DHBW
Duale Hochschule
Baden-Württemberg
Karlsruhe

Thema

Simulator für den Microcontroller PIC16F84A

DOKUMENTATION

von

Claus Burkhart

&

Thomas Coenen

Abgabedatum: 18.05.2015

Bearbeitungszeitraum	12 Wochen
Kurs	TINF13B3
Vorlesung	Rechnertechnik
Betreuer	Herr Stefan Lehman

Inhaltsverzeichnis

Abbildungsverzeichnis

Abkürzungsverzeichnis

GUI	Graphical User Interface
IDE	Integrated Development Environment
MVC	Model View Control

1. Einleitung

1.1. Ziele des Dokuments

Mit Hilfe dieses Dokuments soll erläutert werden, wie das Projekt strukturiert ist und wer welchen Part im Projekt übernimmt. Des Weiteren soll mit die Dokumentation so aufgebaut sein, dass auch ohne das Starten des Programms die Funktionen und die Arbeitsweise nachvollzogen werden können.

1.2. Aufgabenstellung

Als praktischer Bestandteil des Studiums im Studiengang Informatik der Fachrichtung Informationstechnik an der DHBW Karlsruhe, wird im 4. Theoriesemester in der Vorlesung Systemnahe Programmierung II ein Simulator für den PIC16F84A Mikrocontroller der Firma Microchip Technologie Inc. von den Studierenden erwartet. Während eines festgelegten Zeitraums von zwei Monaten, sollen einige der Funktionen dieses Mikrocontrollers abgebildet und eine GUI zur Visualisierung als Gruppenarbeit von 2 Personen realisiert werden. Ziel des Projektes ist es, sich dieses Thema zu erarbeiten, bereits Erlerntes aus den Vorlesungen Rechnertechnik I, Software Engineering und Digitaltechnik zu vertiefen und sich mit einer gewählten Programmiersprache selbstständig auseinander-zusetzen.

Projekt:Simulator für den Microcontroller PIC16F84

an der DHBW Karlsruhe

Bewertung in Form eines Testats:
Mindestpunktzahl: 60 (= 4,0)

		Punkte	Punkte
Zwei Testprogramme (werden zur Verfügung gestellt) Die Testprogramme laufen Befehl für Befehl fehlerfrei und liefern auch die richtigen Teilergebnisse!			**20**
Effektives und sinnvolles Programmieren			**30**
• Bedienungskomfort der Oberfläche	2		
• sinnvolle, ausführliche Kommentare im Quellcode	2		
• Programmstruktur (übersichtliche Funktionen bzw. Prozeduren)	1		
• Hilfefunktion (PDF per Button-Klick aufrufen)	3		
• Variablenhandhabung (sinnvolle Bezeichnungen)	1		
• bestimmte Register beim RESET mit Werten vorbelegen	2		
• externer / interner Takt am TMR0-Pin incl. Vorteiler	4		
• Latchfunktion der Ausgaberegister (incl. Visualisierung)	2		
• einfache und sinnvolle Befehlsnachbildung	1		
• Interruptfunktion (mind. TMR0- und RB0-Interrupt)	5		
• Breakpoints	3		
• Stackfunktionen (incl. Visualisierung)	2		
• Laufzeitzähler (incl. Visualisierung)	2		
Zusatzpunkte			**40**
• EEPROM Funktionen	5		
• Watchdog incl. Vorteiler	3		
• Sleep-Funktion	3		
• externer Taktgenerator (mind. TMR0 und 1 anderer IO-Pin)	5		
• frei wählbare Quarzfrequenz	5		
• eigene Ideen (nach Absprache, bis max. 4 Punkte)	4		
• Hardwareansteuerung (RS232)	10		
• Vorzeitige Abgabe (nach Absprache, max 5 Punkte)	5		
Dokumentation:			**10**
• Darstellung von Programmstruktur und Ablauffunktionen des Simulators	2		
• ausführliche Beschreibung mit Ablaufdiagramm der Funktionen am einem konkreten Beispiel je Befehlsgruppe (siehe Vorgabe)	5		
• Ablaufdiagramme des Simulationsvorganges	1		
• äußeres Erscheinungsbild	2		
erreichte Punkte / mögliche Gesamtpunktzahl (60 Punkte sind für das Testat notwendig)			**100**

