AUFGABE 7: BETRIEBSMITTELPROTOKOLLE

In dieser letzten Übungsaufgabe werden Sie sich mit gegenseitigem Ausschluss und Zugriffskontrolle befassen. Diese Übungsaufgabe zielt auf die Probleme der Prioritätsumkehr und der Verklemmung ab und dient dazu die in der Vorlesung vorgestellten echtzeitfähigen Synchronisationsprotokolle (vgl. VII 11 ff.) praktisch anzuwenden.

Auch in dieser Übung greifen wir der Einfachheit halber auf die folgenden synthetischen Aufgabensysteme zurück:

Tabelle 1: Aufgabensystem 1 – Prioritätsumkehr

Aufgabe	**Periode** ms	**Phase** ms	**WCET** ms	**Betriebsmittel**[1]
T_1	20	4	6	$(R_1, 3, 1)$
T_2	50	3	4	
T_3	200	1	9	$(R_1, 1, 7)$

Tabelle 2: Aufgabensystem 2 – Transitive Blockung

Aufgabe	**Periode** ms	**Phase** ms	**WCET** ms	**Betriebsmittel**[1]
T_4	20	7	2	$(R_2, 1, 1)$
T_5	50	5	6	$(R_2, 1, 5)$ $(R_3, 3, 2)$
T_6	100	3	6	$(R_3, 1, 5)$ $(R_4, 4, 2)$
T_7	200	1	10	$(R_4, 1, 9)$

[1] Notation Betriebsmittel: (Betriebsmittel, Anforderungszeitpunkt, Haltezeit)

Tabelle 3: Aufgabensystem 3 – Verklemmung

Aufgabe	**Periode** ms	**Phase** ms	**WCET** ms	**Betriebsmittel**[1]
T_8	20	3	6	(R_5, 1, 5) (R_6, 4, 1)
T_9	50	8	12	
T_{10}	200	1	6	(R_6, 1, 5) (R_5, 5, 1)

Implementierungshinweise:

1. Nutzen Sie die in der Vorgabe enthaltenen Vorlagen und implementieren Sie die o. g. Aufgabensysteme in separaten Dateien.
2. Vergeben Sie die Prioritäten nach dem RMA, simulieren Sie die WCET wie angegeben.
3. Nutzen Sie das von eCos bereitgestellte Mutexkonzept zur Implementierung der Betriebsmittel.
4. Belassen Sie die Lösungen der vorangegangenen Teilaufgaben deaktiviert im Code für die spätere Abgabe.

1 Aufgabenstellung

Teilaufgabe 1.
Welche Bedingungen müssen erfüllt sein, damit es zu einer Verklemmung (engl. Deadlock) kommen kann? Wie können Verklemmungen vermieden bzw. aufgelöst werden?

Machen Sie sich bei den nachfolgenden Aufgaben zunächst mit Stift und Papier (oder einer anderen Darstellungsmöglichkeit Ihrer Wahl) klar, was bezüglich belegter Betriebsmittel, blockierter Aufgaben, Prioritäten etc. passieren sollte. Implementieren Sie die Aufgabe anschließend wie beschrieben.

1.1 Verdrängungssteuerung

Diese Teilaufgaben beziehen sich auf Aufgabensystem 1:

Teilaufgabe 2.

Welche maximale Blockadezeit (für die höchstpriore Aufgabe) erwarten Sie und wie lässt sie sich messen?

Teilaufgabe 3.

Können Deadlocks auftreten?

Teilaufgabe 4.

Welche Nachteile erfahren unbeteiligte Aufgaben (die kein Betriebsmittel nutzen)?

Teilaufgabe 5.

Nutzen Sie nun das Konzept der Verdrängungssteuerung (NPCS) zur Synchronisation von Aufgabensystem 1 in eCos.

☞ `cyg_scheduler_lock()`
☞ `...unlock()`

1.2 Prioritätsvererbung

Diese Teilaufgaben beziehen sich auf Aufgabensystem 2:

Teilaufgabe 6.

Welche maximale Blockadezeit (für die höchstpriore Aufgabe) erwarten Sie und wie lässt sie sich messen?

Teilaufgabe 7.

Können Deadlocks auftreten?

Teilaufgabe 8.

Welche Nachteile erfahren unbeteiligte Aufgaben (die kein Betriebsmittel nutzen)?

Teilaufgabe 9.

Verwenden Sie Prioritätsvererbung (PI) um Aufgabensystem 2 in eCos umzusetzen. Wählen Sie hierfür bei der Initialisierung der Mutexobjekte `CYG_MUTEX_INHERIT` als Protokoll aus.

