DHBW
Duale Hochschule
Baden-Württemberg

Nappy's Home

Smarthome Steuerung auf einem 8051 Mikrokontroller

von

Heiko Faller, Mehmet Ali Incekara und Tom Wolske

Ehrenwörtliche Erklärung

gemäß § 5 (3) der „Studien- und Prüfungsordnung DHBW Technik“ vom 22. September 2011.

Wir (Heiko Faller, Mehmet Ali Incekara und Tom Wolske) haben die vorliegende Arbeit mit dem Titel

„ Nappy's Home “

selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet.

Karlsruhe, den 18.6.2016

Ort, Datum

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

Die Menschen wollen immer weiter vernetzt sein! Das betrifft auch das Haus in denen Sie leben.

Smart Home dient als Oberbegriff für technische Verfahren und Systeme in Häusern, in deren Mittelpunkt eine Erhöhung von Wohn- und Lebensqualität, Sicherheit und effizienter Energienutzung steht.

1.1 Motiviation

Auf die Idee ein Smarthome zu Entwickeln sind wir schnell gekommen.
Ältere Menschen in unserer Gesellschaft haben oft angst in den Urlaub zu fahren, weil sie ihr Eigenheim nicht alleine lassen wollen. Andere wollen Strom sparen um die Umwelt zu schonen. Aus diesem Grund wollen wir ein SmartHome entwickeln. Damit ältere Menschen wieder in den Urlaub gehen können und Umweltbewusste Menschen die Umwelt besser schonen können.

1.2 Ziel der Arbeit

Das Ziel dieser Arbeit ist es einen Mikrocontroller zur Haussteuerung zu entwickeln. Zunächst findet die Implementierung und Ausführung als Simulation in der MCU 8051 IDE statt.

1.3 Aufbau der Arbeit

Das Ziel dieser Arbeit lässt sich in Teilziele kategorisieren, um die Zielerreichung zu garantieren.
Das nachfolgende Kapitel befasst sich mit den Grundlagen der Mikrocontrollerentwicklung. Anschließend wird das Konzept und die Implementierung näher erläutert. Als Abschluss folgt eine Zusammenfassung.

2 Grundlagen

2.1 Der Mikrocontroller

Ein Mikrocontroller ist ein Ein-Chip-Computersystem, der einen Prozessor und Peripheriefunktionen beinhaltet. In den meisten Fällen befindet sich der Arbeits- und Programmspeicher teilweise oder auch komplett auf demselben Chip.
In der Industrie ist der Mikrocontroller bereits seit vielen Jahren ein nicht mehr wegzudenkendes Bauteil. Sie werden im Alltag in eingebetteten Systemen verwendet, zum Beispiel in Waschmaschinen, Fernsehgeräte und sogar im Fahrzeug als Steuergeräte für das Antiblockiersystem (ABS), Airbag und weitere[1].
Mikrocontroller können in verschienden Sprachen programmiert werden. Welche sich jedoch am besten eignet, ist vom Anwendungsfall abhängig. Assembler eignet sich besonders, da es einen direkten Einstieg in die Hardware bietet und keine Abhängigkeiten zu anderen Compilern hat[2].

2.2 Assembler

Ein Assembler ist ein Übersetzer für Programmcode, der sich aus Maschinenbefehlen zusammensetzt. In der Art der verwendeten Befehle besteht der wesentliche Unterschied zu allen anderen Programmiersprachen.
Während sich Befehle bei den Hochsprachen, wie beispielsweise Java und C++, in der Übersetzung aus mehreren Anweisungen im endgültigen Code zusammensetzen, wird der Assemblerbefehl durch den Assembler lediglich in die entsprechende Binärform übersetzt. Weiterhin ersetzt der Assembler Variablen durch die entsprechenden Speicheradressen[3].

2.3 MSC-8051

Der MSC-8051 ist die Bezeichnung einer von Intel vorgestellten Familie von 8-Bit-Mikrocontrollern.
Die Architektur enthält folgende Features[4].:

- 8 Bit CPU
- 2x 16 Bit Timer

[1] Vgl. http://www.mikrocontroller.net/articles/AVR-Tutorial
[2] Vgl. Kenneth J. Ayala, The 8051 Microcontroller, West Publishing Company, 1991, Seite 3
[3] Vgl. http://assembler.hpfsc.de/
[4] Vgl. Kenneth J. Ayala, The 8051 Microcontroller, West Publishing Company, 1991, Seite 8

- ROM mit 4 KByte ROM
- RAM mit 128 Byte
  - 4 Register mit 8 Bit

2.4 MCU 8051 IDE

Die MCU 8051 IDE ist eine grafische Entwicklungsumgebung für Mikrocontroller, die auf dem MSC-8051 basieren. Das folgende Bild stammt von der Homepage[5] der Entwicklungsumgebung. Es zeigt im Hintergrund den Aufbau der IDE und im Vordergrund einige Simulationsfeatures.

