, 100, 77, 98, // tip 4
- 77, 98, 64, 90,
- 64, 90, 64, 80, // tip 5
- 64, 80, 59, 88,
- 59, 88, 50, 82 // tip 1
};

class AnalogClock {
@@ -164,8 +141,8 @@
    // draw the tree

    const int MAX_LINE = sizeof(lines) / sizeof(lines[0]);
    for (int i = 2; i < MAX_LINE; i += 4) {
        if (lines[i] < 255) {

```

```
+     for (int i = 2; i < MAX_LINE; i += 2) {
+         if ((lines[i - 2] < 255) && (lines[i] < 255)) {
+             _oszi->line(lines[i - 2],
+                         lines[i - 1],
+                         lines[i],
```

4.3.2 (2025-01-02) Test in Schule

funktioniert. War das nur ein Problem mit den Koordinaten 255 ? ... egal.

4.4 Cleanup: for more drawings

- Rename `lines` to `xmas_lines`.
 - Put the cursor on `lines`
 - Press F2 ... the refactoring popup is displayed
 - change `lines` to `xmas_lines`
 - Press Shift-Enter to preview

Note: Only the first occurrence is replaced.
 - Click the word `lines`, press F2, Press Enter

the compiler will find the left overs ... this is not needed, only the preview misses out the repetition in a line.
- Make the `xmas_lines` `const`, not `static`.

Did not change code size at all. ... but so.

Still working.

- Move things outside `if xmas` to enable different drawings.
 - `MAX_LINE` is no longer `const`.
 - `lines` is a pointer to the actual list of points.
 - plotting happens after the `if xmas-check`.

```
@@ -137,24 +137,26 @@
```

```
void drawFace() {
    _oszi->stroke(127, 127, 127); // color
+   int MAX_LINE = 0;
+   uint8_t* lines;
    if (Config.xmas) {
        // draw the tree
        -
        -
        -   const int MAX_LINE = sizeof(xmas_lines) / sizeof(xmas_lines[0]);
        -   for (int i = 2; i < MAX_LINE; i += 2) {
        -       if ((xmas_lines[i - 2] < 255) && (xmas_lines[i] < 255)) {
        -           _oszi->line(xmas_lines[i - 2],
        -                   xmas_lines[i - 1],
        -                   xmas_lines[i],
        -                   xmas_lines[i + 1]);
        -       }
        -   }
        -
+   MAX_LINE = sizeof(xmas_lines) / sizeof(xmas_lines[0]);
+   lines = (uint8_t*)&xmas_lines[0];
    // shift the watch and scale down
    mid[0] = 125;
    mid[1] = 55;
    watch_rad = 25;
}
```

```

+
+ // draw optional lines
+ for (int i = 2; i < MAX_LINE; i += 2) {
+     if ((lines[i - 2] < 255) && (lines[i] < 255)) {
+         _oszi->line(lines[i - 2],
+                     lines[i - 1],
+                     lines[i],
+                     lines[i + 1]);
+     }
+ }
+ // clock face

```

Still no breakage ... maybe stop.

- HTL as default lines, xmas only in december (TODO).

```

@@ -100,12 +100,31 @@

```

```

    OSZIGraphics oszi = OSZIGraphics();

+const uint8_t htl_lines[] = {
+ // H
+ 30, 20,
+ 30, 10, 255, 255,
+ 30, 15,
+ 39, 15, 255, 255,
+ 39, 20,
+ 39, 10, 255, 255,
+ // T
+ 42, 20,
+ 51, 20, 255, 255,
+ 47, 20,
+ 47, 10, 255, 255,
+ // L
+ 54, 20,
+ 54, 10,
+ 62, 10
+};
+
const uint8_t xmas_lines[] = {
    // x y
    30, 10,

@@ -153,11 +176,19 @@
    if (Config.xmas) {
        MAX_LINE = sizeof(xmas_lines) / sizeof(xmas_lines[0]);
        lines = (uint8_t*)&xmas_lines[0];
        // shift the watch and scale down
        mid[0] = 125;
        mid[1] = 55;
        watch_rad = 25;
+    } else {
+        MAX_LINE = sizeof(htl_lines) / sizeof(htl_lines[0]);
+        lines = (uint8_t*)&htl_lines[0];
+        // shift the watch and scale down
+        mid[0] = 125;
+        mid[1] = 55;
+        watch_rad = 25;
+    }

