

Inhalt

Kapitel 1: Task-Management	1
1.1 Task-Grundoperationen	1
1.2 Task-Identifikation	6
1.3 Warteschlangen	9
1.4 Implementierung von Warteschlangen	13
1.4.1 Einketten und Ausketten von Poolelementen	13
1.4.1.1 Einketten nach FIFO	14
1.4.1.2 Ausketten nach FIFO	21
1.5 Implementierung von Task-Queues	26
1.6 Scheduling nach Prioritäten	32
1.7 Die Multitasking-Softwarestruktur	35
1.8 Implementierung eines Multitasking-Basissystems	36
1.8.1 Erzeugung von Zeittakten	37
1.8.2 Anwender-Tasks anmelden und System starten	40
1.9 Taskwechsel-Schnittstellen	50
1.9.1 Der Dispatcher	50
1.9.2 Die Betriebssystem-Prozedur task_wechsel	55
1.10 Stackbereiche der Anwender-Tasks in der Privileg-Ebene 0	59
1.11 Round-Robin-Scheduling gleich wichtiger Tasks	69
1.11.1 Time-Slice-Realisierung	71
1.11.1.1 Vermeiden von «Geisterinterrupts»	79
1.12 Mehrstufiges Herabsetzen der Priorität (Multilevel feedback, Mlf)	82
1.13 Eingabe/Ausgabe	89
1.14 Realisierung eines Preemptiv-Multitasking-Systems	95
Kapitel 2: Task-Kommunikation	107
2.1 Gegenseitiger Task-Ausschluß	107
2.2 Semaphore	108
2.3 Gegenseitiger Ausschluß bei mehreren Tasks	121
2.3.1 Implementierung von Condition-Queues	121
2.4 Erzeuger/Verbraucher-Mechanismus	136
2.4.1 Der beschränkte Puffer	138
2.5 Monitore	147
2.5.1 Der Monitor Ringpuffer	153
2.5.1.1 Realisierung eines Kommunikationssystems	156
2.5.1.2 Signalisieren mit Prioritäten	172
2.6 Die Mailbox	175
2.6.1 Kommunikationsschemata	186
2.6.1.1 Der Monitor mailbox	187
2.6.1.2 Realisierung eines Kommunikationssystems mit mailbox	194
2.6.1.3 Verbessertes broadcasting	223
Kapitel 3: Parallele Numerik	227
3. Management von Numerik-Tasks	227
3.1 Angewandte parallele Programmierung: Numerische Integration	245
3.1.1 Das bestimmte Integral als Mittelwert und Summe	246
3.1.2 Programmierbeispiel: Parallele Winkelberechnung	251

Kapitel 4: Realzeit-Multitasking	265
4. Was ist ein Realzeitsystem?	265
4.1 Interrupt-Task-Management	266
4.1.1 Idle-Task-Management	278
4.2 Zeitverwaltung	294
4.2.1 Der Monitor timer	300
4.2.2 Zeitbasis versus Monitor-Speicherbedarf	301
4.2.3 Pausieren	305
4.2.4 Zeitbegrenztes Warten auf Nachricht	322
4.2.5 Realzeit-Programmierbeispiele	336
Kapitel 5: IO-Management	337
5.1 Betriebsmittel	337
5.2 Gemeinsame Nutzung mehrerer Betriebsmittel	337
5.3 Gemeinsame Nutzung eines einzigen Betriebsmittels	341
5.4 Sequenzielle Nutzung mehrerer Betriebsmittel	344
5.5 Geordnete Betriebsmittelstrategie	348
5.6 Betriebsmittelklassen	362
5.6.1 Deadlock-Erkennung und -Verhinderung	368
5.7 I/O Permission	383
Kapitel 6: Readers and Writers	399
6.1 Analyse einer Monitorlösung	399
6.2 Implementierung	413
6.2.1 Programmierbeispiel: Task_0 und Task_1 betätigen sich als Leser, Task_2 und Task_3 als Schreiber	420
Kapitel 7: Elementare Einführung in verteilte Systeme	423
7.1 Kommunikation in verteilten Systemen	423
7.2 Gegenseitiger Ausschluss in verteilten Systemen	431
7.3 Implementierung der P/V-Simulationen	443
7.3.1 Die Betriebssystem-Funktion coordinate(.)	445
7.4 Nutzung der P/V-Simulationen im zentralen Algorithmus	451
7.5 Deadlocks in verteilten Systemen	457
7.5.1 Gemeinsame Nutzung mehrerer Betriebsmittel	457
7.5.2 Echte und scheinbare Deadlocks in verteilten Systemen	460
7.5.3 Verteilte Deadlock-Vermeidung	466
7.5.3.1 Geordnete Betriebsmittelstrategie	466
7.6 Entfernter Prozeduraufruf (Remote Procedure Call; RPC)	480
7.6.1 Die HLL (High Level Language)-Methode der Parameterübergabe	480
7.6.2 Rückgewinnung der Parameter vom Stack	488
7.6.3 Der Mechanismus des entfernten Prozeduraufrufs	495
7.6.3.1 Der Output-RPC	503
7.6.3.2 Der Input-RPC	510

Kapitel 8: Elementare Einführung in parallele Multicore-Systeme.....	517
8.1 Das Cache-Kohärenz-Problem.....	517
8.1.1 Definition der Cache-Kohärenz.....	521
8.1.2 Kohärenz-Protokolle.....	524
8.1.2.1 Das MOESI-Protokoll mit Write Back-Caches.....	525
8.2 Das Memory-Konsistenz-Problem.....	548
8.2.1 Das grundständige Wesen der sequentiellen Konsistenz.....	551
8.2.1.1 Die Atomarität des Schreibens.....	552
8.2.1.2 Weitere mögliche SC-Ergebnisse für einen ausgewählten Fall.....	553
8.3 Das grundständige Wesen des Total Store Order (TSO/x86)-Speichermodells.....	558
8.3.1 Implementierung von TSO.....	558
8.3.2 FENCES.....	563
8.4 Relaxed (Weak) Memory Consistency-Modelle (XC).....	566
8.4.1 Release Consistency (RC).....	566
8.4.1.1 Der Erzeuger-Verbraucher-Mechanismus.....	566
8.4.1.2 Der gegenseitige Ausschluss (mutual exclusion).....	570
8.4.2 Relaxed Konsistenz mit Daten-Wettlauf (Data Race).....	577
Anhang: Quickreferenz der zugänglichen Betriebssystem-Funktionen	579
Task-Zustände.....	584
Literatur.....	585