Universität der Bundeswehr München
Fakultät für Elektrotechnik und Technische Informatik
Wissenschaftliche Einrichtung 4 - Daten- und Schaltungstechnik

Prof. Dr.-Ing. Ferdinand Englberger
Prof. Dr.-Ing. Thomas Latzel

Dokumentation zur Schrittmotorsteuerung

Meilenstein 2 - NIOS II-Basissoftware mit Stepperkomponente

Autoren: Lt Marc Kossmann
Lt Michael Riedel
Gruppe: SoC04

Abgabedatum: 18.11.2014

Inhaltsverzeichnis

1. Planung des Projekts

Das Gantt-Diagramm in Abbildung 1.1 wurde auf eingetretene Verzögerungen angepasst, sodass die Deadline Mitte Dezember eingehalten werden kann.

Abbildung 1.2 zeigt den geplanten und benötigten Zeitaufwand für die Erstellung des Meilenstein 2 unterteilt in folgende Aufgabenbereiche:

- Einarbeitung
- Zeitplanung
- Design
- Implementierung
- Verifikation
- Dokumentation

Die Darstellung wird gesondert für die Studenten Marc Kossmann und Michael Riedel betrachtet. Diese Zeiten sind unabhängig von gemeinsam bearbeiteten Aufgaben. Die Abbildung 1.3 zeigt die komplette geplante und benötigte Zeit, die durch Aufsummierung der einzelnen Meilensteine entsteht.

