4. Sortiermaschine

Einführungspraktikum

Roboterprogrammierung

Fachhochschule Südwestfalen
University of Applied Sciences

Autor:
Fachhochschule Südwestfalen
Fachbereich Elektrische Energietechnik
Campus Soest

Hinweise zur Bearbeitung

Alle Aufgaben innerhalb dieses Praktikums beinhalten verschiedene Aspekte der Arbeit mit den LEGO Mindstorms Komponenten und der Programmiersoftware RobotC. Um eine möglichst intuitive Bearbeitung der Aufgaben zu ermöglichen werden innerhalb der Dokumentation die folgenden Symbole und Notationen benutzt.

Symbol / Notation	Bedeutung
1.1.1	Das Symbol der Maus deutet auf Bereiche innerhalb des Praktikums hin, in denen bestimmte Bereiche selbstständig gelöst werden sollen. Die nebenstehende Zahl ordnet jeder Lektion eine eindeutige Nummer zu um die Orientierung zu erleichtern.
Menüpunkt 1	Textbausteine, die fett gedruckt und in alternativer Schriftart abgedruckt sind sollen auf Menüpunkte oder Befehle innerhalb von RobotC hinweisen.
Menüpunkt 1 → Befehl A	Der Pfeil zwischen zwei fett gedruckten Texten deutet auf eine Sequenz von Menüpunkten oder Befehlen hin. Hier soll also der Befehl A unter dem Menüpunkt 1 ausgeführt werden.
projekt1.c projekt2.c	Die Diskette deutet auf Dateizugriffe hin. An diesen Stellen werden entweder Dateien gespeichert (Schreibende Hand) oder geöffnet (Offene Hand). Der jeweilige Dateiname ist neben dem Diskettensymbol angeordnet. Hier soll also das aktuelle Projekt unter dem Dateinamen projekt1.c gespeichert werden und ein anderes Projekt mit dem Dateinamen projekt2.c geladen werden.
Abbildung 1	Die hellblaue Schrift deutet auf Abbildungen, Tabellen oder andere im Text eingebettete Elemente hin.
	Texte oder Grafiken in einer solchen geschweiften Zelle deuten daraufhin, dass hier nur Ausschnitte einer größeren Ansicht dargestellt sind.
1 Task main() 2 { 3 Motor[port3] = 0; 4 }	Texte die in der nebenstehenden Art und Weise formatiert sind deuten immer auf Code-Abschnitte hin. Hierbei wird zur besseren Orientierung immer die Zeilennummer angegebenen.

Einführung

In den letzten beiden Praktika wurde Ihnen ein erster Einblick in die Programmentwicklung mit RobotC gegeben und auch die Grundlagen der Programmierung in C sind Ihnen nun vertraut. Aufbauend auf diesen Grundlagen soll in den nun verbleibenden Terminen jeweils ein kleines Projekt umgesetzt werden. Hierbei sind in der Regel einige Teile bereits vorgegeben, so dass es innerhalb der vorgegeben Zeit möglich ist die komplette Funktion des jeweiligen Projektes zu realisieren. Im Gegensatz zu den ersten beiden Versuchen, die sehr instruktiv waren, steht ab jetzt die eigenverantwortliche Arbeit im Vordergrund.

Ziel dieses Praktikums ist dabei die eigenständige Entwicklung eines Sortierautomaten, welcher in der Lage ist LEGO-Bauteile unterschiedlicher Farbe automatisch in zwei unterschiedliche Behälter zu sortieren. Hierzu ist zunächst ein passender mechanischer Aufbau mit Hilfe der LEGO-Bauelemente zu entwickeln, wobei ein Grundgerüst an Ihrem Arbeitsplatz bereist zur Verfügung stehen sollte.
Die Programmierung des Sortierautomaten wird dann mit Hilfe von RobotC realisiert, wobei zunächst eine neue Methode zur vorherigen Programmbeschreibung eingeführt wird: das Flussdiagram. Die folgenden Kapitel führen Sie durch die notwendigen Schritte bis hin zu einem Funktionsfähigen Modell.

Aufgabe 4.1 Mechanischer Aufbau

Als erste Aufgabe ist der mechanische Aufbau des Sortierautomaten zu realisieren, wobei das mechanische Grundgerüst an Ihrem Arbeitsplatz als Grundlage dienen soll. Es sind von Ihnen also „nur“ noch einige Erweiterungen an dem Model vorzunehmen. Hierzu stehen Ihnen alle Bauteile aus den LEGO Mindstorms Basis- und Ergänzungssets zur Verfügung.
Die folgende Abbildung zeigt ein einfaches Schema der Anlage.

