Projektbericht

„Sonnenverfolgung“

Von Niklas Puls

geb. am 17.07.2002

Projektkurs PK6P1 - „Microcontroller“

Einführung

Dieses Projekt dient der Verfolgung der Sonne, d.h. ein Rohr, welches von zwei Motoren horizontal und vertikal gedreht wird, zeigt genau auf die Sonne. Zur Erkennung, inwiefern das Rohr auf die Sonne zeigt, ist in dem Rohr ein Fotowiderstand, welcher je nach Helligkeit dem Arduino verschiedene Werte verschiedene Werte sendet. Zur manuellen Steuerung der Motoren ist ein Joystick angeschlossen.

Meine Motivation für dieses Projekt war der Code selber. Ich interessierte mich seit ich das Projekt habe zwar sehr für das Bauen. Der Code spornte mich aber noch zusätzlich an, da ich mein Können im Bereich des Programmierens beweisen wollte. Ich wollte auch natürlich testen, ob der Code funktioniert, weshalb ich umso mehr motiviert für das Bauen war. Für das Programmieren brauchte ich keine Motivation, da mich Programmieren am meisten interessiert und somit ich in diesem Bereich höchstmotiviert schon war.

Das Projekt

Funktionsweise

Wird der Arduino angeschaltet, so muss man den seriellen Monitor an dem Computer öffnen. Dann öffnet sich der Sperrbildschirm, bei dem man durch Eingabe der Zahl 1 den Arduino entsperrt und in das Hauptmenü kommt. In dem Hauptmenü angelangt wird einem die Liste an möglichen Eingaben angezeigt. Man kann fünf verschiedene Teile des Programmes ausführen. Der erste sperrt den Arduino wieder. Der zweite startet die manuelle Steuerung des Arduinos. Der dritte startet dagegen die Sonnenverfolgung. Bei dem vierten lässt sich umschalten, ob die Werte aus den analogen Eingängen (Sensoren) und dem digitalen Eingang (Schalter) in den seriellen Monitor gesendet werden sollen. Der letzte Teil ermöglicht dem Nutzer, die Motorgeschwindigkeit zu variieren. Führt man den ersten Teil aus und der Arduino wird gesperrt, dann sind alle anderen Funktionen außer dem Entsperren deaktiviert. Dies verhindert ungewollte Eingaben und bietet eine relativ niedrige Sicherheit, wobei dieses Projekt keine hohe Sicherheit erfordert. Startet man die manuelle Steuerung, so lässt sich das Rohr, in dem die Lichtwerte gemessen werden, steuern. Dieses Rohr besteht aus dem Mantel und zwei Deckeln. In dem oberen ist ein Loch hineingebohrt, in dem unteren Deckel ist der Fotowiderstand befestigt. Durch das Loch fällt ein kleiner Lichtstrahl auf den Fotowiderstand.

Dies ist relevant für die Sonnenverfolgung, für die manuelle Steuerung weniger. Es wird durchgängig der X- und der Y-Wert ausgelesen. Geht einer von den beiden Werten unter eine Grenze, so bewegt sich der jeweilige Motor in die Richtung. Wandern die Werte über eine zweite Grenze, so gehen auch dieses Mal die jeweiligen Motoren in die Richtung. Es wird also zwischen oben und rechts sowie unten und links differenziert. So lässt sich das Rohr zu der Sonne drehen. Die manuelle Steuerung lässt sich durch Betätigen des Schalters von dem Joystick verlassen. Man gelangt dann zu dem Hauptmenü zurück. Entscheidet man sich für die Sonnenverfolgung, dann wird als erstes der Fotowiderstand ausgelesen. Dann wird ein Schritt in eine Richtung gemacht. Ist in diese Richtung die Helligkeit höher, dann macht der Arduino nochmal einen Schritt in dieselbe Richtung, usw. Dies führt der Arduino bis die Helligkeit niedriger wird fort. Geschieht das, so wird ein Schritt zurück gemacht, damit der Arduino an der hellsten Stelle für diese Richtung ist. Dann wird geht der Arduino im Uhrzeigersinn eine Richtung weiter. War die vorherige Richtung nach oben, so geht sie nun nach rechts, dann nach unten, usw. Auch hier macht der Arduino dasselbe wie oben erläutert. Dies führt er bis zur Beendigung der Sonnenverfolgung durch. Da der Arduino somit immer um den hellsten Punkte kreist, verfolgt er die wandernde Sonne, da, wenn sie sich weiterbewegt, der Arduino sich wegen dieses Programmes in die jeweilige Richtung auch bewegt. Genauso wie bei der manuellen Steuerung beendet man die Sonnenverfolgung durch Drücken des Schalters von dem Joystick. Der vierte Teil des Programmes schaltet zwischen Werte anzeigen und Werte verstecken. Sollen Werte angezeigt werden, so werden diese während der manuellen Steuerung sowie der Sonnenverfolgung in den seriellen Monitor gesendet. Mit dem fünften Programmteil kann man die Motorgeschwindigkeit ändern. Die Werte liegen zwischen 1 und 10. Man ändert den Wert mithilfe des Joysticks. Bewegt man ihn nach oben, so wird der Wert um eins höher. Um den Wert nochmal zu erhöhen, muss man den Joystick zurück in die Anfangsposition bewegen. Genau dieselbe Mechanik ist auch bei Verringern des Wertes. Man muss den Joystick immer zurücksetzen, da sonst die Auswahl zwischen Werten schwierig wurde, da man den Joystick sehr kurz und schnell bewegen musste. Mit dieser Technik ist die Art, wie man den Joystick dreht, nicht relevant für die Änderung des Wertes. Um den neuen Wert zu bestätigen, muss man den Schalter von dem Joystick betätigen.