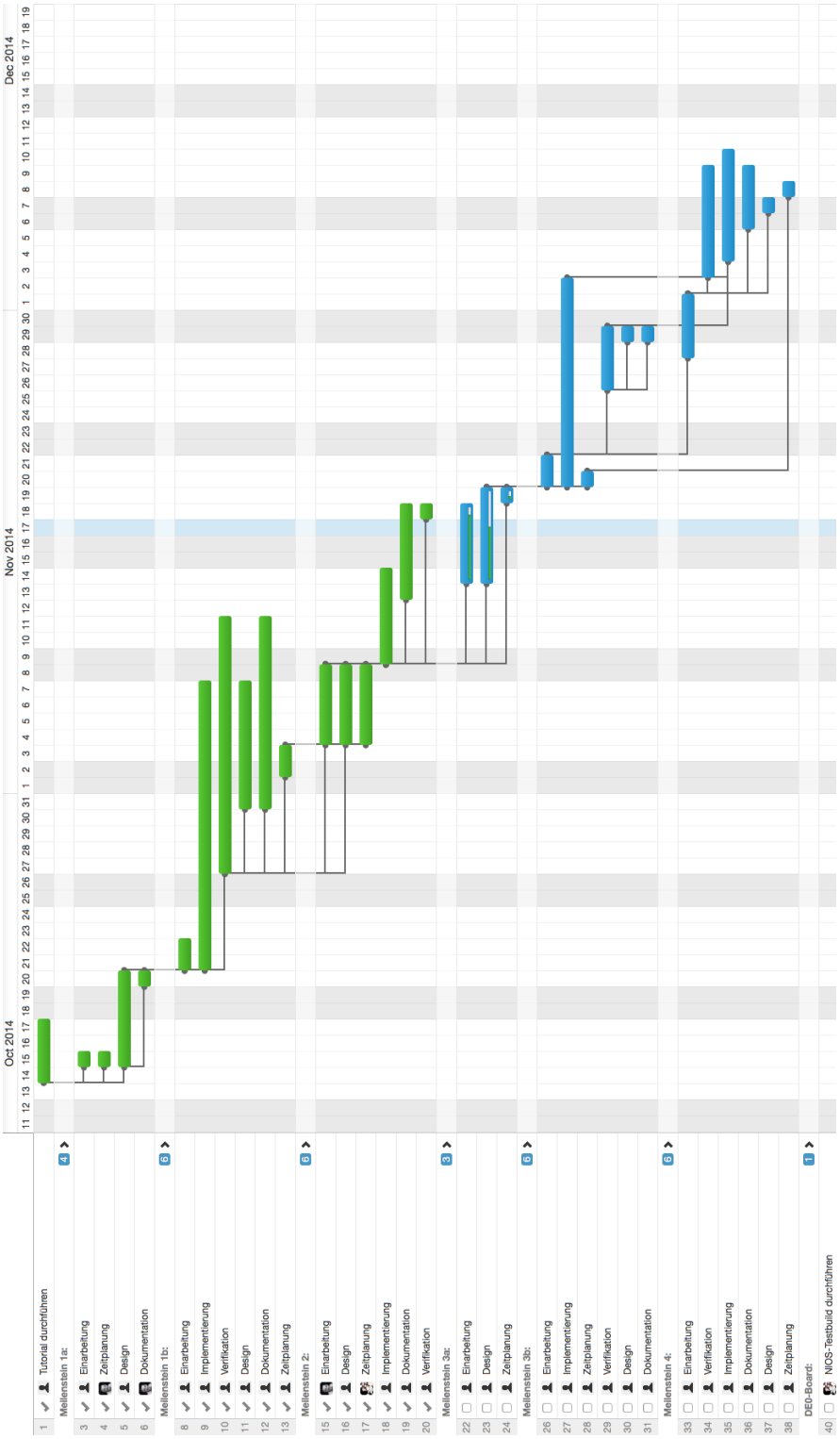


Abbildung 1.1: Gantt-Diagramm zur kompletten Zeitplanung

Meilenstein 2

	Marc		Michael		gemeinsam		gesamt	
	geplant	benötigt	geplant	benötigt	geplant	benötigt	geplant	benötigt
Einarbeitung	0,5	1	0,5	0	0	0	1	1
Zeitplanung	0	0	1	0,68	0	0	1	0,68
Design	2	0	1	0	0	0	3	0
Implementierung	8	16	4	0	12	0	24	16
Verifikation	2	0	0	1	2	2,25	4	3,25
Dokumentaion	1	2	2	5,35	0	0	3	7,35
Summe	13,5	19	8,5	7,03	14	2,25	36	28,28

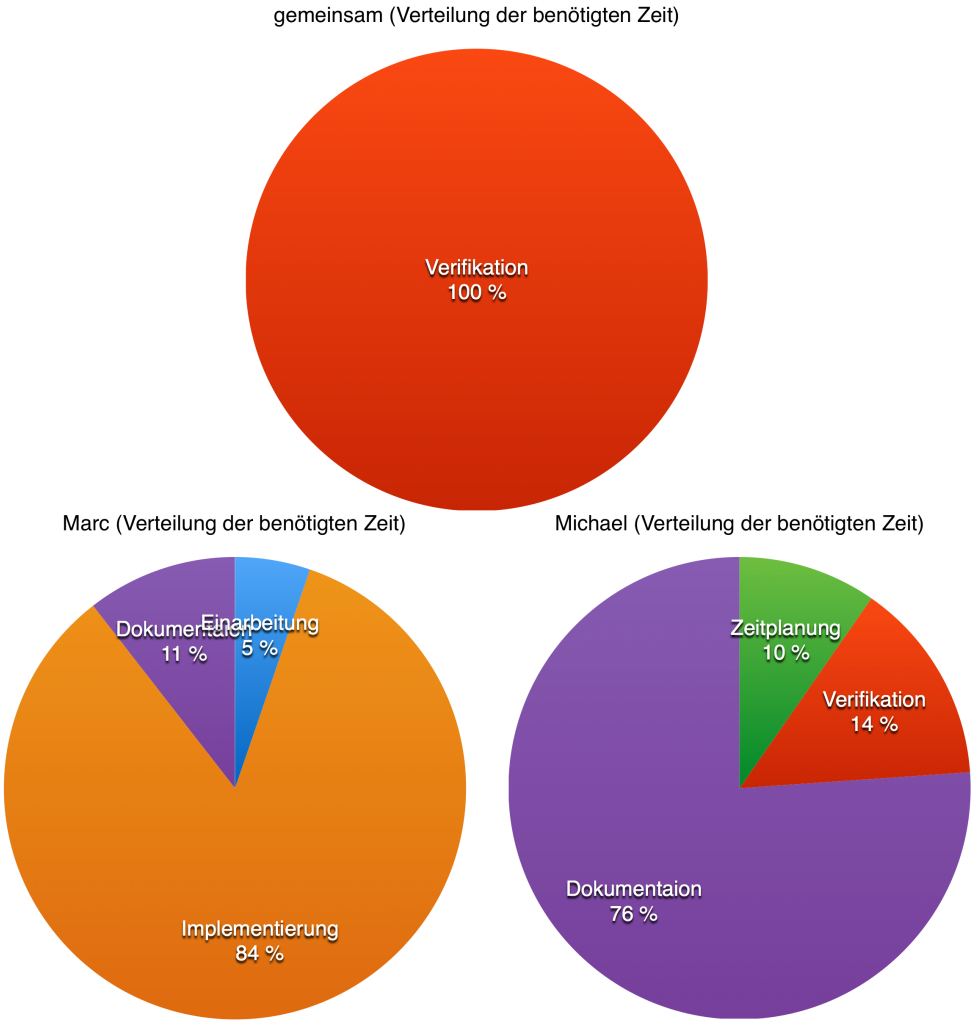
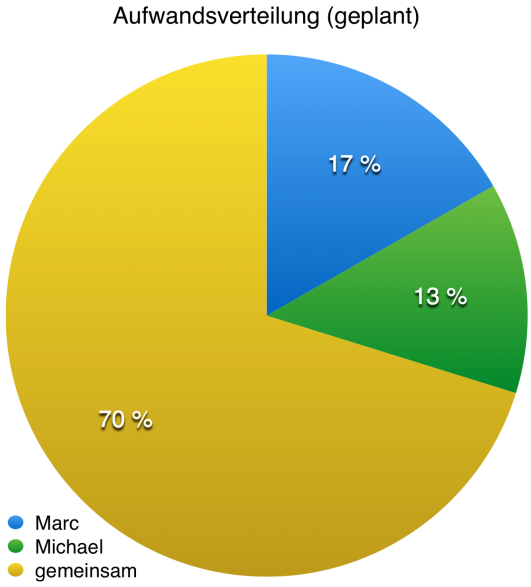


