Betriebssystemtechnik

Adressräume: Trennung, Zugriff, Schutz

XII. Nachlese

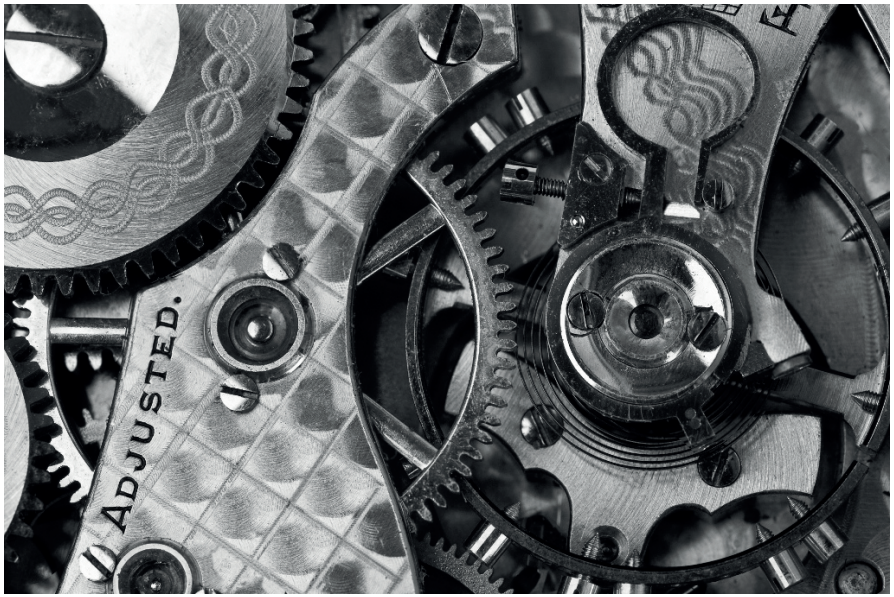
Wolfgang Schröder-Preikschat

11. Juli 2016

Gliederung

Adressraumverwaltung

querschneidender Belang



Verfeinerung/Vertiefung von Betriebssysteme [1, 2]

Adressräume (von Programmen/Prozessen)

- tren|nen: in eine räumliche Distanz voneinander bringen
  - klassisch, hardwarebasiert, durch MMU *und* Betriebssystem
  - unterstützt durch Dienstprogramme (*utility program*)
    - Kompilierer, Assemblierer, Binder, Lader
  - vertikal (vom Betriebssystem) und horizontal (Anwendungsprogramme)
- zu|grei|fen: nach etwas greifen und es festhalten bzw. an sich nehmen
  - Interprozesskommunikation (IPC) 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 typsichere Programmiersprachen

Gliederung

Forschungsschwerpunkte am Lehrstuhl

- **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

Fehler- und Einbruchstoleranz

Betriebsmittel schonende **byzantinische Fehlertoleranz** (BFT)

- **Virtualisierung** als Schlüsseltechnologie
  - Konsolidierung von Diensteeinheiten (*server*)
  - Redundanz durch replizierte virtuelle Maschinen
- **Vorhersage** wahrscheinlicher Ausführungspfade
  - deterministische mehrfädige Ausführung
    - geordnete Sperrreihenfolge (*lock sequence*)
    - Koordinierung zwischen Kopien als Ausnahmefall
  - skalierbare Sperrüberwachung und -verwaltung
- **minimal invasive Operation** für den Normalfall: $f + 1$ Kopien
  - dehnfähige (*resilient*) Einigungsprotokolle für den Ausnahmefall
  - performante Nachrichtenauthentifikation und -verifikation
  - zuverlässiger hardwarebasierter (FPGA) Zählerzuweisungsdienst
- DFG: seit 10/2009, 2 WM (1 FAU, 1 TUBS), 2 SHK