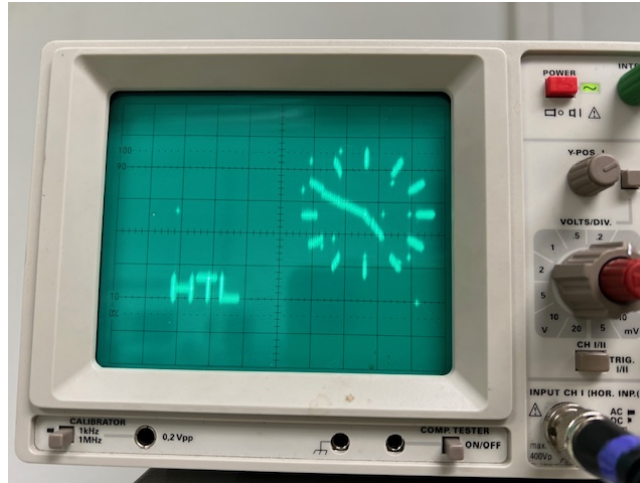
    // draw optional lines
    for (int i = 2; i < MAX_LINE; i += 2) {

```

```

    if ((lines[i - 2] < 255) && (lines[i] < 255)) {
+      // TODO make it relative, only first point is absolute
      _oszi->line(lines[i - 2],

```



4.5 Changes in Nov 2025

- get time from network hangs till time is received
- the line plots are relative to starting point

4.6 Changes in Dec 2025

- moving things ... a snowman
- extract oszigraphics.h and .cpp

4.7 Extract OsziGraphics.h and .cpp

Allen Code in eine Datei zu packen ist unüblich, weil

1. die Datei wird sehr groß, das bedeutet die zum Kompilieren benötigte Zeit steigt an.
2. vs-code hat kürzere Buildzeiten weil es diesen Vorteil ausnutzt. Die arduino-IDE packt vorher alles zusammen und schickt das ganz dann durch den Compiler.
3. Man findet Dinge leichter wenn die Dateien kleiner sind.

4.7.1 Extract class OsziGraphics into a .h

and include the file. Das tut eigentlich gar nichts weil der C-Präprozessor den Text wieder da hin tut wo das `include` steht, aber wir haben eine kleiner ino-Datei und der Code zu OsziGraphics ändert sich selten.

1. In esp-oszidisplay.ino Zeile 41 und folgende

```
#include <ArduinoGraphics.h>
```

```

class OSZIGraphics : public ArduinoGraphics {
    uint8_t xy[1024][2]; // the display ... the dots to display
public:
    int xy_last = 0;

    OSZIGraphics()
        : ArduinoGraphics(256, 256) {}

    void dump() {
        for (int i = 0; i < xy_last; i++) {
            Serial.print(xy[i][0]);

```



```

        Serial.print(" ");
        Serial.println(xy[i][1]);
    }
}

void set(int x, int y, uint8_t r, uint8_t g, uint8_t b) {
    xy[xy_last][0] = x;
    xy[xy_last][1] = y;
    xy_last++;
}

// tick must be called in loop or an isr
// returns true if new screen starts.
bool tick() {
    static int xy_index = 0;
    dac_oneshot_output_voltage(DAC[0], xy[xy_index][0]);
    dac_oneshot_output_voltage(DAC[1], xy[xy_index][1]);
    xy_index++;
    if (xy_index > xy_last) {
        xy_index = 0;
        return true;
    }
    return false;
}

void line(int x0, int y0, int x1, int y1) {
    // do not draw horizontal from higher to lower x
    if ((x0 > x1) && (y0 == y1)) {
        int x = x0;
        x0 = x1;
        x1 = x;
    }
    // do not draw vertical from higher to lower x
    if ((x0 == x1) && (y0 > y1)) {
        int y = y0;
        y0 = y1;
        y1 = y;
    }
    ArduinoGraphics::line(x0, y0, x1, y1);
}
};

```

in eine Datei OsziGraphics.h verschieben und durch:

