Projektarbeit Embedded C
Sortieranlage

Sommersemester 2025

Oliver Groß & Victor Kowlewski

July 1, 2025

Inhaltsverzeichnis

1 Lastenheft/Projektbeschreibung **3**
1.1 Projektbeschreibung 3

2 Anforderungsanalyse **4**
2.1 Anforderungen 4

3 Projektdurchführung **5**
3.1 Arbeitsweise 5

4 Systementwurf **6**
4.1 Softwarearchitektur 6
4.1.1 Softwarearchitektur und Paketstruktur 6
4.1.2 ADC-Komponenten 6
4.1.3 Digital Komponenten 6
4.1.4 LCD Display 7
4.1.5 Zustandsautomat 7
4.2 Statemachine 9

5 Test und Veri kation **10**

6 Fazit **12**

1 Lastenheft/Projektbeschreibung

1.1 Projektbeschreibung

Im Rahmen des Moduls "Embedded C++" wurde die Sortieranlage als semesterbegleitendes Projekt ausgewählt. Die Entscheidung fiel aufgrund der interaktiven Funktionalität der Anlage.

Die Aufgabe war es, eine Steuerung für eine Fischertechnik-Sortieranlage zu entwickeln. Die Sortieranlage besteht aus einigen Komponenten, die zur Vervollständigung der Aufgaben verwendet werden. Zu den Komponenten zählen ein Motor, ein Taster zur präzisen Steuerung des Förderbands, ein Kompressor, ein Drucksensor und Ventile, die gemeinsam die Bewegung und Verarbeitung der Sortiermaschinenteile übernehmen, während Sensoren die Werkstücke an bestimmten Positionen erfassen und vordefinierte Funktionen auslösen.

Die Sortieranlage hat die Aufgabe, Holzscheiben nach Farben zu sortieren und zu bearbeiten. Die Steuerung erfolgte dabei über ein STM32L100-Experimentierboard.

Das Ziel des Projekts ist die Entwicklung einer zuverlässigen, echtzeitfähigen Steuerungssoftware, die automatische Sortierung und Verarbeitung von Holzscheiben ermöglicht. Die Sortiermaschine soll anhand der Farbe eines Werkstücks entscheiden, ob dieses zur Verarbeitung oder zum Schlechtteilauswurf geleitet werden soll.

Figure : Aktivitätsdiagram zur Funktionsweise der Sortieranlage

2 Anforderungsanalyse

2.1 Anforderungen

A1: Nachdem das System gestartet wurde, erfolgt zunächst keine Aktion (System wartet auf Tastendruck).

A2: Es sind folgende Bedienelemente vorhanden:

- Start-/Stop Taste
- Clean Taste

A3: Die Verarbeitung wird nur gestartet, wenn ein Werkstück im Eingangsstapel liegt und die Start-/Stop taste betätigt wurde.

A4: Läuft die Verarbeitung und der Start-/Stop Knopf wird betätigt, dann wird die Verarbeitung pausiert, bis zum erneuten betätigen.

A5: Wenn keine Verarbeitung stattfindet und die Clean-Taste gedrückt wird, startet das Förderband und der Auswurf wird betätigt.

A6: Sobald sich kein Werkstück mehr in der Maschine befindet, wird die Verarbeitung gestoppt.

A7: Der Kompressor wird nur eingeschaltet, wenn nicht genügend Druck vorhanden ist.
Abweichung zum Lastenheft

A8: Es gibt ein LCD-Display, welches folgende Informationen darstellt:

- Zustand der Sortiermaschine
- Status vom aktuellen Zustand Abweichung zum Lastenheft
- Der aktuelle Luftdruck der Pumpe
- Die Anzahl der verarbeiteten Werkstücke

3 Projektdurchführung

3.1 Arbeitsweise

Zum Beginn der Projektarbeit wurde zunächst das Mapping der GPIO-Pins definiert. Für die Entwicklung wurde ein Bottom-Up-Design gewählt, das heißt, die Implementierung begann mit kleinen, unabhängigen Softwaremodulen, beispielsweise der Lichtschranke. Nach der Erstellung einzelner Komponenten erfolgte jeweils eine umfangreiche Testphase. Dabei wurden erkannte Fehler behoben und das Verhalten der Komponenten so angepasst, dass die spezifizierten Anforderungen erfüllt wurden.

