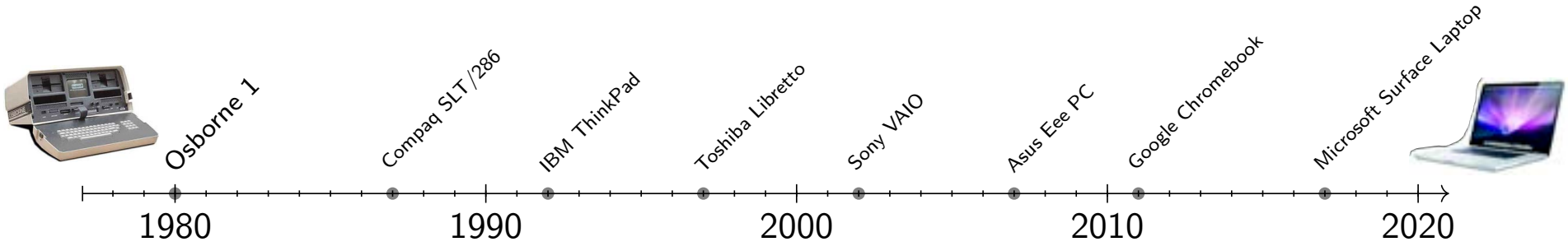
System Software for Energy-efficient Computing

Timo Hönig
Operating Systems and System Software

Ruhr-Universität Bochum (RUB)
2. October 2025

1980er **2020er**

1980er **2020er**



Technologischer Fortschritt der letzten Jahrzehnte

Netzwerk: 3.300.000 x

Übertragungsgeschwindigkeit um Faktor ca. **3,3 Millionen** verbessert

- 300 bit/s vs. 1 Gigabit/s

Technologischer Fortschritt der letzten Jahrzehnte

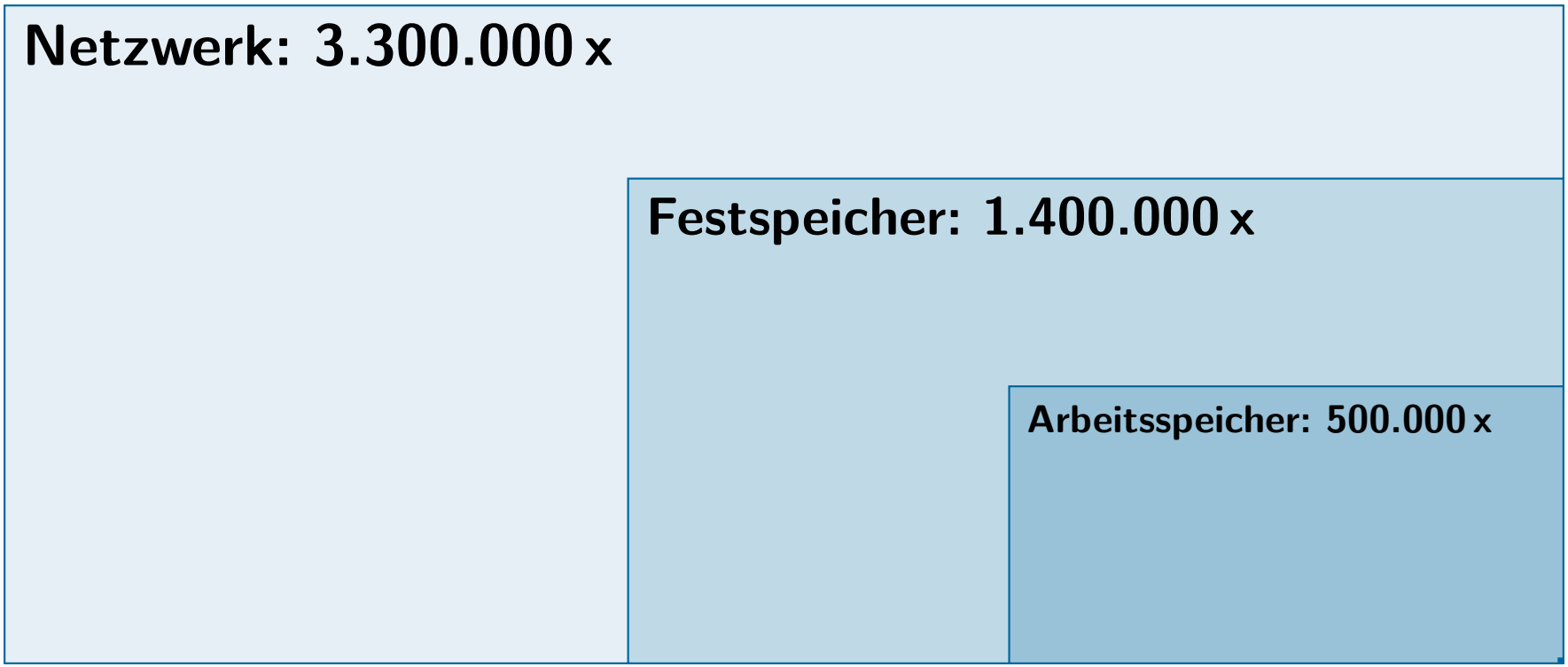
Netzwerk: 3.300.000 x

Festspeicher: 1.400.000 x

Speicherkapazität um Faktor ca. **1,4 Millionen** vergrößert

- 360 KiB vs. 500 GiB

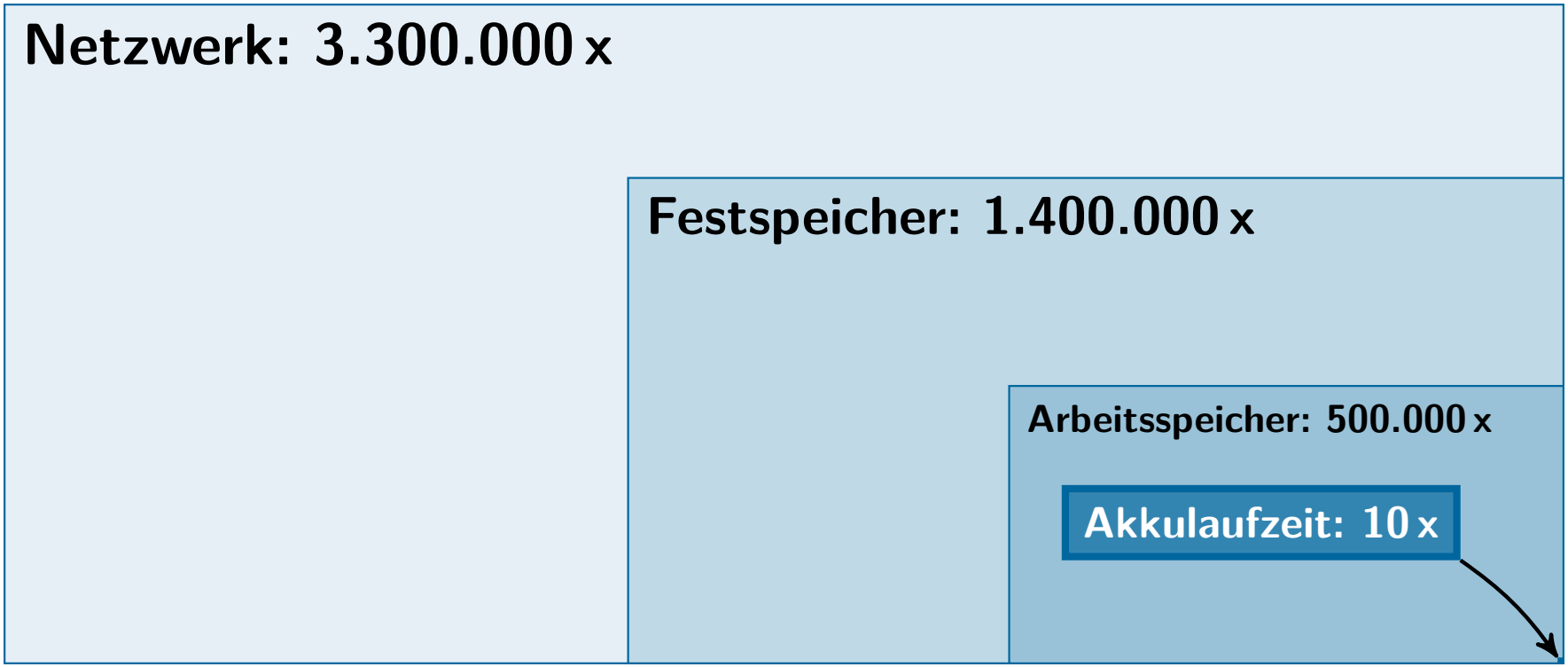
Technologischer Fortschritt der letzten Jahrzehnte



