µC/OS-II FLASHPORTIERUNG FÜR SK167

Peter Balog

Fachhochschule Technikum Wien

ZUSAMMENFASSUNG:

µC/OS-II (MicroC/OS-II) ist ein optimierter, stark skalierbarer Multitasking- / Echtzeit-betriebssystemkern speziell für Mikrocontrolleranwendungen. Im Rahmen von Embedded Systems relevanten Lehrveranstaltungen unserer Fachhochschule-Studiengänge aus dem Bereich der Informations- und Kommunikationstechnologie verwenden wir C167-basierte Hardwareplattformen und die Keil µVision2 Softwareentwicklungsumgebung. Für dieses Umfeld haben wir eine µC/OS-II Flash-Portierung realisiert, um die darauf aufsetzenden Softwareapplikationen in ihrem Speicherbedarf zu minimieren.

MOTIVATION

Als Hardwareplattform steht primär das *Phytec SK-167CR Starterkit* (*KitCon*) zur Verfügung. Für die Softwareentwicklung soll die freie Evaluierungsversion der Keil µVision2 Softwareentwicklungsumgebung (*Keil-IDE*) ausreichen. Die Evaluierungsversion bietet den vollen Funktionsumfang; jedoch ist die Größe einer Applikation auf maximal 8kByte Speicherbedarf beschränkt. Wird der Betriebssystemkern, wie üblich, in die Applikation eingebunden, so wird dieses Speicherlimit selbst bei kleinen Anwendungen sehr bald überschritten.

Deshalb wurde eine flashbare Version des portierten Betriebssystems inklusive eines Rahmenprojekts (*Application Skeleton*) für die Keil-IDE entwickelt. Die Vorteile dieser Umgebung sind:

- Es kann die frei verfügbare Evaluierungsversion der Keil-IDE verwendet werden.

- Die ausführbaren Applikationen bleiben „klein“, wodurch die *Download*-Zeiten fürs *Remote-Debugging* verkürzt werden.

Diese Umgebung ist lediglich fürs *Remote-Debugging* von Anwendungen ausgelegt. Um flashbare (ROM-fähige) Applikationen zu erzeugen, muss auf die Vollversion der Keil-IDE zurückgegriffen werden.

µC/OS-II FLASHPORTIERUNG

Basis für die vorliegende Flash-Portierung war der Betriebssystemkern µC/OS-II Rel. 2.51 von Jean J. Labrosse /1/ und die Version 1.0 der C167-Portierung /2/ für die Keil-IDE von Martin Rodier. Zur Generierung der Flash-Portierung wurde eine lizenzierte Vollversion der Keil-IDE für C167 Prozessoren (µVision2 V2.12, C Compiler V4.11, Assembler V4.11, Linker/Locator V4.10, Hex Converter V4.01) verwendet.

