Reaktionsspiel

**AP-Nummer:** 3

**AP-Titel:** Start

**AP-Verantwortlicher:** Ioannis Christodoulakis, Thomas Ammann   
**Datum:** 29. Mai 2017

**Raumtemperatur:** 24.2 °C  
**Revision:** 01

Inhaltsverzeichnis

[1. Zusammenfassung 2](#_Toc483391593)

[2. Aufgabenstellung 2](#_Toc483391594)

[2.1 Schnittstelleninformationen 2](#_Toc483391595)

[3. Begriffsverzeichnis 2](#_Toc483391596)

[4. Struktogramm 2](#_Toc483391597)

[5. Programm – Code 3](#_Toc483391598)

[6. Verwendete Programme 4](#_Toc483391599)

[7. Schlussfolgerung 4](#_Toc483391600)

[8. Identifikation 4](#_Toc483391601)

# 1. Zusammenfassung

Es wurde ein Software-Programm entwickelt, welches ein Start-Signal benötigt. Wenn das Programm startet, blinken zwei LEDs drei Mal.

# 2. Aufgabenstellung

Zum Starten wird ein Start-Signal benötigt. Wenn das Programm startet soll ein optisches Start-Signal auftreten. Dieses Blinkt drei Mal.

2.1 Schnittstelleninformationen  
Eingänge: Start-Signal (aktiv LOW)  
Ausgänge: Beginn-Signal (Spieler1 und Spieler2)

# 3. Begriffsverzeichnis

|  |  |  |  |
| --- | --- | --- | --- |
| Eingabe | Speicher-Name | Pin-Name | Pin |
| Start-Signal | St\_Si | S\_S | 3 |

|  |  |  |  |
| --- | --- | --- | --- |
| Ausgabe | Speicher-Name | Pin-Name | Pin |
| Optische Meldung1 | Op\_Me1 | O\_M1 | 8 |
| Optische Meldung2 | Op\_Me2 | O\_M2 | 9 |

# 4. Struktogramm

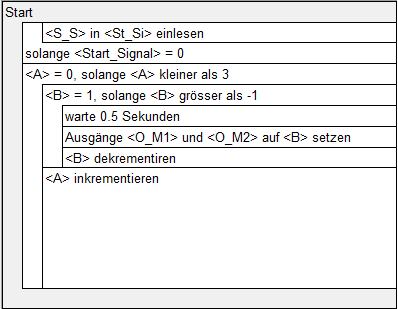


Abbildung , Struktogramm

# 5. Programm–Code

Um eine Variable zu erstellen, müssen wir diese zuerst deklarieren. Wir haben für die Variablen  
Start-Signal, A und B den Datentyp **char** gewählt, da die Variablen nicht mehr Speicherplatz benötigen (1 Byte).

**char** St\_Si, // Variablen

A,

B;

Die Pin-Nummern der Ein- und Ausgänge sind den Pin-Namen gespeichert. Diese sind unveränderbar.

**const char** S\_S = 3, // Konstanten

O\_M1 = 8,

O\_M2 = 9;

**setup()** wird einmalig beim Einschalten des Arduino ausgeführt. Dort definieren wir mit der Funktion **pinMode()**, ob die digitalen Pins Ein- oder Ausgänge sind.

**void setup()** { // I/O

**pinMode** (S\_S, INPUT); // Eingaenge

**pinMode** (O\_M1, OUTPUT); // Ausgaenge

**pinMode** (O\_M2, OUTPUT);

}

**loop()** wird wiederholt ausgeführt. In der Funktion wird das Hauptprogramm ausgeführt.   
Die **do-while**-Schleife liest den Digitalwert am Eingang «S\_S» ein und speichert ihn in «St\_Si». Danach wartet das Programm 50ms, bevor es überprüft, ob «St\_Si» 0 entspricht. Trifft dies zu, wird die Schleife wiederholt.

**void loop()** { // Hauptprogramm

**do** { // Start-Signal einlesen

St\_Si = **digitalRead**(S\_S);

**delay** (50);

} **while** (St\_Si == 0);

Die erste **for**-Schleife wird dreimal durchgeführt. Die zweite Schleife wird zweimal durchgeführt. Im erstem Durchlauf werden die Ausgänge «O\_M1» und «O\_M2» eingeschaltet. Im zweiten Durchlauf werden diese ausgeschaltet. Die **delay()**–Funktion lässt den Arduino die Ausführung des Programms für eine gewisse Zeitspanne anhalten. Damit wird der Logikwechsel der Ausgänge verlangsamt.

**for** (A = 0; A < 3; A++) { // Optische Meldung

**for** (B = 1; B > -1; B--) {

**delay** (500);

**digitalWrite** (O\_M1, B);

**digitalWrite** (O\_M2, B);

}

}

}

# 6. Verwendete Programme

* Arduino Software
* Struktogrammer

# 7. Schlussfolgerung

Das entwickelte Software-Programm erfüllt die Anforderungen der Aufgabenstellung.  
(2. Aufgabenstellung/Seite 2).

# 8. Identifikation

Ort, Datum: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Unterschrift: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_­­\_\_ Unterschrift: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_­­\_\_  
 Ioannis Christodoulakis Thomas Ammann