Kapazität des Arbeitsspeichers um Faktor ca. **0,5 Millionen** verbessert

- 4 KiB vs. 2 GiB

Technologischer Fortschritt der letzten Jahrzehnte

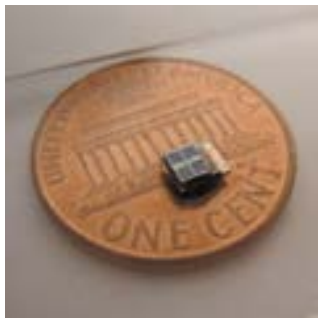


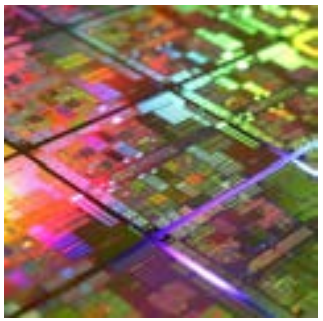
Verbesserung der Akkulaufzeit um Faktor **10** (0,00001 Mio.)

- 1 h vs. 10 h

Energieeffiziente Systeme: Herausforderungen & Ziele

- elektrische Energie ist wichtigste Betriebsressource für Computer

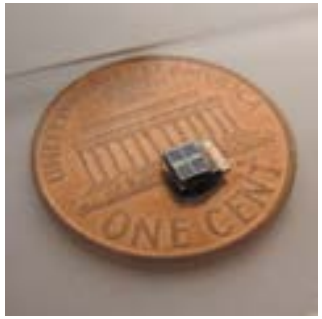


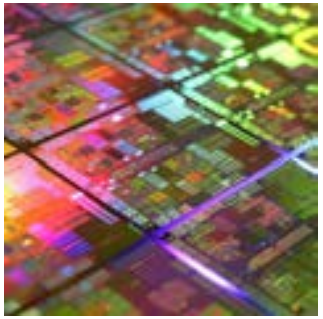




Embedded —— Laptop/Desktop —— HPC/Mainframe

Energieeffiziente Systeme: Herausforderungen & Ziele

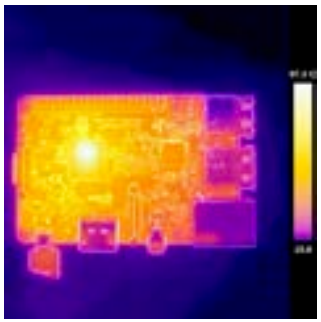
- elektrische Energie ist wichtigste Betriebsressource für Computer



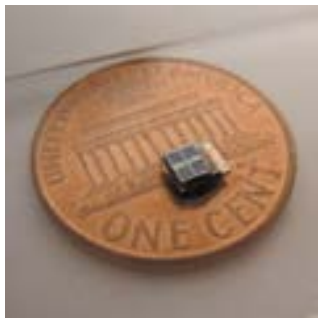

Embedded —— Laptop/Desktop —— HPC/Mainframe

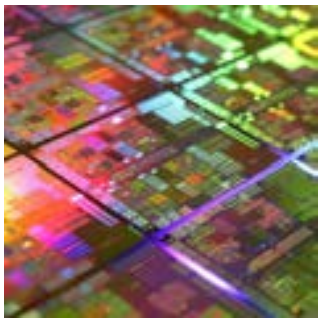
- übermäßige Verlustleistung führt zu unkontrollierbaren Situationen



Energieeffiziente Systeme: Herausforderungen & Ziele

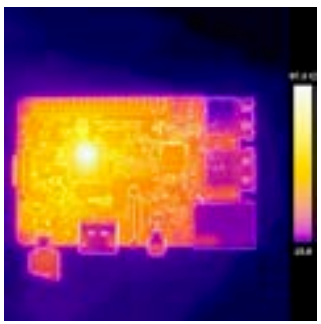
- elektrische Energie ist wichtigste Betriebsressource für Computer