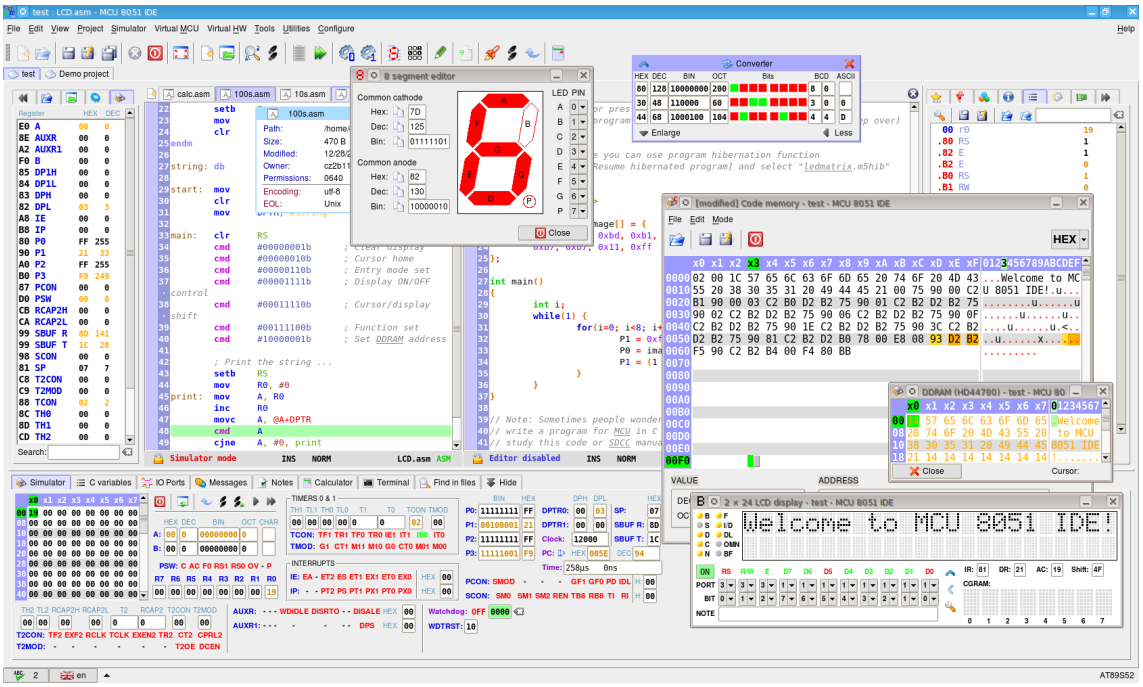

Abbildung 1: MCU 8051 IDE

Die Simulation ist eine Softwarekomponente zur Simulation des Mikrocontrollers in einer virtuellen Umgebung. Zusätzliche Features erweitern die Möglichkeit den Mikrocontroller zu simulieren. Beispielsweise gibt es Hardware-Komponenten wie Schalter, Timer und Temperatur Sensor.

[5] http://www.moravia-microsystems.com/mcu-8051-ide/

3 Konzept

3.1 Konzept und Idee

Die Idee ist es, ein SmartHome zu entwickeln, welches auf einem 8051 Microcontroller als Grundlage aufbaut. Dabei sollen drei Hauptfunktionen eines modernen SmartHomes über diesen gesteuert werden.

