|  |
| --- |
| **Synthetisierung einer Aufzugsteuerung  für einen 8051 Micro Controller**  **Projektbericht**  für die Prüfung zum  Bachelor of Science  des Studienganges Angewandte Informatik  an der  Dualen Hochschule Baden-Württemberg Karlsruhe  von  Christian Verdion, Moritz Gabriel, Oliver Mahlke, Jan Kalweit  31.01.2015 |

Kurs TINF13B2

Erklärung

gemäß § 5 (2) der „Studien- und Prüfungsordnung DHBW Technik“ vom 18. Mai 2009.

Ich habe die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet.

Ort, Datum Unterschrift

Inhaltsverzeichnis

[1 Einleitung 5](#_Toc409770807)

[1.1 Aufgabenstellung 5](#_Toc409770808)

[2 Grundlagen 6](#_Toc409770809)

[2.1 Assembler 6](#_Toc409770810)

[2.2 8051 Micro Controller 6](#_Toc409770811)

[3 Konzept 8](#_Toc409770812)

[3.1 Architektur 8](#_Toc409770813)

[3.2 Programmablaufplan 8](#_Toc409770814)

[4 Implementierung 9](#_Toc409770815)

[4.1 Interrupt 9](#_Toc409770816)

[4.1.1 Timer Interrupt 9](#_Toc409770817)

[5 Zusammenfassung 13](#_Toc409770818)

[6 Literaturverzeichnis 14](#_Toc409770819)

**Abbildungsverzeichnis**

Es konnten keine Einträge für ein Abbildungsverzeichnis gefunden werden.

**Quelltextverzeichnis**

[Quelltext 1: „importModelsFromXML“ Funktion 13](#_Toc399327208)

[Quelltext 2: Beginn der „readXMLContent“ Funktion 14](#_Toc399327209)

[Quelltext 3: Einlesen der Attribute aus dem XML-Dokument 16](#_Toc399327210)

[Quelltext 4: Ende der „readXMLContext“ Funktion 17](#_Toc399327211)

[Quelltext 5: Hinzufügen eines Menüeintrags in JavaScript 19](#_Toc399327212)

[Quelltext 6: Die Ereignis-Methode für den Menüeintrag und deren Registrierung 20](#_Toc399327213)

[Quelltext 7: Methodenaufruf „doImportModelsXML“ im ModelImporter 21](#_Toc399327214)

[Quelltext 8: Das Objekt „Models" wird in der JavaScript-Umgebung angelegt 26](#_Toc399327215)

[Quelltext 9: Erhalten der ModelItems im Umkreis der „simulation\_center“ Position 26](#_Toc399327216)

[Quelltext 10: Beispiel einer JavaScript-Datei zur Simulation 27](#_Toc399327217)

# Einleitung

In nahezu allen Dingen unseres Alltags steckt mittlerweile eine ganze Menge Technik. Kleine Sensoren sorgen dafür, dass ein bestimmter Zustand der Umwelt erkannt wird. Motoren sorgen dafür, dass sich Dinge bewegen. Lichter geben Zustände oder Informationen für den Menschen lesbar aus. In Mitten dieser Komponente findet man immer einen Micro Controller.

Kleine Computer, die wenig Strom verbrauchen und effizient Berechnungen durchführen. Ohne diese Chips wären viele technische Geräte wie wir sie kennen gar nicht möglich. Aufgrund der geringen Größe, und ihrer nahezu universellen Einsetzbarkeit wäre an keine Alternative zu denken, die so leise und komfortable ihren Dienst verrichtet.

Ebenso wie die Größe dieser Bausteine sinkt, steigt auch unser Anspruch an sie. Immer mehr Funktionen müssen eingebaut werden. Mit mehr Funktionen und komplexeren Tätigkeiten muss die Geschwindigkeit steigen um zum Beispiel Echtzeitanforderungen einzuhalten, wenn es um eingebettete Systeme in Autos oder anderen kritische Aufgabenbereiche geht.

## Aufgabenstellung

Im Rahmen dieser Arbeit soll eine Aufzugsteuerung synthetisiert und Implementiert werden, die auf einen 8051 Micro Controller läuft.

# Grundlagen

## Assembler

Bei Assembler handelt es sich um eine hardwarenahe Programmiersprache. Das heißt, das geschriebene Programm ist für ein bestimmten Prozessor oder Chip-Architektur ausgelegt. Es kann dadurch möglich sein, dass ein und dasselbe Programm auf verschiedenen Prozessoren oder Chips nicht lauffähig ist.

In Assembler Programmiert man größtenteils mit einfachen Funktionen wie dem Addieren oder Subtrahieren zweier Werte. Auch Bit-weise Operationen wie „Und“, „Oder“ oder „Shift“ sind möglich. Auf weitere Methoden und deren Funktion wird später eingegangen.