Embedded —— Laptop/Desktop —— HPC/Mainframe

- übermäßige Verlustleistung führt zu unkontrollierbaren Situationen

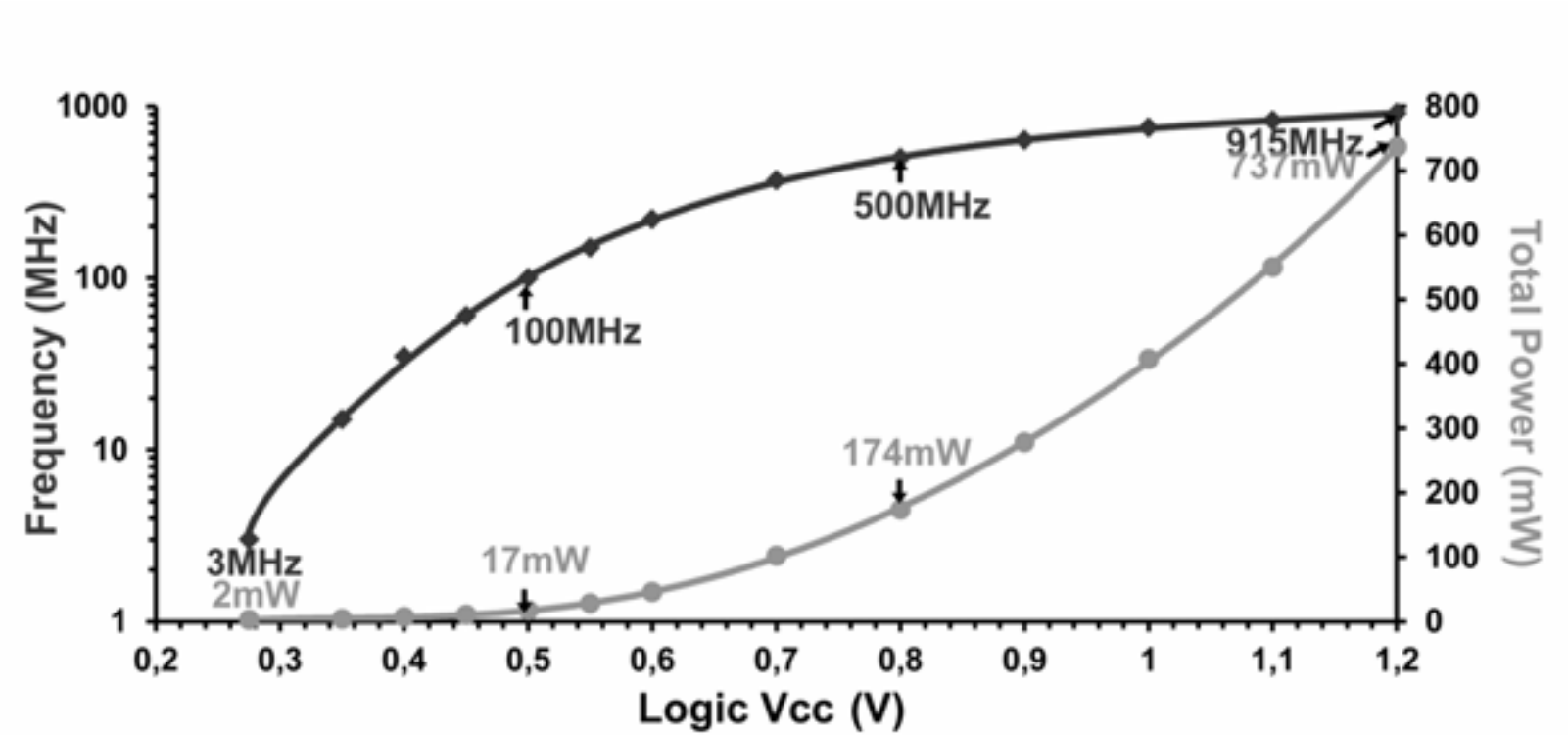



Agenda

1. Motivation ✔
2. Energieeffiziente Systemsoftware
3. Energiebewusste Programmierung
4. Albatross OS
5. Ausblick: Aktuelle Forschung
6. Zusammenfassung und Schlussfolgerung

Systemsoftware für energieeffiziente Systeme

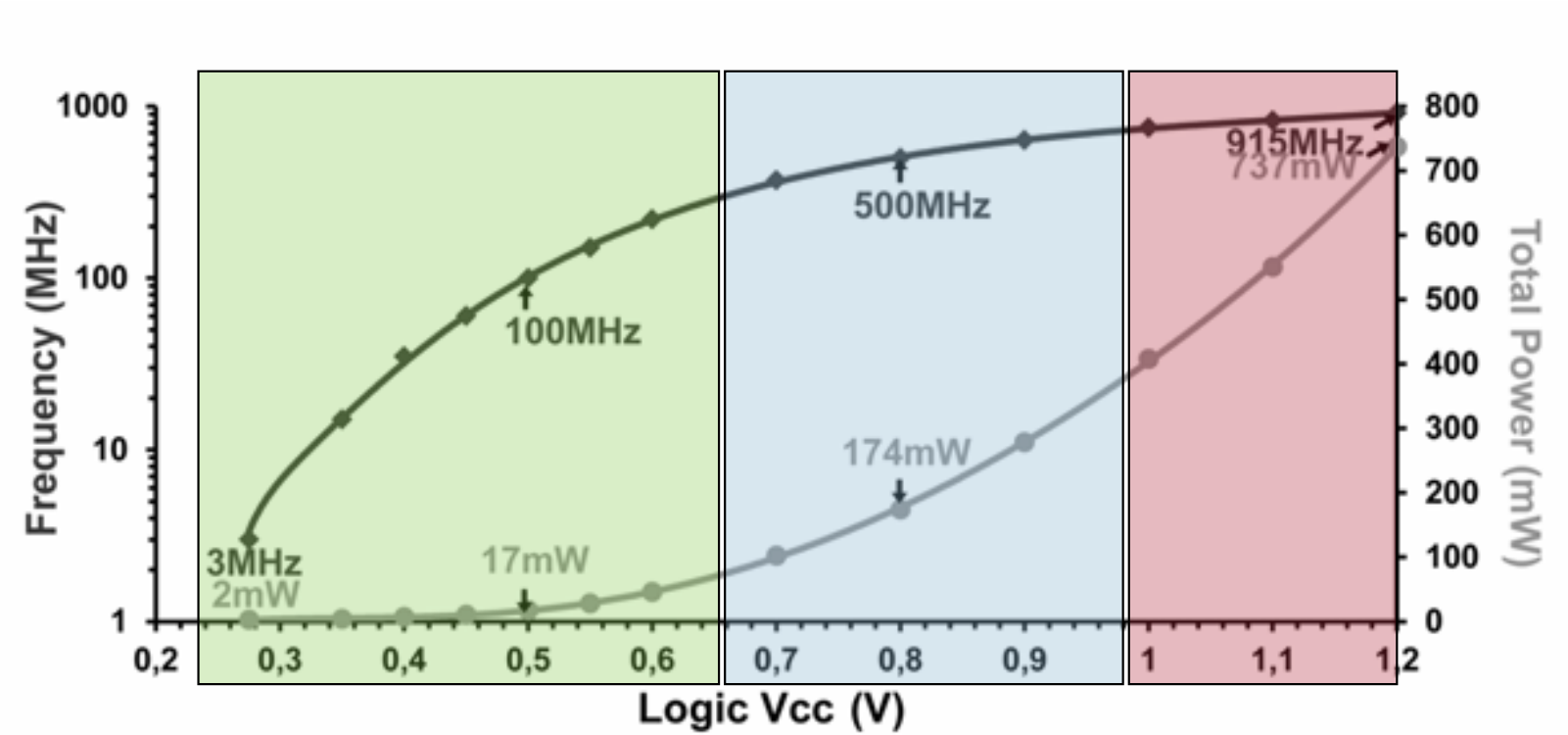
- Energiebedarf ist eine nicht-funktionale Eigenschaft der Hardware
- Energiesparmechanismen der Hardware müssen gesteuert werden



- Shailendra Jain, Surhud Khare, Satish Yada et al.
  A 280mV-to-1.2V Wide-Operating-Range IA-32 Processor in 32 nm CMOS
  IEEE International Solid-State Circuits Conference (ISSCC), 2012.

