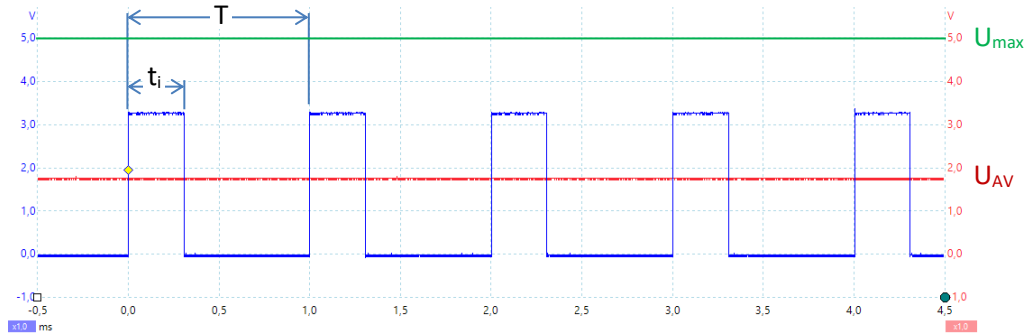


Pulsweitenmodulation Prinzip

Die Pulsweitenmodulation wird oft eingesetzt, wenn ein Verbraucher in der Leistung stetig gesteuert werden soll. Die PWM ist eine Rechteckspannung mit konstanter Periodendauer T . Durch Variieren der Impulsdauer t_i wird der Mittelwert der Spannung am Verbraucher eingestellt.



es gilt:

$$U_{AV} = U_{max} \cdot \frac{t_i}{T}$$

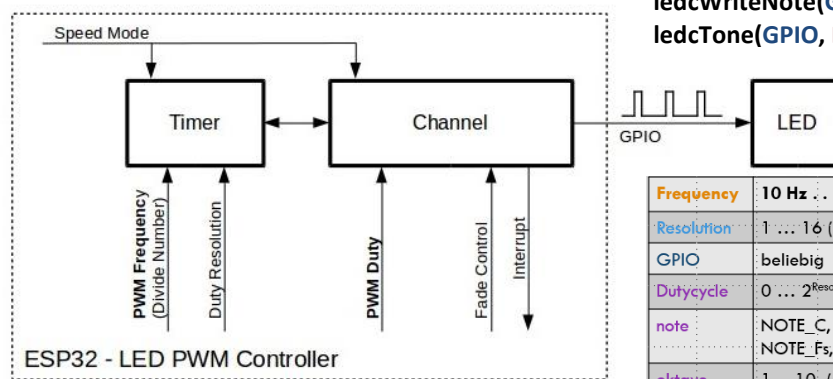
$$U_{AV} = U_{max} \cdot g$$

g : Tastgrad

PWM Hardware-Modul und Funktionen für ESP32

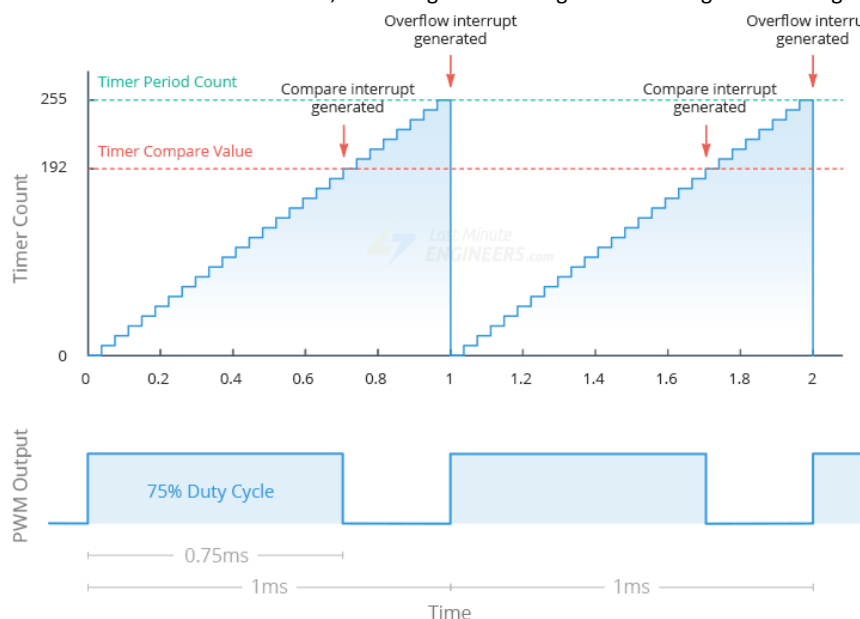
```
ledcAttach(GPIO, Frequency, Resolution);
ledcWrite(GPIO, PWM-Dutycycle);
```

```
ledcFade(GPIO, startcycle, endcycle, time);
ledcWriteNote(GPIO, note, oktave);
ledcTone(GPIO, Frequency);
```




