

CAOS Projekt : 2 x 6 x 12 LED RGB Quader

Joey Zgraggen, Moritz Würth, Viktor Gsteiger

January 16, 2020

TODO: BILD VOM PROJEKTERZEUGNIS

Inhaltsverzeichnis

1	Einführung	2
2	Verwendete Materialien	2
2.1	RGB LED's	2
2.2	Schieberegister	2
2.3	Transistoren	2
2.4	Sensoren	3
3	Hintergrund	3
4	Implementation	3
4.1	Alphabet	4
4.2	Stars	4
4.3	Firework	4
4.4	Welcome	4
4.5	Game of Life	4
4.6	Snake	5
5	Sensoren	5
5.0.1	Temperatur- und Luftfeuchtigkeitssensor	5
5.0.2	Infrarot-sensor	5
6	Aufbau des zweidimensionalen Bildschirmes	6
6.1	Bildschirm	6
6.2	Schieberegister	6
6.3	Probleme beim Aufbau	6
7	Ergebnisse	7
8	Zusammenfassung	8

1 Einführung

Die Grundidee von unserem Projekt war es zuerst einen 5 x 5 x 5 LED RGB Würfel zu bauen, jedoch haben wir uns dann im Prozess der Entscheidungsfindung dazu entschieden, etwas anderes zu bauen was sich vom "typischen" LED RGB Würfel unterscheidet. Dies bedeutete, dass wir uns die gesamte Hard- und Software des Bildschirmes selber überlegen mussten und die Fragestellung der korrekten und effizienten Kommunikation zwischen Arduino und den LED's stellen mussten.

Deshalb haben wir die Dimensionen ein wenig angepasst, sodass wir am Ende einen dreidimensionalen Bildschirm haben werden. Die Anordnung der LED's kann als Koordinatensystem verstanden werden, bei dem man mittels x- und y-Koordinaten auf die einzelnen LED's zugreifen kann. Dies erleichtert es massiv, Effekte zu schreiben. Die gesamte Kommunikationssoftware zwischen dem Arduino und den LED's wurde grösstenteils selber geschrieben und es wird für die Kommunikation nur auf die externe SPI Library zugegriffen[1].

Das Projekt lässt einen grossen Spielraum bezüglich Ideen zu. Sollte man also selbst Ideen haben, welche sich gut zu unserem Projekt ergänzen würden, kann man diese ohne weitere Probleme implementieren. Die von uns bereitgestellte Software hat sehr einfache Schnittstellen, mit denen man einfach weitere Effekte schreiben kann[2].

2 Verwendete Materialien

Im Folgenden ist eine Auflistung der verschiedenen wichtigsten Materialien, welche für das Projekt verwendet wurden.

2.1 RGB LED's

Um den LED-Bildschirm ein wenig bunter gestalten zu können, wurden RGB's verwendet.:

- LED RGB Common Cathode 4-Pin F5 5MM Diode

2.2 Schieberegister

Schieberegister erweisen sich für ein solches Projekt als äusserst nützlich, um die vielen Pins der einzelnen RGB LED's ansprechen zu können.

Wir haben uns dabei an schon bereit existierenden Projekten orientiert [3] und uns für die 74HC595 8-bit Schieberegister entschieden. Diese erweisen sich in der Programmierung als intuitiv, weil man einfach mit Byte-Arrays arbeiten kann, um die einzelnen Schieberegister mit Informationen zu "befüllen". Weiter existieren für diese Schieberegister vorgefertigte Libraries, um die Kommunikation zu erleichtern[1].

Dies nimmt nicht die Denkarbeit ab, wie man die LED's anordnen will und wie man die verschiedenen Farben anspricht, jedoch erleichtert es die direkte Kommunikation.

2.3 Transistoren

Anfangs wollten wir für eine garantierte Langlebigkeit unseres Projektes Transistoren verwenden, aber nachdem wir uns bezüglich der Notwendigkeit von Transistoren für unser Projekt informierten haben wir uns letztendlich dazu entschieden, keine zu verwenden. Dies vor allem, da wir nie viele LEDs gleichzeitig laufen lassen. Durch einen geschickten Multiplex Algorithmus erhalten nie mehr als 8 pins gleichzeitig Strom, was den Anoden viel Arbeit abnimmt.