Abb. 1

2. Analyse

2.1. Was ist ein Simulator?

Ein Simulator oder allgemein eine Simulation dient der Analyse, um das Verhalten eines Systems unter bestimmten Vorgaben und Umständen zu testen. Dieses System simuliert spezifische Bedingungen, die Eigenschaften eines realen Prozesses oder einer Maschine für die Zwecke der Forschung oder Bedienerschulung, ohne dass bei dem Experiment eine Gefahr oder ein Risiko entsteht. Als anschauliches Beispiel kann ein Flugsimulator betrachtet werden.

2.1.1. Vor- und Nachteile

Vorteil einer Simulation ist die Ersparnis von Zeit und Kosten, da mit der Hilfe der Simulation ein Experiment nicht in Wirklichkeit auf die Beine gestellt, sondern wie der Name schon sagt, nur simuliert wird. Ein besonderer Vorteil der Simulation ist, dass sie leicht modifizierbar und so ausgelegt ist, dass man einzelne Vorgänge besonders gut beobachten, testen oder pausieren kann. . Durch heutige Rechner-kapazität lassen sich ohne großen Aufwand viele Experimente parallel durchführen.

Nachteil einer Simulation ist, dass das echte oder reale System nur nachgestellt wird und es dadurch zu Ungenauigkeiten und Abweichungen kommen kann. Ebenso haben die Simulationen einen hohen Datenbedarf und können je nach Aufwand des Modells teuer und zeitintensiver sein

2.1.2. Was ist der PIC16F84A?

PIC-Mikrocontroller (Programmable Interface Controller), sind elektronische Schaltungen, welche programmiert werden können, um eine breite Palette von

Aufgaben zu bewältigen. Sie sind in den meisten elektronischen Geräten wie Alarmsystemen, Computersteuerungen, Telefonen verbaut. Es gibt verschiedene Arten von PIC-Mikrocontrollern, welche durch eine Schaltassistent Software simuliert werden können.
PIC-Mikrocontroller sind relativ günstig und können als vorgefertigte Schaltungen oder als Kits, die vom Benutzer zusammengestellt werden können gekauft werden. Der PIC16F84A ist einer der verschiedenen Arten und die genauere Beschreibung lautet 18-pin Enhanced FLASH/EEPROM 8-Bit Microcontroller.

2.2. Wahl der Programmiersprache

Für die Implementierung des Simulators wird die Programmiersprache Java von SUN Microsystems Inc. verwendet. Die Sprache wurde wegen ihrer hohen Portabilität auf andere Systeme gewählt.

Abb. 2

2.3. Programmierumgebung

Für die Erstellung des Simulationsprogramms, sowie der GUI wurde die Entwicklungsumgebun Eclipse benutzt.

Abb. 3

2.4. Versionsverwaltung

Um eine ordentliche Versionsverwaltung zu erhalten, wurde der webbasierter Hosting-Dienst GitHub von GitHub Inc. verwendet. Mit Hilfe dieses Programms konnte ebenso gewährleistet werden, dass alle an dem Projekt beteiligten Personen den gleichen Stand des Codes in ihrer IDE hatten.

2.5. Planung und Entwurf

Das Programm wird nach dem MVC Muster aufgebaut. Dieses Bezeichnet ein Architekturmuster zur Aufteilung von Softwaresystemen in die drei Einheiten: Datenmodell(engl. Model), Präsentation (engl. View) und Programmsteuerung (engl. Controller).