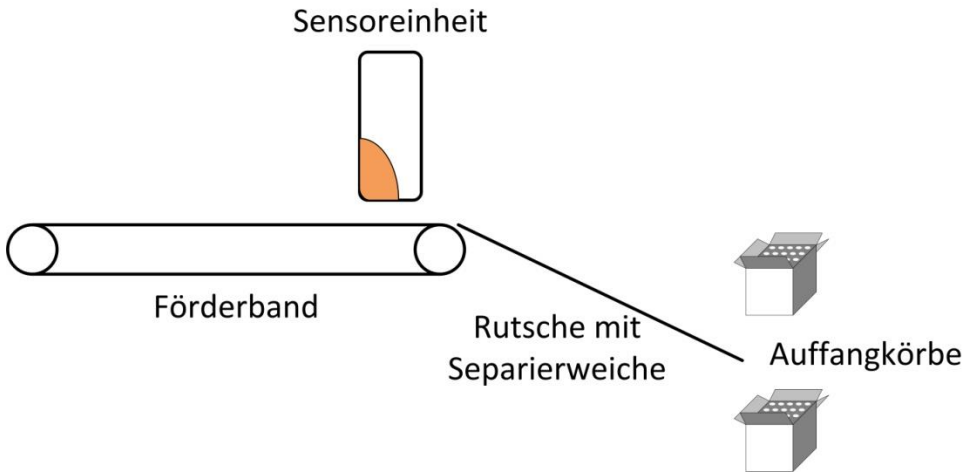

Abbildung 1 – Grundlegender Aufbau des Sortierautomaten

Es soll nun die Möglichkeit bestehen LEGO-Teile unterschiedlicher Färbung (Grau und Schwarz) zu unterscheiden. Hierzu steht als erster Bestandteil eine Förderbandkonstruktion mit Rutsche zur Verfügung, welche die Möglichkeit zur manuellen Aufgabe von LEGO-Teilen per Hand vorgibt.
Auf Basis der Auswertung mit Hilfe der Sensoreinheit soll dann über eine Separierweiche das jeweilige Bauteil in einen der beiden Auffangkörbe geleitet werden. Diese Elemente müssen Ihrem Modell noch hinzugefügt werden.

4.1.1 Aufbau der mechanischen Plattform für die Sortieranlage

1. Erstellen Sie eine einfache Konstruktion für die Separierweiche, welche es ermöglichen soll Teile auf der Rutsche in zwei unterschiedliche Richtungen umleiten zu können. Hierbei ist entscheidend, dass sich die Position der Weiche mit Hilfe eines Motors bequem verstellen lässt. Seien Sie bei der Umsetzung ruhig kreativ.
2. Erstellen Sie eine Vorrichtung welche die Aufnahme von LEGO Mindstorms-Sensoreinheiten an geeigneter Stelle ermöglicht.
3. Bringen Sie zwei Auffangkörbe an Ihr Modell an, so dass die jeweiligen Bauteile in einem der beiden Auffangkörbe aufgefangen werden.
4. Testen Sie den mechanischen Aufbau in dem Sie die Motoren ansteuern und Teile aufgeben.

Aufgabe 4.2 ■ Sensorik

Nachdem in Aufgabe 4.1 der mechanische Aufbau realisiert wurde muss nun über eine geeignete Sensoreinheit nachgedacht werden, damit das endgültige Ziel einer Sortieranlage erreicht werden kann.

4.2.1 Auswahl einer geeigneten Sensoreinheit

Überlegen Sie welcher zur Verfügung stehende Sensor Ihnen für diese Aufgabe am geeignetsten erscheint. Ziehen Sie eventuell die Beschreibung der einzelnen Sensoren aus der Einführungsdokumentation zu Hilfe. Welche Sensoren wurden in den beiden vorangegangen Terminen bereits eingesetzt und welche könnten das aktuelle Problem lösen?

Es empfiehlt sich immer vor der Realisierung einer Automatisierungstechnischen Aufgabe das Verhalten der genutzten Sensoren zu testen, damit die Ergebnisse direkt in den Entwicklungsprozess des Programmes einfließen können. In diesem Fall ist also von besonderem Interesse wie sich der ausgewählte Sensor bei verschiedenen Farben verhält und welchen Einfluss die Größe des Bauteils (also auch der Abstand zwischen Bauteil und Sensor) auf den Messwert hat. Hierzu ist es ratsam eine kleine Testanwendung zu entwickeln, welche die entsprechenden Sensordaten auf dem Display ausgibt.