2.4 Sensoren

Für das Projekt wurden zusätzlich Sensoren verwendet, um weitere Features zu gewährleisten. Diese werden im Kapitel 6 detaillierter behandelt.

Folgende Sensoren stehen zur Zeit zur Auswahl:

- Temperatur und Luftfeuchtigkeitssensor
- Infrarot-Sensor
- Real-time clock

3 Hintergrund

Nachdem wir viele Projekte im Internet gefunden haben, welche unserem Projekt sehr nahe kommen, hat dies bei uns die Motivation erweckt, ebenfalls ein ähnliches Projekt zu machen. Wir haben uns sehr intensiv in einem vorhandenen *online Protokoll* bezüglich dem Bauen eines 8x8x8 Quaders eingelesen, um Informationen zu sammeln[3]. Im weiteren Verlauf haben wir unseren Code jedoch deutlich von dem des Beispielprojektes abgewandelt, da wir erstens andere Dimensionen hatten und der Multiplexing algorithmus für uns nicht die wünschenswerten Resultate lieferte.

Wir haben uns auch überlegt eine Box zu bauen, welche die ganze Lötarbeit in einem Raum verstaucht, um das ganze optisch ein wenig angenehmer zu gestalten. Die verwendete Plexiglasscheibe ist mit Absicht durchsichtig, sodass man trotzdem noch die damit verbundene Arbeit ansehen kann.

4 Implementation

Die Implementation auf der Code-Seite war von Anfang an sehr strukturiert und mithilfe von viel Recherchearbeit konnten wir schon sehr bald einen ersten funktionierenden Prototypen herstellen. Die Kommunikation mit den Bitshifttern war dabei die heikelste Stelle unserer Implementation, von der alles andere abhing. Trotz dem haben wir schon relativ früh die nötigen Schnittstellen definiert, wodurch man parallel an der Grundlogik arbeiten konnte und auch schon Effekte erarbeiten konnte.

Damit wir dem Code einen besseren Überblick verschaffen konnten und damit sich auch alle Gruppenmitglieder besser mit dem Code auseinander setzen können haben wir den Code so unterteilt, dass die verschiedenen Bereiche wie Logik, Features und main separiert sind.

Unser verwendetes Konzept erlaubt es auch aussenstehenden Personen, welche am Projekt nicht teilgenommen haben und mit dem Code nicht sehr vertraut sind ein eigenes Feature leicht zu implementieren und das Projekt zu erweitern[2].

Im Folgenden werden verschiedene von uns implementierten Features kurz erläutert.

4.1 Alphabet

Durch die Klasse Alphabet ist es einfach möglich, auf dem Bildschirm Buchstaben und Sätze darzustellen. Die Klasse unterscheidet nicht zwischen klein und Grossbuchstaben und kann das gesamte Alphabet plus ein paar Sonderzeichen plus alle Zahlen darstellen. Die einzelnen Buchstaben sind alle Hardgecoded und deshalb macht diese Klasse Sinn, weil man dies sonst jeweils einzeln machen müsste. Die Klasse Alphabet wird vom Willkommens Effekt, von den tempereature effects, vom roll credits und vom clock Effekt benutzt und ist dadurch ein sehr zentraler Effekt.

Die Buchstaben werden dabei von rechts nach links einer nach dem anderen in den Buchstaben-Buffer geladen, dann jeweils, sofern man nicht beim letzten Buchstaben ist, vier mal auf den Hauptbildschirm rausgeshiftet und nach jedem einzelnen shiften angezeigt. Beim letzten Buchstaben wird dieser so lange nach links geshiftet, bis er vom Bildschirm verschwindet und die Kontrolle wird zurück an die rufer Methode gegeben.

4.2 Stars

Im Effekt stars werden zufällig Sterne erzeugt, welche entweder von links nach rechts oder umgekehrt fliegen. Wenn sich zwei Sterne treffen explodieren diese in einem Effekt. Die Sterne haben alle verschiedene Farben, welche auch zufällig sind.

Die Sterne sind alle eigene structs, welche die nötigen Informationen beinhalten. In der Update-Methode werden alle existierenden Sterne in ihrem Pfad fortgeführt.

4.3 Firework

Der Feuerwerk Effekt funktioniert ähnlich wie der Sterne-Effekt, jedoch fliegen hier die Raketen nur von unten nach oben und explodieren auf einer zufälligen Höhe. Die Farbe der Raketen ist auch zufällig.