Im Anschluss konnten die getesteten Module schrittweise in das Gesamtsystem integriert werden. Parallel dazu wurde ein erstes UML-Klassendiagramm modelliert, um die Softwarestruktur visuell darzustellen. Nachdem alle Komponenten der Sortiermaschine erfolgreich angesteuert werden konnten und ein lau ähiger Prototyp vorlag, wurde eine di erenzierte Ablaufsteuerung in Form einer Zustandsmaschine entworfen. Diese ermöglichte eine modulare Erweiterbarkeit sowie eine verbesserte Wartbarkeit der Softwarearchitektur.

Auch die Funktionalität des Displays wurde abstrahiert, um die Kopplung an andere Systemteile zu reduzieren und die Flexibilität zu erhöhen. Gegen Ende der Projektlaufzeit wurden feste Parameterwerte ermittelt, die für einen zuverlässigen und stabilen Betrieb der Sortiermaschine erforderlich sind.

4 Systementwurf

4.1 Softwarearchitektur

4.1.1 Softwarearchitektur und Paketstruktur

Die Architektur der Steuerungssoftware wurde modular entworfen, mit dem Ziel, eine einfach erweiterbare und wartbare Software zu entwerfen. Hierzu gibt es zwei Hauptpakete. Das Paket "Components" beinhaltet die Klassen zur Steuerung und Verwaltung der Hardware-Komponenten. Außerdem gibt es das Paket für die Statemachine. Die Komponenten sind zusätzlich in Digital, ADC und Display unterteilt.

4.1.2 ADC-Komponenten

Im Paket ADC befinden sich die Klassen zur Verarbeitung analoger Signale über den integrierten Analog-Digital-Wandler (ADC) des Mikrocontrollers. Die Klasse Farbsensor nutzt den ADC, um analoge Spannungswerte zur Farberkennung auszuwerten. Anhand dieser Werte wird die Farbe des Werkstücks (weiß oder blau) bestimmt, indem der Messwert mit zuvor definierten Schwellwerten verglichen wird.

Die Klasse "MotorPumpe" dient der Steuerung des Luftdrucks in der Sortieranlage. Über einen angeschlossenen Drucksensor wird ein analoges Signal erfasst, aus dem mithilfe des ADC der aktuelle Luftdruck berechnet wird. Abhängig von diesem Wert entscheidet die Komponente, ob der Kompressormotor zur Druckerzeugung aktiviert werden muss.

4.1.3 Digital Komponenten

Die Klassen, welche im Paket "Digital" definiert sind, beschreiben Elemente der Sortiermaschine, die über einen GPIO-Pin angesteuert werden. Die Klasse "Motor" übernimmt die Ansteuerung des Förderbands. Der Motor wird über ein digitales Ausgangssignal aktiviert. Die Positionierung des Motors erfolgt über einen Inkrementalgeber, dessen Signale an einem digitalen Eingangspin ausgewertet werden. Es besteht die Möglichkeit, eine vordefinierte Grenze zu setzen (runUntilLimitReached()). Mithilfe von update() lässt sich die Position vom Förderband aktualisieren und mit der Methode getReached() kann man überprüfen, ob das Limit erreicht wurde. Durch diese Funktionalitäten ist die Klasse in der Lage, präzise Werkstücke über das Förderband zu bewegen. Die Klasse "Lichtschranke" kapselt die Funktion eines optischen Sensors, der prüft, ob sich ein Werkstück an einer bestimmten Position befindet. Dies ist insbesondere an zwei Stellen der Sortieranlage notwendig: einmal beim Eingangslager, um zu erkennen, ob ein Werkstück

bereitsteht, und einmal an der Verarbeitungseinheit, um sicherzustellen, dass das Werkstück diese erreicht hat. Über die Methode isDetected() lässt sich überprüfen, ob eine Holzscheibe durch den Sensor detektiert wurde.

4.1.4 LCD Display

Die Informationen, die auf dem LCD-Display dargestellt werden sollen, werden durch die Klasse "MaschinenStatusAnzeige" realisiert. Sie hat einen Zeiger auf die Klasse "Controller" der Zustandsmaschine und erhält dadurch die Daten, welche ausgegeben werden sollen.

