Christopher O'Connor

BBBaden

M242

Mikroprozessoranwendungen realisieren

Inhaltsverzeichnis

[2 Abbildungsverzeichnis 2](#_Toc61515806)

[3 Management Summary 3](#_Toc61515807)

[3.1 Projektübersicht 3](#_Toc61515808)

[3.2 Lehrbetrieb 3](#_Toc61515809)

[3.3 Involvierte Personen 3](#_Toc61515810)

[4 Aufgabenstellung 4](#_Toc61515811)

[4.1 Ausgangslage 4](#_Toc61515812)

[4.2 Zielsetzung 4](#_Toc61515813)

[4.2.1 Ampelkonfiguration 5](#_Toc61515814)

[4.2.2 Shift Register 5](#_Toc61515815)

[4.2.3 Hardware-Komponenten verbinden 5](#_Toc61515816)

[4.2.4 Queue 5](#_Toc61515817)

[4.2.5 Timed Interupts 5](#_Toc61515818)

[4.3 Erweiterungen 5](#_Toc61515819)

[4.4 Erkannte Risiken und Massnahmen 5](#_Toc61515820)

[4.4.1 Input & Output Erweiterung 5](#_Toc61515821)

[4.4.2 Queue Abhängigkeit 5](#_Toc61515822)

[4.4.3 Sensoren Input 6](#_Toc61515823)

[4.5 Mengengerüst 6](#_Toc61515824)

[4.6 Rahmenbedingungen 6](#_Toc61515825)

[4.6.1 Hilfsmittel 6](#_Toc61515826)

[4.6.2 Vorkenntnisse 6](#_Toc61515827)

[4.6.3 Arbeitsumgebung 6](#_Toc61515828)

[5 Projektplanung 6](#_Toc61515829)

[5.1 Zeitplan 7](#_Toc61515830)

[6 Entscheiden / Analyse 8](#_Toc61515831)

[7 Realisierung 8](#_Toc61515832)

[8 Testing 8](#_Toc61515833)

[8.1 Testfallspezifikation 8](#_Toc61515834)

[8.2 Testprotokoll 8](#_Toc61515835)

[9 Auswertung 8](#_Toc61515836)

[10 Arbeitsjournal 9](#_Toc61515837)

[11 Anhang 10](#_Toc61515838)

# Abbildungsverzeichnis

Abbildung 1 Skizze der Ampelsteuerung, Vogelperspektive 4

# Management Summary

## Projektübersicht

Diese Dokumentation und die daraus entstandene Präsentation sind Bewertungsgrundlagen für das Modul 242.

Nach den Standards des Projekt Managements werden folgende Projektphasen dokumentiert:

* Konzept
* Planung
* Analyse
* Realisierung
* Testphase
* Auswertung

/\*

Zusätzliche Informationen, welche in der nachfolgenden Dokumentation zu finden sind:

* Abbildungsverzeichnis
* Quellenverzeichnis
* Arbeitsjournal
* Glossar
* Anhang

/\*

## Lehrbetrieb

SANTIS Training AG

Hohlstrasse 550

8048 Zürich

[training@santismail.ch](mailto:training@santismail.ch)

## Involvierte Personen

|  |  |  |
| --- | --- | --- |
| Auszubildender: | Christopher O’Connor | [christopher.oconnor@santismail.ch](mailto:christopher.oconnor@santismail.ch) |
| Lehrperson: | Manuel Bachofner | [manuel.bachofner@bbbaden.ch](mailto:manuel.bachofner@bbbaden.ch) |

# Aufgabenstellung

## Ausgangslage

Ich habe mich schon immer gefragt wie eine Ampelsteuerung funktioniert. Ich bin auch davon überzeugt, dass nicht alle gleich sind. Ich habe mich dazu beschlossen meine eigene zu realisieren in einem Arduino Projekt. Ich erhoffe mir dadurch besser zu verstehen wie sie funktionieren könnten.

## Zielsetzung

Mit dem Arduino Projekt soll eine komplexe automatisierte Ampelsteuerung von einer Kreuzung realisiert werden. Dazu werden Sensoren verwendet welche Fahrzeuge frühzeitig erkennen und die Ampeln entsprechend Auslastung korrekt umschalten. Damit auch Fussgänger die Möglichkeit haben die Strasse sicher zu überqueren, gibt es pro Strassenübergang einen Schalter wie bei einer echten Kreuzung. Wird dieser gedrückt, wird auf diese entsprechend Rücksicht genommen.

