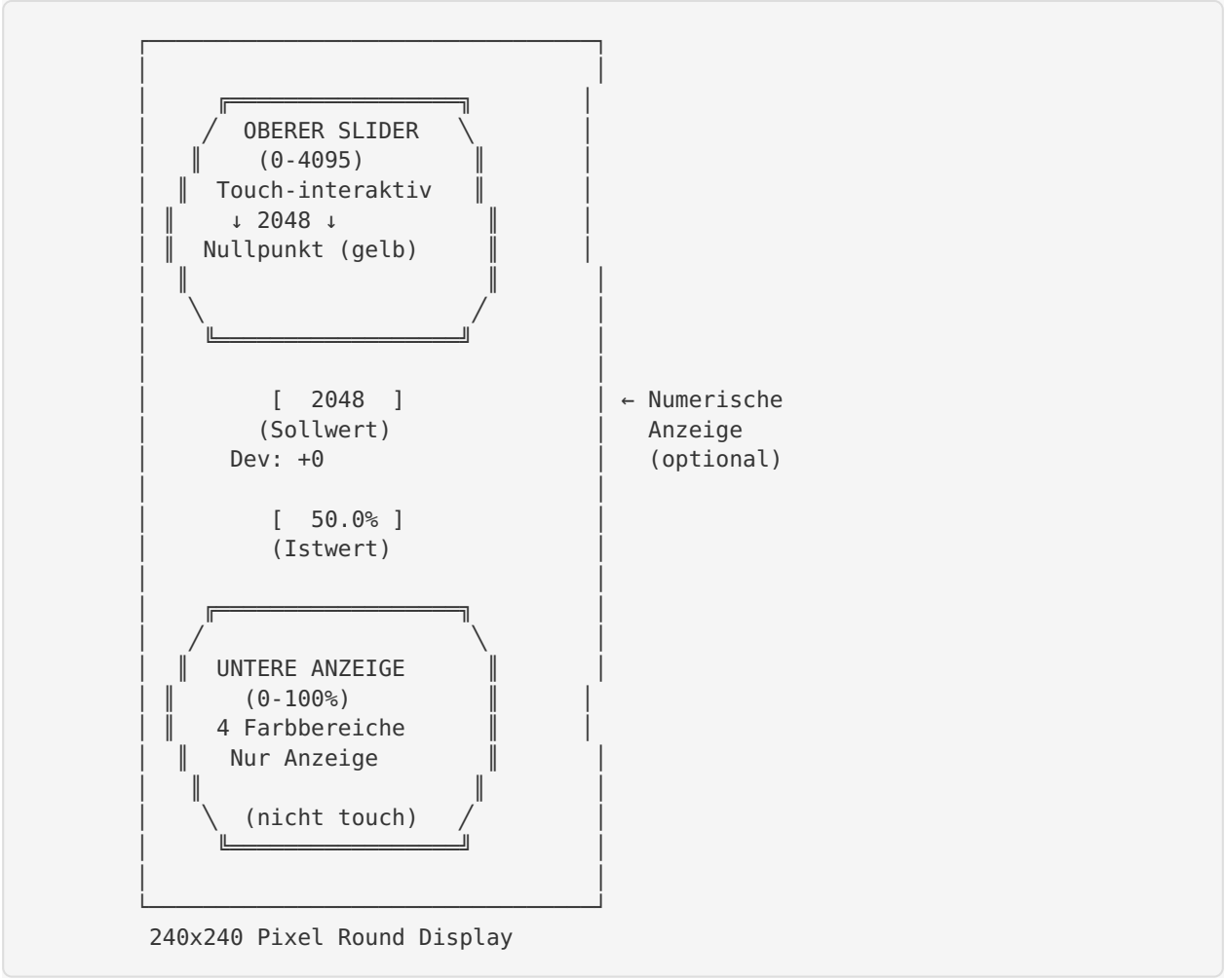
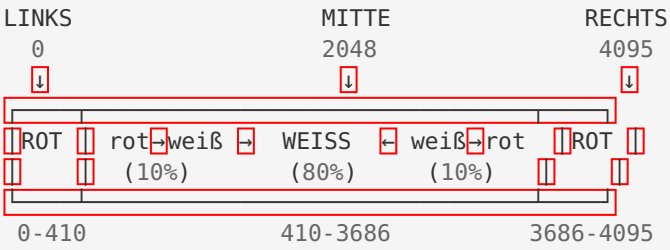
DualArcDisplay - Visueller Leitfaden

📊 Display-Layout Übersicht



🎨 Farbschema im Detail

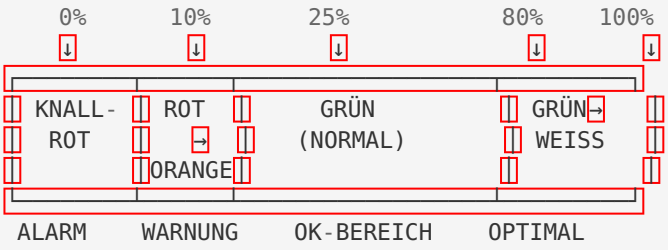
Oberer Slider - Farbverlauf



FARBEN:
- 0-410: Rot (0xF800) → Weiß (0xFFFF) [fließend]
- 410-3686: Weiß (0xFFFF) [konstant]
- 3686-4095: Weiß (0xFFFF) → Rot (0xF800) [fließend]