4.1.5 Zustandsautomat

Die Hauptlogik ist als endlicher Zustandsautomat im Paket "StateMaschine" realisiert. Die abstrakte Klasse "IFactoryState" definiert Funktionen, welche jede konkrete Klasse eines Zustands implementieren muss. Jeder Zustand soll über zwei zentrale Funktionen verfügen: Eine Methode zur Rückgabe des Zustandsnamens sowie eine Methode zur Ausführung der zustandsspezifischen Anweisungen. Dadurch lässt sich die Sortieranlage einfach durch einen neuen Zustand erweitern, indem man eine neue Klasse hinzufügt, welche die Klasse "IFactoryState" erweitert. Jeder Zustand erhält bei der Erzeugung im Konstruktor eine Referenz auf ein "FactoryContext" Objekt, dieses stellt die benötigten Referenzen auf Motoren, Sensoren und andere Komponenten bereit und dient somit als Kontextobjekt für jeden Zustand.
Außerdem gibt es eine Klasse "Controller", welche den aktuellen Zustand des Automaten verwaltet und ausführt.

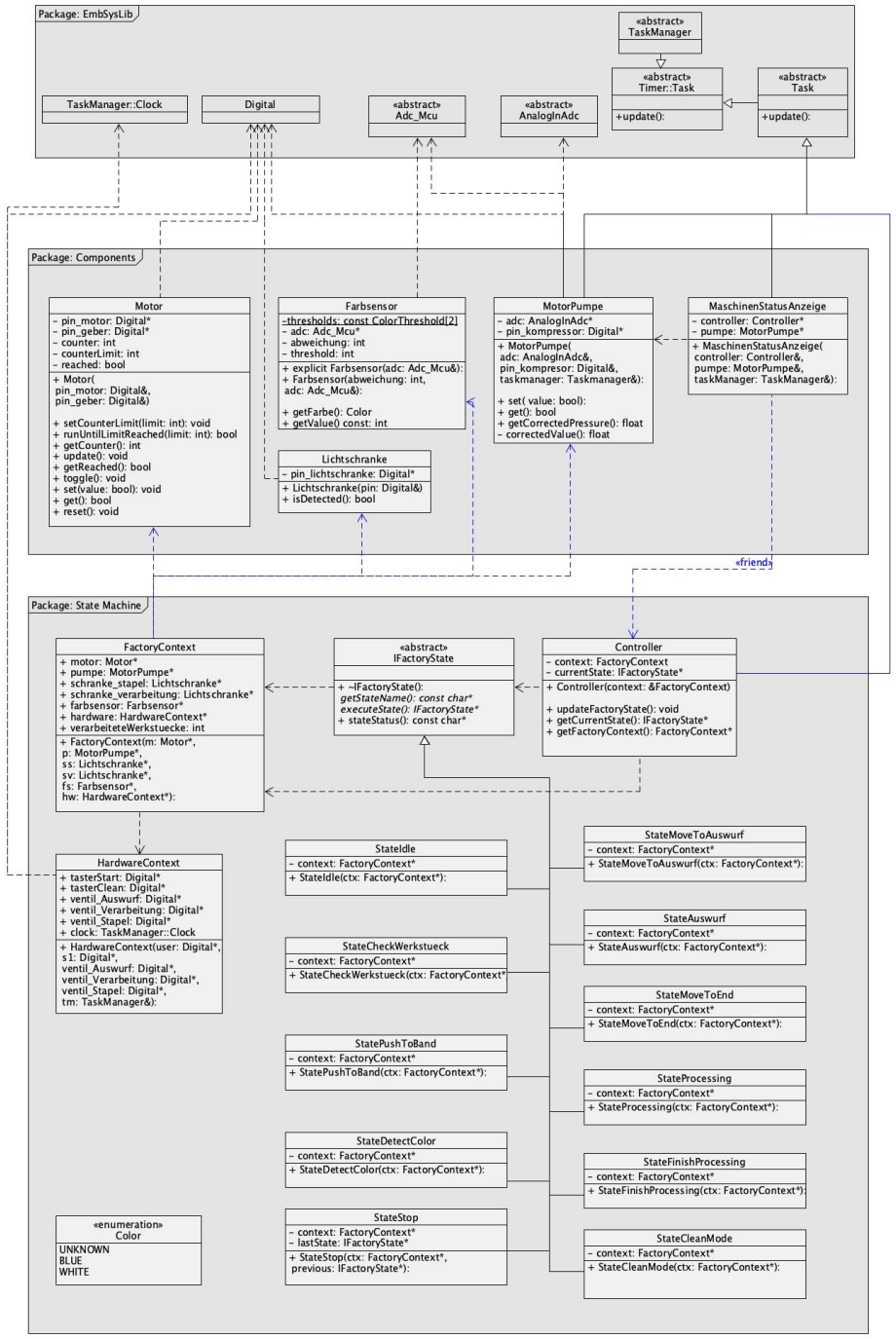

Figure 2: Klassendiagramm

4.2 Statemachine

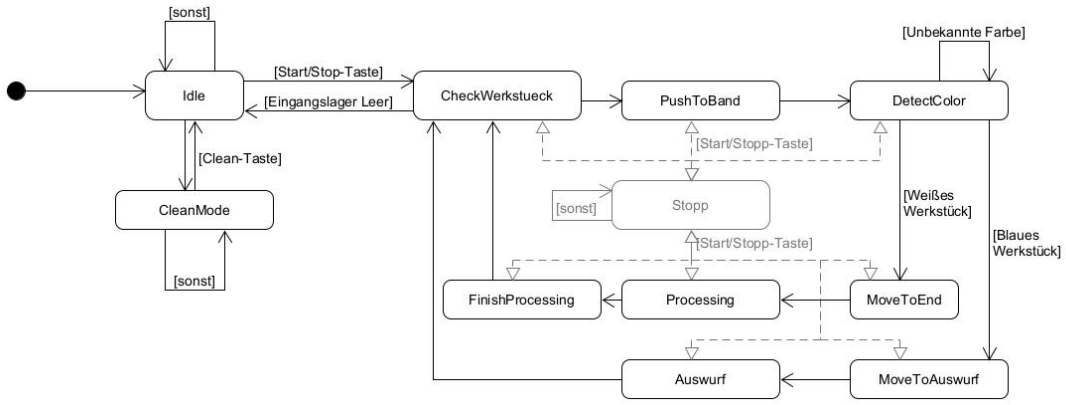

Figure 3: Statemachine

Notation:

Keine `[ edingung]` Nach Abschluss des Zustandes erfolgt ein bedingungsloser Übergang in den nächsten Zustand.

`[ edingung]` Die Bedingung muss erfüllt sein, um dem Übergang zu folgen.

Die Bedingung ist entweder durch Sensorik oder durch Nutzereingabe gesteuert.

Die gestrichelten Übergänge gelten für den Stopp zustand. Aus dem Stopp zustand wechselt man immer über denselben Übergang, über den man ihn betreten hat.

Die *tatemachine* beschreibt die Übergänge der Sortiermaschine zwischen einzelnen Zuständen. Sie war ein hilfreiches Werkzeug, um die Übergänge zu debuggen, einzelne Zustände isoliert zu testen und das Verhalten der Maschine gezielt zu beobachten.

5 Test und Veri kation

Anforderung	**Testablauf**	**Ergebnis**	**Bemerkung**
Eingangslager Detektierung Werkstück	Werkstücke werden in dem Eingangslager platziert. Ihre Detektierung in der Lichtschranke und Software geprüft.	Bestanden	-