Der entstehende Assembler-Code wird danach Kompiliert, also Übersetzt, um dann direkt auf dem Chip ausgeführt zu werden. Das sogenannte Kompilat, also das Übersetzte Programm, ist dann in Maschinensprache geschrieben und für einen Menschen nicht mehr zu lesen. Lediglich die Micro Controller, oder die Micro Controller einer bestimmten Architektur, für die das Programm übersetzt wurde können mit diesem Code etwas anfangen.

## 8051 Micro Controller

Der 8051 Micro Controller gehört zu der von Intel Micro Controller Familie MCS-51 und wurde im Laufe der 80er Jahre eingeführt. Bei ihm handelt es sich um einen 8-Bit Micro Controller, was bedeutet, dass er mit einer Wortlänge von 8-Bit arbeitet, und diese 8-Bit Wörter während eines Taktes verarbeiten kann.   
Heutzutage kommen immer noch viele Prozessoren mit 8-Bit Technologie zum Einsatz, so zum Beispiel bei einer Vielzahl von USB-Peripherie. Diese werden aber nichtmehr alle von Intel hergestellt. Neben Derivaten von Firmen wie Texas Instruments und Motorola gibt es natürlich auch andere Micro Controller, die auf 8-Bit Technologie setzten und gerade für eingebettete Systeme sehr beliebt sind.

Ein 8051 Micro Controller enthält eine CPU, also eine Recheneinheit, die über einen Bus mit anderen Bauteilen verbunden ist. Für dieses Projekt relevant sind zum einen die 4 Ein-/ Ausgabe Ports, die über jeweils 8 Pins oder Bit verfügen und die Möglichkeit externe Interrupts auszulösen.

Des Weiteren verfügt er noch über Anschlüsse einen externen Takt für die Timer anzuschließen, wobei für diesen Anwendungsbereich der interne Timer ausreichend Genauigkeit liefert und eine Datenleitung um auf einen externen zusätzlichen Ram zuzugreifen.

<http://www.mikroe.com/chapters/view/65/chapter-2-8051-microcontroller-architecture/>

<http://de.wikipedia.org/wiki/Intel_MCS-51>

# Konzept

## Architektur

Hier steht dann was wo angeschlossen ist und welche Voraussetzungen getroffen wurden.

## Programmablaufplan

Hier werden die Programmablaufpläne sein und eine Erklärung dazu.

# Implementierung

Hier stehen grundlegende Dinge zur Implementierung und alles bis zu den Interrupts

## Interrupt

Hier steht alles über den Sensor, der am externen Interrupt angeschlossen ist.

### Timer Interrupt

Zur Realisierung der Fahrstuhlsteuerung ist es zweckmäßig sowohl die Tasten zum Rufen des Fahrstuhls und zur Stockwerkwahl als auch die Sensoren der Tür, also die Lichtschranke und der Berührungssensor an der Tür, über Externe Interrupts an dem Micro Controller anzuschließen. Dies setzt allerdings voraus, dass genügend Eingänge für Externe Interrupts zur Verfügung stehen. Da der für diese Lösung eingesetzte 8051 lediglich über zwei Externe Interrupts verfügt, und die Zahl der Sensoren und Tasten deutlich größer ist, muss über so genanntes Polling gearbeitet werden. Das heißt, es wird immer wieder überprüft welche Werte anliegen.

Folgende Probleme können hierbei auftreten:

Systemauslastung:

Das System ist immer wieder damit beschäftigt die anliegenden Eingänge zu Überprüfen, selbst wenn dieser Vorgang nicht nötig wäre, da die Eingänge sich nicht verändert haben. Es kann sogar vorkommen, dass, wenn keine anderen Befehle ausgeführt werden müssen und der Polling-Aufruf explizit im Code steht eine CPU Auslastung von 100% herrscht.

Ereignisdauer zu kurz:

Wird der Aufruf explizit im Code ausgeführt, so kann es sein, dass durch einen externen Interrupt oder einem zu großen Abschnitt zwischen 2 Aufrufen der Polling Routine das Eingangssignal, wie etwa das Drücken der Taste zum Rufen des Aufzugs, zu kurz anliegt und somit der Tastendruck durch die Logik nicht berücksichtigt wird.

Diesen Problemen kann man allerdings dadurch entgegenwirken, indem die Polling Routine nach Ablauf einer bestimmten Zeit von einem Timer Interrupt ausgeführt wird. Hierbei empfiehlt es sich die Verwendung von Intervallen zwischen 5ms und 20ms. Ist der Intervall zu groß, so kann es sein, dass ein kurzer Tastendruck nicht erkannt wird. Ist der Intervall zu klein, so kann es sein, dass die übrigen Befehle nicht schnell genug ausgeführt werden.

