Teil 2

lesson 22

Acht LEDs per 74HC595 ansteuern

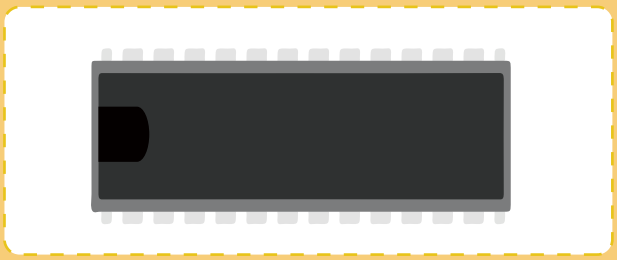
Übersicht

In dieser Lektion lernen Sie, wie man acht rote LEDs mit dem UNO Board verbinden kann, ohne dafür 8 Pins belegen zu müssen.

Sie könnten die acht LEDs auch einzeln mit vorgeschaltetem Widerstand mit dem UNO Board verbinden, Ihnen würden dabei aber sehr schnell die Pins ausgehen. Wenn Sie nur wenige Komponenten mit Ihrem Board verbinden möchten, funktioniert das noch, aber meistens möchte man noch weitere Schalter, Sensoren, Servos usw. anschließen und früher oder später gehen Ihnen die Anschlüsse aus, weil alle belegt sind. Also statt alle LEDs einzeln anzuschließen, werden wir in dieser Lektion den 74HC595 Chip benutzen. Der 74HC595 ist ein Seriell zu Parallel Wandler. Der Chip hat acht Ausgänge (was für unser Vorhaben perfekt ist) und drei Eingänge, über die er mit Daten versorgt werden kann.

Durch den Chip lassen sich die LEDs ein bisschen langsamer ansteuern, so kann man die LEDs nur noch 500.000 pro Sekunde umschalten statt 8.000.000 pro Sekunde, wie es ohne Chip der Fall wäre. Das ist nichtsdestotrotz immer noch enorm schnell, sodass man den Unterschied mit dem Auge niemals wahrnehmen wird.

Benötigte Bauteile:



(1) x Elegoo UNO R3

(1) x 830 tie-points breadboard

(8) x leds

(8) x 220 ohm resistors

(1) x 74hc595 IC

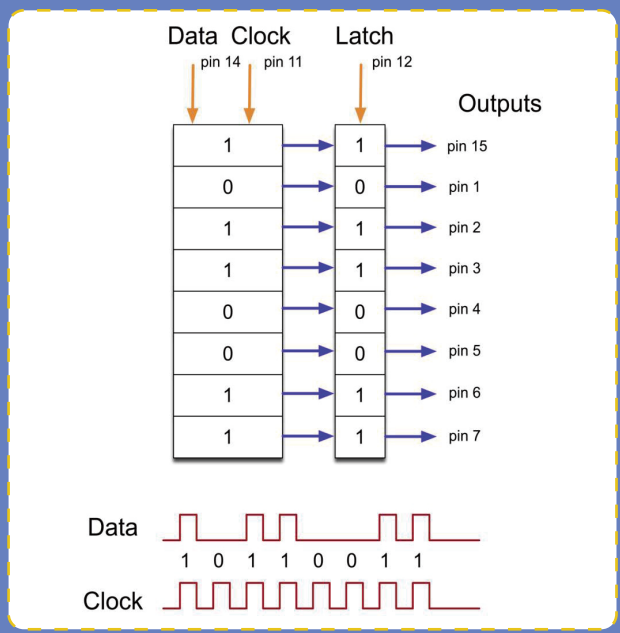
(14) x M-M Kabel (Männlich zu Männlich DuPont Jumper Kabel)