| | |
|------------|-----------------------------------------------------------------------------------------------------|
| Frequency | 10 Hz ... 40.000.000 Hz |
| Resolution | 1 ... 16 (Bit) |
| GPIO | beliebig |
| Dutycycle | $0 \dots 2^{\text{Resolution}} = 0 \dots 100\%$ |
| note | NOTE_C, NOTE-Cs, NOTE_D, NOTE_Eb, NOTE_E, NOTE_F, NOTE_Fs, NOTE_G, NOTE_Gs, NOTE_A, NOTE_Bb, NOTE_B |
| oktave | 1,...10 (Bsp.: Kammerton A' 440Hz: NOTE_A, 4) |

Der ESP32 besitzt 16 PWM-Kanäle, die häufig für die Helligkeitssteuerung von LED eingesetzt werden.



| Resolution Bit | Dutycycle MAX | PWM Frequenz MAX |
|----------------|---------------|------------------|
| 1 | 2 | 40.000.000 |
| 2 | 4 | 20.000.000 |
| 3 | 8 | 10.000.000 |
| 4 | 16 | 5.000.000 |
| 5 | 32 | 2.500.000 |
| 6 | 64 | 1.250.000 |
| 7 | 128 | 625.000 |
| 8 | 256 | 312.500 |
| 9 | 512 | 156.250 |
| 10 | 1024 | 78.125 |
| 11 | 2048 | 39.063 |
| 12 | 4096 | 19.531 |
| 13 | 8192 | 9.766 |
| 14 | 16384 | 4.883 |
| 15 | 32768 | 2.441 |
| 16 | 65536 | 1.221 |

| | | |
|------------------------------------------------------------------------------------------------------------------------------|---------------------------------|-------------------------------------------------|
|  Friedrich-Ebert-Schule Esslingen FES | IT: Hardwarenahes Programmieren | Name: Rahm Datum: 22.01.2023 1_9_PWM.docx |
| | Pulsweitenmodulation (PWM) | 1.9.2 |