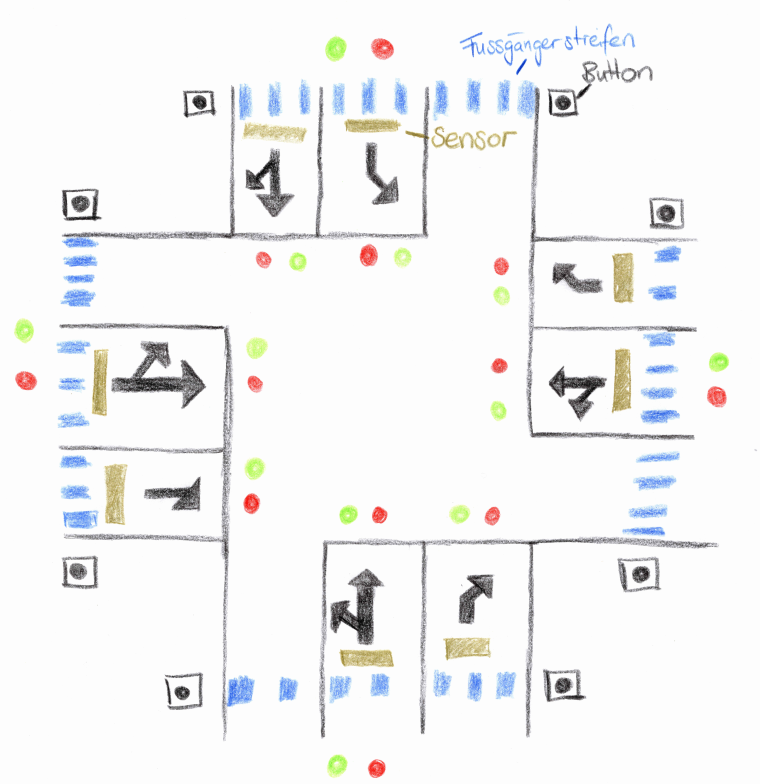
Nachfolgend eine Hand-Skizze der Kreuzung der Vogelperspektive:  


Abbildung Skizze der Ampelsteuerung, Vogelperspektive

### Ampelkonfiguration

Bei einer Kreuzung können mehrere Ampeln gleichzeitig Grün sein. Diesbezüglich muss die Abhängigkeit ausfindig gemacht und notiert werden. Damit auch andere zu einem späteren Zeitpunkt die Konfigurationen nachvollziehen können benötigt es noch eine gute Naming Convention der einzelnen Ampeln.

### Shift Register

Die Anzahl an Input & Outputs sind auf einem Arduino begrenzt. Da das Ampelsystem über 40 Outputs und 16 Inputs verfügt muss mit 5x Shift-Register die Verfügbaren Slots erweitert werden.

### Hardware-Komponenten verbinden

Die Hardware (Arduino, Shift-Register, LEDs, Sensoren und Schalter) müssen auf mehrere Breadboards korrekt miteinander verbunden werden.

### Queue

Eine Queue oder etwas Ähnliches wird verwendet um die Informationen von den Sensoren zwischen zu speichern in einer Liste. Zum Beispiel Ampel 1 kommt ein Auto. Ampel 4 kommt ein weiteres. Fussgänger-Schalter 4 wurde gedrückt. Die Queue wird dann vom Programm fortlaufend abarbeitet und Schaltet die Kreuzung entsprechend.

### Timed Interupts

Die Sensoren müssen in der Lage sein «gleichzeitig» Inputs zu lesen und entsprechend in die Queue zu setzten während eine Ampelkonfiguration aktiv ist. Mit den entsprechenden «Timed Interupts» kann dies ermöglicht werden.

## Erweiterungen

Sollte noch genügen Zeit vorhanden sein, könnten folgende Ideen noch zusätzlich realisiert werden:

* Orange LED als Bestätigung, dass Fussgänger den Knopf gedrückt haben.
* Remote Controll für eine Buslinie. Wird diese betätigt, hat der Bus Priorität vor der Queue.
* Rot Blitzer

## Erkannte Risiken und Massnahmen

### Input & Output Erweiterung

Die Anzahl an Input & Outputs sind auf einem Arduino begrenzt. Da das Ampelsystem über 40 Outputs und 16 Inputs verfügt muss mit 5x Shift-Register die Verfügbaren Slots erweitert werden.

### Queue Abhängigkeit

Eine Queue oder etwas Ähnliches wird verwendet um die Informationen von den Sensoren zwischen zu speichern. Weil die Ampeln aber voneinander Abhängig sind, sprich es ist nicht nur immer eine Ampel grün, sondern mehrere gleichzeitig, muss dies in der Queue berücksichtigt werden damit eine Ampelkonfiguration von mehreren einzelnen Ampeln nicht mehrfach in der Queue ist. Beispiel:

Ampel 1 & 2 haben gleichzeitig grün und bei beiden wurde ein Auto erkannt. In der Queue wird aber nur 1x ein Eintrag gemacht und nicht Ampel 1 und Ampel 2 zweimal nacheinander auf grün geschaltet.