Einführung in die Komponenten

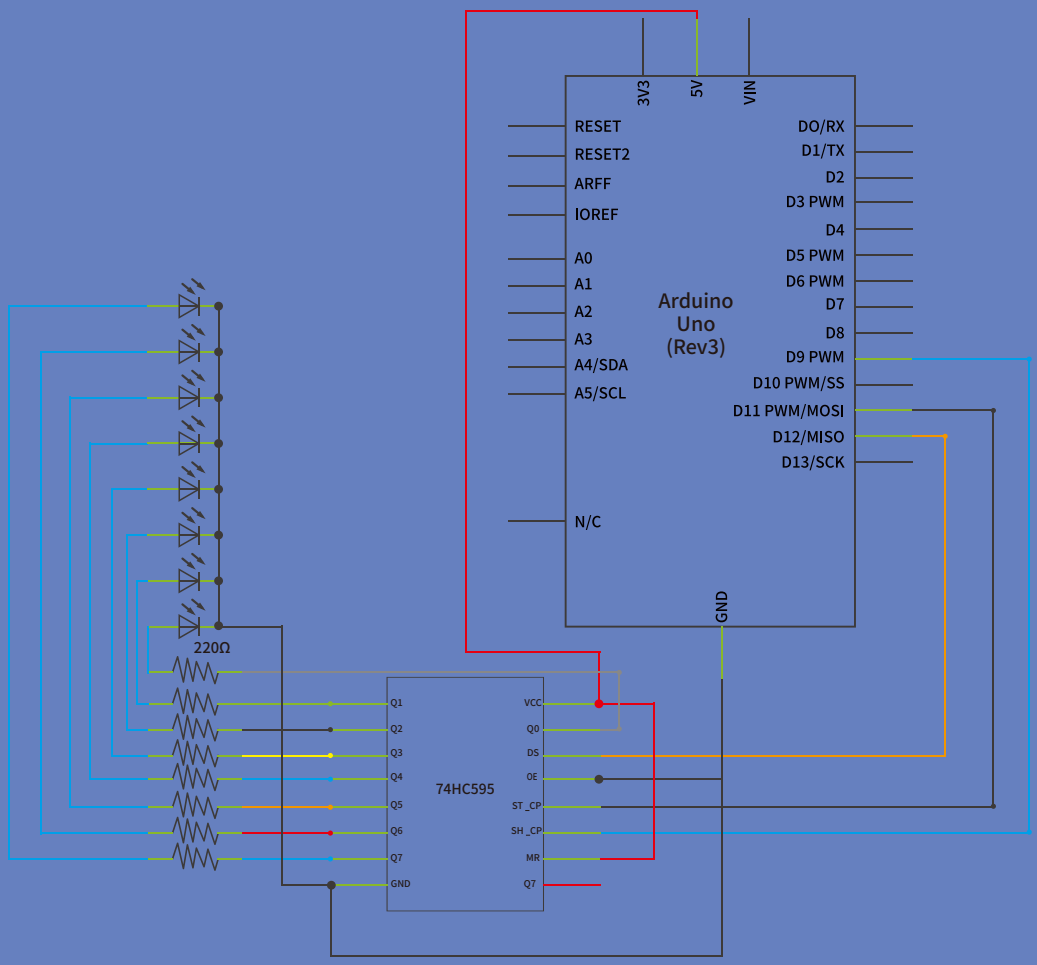
74HC595 Schieberegister:

Ein Schieberegister ist ein Chip, der acht Speicherstellen hat, die jeweils entweder eine 1 oder eine 0 speichern können. Um die Werte zu verändern, werden wir die Daten über den „Data“- und den „Clock“ -Anschluss des Chips senden.

