Projekt:Simulator für den Microcontroller PIC16F84

an der DHBW Karlsruhe

12.03.2014

Der PIC16F84 Simulator

Im Fach Rechnertechnik II soll der Aufbau und die Funktionsweise eines Microcontroller gelernt werden. Über den „Umweg“ ein Simulatorprogramm zu schreiben, das die Funktionen eines realen oder imaginären Controllers nachbildet, müssen die Studenten neben dem Studium des Datenblattes auch die bereits erlernten Fertigkeiten aus der Vorlesung Software-Engineering, Digitaltechnik und Rechnertechnik I anwenden. Eine einfache Hardwarebeschaltung an der seriellen oder parallelen Schnittstelle bildet die Brücke zwischen virtueller und realer Welt.

1.) Notwendige Kenntnisse:

- Bool’sche Algebra, binäre Schaltlogik
- Binäre Rechenverfahren (Addition, Subtraktion, Multiplikation)
- Ablaufsteuerungen (Automaten, Zustandsmaschinen FSM)
- Digitaltechnik (Grundgatter, FF, Latchregister, Schieberegister, ROM, PLA, TRI-STATE etc)
- Rechnerarchitekturen, Busstrukturen
- Registerstruktur eines µPs, alternative Strukturen
- Befehlszyklen, Maschinenzyklen, Befehlsdekodierung, Befehlsabarbeitung
- Funktion der Statusbits
- Analyse und Synthese eines (hypothetischen) Befehlssatzes
- Einfache Assemblerprogrammierung
- Programmtechniken zur syntaktischen und lexikalischen Analyse von Textdateien (Parser, Codegenerator, [Codeoptimierer])

2.) Allgemeine Fragen:

- Gruppeneinteilung, Gruppenarbeit usw.
- Terminplan / Benotung
- Was ist ein Simulator? (u.a. Internetrecherchen)
- Welche Vor- bzw. Nachteile bietet er? Alternativen zum Simulator?
- Was ist der PIC 16F84? (Struktur, Registeraufbau, Befehlsaufbau, Philosophie)

3.) Speziellere Fragen:

- Welche Programmiersprache soll verwendet werden?
- Ist eine Nutzung eines externen Assemblers erlaubt? (Datenformat)
- Welche Hardwaremodule besitzt der PIC und wie können diese per Software simuliert werden (Speicher, Peripherie, Interrupt usw.)?
- Welche Funktionen besitzt der PIC und wie können diese nachgebildet werden (arithm. und logische Funktionen, Statusregister usw.)?
- Wie kann der Befehlssatz des PICs interpretiert und nachgebildet werden (Befehlsdekoder und Steuerregister)
- Simulator soll über die RS232 / den Parallelport Signale ausgeben bzw. einlesen können.

4.) Realisierung:

- Entwurf eines groben Konzeptes (Oberfläche, Bedienung, Datenformate, grobe Variablenstruktur) in Gruppen zu 4 bis 6 Personen
- Ausarbeitung des Konzeptes und Feingliederung mit Hilfe der bereits gelernten Techniken (Software-Engineering) in Zweiergruppen
- Programmcodierung, Test und Debugging in Zweiergruppen
- Dokumentation (Online-Help, PDF-Dokument) in Zweiergruppen
- Repräsentation in Zweiergruppen

Was gehört zur Dokumentation?

Nachfolgend sind nur die allernotwendigsten Punkte aufgeführt. Schreiben Sie die Punkte nicht einfach ab, sondern passen Sie diese individuell an ihre Arbeit an. Machen Sie sich Gedanken welche Informationen zum Verständnis für "Außenstehende" noch wichtig sind. Geben Sie die Arbeit einer Person die die Aufgabe nicht kennt zu lesen und stellen Sie fest was diese Person davon verstanden hat.

Allgemeines:

- Grundsätzliche Arbeitsweise eines Simulators.
- Vor- und Nachteile einer Simulation.
- Programmoberfläche und deren Handhabung.

Realisation:

- Beschreibung des Grundkonzepts
- Beschreibung der Gliederung
- Programmstruktur, Ablaufdiagramme, Variablen usw.
- Welche Programmiersprache wurde gewählt? Warum?
- tiefergehende Beschreibung der Funktionen an Hand ausgewählter Beispiele (BTFSx, CALL, MOVF, RRF, SUBWF, DECFSZ, XORLW). Diese und ggf. weitere Befehle anhand von kurzen Programmsequenzen und Ablaufdiagrammen erläutern.
- Realisierung der Flags und deren Wirkungsmechanismen.
- Wie wurden Interrupts implementiert? (Auszug aus dem Listing)
- Wie wurde die Funktion des TRIS-Registers realisiert? (Latchfunktion?)
- Alle Register müssen manuell veränderbar sein.
- Breakpoints sollten implementiert sein
- Hardwareansteuerung
- Helpfunktion ruft nur die Dokumentation auf.
- Diagramme und Beschreibung der Interruptfunktion.
- Diagramm und Beschreibung mittels State-Machine z.B. für EEPROM

Zusammenfassung:

- Wie weit konnten die Funktionen des Bausteins per Software nachgebildet werden?
- Fazit, persönliche Erfahrung und Erkenntnis. Was passierte während der Entwicklung des Projektes? Welche Probleme tauchten auf und wie wurden Sie gelöst. Vermeiden Sie dabei negative Formulierungen. Was würde ich anderst machen, wenn ich das Projekt nochmals realisieren müsste? (Umfang des Fazits ca. ¾ bis 1 Seite oder 10 % des Gesamtumfangs)

Bemerkung:
Das Programmlisting ist Bestandteil des Anhangs.
Das Datenblatt des PIC-Prozessors gehört nicht zur Beschreibung, allenfalls teilweise in den Anhang.
Die Dokumentation sollte so aufgebaut sein, dass auch ohne das Starten des Programms die Funktionen und Arbeitsweise nachvollzogen werden kann.

Arbeiten Sie, wenn möglich, mit einer Versionsverwaltung. Planen und dokumentieren Sie den zeitlichen Verlauf der einzelnen Projektschritte.

Gutes Deutsch ist selbstverständlich. Die Arbeit und deren einzelne Kapitel müssen strukturiert sein. Es gilt auch hier: Einleitung - Hauptteil - Schluss. Eine gewisse Dramaturgie (Spannungsbogen) ist durchaus von Vorteil.

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**

!!! Von diesem Bewertungsschema kann im Bedarfsfall jederzeit abgewichen werden !!!