### Sensoren Input

Die Sensoren müssen in der Lage sein «gleichzeitig» Inputs zu lesen und entsprechend in die Queue zu setzten während eine Ampelkonfiguration aktiv ist.

Leider wird Multithreading vom Arduino nicht unterstützt. Hier heisst die Lösung «Timerinterrupt». Dadurch kann Multithreading ähnliches verhalten simuliert werden.

## Mengengerüst

|  |  |  |
| --- | --- | --- |
| Anzahl | Produkt | Beschreibung |
| 1x | Arduino UNO REV3 |  |
| 8x | HC-SR501 | PIR Motion Detection Sensor |
| 3x | 74HC595 | Shift Register 8bit Output |
| 2x | 74HC165 | Shift Register 8bit Input |
| 8x | Schalter / Button |  |
| 12x | LED-Grün | Grüne Ampel |
| 12x | LED-Rot | Rote Ampel |
| 4x | LED-Orange |  |
| 4x | Breadboard |  |

## Rahmenbedingungen

### Hilfsmittel

Zur Realisierung der Arbeit sind sämtliche Hilfsmittel erlaubt, welche im Rahmen einer LB zugelassen sind. Inbegriffen sind sowohl bestehende Dokumentationen, Code und Literatur als auch das Internet und andere, deklarierte Quellen. Befragung von Mitarbeiter und/oder externen Hilfspersonen müssen nachweisbar dokumentiert werden im Q&A Dokument.

Primäres Dokumentationsmittel ist Microsoft Word. Zusätzliche Hilfsprogramme wie PowerPoint, Notepad++ und AVR-Studio sind ebenfalls zugelassen. Bildschirmaufzeichnungen können mit der Software Snipping Tool durchgeführt werden.

### Vorkenntnisse

Eine transparente Bewertung der durchgeführten Projektarbeit erfordert die Preisgabe der fundierten Vorkenntnisse. Projektrelevante Vorkenntnisse werden stichwortartig festgehalten:

* ECDL Advanced Zertifikat
* Modul 121 Steuerungsaufgaben bearbeiten
* Programmiersprache C – Grundlagen

### Arbeitsumgebung

Primäre Arbeitsort ist ein Schulraum an der BBBaden. Ausweichungen ins HomeSchooling sind möglich.

# Projektplanung

Die Planung stellt ein wichtiges Hilfsmittel dar und wurde deshalb äusserst sorgfältig erstellt. Hier kann der Projektablauf mit der verfügbaren Zeit überprüft werden. Gegebenenfalls können Zeitknappheiten frühzeitig erkannt und angegangen werden.

## Zeitplan

asdf

# Entscheiden / Analyse

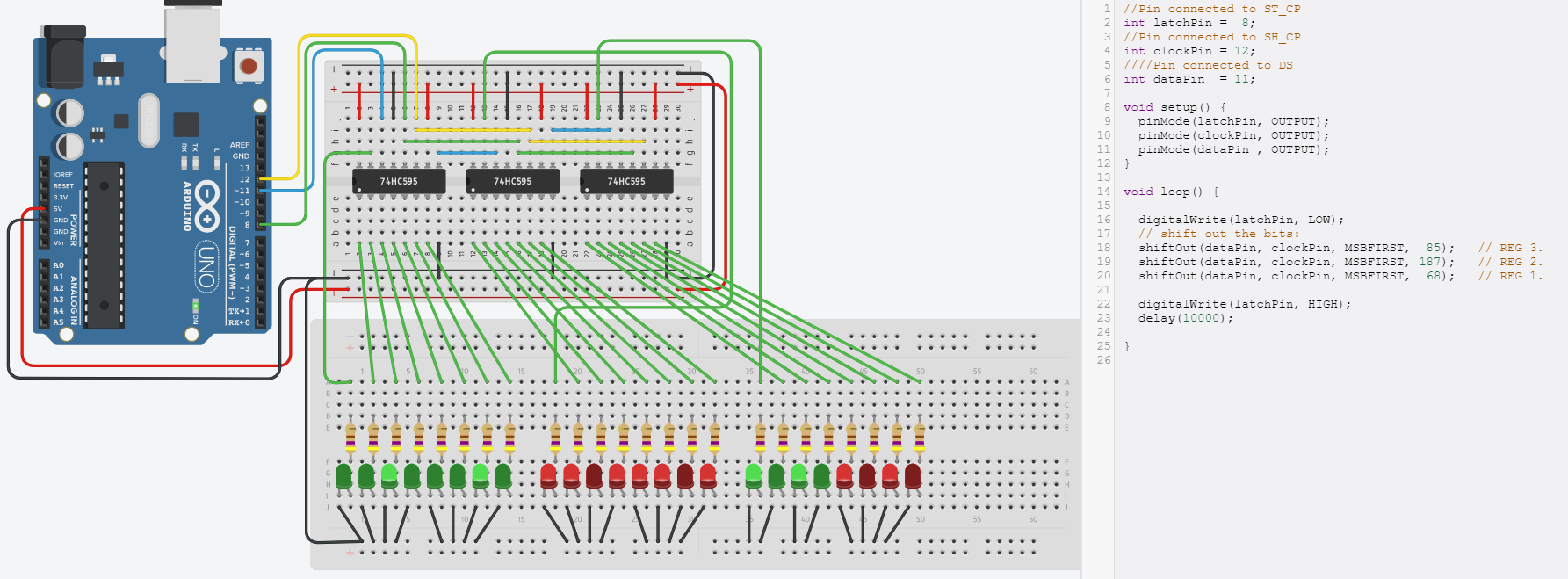
* Alle Varianten
* Wer hat was, wann und wie entschieden.