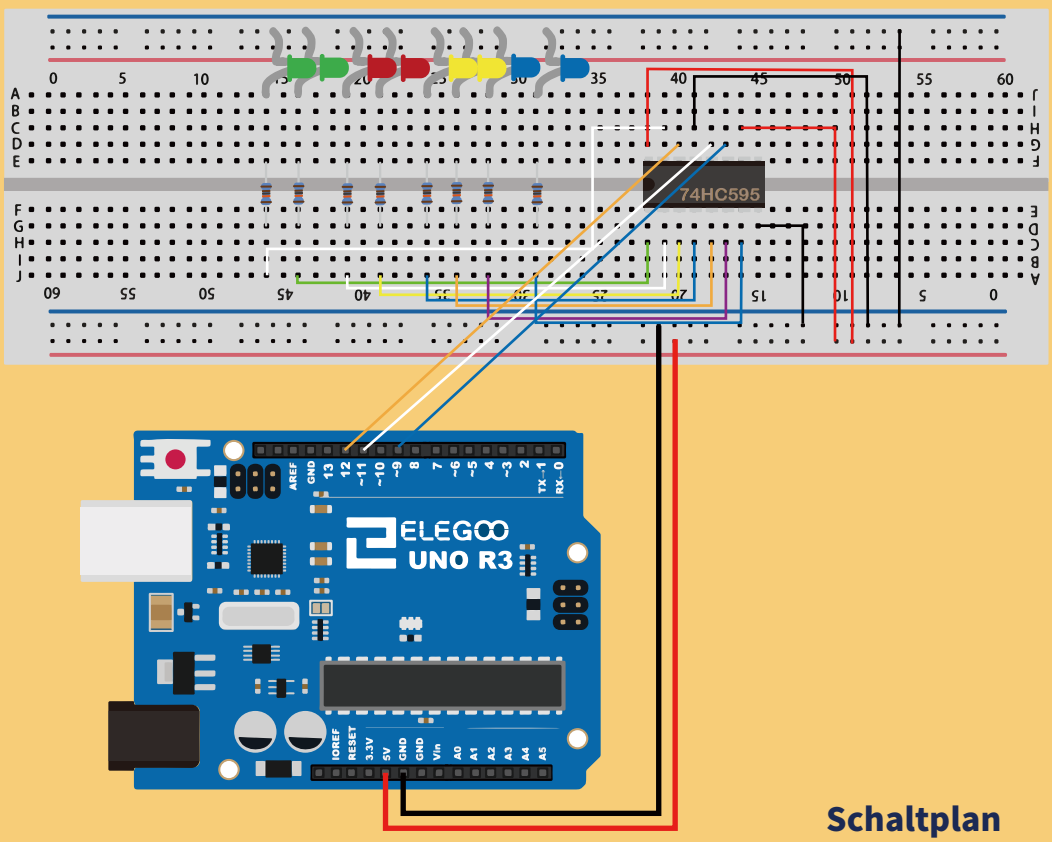
An den Clock-Pin müssen acht Pulse geschickt werden. Bei jedem Puls wird der Data- Pin überprüft. Wenn der Data-Pin HIGH geschaltet ist, wird eine 1 in das Schieberegister an die jeweilige Stelle geschrieben. Wenn der Pin LOW ist, wird eine 0 geschrieben. Nachdem alle 8 Pulse empfangen wurden, muss der „Latch“-Pin aktiviert werden und die acht Werte werden im Register gespeichert. Das ist unbedingt notwendig, da sonst die Daten an die falschen Stellen geschrieben werden könnten und die falschen LEDs aufleuchten würden.



Der Chip hat zusätzlich einen „Output Enable “ (OE)-Pin, durch den alle Ausgänge auf einmal an bzw ausgeschaltet werden können. Sie könnten diesen Pin an einen PWM-fähigen Pin des Boards anschließen und mit der analogWrite-Funktion die Helligkeit der LEDs kontrollieren. Der OE-Pin ist aktiv LOW, also muss er mit GND verbunden werden.



Verbindungsschema



Schaltplan

- Da wir insgesamt acht LEDs und acht Widerstände anschließen müssen, müssen einige Verbindungen hergestellt werden.
- Am einfachsten ist es, wenn man den 74HC595-Chip zuerst einsetzt, da so ziemlich alles mit ihm verbunden wird. Setzen sie den Chip so ein, dass die kleine Einkerbung zum oberen Ende (der kurzen Seite) des Breadboards zeigt. Der Pin 1 befindet sich am Chip unter dieser Einkerbung.
- Der digitale Pin 12 des UNOs geht an Pin 14 des Schieberegisters,
  der digitale Pin 11 des UNOs geht an PIN 12 des Schieberegisters
  und der digitale Pin 9 des UNOs geht an PIN 11 des Schieberegisters.
- Alle Ausgänge außer einem sind auf der unteren Seite des Chips. Daher werden wir die LEDs der Einfachheit halber auf der linken Seite platzieren.
- Nach dem Chip müssen die Widerstände an ihre Stelle gesetzt werden. Sie müssen aufpassen, dass die Kontakte der verschiedenen Widerstände sich nicht berühren. Sie sollten dies nocheinmal überprüfen, bevor Sie das Board später mit Strom versorgen. Wenn es zu schwer ist die Widerstände einzusetzen, ohne dass sich dessen Kontakte berühren, kann es helfen die Kontakte zu kürzen, sodass die Widerstände tiefer am Breadboard anliegen.
- Als nächstes platzieren Sie die LEDs auf dem Breadboard. Die langen positiven Enden müssen alle in Richtung Chip zeigen.
- Schließen Sie die Jumper Kabel wie im obigen Bild gezeigt an. Vergessen Sie nicht das Kabel, das vom Pin 8 des Chips zum GND-Anschluss des Breadboards führt.
- Laden Sie den Sketch hoch und probieren Sie es aus. Alle LEDs sollten nacheinander aufleuchten, bis alle an sind. Danach schalten sich alle LEDs aus und das Muster wiederholt sich anschließend.

