|  |
| --- |
| **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](#_Toc408554678)

[1.1 Aufgabenstellung 6](#_Toc408554679)

[2 Grundlagen 7](#_Toc408554680)

[2.1 Assembler 7](#_Toc408554681)

[2.2 8051 Micro Controller 8](#_Toc408554682)

[3 Konzept 9](#_Toc408554683)

[3.1 Interrupt 9](#_Toc408554684)

[3.2 Timer Interrupt 9](#_Toc408554685)

[4 Implementierung 10](#_Toc408554686)

[5 Zusammenfassung 11](#_Toc408554687)

[6 Literaturverzeichnis 12](#_Toc408554688)

**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

Das Konzept, virtuelle Räume zu erstellen, gibt es schon fast so lange, wie es Computerspiele gibt. Annäherungsweise kann man sogar bei dem Spielfeld des einfachen Pong[[1]](#footnote-1) von einer virtuellen Umgebung sprechen. Diese Umgebungen wurden mit der Zeit immer aufwendiger und größer, sodass man mittlerweile durchaus von virtuellen Welten reden kann.

Inzwischen ist es sogar möglich mithilfe optischer und mechanischer Sensoren, die am Körper befestigt werden, die Bewegungen aufzuzeichnen und in eine virtuelle Umgebung zu projizieren. Diese Technologie wird irgendwann die „altertümliche“ Fortbewegung mittels Joystick oder Tastatur und Maus ersetzten. Bisher sind solche Sensoren allerdings zu kostspielig, sodass es kaum Endnutzer gibt, die solche Komponenten benutzen.  
Aber gerade durch Produkte wie die Kinect Kamera von Microsoft für die „Xbox One“ und „Xbox 360“ ist Motion Tracking schon fast in jedem Haushalt. Außerdem wird durch Sony´s „Project Morpheus“ und Oculus VR´s „Oculus Rift“ bald jeder die Möglichkeit haben, so noch dichter am virtuellen Geschehen teil zu haben. In Kombination dieser Komponenten wird es möglich sein, sich nicht nur in dieser virtuellen Welt zu befinden und sich dort umzusehen, sondern auch mit dieser Welt und deren Objekten zu interagieren und diese zu steuern. Derzeit ist dies vor allem im Bereich Computerspiele auf dem Vormarsch und weist immer größere Beliebtheit auf.

Im Bereich der Automatisierungs- und Anlagensteuerung ist dieses Thema bisher lediglich im hochpreisigen Segment etabliert. Das bietet uns somit sowohl eine gute Plattform für einen Einstieg, als auch ein gewisses Alleinstellungsmerkmal, um sich in diesem Bereich, mit einer für den Endnutzer bezahlbaren Softwarelösung, einen Namen zu machen.

Mit einem solchen System kann man Fabrikhallen virtuell erstellen noch bevor sie gebaut werden. Man kann diese virtuellen Hallen mit den Fertigungsanlagen füllen und so erste Besichtigungen durchführen und sogar Simulationen durchführen lassen, um die ideale Anordnung der Maschinen festzulegen.  
Ein weiteres Anwendungsszenario ist es eine schon existierende Fabrikanlage mit ihrem virtuellen Ebenbild zu verbinden. Somit ist man in der Lage, in Echtzeit, den derzeitigen Status der Maschinen zu betrachten, zu sehen Bauteil gerade in Bearbeitung ist oder ob es an einer Stelle Probleme gibt. So soll es auch möglich sein die Anlagensteuerung über eine virtuelle Umgebung durchzuführen.

## Aufgabenstellung

Im Rahmen dieser Arbeit soll das bereits bestehende, in C++ geschriebene, Projekt „High Fidelity“ um weitere Funktionen erweitert werden.

# Grundlagen

## Assembler

Bei einer Virtual Reality, zu Deutsch virtuellen Realität, handelt es sich um eine Umgebung, in der Objekte dargestellt werden, die es nicht zwangsläufig so auch in der realen Welt geben muss. In der Regel wird hier versucht einen Teil der Welt, zum Beispiel einen Raum, möglichst detailgetreu mit seinem Inhalt nachzubilden. Diese Nachbildungen sind rein digital und meist völlig von ihrer realen Vorlage losgelöst. Erstellt werden diese Nachbildungen zumeist als Mesh. Auf diesen Objekttyp setzt man in modernen, grafisch aufwändigeren Computerspielen. Hierbei handelt es sich um ein Gitternetz, welches wiederum aus Flächen mit mindestens 3 Kanten, auch Polygone genannt, zusammengesetzt wird. Dadurch ist es nämlich möglich die Flächen und Konturen der Objekte noch detailgetreuer nachzubauen. Auf dieses Gitternetz wird eine Textur aufgesetzt. Texturen sind Bilder, welche auf das Gitternetz aufgelegt werden, um dem 3D-Objekt sein äußerliches Erscheinen zu geben.

Aufgrund von immer leistungsfähigeren Computern werden Voxel immer beliebter, da es durch diesen Ansatz möglich ist, den virtuellen Körpern ein Volumen zu geben. Die Wortschöpfung Voxel besteht auch aus den Wörtern „Volumen“ und Pixel“. Also ein Pixel, der über ein Volumen verfügt. Oder quasi die 3D Version eines Pixels wie man ihn bei Digitalen Bildern kennt. Statt einem Quadrat ist das in diesem Fall natürlich ein Würfel. Aus diesen Würfeln kann man dann letztendlich 3D Objekte bauen, wie man es aus Spielen wie „EverQuest Next“ kennt.

Ist diese Nachbildung fertig, so ist es möglich sich in dieser Umgebung zu bewegen, oder die Objekte von allen Seiten zu betrachten. So etwas kann zum Beispiel nützlich sein, wenn das Besuchen dieser Orte aufgrund von Entfernung oder mangelnder Zeit nicht möglich ist. So gibt es zum Beispiel auch einen Virtuellen Louvre Rundgang, der gerade für Studenten sehr attraktiv ist.

## 8051 Micro Controller

Qt ist ein C++ Framework, dessen Haupteinsatzgebiet in der plattformübergreifenden Programmierung für graphische Benutzeroberflächen liegt. Unteranderem verfügt das Framework über Datenbankfunktionen, XML-Unterstützung, Multimedia-Services, OpenGL-Unterstützung und Dienste auf Netzwerk und Dateiebene. Der Qt-Code folgt dem C++ Standard und ist daher mit jedem C++ Compiler übersetzbar. Diesem Projekt kommt zugute, dass Qt über eine Laufzeitumgebung verfügt, um JavaScript Code auszuführen, und dass Qt eigene Klassen mit einem vorangestellten „Q“ schnell zu identifizieren sind.   
Das Qt-Framework wird mittlerweile von dem finnischen Softwareunternehmen „Digia“ weiterentwickelt, nachdem „Digia“ den Qt-Entwicklungsbereich von „Nokia“ aufgekauft hat, wodurch das Projekt einen deutlichen Entwicklungsschub erfahren hat.

# Konzept

## Interrupt

## 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 welcher 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.

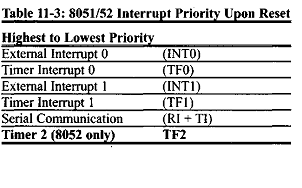


Abbildung : Interrupt Priorität (1)

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!

# Implementierung

# 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.]

1. Von Atari 1972 veröffentlichtes Videospiel, das auf Tischtennis basiert [↑](#footnote-ref-1)