Kurzzeitmessung mit dem Timer1

1*. Gib das folgende Programm "Kurzzeitmessung" bei BASCOM ein. Benutze dazu die Original-Vorlage-Datei (mit der korrekten cfg-Datei) und füge erforderliche Deklarationen ein. Speichere das Programm ab und kompiliere es. Schließe das LCDisplay an die Attiny-Platine an und lade das Programm hoch.

```
'*********************************************************
'******************** Hauptprogramm ***********************

Do
  Cls
  Lcd "Test"
  Wait 2
  Cls

' Messung
  Timer1 = 0                      'Timer1-Counter auf 0
  Tccr1b = &B000000001            'Timer1 an; Clock/1, d. h. 1 Count = 0,25 us
  waitus 100                      'Wartet 100 us
  Zeit = Timer1
  Tccr1b = 0                      'Timer1 aus

  Lcd Zeit                       'Zeit-Angabe in Counts
  Wait 2
Loop
```

Kontrolliere, ob das Ergebnis von Messung zu Messung gleich bleibt. Notiere den Messwert. Welcher Zeit in µs entspricht der Messwert? Überprüfe ob das Ergebnis sich ändert, wenn man die Befehle **`Zeit = Timer1`** und **`Tccr1b = 0`** austauscht.

2*. Ändere das Programm so ab, dass es die Zeit in µs angibt; ändere auch die Kommentare entsprechend.

3. Ermittle, wie viel Zeit in µs der Mikrocontroller für eine for-next-Zählschleife (Index vom Typ Byte) von 1 bis 100 braucht.

4*. Ändere das Programm aus Aufg. 2 ab, indem Du als Prescale für den Timer1 den Wert 8 wählst. (Vgl. Info-Box auf der Seite 2!) Kontrolliere, ob das Programm denselben Zeitwert (in µs) anzeigt!

5. Bestimme für 4 weitere BASCOM-Befehle die benötigte Zeitdauer: print "HALLO", LCD "Hallo", CLS und einen weiteren selbst gewählten Befehl. Welcher braucht die meiste Zeit, welcher am wenigsten?

Hinweis: Achte darauf, die Wertzuweisung der Variablen nicht mitzumessen!

Kurzzeitmessung mit dem Timer1

Der Timer1 kann mit verschiedenen Geschwindigkeiten laufen. Die normale Geschwindigkeit bei unserer Platine ist 1 count (Zähler) pro 1/4 Mikrosekunde, d. h. 4 counts in 1 Mikrosekunde.

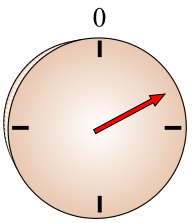
Man kann den Timer1 aber auch langsamer laufen lassen. Das ist so wie bei einem Getriebe mit verschiedenen Untersetzungen: Je höher die **Untersetzung** ist, desto langsamer läuft der Timer1. Bei den Mikrocontrollern bezeichnet man diese Untersetzung als Prescale-Wert oder auch einfach nur kurz **Prescale**. Manchmal spricht man auch von einem **Unterteiler**.

Um den Prescale einzustellen, benutzt man wieder TCCR1B. Beispiel: Durch den Befehl TCCR1B = 2 wird der Timer1 mit dem Prescale 8 gestartet. Nun läuft der Timer1 8 mal so langsam wie normal; erst nach 8*1/4 Mikrosekunden geht der Timer1 um 1 count weiter.

Tccr1b	Aktion u. Unterteiler	Zeit/Count
0	Timer1 stoppen	
1	Timer1 starten; *1	0,25 us
2	Timer1 starten; *8	2,0 us
3	Timer1 starten; *64	16,0 us
4	Timer1 starten; *256	64,0 us
5	Timer1 starten; *1024	256,0 us

6. Etwas Theorie

6.1* Nach einem vollständigen Umlauf des Zeigers kommt es zu einem Überlauf (Overflow). Bestimme die Zeit für einen solchen Umlauf bei den Prescale-Werten 1, 64, 1024 (Angaben in µs bzw. ms).



6.2* Wie groß ist die maximale Zeit, die mit dem Timer1 gemessen werden kann, welche die minimale?

6.3 Du sollst Zeiten von etwa 20 ms mit dem Mikrocontroller möglichst genau messen. Warum sind Unterteilungen von 1024 bzw. 1 hier nicht sinnvoll.

7. Reaktionstester (Mini-Projekt)

Mit der Timer1-Komponente können wir einen äußerst genauen Reaktionstester bauen. Vorschlag: Ein kurzes Piep-Signal oder das Aufblitzen einer LED bei PortD.6 fordern den Nutzer auf, möglichst rasch die Taste Ta1 zu betätigen. (Es ist ratsam, diese Taste zu benutzen, da diese nicht entprellt ist und daher rascher reagiert.) Die Zeit zwischen dem Signal/Aufblitzen und dem Drücken der Taste wird dann auf einem LCD in Millisekunden angezeigt. Gegebenenfalls kann anschließend auch eine Bewertung der Reaktionsleistung angezeigt werden.