Praktikum 15. November 2010

Jörn Loviscach

Versionsstand: 12. November 2010, 18:10

In diesem Praktikum soll ein Fotowiderstand durch einen Modellbau-Servo automatisch ins optimale Licht gedreht werden – nun aber anders als im ersten Praktikum mit effizienten, sauber ausgelagerten Funktionen.

Aufgabe zu Hause als Vorbereitung: Wie müssen die .c-Datei und die .h-Datei für die zweite Aufgabe (siehe nächste Seite) aussehen? Schreiben Sie die auf oder programmieren Sie die ohne Servo.

Das C-Projekt setzt Ihre Ergebnissen aus dem ersten Praktikum fort. Es basiert weiterhin auf der kleinen Funktionsbibliothek, die aus servo01.c und servo01.h besteht.

Verbinden Sie das LaunchPad zu Beginn noch nicht mit dem PC. Schließen Sie erst den 1 kΩ-Widerstand, den Fotowiderstand und den Servo wie im ersten Praktikum an das LaunchPad an. Lassen Sie den Aufbau sicherheitshalber kontrollieren, bevor Sie ihn mit dem PC verbinden.

Programmieraufgaben:

1. Schreiben Sie in main.c eine Funktion mit dem Prototypen `int readAtPosition(int position)`, die den Servo auf die in Prozent von 0 bis 100 angegebene Position stellt, dann

   $$\left(200 + 2 \cdot \left| \textit{neue Position} - \textit{alte Position} \right|\right) \text{Millisekunden}$$

   wartet, um dem Servo Zeit zum Bewegen zu geben, und zum Schluss die Spannung misst und als Ergebnis zurückliefert. Den Absolutbetrag für `int` finden Sie als Funktion `abs` in `<stdlib.h>`. Schreiben Sie eine `main`-Funktion, die einige Aufrufe von `readAtPosition` testet.

2. Lagern Sie die Funktion `readAtPosition` in eine eigene .c- und eine eigene .h-Datei aus. Benutzen Sie Include-Guards in der .h-Datei.

3. Ergänzen Sie diese .c- und diese .h-Datei um eine Funktion mit dem Prototypen `int findBestPosition(int start, int end)`, die den Servo von der Position `start` bis zur Position `end` schwenkt und dann die Position

mit der größten Lichtintensität zurückliefert. Diese Funktion darf sich darauf verlassen, dass `start` nie größer ist als `end`. Sie greift ausschließlich mit der Funktion `readAtPosition` auf die Elektronik zu.

4. Erweitern Sie das Programm so, dass es einmal von 0 bis 100 schwenkt, um die aktuell beste Position zu suchen, und dann laufend in kleinen Bereichen um die bisher beste Position schwenkt, um Änderungen des Lichts zu berücksichtigen. Der Servo darf dabei nie auf Positionen unter 0 oder über 100 gestellt werden.

5. Erweitern Sie `findBestPosition` so, dass die Suche möglichst wenig Fahren benötigt. Steht der Servo derzeit zum Beispiel auf der Position 93 und ruft man `findBestPosition(13, 42)` auf, soll die Suche von 42 bis hinab zu 13 erfolgen, statt dass der Servo erst bis 13 hinunter und dann wieder bis 42 herauf fährt.

6. Je nach verbleibender Zeit erweitern Sie das Programm zum Beispiel so:

   - Es wird berücksichtigt, dass die Ausgangsposition des Servos unbekannt ist.
   - Die Leuchtdioden auf dem Board zeigen an, in welche Richtung der Servo gerade fährt.
   - Das Programm führt regelmäßig wieder einen vollen Schwenk durch, um keine Änderungen zu verpassen, die weit weg von der aktuellen Position passieren.
   - Die Wartezeit in `readAtPosition` ist optimiert, also gerade so lang wie nötig, um den Servo das angegebene Ziel erreichen zu lassen.