Transport Eingangslager zum Farbsensor	Aktivierung der Ventile und des Kompressors für den Druckausgleich. Werkstück aus Eingangslager wird auf das Förderband unter den Farbsensor befördert. Kompressor baut Druck bei bedarf wieder auf.	Bestanden	Die resultierende Position des Werkstücks variiert, da die Distanz zum Kolben und der Tre punkt des Kolbens auf das Werkstück nicht deterministisch gleich sind.
Farberkennung	Der Farbsensor erhält unterschiedlich farbige Werkstücke aus dem Eingangslager. Die Kategorisierung in die vorgegebenen Farbklassen wird geprüft und Verifiziert.	Bestanden	Die Kategorisierung hängt von konstant Diskreten werten ab welche bei kleinen Änderungen am Aufbau die Funktionalität beeinträchtigen. Dazu zählt ebenfalls die Positionierung des Werkstücks unter dem Sensor.

Bandtransport	Werkstücke sollen eine voreingestellte Distanz auf dem Band transportiert werden. Stimmt die Anzahl der Tastenbetätigungen mit der Position des Werkstücks überein.	Bestanden	Werkstück klemmt bei ungenauer Positionierung zwischen Werkstücken vom Eingangslager und Aufbau. Dies Verschiebt die Endposition von auszuwerfenden Werkstücken auf dem Band.
Auswurf von Werkstücken	Aktivierung des Ventils. Überprüfung des korrekten Auswurfs.	Bestanden	Positionierung des Werkstücks auf dem Band nicht immer gleich. Siehe Bandtransport
Verarbeitung von Werkstücken	Aktivierung des Ventils. Überprüfung des korrekten Verarbeitens.	Bestanden	-
LCD-Display	Das Display zeigt Informationen über den zustand der Sortiermaschine an.	Bestanden	-
Start-/Stopp-Taste	Idle Modus Startet Verarbeitung Verarbeitender Modus Stoppen und speichern des aktuellen Zustandes sowie fortfahren bei erneutem Betätigen der Start-/Stopp-Taste.	Bestanden	-
Clean-Taste	Funktion verfügbar nur im Idle Zustand. Bei Betätigung wird Auswurf und Förderband aktiviert. Deaktivierung bei erneutem betätigen.	Bestanden	-

