TUG

Elektronik und computerunterstützte Messtechnik
LU PHY.M20 (SS 2022)

Übung 5: Mikrocontroller
1. Ampelschaltung
2. Reaktionsspiel

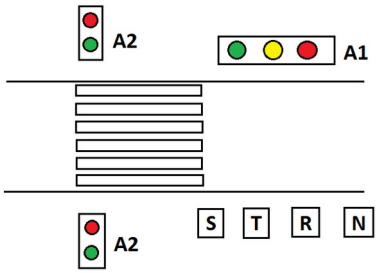
Abgabe der Vorbereitung: bis Freitag, 17. Juni 2022, 12:00 Uhr im TeachCenter

Praktikumstermin: KW25 (22.06 bis 24.06.2020)

Anmerkung: Zur Vorbereitung können eigene Arduinoprogramme mit dem Simulator ‚SimulIDE' getestet werden. Dieser ist im TeachCenter unter ‚Laborübung' verlinkt.

Aufgabe 1: Ampelschaltung mit Fußgängerübergang

Das Ziel der Übung ist die Simulation einer geregelten Ampelschaltung an einem Fußgängerübergang (s. Abb. rechts) mit Hilfe eines Arduino UNO Boards. Programmiert wird in der Arduino Entwicklungsumgebung.

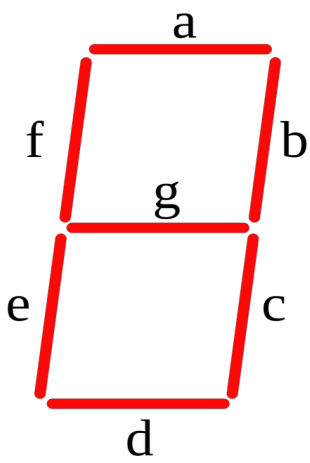


Grundzustand: Der Verkehr fließt auf der Straße, Ampel A1 ist auf Grün und Ampel A2 ist auf Rot geschaltet.

Schaltzustand: Die Taste „S" wird gedrückt und die Ampeln wechseln das Signal. Dabei sollen die in Österreich üblichen Ampel Schaltregeln eingehalten werden. Nach 5s Grünphase der Ampel A2 soll in den Grundzustand zurückgewechselt werden.

Nachtfunktion: Durch einen Sensor „N" (simuliert durch einen Photowiderstand/LDR) soll die Ampelschaltung beim Unterschreiten eines bestimmten Schwellwertes in einen Nachtmodus wechseln. Dabei soll die Ampel A1 auf gelb blinkend umschalten und die Ampel A2 ausgeschaltet werden.

Aufgabe 2: Reaktionsspiel



In diesem Teil der Übung wird ein einfaches Reaktionsspiel mithilfe einer Siebensegmentanzeige und zweie Tastern realisiert.

Es soll immer nur ein Segment leuchten. Das leuchtende Segment soll entlang einer 8-er Kurve laufen, also a → b → g → e → d → c → g → f bzw. umgekehrt. Die Leuchtdauer der einzelnen Segmente beträgt zu Beginn des Spieles 500 ms und wird pro erzieltem Punkt kürzer (zB. T_{on} = 500 ms / ($n_{Gesamtpunke}$ +1)).

Wenn nun das Segment (a / d) leutet muss Spieler (1 / 2) seinen Taster drücken. Geschieht dies innerhalb der Leuchtdauer vom entsprechenden Segment erhält der Spieler einen Punkt und der Punktestand wird per UART ausgegeben. Mit jedem erziehlten Punkt läuft das leuchtende Segment schneller im Kreis. Wird das Zeitfenster verpasst oder zu einem ganz falschen Zeitpunkt gedrückt ändert der Leuchtpunkt seine Richtung. Sind insgesamt 10 Punkte erreicht ist das Spiel vorbei, alle Segmente blinken nun im Gleichtakt und der Endpunktestand wird nochmals per UART ausgegeben. Werden nun beide Taster zeitgleich gedrückt beginnt das Spiel erneut.

Zusatz: Wird das Zeitfenster zum drücken des Tasters verpasst verliert der Spieler einen Punkt. Die Leuchtdauer des Segmentes verändert sich nicht und auch die erzielten Gesamtpunkte werden dadurch nicht beeinflusst. (Bei einem Fehler im Spiel wäre der Endpunktestand zB. 3:6).

Zusatz: Der Punktestand soll in auf einem monochromen LC-Display angezeigt werden (HD44780). Hierfür steht die Library ‚LiquidCrystal' im Arduino Library Manager zur Verfügung. Das Display soll im 6-Pin Modus betrieben werden.

Informationen zur Library:
https://www.arduino.cc/reference/en/libraries/liquidcrystal/
Minmales bsp. Programm der Library ist hier enthalten:
https://www.arduino.cc/reference/en/libraries/liquidcrystal/liquidcrystal/

Vorbereitung

Für beide Aufgaben ist ein Programmablaufplan (PAP) nach DIN 66001 anzufertigen. Hierzu stehen dieses Mal verschieden digitale Helfer zu Verfügung, wie zum Beispiel die von uns empfohlenen Programme PAPDesigner oder yEd von YWORKS. Es ist darauf zu achten dass die Reihenfolge der verschiedenen Abfragen bzw. Zustände auch die Prioritäten/Hierarchien im Programm beeinflussen.

Die Zusatzaufgaben müssen nicht in der Vorbereitung berücksichtigt werden und werden nur bei ausreichender Zeit im Labor durchgeführt.

