

Betriebssystemtechnik

Adressräume: Trennung, Zugriff, Schutz

XIV. Nachlese

Wolfgang Schröder-Preikschat / Volkmar Sieh

26. Juli 2023



Rekapitulation

Prozessadressräume

Perspektiven

Forschungsschwerpunkte und -projekte

Rechnerausstattung

Lehrstuhl Systemsoftware

Weiterqualifikation





Adressräume (von Programmen/Prozessen)

- **tren|nen**: in eine räumliche Distanz voneinander bringen
 - klassisch, hardwarebasiert, durch MMU *und* Betriebssystem
 - unterstützt durch Dienstprogramme (*utility program*)
 - Compiler, Assembler, Binder, Lader
 - vertikal (vom Betriebssystem) und horizontal (Anwendungsprogramme)
 - **zu|grei|fen**: nach etwas greifen und es festhalten bzw. an sich nehmen
 - Interprozesskommunikation (VSM) und Mitbenutzung (*sharing*)
 - Mitbenutzung durch Daten- (*data*) und Textverbund (*code sharing*)
 - kopieren beim Schreiben/Referenzieren (*copy on write/reference*)
 - **schüt|zen**: einer Sache Schutz gewähren, einen Schutz [ver]schaffen
 - Angriffssicherheit (*security*) und Betriebssicherheit (*safety*)
 - Immunität einerseits und Isolation andererseits
 - Eindrang bzw. Ausbruch von Prozessen verhindern
- ergänzend: softwarebasiert, durch typischere Programmiersprachen



Rekapitulation

Prozessadressräume

Perspektiven

Forschungsschwerpunkte und -projekte

Rechnerausstattung

Lehrstuhl Systemsoftware

Weiterqualifikation



- **Komponierbarkeit** und **Konfigurierbarkeit**
 - anwendungsorientierte (variantenreiche, typsichere) Systemsoftware
- **Sparsamkeit**
 - ressourcen-gewahrer Betrieb von Rechensystemen
- **Zuverlässigkeit**
 - Betriebsmittel schonende Fehler- und Einbruchstoleranz
- **Rechtzeitigkeit**
 - Migrationspfade zwischen zeit- und ereignisgesteuerten Echtzeitsystemen
- **Spezialisierbarkeit**
 - dedizierte Betriebssysteme: integriert, adaptiv, parallel
- **Gleichzeitigkeit**
 - Koordination der Kooperation und Konkurrenz zwischen Prozessen

→ Prozessadressräume sind mehr oder weniger querschneidend dazu



- **Octo** — der Bezeichnung eines Wesens entnommen, das:

- i hoch parallel in seinen Aktionen ist und
- ii sich sehr gut an seine Umgebung anpassen kann

→ der Krake (Ordnung *Octopoda*)

- kann kraft seiner (acht) Tentakel parallel agieren
- vermag sich durch Farbänderung anzupassen und
- verfügt über ein hoch entwickeltes Nervensystem
 - um sich auf dynamische Umgebungsbedingungen und -einflüsse einzustellen



- **POS** — Abk. für (engl.) *Parallel Operating System*

- ein Betriebssystem, das nicht bloß parallele Prozesse unterstützt
- sondern dabei selbst **inhärent parallel** arbeitet
- sowie sich einem wechselnden Anwendungsprofil entsprechend anpasst
 - Adressraumvirtualisierung und -devirtualisierung zur Laufzeit bei Bedarf
 - anwendungsorientierter virtuell gemeinsamer Speicher (VSM)

- DFG: seit 06/2011, 3.5 WM (2.5 FAU, 1 KIT), 1 WHK, 3 SHK

- <https://sys.cs.fau.de/research/irtss>



Definition (*NVM-pure* Betriebssystem)

Ein Betriebssystem, das NVM nicht nur für die Maschinenprogramme verwaltet, sondern auch für eigene Zwecke nutzt: das selbst komplett im NVM liegt, darin abläuft sowie bis auf Register-/Zwischenspeicher nirgends flüchtigen Speicher benutzt.