Eine Rakete ist ein struct, welche alle nötigen Informationen beinhaltet und die Raketen werden in der Methode burnRocket in ihrem Pfad fortgeführt. Dabei können sie sich auch leicht nach links oder rechts bewegen.

4.4 Welcome

Der Willkommens Effekt ist ein sehr einfacher Effekt, welcher einfach mit der Methode Alphabet eine Willkommensnachricht auf dem Bildschirm anzeigt und einmal nach dem Aufstarten angezeigt wird.

4.5 Game of Life

Der Game of Life Effekt ist eine repräsentation des berühmten Game of Life des britischen Mathematikers John Horton Conway. Dabei kommen zufällig Zellen zum Leben und sterben beziehungsweise pflanzen sich nach bestimmten Regeln fort. Dabei können sie auch über den Bildschirmrand einen Effekt haben und zählen dann zu den Nachbaren auf der anderen Seite. Für mehr Informationen empfieilt es sich, den Wikipedia Artikel zum Game of Life zu lesen[4].

4.6 Snake

Hierbei geht es um das klassische Snake Spiel, welches die meisten sicherlich schon kennen[5]. Man spielt eine Schlange (Snake), welche man in vier Richtungen mittels einer Infrarot-Fernbedienung steuern kann.

Ziel ist es die verschiedenen "Pixel" (grün leuchtender RGB LED) zu erreichen, um damit die Schlange zu vergrössern. Sollte man sich in die in die Wand (Rand) bewegen oder in sich selber beißen, dann ist das Spiel vorbei und man muss von vorne beginnen.

Hierbei war bei der Implementation vor allem die Schwierigkeit, dass die Infrarot Informationen per polling geholt werden. Dadurch brauchte es ausgeklügelte Methoden, um sich die Bewegungsmeldungen in der richtigen Zeit und ohne zu viel Zeitverlust zu holen.

5 Sensoren

5.0.1 Temperatur- und Luftfeuchtigkeitssensor

Der Temperatur- und Luftfeuchtigkeitssensor, welchen wir für unser Projekt verwenden ist vom Typ DHT11. Dies ist ein eher einfacher Sensor, welcher jedoch mehr als genügt für unsere Zwecke. Dieser liest die aktuelle Temperatur und Luftfeuchtigkeit jede Sekunde und diese Information wird dann auf dem Bildschirm angezeigt.

Mit Hilfe dieser Informationen wird dann aus einem von drei Effekten ausgewählt. Entweder wird dann ein Regen, Wolken oder eine Sonne auf dem zweidimensionalen Bildschirm angezeigt. Die Berechnung hierfür ist jedoch sehr einfach und rudimentär, weshalb der Bildschirm sich nicht für eine genaue Wettervorhersage eignet.

5.0.2 Infrarot-sensor

Der Infrarot-sensor empfängt Input von einer externen Fernbedienung und man kann damit zwischen den verschiedenen Effekten wechseln. Dabei werden bei allen Effekten regelmässig die Informationen vom Sensor abgefragt und dann wird dies abgearbeitet. Die Effekte haben alle Nummern auf der Fernbedienung, man könnte jedoch relativ einfach auch eine andere Fernbedienung nehmen.

Wir haben uns hierbei extra gegen einen Interrupt gesteuerten Dateninput entschieden, weil dies das Effekthandling erschweren würde. So ist maximal ein Effekt zur gleichen Zeit am laufen und bricht ab, sobald man mit der Fernbedienung einen anderen Effekt auswählt. Speziell gehandhabt wird dies jedoch bei Snake, bei der eine Bewegungsänderung das Spiel nicht abbrikt, sondern die Schlange bewegt.

6 Aufbau des zweidimensionalen Bildschirmes

6.1 Bildschirm

6.2 Schieberegister

Wir bereits angesprochen helfen uns die Schieberegister die vielen verschiedenen Pins der RGB LED's anzusprechen. Die Schieberegister sind auf zwei Platinen angeordnet und sind mit einander mittels den entsprechenden outputs miteinander verknüpft. Wichtig war es dabei zu beachten, dass der Daten in- und output korrekt verlötet wurden, weil wir die Teilaufgaben unseres Projektes untereinander aufgeteilt haben und die Anordnung der Schieberegister mit dem Code übereinstimmen müssen.