Arbeitsauftrag

- Erstellen Sie das untenstehende Programm und überprüfen Sie die Änderung der Helligkeit der LED an GPIO 12 (= LED_BUILDIN).
- Messen Sie die PWM-Spannung mit dem PicoScope an GPIO 2 und die Analogspannung an GPIO 34 (beide auf der 3,3V Seite). Beobachten Sie die Spannung bei Änderung des Tastgrades.
- Ergänzen Sie eine zweite identische PWM-Steuerung für GPIO 2 (= BACKLIGHT) um die Displayhelligkeit zu steuern. Stellen Sie dazu den Jumper "Backlight" auf **pwm/2**.
- Dokumentieren Sie ihre Ergebnisse.

```
#include <esp32CarrierBoard.h>


const int PWM_FREQ = 1'000;
const int PWM_RESOLUTION = 10;
const int MAX_DUTY_CYCLE = (int)(pow(2, PWM_RESOLUTION) - 1);

void setup()
{
    // (GPIO, PWM-Frequenz, Auflösung)
    ledcAttach(LED_BUILDIN, PWM_FREQ, PWM_RESOLUTION);    //GPIO 12
}

int play = false;

void loop()
{ // ADC hat 12-Bit Auflösung
    int hell = analogRead(A3);    // Poti an GPIO 34 (ADC: 12Bit-Auflösung)
    hell = map(hell,0,4095,0,MAX_DUTY_CYCLE);
    ledcWrite(LED_BUILDIN, hell);    // (GPIO, duty_cycle)

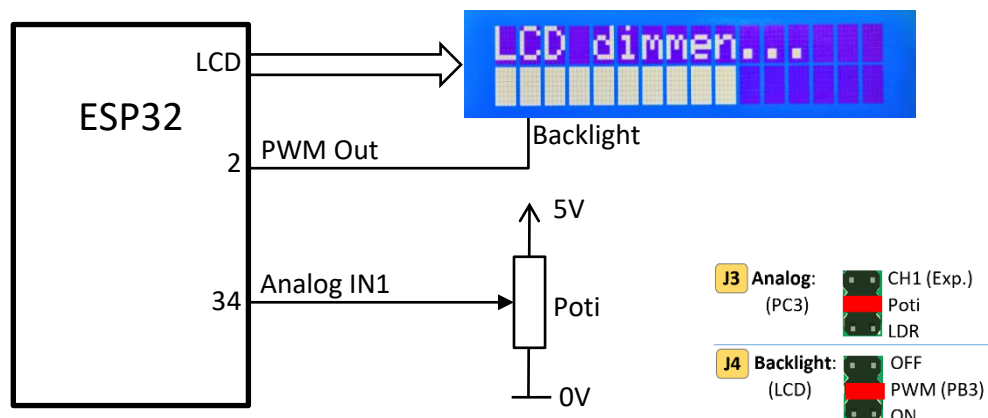
    delay(50);
}
```

| | | |
|------------------------------------------------------------------------------------------------------------------------------|---------------------------------|-------------------------------------------------|
|  Friedrich-Ebert-Schule Esslingen FES | IT: Hardwarenahes Programmieren | Name: Rahm Datum: 22.01.2023 1_9_PWM.docx |
| | Pulsweitenmodulation (PWM) | 1.9.3 |

Projekt: Balkenanzeige für Hintergrundbeleuchtung des LC-Displays

Die Helligkeit der Hintergrundbeleuchtung (LED-Backlight) des LC-Displays kann effektiv über eine vom ESP32 bereitgestellte PWM-Einheit realisiert werden. Zur Visualisierung der Displayhelligkeit soll eine Balkenanzeige realisiert werden. Beim Schreiben des Balkens wird das Blocksymbol (█ = 0xFF) von Links an jeder Cursor-Position angezeigt, deren Wert < der aktuellen Helligkeit ist. Alle Cursor-Positionen die > dem aktuellen Helligkeitswert sind, werden mit Leerzeichen (░ = 0x20) befüllt. Die Anzeige der Balkenelemente erfolgt jeweils mit:

```
lcd.write(0xFF); // Block-Symbol █
lcd.write(0x20); // Leerzeichen (Blank) ░
```



Arbeitsauftrag

- Erstellen Sie das Programm zur Balkendarstellung auf dem LC-Display nach dem nebenstehenden Algorithmus.

Die map()-Funktion wird benötigt, um die 12-Bit-Auflösung (0...4095) des AD-Wandlers auf die 16 Spalten der Anzeige runter zu mappen. Recherchieren Sie die Benutzung der Funktion in der Arduino-Referenz.

Testen Sie das Programm durch Drehen am Potenziometer.

- Anstelle des Potis kann auch der LDR als Analogeingabe dienen. Für eine adaptive Helligkeitseinstellung muss die Hintergrundbeleuchtung mit zunehmender Umgebungshelligkeit steigen. Ist es dunkel, reicht eine geringe Displaybeleuchtung. Stecken Sie dazu nur den Jumper J3 (Analog) auf LDR um.

Testen Sie die Steuerung durch Abdunkeln des LDR.

