Übung#3- RTOS

Einführung:

Die Grundidee der Übung ist Demoprogramme für das Echtzeitbetriebssystem RTOS zu realisieren, die **reale Anwendungen simulieren**. Jedes Demoprogramm soll **Threads** erzeugen und Synchronisationsmechanismen (**Semaphore, Mutexe, Events**) verwenden. Das Demoprogramm soll auf dem HTL eigen Mikrocontrollersystem auf Basis des Cortex M3 Mikrocontroller lauffähig sein. Der Zugriff auf die Peripherieeinheiten (GPIO, ADC, Timer, UART,….) des Mikrocontrollers soll über die **Standard Peripherial Library** der Entwicklungsumgebung Keil µVision erfolgen. Zustandsänderungen im System sollen über UART protokolliert werden und können mit einem Terminalprogramm angesehen werden. Über dieses **Logging** sollte der Funktionsbeweis geführt werden.

Hinweise findet zu RTOS findet ihr im RTOS API Tutorial. (Im Untericht besprochen)
http://www.keil.com/pack/doc/CMSIS/RTOS2/html/index.html

Speziell für Synchronisationsprobleme ist das ebook „**Little book of Semaphore**“ zu empfehlen. Befindet sich unter Dateien in der MS-Teams Gruppe.

Abgabe:

Abzugeben ist ein Protokoll (*.pdf bzw. *.doc) welches die **Aufgabestellung** beschreibt. Die Aufgabe erläutert eine reale Situation für das gestellte Thema bzw. wie diese auf dem ARM-Minimalsytem simuliert wird. Desweitern ein **Blockschaltbild** zur Aufgabenstellung sowie den dokumentierten Source Code. Anhand des Blockschaltbilds soll die Funktionsweise des Systems erläutert werden, sodass man in der Lage ist den Source Code zu verstehen. Schließlich soll das Programm einen Funktionsnachweis in Form von **Screenshots** mit entsprechenden Erläuterungen enthalten die nachweisen das Aufgabenstellung erfüllt ist.

Die Abgabe erfolgt über die entsprechende MS-Teams Gruppe. Alle Files (Quelldateien, ausführbare Dateien, Dokumente) sind in ein Archiv Datei (*.ZIP) zu geben, welches das bei der entsprechenden MS-Teams Aufgabe hochzuladen ist.

Team#1: Mottl / Pruggmayer: (Demoprogramm Messwertvisualisierung – Erzeuger / Verbraucher Problem)

Idee: Es soll eine **nicht blockierende** Version der SvVis Library implementiert werden, die eine Messwertvisualisierung für Echzeitsysteme erlaubt. Visalisierung der Messdaten (Logging) über ein Acquisition on Event ausgelöst werden können. Die Messdaten werden in einen Ringbuffer geschrieben , von wo sie von einem Sende Thread abgeholt werden, der sie dann über den UART zu einem PC sendet. Der PC kann auch Kommandos an des Messsystem Senden, die entsprechend interpretiert werden. (z.B.: ACQ_On, ACQ OFF). Details sind noch mit Betreuer zu klären. Das System soll auch funktionieren, wenn PC sehr langsam ist (niedrige Baudrate).

Team#2: Lohwasser/Strunz: (Demoprogramm Multiplex – Tutorial RTOS)

Simulation einer realen Situation für Multiplex (z.B.: Befüllen einer Parkgarage, Busy

z.B: Befüllen eine Parkgarage oder Busy Nightclub. In der Parkgarage stehen nur N Plätze zur Verfügung bzw. in Nachtclub können nur N Leute aus Sicherheitsgründen eintreten

Team#3: Marx/Strunz: (Demoprogramm Barrier - Tutorial RTOS)

Simulation eine realen Problems für eine Barrier Turnstile (Drehweiche)
z.B:: Simulation des Beladens einer Seilbahn einer Schistation. Viele Leute stürmen zur Seilbahn. Wenn N Leute eingetroffen (Beladung vollständig) startet Seilbahn und transportiert sie zur Bergstation. Wenn Seilbahn wieder zum Tal gekommen ist kann die Befüllung wieder beginnen.

