PWM

1. Wiederholung: PWM bei einer LED (ohne Timer1)

Wenn man eine Leuchtdiode rasch abwechselnd ein- und ausschaltet, sehen wir ein schwächeres Leuchten, als wenn sie die ganze Zeit eingeschaltet ist. Je länger die An-Phase im Verhältnis zur Aus-Phase ist, desto heller leuchtet sie.

	An-Phase	Aus-Phase	Tastverhältnis	Helligkeit
Beispiel 1	8 ms	2 ms	8:2 bzw. 4:1	recht hell
Beispiel 2	3 ms	3 ms	3:3 bzw. 1:1	mittel hell
Beispiel 3	3 ms	7 ms	3:7	wenig hell

Das Tastverhältnis ist das Verhältnis von An- und Aus-Phase. Es beschreibt den Helligkeitseindruck.

1.1 Vervollständige den Satz: Je _____ das Tastverhältnis ist, desto _____ sehen wir die LED leuchten.

1.2 Die Helligkeit einer LED an PortB.0 soll über das Terminal gesteuert werden. Wir geben dazu die Länge der An-Phase über das Terminal ein. ***Die Summe aus An- und Aus-Phase soll immer 20 ms betragen;*** die eingegebene Zahl darf also höchstens 20 sein.
Das Programm soll die LED nun so lange leuchten lassen, bis der Taster Ta0 betätigt wird. Danach soll eine neue Eingabe der Helligkeit möglich sein, usw.

Schreibe ein entsprechendes Programm mit Hilfe von waitms-Befehlen.

Merke Dir die folgenden Fachausdrücke:

Bei der Helligkeitssteuerung der LED bestimmt das Verhältnis aus An- und Aus-Phase die Helligkeit. An- und Aus-Phase bilden zusammen die sogenannte **Periode**; deren Zeitdauer wird auch **Periodendauer** genannt. Das 1-Signal wird oft als **Puls** bezeichnet; seine Länge wird auch **Pulsweite** genannt. Indem wir bei konstanter Periodendauer die Pulsweite **modulieren (d. h. verändern)**, können wir also die Helligkeit der LED steuern. Auf die gleiche Art und Weise kann man auch Motoren schneller und langsamer laufen lassen.

PWM

2. PWM mit dem Timer1

Bei dem Programm aus 1.2 ist der Mikrocontroller (genauer: die ALU) unaufhörlich mit dem Ein- und Ausschalten sowie den Warte-Befehlen beschäftigt. Für weitere Aufgaben (z. B. Abfragen von Sensorwerten) hat er keine Zeit mehr. Hier kommt unser Timer1 ins Spiel.

Wir wollen eine LED bei PortB.4 nun mit dem Timer1 dimmen.

2.1 Trage in den Initialisierungsteil Deines Programms den Code zum Starten des PWM-Modes 15 ein:

```
Tccr1a = &B00100011     'PWM-Mode 15 mit Prescale 8 einschalten
Tccr1b = &B00011010
Compare1a = 10000       'Periodendauer 2000 us
Compare1b = 1000        'Vergleichswert klein -> geringe Helligkeit
```

2.2. Der Mikrocontroller soll warten, bis der Taster Ta0 betätigt worden ist; er soll dann den Compare1b-Wert um 1000 erhöhen (falls der Compare1a-Wert noch nicht erreicht ist) und anschließend 1 Sekunde warten. Dann soll alles von vorne beginnen.

2.3 Der Mikrocontroller soll die LED mit Hilfe des Tasters a1 auch herunterdimmen können. Erweitere das Programm aus 2.2 entsprechend.

2.4 Skizziere das Timing-Diagramm des PWM-Signals für den Compare1b-Wert 2500. Gib auch die Pulsweite an.

3. Arbeiten mit einem Servo

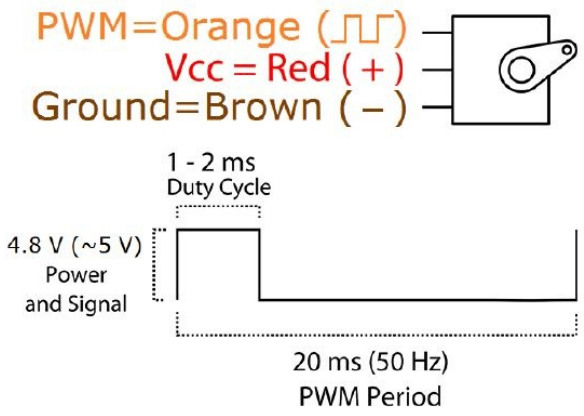
Manche haben schon in der zweiten Veranstaltung mit Servos gearbeitet: Hier noch einmal die wichtigsten Informationen: Servos sind Motoren mit bestimmten Eigenschaften: Sie werden mithilfe spezieller Signale gesteuert. Je nach Signal wird die Drehachse in die gewünschte Position gebracht.

