

Versuch 4 - Messung der relativen Luftfeuchtigkeit

Einleitung

In diesem Versuch werden Sie die relative Luftfeuchtigkeit mit einem kapazitiven Feuchtesensor messen. Angeschlossen an eine astabile-Kippstufe (Multivibrator) erhält man einen Oszillator mit Rechtecksignal, dessen Frequenz direkt von der Sensorkapazität, und damit von der relativen Luftfeuchtigkeit, abhängt. Die Implementierung erfolgt auf dem PSoC5.

Über ein definiertes Zeitfenster werden die Rechteckschwingungen gezählt, auf dem LCD-Panel ausgegeben und über die RS232-Schnittstelle an den Miniserver gesendet. Dort werden die Messwerte in % relative Luftfeuchte (%RH) umgerechnet und grafisch dargestellt.

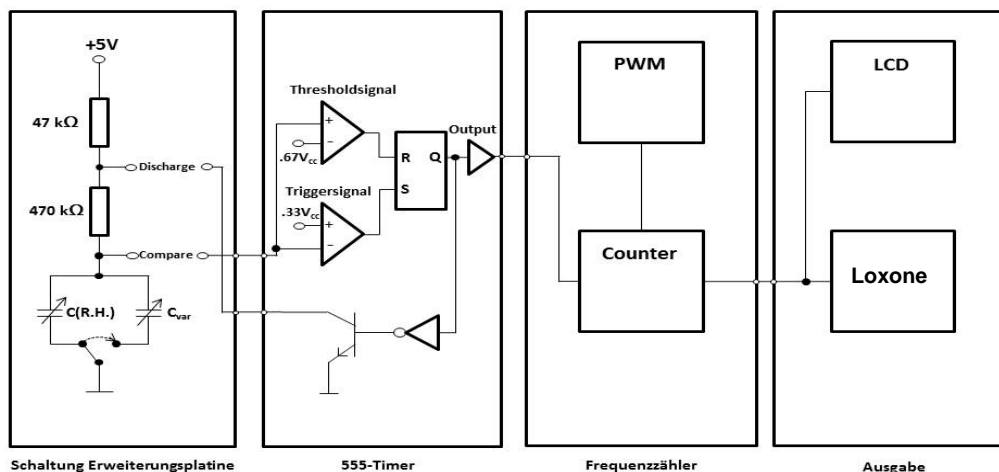
Feuchtesensor HS1101

Der Feuchtesensor HS1101 der Firma Humirel besteht aus einer Kunststoffolie, die beidseitig mit Gold beschichtet ist. Die Beschichtungen bilden die Kondensatorplatten und die Kunststoffolie das Dielektrikum. Das Dielektrikum (und damit die Sensorkapazität) ändert sich mit der relativen Luftfeuchtigkeit.

Messbereich	0 – 100 %RH
Betriebstemperaturbereich	-40 °C – 100 °C
Kapazität bei 25 °C, 55 %RH, 10kHz,	180 pF ± 3 pF
Ansprechzeit (63%-Wert) von 33% auf 76%-RH, bei 25 °C in ruhender Luft	ca. 5 s

Relative Feuchte R.H. [%]	Kapazität C(R.H.) [pF] bei 25°C, 10kHz
0	162
10	166
20	169
30	172
40	175
50	178
60	182
70	185
80	189
90	194
100	199

Versuchsaufbau / Blockschaltbild



Beschreibung:

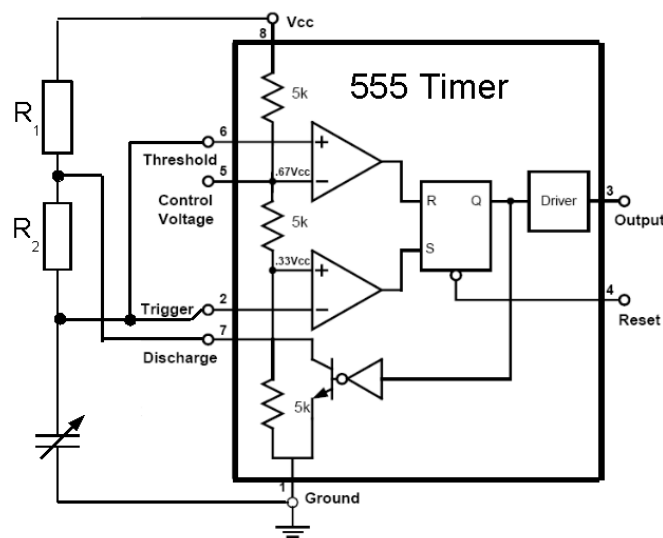
Der Feuchtesensor und die beiden Widerstände R1 und R2 bilden zusammen mit dem 555-Timer einen Oszillator mit Rechteckausgang. Die Kapazität des Feuchtesensors wird über R1 und R2 aufgeladen und über einen Low-Pegel des Discharge-Signals über R2 entladen. Die Spannung des Kondensators steht dem 555-Timer über die Compare-Leitung zur Verfügung.