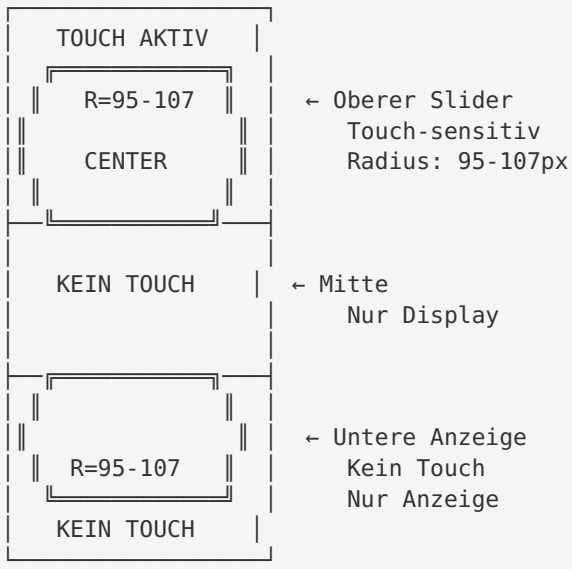
NULLPUNKT:
- **Position**: 2048 (exakte Mitte)
- Markierung: Gelbe Linie (0xFFE0)
- Bedeutung: Referenzpunkt / Neutral

Untere Anzeige - Farbzonen

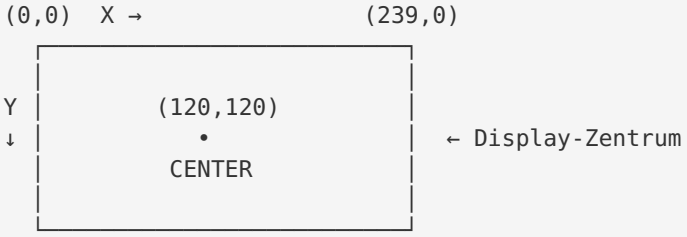


FARBEN:
- 0-10%: Knallrot (0xF800) [konstant]
  → ALARM / Kritisch
- 10-25%: Rot (0xF800) → Orange (0xFC00) [fließend]
  → WARNUNG / Niedrig
- 25-80%: Grün (0x07E0) [konstant]
  → OK / Normal-Betrieb
- 80-100%: Grün (0x07E0) → Weiß (0xFFFF) [fließend]
  → OPTIMAL / Voll

🖱️ Touch-Bereiche



Touch-Koordinaten-System



Touch-Berechnung:
• Distanz = √[(x-120)² + (y-120)²]
• Winkel = atan2(y-120, x-120)
• Gültig wenn:
  - 95 ≤ Distanz ≤ 107 (im Ring)
  - y < 120 (obere Hälfte)

📐 Geometrie & Dimensionen

```
Display: 240x240 Pixel (rund)
Zentrum: (120, 120)

OBERER SLIDER:
├─ Start-Winkel: 180° (links, 9 Uhr)
├─ End-Winkel:   0° (rechts, 3 Uhr)
├─ Bogen-Länge:  180° (halber Kreis)
├─ Außen-Radius: 107 px
├─ Innen-Radius: 95 px
├─ Ring-Breite:  12 px
└─ Segmente:     60 × 3° = 180°

UNTERE ANZEIGE:
├─ Start-Winkel: 0° (rechts, 3 Uhr)
├─ End-Winkel:   180° (links, 9 Uhr)
├─ Bogen-Länge:  180° (halber Kreis)
├─ Außen-Radius: 107 px
├─ Innen-Radius: 95 px
├─ Ring-Breite:  12 px
└─ Segmente:     60 × 3° = 180°

NULLPUNKT-MARKIERUNG:
├─ Position:     270° (12 Uhr, oben)
├─ Innen-Radius: 85 px
├─ Außen-Radius: 107 px
├─ Breite:       2 px (doppelt gezeichnet)
└─ Farbe:        Gelb (0xFFE0)
```

🔢 Wertebereiche & Umrechnung

Oberer Slider

```
Wertebereich:  0 ═══════════ 2048 ═══════════ 4095
               MIN          NULLPUNKT          MAX
                ▯             ▯                ▯
Position:    180°           270°             0°/360°
             (9h)           (12h)            (3h)

Umrechnung:
• Wert → Winkel:  angle = 180° + (value / 4095 * 180°)
• Winkel → Wert:  value = (angle - 180°) / 180° * 4095
• Touch → Wert:   value = touch_angle / 180° * 4095

Beispiele:
├─ Wert 0:     → 180° (ganz links)
├─ Wert 1024:  → 225° (25%)
├─ Wert 2048:  → 270° (Mitte/Nullpunkt)
├─ Wert 3072:  → 315° (75%)
└─ Wert 4095:  → 360°/0° (ganz rechts)
```