```
#include "OsziGraphics.h"
```

ersetzen.

2. Guard statements in die header-Datei.

Um zu verhindern, dass eine Datei mehrfach inkludiert wird und der Compiler Symbole/Code mehrfach bekommt, setzt man Guard(Wächter)-Zeilen.

Ganz am Anfang der h-Datei

```
#ifndef _OSZIGRAPHICS_H_
#define _OSZIGRAPHICS_H_
```

und am Ende dann

```
#endif
```

Das heisst wenn `_OSZIGRAPHICS_H_` noch nicht definiert ist `ifndef`, IF Not DEFINed, werden die Zeilen bis zum `#endif` gelesen ... sonst nicht.

KOMISCH: Warum ist die Uhr jetzt halb so breit ? ... später.

3. Extract code to .cpp

In der .h sollte kein Code sein nur die Datenstruktur und Methoden.

Eine Datei OsziGraphics.cpp anlegen.

```
#include "OsziGraphics.cpp"
```

ERROR: error: 'DAC' was not declared in this scope:

```
    dac_one_shot_output_voltage(DAC[0], xy[xy_index][0]);

in OsziGraphics.cpp
```

Wir verschieben die Deklaration der Variable aus der ino-Datei in die ... cpp-Dateih ginge auch, aber besser ist wenn die Variable nur lokal im Modul OsziGraphics ist.

```
// digital to analog channel handles
dac_one_shot_handle_t DAC[2];
```

ERROR: error: 'DAC' was not declared in this scope:

```
    dac_one_shot_output_voltage(DAC[0], xy[xy_index][0]);

in esp-ozidisplay.ino
```

logisch wir haben die Deklaration verschoben ... versteckt.

ABER: weil der ganze Code noch in OsziGraphics.h steht und dieser in die .ino inkludiert wird, braucht es die Variable dort auch noch.

Verschieben des Codes aus der Header-Datei.

```
bool tick() {
    static int xy_index = 0;
    dac_one_shot_output_voltage(DAC[0], xy[xy_index][0]);
    dac_one_shot_output_voltage(DAC[1], xy[xy_index][1]);
    xy_index++;
    if (xy_index > xy_last) {
        xy_index = 0;
        return true;
    }
    return false;
}
```

in die cpp

```
bool OsziGraphics::tick() {
    static int xy_index = 0;
    dac_one_shot_output_voltage(DAC[0], xy[xy_index][0]);
    dac_one_shot_output_voltage(DAC[1], xy[xy_index][1]);
    xy_index++;
    if (xy_index > xy_last) {
        xy_index = 0;
        return true;
    }
    return false;
}
```

ERROR: DAC was not declared in this scope.

in esp-ozidisplay.ino.

am Ende der setup-Funktion

```
dac_one_shot_config_t dacfg;
dacfg.chan_id = DAC_CHAN_0;
dac_one_shot_new_channel(&dacfg, &DAC[0]);
dacfg.chan_id = DAC_CHAN_1;
dac_one_shot_new_channel(&dacfg, &DAC[1]);
```

wir verschieben das in eine neue Funktion `OsziGraphics::Init()` in der *cpp*-Datei ... und rufen diese im Konstruktor auf und deklarieren sie.

```
OSZIGraphics()
: ArduinoGraphics(256, 256) {
    Init();
}
```

```
void Init();
```

ERROR: der nächste in der *cpp*-Datei:

```
'dac_one_shot_handle_t' does not name a type
```

klar der `DigitalAnalogConverter` ist in *driver/dac_one_shot.h* deklariert ... die muss in der *cpp*-Datei inkludiert werden.

ERROR: in der *cpp*-Datei:

```
'OsziGraphics' has not been declared
```

klar die Klasse heisst `OSZIGraphics` .. ich ändere den Klassennamen in *OsziGraphics.h* zu `OsziGraphics`, dann passt er zum Dateinamen.

ERROR: in der *cpp*-Datei