Code

- Nach dem Verbinden der Komponenten öffnen Sie bitte den Sketch im Code-Ordner unter „Eight LED with 74HC595“ und laden ihn auf Ihr UNO Board hoch. Bei Fragen zum Hochladen eines Sketches schauen Sie sich bitte Lektion 5 in Teil 1
- Als erstes im Sketch bestimmen wir die drei Pins, die wir benutzen werden. Dabei handelt es sich um die digitalen Ausgänge des UNO Boards, die mit den Latch-, Clock- und Data-Pins vom 74HC595 verbunden werden.

```
int latchPin = 11;
int clockPin = 9;
int dataPin = 12;
```

- Als nächstes wird eine Variable mit dem Namen „leds“ gesetzt. Diese Variable werden wir dazu nutzen, um zu speichern welche LEDs gerade ein- und welche ausgeschaltet sind. Variablen des Typs „byte“ repräsentieren acht Bit große Zahlen. Jeder Bit kann dabei an oder aus sein. Also perfekt für die Zustände unserer LEDs.

```
byte leds = 0;
```

- Die setup-Funktion bestimmt nur, dass wir unsere drei Pins als Ausgänge nutzen.
- Die loop-Funktion schaltet zu Beginn erst einmal alle LEDs aus, indem die leds- Variable auf 0 gesetzt wird. Dann wird „updateShiftRegister“ aufgerufen, dass den Wert der leds-Variable an das Schieberegister sendet, was somit alle LEDs ausschaltet. Um die Funktionsweise der updateShiftRegister-Funktion kümmern wir uns später.

```
void setup()
{
pinMode(latchPin, OUTPUT);
pinMode(dataPin,
OUTPUT);
pinMode(clockPin,
OUTPUT);
}
```

- Die loop-Funktion wird durch eine 0,5 sekündige Pause unterbrochen und beginnt dann in einer for-Schleife in Durchgängen von 0 bis 7 zu zählen, wobei die Variable i die aktuelle Zahl repräsentiert. Bei jedem Durchgang wird die Funktion„bitSet“ aufgerufen, die die Zahl in der leds-Variable ändert. Danach wird wieder updateShiftRegister aufgerufen, damit der Chip die LEDs anpasst. Dann gibt es wieder eine 0,5 sekündige Pause und danach folgt der nächste Durchgang.

- Die Funktion updateShiftRegister setzt zuerst den latch-Pin auf LOW, und ruft dann die Funktion „shiftOut“ auf, bevor sie den latch-Pin wieder auf HIGH umschaltet.

- Die shiftOut-Funktion benötigt vier Parameter. Die ersten beiden sind die Pins, die an Data und Clock angeschlossen sind. Der dritte Parameter bestimmt an welchem Ende wir starten wollen. Wir starten bei dem Bit ganz rechts, der „Least Significant Bit“ (LSB) genannt wird.

- Der letzte Parameter sind die eigentlichen Daten, die wir an das Schieberegister übertragen wollen. Bei uns ist der letzte Parameter die leds-Variable.

```
void loop()
{
leds = 0;
updateShiftRegister();
delay(500);
for (int i = 0; i < 8; i++)
{
bitSet(leds, i);
updateShiftRegister();
delay(500);
}
}
```

- Wenn sie stattdessen lieber eine LED aus- statt einschalten möchten, müssen sie eine ähnliche Arduino-Funktion, die „bitClear“-Funktion, mit der leds-Variable als Parameter aufrufen. Diese Funktion setzt den entsprechenden Bit auf 0. Danach müssen Sie nur noch die updateShiftRegister-Funktion aufrufen.

```
void updateShiftRegister()
{
digitalWrite(latchPin, LOW);
shiftOut(dataPin, clockPin,
LSBFIRST, leds);
digitalWrite(latchPin, HIGH);
}
```