Der 16-bit-Counter wird mit dem Rechtecksignal des 555-Timers getaktet. Mit dem Pulsweitenmodulator (PWM) wird der Counter in äquidistanten Zeitabschnitten ausgelesen, zurückgesetzt und neu gestartet. Dieser 16-Bit Wert wird per Software an das LCD-Panel und die serielle Schnittstelle ausgegeben.

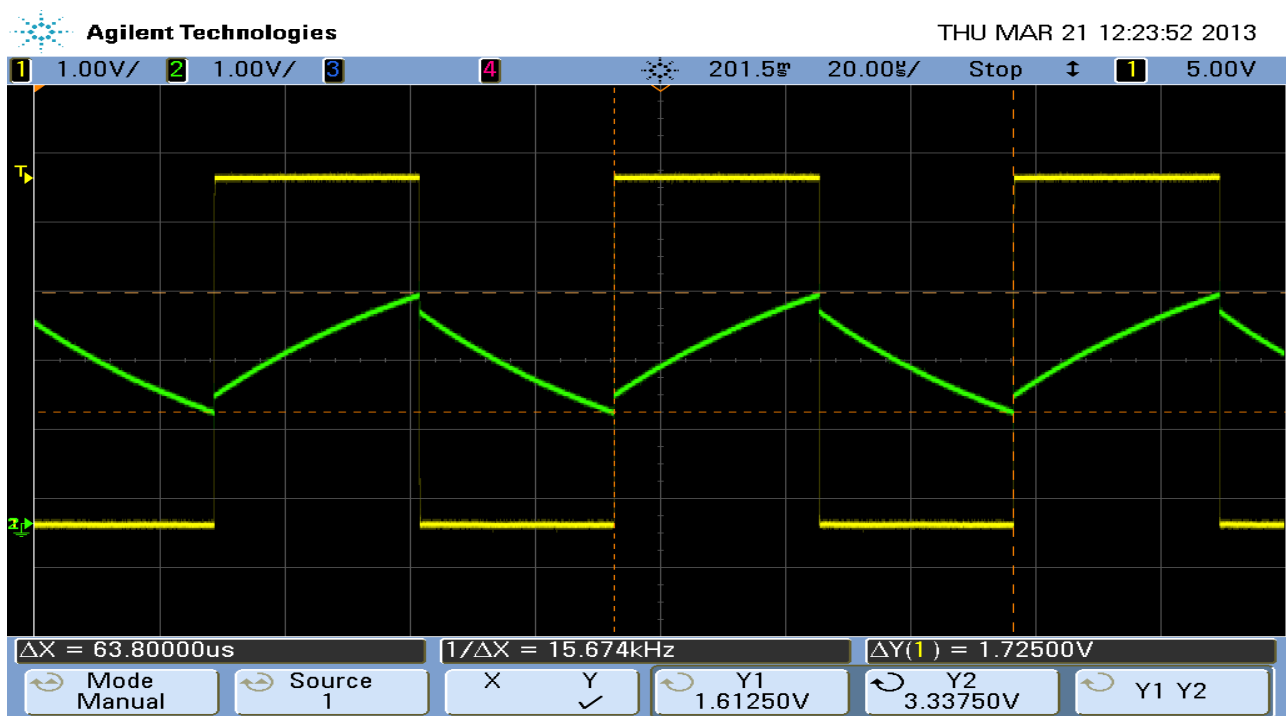
555-Timer

Der 555-Timer Baustein, der 1971 von Hans Camenzind für die Firma Signetics entwickelt wurde, ist der weltweit meistverkaufte IC. Mit nur wenigen externen Bauteilen können damit viele einfache Funktionen, wie Timer, Frequenzteiler, Sägezahngenerator, PWM, Oszillator, Pulsgenerator, FlipFlop, Schalterentprellung, Tongenerator,..... realisiert werden. Von den drei Betriebsmoden (monostabil, astabil und bistabil) wird im Praktikum für den Bau eines Oszillator nur der astabile Modus benutzt. Der 555-Timer kommt nicht als separater IC zum Einsatz, sondern wird im PSoC mit digitalen und analogen Blöcken nachgebildet.

Im unten stehenden Bild ist der klassische 555-Timer, in der Beschaltung mit 2 Widerständen und einem Kondensator, als astabile Kippschaltung zur Erzeugung eines Rechtecksignals abgebildet. Er beinhaltet 2 Spannungskomparatoren, deren Schwellwert über drei 5kΩ Widerstände eingestellt werden, ein bistabiles Flip-Flop, einen Entladetransistor und einen Ausgangstreiber.



Über die Versorgungsspannung V_{cc} und die beiden Widerstände R_1 und R_2 wird der Kondensator zunächst aufgeladen. Überschreitet die Kondensatorspannung den Schwellwert des Threshold-Komparators, so schaltet dieser den Q-Ausgang des Flip-Flops auf Low. Mit dem negierten Q-Signal wird dann der Entladetransistor durchgeschaltet und somit der Kondensator über die Discharge-Leitung und R_2 entladen. Sobald der Schwellwert des Trigger-Komparators unterschritten wird, setzt dieser das Flip-Flop wieder auf High und der Ladevorgang beginnt von neuem.