4.2.2	Test der Sensoreinheit

1. Schreiben Sie ein Programm mit Hilfe von RobotC welches Ihnen ermöglicht den aktuellen Sensorwert auf dem Display angezeigt zu bekommen. Außerdem soll der Antriebsmotor des Förderbandes in Dauerbetrieb sein.
2. Notieren Sie die unterschiedlichen Sensorwerte für LEGO-Bauteile unterschiedlicher Farbe und den Wert der bei keinem aufgelegten Bauteil gemessen wird.
3. Testen Sie in wie weit sich die Sensorwerte bei Bauteilen unterschiedlicher Größer aber gleicher Farbgebung unterscheiden. Ist die Beurteilung unterschiedlicher Bauteile als kritisch anzusehen?
4. Stellen Sie abschließend fest ob und wie eine robuste Unterscheidung von verschiedenen Bauteilen mit Hilfe der gewählten Sensoreinheit möglich ist.

Aufgabe 4.3 ▪ Erstellung des Programms

Nun steht sowohl der mechanische Aufbau als auch die notwendige Sensorik zur Verfügung. Es kann also mit der Erstellung des eigentlichen Programmes begonnen werden. Bei komplexeren Programmieraufgaben empfiehlt es sich vor der eigentlichen Erstellung des Programmcodes ein Schema des Programmflusses zu erstellen, welches dann später als Grundlage für den Code genutzt wird. Sie haben während des letzten Praktikums bereits die Beschreibung von Algorithmen mit Hilfe von Pseudo-Code kennengelernt. Hierbei handelt es sich um eine sehr einfache textuelle Form der Programmbeschreibung. In vielen Fällen kann es ratsam sein den Programmfluss graphisch darzustellen. Hierzu gibt es zahlreiche genormte Notationen für die Modellierung mit Hilfe von Flussdiagrammen.
Die folgende Abbildung zeigt ein simples Beispiel welches die notwendigen Elemente eines Flussdiagrams einführt. Hierbei wird ein sehr einfaches Beispiel für das automatische Abschalten eines fahrbaren Roboters mit Hilfe eines Flussdiagrammes visualisiert.

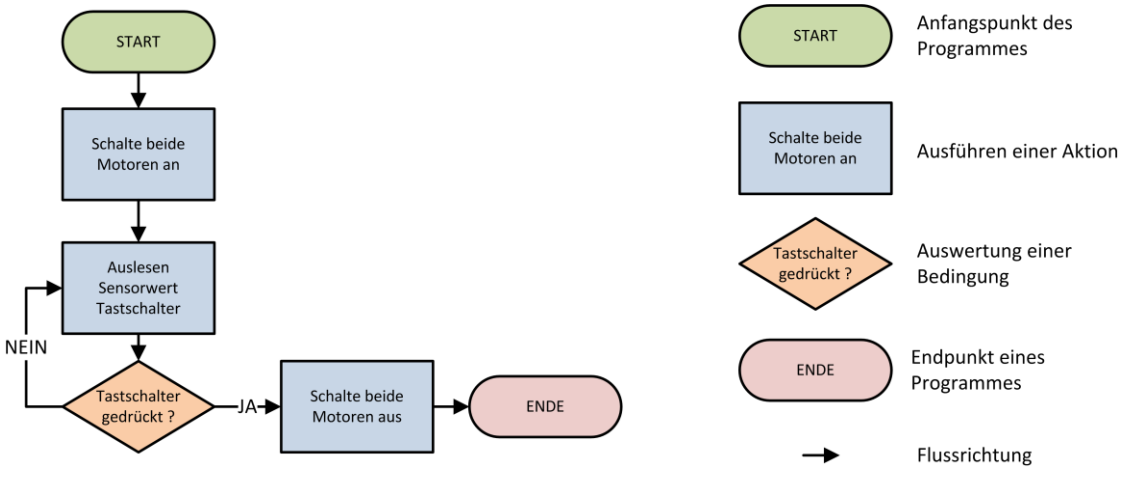

Abbildung 2 – Beispiel für die Beschreibung eines Programmes mit Hilfe eines Flussdiagrams und die notwendigen Elemente

