Praktikum 6 – Function Pointer, Structs, Arrays

Dauer: 2 Lektionen

Aufgabe: Scheduler

Schreiben Sie ein C-Programm, welches es ermöglicht, definierte Task's nacheinander abzuarbeiten. Dazu kann jedem Task eine Intervalzeit und eine entsprechende Funktion zugewiesen werden. Die Intervalzeit, ist die Zeit, in der die zugewiesene Funktion wiederholt aufgerufen wird. Die Intervalzeitangabe wird in Ticks angegeben. Studieren Sie den vorgegebenen Code und versuchen Sie zu erklären, wie die Ticks mit der wirklichen Zeit (s, ms, us) in Beziehung steht und von was sie abhängig ist.
Damit der Task weiss, welche Funktion im zugewiesen ist muss diese zuvor dem Scheduler mittels Pointer auf die entsprechende Funktion bekannt gegeben werden.
Spielen Sie mit den Intervalzeiten beim Aufruf der Funktion *addTask* ein wenig und versuchen Sie zu erklären warum und wo ein unerwarteter System-Interrupt Probleme verursachen könnte. Beachten Sie zudem, dass die ganzen Timer vom Betriebssystem abhängig sind und bei einer Portierung eventuell anders implementiert werden müssten. Überlegen Sie sich, wie die Timer auf einem Mikroprozessor ohne Betriebssystem ersetzt werden könnten.

```
./scheduler

Task "TaskA" added.
Task "TaskB" added.
Task "TaskC" added.
Trigger Task "TaskA" at 10ms
Task A is running...
Trigger Task "TaskA" at 20ms
Task A is running...
Trigger Task "TaskB" at 20ms
Task B is running...
Trigger Task "TaskA" at 30ms
Task A is running...
Trigger Task "TaskA" at 40ms
Task A is running...
Trigger Task "TaskB" at 40ms
...
```

Hinweise:

- Die Standard IO Funktion *printf()* wurde mittels *#define* substituiert. Dies erlaubt das Programm flexible zu halten und das Portieren auf andere Architekturen zu vereinfachen.
- Das Define *SYSTEM_TICK_MS* definiert, wie viele System Ticks in einer Millisekunde vorhanden sind.
- Das Struct *task* in der Datei *scheduler.h* definiert ein Task.
- Implementieren Sie die Funktion *addTask()* so, dass sie ein Element im Array *tasks[32]* mit den entsprechenden Werten abfüllt. Verwenden Sie die String Funktion *strcpy()*, um den Funktionsnamen zu kopieren. (Geben Sie in der Konsole *man strcpy* ein, um mehr über die Funktion zu erfahren).
- Die Funktion *sysTick()* wird bei jedem System Tick automatisch vom Timer aufgerufen. In dieser müssen Sie nun den eigentlichen Scheduler implementieren. Eine Funktion über einen Pointer aufrufen, können Sie mit folgendem Code bewerkstelligen:

  ```
  void (*func)();
  func = tasks[i].funcPtr;
  (*func)();
  ```

- Die Funktion *initSysTick()* initialisiert den benötigten Timer und sorgt dafür, dass die Funktion *sysTick()* wie gewünscht aufgerufen wird.