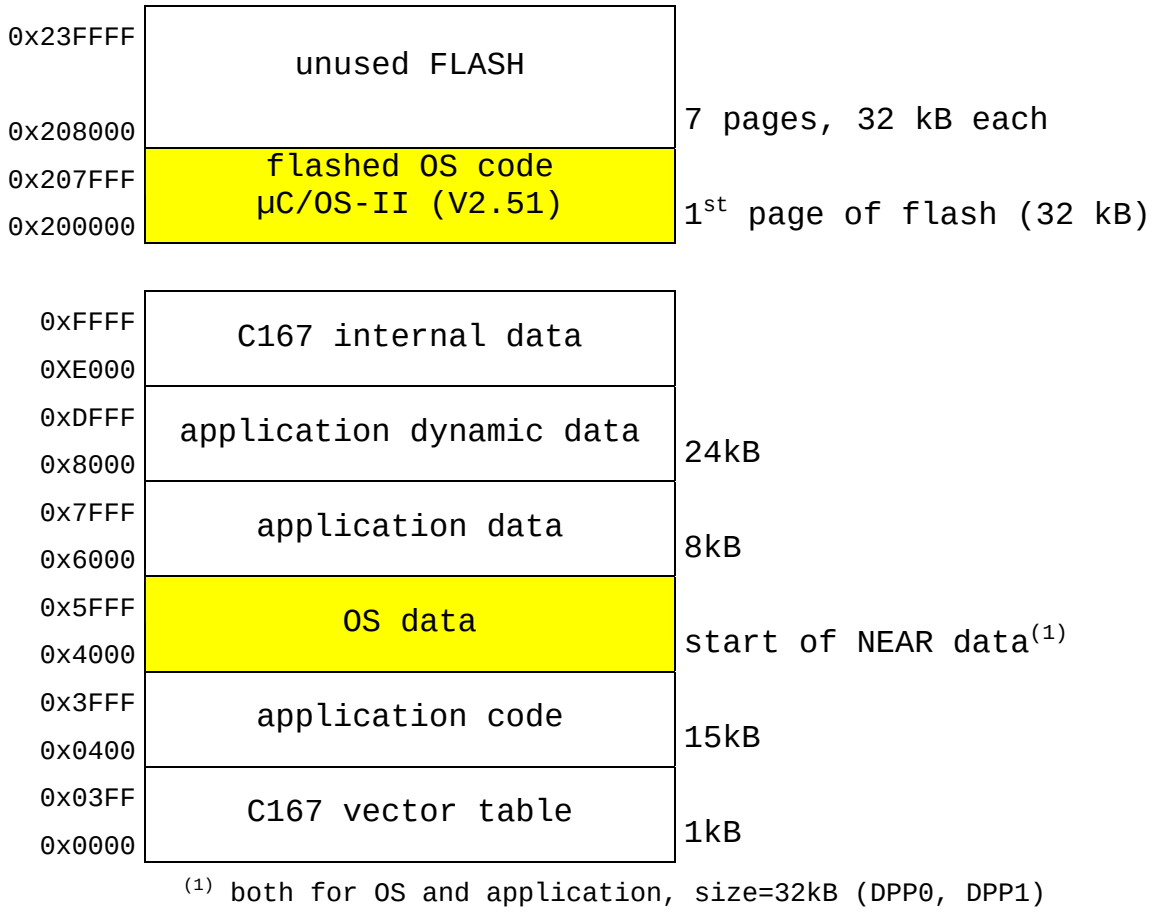
SPEICHERMODELL

Es wird von einem *Phytec SK-167CR Starterkit* in Grundausstattung und der üblichen Monitorkonfiguration für *Remote-Debugging* mit der Keil-IDE ausgegangen:

- 64kB RAM, Adressbereich: 0x0000-0xFFFF
- 256kB Flash, Adressbereich: 0x200000-0x23FFFF

Die Speicherkonfiguration des flashbaren Betriebssystems und die Speicherkonfiguration für Anwendungen müssen sorgfältig aneinander angepasst werden. Dies geschieht in den *Options for Target* des *Application-Projects* innerhalb der Keil-IDE (siehe *Application Skeleton*).

Das Speicherlayout der folgenden Abbildung zeigt die physikalische und logische Aufteilung des Hauptspeichers. Im Adressbereich von 0x0000-0xFFFF liegen die 64kB RAM-Speicher. Nutzbar ist dieser Speicher allerdings nur bis zur Adresse 0xDFFF. Im Bereich von 0xE000 bis 0xFFFF (insgesamt 8kB) werden die internen Speicherbereiche (Register, Special Function Registers, Stack, ...) des Mikrocontrollers C167 eingeblendet.



Im Adressbereich von 0x200000 bis 0x23FFFF liegt der 256kB große Flash-Speicher. Dieser Speicher ist intern in 8 Seiten zu je 32kB organisiert. Die flashbare µC/OS-II Portierung wird in die erste Seite des Flash-Speichers programmiert.

Der RAM-Speicher (0x0000 - 0xDFFF) wird logisch zwischen dem Mikrocontroller (Interrupt Vektor Tabelle, 0x0000-0x03FF), dem Betriebssystem und der Applikation aufgeteilt. Das Betriebssystem benötigt für interne Datenstrukturen den Speicherbereich von 0x4000 bis 0x5FFF (8kB). Diese Einstellung ist „hart“ in der flashbaren µC/OS-II Portierung kodiert.

Die Applikation besteht aus dem Programmcode (ab 0x0400 bis maximal 0x3FFF), den Daten (ab 0x6000 bis maximal 0x7FFF) und den dynamischen Daten (ab 0x8000 bis maximal 0xDFFF). Dynamische Daten werden zur Laufzeit der Applikation angelegt. Die Verwaltung der dynamischen Applikationsdaten wird vom Speichermanagement des Betriebssystems übernommen, welches dafür Funktionen zur Verfügung stellt (`OSMemCreate()`, `OSMemGet()`, etc.). Die Stackbereiche der Applikationstasks liegen typischer Weise in einer solchen dynamischen Speicherpartition (siehe *Application Skeleton*). Alle zur Laufzeit

erzeugten Datenobjekte „belasten“ nicht die maximale Größe von 8kB für eine Applikation welche mit der Evaluierungsversion der Keil-IDE entwickelt werden kann.

Die Datenbereiche des Betriebssystems (0x4000 - 0x5FFF) und der Applikation (0x6000 - 0x7FFF) liegen hintereinander und bilden aus Sicht des Speichermodells des Applikations-*Projects* (der Keil-IDE) einen 16kB großen, „near“ adressierbaren Datenbereich beginnend mit der Adresse 0x4000.

