ingo.korb [☺] tu-dortmund.de
thomas.hess [☺] tu-dortmund.de
niklas.ueter [☺] tu-dortmund.de

Übung zur Vorlesung
Eingebettete Systeme
Wintersemester 15/16

Übungsblatt 8

(10 Punkte)

Präsenzaufgaben zur Woche ab Montag, 4. Januar 2016

Für die Bearbeitung dieses Blatts wird die virtuelle Maschine **RTEMS-de** verwendet.
Die Anmeldung erfolgt mit den gleichen es15***-Accounts wie in der
Windows-VM und die Daten bleiben diesesmal (hoffentlich) auch erhalten.

8.1 Vorbemerkungen (0 Punkte)

8.1.1 Dinge, die auch noch in den Orginalfolien standen

- Ursprünglich “Real-Time Executive for Missile Systems”
- Inzwischen “Real-Time Executive for Multiprocessor Systems”
- Open-Source Real Time Operating System mit Unterstützung mehrerer APIs, z.B. POSIX
- Findet u.a. Verwendung in der Raumfahrt, Medizintechnik, Netzwerktechnik u.a.
- Portabel auf diverse Architekturen, darunter ARM, PowerPC, x86, Blackfin, MIPS, Microblaze u.a.
- `http://www.rtems.org`

Features von RTEMS:

- Multitasking-fähig
- für homogene und heterogene Multiprozessorsysteme geeignet
- event-gesteuertes, prioritäts-basiertes, preemptives Scheduling
- optional rate-monotonic scheduling
- Inter-Task-Kommunikation und -Synchronisation
- Prioritätsvererbung
- reaktionsfähiges[1] Interrupt-Management
- dynamische Speicherallokation
- Hoher Grad von Konfigurierbarkeit

Gelegentlich nützliche Doku-URLs:

- `https://docs.rtems.org/doxygen/cpukit/html/modules.html`
- `https://docs.rtems.org/doc-current/share/rtems/html/`

8.2 Einrichten und Beispiel laufenlassen (3 Punkte)

8.2.1 Hardware vorbereiten

~~Da VirtualBox immer noch keine geschachtelte Virtualisierung unterstützt~~ Damit die Übung noch ein wenig näher an der Realität eingebetteter Systeme ist, wird das zu bauende System auf einem Raspberry Pi (Abbildung 1) mit Pibrella-

[1] responsive

Board ausgeführt werden und lediglich über eine serielle Schnittstelle kommunizieren, die mittels USB-Seriell-Wandler (Abbildung 2) mit dem PC verbunden wird.

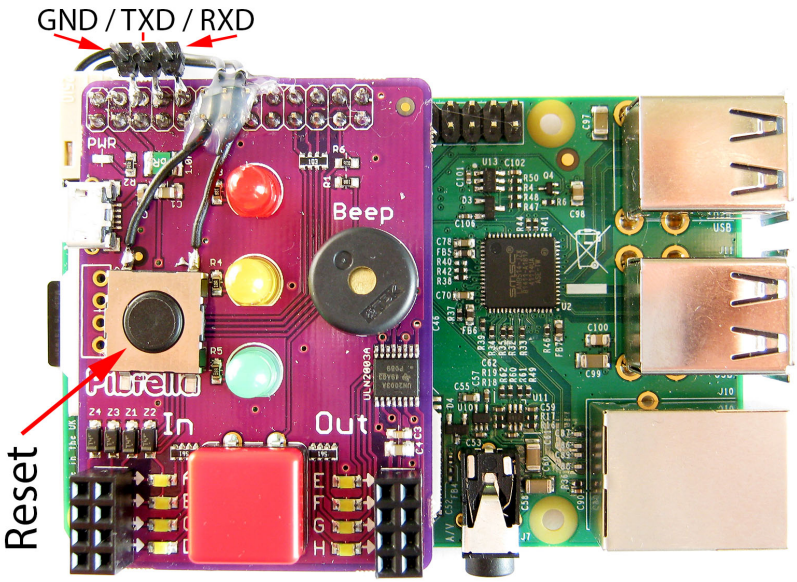


Abbildung 1: Raspberry Pi mit Pibrella (Aussehen des Reset-Tasters kann abweichen)

Zur Verbindung des Raspberry Pi und des USB-Seriell-Adapters dient die auf dem Pi aufgelötete dreipolige Stiftleiste, die in der Abbildung mit “GND/TXD/RXD” markiert ist. Diese wird mit dem kurzen bunten Kabel (Farben können abweichen) mit den GND/RXD/TXD-Pins des Seriell-Adapters verbunden, wobei jeweils der TXD-Anschluss (Transmit) des einen Geräts mit dem RXD-Anschluss (Receive) des anderen verbunden werden muss. Bitte achten Sie darauf, die Stiftleiste beim Aufstecken des Verbindungskabels nicht abzubrechen – die mechanische Belastbarkeit von Lötstellen ist ziemlich gering und Ihr Tutor hat keine Lust, den Anschluss an allen Boards durch eine mechanisch stabilere Variante zu ersetzen. Damit die Konstruktion nicht frei in der Luft hängt, sollte Ihnen ein USB-Verlängerungkabel für den USB-Seriell-Adapter zur Verfügung stehen.