Gelbes Signal	Rechtecksignal am Output-Pin
Grünes Signal	Spannungsverlauf des Kondensators während dem Lade- und Entladevorgang
Periodendauer Rechtecksignal	$T = (R_1 + 2 \cdot R_2) \cdot C \cdot \ln 2 \approx 0,7 \cdot (R_1 + 2 \cdot R_2) \cdot C$
Ausgang Flip Flop	Output = ((Output \wedge !Threshold) \vee Trigger \vee Reset

Die Funktionalität eines 555-Timers kann vollständig auf einem PSoC realisiert werden.

Frequenzzähler

Mit einer PWM und einem Counter kann ein einfacher Frequenzzähler erstellt werden. Mit dem Counter werden die steigenden Taktflanken des zu messenden Signals gezählt. Der PWM aktiviert mit seinem Ausgangssignal den Counter über dessen *Enable*-Eingang. Der Counter soll zählen, während der PWM einen Low-Pegel (Zeitfenster) ausgibt. Sobald der PWM-Ausgang auf High wechselt, wird ein Interrupt ausgelöst. In dessen Unterbrechungsroutine (Interrupt Service Routine, ISR) wird der Counter ausgelesen, gestoppt, des Zähl-

lerregisters zurückgesetzt und neu gestartet. Nach dem Neustart beginnt er noch nicht zu zählen, sondern erst, wenn der *Enable*-Eingang wieder auf High geht.

Komponenten

Im Folgenden finden Sie einen Überblick zu den benötigten Komponenten.

Clock

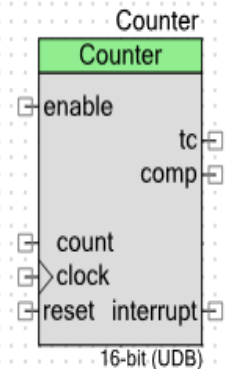
Komponente zur Erzeugung von Taktsignalen. In der Konfiguration der Komponente kann entweder eine vorhandene Taktquelle ausgewählt oder ein neuer Takt erzeugt werden. Dies geschieht entweder durch Wahl der Taktquelle und eines Teilers oder einfach durch Angabe des gewünschten Taktes. PSoC Creator generiert diesen dann automatisch.



Counter

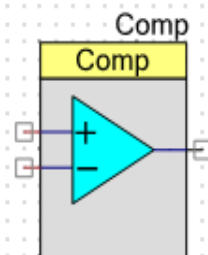
Der Counter ist mit dem PWM nahezu identisch. Über einen Enable-Eingang kann der Counter16 gesteuert werden. D.h. liegt ein High-Pegel an, ist der Counter aktiv. Der minimal einstellbare Zählerwert ist 2, der größte wird durch die Bitbreite des Counters vorgegeben (im Beispiel rechts: 16bit).