6 Fazit

Durch die Veranstaltung konnten wir unsere Kenntnisse in der Programmierung von eingebetteten Systemen erheblich verbessern. Die Gruppenarbeit an der Sortiermaschine hat viel Spaß gemacht, auch wenn es häufiger Herausforderungen gab, die es zu bewältigen galt. Das Lesen von Dokumentation und Testen an der Sortiermaschine selbst hat sehr viel Freude gebracht, aber es gab leider auch ein paar Momente der Verzweiflung. Zudem haben wir es, als sehr angenehm empfunden, dass es fast immer die Möglichkeit gab, in das Labor zu gehen, um seine Codeänderungen am Fischertechnik-Bausatz auszutesten oder allgemein am Projekt zu arbeiten.

Eine der größten Herausforderungen war das Verstehen und Anwenden der technischen Dokumentationen. Insbesondere die Informationsdichte und teilweise verstreute Struktur von Hardware- und Bibliotheksdokumentationen machten das gezielte Nachschlagen anspruchsvoll.

Die Implementierung mit der "EmbSysLib" war komplex, z.B. für den ADC Sensor, aber gleichzeitig wurde uns der Einstieg in die Hardware-nahe Programmierung vereinfacht, da wir nicht die einzelnen Register des Mikrocontrollers händisch verwalten mussten.

Die Programmiersprache C++ war eine besondere Erfahrung zu lernen. Zu Anfang haben wir nur von Kontrollstrukturen und der Main-Klasse Gebrauch gemacht. Mit steigender Anzahl von Zeilen in der Main-Methode, haben wir Funktionen in Klassen ausgelagert. Dies führte zu einer Verbesserung der Struktur im Code, trotzdem gab es anfänglich keine Modularität oder Flexibilität im Quellcode. Dies ließ sich zum einen darauf zurückführen, dass die Sortiermaschine sehr linear ist, bedeutet eine Komponente agiert nach der anderen. Ein weiterer Grund war, dass die Funktionalität für Komponenten, wie z.B. das Ventil, durch die "EmbSysLib" bereits vorhanden war. Wrapper-Klassen empfanden wir als überflüssig, da es ein reines Propagieren von Variablen war. Erst als unser erster Prototyp, der auf einem Enum und einer Switch-Case-Struktur basierte, zu umfangreich und unhandlich wurde, haben wir die objektorientierte Statemachine implementiert. Diese Erfahrung war ausschlaggebend dafür, dass wir eine höhere Abstraktion eingeführt und das Klassendesign flexibler gestaltet haben. Die Refaktorierung

stellte sich als anspruchsvoll heraus, da fehlende Kenntnisse im Umgang mit Imports, Namespaces und Forward Declarations zu unerwarteten Problemen im Code führten. Letztendlich führte die Refaktorierung zu einer signifikanten Verbesserung der Wartbarkeit und Testbarkeit des Quellcodes.
Die Zusammenarbeit über GitHub erwies sich in diesem Projekt als schwieriger als erwartet. Insbesondere die Verwendung von bestimmten Dateien aus dem STM32CubeIDE-Projekt stellte sich als problematisch heraus, da diese auf den unterschiedlichen Betriebssystemen der Teammitglieder teilweise nicht kompatibel waren. Dadurch kam es wiederholt zu Versionskonflikten und technischen Problemen beim Austausch der Projektstände.
Aus diesem Grund wurde entschieden, den Fokus verstärkt auf die eigentliche Entwicklung der Steuerungssoftware zu legen, während der strukturierte Einsatz von Git und Versionskontrolle im weiteren Verlauf des Projekts in den Hintergrund trat. Auch wenn die Nutzung von GitHub in diesem Fall nicht wie geplant funktionierte, zeigte sich dennoch, dass es für eine gute Kollaboration essenziell ist. Zukünftig sollte zu Beginn bereits ein robuster Workflow für die Entwicklungsumgebung und Versionsverwaltung eingeführt werden.

Insgesamt war das Projekt eine intensive, aber sehr lohnende Erfahrung, die sowohl technische Fähigkeiten als auch Teamarbeit und strukturiertes Arbeiten gefördert hat.