Kapitel 2

Pulsweitenmodulation

Die sogenannte Pulsweitenmodulation (kurz PWM) ist ein Rechtecksignal mit konstanter Periodendauer, das zwischen zwei verschiedenen Spannungspegeln oszilliert. Prinzipiell wird das Signal also in schneller Folge ein- und ausgeschaltet. Das Verhältnis von Einschaltzeit zu Ausschaltzeit kann dabei variieren und bildet das Tastverhältnis (den Duty-Cycle).

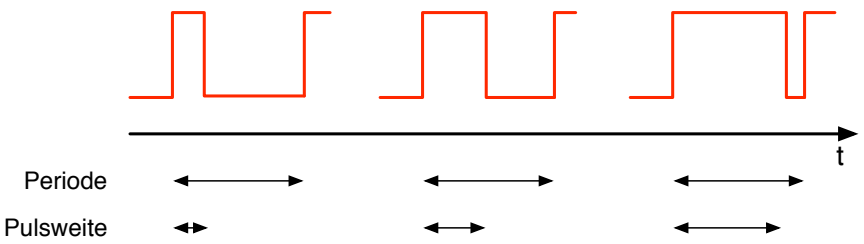


Abb. 2.1 Pulsweitenmodulation

Wie in der Skizze zu sehen ist, bleibt die Periodendauer / Frequenz konstant und nur die Pulsweite ändert sich.

Dieses Signal muss nun demoduliert werden, um daraus eine nutzbare Größe zu erhalten. Das geht z. B. mit einem Tiefpassfilter, mit dem ein einfacher Digital-Analog-Wandler entstehen würde.

Die mittlere Spannung berechnet sich dabei durch

$$U_m = U_{aus} * (U_{ein} - U_{aus}) * \frac{t_{ein}}{t_{ein} + t_{aus}}$$

Da in den meisten Fällen direkt zwischen 0 V und Vcc umgeschaltet wird, lässt sich die Formel in

$$U_m = V_{cc} * \frac{t_{ein}}{t_{ein} + t_{aus}} = V_{cc} * DC$$

vereinfachen. Wie man sieht, ist der Mittelwert der Spannung direkt vom Duty-Cycle DC abhängig.

Träge Verbraucher, wie beispielsweise Glühlampen und Elektromotoren, können direkt an ein PWM-Signal angeschlossen werden und auf diese Weise gedimmt bzw. in ihrer Drehzahl beeinflusst werden. Hingegen wird bei LEDs die Demodulation in diesem Fall durch das Auge vorgenommen, das entsprechend träge bei der Verarbeitung der eintreffenden Lichtsignale ist.

Aufgabe 1

In diesem Teil soll die Helligkeit von LEDs über eine PWM gesteuert werden, da sich LEDs nicht (richtig) über die Spannung dimmen lassen.

Erstellen Sie eine Komponente, die eine PWM mit 8 Bit Auflösung realisiert. Für den Duty-Cycle sollen also Werte von 0x00 bis 0xFF übergeben werden können. Sie kommen dafür mit einem einfachen Zähler und einem Komparator (realisierbar mit einem Vergleichsoperator) aus. Achten Sie darauf, dass die Komponente taktsynchron arbeitet. Dabei sollen wieder alle Einheiten der Schaltung direkt vom 50 MHz Takt CLOCK_50 gespeist werden.

Die PWM-Komponente soll über Eingänge für den Takt und den Duty-Cycle sowie über einen Ausgang für das PWM-Signal verfügen.

Es sollen vier Instanzen der PWM jeweils eine grüne LED treiben. Verwenden Sie bei dreien einen festen Duty-Cycle von 25 %, 50 % und 75 %. Der Duty-Cycle der vierten PWM soll über Schiebeschalter frei eingestellt werden können.

Führen Sie eine Timing-Simulation durch und stellen Sie ein „sauberes“ Verhalten ihrer PWM-Signale sicher.

Programmieren Sie die Schaltung auf das Board und beobachten Sie das Verhalten der LEDs, insbesondere bei der PWM mit dem veränderlichen Duty-Cycle.

Aufgabe 2

Berechnen Sie die Frequenz der von Ihnen erstellten PWM. (Ist sehr einfach.) Überprüfen Sie Ihr Ergebnis durch die Simulation.

Aufgabe 3

Nun soll es ermöglicht werden, die Frequenz der PWM frei einzustellen. Dazu muss die Dauer bis zum Überlauf des Zählers verändert werden können. Wird beispielsweise der Zählraum bis zum Überlauf verdoppelt, so halbiert sich die Frequenz der PWM. Durch freie Einstellbarkeit der Überlaufbedingung lassen sich auch Zwischenfrequenzen mit akzeptabler Genauigkeit erzeugen. Die Vergleichsbedingung am Komparator muss natürlich entsprechend angepasst werden, da sonst der Duty-Cycle nicht mehr stimmt.