Systemsoftware für energieeffiziente Systeme

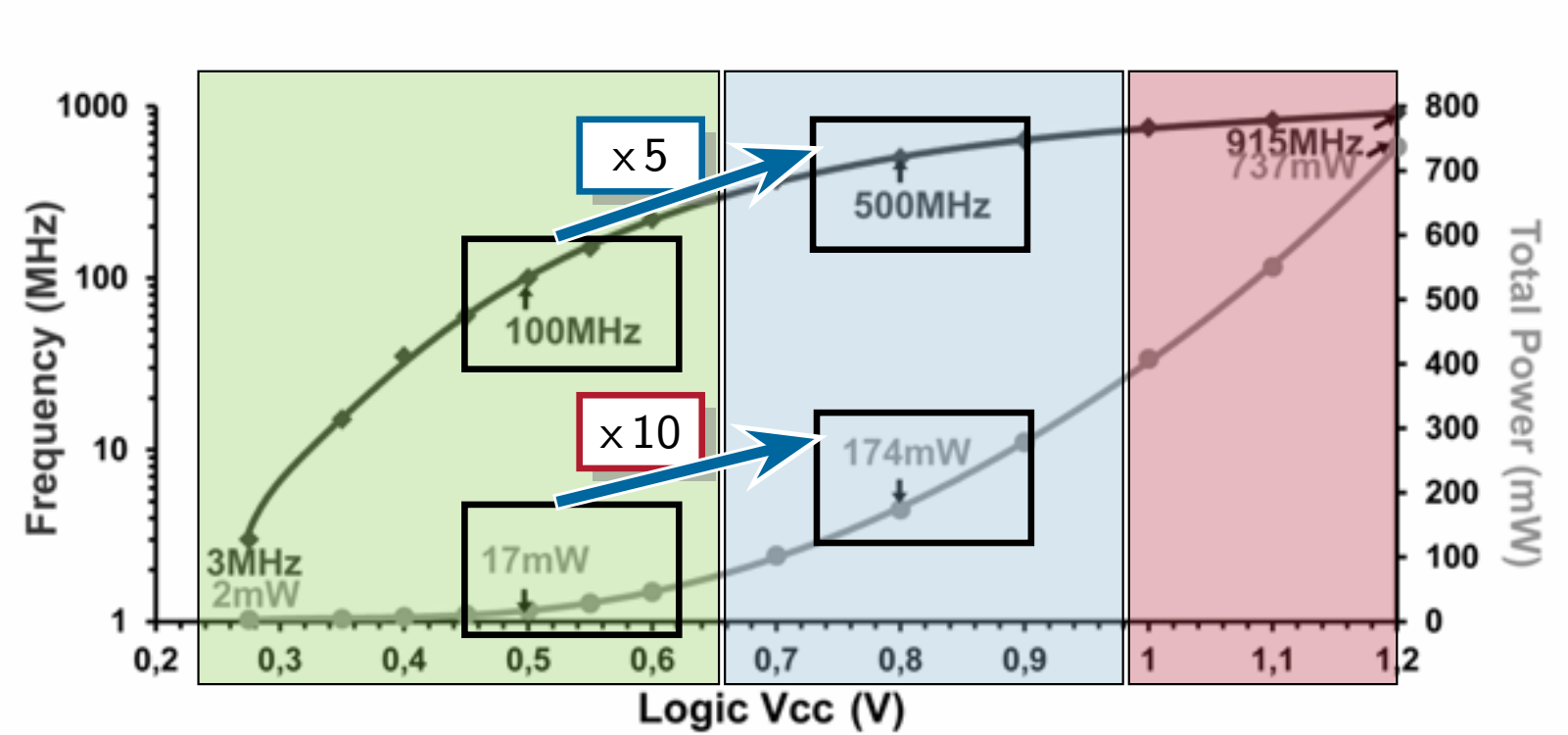
- Energiebedarf ist eine nicht-funktionale Eigenschaft der Hardware
- Energiesparmechanismen der Hardware müssen gesteuert werden



- Shailendra Jain, Surhud Khare, Satish Yada et al.
  A 280mV-to-1.2V Wide-Operating-Range IA-32 Processor in 32 nm CMOS
  IEEE International Solid-State Circuits Conference (ISSCC), 2012.

Systemsoftware für energieeffiziente Systeme

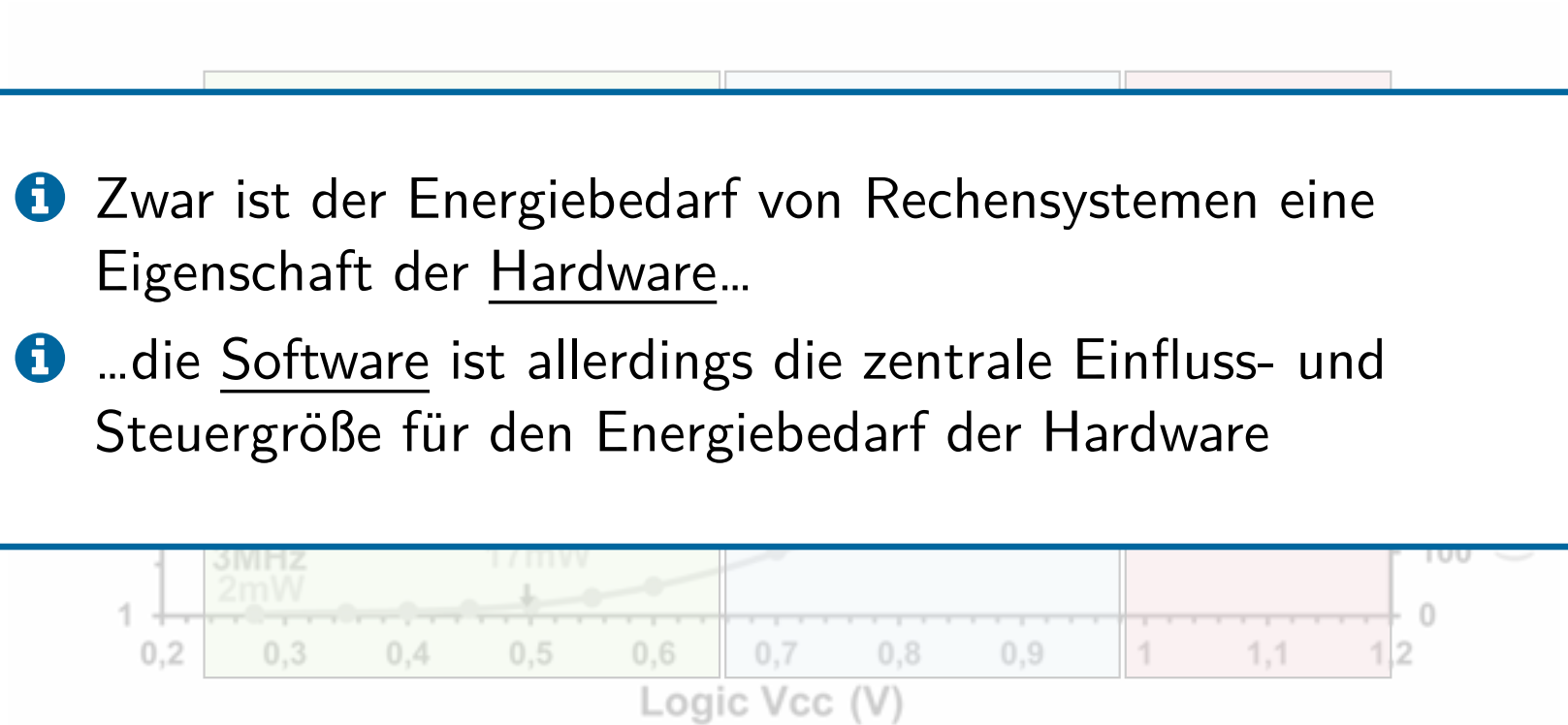
- Energiebedarf ist eine nicht-funktionale Eigenschaft der Hardware
- Energiesparmechanismen der Hardware müssen gesteuert werden



- Shailendra Jain, Surhud Khare, Satish Yada et al.
  A 280mV-to-1.2V Wide-Operating-Range IA-32 Processor in 32 nm CMOS
  IEEE International Solid-State Circuits Conference (ISSCC), 2012.

Systemsoftware für energieeffiziente Systeme

- Energiebedarf ist eine nicht-funktionale Eigenschaft der Hardware
- Energiesparmechanismen der Hardware müssen gesteuert werden