Zunächst wird der Startpunkt des Programms markiert, dieser folgt dann das Einschalten beider Antriebsmotoren des fahrbaren Roboters, so dass dieser Geradeaus fährt. Dieses wird mit einem Aktionsblock (Blau) dargestellt. Danach wird der Sensorwert eines Tastsensors abgefragt. In diesem Beispiel soll der Roboter wenn der Tastsensor angesprochen hat, der Roboter also auf ein Hindernis gestoßen ist, automatisch anhalten. Dies wird nun über einen Bedingungsblock (Orange) signalisiert. Ist der Tastschalter gedrückt (rechter Pfad) werden die beiden Motoren ausgeschaltet und das Programm endet mit dem Endpunkt (Rot). Falls der Tastschalter nicht gedrückt ist wird der Sensorwert erneut abgefragt. Diese Programmpunkte werden solange wiederholt, bis der Sensor auf ein Hindernis gestoßen ist, das heißt diese Rückkopplung lässt auf die Benutzung einer `while`-Schleife innerhalb des Programmes schließen.

4.3.1 Erstellen eines Flussdiagrammes für den Sortierautomat

1. Erstellen Sie unter Zuhilfenahme des Beispiels aus Abbildung 2 ein Flussdiagramm des Sortierautomaten. Dieses soll nachher die Grundlage für die Struktur Ihres Programmes darstellen. Hierbei soll der Einfachheit halber verabredet werden, dass jeweils nur ein Teil nach dem anderen auf das Förderband aufgelegt wird.

Auf Basis des Flussdiagramms lässt sich nun relativ einfach das C-Programm mit Hilfe von RobotC erstellen, hierbei müssen nämlich die einzelnen Elemente „nur“ in entsprechenden Code „übersetzt“ werden, ähnlich der Prozedur bei Benutzung von Pseudo-Code. Hierbei stellen Aktionsblöcke (Blau) in der Regel die Ausführung von Funktionen dar (z.B. `motor[port2]=100;`). Bedingungsblöcke werden meist durch `if-else`-Konstrukte im C-Code dargestellt. Die Rückkopplung am „NEIN“-Ausgang der Bedingung wird durch eine `while`-Schleife realisiert die dieser Teil des Programms solange wiederholt bis der Tastschalter ausgelöst wird

Der folgende Codeausschnitt zeigt in beispielhafter Weise den C-Code für das Flussdiagram aus Abbildung 2, wobei die Kommentare verdeutlichen welche Codezeile welchem Block zuzuordnen ist.

```
1  Task main()
2  {
3     Motor[motorA] = 100;              // Einschalten des ersten Motors (AKTIONSBLOCK 1)
4     Motor[motorB] = 100;              // Einschalten des zweiten Motors (AKTIONSBLOCK 1)
5     while (SensorValue(S1) != 1);     // Abfrage des Sensorwertes und Bedingung
6                                       // (AKTIONSBLOCK 2 und BEDINGUNGSBLOCK 1)
7     Motor[motorA] = 0;                // Abschalten des ersten Motors (AKTIONSBLOCK 3)
8     Motor[motorB] = 0;                // Abschalten des zweiten Motors (AKTIONSBLOCK 3)
9  }
```

Abbildung 3 – Beispiel für die „Übersetzung“ eines Flussdiagrammes in C-Code

4.3.2 Erstellen des RobotC-Programmes für den Sortierautomat

1. Erstellen Sie unter Zuhilfenahme Ihres Flussdiagrammes den C-Code in der Programmierumgebung RobotC. Beginnen Sie hierbei zunächst mit der Grundlegenden Struktur und füllen Sie dann die unterschiedlichen Abschnitte des Codes mit den notwendigen Aktionen. Achten Sie dabei auch auf die Eingabe sinnvoller Kommentare.
2. Testen Sie Ihr erstelltes Programm für unterschiedliche Bauteile.

3. Wo liegen eventuelle Probleme der Implementierung?
4. Wie könnte der Sortierautomat erweitert werden, so dass auch eine Sortierung von Bauteilen von mehreren unterschiedlichen Farben realisiert werden könnte?