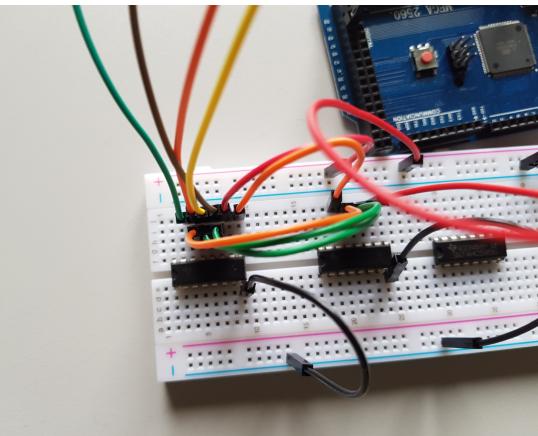
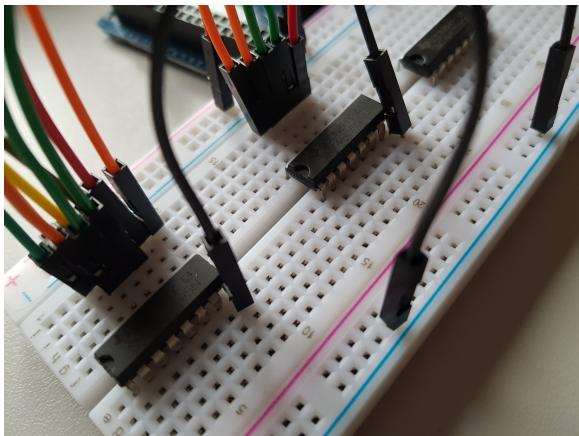
6.3 Probleme beim Aufbau

Durch die anspruchsvolle Steuerung von 216 einzelnen Pins kommt es natürlich zu Schwierigkeiten. Unsere erste Schwierigkeit war, dass wir zuerst mit einer zu kleinen und einzelnen Platine gearbeitet haben. Dies hatte zur Folge, dass die Lötarbeit schnell unübersichtlich und ungenau wurde. Auch war es so extrem schwer, die Lötstellen sauber zu verlöten. Wir haben uns nach einem herumprobieren dafür entschieden, von einer auf zwei Platinen umzusteigen, was die Arbeit massiv vereinfacht hat.

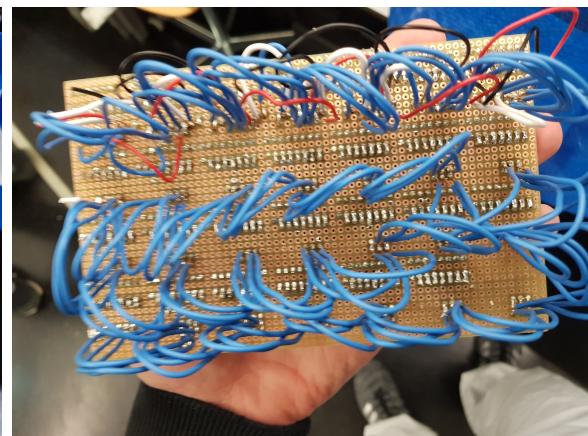
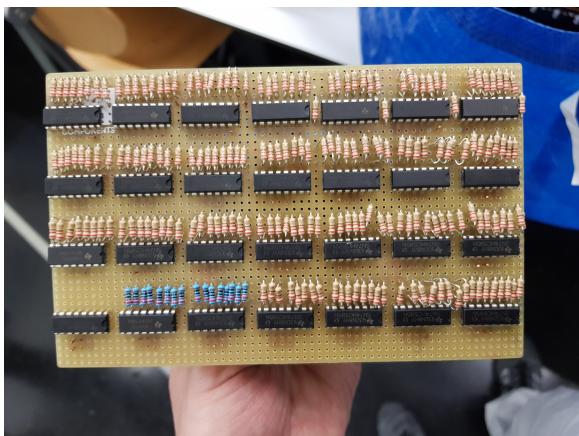
7 Ergebnisse

Im Folgenden ist der Prozess von unserem Projekt veranschaulicht:

