AUFGABE 1: HALLO WELT

Diese Aufgabe dient dem initialen Einrichten Ihrer Entwicklungsumgebung und dem ersten Kontakt mit dem Echtzeitbetriebssystem *eCos* und dem Triboard. Ziel ist es einen ersten Einblick in die Möglichkeiten der Entwicklungsumgebung zu erhalten.

Fragestellungen, die für die Abgabe der Aufgabe von besonderem Interesse sind, setzen wir in dieser und den folgenden Übungsaufgaben kursiv. **Sie können also davon ausgehen, dass wir Ihnen diese Fragen bei der Abgabe der Übungsaufgaben stellen werden!**

Aufgabenstellung

1. *Vorbereitung:* Kopieren und entpacken Sie die Vorgabe in ein beliebiges Arbeitsverzeichnis und setzen Sie die nötigen Umgebungsvariablen durch den Aufruf von `source ecosenv.sh` Sie können den Inhalt der Datei auch in Ihre `~/.bashrc` einfügen, um den Vorgang zu automatisieren. Nun können Sie die Makefiles generieren und die noch funktionslose Anwendung erstmals kompilieren. Sie können die erzeugte Anwendung mit Hilfe des Lauterbach-Debuggers in den SRAM des Triboards laden und starten. Eine doxygen Dokumentation kann man mittels `make doc` erzeugen.

☞ `source ecosenv.sh`

☞ `cmake ..`
☞ `make`
☞ `make flash`

2. *Fadensystem:* Die Erzeugung eines geeigneten Threadsystems ist Bestandteil aller folgenden Übungsaufgaben.
Implementieren Sie in `hello.c` die `cyg_user_start()`-Funktion, in der Sie einen Thread mit Hilfe von `cyg_thread_create()` erzeugen. Die Threadfunktion soll zunächst periodisch, jede Sekunde, mittels `diag_printf()` die Zeichenkette „`Hallo Welt!\n`“ auf der seriellen Schnittstelle ausgeben. Sobald Sie den RS232-Ausgang des Triboards mit der seriellen Schnittstelle Ihres Rechners verbunden haben, können Sie sich die serielle Ausgabe folgendermassen anzeigen lassen:
`screen /dev/ttyS0 115200`.
Um einen periodischen Thread zu simulieren, bietet sich die Funktion `cyg_thread_delay()` an. Bei dem zu übergebenden Wert handelt es sich um Millisekunden. Die kleinstmögliche Verzögerung beträgt 2 ms
Welchen Nachteil hat es, periodische Ausführung auf diese Art umzusetzen? Haben Sie eine Idee für einen sinnvolleren Ansatz?

3. *Debugging:* Für das Echtzeitverhalten ist neben der eigenen Implementierung immer das Gesamtsystem zu betrachten. Um ein Gefühl hierfür zu bekommen, setzen Sie im Lauterbach-Debugger einen Breakpoint auf die Funktion `cyg_hal_plf_serial_putc()` und setzen Sie die Ausführung fort. Sobald die Ausführung an dem gesetzten Breakpoint gestoppt hat, können Sie die Aufrufhierarchie im Backtrace-Fenster betrachten.

Erkunden Sie den Aufrufgraph der Funktion `cyg_hal_plf_serial_putc()`. *Wieso ist diese Funktion für Ihr Programm relevant?*
Lassen Sie sich den Quellcode von `.._putc()` im Debugger anzeigen. *Welchen Zweck erfüllt die Funktion? Wie erbringt sie diese Leistung? Welche aus Echtzeitsicht kritischen Stellen fallen in der Funktion besonders auf? Die Verwendung welcher Art von Funktionen ist also aus Sicht eines Echtzeitsystems im Allgemeinen problematisch?*

4. *Signalerzeugung:* Mittels des periodischen Threads kann nun ein erstes Signal erzeugt werden. Sie können den GPTA des Tricore-Prozessors mit Hilfe der Funktion `ezs_dac_write()` als Digital-Analog-Wandler verwenden (Wertebereich 0 bis 2^8-1) und sich das erzeugte Signal am Oszilloskop anschauen. Verbinden Sie hierzu die Masseklemme des Tastkopfes mit dem Massepin der Filterplatine. Nun können sie mit Hilfe der Klemmprüfspitze das ungefilterte und das gefilterte Signal an den entsprechnden Pins ansehen. Erzeugen Sie ein Sinussignal und verwenden Sie für diesen Zweck die Funktion `sinf()`. *Welche Parameter sind bei einer sinusförmigen Schwingung von Interesse?*[1] ☞ `math.h`
Verfolgen Sie bei Ihrer Implementierung einen systematischen Ansatz bei der Erzeugung des Signals und sorgen Sie dafür, dass die relevanten Parameter der Sinusschwingung flexibel einstellbar sind. Verringern Sie nun die Periode des Fadens drastisch (z. B. 2 ms). *Was beobachten Sie? Wie müssen Sie Abtastrate und Parameter des Sinus wählen um ein Signal einer bestimmten Frequenz zu erzeugen?*
Beaufschlagen Sie nun die Periode Ihres Signalerzeugungsthreads mit einem *Jitter* und experimentieren Sie mit verschiedenen Werten für dessen Größe. *Was beobachten Sie?*
Betrachten Sie abschließen die Funktion `ezs_dac_write()` im Debugger. *Wie unterscheidet sie sich von* `cyg_hal_plf_serial_putc()` *im Bezug auf das Echtzeitverhalten?*

Hinweise

- Bearbeitung: Gruppe mit je drei Teilnehmern.
- Abgabezeit: ab 29.10.2015
- Fragen bitte an `i4ezs@lists.cs.fau.de`

[1] `https://de.wikipedia.org/wiki/Schwingung#Harmonische_Schwingung`