Das Programm

Da das Programm mit über 400 Zeilen Code zu lang für eine vollständige Erläuterung ist, wird die Funktionsweise anhand spezifischer, exemplarischer Ausschnitte erläutert. Auch werden die Teile des Codes erläutert, die für die Grundfunktionen (manuelle Steuerung und Sonnenverfolgung) elementar sind. Funktionen, mit denen sich verschiedenes anpassen lässt, werden nicht ausführlich erläutert.

1. Abschnitt: Die Deklarierungen

Die Deklarierung unterteilt sich in drei Abschnitte, die Libraries, die Variablen und die Motoren. Im ersten Abschnitt werden die Libraries deklariert. In diesem Fall wird die Library für die Schrittmotoren und für den EEPROM-Speicher deklariert. Der EEPROM-Speicher kann Variablen so speichern, dass sie nach einem Neustart denselben Wert behalten und nicht den von der Deklarierung. Im zweiten Abschnitt werden die Variablen deklariert. Diese sind global und können somit in dem gesamten Code abgerufen werden. In dem dritten Ausschnitt werden die Motoren deklariert.

1. Abschnitt: Die Mittelwertfunktion

Bei der Mittelwert-Funktion wird eine Schleife gestartet, welche 5000-mal durchläuft. Pro Durchlauf wird die Variable average mit einem Sensorwert addiert, der in dem Durchlauf ausgelesen wurde. Hat die Schleife aufgehört, so wird der Endwert durch 5000 geteilt und dadurch erhält man einen Mittelwert. Die jeweilige Variable, die die Funktion ausgelöst hat, nimmt dann diesen Mittelwert an.

1. Abschnitt: Der Switch-Case

Der hier gegebene Switch-Case entscheidet zwischen den Handlungen, den der Arduino in dem Programm machen kann. Ein Switch-Case führt den Fall aus, welcher den Wert der ausgelesenen Variable besitzt. Als Beispiel würde ein Switch-Case, wenn die Variable den Wert 3 hat, den dritten Fall ausführen. Ein Switch-Case geht aber nicht nur nach Zahlen. Genauso kann ein Fall auch ein Symbol sein und wird dann ausgeführt, wenn die Variable auch dieses Symbol als Wert hat. Die hier benutzte Variable „nextAction“ hat der Arduino vorher aus dem seriellen Monitor ausgelesen. Der erste Fall, also nextAction = 1, sperrt bzw. entsperrt den Arduino. Im gesperrten Modus können andere Funktionen nicht ausgeführt werden. Das soll ungewollte Eingaben verhindern. Der zweite Fall stellt die manuelle Steuerung per Joystick dar (s. Abschnitt 4). Der dritte Fall beinhaltet die Sonnenverfolgung (s. Abschnitt 5). Der vierte Fall schaltet zwischen Werte zeigen und verstecken. Während der manuellen Steuerung oder der Sonnenverfolgung werden dann entweder die Werte in den seriellen Monitor gesendet oder versteckt. Der fünfte Fall ermöglicht die Veränderung der Drehgeschwindigkeit der Motoren von 1 bis 10. Zusätzlich gibt es den Fall „default“. Dieser wird immer dann ausgeführt, wenn keiner der Fälle ausgeführt wurde. In diesem Fall wird in den seriellen Monitor geschrieben, dass der Nutzer eine falsche Eingabe gemacht hat.