Die Stromversorgung des Raspberry Pi erfolgt mittels USB-Kabel vom PC, wobei Sie entweder die im Bild sichtbare Mikro-USB-Buchse auf der Pibrella-Platine oder die Mikro-USB-Buchse des Raspberry Pi (im Bild verdeckt) verwenden können. Wenn der Raspberry Pi mit Strom versorgt wird, sollte auf ihm eine rote LED und auf dem Pibrella-Board eine blaue LED leuchte. Zudem sollte eine grüne LED neben der roten auf der Raspberry-Platine bei Druck auf den Reset-Taster kurz aufleuchten.

Achten Sie ggfs. darauf, dass die Mikro-SD-Karte korrekt im Slot eingelegt ist - sie sollte nur ca. 2mm über der Platinenkante herausstehen.

8.2.2 Entwicklungsumgebung vorbereiten

Zur Bearbeitung der Aufgaben müssen Sie nach der Anmeldung in der virtuellen Maschine den von uns vorbereiteten RTEMS-Quellcode in Ihr Home-Verzeichnis entpacken. Dies geschieht durch Eingabe des Befehls `rtems-setup` im

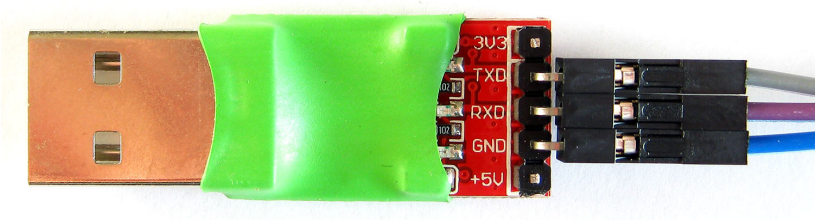


Abbildung 2: USB-Seriell-Adapter (Schrumpfschlauch-Farbe kann abweichen)

Terminal[2]. Nach Eingabe des Befehls sollte in Ihrem Home-Verzeichnis ein Unterverzeichnis `rtems` existieren, in dem sich wiederum die Unterverzeichnisse `rtems-gpio`, `build` und `install` befinden. Ersteres ist das Verzeichnis mit den Quelltexten, zweites das Verzeichnis in dem RTEMS compiliert wird und letzteres das Verzeichnis, in dem die compilierten Binaries abgelegt werden. Im Quellcodeverzeichnis (siehe Abbildung 3) befinden sich unter anderem folgende gelegentlich relevante Verzeichnisse:

`cpukit/score/src` Beinhaltet die gemeinsame Basisschicht für alle RTEMS-APIs (SuperCore)
`cpukit/rtems/src` Beinhaltet das klassische RTEMS-API
`testsuites/samples` Beinhaltet eine Reihe von Beispielapplikationen

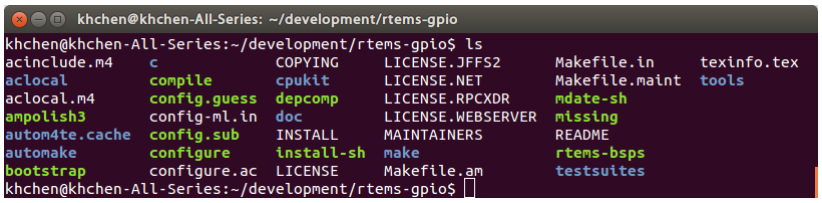


Abbildung 3: Terminal-Screenshot

Wechseln Sie nun in das `build`-Verzeichnis (`cd ~/rtems/build`) und starten Sie die Compilation mittels `make install`. Nach einigen Minuten Wartezeit sollte die Compilation abgeschlossen sein.