Durch den Aufruf durch einen Timer Interrupt ist es sogar möglich innerhalb eines andern Interrupts den Timer Interrupt auszuführen, wodurch ungeplante, längere Pausen vermieden werden können. Außerdem bleibt die Prozessorauslastung relativ niedrig, sofern das Programm derzeit keinen Befehl ausführen muss.

Es empfiehlt sich die Tasten von den Türsensoren zu trennen und da der Micro Controller über 2 Timer Interrupts verfügt werden diese in 2 verschiedenen Timer Interrupt Service Routinen überprüft die durch 2 verschiedene Timer Interrupts ausgelöst werden.

Der 8051 Micro Controller verfügt über 2 Timer, Timer0 und Timer1.

Die Timer zählen Rückwärts und sind in 2 hälften unterteilt. So gibt für den Timer0 zum einen TL0, welcher für Timer0\_Low steht und von einer 8-Bit Zahl dargestellt wird und zum anderen TH0, welcher für Timer0\_High steht. Dieser stellt auch eine 8-Bit Zahl dar. Beide Timer verfügen über 2 Modi .Im Modus 1 werden beide 8-Bit Zahlen zusammen genommen und stellen somit einen 16-bit Zähler dar, bei dem TH0 immer dann reduziert wird, sobald TL0 überläuft. Hierbei muss allerdings der Timer bei jedem Überlauf manuell neu initialisiert werden.

Alternativ lässt sich Modus 2 nutzen, bei dem lediglich eine 8-Bit Zahl im TL0 runtergezählt wird. Dieser Modus bringt den Vorteil, dass bei einem Überlauf neben der Interrupt Service Routine TL0 automatisch mit dem Wert aus TH0 initialisiert wird.

Beide Modi stehen sowohl für Timer0 als auch Timer1 zur Verfügung. Die Namenskonventionen sind für Timer1 analog.

Der 8051 wird in der Regel mit 12MHz betrieben und zum Ausführen eines Befehls werden mindestens 12 Maschienenzyklen benötigt. Das bedeutet, dass nach etwa 1 Micro Sekunde ein Befehl abgearbeitet sein sollte. In einer Millisekunde werden also 100 Befehle abgearbeitet.

Möchte man nun 5 Millisekunden abstände, so muss der Timer von 1000 auf 0 Zählen. Dafür reichen allerdings die 8-Bit oder 256 Werte nicht aus.

Aus diesem Grund ist es nötig die Werte auf beide Timer hälften aufzuteilen, und den Timer im Modus 1 zu betreiben, sodass (TL0+1)\*(TH0+1) = 500.

Im Falle des Timer0 für die Türsensoren ergibt sich also folgende Rechnung: (249+1)\*(1+1) = 500

Aus diesem Grund hat man sich dafür entschlossen für die Türsensoren Timer0 in Modus2, TL0 mit 249 und TH0 mit 1 zu initialisieren.

Der Timer für die Tasten wird Timer1 sein, der auch in Modus 1 initialisiert wird. Die WerteTL1 und TH1 werden mit 249 und 3 initialisiert.

Die Wahl der Timer 0 und 1 ist auch dadurch begründet, dass die Interrupts in einer bestimmten Reihenfolge ausgeführt werden, sollten mehrere gleichzeitig auftreten.  
So ist es wichtiger dass die Tür wieder auf geht, sollte der Türsensor berührt oder die Lichtschranke durchbrochen werden, als dass der Knopfdruck erkannt wird.

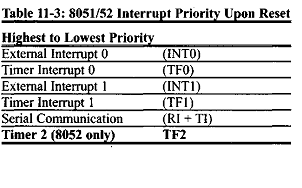


Abbildung 1: Interrupt Priorität (1)

Auch hierfür genügen die 256 Werte des TL1 nicht, sodass nicht auf das Auto Reload des Modus1 Zurückgegriffen werden kann.

Also müssen in beiden Fällen die Timer im Laufe der Service Routine neu initialisiert werden.

Um einen Wert von 500 zu erreichen genügt es TL1 mit 250 und TH1 mit 1 zu initialisieren, da dies ein Zählen der gewünschten 500 Werte bewerkstelligt.

(8051 arbeitet mit 12mHz. 1 Befehl benötigt in der Regel 12 oder mehr Takte -> Man kann etwa davon ausgehen 1 Befehl pro Micro Sekunde -> 100 Befehle in einer Millisekunde)

Also muss der Zähler im einen Fall bis 500 Zählen und im anderen bis 1.000!

# Zusammenfassung

# Literaturverzeichnis

1. **http://mohitjoshi999.files.wordpress.com/2009/08/080409\_0748\_interruptpr1.png.** [Online] [Zitat vom: 08. 01 2015.]

2. **http://www.mikrocontroller.net/articles/8051\_Timer\_0/1.** [Online] [Zitat vom: 09. 01 2015.]