



FACHHOCHSCHULE DARMSTADT

FACHBEREICH INFORMATIK

MIKROPROZESSORPRAKTIKUM

WS2005/06

Termin 5

C-Programmierung für eingebettete Systeme

Pointer, Peripherie, USART, SWI

Arbeitsverzeichnis:

Sie arbeiten in dem Verzeichnis /home/milabuser/mi2/Termin5. Dort stehen die Dateien *Termin5Aufgabe1.c* als Programmgerüstbeispiel und *makefile1* zur Verfügung.

Lernziele:

Die Programmierung von Funktionen die wiederum andere Funktionen aufrufen. Die Kenntnis der Basistechnologie zur Implementierung einer Schnittstelle zwischen Anwendungsprogrammen und Betriebssystemen. Die Bedeutung und Anwendung von Supervisor- und User-Mode.

Aufgabenstellung:

Der SWI Befehl führt zu einer Ausnahmebehandlung im Prozessor die mit einem Wechsel vom User in den Supervisor Mode verbunden ist. Dem SWI Befehl kann beim Aufruf eine bis zu 23 Bit große Zahl übergeben werden, die der SWI Handler dazu benutzen kann die gewünschte Funktion auszuwählen. Der SWI Handler ist ein Programm, das nicht in einer anderen Programmiersprache als Assembler geschrieben werden kann. (Warum?).

In der Aufgabe soll ein SWI Handler genutzt werden, um eine Trennung des Low Level IOs vom Anwendungsprogramm zu erreichen. Dazu sind die Funktionen die den SW-Interrupt aufrufen in Assembler zu implementieren und mit dem Debugger ist die Funktionsweise des Interrupthandlers zu untersuchen.

Aufgabe 1:

Sie sollen die Funktion puts (siehe *ser_io.S*) in Assembler schreiben. Die IO-Funktionen putchar und getchar (siehe *seriell.S*) werden dabei über einen SWI (siehe *swi.S*) aufgerufen. *void puts(char *)* Ausgabe eines nullterminierten Strings und Ersetzung von Newline durch Carriage Return (0x0d) und Linefeed (0x0a).

Aufgabe 2:

Erklären Sie den Code des SWI-Handlers (siehe *swi.S*). Debuggen Sie Ihr Programm und lokalisieren Sie die Umschaltung vom User Mode in den Superuser Mode und zurück. Dokumentieren Sie die Vorgänge im CPSR.

Aufgabe 3:

Entwickeln Sie eine Routine mit der Sie sich eine Integerzahl dezimal auf das Terminal ausgeben lassen können.