Programmendwurf

Systemnaheprogrammierung

Task-Verwaltung

Matrikelnummer 1

1641654

TINF 17 B

3. Semester

2018

# Problembeschreibung

Auf einem System mit dem Prozessor Infineon C517a (ohne ROM) mit einer Prozessorfrequenz von 24MHz soll ein Scheduler programmiert werden. Dieser Scheduler soll in der Programmiersprache Assembler geschrieben werden und die Funktion eines Task-Verwaltung-Systems übernehmen. Als Task-Verwaltung-System ist dabei definiert, dass das Programm Prozessorzeit auf unterschiedliche Prozesse verteilen soll. Diese Prozesse sollen dabei unterschiedliche Prozessorzeit zugewiesen bekommen und sie benutzen. Das Programm soll dazu in der Lage sein, neue Prozesse beziehungsweise Kinderprozesse von Prozessen (???) zu starten. Des Weiteren soll der Scheduler in der Lage sein, andere Prozesse zu beenden. Einiger dieser Prozesse sollen dafür implementiert werden. Zum einen ist da der Konsolen Prozess, welcher ständig Befehle/ Zeichen von der seriellen Schnittstelle 0 mithilfe des Verfahrens Polling. Dabei sind die Möglichen Befehle die Zeichen ’a’, ’b’, ’c’, ’z’. Dabei startet das Zeichen ’a’ den Ausgabe Prozess A, ’b’ den Ausgabe Prozess B, welcher durch das Einlesen des Zeichens ’c’ beendet wird. Das Zeichen ’z’ startet und beendet einen Text Prozess. Dabei ist zu beachten, dass das Programm vollkommen gestoppt werden soll, falls das Programm bereits gestartet wurde, und nicht wieder dort anfangen soll wo es aufgehört hat. Dieser Prozess ist gegeben und soll nicht programmiert werden. Hingegen sollen die Prozesse A und B neben dem Consolen Prozess programmiert werden. Prozess A soll dabei die Sequenz ’uvwxyz’ über die Serielle Schnittstelle 0 ausgegeben werde und das Programm soll dann sich selber beenden. Der Ausgabe Prozess B gibt das Zeichen ’\*’ alle ¾ Sekunde über die Serielle Schnittstelle 0 aus.

# Fragen:

## a) Wie behandeln sie den doppelten Aufruf eines Prozesses solange dieser aktiv ist?

## (z. B. bei der Befehlsfolge: ’b’, ’b’)

Bei Erkennen der Eingabe ’b’ wird darauf überprüft ob bereits der Prozess bereits gestartet wurde. In dem Fall, dass der Prozess B schon gestartet wurde wird kein weiterer Prozess gestartet, sondern die Eingabe ignoriert.

## b) Wie reagiert ihr Programm auf die Benutzung der Seriellen Schnittstelle 0 durch mehrere Prozesse?

## (Z.B.: Können sich die Prozesse blockieren? Können ”gesendete“ Zeichen ”verschwinden“?)

Programme, welche die Serielle Schnittstelle benutzen setzten im RAM an eine bestimmte Stelle (0x8f) ein Bit. Dieses beschreibt ob die Serielle Schnittstelle in Benutzung ist. Jedes Programm, welches eine Ausgabe auf derselben Schnittstelle tätigen möchte überprüft zuerst ob die Schnittstelle frei ist, in dem es die Flag im RAM überprüft. Gesetzt bedeutet dabei, dass die Schnittstelle in Benutzung ist.

## c) Wie haben sie die Anforderung der unterschiedlichen Rechenzeit gelöst?

Jedes Programm besitzt einen eigenen Bereich im RAM. In diesem steht unter anderem auch seine Priorität. Die Priorität gibt an, wie viele Zeitscheiben dieser Prozess bei jedem Umlauf erhält. Die Zeitscheiben werden in einem weiteren Bereich gespeichert. Sobald eine Zeitscheibe abgelaufen ist reduziert der Scheduler die restlichen Zeitscheiben des Prozesses. Sobald keine mehr vorhanden sind erhält der nächste Prozess seine Zeitscheiben und wird gestartet beziehungsweise fortgeführt (mit Wechseln von Registern usw.).

# Kontroll-Ausgaben

## os1.inc

uvwxyz\*\*\*\*uvwxyz\*uvwxyz\*\*\*\*\*\*uvwxyz\*\*\*

## os2.inc

uvwxyzDHBW@Stuttgart\_uvwxyzDuvwxyz

## os3.inc

DHB\*\*\*\*W\*\*\*\*\*@\*\*\*\*S\*\*\*\*\*t\*\*uvwxyz\*\*\*\*\*\*\*\*uvwxyzD\*\*\*HBuvwxyz

# Programmbeschreibung

Das Programm ist in mehrere Teile unterteilt. Dabei