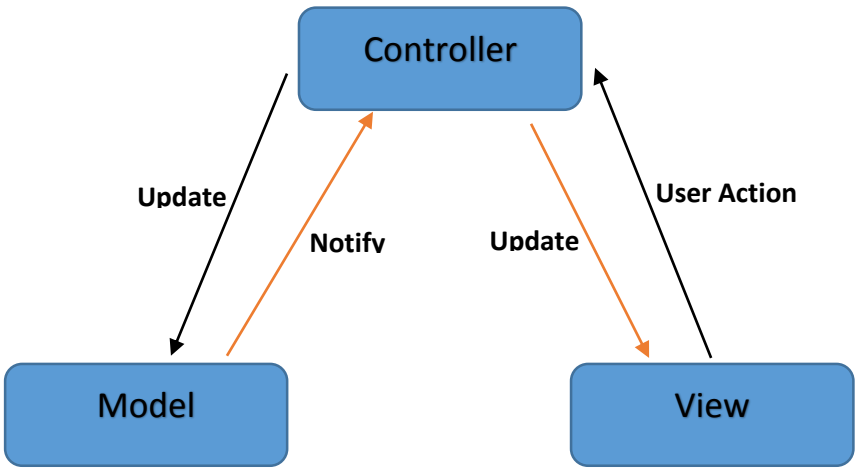

Abb. 4

2.4. Zeitplanung

Das Projekt startete Anfang April 2015 und sollte bis Mitte Juni 2015 fertiggestellt werden. Um einen der möglichen Zusatzpunkte zu erhalten, wird die Abgabe des Projekts auf den 18. Mai gesetzt. Für die gesamte Arbeit ergibt sich somit eine Zeitvorgabe von knapp 2 Monaten.
Damit das Projekt effizient und effektiv durchgeführt werden kann, muss eine ordentliche Zeitplanung erstellt werden.
Die Tabelle 1 zeigt eine solche Zeitplanung. In dieser können die Meilensteine und dafür geplante Entwicklungszeit eingesehen werden.

Meilenstein	Geplante Eintwicklungszeit	Aufgabe / Funktion
1	1 Woche	Grundsätzliches Verstehen der Aufgabe und des PIC16F84
2	5 Tage	Grafische Oberfläche erstellen
3	1 Woche	Einlesen von Dateien
4	5 Tage	Erstellung und Ansprechen des Registers
5	4 Tage	PIC Befehle implementieren
6	6 Tage	Highlighting und Breakpoints
7	5 Tage	Tris und Port-Schaltung
8	5 Tage	Stack Bereich
9	1 Woche	Spezielle Funktionen implementieren und Anpassung der Grafischen Oberfläche
10	2 Wochen	Abschließende Dokumentation

Ohne die Abschließende Dokumentation ergibt sich eine Entwicklungszeit von ca. 6 Wochen. In der geplanten Zeit sind die Überarbeitung der Oberfläche und des Codes (Refactoring) inbegriffen.

2.5. Aufgabenverteilung

Für das Projekt wurde das Entwicklungsmodell XP (Extreme Programming) verwendet. Dieses Model ist sehr flexibel, da die Zusammenarbeit durch ständigen Wissensaustausch und Kommunikation untereinander unterstützt wird. Das Programmieren in Paaren führt dazu, dass sich die Entwickler ständig gegenseitig kontrollieren und auf Fehler aufmerksam machen. Zu guter Letzt hat XP eine positive Wirkung auf die Teamarbeit und die Motivation der Entwickler, da sie sich mit dem hochwertigen Code identifizieren können und feiern Erfolgserlebnisse, wenn wieder eine neue Version ausgeliefert werden kann.

