Kapitel 9

Lüfterregelung

Als abschließendes Projekt wird nun ein etwas komplexeres eingebettetes System aus einer Kombination von Hard- und Software entwickelt und getestet. Es soll eine Motorregelung entworfen und implementiert werden, die ein System mit einem Gleichstrommotor auf einer konstanten Drehzahl hält. Als Grundlage für die Ansteuerung dient dabei das bereits entwickelte PWM-Modul. Für die Rückkopplung der Motordrehzahl muss noch ein weiterer Hardwarebaustein erstellt werden, der das Tachosignal des Motors auswerten kann. Die Regelung des Systems soll dann in Software auf einem integrierten Nios II Softcore-Prozessor erfolgen.

Aufgabe 1 - Drehzahlmessung - Hardware

Der zu regelnde Motor besitzt ein rechteckiges Tachosignal mit mehreren Impulsen pro Umdrehung, wobei das Signal in den üblichen Drehzahlbereichen einen Duty-Cycle von etwa 50% aufweist. Entwickeln Sie eine Hardwarekomponente in VHDL, welche aus einem gegebenen Tachosignal die Motordrehzahl ermittelt und über den Avalon-Bus von einem Nios II Prozessor ausgelesen werden kann. Berücksichtigen Sie in Ihrem Konzept, welche der Funktionen und Umrechnungen in Hardware ablaufen müssen/sollen und welche sinnvoller in der Treibersoftware implementiert werden. Da es bei dem von uns verwendeten Tachosignal je nach Motor nur einen bis acht Pulse pro Umdrehung gibt, erscheint es aus Gründen der Messgenauigkeit wenig sinnvoll, die Anzahl der Flanken in einem festen Zeitraum zu zählen. Deutlich genauer und mit einer höheren Update-Rate bzw. geringeren Latenz ver-

bunden ist die Messung der Zeitdauer zwischen den Impulsen oder die Dauer der Impulse. Bezogen auf das Rechteck-Signal also die Dauer zwischen zwei gleichartigen Flanken (beide steigend bzw. fallend) oder die Dauer von einer steigenden zur nächsten fallenden Flanke (oder invers). Das entsprechende Messverhalten (die Wahl der Flanken) soll in Ihrer Komponente zur Laufzeit änderbar sein. Überlegen Sie sich nun, wie Sie die jeweiligen Zeiten messen können und implementieren Sie eine Komponente, welche diese Messungen durchführt. Sie können davon ausgehen, dass der Motor mindestens eine Umdrehung pro Sekunde macht. Leider kommt es nach den Flanken jeweils zu einem Prellen von einigen µs, Sie sollten daher bereits in der Hardwarekomponente eine Entprellung vorsehen. Legen Sie diese so an, dass deren Dauer zur Laufzeit konfigurierbar ist.

Aufgabe 2 - Drehzahlmessung - Hardwareanbindung

Erweitern Sie Ihr Modul aus Teil 1 nun um eine Anbindung an den Avalon-Bus. Überlegen Sie, welche Daten in welche Richtung übertragen werden müssen (Messdaten, Konfigurationsdaten, etc.). Machen Sie sich Gedanken um die Datenformate und eine sinnvolle Aufteilung in vom Prozessor adressierbare Register. Behalten Sie bei allen diesen Überlegungen auch die Treiber auf der Softwareseite im Hinterkopf. An welche Stelle wird der Teilfaktor (1 – 8 Impulse pro Umdrehung) berücksichtigt, wo erfolgt die Umrechnung in Umdrehungen pro Minute? Wie kann der Treiber zur Laufzeit das Messverhalten konfigurieren (Wahl der Flanken)?

Aufgabe 3 - Drehzahlmessung - Software

Erstellen Sie einen Treiber für das zuvor entwickelte Hardwaremodul zur Drehzahlmessung. Schreiben Sie Funktionen, welche die grundlegenden Aufgaben erfüllen wie

- Rücklieferung der Motorgeschwindigkeit in Umdrehungen pro Minute
- Anpassung an den Motor bzw. das Tachosignal (Pulse pro Umdrehung)
- Konfiguration des Messverhaltens (Flanken)

Die bereitgestellte Platine erwartet am Stecker auf Pin 2 das PWM-Signal für den Motor und liefert auf Pin 4 das Tachosignal zurück. Dies entspricht dann den Signalen `GPIO_x(1)` bzw. `GPIO_x(3)`.