- einen auf **Energieeffizienz** und **Rechenleistung** maximierten und **Latenzzeit** minimierten Betrieb eines Rechensystems erreichen
 - Verzicht auf viele, wenn nicht sogar sämtliche, für gewöhnlich sonst zu realisierende Persistenzmaßnahmen
 - Metadatenpersistenz eines Dateisystems (*Superblock, in-core inode(7)*)
 - Zwischenspeicherung geschriebener Daten (*delayed write, lazy write*)
 - Datensynchronisation (*sync(8), update(8)*)
 - *flush*-Dämon (*bdflush(2)*), ab Version 2.6 der *pdflush*-Faden
 - dadurch **Hintergrundrauschen** (*background noise*) im System verringern
- DFG: seit 08/2021, 2 WM (1 FAU, 1 RUB), 2 SHK
 - <https://sys.cs.fau.de/research/neon>



Definition (*NVM-only* Betriebssystem)

Ein *NVM-pure* Betriebssystem, das herkömmlichen DRAM-basierten Hauptspeicher nur noch benutzt, um die höheren Zugriffszeiten oder Latenzen zu kaschieren, die bei NVRAM noch vorhanden sind.

- das Betriebssystem macht den Maschinenprogrammen nichtflüchtigen Hauptspeicher funktional transparent zugänglich
 - **Hochskalieren der Speicherkapazität** auf NVRAM-Basis
 - Tolerierung unvollständiger, aber unterbrochener Schreiboperationen
 - Vorbeugung vollständiger, aber wiederholter Schreiboperationen
 - Gewähr eines Restenergiefensters zum **Fixieren der Übergangszustände**
- Symbiose von NVRAM und virtueller Speicher, **Altsoftware** den Weg ebnen für direkte Ausführung im nichtflüchtigen Hauptspeicher
- DFG: ab 09/2022, 2 WM (1 BTU, 1 FAU), 2 SHK
 - <https://sys.cs.fau.de/research/pave>



- Ausfälle, Überlastung, Angriffe und das Unerwartete meistern
 - byteadressierbaren NVM als primären Hauptspeicher begreifen
 - dem Paradigma der transaktionalen Programmierung folgen
- Kommunikations- und Betriebssystem für mobile IoT-Gerätschaften
 - Sensoren und Aktuatoren
 - heterogene Speicherarchitekturen und Kommunikationsschnittellen
- Widerstandsfähigkeit gegenüber **Betriebsstörungen**
 - **Mikrotransaktionen** im Sinne nichtblockierender Synchronisation
 - durch Ausnahmen ausgelöste **Fixpunkte** des Übergangszustands
- Widerstandsfähigkeit gegenüber **Funktionsstörungen**
 - dynamisches **Vorhersagen** Kapazität des Kommunikationskanals
 - dynamisches **Anpassen** der Reparaturtechnik und des Ablaufplans
 - **Schätzen** des Informationalters und passenden Kommunikationstempos
- DFG: ab 09/2022, 2 WM (1 FAU, 1 UDS), 2 SHK
 - <https://sys.cs.fau.de/research/respect>



- Beschränkung in zweierlei Hinsicht:
 - funktional** ■ maßgeschneiderte echtzeitabhängige/-fähige Software
 - Anwendungsfall (*use case*)
 - nichtfunktional** ■ Raum, Zeit, Energie
 - ungünstigster Fall (*worst-case*)
- Schwerpunkt sind **energiebeschränkte Echtzeitsysteme**
 - *a priori* Wissen zum möglichen/absehbaren Ablaufverhalten von Prozessen
 - WCRT** ■ *worst-case response time*
 - WCRE** ■ *worst-case response energy consumption*

Ungünstigste Reaktion (*worst-case response*)

Ressourcenverbrauchsbedarf vom Beginn einer Aufgabe bis zu ihrer Beendigung, einschließlich aller möglichen Störungen.