8.2.3 Hello World-Programm starten

Wechseln Sie in das Verzeichnis `~/rtems/build/arm-rtems4.11/c/raspberrypi/testsuites/samples/hello` und starten Sie von dort das Kommando `raspbootcom /dev/ttyUSB0 hello.ralf`. Das Tool `raspbootcom` ist ein kombiniertes Terminal- und Upload-Tool, welches mit dem Bootloader auf der Raspberry Pi-SD-Karte kommuniziert. Unmittelbar nach dem Start verhält es sich wie ein normales Terminalprogramm, d.h. es sendet Tastatureingaben per serieller Schnittstelle an den Raspberry Pi und zeigt dessen serielle Ausgaben an. Wenn es die Startnachricht des Bootloaders erhält, lädt es automatisch die angegebene Datei (hier: `hello.ralf`[3]) zum Raspberry hoch, der diese nach erfolgreichem Upload automatisch startet. Da raspbootcom immer die gerade aktuelle Version der Datei sendet, muss man während der Entwicklung lediglich eine neue Version seiner Applikation compilieren und die Reset-Taste auf dem Raspberry Pi betätigen, um diese sofort hochzuladen und zu testen.[4]

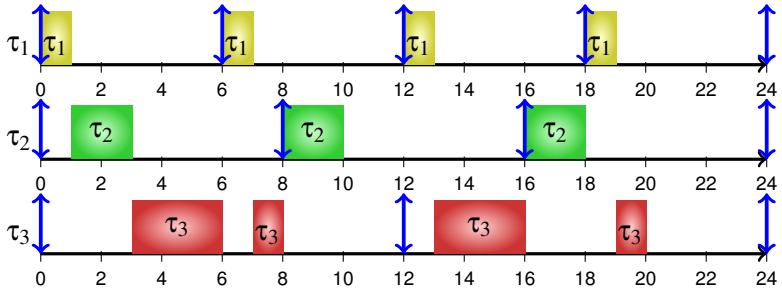
Der Quellcode des gerade gestarteten Beispielprogramms befindet sich in `~/rtems/rtems-gpio/testsuites/samples/hello/init.c`. In dieser übernimmt die Funktion `Init()` einen ähnlichen Zweck wie `main()` in einem normalen C-Programm. Am Ende von `init.c` befinden sich eine Reihe von #define-Anweisungen, die RTEMS für die Anwendung maßschneidern. Wenn eine Applikation mehrere Tasks benötigt, werden diese typischerweise in der `Init()`-Funktion gestartet und sie können sich mit der Systemfunktion `rtems_task_delete(RTEMS_SELF)` selbst wieder löschen. Ein Beispiel für eine Applikation mit mehreren Tasks können Sie in `~/rtems/rtems-gpio/testsuites/samples/ticker/` finden. Ein Blick in den RTEMS Applications C User's Guide (unter `https://docs.rtems.org/doc-current/share/rtems/html/`) kann auch lohnenswert sein.

[2] In den Übungen zu RTEMS wird ziemlich vieles im Terminal gemacht werden.
[3] Really Awesome Linked File oder so
[4] Die Alternative wäre, für jeden Test die MicroSD-Karte zwischen RasPi und PC umzustecken.

8.3 Rate-Monotonic Scheduling (7 Punkte)

Prioritätsdefinition: Ein Task mit einer kleinen Periode hat eine höhere Priorität, innerhalb welcher Gleichstände arbiträr aufgelöst werden. In RTEMS müssen die Prioritäten von Tasks definiert werden, wenn der Task erstellt wird.

Gegeben sein folgender Schedule:
$\tau_1 = (1,6,6)$, $\tau_2 = (2,8,8)$, $\tau_3 = (4,12,12)$. $[(C_i, T_i, D_i)]$



Verändern Sie das Hello World-Beispielprogramm so, dass es drei Tasks startet, welche ein Verhalten entsprechend dem obigen Diagramm auf der seriellen Schnittstelle darstellen.

Hinweise:

- Das Hello World-Programm konfiguriert RTEMS ohne Rate-Monotonic Scheduling, ohne Uhr und mit nur einem Task. Falls Sie Konfigurations-Direktiven aus anderen Beispielen wie z.B. dem Ticker-Beispiel übernehmen wollen[5], finden Sie diese im jeweiligen Verzeichnis in der Datei `system.h`.
- Rate-Monotonic Scheduling wird in RTEMS über den Rate Monotonic Manager konfiguriert, siehe `https://docs.rtems.org/doc-current/share/rtems/html/c_user/Rate-Monotonic-Manager.html`.
- In der Doku findet sich ein Anwendungsbeispiel, ausserdem verwendet as Beispielprogramm `testsuites/samples/SEMAPHORE_TEST` den RMM.
- Es empfiehlt sich zur besseren Lesbarkeit der Ausgaben, die Zeiten im Scheduling-Diagramm als Sekunden zu interpretieren. Der Umrechnungsfaktor von Sekunden zu RTEMS-Ticks ist mit der Funktion `rtems_clock_get_ticks_per_second()` abfragbar.
- Zur Simulation der aktiven CPU-Zeiten der Tasks eignen sich beispielsweise leere Schleifen.

Allgemeine Hinweise: Die Übungstermine und weitere Informationen finden Sie über
`http://ls12-www.cs.tu-dortmund.de/daes/de/lehre/lehrveranstaltungen/wintersemester-20152016/es-1516.html`

[5] empfehlenswert