Abbildung 1.2: Projektplanung für Meilenstein 2

Zeitbedarfsübersicht

	Marc	Michael	gemeinsam	gesamt
geplant	41	32	172	245
benötigt	70,25	57,66	31,52	159,43



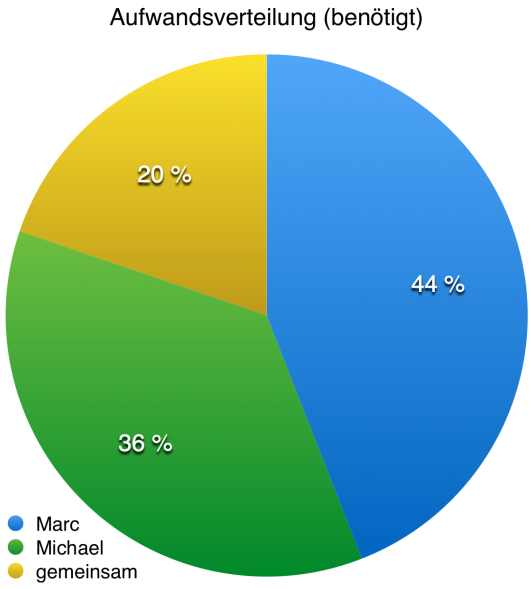


Abbildung 1.3: Zeitbedarfsübersicht für das gesamte Projekt

2. Design der Komponente *Register-Interface*

Aufgrund der geringen Komplexität wurde für diese Komponente kein expliziter Designprozess wie z. B. in Meilenstein 1 durchlaufen. Nach einer kurzen Einarbeitung in die Aufgabenstellung und unter Berücksichtigung der Erkenntnisse aus dem Tutorial, wurde der Quellcode direkt entworfen. Anschließend wurde das gewünschte Verhalten mithilfe einer Modelsim-Simulation verifiziert.

Abbildung 2.1 zeigt die daraus entstandene Komponente `register_interface`, die jetzt in Quartus zur Verfügung steht.