3.1.1 Heizung

Die erste Funktion ist eine Temperatursteuerung für die Fußbodenheizung. Dabei soll die Fußbodenheizung in einem Haus so gesteuert werden, dass diese automatisch eingeschaltet wird, sobald die Temperatur unter einen festen Wert fällt. Außerdem soll der Nutzer mit Hilfe eines Schalters festlegen können, ob die Heizung dauerhaft an oder in einen automatischen Modus eingestellt ist. Der Temperatursensor ist zusätzlich abschaltbar, damit die Heizung ausgeschaltet werden kann.
Wird der Heizungsschalter auf An gestellt, läuft die Heizung unabhängig von der Temperatur die ganze Zeit. Wird dieser auf den Automodus gestellt, kommt es auf den Temperatursensor an, da dieser unabhängig von der Heizung ein bzw. ausgeschaltet werden kann. Ist dieser eingeschaltet, schaltet sich die Heizung an, sobald die Temperatur unter einen bestimmten Wert fällt. Aus diesen Anforderungen ergibt sich eine Tabelle aus der alle möglichen Eingaben und Ausgabemöglichkeiten ausgelesen werden können. Aus diesen geht ein Schaltplan hervor, welcher für die Programmierung des 8051 Microcontrollers genutzt wird. Die Tabelle sieht wie folgt aus:

Tabelle 1: Schaltplan Heizungssteuerung

Heizungsschalter	**Temperatursensor Schalter**	**Temperatur < Wert**	**Heizung An?**
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

Für diese Steuerung wird die Schaltung nach diesem Schaltplan programmiert:
Dabei kommen die Eingaben aus dem Port 2.0 bis 2.2 und die Ausgabe ist auf dem Port 3.4.

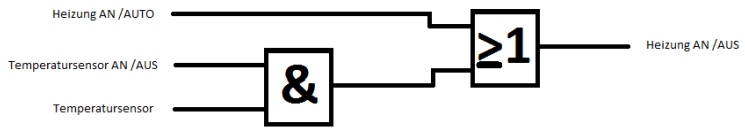

Abbildung 2: Schaltung für die Heizung

3.1.2 Licht

Die zweite Funktion ist eine Lichtsteuerung für das Licht. Der Nutzer soll die Möglichkeit haben das Licht mit Hilfe eines Bewegungssensors ein beziehungsweise aus zu schalten. Dieser Sensor ist zusätzlich an einen Schalter angeschlossen, um diesen An bzw. Aus zu schalten.
Aus diesen Anforderungen ergibt sich eine Tabelle aus der alle möglichen Eingaben und Ausgabemöglichkeiten ausgelesen werden können. Aus diesen geht ein Schaltplan hervor, welcher für die Programmierung des 8051 genutzt wird.

Tabelle 2: Schaltplan Lichtsteuerung

Bewegungssensor	**Schalter**	**Licht an?**
0	0	0
0	1	0
1	0	0
1	1	1

Für diese Steuerung wird die Schaltung nach diesem Schaltplan programmiert:

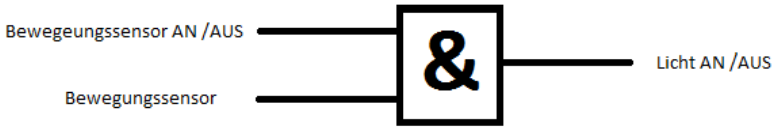

Abbildung 3: Schaltung für das Licht

Dabei kommen die Eingaben aus den Ports P1.0 und 1.1 und die Ausgabe ist auf dem Port 3.5.

3.1.3 Rollläden

Der dritte Teil der SmartHome Steuerung ist eine Steuerung für Rollläden. Diese werden in zwei Modi betrieben. Im automatischen Modus sollen sie sich automatisch öffnen, sobald es draußen hell, beziehungsweise schließen, sobald es draußen dunkel wird. Es wird ein Helligkeitssensor benötigt, der ein bit setzt, sobald es hell ist und umgekehrt. Der zweite Modus ist eine manuelle Steuerung. Für jeden Rollladen gibt

Tabelle 3: Rolladensteuerung manuell

SR Oben	**SR Unten**	**SR Hoch**	**SR Runter**	**MR Hoch**	**MR Runter**
0	0	0	0	0	0
0	0	0	1	0	1
0	0	1	0	1	0
0	0	1	1	0	0
0	1	0	0	0	0
0	1	0	1	0	0
0	1	1	0	1	0
0	1	1	1	0	0
1	0	0	0	0	0
1	0	0	1	0	1
1	0	1	0	0	0
1	0	1	1	0	0
1	1	0	0	0	0
1	1	0	1	0	0
1	1	1	0	0	0
1	1	1	1	0	0