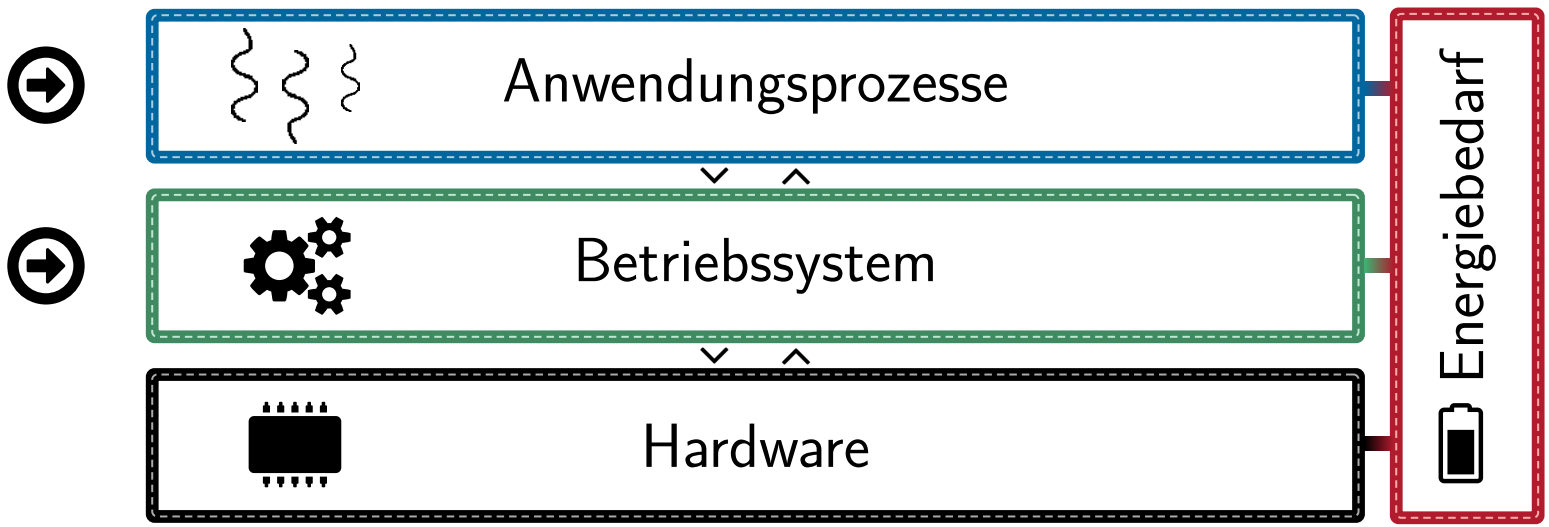


- Zwar ist der Energiebedarf von Rechensystemen eine Eigenschaft der Hardware...
- ...die Software ist allerdings die zentrale Einfluss- und Steuergröße für den Energiebedarf der Hardware

- Shailendra Jain, Surhud Khare, Satish Yada et al.
  A 280mV-to-1.2V Wide-Operating-Range IA-32 Processor in 32 nm CMOS
  IEEE International Solid-State Circuits Conference (ISSCC), 2012.

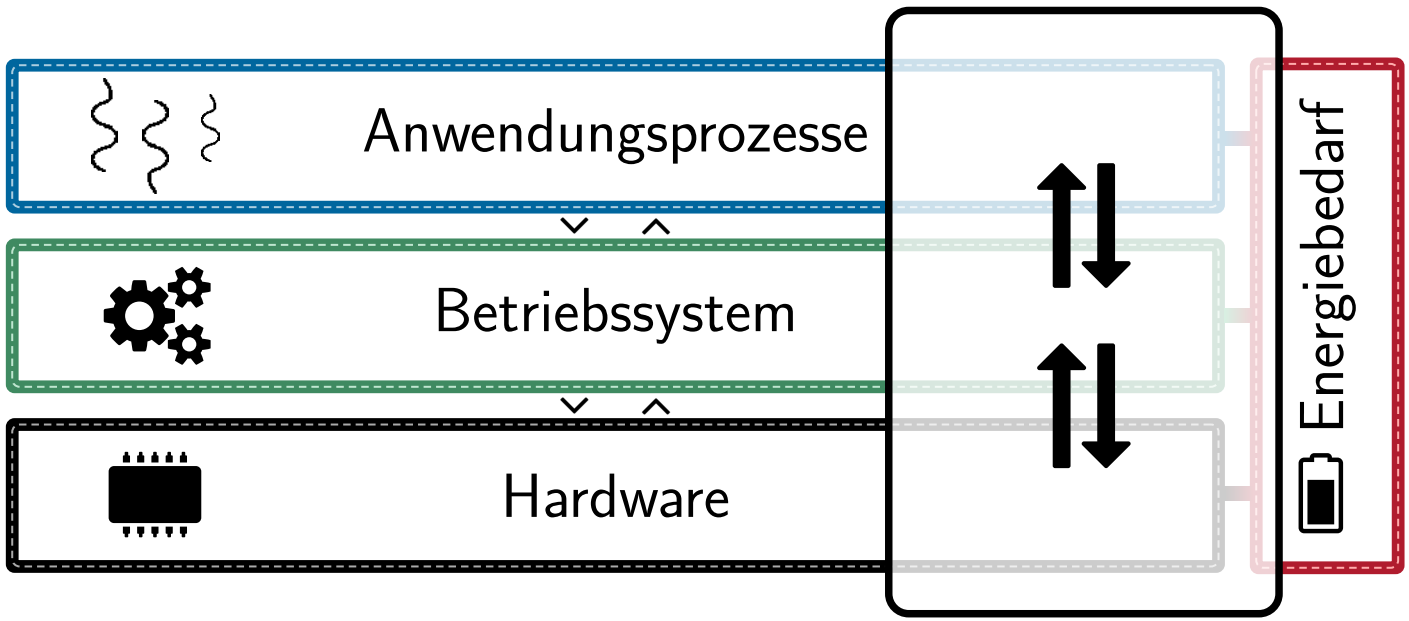
Systemsoftware für energieeffiziente Systeme

- Energiebewusste Programmierung
- Albatross OS: Betriebssystem für energieeffiziente HPC-Systeme



Systemsoftware für energieeffiziente Systeme

- Energiebewusste Programmierung
- Albatross OS: Betriebssystem für energieeffiziente HPC-Systeme