Des Weiteren müssen zwei Prinzipskizzen für Aufgabe 1 entworfen werde. Diese sind jedoch mit der Hand & Lineal zu zeichnen und wie üblich sind alle Anschlüsse zu bezeichnen.

1. Skizze Vorwiderstände:

Bei der Dimensionierung der Vorwiderstände für die LEDs ist zu beachten dass der Strom die 10 mA nicht überschreitet.

2. Skizze Nachtsensor:

Zur Realisierung des Nachtsensors muss eine geeignete Beschaltung des LDR skizziert werden, so dass eine lichtabhängige Spannung gemessen werden kann.

Programmierung:

Programmiert wird in der Arduino Entwicklungsumgebung. Die Programmiersprache ist C/C++. Eine „Language Reference“ für Arduino ist unter nachfolgendem Link verfügbar. https://www.arduino.cc/reference/en

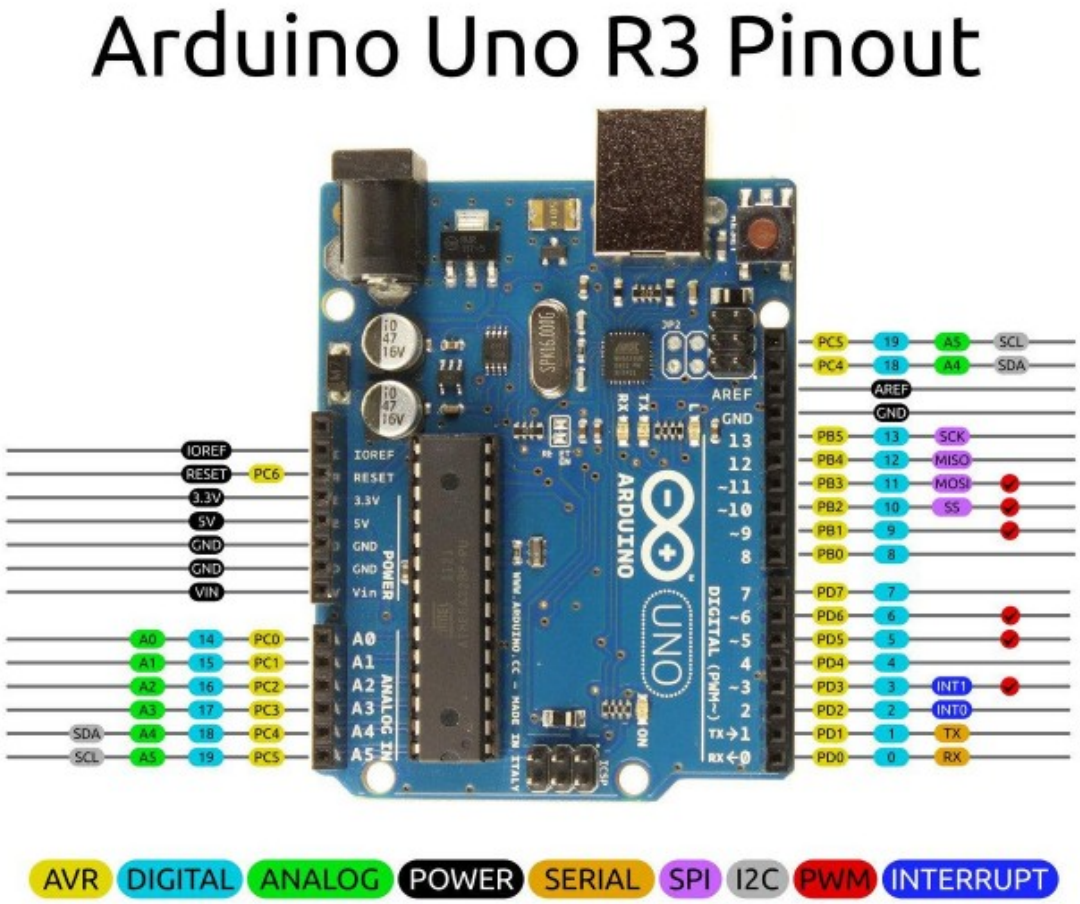
Für die Programmierung der Ampelsteuerung ist es empfehlenswerteinen der zwei verfügbaren Interrupt Pins für den Taster „R“ zu verwenden. Die „delay“ Funktion kann verwendet werden solange die Funktion der Ampelsteuerung nicht eingeschränkt wird.

Coding Standard:

Zur besseren Lesbarkeit des Programms ist es notwendig einen „Coding Standard“ festzulegen der im gesamten Programm angewendet wird. Zum Beispiel:

- Konstanten ALL_UPPERCASE_WITH_UNDERSCORES
- Variablen firstWordLowerCaseRestCapitalizedWithoutUnderscores
- Funktionen firstWordLowerCaseRestCapitalizedEndsWithUnderscore_
- Kommentare //comments are necessary

Auf eine sinnvolle und AUSREICHENDE Kommentierung des erstellten Codes ist zu achten!



Ein paar nützliche Funktionen in der Arduino-Umgebung:

erzeugen des PWM-Signals	analogWrite (pin, val);
Seriellen Port verwenden	Serial.begin(9600);
Daten (Text) am Port schreiben	Serial.print("Hallo World");
Daten (Zahlen) am Port schreiben mit allen Nachkommerstellen (bei Float/ Double)	Serial.print(Variable, DEC);
Wartefunktionen	delay(val); val in ms delayMicroseconds(val); in µs
Einstellen der Pins	pinMode(pin, Modus);
Pin setzen	digitalWrite(pin, Zustand);