SuperMario says ...

Inhaltsverzeichnis

[Einleitung 1](#__RefHeading__197_443978008)

[Spielprinzip Senso 2](#__RefHeading__199_443978008)

[Die Entwicklungsumgebung 2](#__RefHeading__201_443978008)

[MCU 8051 IDE 2](#__RefHeading__203_443978008)

[Beschreibung der IDE 2](#__RefHeading__205_443978008)

[Eingesetzte Simulierte Hardware 2](#__RefHeading__207_443978008)

[GIT 2](#__RefHeading__209_443978008)

[Das Programm 2](#__RefHeading__211_443978008)

[Unterprogramme 2](#__RefHeading__213_443978008)

[Hauptspielschleife 2](#__RefHeading__215_443978008)

[Start / Stop / Neustart 3](#__RefHeading__217_443978008)

[Pseudozufallszahlen Generator 3](#__RefHeading__219_443978008)

[Feldsequenz-Generator 3](#__RefHeading__221_443978008)

[Feldsequenz visualisieren 3](#__RefHeading__223_443978008)

[Benutzereingabe 3](#__RefHeading__225_443978008)

[Vergleichen der benutzereingabe mit der sequenz 3](#__RefHeading__227_443978008)

[Korrektheits-Anzeige 3](#__RefHeading__229_443978008)

[Implementierung 3](#__RefHeading__231_443978008)

[Benutzertest 3](#__RefHeading__233_443978008)

[Fazit 3](#__RefHeading__235_443978008)

# Einleitung

Dieses Dokument beschreibt das Projekt "SuperMario says ..." im Fach Systemnahe Programmierung an der DHBW Karlsruhe im Sommersemester '16 bei Professor Lausen. Ziel des Projektes soll es sein, das Spiel Senso[[1]](#footnote-2) in Assembler nachzuprogrammieren. Der Code soll auf einem Simulator des 8051-Mikroprozessors[[2]](#footnote-3) und simulierter Hardware laufen. Hierzu wird die MCU 8051 IDE[[3]](#footnote-4) genutzt.

Entwickelt wird das Projekt von den Studenten Jasper Bröker, Mario Kaiser und Dominik Schaufelberger.

# Spielprinzip Senso

# Die Entwicklungsumgebung

## MCU 8051 IDE

Beschreibung der IDE

Eingesetzte Simulierte Hardware

## GIT

# Das Programm

*SuperMario says ...* nutzt die bereits genannte simulierte Hardware zur Benutzerinteraktion. Über das Keypad kann der Spieler das Programm starten, stoppen und neustarten. Sobald das Spiel gestartet wird wird dem Benutzer eine zufällige Folge von Feldern auf der LED Matrix angezeigt. Die Matrix ist hierbei, wie in (ABBILDUNG EINFÜGEN) zu sehen, in 4 Felder unterteilt. In einem aktiven Feld werden alle LEDs zum Leuchten gebracht. Sobald die Sequenz beendet wurde, muss der Spieler über das Keypad die zu den entsprechenden Feldern zugewiesenen Schalter drücken und die zuvor angezeigte Sequenz nachstellen.

Das Programm ist in eine Menge von Unterprogrammen unterteilt. Diese Unterpgrogramme führen bestimmte Routinen aus wie zum Beispiel das Generieren von Pseudozufallszahlen. Die Aufgaben der jeweiligen Unterprogramme werden im Folgenden erklärt.

Angabe der Benutzung von Speicher und Ports

## Unterprogramme

Hauptspielschleife

Die Hauptspielschleife steuert den kompletten Ablauf des Programms und ruft die jeweiligen anderen Unterprogramme auf. Wenn das Programm gestartet wird, wird gleichzeitig ein Timer des Mikrocontrollers gestartet. Über die *Start / Stop / Neustart* Routine kann dann das Spiel begonnen werden. Wenn das Spiel gestartet wurde, wird der *Feldsequenz-Generator* aufgerufen. Dieser erzeugt eine zufällige Reihenfolge für das ansprechen der Felder. Nachdem die Sequenz vollständig generiert wurde, wird das Unterprogramm *Feldsequenz Visualisieren* ausgeführt und visualisiert dem Spieler die Sequenzreihenfolge. Wenn die Anzeige der Sequenz beendet ist, wird die Benutzereingabe erwartet. Diese wird dann auf Korrektheit geprüft. Der Spieler bekommt dann Rückmeldung ob seine eingegebene Sequenz korrekt war oder nicht.

Start / Stop / Neustart

Pseudozufallszahlen Generator

Wenn der Pseudozufallszahlen Generator aufgerufen wird, dann wird zuerst der aktuelle Wert des Timers abgerufen. Anhand dessen wird dann nach der Formel (FORMEL EINFÜGEN) die Pseudozufallszahlen erzeugt. Diese werden im Speicher abgelegt um sie später für den *Feldsequenz-Generator* zu verwenden.

Feldsequenz-Generator

Der Feldsequenz-Generator erzeugt aus den generierten Pseudozufallszahlen die für die Visualisierung benötigten Portbelegungen. Aus einer Pseudozufallszahl wird über eine Modulo-Operation der Index eines Feldes berechnet. Für jeden eindeutigen Index wird eine bestimmte eindeutige Portbelegung erzeugt. Diese Portbelegung wird wieder im Speicher abgelegt um diese in der *Feldsequenz Visualierung* zu nutzen.

Feldsequenz visualisieren

Das Unterprogramm zur Visualisierung

Benutzereingabe

Vergleichen der benutzereingabe mit der sequenz

Korrektheits-Anzeige

# Implementierung

# Benutzertest

# Fazit

1. https://de.wikipedia.org/wiki/Senso\_(Spiel) [↑](#footnote-ref-2)
2. https://en.wikipedia.org/wiki/Intel\_MCS-51 [↑](#footnote-ref-3)
3. https://en.wikipedia.org/wiki/MCU\_8051\_IDE [↑](#footnote-ref-4)