Sortierautomat.c	Speichern Sie Ihr Programm

4.3.1 Implementierung von Zusatzfunktionen

Nachdem nun die Grundfunktionalität gegeben ist, soll die Anlage noch durch einige Zusatzfunktionen erweitert werden. Hierzu zählt als erster Punkt die Inbetriebnahme einer Anfahrwarnung, wie sie bei großen Anlagen üblich ist.
Hierzu wird vor dem Start des Förderbandes eine akustische und optische Warnung ausgegeben um allen Mitarbeitern zu signalisieren, dass die Anlage nun anfahren wird. Somit kann jeder Mitarbeiter eventuell Gefahrbereiche verlassen.
Zur Simulation dieser Anfahrwarnung soll sowohl eine akustische Meldung mit dem NXT-Brick erzeugt, als auch eine optische Meldung mit Hilfe einer Lampe generiert werden. Hierzu ist es zunächst notwendig die C-Funktion die zum Abspielen eines Sounds notwendig ist einzuführen. Hierzu zeigt die folgende Abbildung ein Beispiel für die Anwendung der Funktion `PlaySound` aus der Sound-Bibliothek von RobotC.

```
1  Task main()
2  {
3       PlaySound(soundBeepBeep);       // Abspielen eines vordefinierten Sounds auf dem
4  }                                    // NXT-Brick. Es stehen folgende Sounds zur Ver-
5                                       // fügung:
6                                       // - soundBeepBeep
7                                       // - soundBlip
8                                       // - soundDownwardTones
9                                       // - soundException
10                                      // - soundFastUpwardTones
11                                      // - soundLast
12                                      // - soundLowBuzz
13                                      // - soundLowBuzzShort
14                                      // - soundShortBlip
15                                      // - soundUpwardTones
```

Abbildung 4 – Abspielen eines Sounds in RobotC

Das Ansprechen einer Lampe ist in RobotC denkbar einfach: Da die Lampe an einem der Ausgangsports (A-C) genutzt wird, kann die Lampe wie ein Motor behandelt werden. Die Zuweisung eines Geschwindigkeitswertes von 100 (`motor[motorC] = 100;`) würde zum Beispiel bedeuten, dass die Lampe mit voller Leistung leuchtet. Dementsprechend kann man die Lampe durch Zuweisung einer Geschwindigkeit von 0 ausschalten.

4.3.3	Implementierung der Anfahrwarnung

1. Fügen Sie in Ihr Programm eine Anfahrwarnung ein. Hierbei soll ein Alarmton ausgegeben werden und die Lampe leuchten bevor die Anlage anfährt.
2. Als letzter Schritt soll nun die Anfahrwarnung fünfmal wiederholt werden, bevor das Förderband anläuft. Wie könnten Sie dies ganz einfach realisieren?
3. Um bestimmte Code-Abschnitte innerhalb eines Programmes zu wiederholen wurden in den vorherigen Übungen immer `while`-Schleifen eingesetzt. Hierbei werden bestimmte Codefragmente so oft wiederholt bis eine bestimmte Bedingung wahr oder unwahr wird. Möchte man hingegen bestimmte Teile eines Programmes zum Beispiel genau fünfmal wiederholen ist dies natürlich auch mit einer `while`-Schleife möglich. Wie?
4. Eine elegantere Form für eine solche Schleife die eine definiert Anzahl von Iterationen ausführt steht in RobotC die sogennante `for`-Schleife zur Verfügung. Die folgende Abbildung zeigt ein Beispiel für eine solche Schleife in C.

```
1   Task main()
2   {
3       int i;
4
5       for (i=0; i<5; i++)
6       {
7
8           nxtDisplayTextLine(1, "Iteration: %i", i);
9
10      }
11  }
```

Deklaration einer Zählvariablen

`for`-Schleife, die genau 5 mal ausgeführt wird

Darstellung der Zählvariablen auf dem Display des NXT-Bricks

Abbildung 5 – Beispiel für die Benutzung einer for-Schleife in C

Die Syntax des C-Codes ist dabei folgendermaßen zu verstehen:

- Die Zählvariable `i` startet bei dem Wert 0 (`i=0`).
- Vor jedem Schleifendurchlauf wird überprüft ob `i` kleiner als 5 ist und nur dann wird der Schleifenrumpf ausgeführt. (`i<5`)
- Nach jedem Schleifendurchlauf wird `i` um eins erhöht (`i++`).
- Das heißt also für das vorliegende Beispiel, dass auf dem NXT-Display nacheinander `0,1,2,3,4` angezeigt wird.

5. Erweitern Sie Ihre Anfahrwarnung nun in der Form, dass der Warnton fünfmal hintereinander ertönt und die Lampe fünfmal aufblinkt, bevor die Anlage anfährt.

Sortierautomat.c	Speichern Sie Ihr Programm