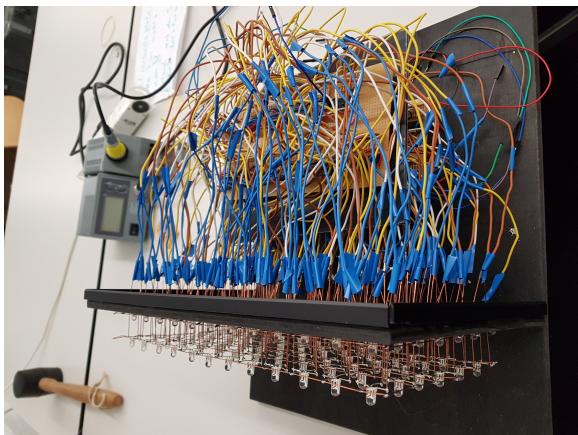
Damit wir sichergehen konnten, dass alle gekauften Materialien in einem guten Zustand sind, haben wir diese erstmal getestet. Es wäre mühsam gewesen ein Schieberegister oder RGB LED nach Fertigstellung des Projektes auszutauschen:



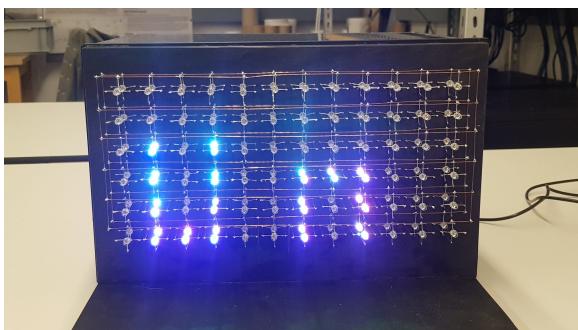
Wie bereits erwähnt haben wir uns am Anfang dazu entschieden eine Platine für all unsere Schieberegister zu verwenden und haben dann im Verlaufe des Projektes bemerkt, dass das am Schluss sehr viel Aufwand mit sich bringt alle Kabel gut auseinanderhalten zu können:



Danach haben wir zwei Platinen verwendet, wodurch das Arbeiten um einiges erleichtert wurde:



Am Schluss haben wir noch wie bereits erwähnt eine Verschalung aus Plexiglas (durchsichtig) gebaut, damit das ganze optisch besser aussieht:



8 Zusammenfassung

Da wir uns innerhalb der Gruppe schon sehr gut kennen und auch schon öfters zusammen gearbeitet haben, sei es in einem Projekt oder innerhalb der Übungsaufgaben war es für uns sehr angenehm, uns untereinander zu organisieren. Wir waren uns immer schnell einig, beispielsweise zwei statt eine Platine zu verwenden, um die ganze Lötarbeit ein wenig effizienter zu gestalten, damit Parallel gearbeitet werden kann und der ganze Hardware Teil unseres Projektes ein wenig übersichtlicher erscheinen zu lassen.

Auch wenn die Arbeit innerhalb der Gruppe in Hardware und Software unterteilt wurde, so haben wir doch immer wieder mal kleine Austausche durchgeführt, sodass jeder von beiden Bereichen erfolgreich etwas mitnehmen kann und die Zusammenhänge besser verstehen kann.

Jedes Gruppenmitglied hat mit dem Projekt neben der Vorlesung vie dazugelernt und es war eine sehr gute Erfahrung die groben Ansätze der verschiedenen Themen der Vorlesung mit der Projektarbeit zu vertiefen.

References

- [1] SPI library. *arduino.cc*. Arduino, 2020,
<https://www.arduino.cc/en/reference/SPI/>
- [2] Viktor Gsteiger, Arduino Code for the 144 project. *Github.com*. GitHub Inc., 2020,
https://github.com/vGsteiger/caos_gruppe_144/blob/master/src/README.md
- [3] JamesP383, Arduino Mega 8x8x8 RGB LED Cube. *instructables.com*. Autodesk, Inc., 2016,
<https://www.instructables.com/id/Arduino-Mega-8x8x8-RGB-LED-Cube/>
- [4] Conway's Game of Life. *wikipedia.com*. Wikimedia Foundation, 2020,
https://en.wikipedia.org/wiki/Conway%27s_Game_of_Life/
- [5] Snake (video game genre). *wikipedia.com*. Wikimedia Foundation, 2020,
[https://en.wikipedia.org/wiki/Snake_\(video_game_genre\)](https://en.wikipedia.org/wiki/Snake_(video_game_genre))