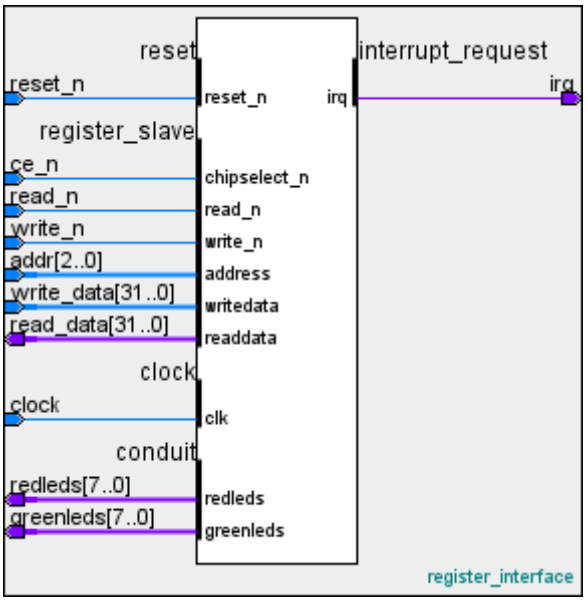


Abbildung 2.1: Block Diagramm des Register Interface

Gemäß Aufgabenstellung ermöglicht sie das Lesen und Schreiben über einen 3-Bit Adressbus. Dabei können die Register `stepsReg` und `speedReg` direkt, das `ctrlReg`-Register direkt sowie über sogenannte *Set* und *Clear*-Register beschrieben werden. Durch die `ctrlSetReg` und `ctrlClrReg`-Register können einzelne auf `1` gesetzte Bits verändert werden, die mit `0` maskierten Bits behalten den bisherigen Wert.

Die Tabelle 2.1 zeigt die notwendige Beschaltung für den Registerzugriff. Tabelle 2.2 beschreibt die notwendige Beschaltung des `register_interface` um lesend oder schreibend auf die Register zugreifen zu können.

Adresse	Register
000	ctrlReg
001	ctrlSetReg
010	ctrlClrReg
011	speedReg
100	stepsReg

Tabelle 2.1: Adressbeschaltung zum Registerzugriff

ce_n	read_n	write_n	Zugriff
0	0	0	*keiner*
0	0	1	*keiner*
0	1	0	*keiner*
0	1	1	*keiner*
1	0	0	undefiniert
1	**0**	**1**	**lesend**
1	**1**	**0**	**schreibend**
1	1	1	undefiniert

Tabelle 2.2: Beschaltung der Eingänge zur Einstellung des Zugriffsmodus

3. Änderungen an der Steuersoftware

Zugunsten der Übersichtlichkeit wurden alle definierten Datentypen in dem Header `dataTypes.h` zusammengefasst.

Wie für diesen Meilenstein verlangt, verwendet die Steuersoftware nun die tatsächlichen Register der VHDL-Komponente `register_interface`. Dazu wurden Makros für den Hardwarezugriff eingeführt und die entsprechenden Funktionen im Header `registerAcces.h` angepasst.

4. Darstellung der internen Kommunikation

Die `UserInput`- und `UserOutput`-Task hat bisher über eine Mailbox die Daten der Registerinhalte ausgetauscht. Diese Mailbox wurde in eine globale Variablenstruktur umgewandelt, da es in der Steuersoftware nur einen aktuellen Inhalt in den Registern gibt. Es ist nicht notwendig, vorherige Inhalte zwischenzuspeichern. Der geregelte Zugriff auf die globale Struktur wird durch die Verwendung eines Mutexes sichergestellt. Die Anpassungen sind in Abbildung 4.2 ersichtlich.