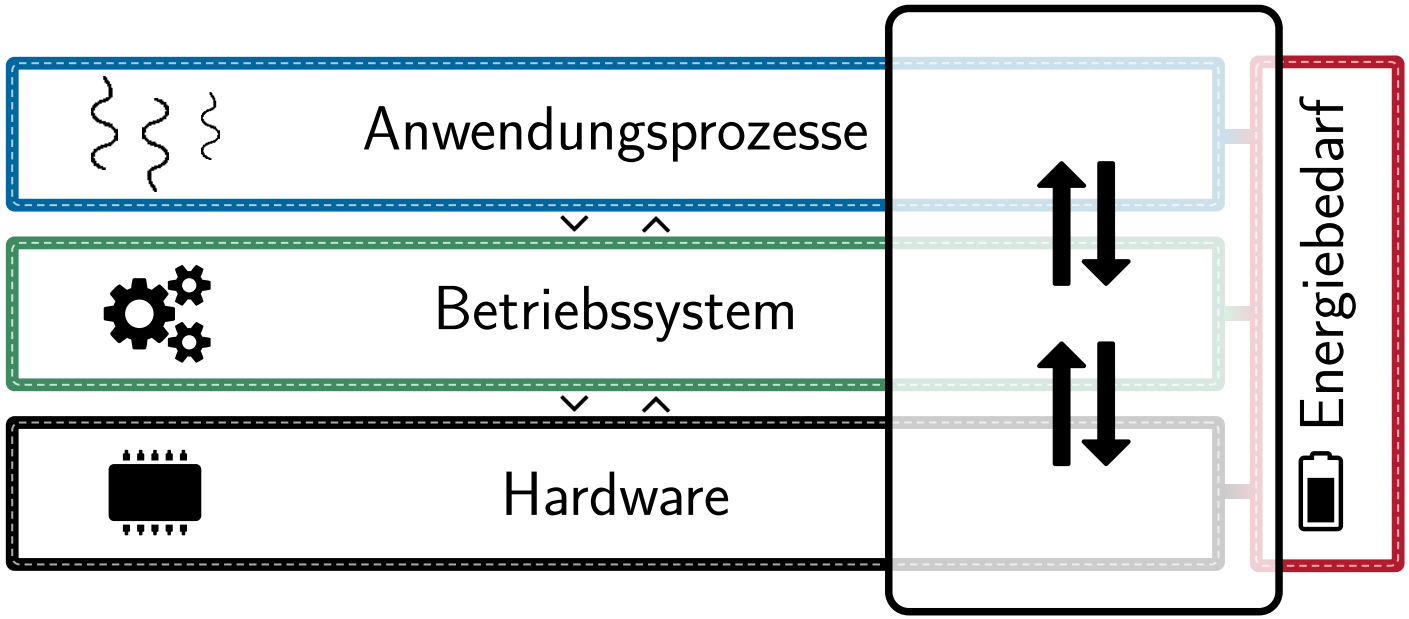
→ Schichtenübergreifende Ansätze

→ Erreichen bislang unausgeschöpfter Einsparpotentiale

Systemsoftware für energieeffiziente Systeme

Vor der Laufzeit

Zur Laufzeit



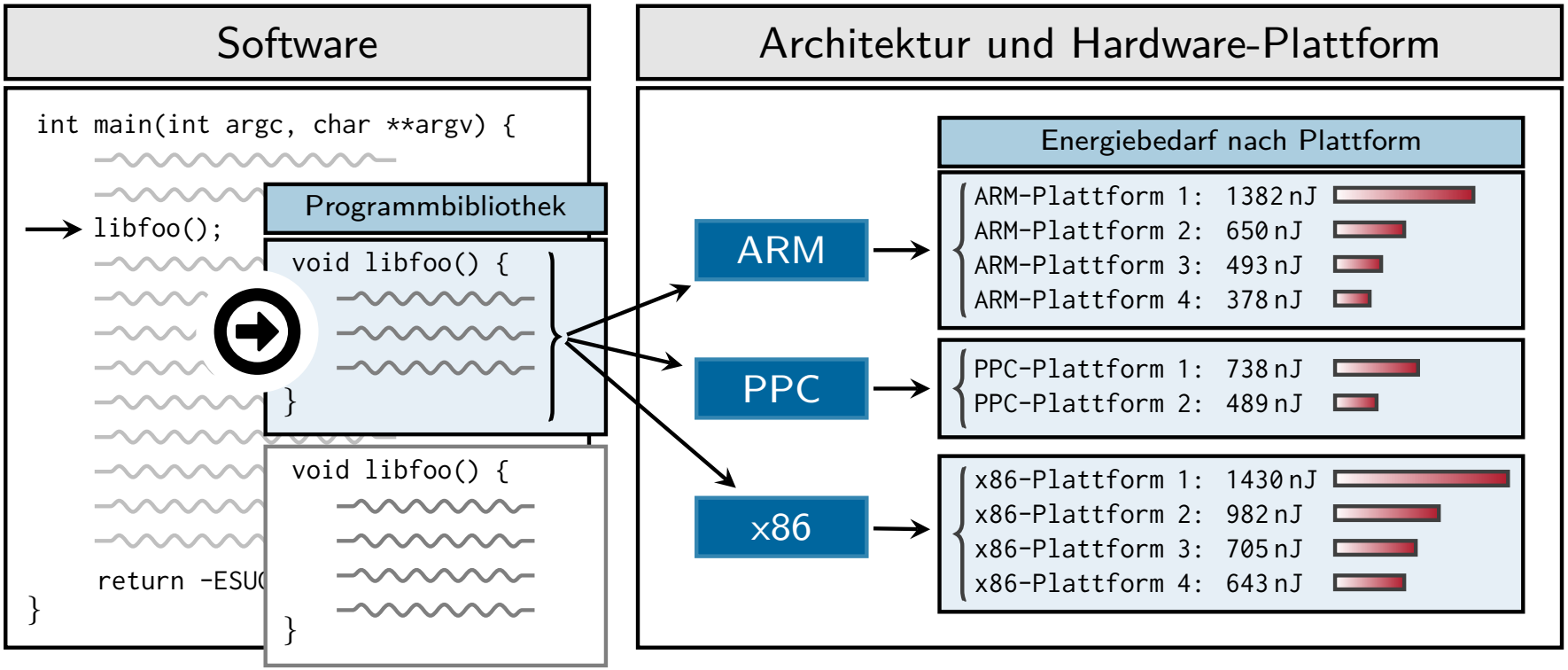
→ Schichtenübergreifende Ansätze

→ Erreichen bislang unausgeschöpfter Einsparpotentiale

Agenda

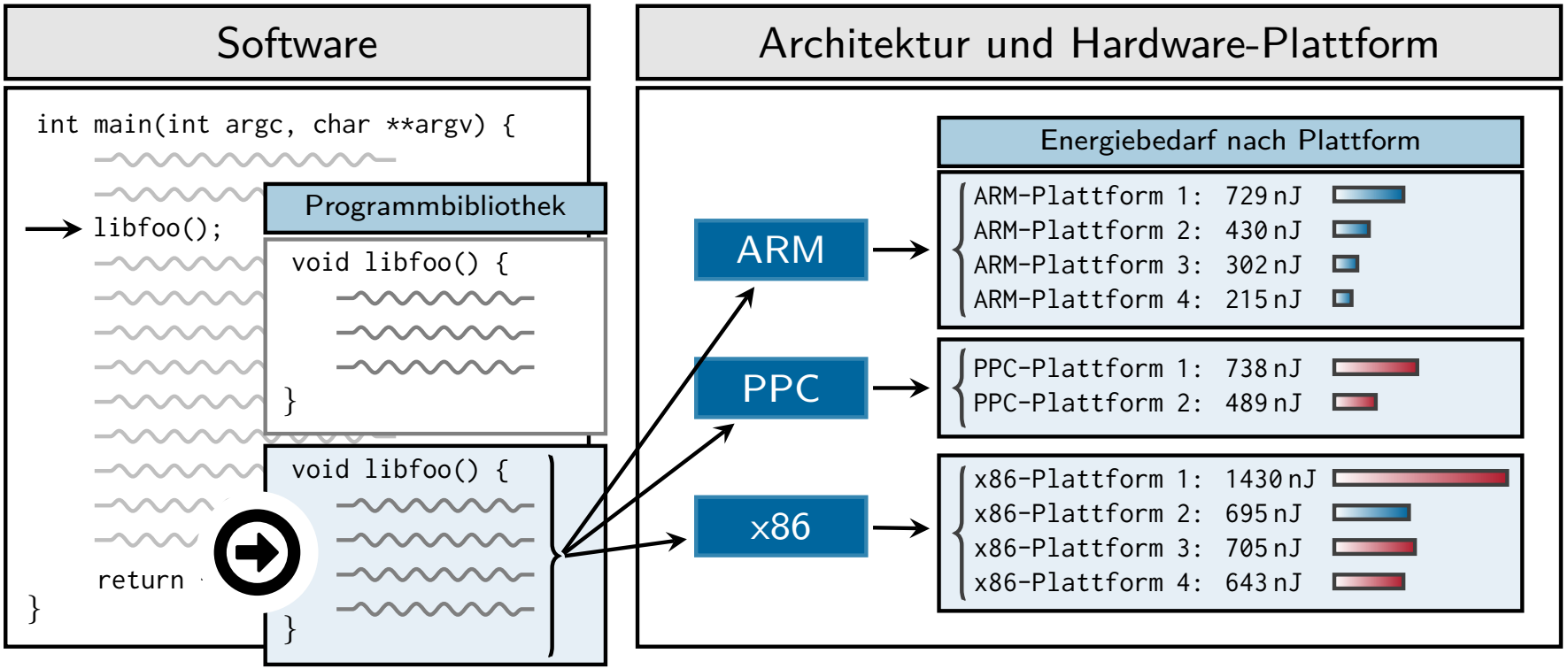
1. Motivation ✔
2. Energieeffiziente Systemsoftware ✔
3. Energiebewusste Programmierung
4. Albatross OS
5. Ausblick: Aktuelle Forschung
6. Zusammenfassung und Schlussfolgerung