Bei jeder Flanke am Eingang 'count' wird das Zählerregister dekrementiert. Der Eingang 'clock' dient der Synchronisation (siehe Datenblatt). Der Ausgang 'tc' geht auf High, sobald das Zählerregister 0 erreicht. Der Ausgang 'comp' ist optional, er geht auf High, wenn das Zählerregister einen voreingestellten Vergleichswert erreicht hat. Viele Funktionen des Counters lassen sich auch über seine API ansteuern.



Comparator

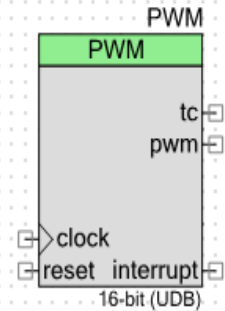
Die Comparator-Komponente vergleicht zwei analoge Signale. Bei nichtinvertierendem Betrieb geht der Ausgang auf High, wenn die Spannung am positiven Eingang größer ist als die Spannung am negativen Eingang. Der Ausgang des Komparators kann entweder fest mit anderen Komponenten verdrahtet werden, oder in Software durch die API abgerufen werden.



PWM

Die Komponente PWM stellt einen Pulsweitenmodulator zur Verfügung und kann entweder in einem Fixed Function Block (FF) oder in einem Universal Digital Block (UDB) untergebracht werden. Eine UDB-PWM bietet vielfältigere Konfigurationsmöglichkeiten, beide Implementierungen unterstützen 8 und 16 bit Auflösung.

Zu Beginn wird der Wert des Periodenregisters ins Zählerregister geladen. Der Zähler wird mit jeder Flanke am Takteingang **clock** heruntergezählt. Hat der Zähler den durch das Tastverhältnis (CMP Value) vorgegebenen Wert erreicht, geht der Ausgang **pwm** auf High. Sobald der Zähler null erreicht hat, geht der Ausgang **tc** (für Terminal Count) für einen Takt auf High, der Ausgang **pwm** geht wieder auf Low und der Vorgang beginnt von neuem.



ISR

ISR steht für Interrupt Service Routine. An diese Komponente angeschlossene Signale können einen Interrupt auslösen. In der Konfiguration stehen dazu drei Interrupttypen zur Wahl: Rising Edge, Level und Derived. Bei letzterem wählt der PSoC Creator je nach Interruptsignalquelle zwischen Rising Edge und Level automatisch den passenden aus.



Aufgabenstellung

Jede gelöste Teilaufgabe muss immer zuerst getestet werden, bevor die nächste bearbeitet wird. Sie sollten sich an die vorgegebene Reihenfolge halten und bei jeder Teilaufgabe mit der Markierung "→**Betreuer**" einem Betreuer das Programm zeigen und das Projekt unter einem anderen Namen fortführen. Klicken Sie hierzu im *Workspace Explorer* mit einem Rechtsklick auf *Project ,Projektname'* und wählen Sie *copy*. Anschließend fügen Sie, mit einem Rechtsklick auf eine freie Stelle im *Workspace Explorer* und Auswahl *paste*, das neue Projekt hinzu. Ein neues Projekt wird erstellt. Dieser Vorgang dauert einige Sekunden. Geben Sie dem Projekt einen prägnanten Namen.

Aufgabe 1: 555 Timer

Bilden Sie mit Komparator-Modulen und Logikgattern eine astabile Kippschaltung.

Hierbei können Sie auf zwei verschiedene Weisen vorgehen. Entweder Sie bauen das benötigte RS-FlipFlop mit NOR oder NAND Gattern nach und verdrahten die Set- und Reseteingänge mit den entsprechenden Signalen, oder sie bauen die Schaltung direkt gemäß der in der Einleitung genannten logischen Gleichung auf. Diese beschreibt die cha-

rakteristische Funktion eines RS-FlipFlops. In beiden Fällen sollten Sie darauf achten ihre Logikschaltung mit einer Clock zu takten, um jederzeit einen stabilen Ausgangszustand zu erhalten. Zudem gilt: Output = Discharge. Ordnen Sie dem Discharge den Port 3_4 zu. Setzen Sie den Drive Mode auf *Open Drain, Drives Low*. Die benötigten Digitalbausteine finden Sie im *Component Catalog* unter *Digital/Logic*. Das *Compare* Eingangssignal liegt an Port 3_5 an.