1. Abschnitt: Manuelle Steuerung per Joystick

Der Prozess beginnt damit, dass getestet wird, ob der Arduino entsperrt wurde. Wenn ja, kann die manuelle Steuerung starten, sonst wird eine Fehlermeldung in den seriellen Monitor gesendet. Ist der Arduino entsperrt, wird der Zeitgeber für den Timeout auf 0 gesetzt, da nun eine Eingabe erfolgt hatte. Danach wird die Variable für die Schleife repeat1 auf true gesetzt, damit sich die Steuerung der Motoren auch wiederholt. Als nächstes macht der Motorx einen Schritt nach unten und danach wieder hoch. Dies dient dazu, damit die Werte des Fotowiderstandes in dem seriellen Monitor korrekt angegeben werden. Grund hierfür konnte ich auch nach Recherche nicht finden. Dies ist die einzige, mir bekannte, Lösung. Als nächstes wird der serielle Monitor vorbereitet (clearScreen() ist eine Funktion, bei der 100mal Leerzeile gesendet wird. Dies „löscht“ den momentanen Inhalt des Bildschirms, sodass keine Unordnung in dem Monitor oder ein Missverständnis aufkommt), und die Schleife für die manuelle Steuerung beginnt. In dieser Schleife werden als erstes pro Durchlauf, wenn giveValues auf true ist, die Werte von dem Fotowiderstand und der Y- und X-Achse des Joysticks in den Monitor gesendet. Bei dem Schalter des Joysticks wird hinter dem Wert noch die Bedeutung geschrieben. Als nächstes wird der Sensor für die y-Richtung des Joysticks ausgelesen. Es wird zunächst abgefragt, ob der y-Wert größer 570 oder kleiner 470 ist. Dies entscheidet, ob der Arduino überhaupt die nächsten Abfragen durchführen muss. Das macht den Arduino ein wenig schneller, aber dieser Unterschied ist relevant Dann wird abgefragt, ob der Wert kleiner 470 ist. Wenn das zutrifft, macht der Motorx „steps“-Schritte in positive y-Richtung. Trifft dies nicht zu, wird abgefragt, ob der Sensorwert größer 570 ist. Auch hier macht der Motorx bei Zutreffen der Bedingung „steps“-Schritte, bloß diesmal in negative y-Richtung. Genau dasselbe geschieht danach mit dem Sensor des Joysticks für Bewegung in X-Richtung und dem Motory. Der Bereich zwischen 470 bis 570 (y-Achse) und 450 bis 550 (X-Achse) sind wegen der Genauigkeit der Sensoren gewählt. Obwohl bei Ruhelage des Joysticks die Werte nur selten schwanken, gibt es dieses Schwanken. Würde es keinen Bereich zwischen den beiden Werten geben (Bereich von470 bis 570 bzw. 450 bis 550), so würden manchmal die Motoren sich bewegen, obwohl sich der Joystick nicht bewegte. Um dies zu verhindern, wählte ich diese Grenzen, damit solches ungewollte Bewegen verhindert wird. Als letztes wird der Schalterwert ausgelesen. Ist dieser kleiner 1, wurde der Schalter betätigt. Der Wert kann zwischen 0 und 1 wechseln. Wurde der Schalter gedrückt, werden ein letztes Mal die Werte gesendet (wenn giveValues auf true ist). Dann wird die Variable der Schleife auf false gesetzt, sodass die Schleife nicht fortgeführt wird. Dann wird nextAction auf 0 gesetzt, damit nicht erneut dieser Case des switch-cases ausgeführt wird. Dann wird geschrieben, dass man zum Menü zurückkehrt. Nach zwei Sekunden wird der Bildschirm gelöscht und das Menü erscheint.

1. Abschnitt: directs(dir)-Funktion

Diese Funktion ist elementar für die Sonnenverfolgung (Abschnitt 6). Diese Funktion beinhaltet einen Switch-Case, der je nach Wert der Variable dir einen Motor bewegt. Es gibt vier Fälle, davon bestimmt jeder eine Richtung.