Tabelle 4: Rolladensteuerung automatisch

SR Oben	**SR Unten**	**Helligkeitssensor**	**MR Hoch**	**MR Runter**
0	0	0	0	1
0	0	1	1	0
0	1	0	0	0
0	1	1	1	0
1	0	0	0	1
1	0	1	0	0
1	1	0	0	0
1	1	1	0	0

es je zwei Schalter, die für Rollladen Hoch, beziehungsweise für Rollladen runter, stehen. Beide Modi werden durch Sensoren unterstützt, die jeweils anzeigen, ob ein Rollladen geschlossen oder ganz geöffnet ist. Aus diesen Anforderungen gehen wie zuvor auch diese Tabellen für die beiden Modi hervor:
Die Signale aus Sensoren und Schalter sollen über P0, sowie über P2.6 und P2.7 eingegeben werden. Die Ausgabe an die Motoren erfolgt über P3.0 - P3.3

4 Umsetzung

4.1 Rolladen-Steuerung

Die Schwierigkeit bei der Entwicklung der Rollladenschaltung war, dass eine Menge Port-Bits benötigt werden, weshalb wir uns dazu entschieden haben, ein Modell zur Steuerung von zwei Rollläden zu wählen. Als erstes wird das Automodus-Bit abgefragt, um zu entscheiden, ob der Automodus eingesetzt wird oder die Röllläden manuell gesteuert werden.

4.1.1 Manuelle Steuerung

Im manuellen Modus wird bei den Röllläden abwechselnd überprüft, ob sie hoch oder runter fahren sollen. Dazu wird mittels Polling jeweils abgefragt, wie die Schalterpositionen sind und ob der zu überprüfende Rollladen oben oder unten ist. Dementsprechend werden dann die Motoren angesteuert. Es werden dabei immer die Zustände beider Schalter, also sowohl für hoch als auch für runter, abgefragt. Das dient dazu, den fehlerhaften Zustand, dass beide aktiviert sind zu umgehen. In diesem Fall würde der Rollladen dann einfach nicht starten, als wären beide Schalter inaktiv.
Die Abfragen laufen folgendermaßen ab: Zuerst wird bei beiden Rollläden nach folgendem Schema geprüft, ob sie nach oben fahren sollen.

1. Rolladen nicht oben?
2. Schalter hoch aktiviert?
3. Schalter runter deaktiviert?
4. Rolladen abhängig davon hochfahren

Daraufhin folgen, wieder für beide Rollläden, die Abfragen, um zu prüfen ob sie nach unten fahren müssen.

1. Rolladen nicht unten?
2. Schalter runter aktiviert?
3. Schalter hoch deaktiviert?
4. Rolladen abhängig davon heruntergefahren

4.1.2 Automodus

Ist der automatische Modus aktiviert, ist die Position der Schalter egal, da diese nicht mehr abgerufen werden. Stattdessen kommt ein Helligkeitssensor zum Einsatz, der bei anschlägt, wenn es draußen hell ist. In diesem Fall werden also das Sensor-Bit, sowie wieder die Positionssensoren, abgefragt. Zuerst kommt wieder für beide Rollläden die Prüfung, ob er nach oben fahren soll.

1. Rolladen nicht oben?
2. Helligkeitssensor schlägt an?
3. Rolladen abhängig davon hochfahren

Abschließend kommen nun noch die Abfragen, ob die Rollläden nach unten fahren sollen.

1. Rolladen nicht unten?
2. Helligkeitssensor schlägt nicht an?
3. Rolladen abhängig davon heruntergefahren

4.2 Licht-Steuerung

Bei der Lichtsteuerung werden einfach die Eingänge verundet und das Ergebnis an das Licht ausgegeben. Der Ablauf sieht also folgendermaßen aus.

1. Automatische Lichtsteuerung aktiviert?
2. Sensor schlägt an?
3. Licht davon abhängig an- beziehungsweise ausschalten

4.3 Heizung-Steuerung

Ein Temperatursensor liefert der Heizungssteuerung die Information, ob die Temperatur sich unter dem Fixwert befindet. Wenn das der Fall ist und die automatische Steuerung aktiv ist wird die Heizung aktiv. Unabhängig davon wird die Heizung auch aktiv, wenn dier Schalter betätigt wird. Daraus ergibt sich folgender Ablauf:

1. Sensor schlägt an?
2. Automatische Steuerung aktiviert?

3. oder manuelle Steuerung aktiviert?

4. Heizung davon abhängig an- beziehungsweise ausschalten

5 Zusammenfassung

shudubinoimapkül