Proaktive energiebewusste Programmierung



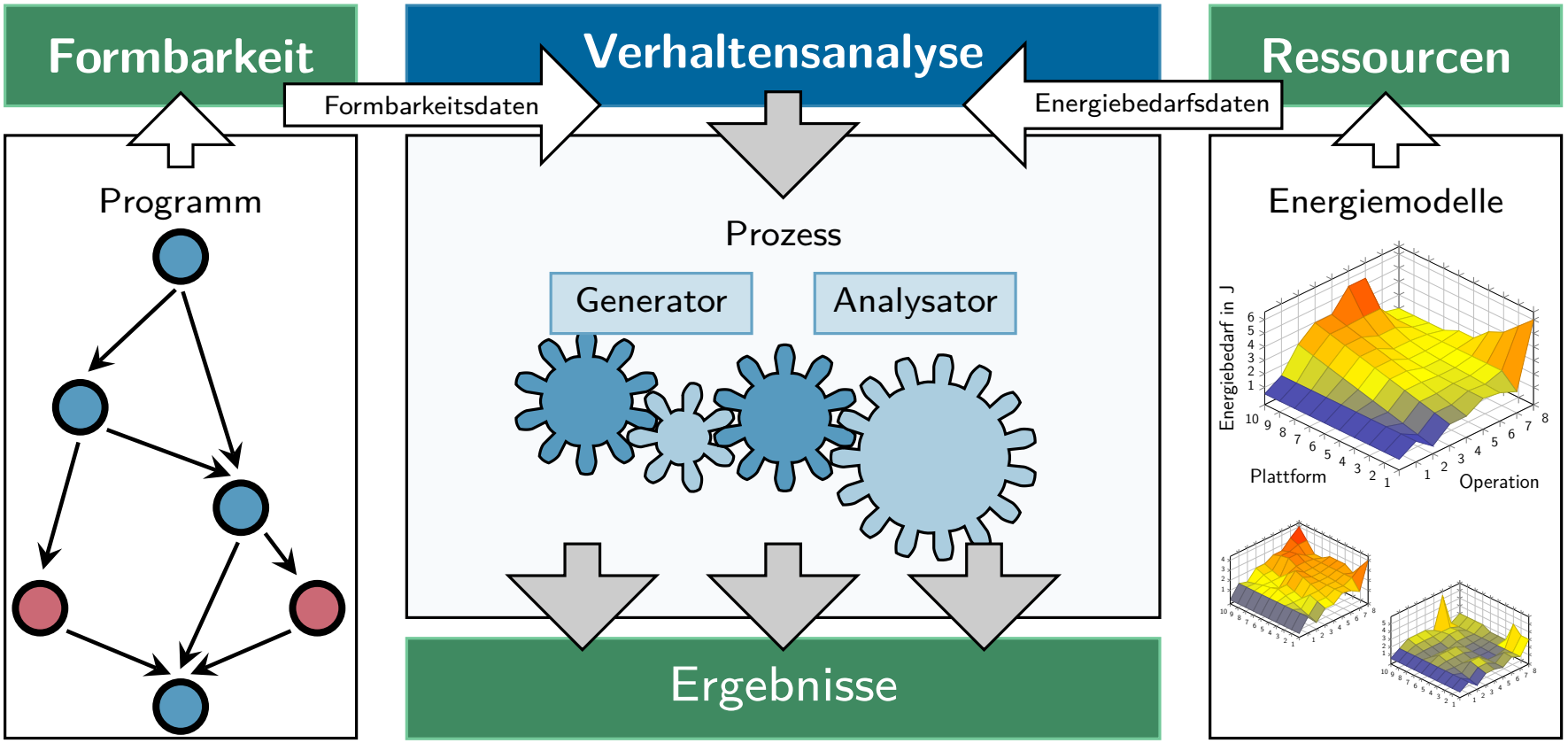
- Bereitstellung von Energiebedarfsvorhersagen auf Funktionsebene bereits während der Software-Entwicklung
- Grundlage für energiebewusste Programmierentscheidungen

Proaktive energiebewusste Programmierung



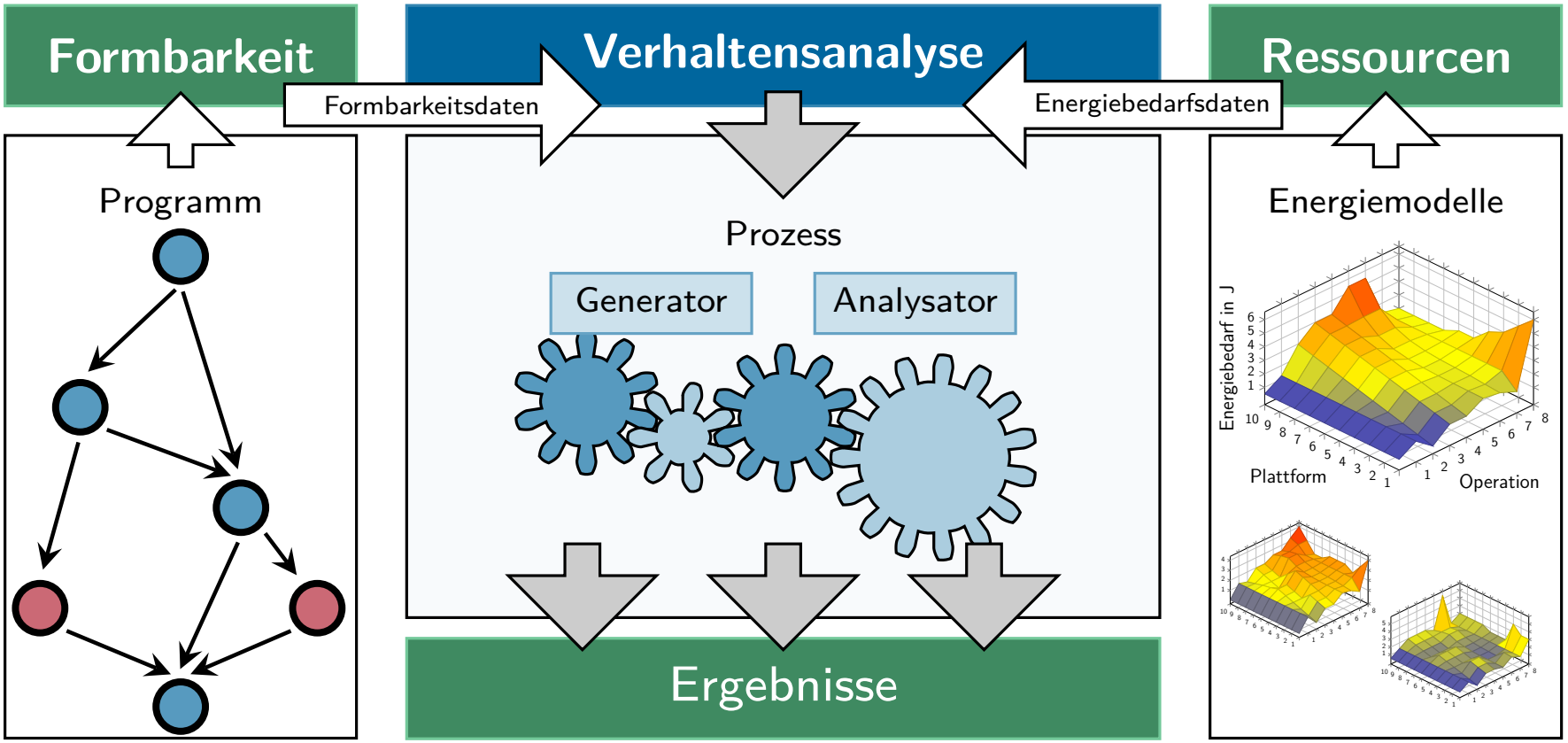
- Bereitstellung von Energiebedarfsvorhersagen auf Funktionsebene bereits während der Software-Entwicklung
- Grundlage für energiebewusste Programmierentscheidungen