[1] `http://univis.uni-erlangen.de` → Forschungsprojekte → REFIT

Adaptive responsive eingebettete Systeme

Bereitstellung selektiver und adaptiver Fehlertoleranzmaßnahmen

- **kombinierter Redundanzansatz**
  - holistische Absicherung v. Regelungsanwendungen
    - bestehend aus Sensorik, Regelung und Aktuatorik
    - *der* sicherheitskritische Teil des Gesamtsystems
  - softwarebasierte Fehlertoleranz
    - transiente Hardwarefehler tolerieren
- auf **gemischte Kritikalität** (*mixed criticality*) ausgerichtet
  - Echtzeitfähigkeit (weich, fest, hart) eng gekoppelt mit
  - Betriebs- (*safety*) und Angriffssicherheit (*security*)
- **Messunsicherheit** refl. Entwurfskonzept für Regelungssysteme
  - problemspezifische Modularisierung von Regelungsanwendungen
  - Aufbau zuverlässiger System aus unzuverlässigen Hardwarekomponenten
- BaySt M WIVT: seit 01/2010, 2 WM, 2 SHK

[2] `http://univis.uni-erlangen.de` → Forschungsprojekte → ARES

dOSEK: Dependability-Oriented Static Embedded Kernels

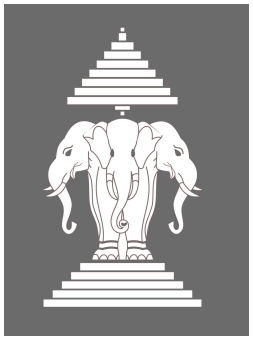
Zuverlässigkeitsorientierter Systemsoftwareentwurf

- gezielte **Vermeidung von Bitfehlern**, durch
  - extreme statische Maßschneiderung der Funktionalität
  - extreme Reduktion von Indirektionen (insbes. Zeiger) in Code und Daten
- gezielte **Erkennung möglicher Restfehler**, durch
  - arithmetische Codierung des Schedulers
  - redundante Datenstrukturen
  - MPU/MMU-basierter Hardwaremaßnahmen
- gezielte **Optimierung des Gesamtsystems**, durch
  - statische Analyse der Interaktionen Anwendung $\longleftrightarrow$ Kern
  - vollständige Generierung des Betriebssystems
  - Spezialisierung von Systemfunktionen pro Aufrufer
- DFG: seit 10/2010, 3 WM (1 FAU, 1 TUBS, 1 TUDO), 1 SHK

[3] `https://www4.cs.fau.de/Research/dOSEK/`
[4] `https://www4.cs.fau.de/Research/DanceOS/`

Latency Awareness in Operating Systems

Latenzgewahrheit in Betriebssystemen für massiv-parallele CPUs



- **Latenzvorbeugung**
  - domänenspezifische Entwurfsmuster
  - sperr- und wartefreie Synchronisation
- **Latenzvermeidung**
  - Interferenzschutz
  - Eindämmung von Wettstreitigkeiten
- **Latenzverbergung**
  - asynchrone Prozedurfern-/Systemaufrufe
  - Kerne für Betriebssysteme nutzen
- Experimente mit unterschiedlichen **Betriebssystemarchitekturen**
  - Entwicklung eigener prozess- und ereignisbasierte Betriebssystemkerne
  - Übertragung mancher Konzepte und Techniken in Linux
- DFG: seit 05/2011, 2 WM, 2 SHK

[5] `http://univis.uni-erlangen.de` → Forschungsprojekte → LAOS

Laufzeitunterstützungssystem für invasives Rechnen

- ***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 Nevensystem
    - 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
    - einhergehend mit dem Auf- und Rückbau des Betriebssystemkerns im Betrieb
- DFG: seit 06/2011, 3.5 WM (2.5 FAU, 1 KIT), 1 WHK, 3 SHK

[6] `http://univis.uni-erlangen.de` → Forschungsprojekte → iRTSS

Aspektorientierte Echtzeitsystemarchitekturen

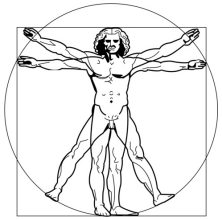
Reflektion kausal und temporal abhängiger gleichzeitiger Aufgaben