Die beiden logischen Signale Threshold und Trigger erzeugen Sie mit Komparatoren. Um die Threshold- und Trigger-Referenzspannungen ($0,67 \cdot V_{CC}$ bzw. $0,33 \cdot V_{CC}$) zu erzeugen verwenden Sie zwei DA-Wandler. Die Reset- und Control-Voltage-Funktionen müssen nicht implementiert werden.

Testen Sie mit einem Oszilloskop den Ausgang des 555-Timers.

→**Betreuer**

Aufgabe 2: Frequenzzähler

Erzeugen Sie mit einer **16-bit-UDB-PWM** ein Rechtecksignal mit der von Ihnen in der Vorbereitung berechneten Periodendauer und einem Tastverhältnis von ca. 10%. Erzeugen Sie mit einem Clock-Modul das entsprechende Taktsignal.

Prüfen Sie das Rechtecksignal, indem Sie eine LED anschließen.

Lösen Sie mit dem PWM Interrupts aus (siehe Einleitung *Interrupts*). Setzen Sie dazu in den Eigenschaften *CMP Type 1* auf *Less or Equal* und aktivieren im *Advanced*-Tab die Option *Interrupt on Compare 1 Event*. Klicken Sie dann auf *Generate Application*. In der daraufhin erstellten Datei (*isr_1.c* im *Workspace Explorer*) setzen Sie eine im Hauptprogramm definierte, globale Variable (siehe *Praktikum_Einfuehrung.pdf*) auf eins. Beachten Sie, an welcher Stelle in der ISR Benutzercode eingefügt werden kann und wie bei einem PWM mit sich wiederholenden Interrupts umzugehen ist. Hierzu finden Sie in den jeweiligen Datenblättern reichliche Information. Warten Sie im Hauptprogramm in einer Schleife, bis diese Variable gesetzt wurde (pollen) und setzen Sie sie anschließend zurück. Testen Sie die Programmverzögerung, indem Sie in Ihrem Hauptprogramm eine lokale Variable inkrementieren und auf dem LCD-Panel ausgeben. →**Betreuer**

Platzieren sie einen **16-bit-UDB-Counter** neben dem PWM und takten Sie den Counter mit dem Rechtecksignal des 555-Timers (verwenden Sie hierzu den *Count*-Eingang) und schließen Sie an den *Enable*-Eingang den Ausgang des PWM an. Geben Sie auf dem LCD-Panel den Zählerstand, sowie die Differenz zum Anfangswert aus. →**Betreuer**

Übertragen Sie die Daten wie im vorherigen Versuch per RS232-Schnittstelle an den Miniserver. Als Steuercode verwenden Sie *0010b* (für Sensor 2).

→**Be-**

treuer

Teil 3: Verarbeitung/Darstellung der Messdaten

Zur Messwertverarbeitung und -darstellung erweitern Sie Ihr LoconeConfig-Programm aus dem vorherigen Versuch.

Verwenden sie zur Berechnung der relativen Luftfeuchtigkeit aus den Kapazitätswerten verwenden Sie eine Polynomapproximation dritter Ordnung der Form:

$$A+Bx+Cx^2+Dx^3$$

Mit den Koeffizienten:

Koeffizient	Wert
A	3731,68
B	-70,86
C	0,43
D	$-8,33 \cdot 10^{-4}$

Bedingt durch Rauschen und äußere Störeinflüsse schwankt der Messwert ständig um kleine Beträge. Damit Sie die relative Luftfeuchtigkeit besser ablesen können sollten Sie das Signal glätten.

Mit dem Jumper JP3 können Sie den Sensor durch einen Trimmkondensator ersetzen und Kapazitäten im Bereich von 152 bis 185 pF einstellen. Stellen Sie sicher, dass sich die Anzahl der Rechteckflanken mit 12 Bit (max.4096 Flanken) darstellen lassen und passen Sie evtl. das Zeitfenster an.

→**Betreuer**