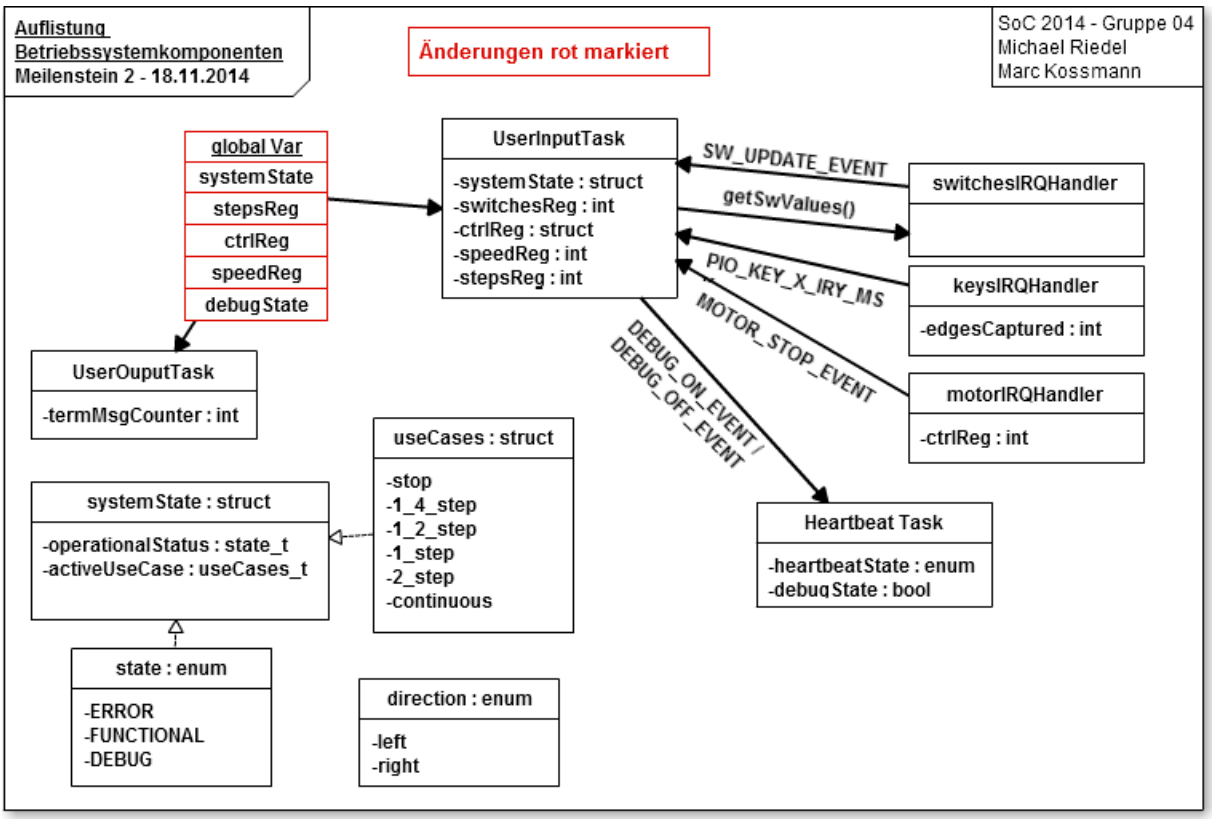


Abbildung 4.1: Auflistung Betriebssystemkomponenten

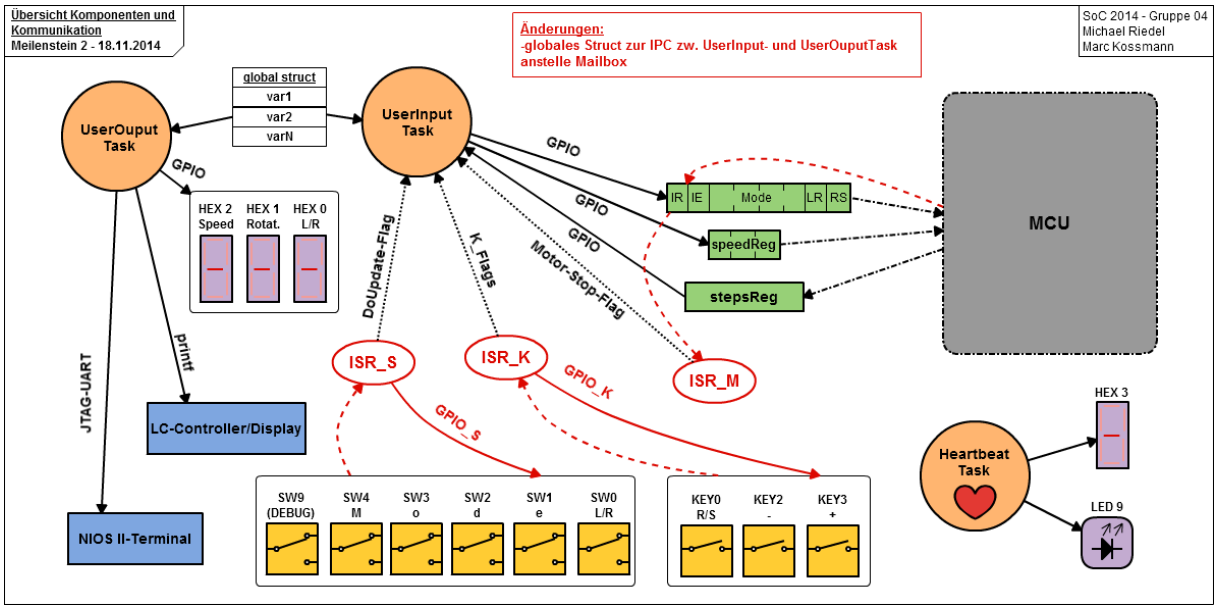


Abbildung 4.2: Übersicht der Komponenten und Kommunikation