- **zeitgesteuerte** (*time-triggered*, TT) **Systeme**
  - Vorabwissen zwingend erforderlich
  - statische Ablaufplanung, vor Laufzeit
  - implizit koordinierte Prozesse, kein Laufzeitaufwand
  - von eher simpler Struktur, leichter analysierbar
- **ereignisgesteuerte** (*event-triggered*, ET) **Systeme**
  - Vorabwissen nicht erforderlich, aber vorteilhaft
  - dynamische Ablaufplanung, zur Laufzeit
  - explizit zu koordiniernde Prozesse, Laufzeitaufwand
  - von eher komplexer Struktur, schwieriger analysierter
- **Migrationspfad** zwischen verschiedenen Echtzeitsystemarchitekturen
  - übersetzergestützte Transformation von ET- zu TT-Programmen *v.v.*
  - kanonische Programmierschnittstelle für Echtzeitanwendungssysteme
- DFG: seit 08/2011, 2 WM, 2 SHK

[7] `http://univis.uni-erlangen.de` → Forschungsprojekte → AORTA

Systemsoftwareinfrastruktur heterogener Bildsysteme

Resource-Aware Multi-Processing für heterogene Vielkernprozessoren

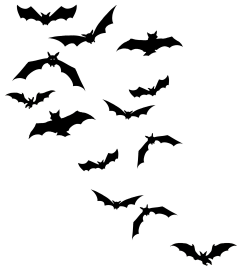


- GPU-zentrische **Betriebsmittelverwaltung**
  - zeitlich berechenbares Laufzeitsystem
    - nichtverdrängbarer Kern (*run to completion*)
  - Periodisierung und Isolierung von GPU-Aufträgen
    - Einplanung nach Ausführungskosten
  - Abstimmung (*tradeoff*) von Durchsatz und Antwortzeit
- RAM-zentrische **Laufzeitexekutive** für heterogene Vielkernsysteme
  - applikationsspezifische und datenstrukturorientierte Speicherverwaltung
  - Laufzeitanpassung und -relokation dynamischer Datenstrukturen
- für Bildsystemanwendungen **maßgeschneiderte Systemsoftware**
  - Unterstützung der inkrementellen Verbesserung visueller Qualität
  - Muster zur adaptiven Detailanpassung von Geometrie oder Texturen
- DFG: seit 02/2012, 1 WM, 1 SHK

[8] `http://univis.uni-erlangen.de` → Forschungsprojekte → RAMP

Softwareinfrastruktur vernetzter Sensorsysteme

Adaptive Run-Time Environment betriebsmittelarmer Sensorsysteme



- **mobiles** (agiles) **Sensornetz** einerseits
  - Mausohr *Myotis myotis* mit „Sensorknotenrucksack“
    - Ortswechsel, Migration und Soziobiologie
  - Fliegengewicht (20 g), fordert extrem leichten Aufbau
    - Kleinstknoten (2 g) ⇝ wenig Hardware, hochintegriert
  - Systemsoftware für winzigste Rechensysteme
    - extrem wenig Speicher und Rechenleistung
    - dynamisches Laden bei *ad-hoc* Netzwerkverbindungen
    - Betriebsdauer maximieren, Energieverbrauch minimieren
  - statische Programmanalyse zur Energiebedarfsabschätzung
- **stationäres Sensornetz** andererseits: Netzkoppler (*gateway*)
  - spontane und kurzzeitige Bewirtung der vorbeifliegenden Sensorknoten
- DFG: seit 07/2012, 1 WM (0.5 FAU, 0.5 TUBS), 1 SHK

[9] `http://univis.uni-erlangen.de` → Forschungsprojekte → BATS

Coherency Kernel

Softwarekontrollierte Konsistenz/Kohärenz für vielkernige Prozessoren

- **ereignisbasierter Minimalkern**
  - zwischenspeichergewahrer Speicherabdruck
  - „überfaden“ (*hyper-threading*) latenter Aktionen
- federgewichtige **Einigungsprotokolle**
  - kernübergreifende Synchronisation
  - Familie von Konsistenzkernen
- **problemorientierte Konsistenzmaschinen**
  - sequentielle, Eintritts- und Freigabekonsistenz
  - funktionale Hierarchie von Konsistenzdomänen
  - Speicherdomänen für NUMA-Architekturen
- Auslegungen für unterschiedliche **Prozessorarchitekturen**
  - partiell bzw. total, {in,}kohärenter gemeinsamer Speicher
- DFG: seit 08/2012, 2 WM (1 FAU, 1 BTU)