```
no declaration matches 'void OsziGraphics::init()'
```

Ein Tippfehler, in der Header Datei steht `Init`.

ERROR: in der *ino*-Datei

```
error: 'OSZIGraphics' does not name a type; did you mean 'OsziGraphics'?
43 | OSZIGraphics oszi = OSZIGraphics();
```

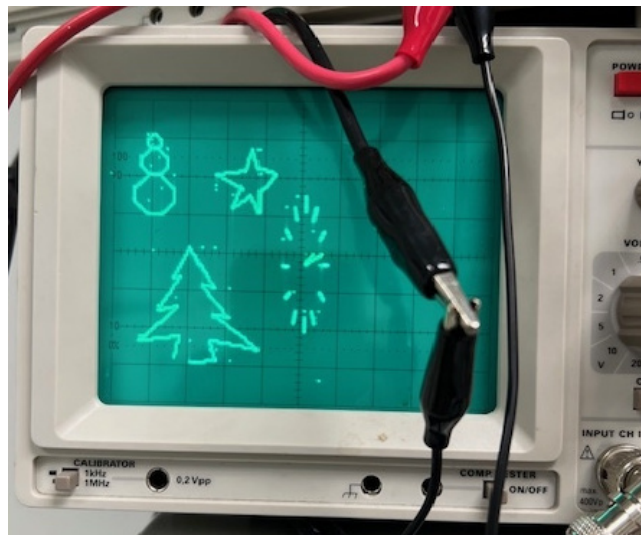
stimmt es heisst jetzt `OsziGraphics`.

ERROR: in der *ino*-Datei:

```
'OSZIGraphics' does not name a type; did you mean 'OsziGraphics'?
```

der Compiler schlägt schon die Korrektur vor.

DONE: kompiliert und upload ... die Uhr ist immer noch nur halb so breit.



4. move the rest of code from *.h* into *.cpp*

`OsziGraphics::dump`, `set` and `line`.

in der header-Datei stand

```
void set(int x, int y, uint8_t r, uint8_t g, uint8_t b) {
    xy[xy_last][0] = x;
    xy[xy_last][1] = y;
    xy_last++;
}
```

danach nur mehr

```
void set(int x, int y, uint8_t r, uint8_t g, uint8_t b);
```

in der cpp-Datei

```
void OsziGraphics::set(int x, int y, uint8_t r, uint8_t g, uint8_t b) {
    xy[xy_last][0] = x;
    xy[xy_last][1] = y;
    xy_last++;
}
```

4.8 Plot commands from web

To be able to change the drawing without recompilation.

TODO:

- File from web into esp-file
- Text tokenizer
- Translate to lines array.

5 Aktueller Code

5.1 esp-oszidisplay.ino

```
// Use an oszilloscope as display

// $Date: 2025-12-22 14:18:49 +0100 (Mo, 22. Dez 2025) $

// for brownout detection
#include "soc/soc.h"
#include "soc/rtc_cntl_reg.h"

#include "wifitime.h"

struct {
    bool minute_ticks;
    int8_t theme; // 0=ANICH CCA, 1=tree+star, 2=snowcrystal
    String plot_uri;
} Config = {
    false, 1, "http://192.168.227.122/walls/oszi.plt"
};

// internal time format 4 digit HHMM
int getLocalTime_HHMM() {
    struct tm timeinfo;
    int t = -1;
    if (getLocalTime(&timeinfo)) {
        int h = timeinfo.tm_hour;
        t = h * 100 + timeinfo.tm_min;
    }
    return t;
}

// return month number: 1..12
```

```

int getMonth() {
    struct tm timeinfo;
    int m = -1;
    if (getLocalTime(&timeinfo)) {
        // month starts with 0
        m = timeinfo.tm_mon + 1;
    }
    return m;
}

#include "OszGraphics.h"