Team#4: Meichenitsch T/Stella: (Demo Programm Multiple Instances - Tutorial RTOS)

Ein Demoprogramm für Multiple Thread Instances mit ARM Minimalsystem soll realisiert werden.

z.B.: Ein Thread (Worker) soll in der Lage sein eine Text über einen UART wegzusenden. Über einen Thread Parameter soll gesteuert werden können, ob Text über UART1 oder UART2 weggesendet wird. Von diesem Thread können N Instanzen laufen. Zugriff auf UART muss synchronisiert werden.– In welcher realen Situation könnte das benötigt werden ?

Team#5: Platjas/Roll: (Demoprogramm Rendezvous - Tutorial RTOS)

Ein Demoprogramm für ein Rendezvous mit ARM Minimalsystem soll realisiert werden.

z.B: Weiterverarbeitung von Messdaten kann erst dann erfolgen wenn Messwert von ADC Ch1 und Messwert von ADC Kanal 2 eingetroffen sind. (Simulation über die 2 Potentiometer des ARM Systems)

Team#6: Kopper/Brenninger: (Demoprogramm Shared Memory – Tutorial RTOS)

Ein Demoprogramm für einen **Datenaustausch zwischen Threads über einen „Shared Memory“** soll mit ARM Minimalsystem soll realisiert werden. Der Sharded Memory liegt im Memory Pool des RTOS Betriebssystem. Zugriff auf den shard Memory Bereich muss natürlich synchronisiert werden.

Idee: Ein Messprozess liest Messdaten von einen ADC Kanal ein. Die Messdaten werden in den gemeinsamen „Shared memory“ Bereich anderen zur Verfügung gestellt. Dieser holt die Daten zur Weiterarbeitung (z.B. Regelungprozess) von dort ab und steuert ein Stellglied (simuliert durch PWM Signal welches Helligkeit steuert)

Team#7: Raffer/Grubmüller: (Prioritäten, Events, – Tutorial RTOS, RTOS Theorie of Operation - Scheduler)

Ein Demoprogramm für Prioritätensteueung und Events und Mutex soll realisiert werden.

Idee: Es gibt es 4 Threads (Thread1 prio normal, Thread2 prio normal, Thread3 prio above normal, Thread 4 high priority). Nach dem Start arbeiten nur Thread 1 und thread 2. Thread 3 und 4 sind im blocked Zustand. Thread 2 weckt Thread3 durch einen Event auf. Scheduler wechselt zu Thread 3. Durch einen Interrupt wird Thread4 aufgeweckt. Scheduler wechselt zu Thread4 . Nach erledigter Arbeit geht Thread 4 wieder schlafen. Thread 3 geht ebenso nach erledigter Arbeite wieder schlafen. So bin ich wieder im Ausgangszustand.

Team#8: Meichenitsch N/ Hasenzagl: (Demoprogramm Joinable Threads, Semaphore Events – Tutorial RTOS)

Ein Demoprogramm für Joinable Thread und Events soll realisiert werden.

Idee: to be defined

Team#9: Reim/Kilincarslan: (Demoprogramm River Crossing Problem)

Ein Demoprogramm für das klassische Synchronisationsprobleme – „River Crossing Problem“ soll realisiert werden. Details dazu findet ihr z.B. in dem ebook „ Littele book of Semaphores“ in Kapitel 5.7.

Team#10: Baldauf/Radu: (Demoprogramm - Zero Mailbox)

Ein Demoprogramm für eine Zero Mailbox soll realisiert werden. Der Speicher für die Nachrichten soll vom Memory Pool bereit gestellt werden. Übertragen werden sollen über die Mailbox nur die Pointer auf die Messages.