☞ `cyg_mutex_set_protocol()`

1.3 Prioritätsobergrenzen

Diese Teilaufgaben beziehen sich auf Aufgabensystem 3:

Teilaufgabe 10.

Welche maximale Blockadezeit (für die höchstpriore Aufgabe) erwarten Sie und wie lässt sie sich messen?

Teilaufgabe 11.

Können Deadlocks auftreten?

Teilaufgabe 12.

Welche Nachteile erfahren unbeteiligte Aufgaben (die kein Betriebsmittel nutzen)?

Teilaufgabe 13.

Setzen Sie als nun noch die Prioritätsobergrenzen (PCP) ein um Aufgabensystem 3 in eCos zu realisieren. Wählen Sie hierfür bei der Initialisierung der Mutexobjekte `CYG_MUTEX_CEILING` als Protokoll und konfigurieren Sie die Prioritätsobergrenzen der Betriebsmittel passend – beachten Sie hierbei, dass die Priorität des Betriebsmittels um eins höher sein muss als die des höchstprioren verwendenden Fadens und nicht, wie eigentlich üblich, gleich der Fadenpriorität. Zeichnen Sie den Ablauf mittels Tracer auf und überlagern Sie diese Aufzeichnung mit der *vermuteten* Systemobergrenze. Welche Variante von PCP ist in eCos implementiert?

Teilaufgabe 14.
Welche Variante von PCP ähnelt in Bezug auf ihr Verhalten zur Laufzeit eher NPCS? Welche eher PI?

Teilaufgabe 15.
Wieso kommt es zu keiner Verklemmung?

2 Erweiterte Aufgabe

Teilaufgabe 16.
Implementieren Sie nun noch die anderen Kombinationen (jedes Aufgabensystem mit jeder Synchronisationstechnik). Messen Sie wiederum die Blockadezeit.

Teilaufgabe 17.
Was fällt Ihnen insbesondere bei den Aufgabensystemen 2 und 3 auf?

---

3 Bonusaufgabe

Bei dieser Aufgabe handelt es sich um ein aktuelles, bisher nicht zufriedenstellend gelöstes Problem bei der Umsetzung des I4Copters. Die Bearbeitung dieser Aufgabe ist **freiwillig**. Bei erfolgreicher Bearbeitung (textuell oder Implementierung) erhalten Sie fünf Euro Guthaben für die I4-Kaffeekasse – das entspricht **20 Tassen Kaffee**.

Hintergrund: Die Belegung eines Betriebsmittels ist ein kritischer Abschnitt. Normalerweise macht es keinen Sinn, dass eine Aufgabe in einem kritischen Abschnitt die Kontrolle über die CPU abgibt, da sie den Abschnitt ja so schnell wie möglich wieder verlassen möchte bzw. sollte. Die vorgestellten Synchronisationsmechanismen unterstützen die Aufgabe bei diesem Vorhaben indem sie beispielsweise ihre Priorität anheben.

Problembeschreibung: Der I4Copter besitzt einen seriellen SPI-Bus (Betriebsmittel) über den drei Aufgaben (T_A, T_B, T_C) unterschiedlicher Priorität Nachrichten versenden. Senden und empfangen erfolgt beim SPI gleichzeitig, der Bus ist also *synchron*. Die Hardware übernimmt den eigentlichen Versand und signalisiert den Empfang mittels eines Interrupts. Die Aufgaben müssen also lediglich eine Bustransaktion aufsetzen, die Nachricht an eine beliebige Stelle im Speicher legen und bis zum Empfangsinterrupt warten. Sie benötigt die CPU also nicht, es muss aber verhindert werden, dass eine andere Aufgabe eine neue Transaktion aufsetzt – dies würde zum Abbruch der aktuellen Transaktion führen.

Forderungen:

1. Der Faden gibt die Kontrolle über die CPU während der Übertragung ab. Aktives Warten findet nicht statt.

2. Unbeteiligte, d. h. niederpriore Fäden sollen laufen dürfen.

3. Nachrichten sollen mit der Fadenpriorität versendet werden.

4. Kein extra Faden für die Buskoordinierung.

Hinweise: Die klassischen Verfahren versagen hier. Obwohl sie die Nachrichtenpriorität berücksichtigen, ist das Problem, dass sie niederpriore Fäden ausschließen und die CPU dadurch u. U. untätig bleiben würde.

Teilaufgabe 18.
Falls Sie eine Lösung für dieses Problem haben, wenden Sie sich bitte an Peter Ulbrich.

Hinweise

- Bearbeitung: Gruppe mit je zwei/drei Teilnehmern.
- Abgabezeit: 04.02.2016
- Fragen bitte an `i4ezs@lists.cs.fau.de`