1. Abschnitt: Sonnenverfolgung

Auch dieser Programmabschnitt beginnt wie der von der manuellen Steuerung. Zunächst wird getestet, ob der Arduino entsperrt ist, wenn ja, wird der Code für die Sonnenverfolgung ausgeführt, sonst wird der Nutzer über eine falsche Eingabe benachrichtigt. Wird der Code ausgeführt, so wird zunächst die Variable repeat2 für die Sonnenverfolgung auf true gesetzt, damit sich der Code und somit die Verfolgung wiederholt. Auch wird wie bei dem vierten Abschnitt der Zeitgeber für den Timeout auf 0 gesetzt und der Bildschirm gelöscht. Bei Beginn eines Durchlaufs der Schleife wird die Funktion directs() ausgeführt (erklärt in Abschnitt 5). Danach wird der Wert des Fotowiderstandes ausgelesen mithilfe der light()-Funktion. Als nächstes wird abgefragt, ob der neue, gerade ausgelesene Wert (valueNext), größer ist als der alte Wert (valueMid). Der neue Wert valueNext ist hierbei ein „steps“-Schritte von der Mitte entfernter Punkt. Der alte Wert valueMid ist der von der momentanen Mitte. Ist also der neue Wert größer als der alte, so wird valueNext = valueMid. Der neue Wert ist nun der alte Wert und liegt in der neuen Mitte. Hiernach wird noch getestet, ob Werte angezeigt werden sollen. Wenn ja, wird der Lichtwert eingegeben. War der neue Wert größer, wird zu Beginn der Zeile ein „^“ eingegeben. War der neue Wert kleiner oder gleich groß, dann wird am Anfang ein „v“ stehen. Diese Symbole sollen Pfeile darstellen und anzeigen, ob der Wert gestiegen oder gefallen bzw. gleich geblieben ist. Trifft die Bedingung, dass der neue Wert größer ist als der alte, dann wird directs() für den Wert von dir ausgeführt. In der nächsten Zeile ändert sich der Wert dann. Und zwar zu dem, der aus der Rechnung entsteht. Hierbei wird der momentane dir-Wert mit 2 addiert und dann durch 4 geteilt. Dabei wird aber mit Rest gerechnet. dir nimmt dann diesen Restwert an. Hiernach wird wieder getestet, ob Werte gesendet werden sollen, und wenn ja wird der Wert des Fotowiderstandes gesendet. Da hier der neue Wert kleiner war als der alte, wird anstelle eines „^“ ein „v“ gesendet. Als letztes wird abgefragt, ob der Schalter des Joysticks betätigt wurde. Wenn ja, wird die Schleife beendet, der Bildschirm gelöscht und man kehrt zum Menü zurück. Der Schalter des Joysticks dient in meinem Programm hauptsächlich zur Beendigung entweder der manuellen Steuerung oder der Sonnenverfolgung.

Entstehung des Projektes

|  |  |  |
| --- | --- | --- |
| Nr. | Phase | Arbeitszeit |
| 1 | Planung (Erstellen Teileliste) | 3 h |
| 2 | Erste Programmskizzen | 5 h |
| 3 | Programmieren (Erste Version) [Grundversion] | 8 h |
| 4 | Programmieren (Zweite Version) [Grundversion] | 6 h |
| 5 | Konstruieren | 7 h |
| 6 | Programmieren (Letzte Version) [Grundversion] | 6 h |
| 7 | Erstes Debugging | 4 h |
| 8 | Optimierung des Programms | 3 h |
| 9 | Letztes Debugging | 2 h |
| 10 | **Projektbericht** | **9 h+ [Unfertig]** |
| 11 | Insgesamt | 52 h |

Verbesserungsmöglichkeiten und wirtschaftliche Umsetzbarkeit

Dieses Projekt lässt sich in vielerlei Hinsicht verbessern. Zum einen können natürlich präzisere Sensoren und Motoren genutzt werden, um die Genauigkeit zu erhöhen. Auch könnte man einen besseren Microcontroller. Damit ist gemeint, dass der Microcontroller bei selber oder sogar besserer Leistung deutlich kleiner ist und somit das Projekt kompakter wird. In der Wirtschaft würde sich das Projekt für Solarpanels eignen. Durch dieses Projekt würde ein hieran befestigtes Solarpanel immer zur Sonne zeigen und somit maximal effizient sein. Die Technik dieses Projektes würde, bei idealen Bedingungen, sich als Teleskopsteuerung eignen. Jedoch müsste hierfür der Himmel völlig klar sein, es dürften sich keine anderen Lichtquellen im Umkreis befinden und die Motoren und der Sensor müssten sehr viel genauer sein. Die Genauigkeit lässt sich auch durch ein kleineres Loch in dem Rohr erhöhen.

Anhang

Aufzählung aller Teile und die Gesamtkosten:

|  |  |  |
| --- | --- | --- |
| Anzahl | Name | Preis |
| 2 | Schrittmotor (Teil eines 5er-Sets) |  |
| 1 | Fotowiderstand  (Teil eines Sets von 5 versch. Fotowiderständen in 100er-Packungen) |  |
| 1 | Joystick |  |
| 1 | Plastikplatte (20cm x 20cm x 0,5cm) |  |
| 1 | Kabelset (m-w, m-m, w-w, jeweils in 50er-Packungen) |  |
| 1 | Sekundenkleber |  |

Referenz von Programmteilen:

* Funduino-Tutorials (https://funduino.de)