Reaktionsspiel

**AP-Nummer:** 8

**AP-Titel:** Neustart

**AP-Verantwortlicher:** Ioannis Christodoulakis, Thomas Ammann   
**Datum:** 24. Mai 2017

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

Inhaltsverzeichnis

[1. Zusammenfassung 2](#_Toc483395672)

[2. Aufgabenstellung 2](#_Toc483395673)

[2.1 Schnittstelleninformationen 2](#_Toc483395674)

[3. Begriffsverzeichnis 2](#_Toc483395675)

[4. Struktogramm 2](#_Toc483395676)

[5. Programm–Code 3](#_Toc483395677)

[6. Verwendete Programme 4](#_Toc483395678)

[7. Schlussfolgerung 4](#_Toc483395679)

[8. Identifikation 5](#_Toc483395680)

# 1. Zusammenfassung

Es wurde ein Software-Programm entwickelt, welches einen Neustart durchführt. -

# 2. Aufgabenstellung

Nachdem das Spiel beendet wurde, kann das dieses mit einem Start-Signal zurückgesetzt werden.

2.1 Schnittstelleninformationen

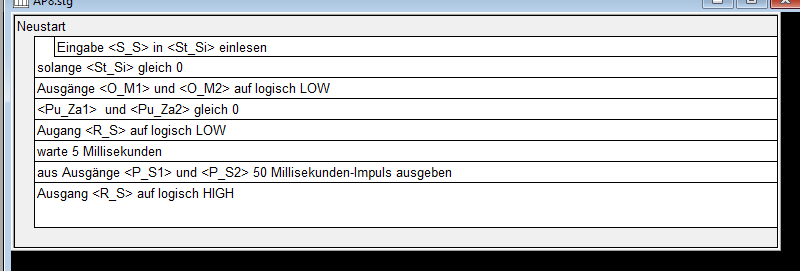
|  |  |
| --- | --- |
| Eingabe | Ausgabe |
| Punktzahl1 | Reset-Signal (aktiv LOW) |
| Punktzahl2 | Punktzahl-Signal1 |
|  | Punktzahl-Signal2 |
|  | Optische Meldung1 |
|  | Optische Meldung2 |

# 3. Begriffsverzeichnis

|  |  |  |  |
| --- | --- | --- | --- |
| Ausgabe | Speicher-Name | Pin-Name | Pin |
| Punktzahl-Signal1 | Pu\_Si1 | P\_S1 | 6 |
| Punktzahl-Signal2 | Pu\_Si2 | P\_S2 | 7 |
| Optische Meldung1 | Op\_Me1 | O\_M1 | 8 |
| Optische Meldung2 | Op\_Me2 | O\_M2 | 9 |
| Reset-Signal (aktiv LOW) | Re\_Si | R\_S | 10 |

|  |  |
| --- | --- |
| Variable | Speicher-Name |
| Punktzahl1 | Pu\_Za1 |
| Punktzahl2 | Pu\_Za2 |

# 4. Struktogramm



# 5. Programm–Code

Um eine Variable zu erstellen, müssen wir diese zuerst deklarieren. Wir haben für die Variablen  
Start-Signal, Punktzahl1 und B den Datentyp **char** gewählt, da die Variablen nicht mehr Speicherplatz benötigen (1 Byte).

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

Pu\_Za1,

Pu\_Za2;

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

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

P\_S1 = 6,

P\_S2 = 7,

O\_M1 = 8,

O\_M2 = 9,

R\_S = 10;

**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** (P\_S1, OUTPUT); // Ausgaenge

**pinMode** (P\_S2, OUTPUT);

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

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

**pinMode** (R\_S, OUTPUT);

}

**loop()** wird wiederholt ausgeführt. In der Funktion wird das Hauptprogramm ausgeführt.   
Der Inhalt der **do-while**-Schleife liest den Digitalwert am Eingang «S\_S» ein und speichert ihn in «St\_Si». Dies wird solange ausgeführt, bis «St\_Si» 0 entspricht. Danach werden die optischen Meldungen ausgeschaltet und die Punktzahlen und die Anzeigen auf 0 zurückgesetzt.

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

**do**{ // Auswertung

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

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

**digitalWrite** (O\_M1, LOW); // Optische Meldung ausschalten

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

Pu\_Za1 = 0; // Punktzahl1 zurücksetzten

Pu\_Za2 = 0; // Punktzahl2 zurücksetzten

**DigitalWrite** (R\_S, LOW); // Anzeigen zurücksetzten

**delay** (5);

**digitalWrite** (P\_S1, HIGH);

**digitalWrite** (P\_S2, HIGH);

**delay** (50);

**digitalWrite** (P\_S1, LOW);

**digitalWrite** (P\_S2, LOW);

**digitalWrite** (R\_S, HIGH);

}

# 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