KONFIGURATION DER µC/OS-II FLASHPORTIERUNG

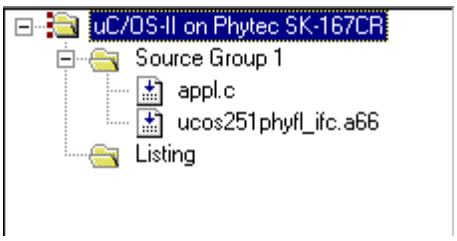
Da das Einsatzgebiet von µC/OS-II im Bereich der Mikrocontroller basierten Embedded Systems liegt, ist es hochgradig skalierbar, um den vom Betriebssystem belegten Programm- und Datenspeicher zu minimieren (*footprint optimization*). Die Konfiguration geschieht in der OS_CFG.H Datei des Betriebssystems welche im Zuge der Applikationsentwicklung geeignet angepasst wird. Beim *build*-Prozess werden dann nur jene Teile des Betriebssystems mitgebunden, die tatsächlich von der Applikation benötigt werden.

Event Control Blocks	32
Event Flag Groups	8
Memory Partitions	4
Queue Control Blocks	8
Number of Tasks	32
Lowest Task Priority	63
Idle Task Stack Size	512 (16 bit entries)
Statistic Task	enabled
Statistic Task Stack Size	512 (16 bit entries)
Argument Checking	enabled
µC/OS-II hooks in the port	present
Event Flags	fully supported
Message Mailboxes	fully supported
Memory Management	fully supported
Mutual Exclusion Semaphores	fully supported
Message Queues	fully supported
Semaphores	fully supported
Task Management	fully supported
Time Management	fully supported
Code for Schedule Lock / Unlock	present
Ticks per Second	250
Datatype for Event Flags	INT16U (i.e. 16 bit wide)

Bei der vorliegenden Flash-Portierung des Betriebssystems musste vorab eine Konfiguration gefunden werden, welche für typische (Test-) Applikationen ausreichend ist und nach Möglichkeit zu keinen funktionalen Einschränkungen im Zuge der Applikationsentwicklung führt. Die OS_CFG.H Datei ist Teil des *Application Skeletons* und darf nicht verändert werden. Der Konfiguration der Flash-Portierung ist in der obigen Tabelle zusammengefasst.

APPLICATION SKELETON

Das *Application Skeleton* ist ein Rahmenprojekt für µC/OS-II basierte Anwendungen welche mit der Evaluierungsversion der Keil-IDE entwickelt werden sollen. Bevor die Applikation ausgeführt werden kann, muss die Flash-Portierung des Betriebssystems in den Flashspeicher des SK167-Boards programmiert werden. Das Rahmenprojekt realisiert eine Softwareschnittstelle (`ucos251phyfl_ifc.a66`) zum geflashten Betriebssystem welche für den Anwender vollkommen transparent in die Anwendung eingebunden wird.



Weitere Elemente des Skeletons sind:

`ucos251phyfl.h86`	Image der µC/OS-II Flashportierung
`ucos251phyfl.h`	*Master Include File* der Applikation
`os_cfg.h`	Konfiguration der Flashportierung
`os_cpu.h`	Teil der C167 Portierung
`ucos_ii.h`	µC/OS-II relevante Defintionen
`ucos251phyfl_ifc.h`	Definition der SW-Schnittstelle zur Flashportierung

Die, für den Anwender, zentrale Komponente des *Application Skeletons* ist sicherlich der C-Quelltext einer Rahmenapplikation (`appl.c`), welche einen schnellen Einstieg in die Programmierung einer µC/OS-II Anwendung ermöglicht. Der prinzipielle Aufbau einer typischen µC/OS-II Anwendung ist der folgende:

```
main()
  Aufruf von OSInit()
  anlegen der Memory Partition für die Task Stacks mit OSMemCreate()
  für jede Task mit OSMemGet() einen Stack anlegen
  die Wurzel Task „TaskStart“ mit OSTaskCreate() kreieren
  Aufruf von OSStart()

TaskStart()
  mit OSTickISRInit() den Softwaretakt starten
  mit OSStatInit() die Statistik-Task starten (optional)
  alle Applikationstasks mit OSTaskCreate() anlegen
  Endlosschleife die jede Sekunde durch OSTimeDlyHMSM() aktiv wird

TaskXXX()
  einmalige Initialisierung (optional)
  Endlosschleife mit eigentlicher Funktion der Task
```

WEITERE PORTIERUNGEN

Neben dem SK167-Board funktioniert die vorliegende µC/OS-II Flashportierung mit unwesentlichen Modifikationen im *Application Skeleton* auch mit dem *Keil MCB167net Board*, mit dem *Embedded Internet Applications* realisiert werden können. Infolge ist eine erweiterte Flashportierung von µC/OS-II mit TCP/IP-Funktionalität geplant.

LITERATUR

1. Jean J. Labrosse, "MicroC/OS-II The Real-Time Kernel". CMP Books, 2nd Edition, ISBN 1-57820-103-9
2. http://www.ucos-ii.com