Abbildung 1

PWM

Informationen des Herstellers:

Servo can rotate approximately 180 degrees (90 in each direction)
Operating speed: 0.1 s/60 degree



Position "0" (1.5 ms pulse) is middle, "90" (2 ms pulse) is all the way to the left.

3.1 Welche Zeit benötigt das Servo für einen Schwenk um 180°?

3.2 Auch beim Servo kommt wieder die PWM zum Einsatz. Was wird hier durch die Pulsweite gesteuert?

3.3 Gib an:
(1) die Periodendauer;
(2) die Pulsweite bei der Mittel-Position (Fahne zeigt nach oben, "12 Uhr");
(3) die Pulsweite bei der Links-Position (Fahne zeigt nach links, "9 Uhr")
(4) die Pulsweite bei der Rechts-Position (Fahne zeigt nach rechts, "3 Uhr")

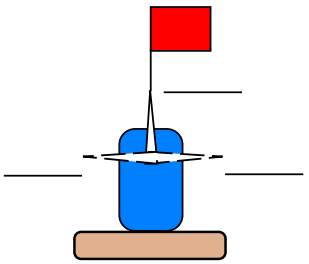


Abbildung 2

3.4 Trage die Ergebnisse aus 3.3 in die Abb. 2 ein.

3.5 Um das Servo anzusteuern, benutzen wir die Vorlage-Datei "servo_sg90_vorlage.bas". In ihr ist der Initialisierungsteil schon wie in 2.1

ergänzt; außerdem findet man dort weitere Erläuterungen. Schreibe ein Programm, welches das Servo im 1-Sekunde-Rhythmus auf 9 Uhr, 12 Uhr, 3 Uhr, 9 Uhr, ... stellt. Nach Betätigen des Tasters Ta0 soll das PWM-Signal ausgeschaltet werden.

Hinweise:

- Achte beim Anschließen des Servos auf die Bedeutung der einzelnen Farben!
- Ermittle die benötigten compare1b-Werte.
- Manchmal steckt das "Horn" nicht richtig auf dem Zahnrad; das kann man mit der 12-Uhr-Einstellung kontrollieren und ggf. korrigieren.
- Auch nehmen es die Hersteller mit den Angaben für den Drehwinkel nicht immer genau; gegebenenfalls kann man die compare1b-Werte anpassen.

3.6 Bestimme die Periodendauer und Pulsweite bei den folgenden Diagrammen. Beachte: Ein Kästchen entspricht der rot markierten Zeitangabe!

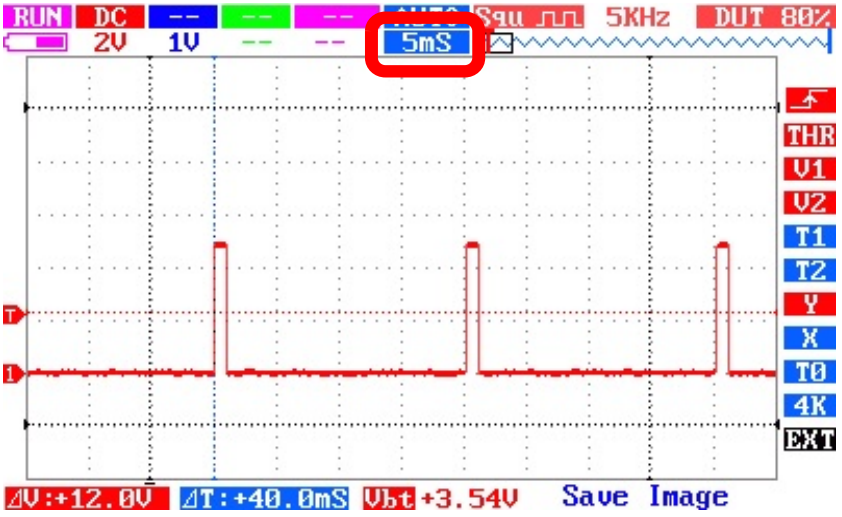


Abbildung 4

3.7 Das Servo soll nun **langsam** permanent hin- und herschwenken. Schreibe ein entsprechendes Programm und teste es aus. Benutze "hinreichend" viele "Zwischenwinkel".

PWM

4. **Projekt: "Servo-Choreographie"**

Kriterien für eine gute Choreographie

- ☞ kreativ (nicht monoton, sondern abwechslungsreich)
- ☞ rhythmisch, nicht chaotisch
- ☞ vernünftige Länge
- ☞ Komplexität (z. B. verschieden Bewegungsgeschwindigkeiten)
- ☞ Gadgets (z. B. Kombination mit Licht und/oder Sound)
- ☞ Übersichtlichkeit des Programms

Hinweise/Tipps:

1. Choreographie planen, z. B. durch Stichworte oder Bildfolge
2. Chorographie-Bausteine (Module) erkennen, zugehörige Unterprogramme schreiben und testen
3. Programm für Gesamtchoreographie schrittweise ausbauen und testen
4. Gadgets ggf. erst am Schluss einbauen