Mind. 4x!

# Realisierung

## Shift Register Out (LEDs)

Ein Shift Register bestehend aus 8bit und speichert entsprechend denn Zustand von jedem bit als 1 -ON oder 0 - OFF. Die Shift Register können aneinandergekoppelt werden. Für meine Kreuzung benötige ich genau 24 LEDs, 3x 8bit.



Ich habe mich dazu entschieden den ersten Shift Register für die grünen Ampeln, den zweiten Shift Register für die roten Ampeln und den dritten Shift Register für die Fussgänger Ampeln (0-4 grün / 4-8 rot) zu verwenden. Möchte man nun alle Ampeln auf Rot setzen wäre dies:

**[00000000] [11111111] [00001111]**

Die Standard Library von Arduino ermöglicht die Kommunikation mit einem Shift Register auch im Dezimalsystem. Dies ist zwar schwerer zu lesen aber weniger Fehleranfällig. Folglich wird aus den 3Byte: **[0] [255] [15]**

## Ampel Konfiguration

Bei einer Kreuzung können mehrere Ampeln gleichzeitig Grün sein. Diesbezüglich muss die Abhängigkeit ausfindig gemacht und notiert werden. Mithilfe der Skizze, habe ich alle Scenarios durchgespielt und mir Notizen dazu gemacht. Ausgangspunkt war jeweils der Sensor welcher direkt mit einer Ampel abhängig ist. Danach konnte ich alle anderen Ampeln welche grün sind ausfindig machen. Aus nachfolgendem Bild kann die Nummerierung entnommen werden:

ASDF

### Konfigurationen

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | 1. Byte in Dez | 2. Byte in Dez | 3. Byte in Dez | Sensoren IDs |
| 1 | 68 | 187 | 85 | 2, 6 |
| 2 | 145 | 110 | 15 | 1, 4, 8 |
| 3 | 32 | 223 | 45 | 3, 13, 14 |
| 4 | 8 | 247 | 150 | 5, 9, 10, 15, 16 |
| 5 | 2 | 253 | 135 | 7 |
| 6 | 129 | 126 | 105 | 11, 12 |

## Shift Register In (Sensoren / Schalter)

## Hardware-Komponenten verbinden

Ich habe alle Teil-Komponenten auf <https://www.tinkercad.com/> zusammengesetzt und getestet. Dies ermöglichte schnelle Anpassungen und eine gute Fehlersuche. Die daraus entstandenen Vorlagen werden für die reale Schaltung verwendet und daraus abgeleitet.

ASDF

## Queue

## Timed Interrupts

# Testing

## Testfallspezifikation

## Testprotokoll

# Auswertung

asdf

# Arbeitsjournal

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Datum | Benötigte Zeit | Arbeitsschritt | Bemerkungen | Visum |
| 05.01 | 60’ | Recherche | Teilkomponenten & Mögliche Probleme | CO |
| 05.01 | 120’ | Vorbereiten Doku | Aufgabenstellung + Mengengerüst | CO |
| 05.01 | 60’ | Planen | Grobe Planung des Projekts | CO |
| 12.01 | 60’ | Teilaufgaben Definiert | Genauere Spezifikation von den Zielen und Unterteilung in unterziele | CO |
| 12.01 | 60’ | Zeitplan erstellen |  |  |
| 12.01 | 200’ | Shift Register | Ich hatte Probleme eine gute Quelle zu finden. Viele Anleitungen / Erklärungen hatten falsche oder nicht vollständige Informationen und verwendeten andere Methoden oder gar andere Librarys. Das Umsetzen auf Tinkercard hat auch viel Zeit gekostet. | CO |
|  |  |  |  |  |

# Anhang