OszGraphics oszi = OszGraphics();

const int cwid = 10;
const int cht = 10;

const int8_t JMP = 127;

const int8_t htl_lines[] = {
    // A
    30, 70,
    5, 10,
    5, -10, JMP, JMP,
    -7, 4,
    4, 0, JMP, JMP,
    // N
    -7, -18, 0, 10,
    10, -10, 0, 10,
    JMP, JMP,
    // I
    -5, -24, 0, 10,
    JMP, JMP,
    // C
    -7, -22, -2, -2,
    -5, 0, -3, 3,
    0, 4, 3, 3,
    5, 0, 2, -2, JMP, JMP,
    // C
    12, -6, -2, -2,
    -5, 0, -3, 3,
    0, 4, 3, 3,
    5, 0, 2, -2, JMP, JMP,
    // A
    2, -8,
    5, 10,
    5, -10, JMP, JMP,
    -7, 5,
    4, 0, JMP, JMP,
    // H 37+12 33
    -10, -9,
    0, -10, JMP, JMP,
    0, 5,
    -9, 0, JMP, JMP,
    0, 5,
    0, -10, JMP, JMP,
    // T
    13, 10,
    9, 0, JMP, JMP,
    -4, 0,
    0, -10, JMP, JMP,

```

```

// L
7, 10,
0, -10,
8, 0
};

const int8_t xmas_lines[] = {
// x y
30, 10,
2, 10,
-17, -2,
-1, 0,
17, 16,
-9, -2,
10, 16,
-3, -1,
7, 15, // tip
12, -19,
-6, 1,
13, -15,
-8, 1,
23, -16,
-24, 4,
3, -9, -12, -1,
// no line
JMP, JMP,
// star
19, 73,
4, 10, JMP, JMP,
1, -2,
-13, 7, // tip2
14, 2,
4, 10, // tip 3
3, -9, JMP, JMP,
1, 0,
13, -2, // tip 4
-13, -8,
0, -10, // tip 5
-5, 8, -9, -6 // tip 1
};

const int8_t move_lines[] = {
// snow flake?
// x y
40, 120,
// 6 star
6, 0, JMP, JMP,
-5, -3, 5, 5, JMP, JMP,
-3, 0, 5, -5
};

const int8_t snowman_lines[] = {
// snow flake?
// x y
40, 120,
// head
2, 0, 2, 2, 0, 2, -2, 2, -2, 0, -2, -2, 0, -2, 2, -2,
3, 0, 4, -4, 0, -4, -4, -4, -4, 0, -4, 4, 0, 4, 4, 4, JMP, JMP,
0, -12,
5, 0, 6, -6, 0, -6, -6, -6, -6, 0, -6, 6, 0, 6, 6, 6
};

```

```

class AnalogClock {
    bool hand_drawn = false;
public:
    // index after clock face
    int xy_index_eof = 0;

    int mid[2] = { 100, 100 };
    int watch_rad = 50;
    OszGraphics* _oszi;

    int movement_MAX_LINE = 0;
    int8_t* movement_lines;
    char movement = 'r'; // falling snow

    AnalogClock(OszGraphics* disp) {
        _oszi = disp;
        randomSeed(micros());
        movement_MAX_LINE = sizeof(snowman_lines) / sizeof(snowman_lines[0]);
        movement_lines = (int8_t*)&snowman_lines[0];
    }

    void drawFace() {
        _oszi->stroke(127, 127, 127); // color
        int MAX_LINE = 0;
        int8_t* lines;
        switch (Config.theme) {
            case 1:
                MAX_LINE = sizeof(xmas_lines) / sizeof(xmas_lines[0]);
                lines = (int8_t*)&xmas_lines[0];
                // shift the watch and scale down
                mid[0] = 125;
                mid[1] = 55;
                watch_rad = 25;
                break;
            default:
                MAX_LINE = sizeof(htl_lines) / sizeof(htl_lines[0]);
                lines = (int8_t*)&htl_lines[0];
                // shift the watch and scale down
                mid[0] = 110;
                mid[1] = 50;
                watch_rad = 30;
                break;
        }