- vorauswissen, nichtfunktionales Verhalten automatisch zu beeinflussen
- dabei aber funktional äquivalente Systemdarstellungen beibehalten

- DFG: voraussichtlich Q3/2022, 1 WM, 1 SHK
- <https://sys.cs.fau.de/research/watwa>



Migrationsentscheidungen anstatt auf globalen Lastparametern systematisch auf Basis von Hinweisen der unter (strikten) Echtzeitbedingungen ablaufenden Maschinenprogramme treffen.

- **Prozessmigration** in mehrkernigen Echtzeitsystemen
 - Hinweise zu zeitlichen und räumlichen Aspekten von Echtzeitprozessen
 - Markierung potentieller Migrationspunkte mehrfädiger Programme
- das zur Abwanderung bestimmte Prozessexemplar ist der Faden (*thread*)
- das Betriebssystem zu günstigen Entscheidungen befähigen
 - betrifft **Vorhersagbarkeit** und **Leistungsfähigkeit** des Gesamtsystems
 - angestrebte Verbesserung hinsichtlich **Antwortzeit** und **Planbarkeit**
- betrachtet werden Systeme von heterogener Speicherarchitektur
 - die durch Migration verursachten Verwaltungsgemeinkosten
 - die von Speicherort und Umfang der Migrationsdaten abhängen
- Eigenmittel: 2 WM (1 FAU, 1 TUDO), 2 SHK
 - <https://sys.cs.fau.de/research/mare>



Große Systeme, einmal in Betrieb genommen, unterliegen in der Regel häufigen Änderungen — auch, um die Passgenauigkeit an sich ändernde Anwendungsanforderungen zu verbessern.

- gemeinhin werden **Allzwecksysteme** vorgefertigt und im Binärformat geliefert, ohne auf ein individuelles System zugeschnitten zu sein
 - eingeschränkte zielsystemspezifische Optimierungen zur Herstellungszeit
 - Erweiterungen der konkreten Befehlssatzebene bleiben unausgenutzt
- **Leistungspotential** der gegebenen Hardware **wird nicht ausgeschöpft**
- **weniger anspruchsvolle Anwendungen** sollten nicht für verbrauchte Ressourcen durch nicht benötigte Funktionen zahlen müssen
 - ideale Betriebssysteme bieten genau das, was eine Anwendung benötigt
 - sie wachsen/schrumpfen mit den jeweiligen Anwendungsanforderungen
- **bedarfssynchrone (just in time) Übersetzung** des Betriebssystem(kern)s
- DFG, Projektkampagne: 2 WM (1 FAU, 1 RUB), 2 SHK
 - <https://sys.cs.fau.de/research/doss>



Systeme mehr-/vielkerniger Prozessoren

fau4*	clock	cores per domain		domain		#	
		physical	logical	NUMA	tile		
*8e *8f	2.9 GHz	8	16	2	1	32	Xeon
*9big01	2.5 GHz	6	6	8	1	48	Opteron
*9big02	2.2 GHz	10	20	4	1	80	Xeon
*9big03	2.1 GHz	12	24	4	1	96	Xeon
*9big04	2 GHz ¹	64	128	2	1	256	Epyc
*9big05	2.5 GHz	16	128	2	4	1024	ThunderX2
*9phi01	1.2 GHz	6	12	2	1	24	Xeon
	1.1 GHz	57	228	2	1	456	Xeon Phi
*scc	1.5 GHz	4	8	1	1	8	Xeon
	800 MHz	2	—	—	24	48	Pentium
fastbox <i>InvasIC</i>	3.5 GHz	4	8	1	1	8	Xeon TSX
	50 MHz	5	5	16		80	LEON/SPARC

2160

¹ mit *boost* 3.35 GHz







- [1] SCHRÖDER-PREIKSCHAT, W. ; KLEINÖDER, J. :
Systemprogrammierung.
http://www4.informatik.uni-erlangen.de/Lehre/WS08/V_SP, 2008 ff.

- [2] SIEH, V. :
Betriebssysteme.
http://www4.cs.fau.de/Lehre/WS17/V_BS, 2017 ff.