[10] `http://univis.uni-erlangen.de` → Forschungsprojekte → COKE

CADOS

Configurability Aware Development of Operating Systems

- Beherrschung von **Variabilität** in Systemsoftware
  - skalierbare Methoden und Werkzeuge: ganzheitlich und ebenübergreifend
  - Erfassung, Darstellung, Bewertung, Beeinflussung der Implementierung konfigurierbarer Merkmale als „Merkmalseinfassungen“
- mit besonderem **Schwerpunkt** auf drei Themengebieten:
  - i Verwaltung von Variabilität in Betriebssystemen
  - ii Implementierung von Betriebssystemmerkmalen
  - iii Variantengewahre Analyse von Betriebssystemquelltext

- **LIFE** — Der Linux *Feature Explorer*
  - exploratives Werkzeug zur Erfassung und Visualisierung von Variabilität
  - mit Fokus auf Konfigurations- und Implementierungsraum
- DFG: seit 09/2014, 1 WM, 1 SHK

[11] `http://univis.uni-erlangen.de` → Forschungsprojekte → CADOS

Energiegewahre kritische Abschnitte

- skalierbare Synchronisation durch **agile kritische Abschnitte**
  - Infrastruktur: lastabhängiger und selbstorganisierter Wechsel des Schutzes vor Wettlaufsituationen
  - Sprachunterstützung: Vorbereitung, Charakterisierung und Erfassung deklarierter kritischer Abschnitte
- automatisierte Extraktion kritischer Abschnitte
  - Beschreibungssprache für kritische Abschnitte
  - Programmanalyse und LLVM Integration/Adaption

PAX

- energie-gewahre Systemprogrammierung
  - wechselseitiger Ausschluss, überwachter Abschnitt, Transaktion
  - dynamisches Binden von Schutzprotokollen bzw. kritischen Abschnitten.
- manipulationssichere Energieverbrauchsmessung
  - Befehlsübersicht und -statistik mit realen und virtuellen Maschinen
  - Vorhersage bzw. Abschätzung des Energieverbrauchs]
- DFG: seit 01/2015, 2 WM, 2 SHK

[12] `http://univis.uni-erlangen.de` → Forschungsprojekte → PAX

Systeme mehr-/vielkerniger Prozessoren

faui4*	clock	cores per domain		domain		
		physical	logical	NUMA	tile	
8e 8f	2.9 GHz	8	16	2	–	Xeon
9big01	2.5 GHz	6	–	8	–	Opteron
9big02	2.2 GHz	10	20	4	–	Xeon
9phi01	1.2 GHz	6	12	2	–	Xeon
	1.1 GHz	57	228	2	–	Xeon Phi
scc	1.5 GHz	4	2	1	–	Xeon
	800 MHz	2	–	–	24	Pentium
InvasIC	3.5 GHz	8	16	2	–	Xeon
	25 MHz	4	–	6		LEON/SPARC

- budgeted acquisition: weitere n-kernsysteme, transaktionaler Speicher
  - OctoPOS: $n \geq 64$, in 2015
  - PAX: $n \geq 16$, in 2016, zusätzlich mehrkernige Mikrocontroller

Bachelor-, Master- oder Doktorarbeit

Literaturverzeichnis I

[1] Lohmann, D. ; Schröder-Preikschat, W. :
Betriebssysteme.
http://www4.informatik.uni-erlangen.de/Lehre/WS08/V_BS, 2008 *ff*.

[2] Schröder-Preikschat, W. ; Kleinöder, J. :
Systemprogrammierung.
http://www4.informatik.uni-erlangen.de/Lehre/WS08/V_SP, 2008 *ff*.