3. Umsetzung

3.1. Erstellung der GUI

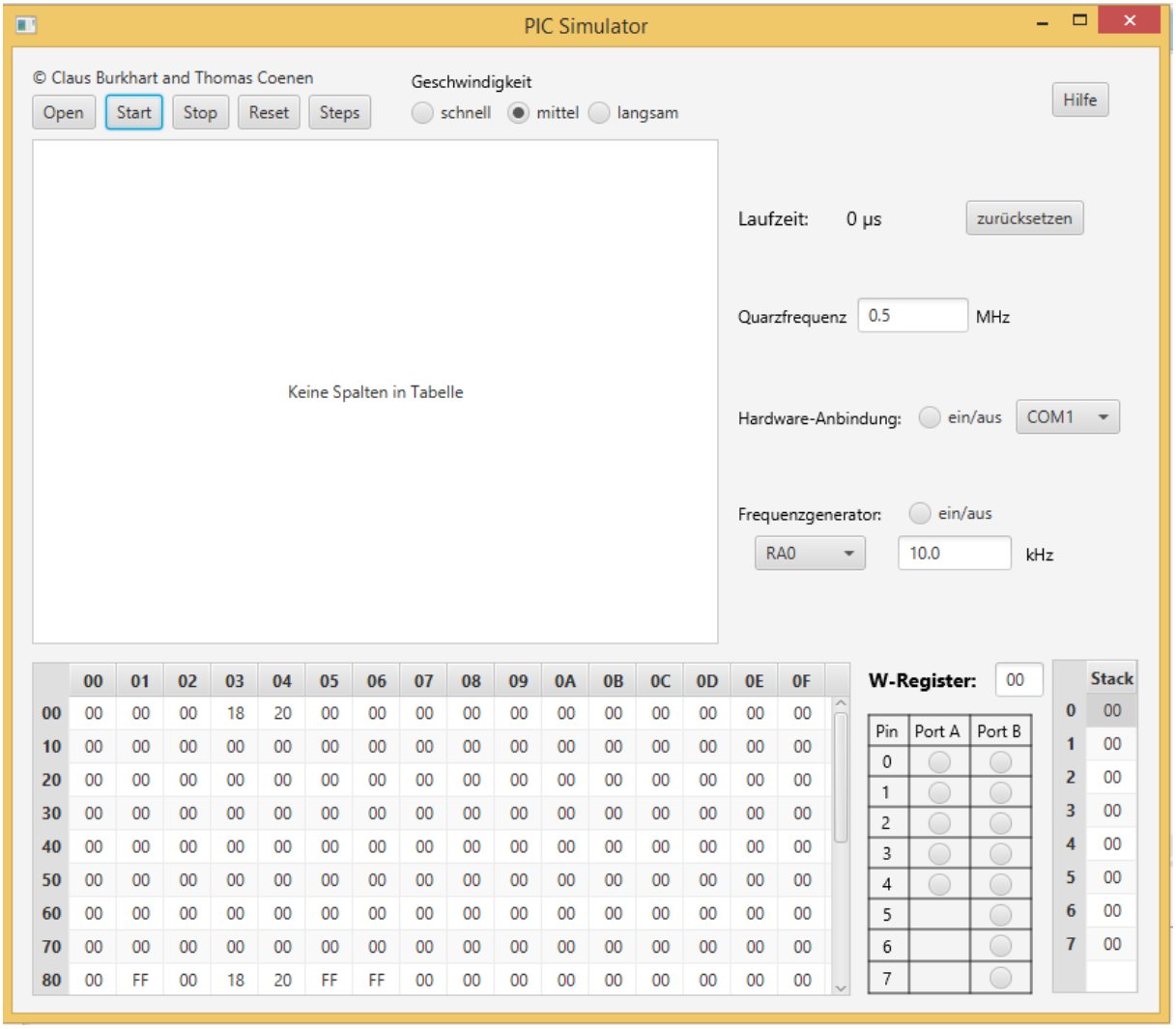

Abb. 5

3.2. Einlesen von Dateien

3.3. Erstellung und Ansprechen des Registers

3.4. Stack Bereich

3.5. RAM Bereich

3.6. Port-State Bereich

3.7. Implementierung der Befehle

3.5.1. BTFSS

3.5.2. CALL

3.5.3. MOVF

3.5.4. ADDWF

Implementiert, nicht implementiert