Aufgabe 4 - Regelung

Zu einer lauffähigen Motorregelung fehlt nun noch die Software, welche die eigentliche Regelung durchführt. Einen Überblick über die Methoden der Regelungstechnik ist beispielsweise unter
http://www.roboternetz.de/wissen/index.php/Regelungstechnik
zu finden.

Erstellen Sie nun ein Softwareprojekt, das eine Regelschleife enthält, die die Motordrehzahl auf einem vorgegebenen, zur Laufzeit wählbaren Wert hält. Dafür sollte bereits ein PI-Regler ausreichen. Können Sie mit einem PID-Regler ein besseres Ergebnis erzielen?

Verwenden Sie das Echtzeitbetriebssystem MicroC/OS-II und teilen Sie die benötigten Funktionen sinnvoll in verschiedene Tasks auf. Zu trennen sind beispielsweise die Regelschleife, Ausgaben, Benutzereingaben, etc.

Einige Anmerkungen, die Ihnen vielleicht bei der Implementierung helfen können:

- Beginnen Sie mit einem P-Regler. Welchen Effekt können Sie beobachten?
- Wird der Lüfter mit der Saug-Seite flach auf den Tisch gelegt, so kann er höhere Drehzahlen leichter erreichen.
- Geben Sie die gemessene Drehzahl und den Duty-Cycle der PWM aus. So können Sie sofort feststellen, wenn die Störgröße (Belastung) zu groß ist und nicht mehr kompensiert werden kann.
- Stellen Sie sicher, dass die Regelschleife nicht zu schnell nacheinander durchlaufen wird
- Berücksichtigen Sie den Schleifenzyklus bei der Bestimmung der I- und D-Anteile

- Schließen Sie aus, dass unsinnige Werte des Tachosignals in die Regelung einfließen
- Begrenzen Sie die Fehlersumme (Wind-Up-Effekt)
- Benutzen Sie zur Eingabe der Regelparameter die Tasten auf dem Board
- Bei geringer Spannung am Lüfter gibt es Fehler im Tachosignal bzw. fällt es ganz aus, so dass Ihre Regelung instabil werden kann (also Vorsicht beim zu schnellen/starken Herunterregeln)
- Wählen Sie bei der Frequenz Ihrer PWM eher niedrigere Werte (ca. 1 kHz)
- Welche Auswirkungen hat das Ändern des Duty-Cycle in ihrer PWM-Komponente im laufenden Betrieb? Wie können Sie eventuell daraus resultierende Probleme vermeiden?
- Wie genau ist Ihre Regelung? Wie schnell erreicht der Lüfter seine Sollgeschwindigkeit aus dem Stillstand? Wie groß ist das Überschwingen dabei?

Bei der endgültigen Realisierung der *Lüfterregelung* soll dann die Soll-Drehzahl über die Taster (Interrupts verwenden) eingegeben werden und die Ist- und Soll-Werte für die Umdrehungen pro Minute auf dem LCD angezeigt werden.

Aufgabe 5 - Dokumentation

Die Umsetzung des Projektes *"Lüfterregelung"* soll in einem kurz gehaltenen Abschussbericht festgehalten werden. Dieser muss beantworten, warum eine Umsetzung des Projektes mittels eines Hardware/Software Designs sinnvoll ist, warum bestimmte Teile in Hardware und andere Teile in Software realisiert wurden, was die Funktionsweise von Treibern ist, wie Ihre Hardwarekomponenten mit Registern aussehen und wie Sie Ihre Treiber und Hardwarekomponente realisiert haben. Weiterhin soll der Einsatz eines Echtzeitbetriebssystems begründet werden sowie eine Erklärung von Tasks. Auch hier muss auf ihre Realisierung eingegangen werden und erklärt werden wie Sie die Aufgabe umgesetzt haben.