Untere Anzeige

```
Wertebereich:  0% ═══════════ 50% ═══════════ 100%
               MIN           MITTEL          MAX
                ▯             ▯               ▯
Position:      0°           90°             180°
              (3h)          (6h)            (9h)

Umrechnung:
• Prozent → Winkel:  angle = percentage / 100 * 180°
• Winkel → Prozent:  percentage = angle / 180° * 100

Beispiele:
├─ 0%:    → 0° (ganz rechts)
├─ 25%:   → 45° (Grün-Start)
├─ 50%:   → 90° (unten)
├─ 80%:   → 144° (Grün-Ende)
└─ 100%:  → 180° (ganz links)
```

🎬 Animations-Sequenzen

Startup-Animation (empfohlen)

```
// 1. Fade-In (0.5s)
for (int i = 0; i <= 100; i += 10) {
  display->set_lower_percentage(0);
  display->set_upper_value(2048);
  // Setze Display-Helligkeit zu i%
  delay(50);
}

// 2. Slider-Sweep (1s)
for (int v = 0; v <= 4095; v += 100) {
  display->set_upper_value(v);
  display->draw();
  delay(10);
}
display->set_upper_value(2048); // Zurück zur Mitte

// 3. Gauge-Fill (1s)
for (float p = 0; p <= 100; p += 2) {
  display->set_lower_percentage(p);
  display->draw();
  delay(20);
}
```

Demo-Loop

```
void demo_loop() {
  static int phase = 0;
  static int counter = 0;

  switch(phase) {
    case 0: // Slider links → rechts
      display->set_upper_value(counter * 20);
      if (counter++ >= 205) { phase++; counter = 0; }
      break;

    case 1: // Slider rechts → mitte
      display->set_upper_value(4095 - counter * 20);
      if (counter++ >= 102) { phase++; counter = 0; }
      break;

    case 2: // Gauge 0% → 100%
      display->set_lower_percentage(counter);
      if (counter++ >= 100) { phase++; counter = 100; }
      break;

    case 3: // Gauge 100% → 0%
      display->set_lower_percentage(counter);
      if (counter-- <= 0) { phase = 0; counter = 0; }
      break;
  }

  display->draw();
  delay(20);
}
```

🔍 Debug-Visualisierung

Serial Monitor Ausgabe

```
=== Display Status ===
Oberer Slider:
├─ Wert:       2567
├─ Position:   56.2% (rechts von Mitte)
├─ Abweichung: +519 vom Nullpunkt
├─ Farbe:      Weiß (Normal-Bereich)
└─ Touch:      Aktiv

Untere Anzeige:
├─ Prozent:    73.5%
├─ Position:   132.3° (Bogen-Position)
├─ Farbe:      Grün (OK-Bereich)
└─ Status:     Normal-Betrieb

Display:
├─ Numerisch:  Aktiv
├─ Framerate:  ~25 FPS
└─ RAM:        ~8KB Buffer
```

💡 Verwendungs-Szenarien

Szenario 1: Temperatur-Regler

```
OBERER SLIDER: Soll-Temperatur
├─ 0 → 4095  =  0°C → 40.95°C
├─ Nullpunkt =  20.48°C (Raumtemperatur)
├─ Rot       =  <2°C oder >39°C (Grenzwerte)
└─ Weiß      =  2-39°C (Normal)

UNTERE ANZEIGE: Heizleistung
├─ 0-10%:    Aus/Minimal (rot)
├─ 10-25%:   Aufheizen (orange)
├─ 25-80%:   Normal-Betrieb (grün)
└─ 80-100%:  Maximale Leistung (weiß)
```

Szenario 2: Audio-Mixer

```
OBERER SLIDER: Balance (L↔R)
├─ 0 → 4095  =  100% Links → 100% Rechts
├─ Nullpunkt =  Mitte (Stereo-Balance)
├─ Rot       =  Extreme L/R (>90%)
└─ Weiß      =  Normal-Balance

UNTERE ANZEIGE: Master-Lautstärke
├─ 0-10%:    Stumm/Sehr leise (rot)
├─ 10-25%:   Leise (orange)
├─ 25-80%:   Normal-Lautstärke (grün)
└─ 80-100%:  Laut (weiß → Clipping-Warnung)
```

Szenario 3: Autopilot (Marine)

```
OBERER SLIDER: Kurs-Sollwert
├─ 0 → 4095  =  0° → 359.87° (Kompass)
├─ Nullpunkt =  180° (Süd)
├─ Rot       =  ±5° Abweichung (Kurs-Fehler)
└─ Weiß      =  Auf Kurs

UNTERE ANZEIGE: Batterie SOC
├─ 0-10%:    Kritisch (rot)
├─ 10-25%:   Niedrig (orange)
├─ 25-80%:   Normal (grün)
└─ 80-100%:  Voll geladen (weiß)
```

Tipp: Für eigene Anwendungen die Wertebereiche einfach mit linearer Interpolation umrechnen:

```
float real_value = min_value + (display->upper_value / 4095.0) * (max_value -
min_value);
```

Version: 1.0.0
Ergänzung zu: DUAL_ARC_DISPLAY_README.md