Im Folgenden sollen weiterhin 8 Bit Auflösung für den Duty-Cycle verwendet werden. Der Zeitpunkt des Überlaufs soll mit 16 Bit frei einstellbar sein. Sie müssen also den Port Ihrer PWM-Komponente um ein einsprechendes Eingabesignal erweitern.

Stellen Sie eine Formel auf, welche die 8 Bit Eingabe des Duty-Cycle auf die Periodendauer des Zählers bis zum Überlauf abbildet. *Tipp:* Es handelt sich um eine einfache lineare Skalierung. Sie benötigen dazu neben dem 8 Bit Wert des Duty-Cycle den 16 Bit Wert für den Zeitpunkt des Zählerüberlaufs als Eingaben. Beachten Sie die Bitbreite von Zwischenergebnissen und stellen Sie sicher, dass der Ergebniswert wieder 16 Bit breit ist, um als Eingabe für den Komparator dienen zu können.

Führen Sie die entsprechenden Erweiterungen Ihrer PWM-Komponente durch und implementieren Sie die erstellte Formel in Hardware, um die freie Einstellbarkeit der PWM-Frequenz zu erreichen.

Führen Sie Timing-Simulationen durch, um Ihre Implementierung (Frequenz und Duty-Cycle) zu überprüfen. Stellen Sie beispielsweise eine PWM-Frequenz von 30 kHz ein.

Aufgabe 4

Die PWM-Komponente soll nun frei programmierbar werden. Da das Board jedoch nur über 18 Schiebeschalter verfügt, müssen die Werte für den Teilfaktor (an welcher Stelle der interne Zähler überläuft) und den Duty-Cycle in Registern zwischengespeichert werden. Der Zugriff auf die Komponente soll deshalb über einen 8 Bit breiten Datenbus erfolgen, wobei die Datenleitungen für den Lese- und den Schreibzugriff getrennt sein dürfen. Eine Übersicht finden Sie in Abbildung 2.2.

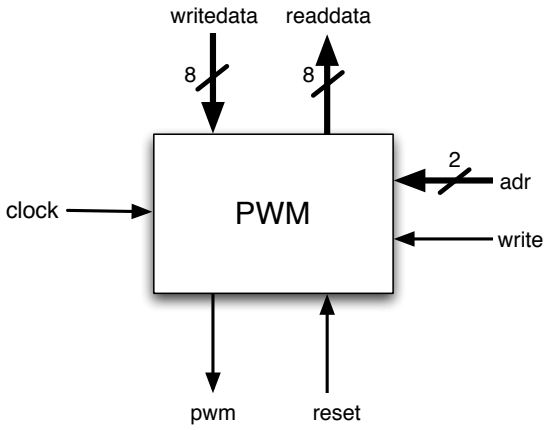
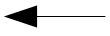


Abb. 2.2 PWM-Komponente

Über den Datenbus sollen vier verschiedene Register angesteuert werden, die über die Adressleitung auswählbar sind. Die Registertabelle soll folgendermaßen aufgebaut sein:

Adresse	**Funktion** Bitreihenfolge: 7 .. 0	
0	Duty-Cycle	
1	Teilfaktor – Bits 7 bis 0	
2	Teilfaktor – Bits 15 bis 8	
3		E



Bit 0 ist enable

Über das Enable-Bit an der Stelle 0 an Adresse 3 soll der PWM-Generator ein- und ausgeschaltet werden können, damit z. B. während der Programmierung der Parameter keine möglicherweise für die angeschlossene Hardware fatalen Signale entstehen können. Über das write-Signal soll gesteuert werden, wann die Eingabe auf dem Datenbus in das momentan adressierte Register übernommen wird. Das Reset soll asynchron erfolgen und die ganze Komponente (alle Register) auf Null zurücksetzen (und damit natürlich auch die Ausgabe des PWM-Signals unterbinden).

Implementieren Sie die oben vorgestellte Komponente.

Binden Sie die Komponente in Ihre Top-Level-Entity ein. Als Taktsignal soll wieder direkt der 50 MHz Takt verwendet werden. Als Dateneingabe dienen die Schiebeschalter 7 bis 0, als Ausgabe die roten LEDs 7 bis 0. Benutzen Sie SW17 für den Reset, SW16 für das write-Signal, SW15 für das Adressbit 1 und SW14 für das Adressbit 0. Die Ausgabe des PWM-Signals soll über eine grüne LED erfolgen.