        // clock face
        for (int i = 0; i < 12; i++) {
            float _angle = PI * 2 * i / 12.;
            _oszi->line(mid[0] + watch_rad * cos(_angle),
                      mid[1] + watch_rad * sin(_angle),
                      mid[0] + (watch_rad + 7) * cos(_angle),
                      mid[1] + (watch_rad + 7) * sin(_angle));
            if (Config.minute_ticks) {
                int _sub = 10;
                for (int j = 0; j < _sub; j++) {
                    float _angle = PI * 2 * (i + (j / float(_sub))) / float(_sub);
                    _oszi->line(mid[0] + (watch_rad + 7) * cos(_angle),
                              mid[1] + (watch_rad + 7) * sin(_angle),
                              mid[0] + (watch_rad + 7) * cos(_angle),
                              mid[1] + (watch_rad + 7) * sin(_angle));
                }
            }
        }
    }
}

```

```

    }
}

// draw optional lines
int8_t x = lines[0];
int8_t y = lines[1];
for (int i = 2; i < MAX_LINE; i += 2) {
    if (lines[i] == JMP) {
    } else if (lines[i - 2] == JMP) {
        x = x + lines[i];
        y = y + lines[i + 1];
    } else {
        _oszi->line(x, y, x + lines[i], y + lines[i + 1]);
        x = x + lines[i];
        y = y + lines[i + 1];
    }
}

xy_index_eof = _oszi->xy_last;
}

int dx = 0;
int dy = 0;
void drawMovement() {
    if (!hand_drawn) {
        return;
    }
    // draw move lines
    _oszi->xy_last = xy_after_hands;
    switch (movement) {
        case 'f':
            // random downwards
            dx += random(0, 3) - 1;
            dy += random(0, 2) - 1;
            break;
        default:
            // random
            dx += random(0, 3) - 1;
            dy += random(0, 3) - 1;
            break;
    }
    int8_t x = movement_lines[0] + dx;
    if (x < 0) { dx = 0; }
    int8_t y = movement_lines[1] + dy;
    if (y < 0) { dy = 0; }
    for (int i = 2; i < movement_MAX_LINE; i += 2) {
        if (movement_lines[i] == JMP) {
        } else if (movement_lines[i - 2] == JMP) {
            x = x + movement_lines[i];
            y = y + movement_lines[i + 1];
        } else {
            _oszi->line(x, y, x + movement_lines[i], y + movement_lines[i + 1]);
            x = x + movement_lines[i];
            y = y + movement_lines[i + 1];
        }
    }
}

void drawHand(int len, float angle) {
    int cos_ = len * cos(angle);
    int sin_ = len * sin(angle);

```



```

    // BUG line is direction dependend
    _oszi->line(mid[0], mid[1], mid[0] + cos_, mid[1] + sin_);
}

int xy_after_hands = 0;
void drawHands(int number) {
    // hmmm
    int mm = number % 100;
    float hh = int(number / 100) + float(mm) / 60;
    _oszi->xy_last = xy_index_eof;

    float mm_angle = PI / 2 + PI * 2 / 60. * (60 - mm);
    drawHand(watch_rad, mm_angle);

    float hh_angle = PI / 2 + PI * 2 / 12. * (12 - hh);
    drawHand(watch_rad - 10, hh_angle);
    xy_after_hands = _oszi->xy_last;
    hand_drawn = true;
}

void animation() {
    static int angle = 0;
    _oszi->xy_last = xy_index_eof;
    drawHand(30, angle);
    angle -= 1;
}

void dump() {
    _oszi->dump();
}
};

AnalogClock aClock = AnalogClock(&oszi);

void setup() {
    Serial.begin(115200);
    // brownout detection off ... at least for wifi connect, when on PC USB
    WRITE_PERI_REG(RTC_CNTL_BROWN_OUT_REG, 0); //disable brownout detector

    Serial.println("ESP32 Osziclock using DAC1 and DAC2, pin 25 and 26.");
    if (!oszi.begin()) {
        Serial.println("Failed to initialize the display!");
        // impossible
    }
    // TODO face might be time/month depending. But time is after WiFi.
    Serial.print("Draw clock face:");
    aClock.drawFace();
    Serial.print(aClock.xy_index_eof);
    Serial.println("  points");

    Serial.println("\nWiFi setup");
    wifiSetup();
}

void loop() {
    static bool fetch_time = true;
    // TODO refetch_time
    if (fetch_time) {
        if (getTime_from_network()) {
            fetch_time = false;
        }
    }
}

```

