Institut für Informatik
Arbeitsgruppe Software Engineering
Barry Linnert

Betriebssysteme WiSe 2016/17· Übungsblatt 2
Bearbeitungszeit: 10.11.2016 – 24.11.2016 um 09:59 Uhr

Bereiten Sie Ihre **Lösungen** grundsätzlich so vor, dass Sie diese in der Übung Ihren Kommilitonen in geeigneter Form **zeigen** und **diskutieren** können. Geben Sie bitte stets Ihre verwendeten **Quellen** an.

Die Abgabe erfolgt elektronisch über das KVV.

Aufgabe 2-1 (Threads) – 6 Punkte

Wann ist es sinnvoll, nebenläufige Programmteile mit Hilfe von Threads anstatt Prozessen (heavyweight processes) zu implementieren?

Welche Vorteile bieten User-Level-Threads gegenüber den Kernel-Level-Threads? Gibt es auch Nachteile?

Welche Vorteile und Nachteile gibt es, wenn man Thread-Kontrollblöcke (TCB) als Skalare, in Arrays, Listen, Bäumen oder invertierten Tabellen speichert?

In welchem Adressraum (Prozess-Eigner, Dienste-Prozess, BS-Kern) wird ein TCB gespeichert?

Aufgabe 2-2 (Behandlung von Ausnahmen) – 15 Punkte

Nach der ersten Kontaktaufnahme mit der Zielplattform soll nun der ARM-Kern vollständig initialisiert werden und Ihr Betriebssystem-Code erste Aufgaben übernehmen, die über Low-level-Anwendungsentwicklung hinausgehen. Konkret sollen Sie Ausnahmesituationen abfangen, die entstehen, wenn Ihr Anwendungsprogramm derart Unsinn macht, dass der Prozessorkern an der weiteren Ausführung von Instruktionen gehindert wird.

Die Erkennung von solchen Ausnahmesituationen erledigt der ARM-Kern von selbst; Sie sollen daran anknüpfen, eine Meldung über Art und Ort der Ausnahme ausgeben und das System anhalten. Es darf auch gerne eine hilfreichere Meldung mit weiteren Informationen sein.

Um dies zu erreichen, sind folgende Teile zu erledigen (nicht unbedingt in dieser Reihenfolge):

1. Entwickeln Sie entsprechende Handler für die verschiedenen Ausnahmen.
2. Bei der Ausführung eines Handlers befindet sich der ARM-Kern in einem anderen Modus. Bei ARM haben die verschiedenen Modi unterschiedliche Stacks. Überlegen Sie also, wo Sie die Stacks im Speicher ablegen wollen und initialisiert **sämtliche** Stackpointer des Prozessors. (Stacks sind bei ARM übrigens *full-descending* gemäß des *Procedure Call Standard for the ARM Architecture*, an den sich der gcc hält.)
3. Wenn eine Ausnahme auftritt, setzt der ARM-Kern den *program counter* (PC) auf eine feste, Ausnahme-spezifische Adresse. Bei ARM ist die Interrupt Vektor Tabelle (IVT) hart verdrahtet, sodass Sie mit den vorgegebenen Adressen zurechtkommen müssen, die sehr eng beieinander liegen. Schreiben Sie entsprechende Instruktionen in diesen Speicherbereich (der mangels besserer Begriffe dennoch als IVT bezeichnet wird), sodass die tatsächlichen Handler ausgeführt werden.

4. Im gegenwärtigen Zustand befindet sich im Speicherbereich der IVT **kein** RAM! Bevor Sie also Ihre Handler installieren, führen Sie ein Memory-Remap durch, um dort änderbaren Speicher einzublenden.

Zum Testen bzw. zur Demonstration von zumindest einem Teil Ihrer Handler müssen Sie Code schreiben, der entsprechende Ausnahmen provoziert. Also:

5. Schreiben Sie eine „Anwendung“, die in Abhängigkeit von einer Benutzereingabe entweder einen *Data abort* erzeugt, einen *Software interrupt* auslöst oder eine *Undefined instruction* ausführt. (Ein *Prefetch abort* ist derzeit noch nicht möglich. Interrupts sind Teil der nächsten Aufgabe.)