```

}
if (oszi.tick()) {
    // one page for a watch is 10ms
    static int cnt = 32000;
    if (cnt > 1000) { // do not fetch time always
        cnt = 0;
        if (fetch_time) {
            aClock.animation();
        } else {
            static int time_ = 0;
            int t = getLocalTime_HHMM();
            if (t != time_) {
                time_ = t;
                Serial.print("localtime ");
                Serial.println(time_);
                aClock.drawHands(time_);
                //get_plot_file(Config.plot_uri);
            }
        }
    }
    if (0 == (cnt % 50)) {
        aClock.drawMovement();
    }
    cnt++;
}
}

```

5.2 wifitime.h

```

#include <HardwareSerial.h>
#include <WString.h>
// time from wlan
#ifndef _WIFITIME_H_
#define _WIFITIME_H_

bool getTime_from_network();
bool http_get(String uri);
bool get_plot_file(String uri);

void wifiSetup();

#endif

```

5.3 OsziGraphics.h

```

#ifndef _OSZIGRAPHICS_H_
#define _OSZIGRAPHICS_H_

#include <ArduinoGraphics.h>

class OsziGraphics : public ArduinoGraphics {
    uint8_t xy[1024][2]; // the display ... the dots to display
public:
    int xy_last = 0;

    OsziGraphics()
        : ArduinoGraphics(256, 256) {
        Init();
    }

    void Init();

```

```

void dump();

void set(int x, int y, uint8_t r, uint8_t g, uint8_t b);

// tick must be called in loop or an isr
// returns true if new screen starts.
bool tick();

void line(int x0, int y0, int x1, int y1);
};

#endif

```

5.4 OszGraphics.cpp

```

#include "OszGraphics.h"

#include <driver/dac_oneshot.h>

// digital to analog channel handles
dac_oneshot_handle_t DAC[2];

void OszGraphics::Init() {
    dac_oneshot_config_t dacfg;
    dacfg.chan_id = DAC_CHAN_0;
    dac_oneshot_new_channel(&dacfg, &DAC[0]);
    dacfg.chan_id = DAC_CHAN_1;
    dac_oneshot_new_channel(&dacfg, &DAC[1]);
}

void OszGraphics::dump() {
    for (int i = 0; i < xy_last; i++) {
        Serial.print(xy[i][0]);
        Serial.print(" ");
        Serial.println(xy[i][1]);
    }
}

bool OszGraphics::tick() {
    static int xy_index = 0;
    dac_oneshot_output_voltage(DAC[0], xy[xy_index][0]);
    dac_oneshot_output_voltage(DAC[1], xy[xy_index][1]);
    xy_index++;
    if (xy_index > xy_last) {
        xy_index = 0;
        return true;
    }
    return false;
}

void OszGraphics::set(int x, int y, uint8_t r, uint8_t g, uint8_t b) {
    xy[xy_last][0] = x;
    xy[xy_last][1] = y;
    xy_last++;
}

void OszGraphics::line(int x0, int y0, int x1, int y1) {
    // do not draw horizontal from higher to lower x
    if ((x0 > x1) && (y0 == y1)) {
        int x = x0;

```

```
    x0 = x1;
    x1 = x;
}
// do not draw vertical from higher to lower x
if ((x0 == x1) && (y0 > y1)) {
    int y = y0;
    y0 = y1;
    y1 = y;
}
ArduinoGraphics::line(x0, y0, x1, y1);
}
```

5.5 wlan.h.in

```
#ifndef _WLAN_H_
#define _WLAN_H_

// kopieren in Datei wlan.h
// W-Lan-Zugang einrichten
const char* ssid = "HTL-IoT";
const char* password = "password";
// ntpServer from DHCP will overwrite this one
const char* ntpServer = "10.10.204.254";
// htl: "10.10.204.254";
// at.pool.ntp.org: 151.236.30.71
// TimeZone rule for Europe/Vienna including daylight adjustment rules
const char* time_zone = "CET-1CEST-2,M3.5.0/02:00:00,M10.5.0/03:00:00";

#endif
```