Architektur und Implementierung



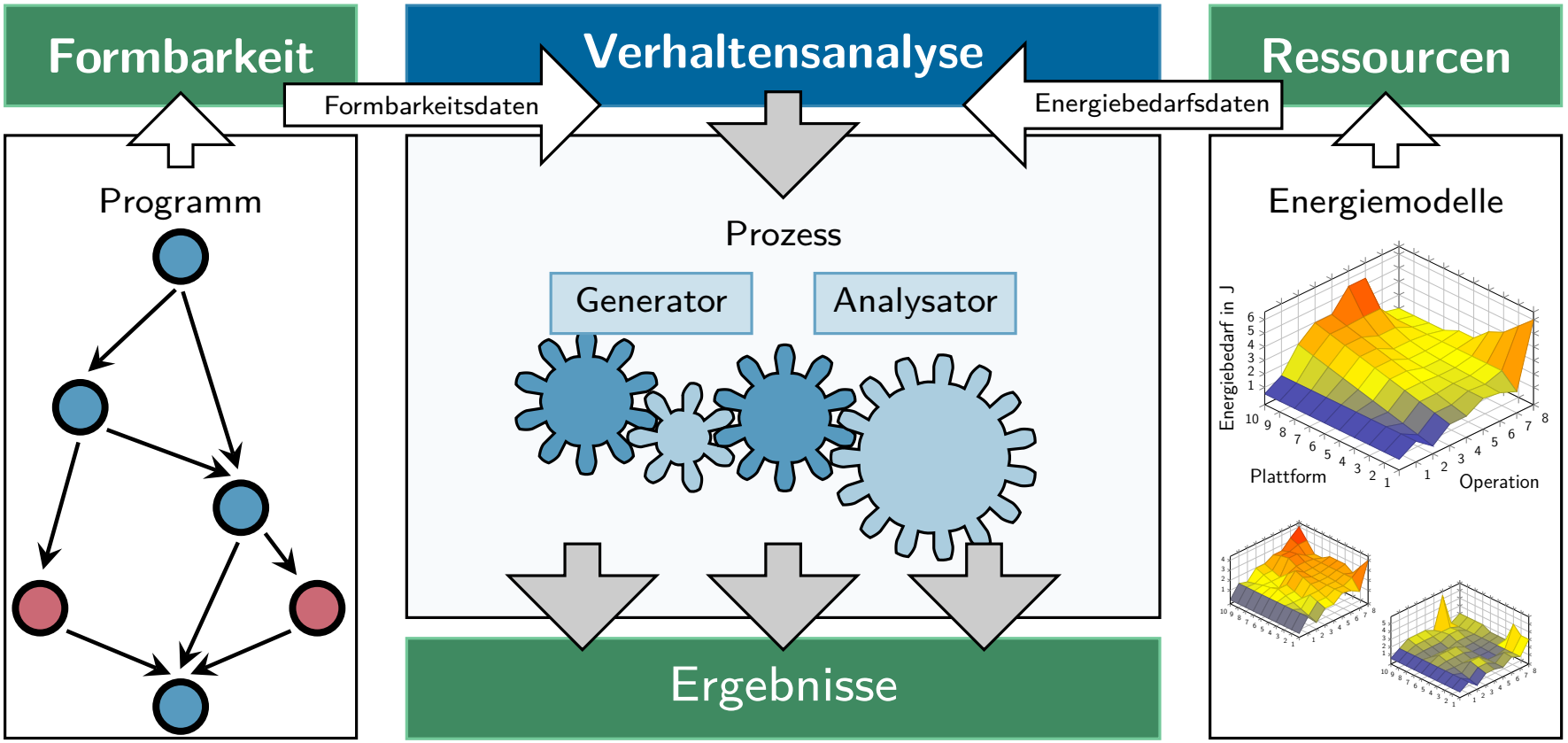
- Formbarkeitsfeststellung durch Programmanalyse
- Laufzeitanalyse mit Prozessausführung und -bewertung
- Ressourcenbedarfsanalyse anhand von Energiemodellen

Architektur und Implementierung



- Formbarkeitsfeststellung durch Programmanalyse
- Laufzeitanalyse mit Prozessausführung und -bewertung
- Ressourcenbedarfsanalyse anhand von Energiemodellen

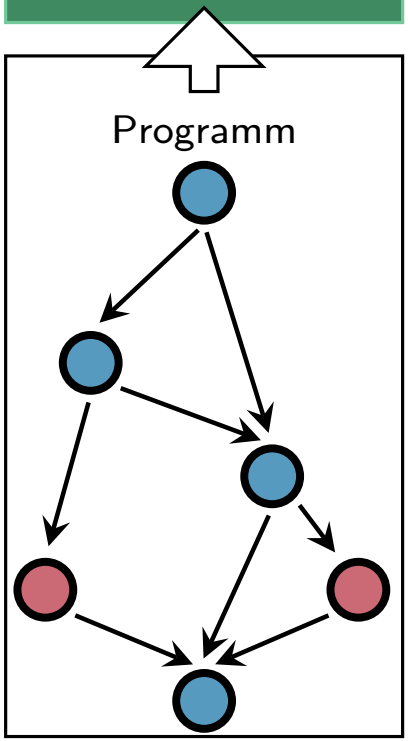
Architektur und Implementierung



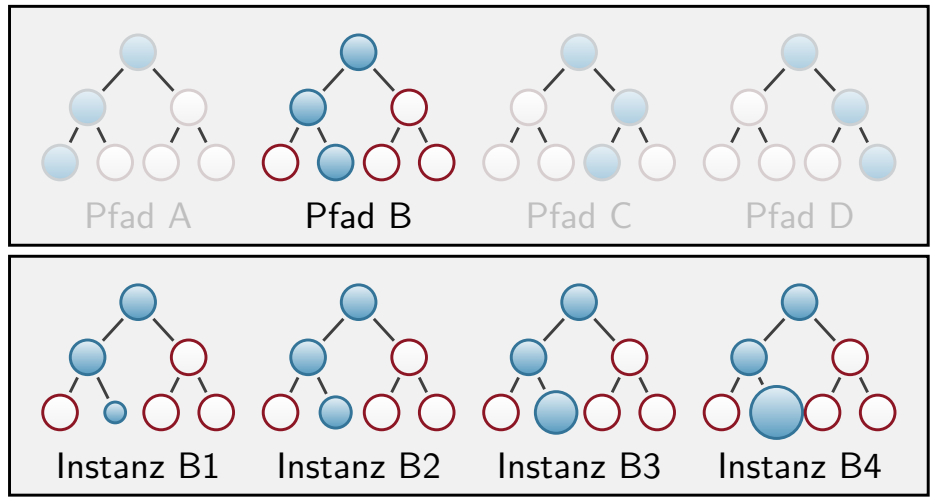
- Formbarkeitsfeststellung durch Programmanalyse
- Laufzeitanalyse mit Prozessausführung und -bewertung
- Ressourcenbedarfsanalyse anhand von Energiemodellen

Architektur und Implementierung

Formbarkeit | Verhaltensanalyse | Ressourcen



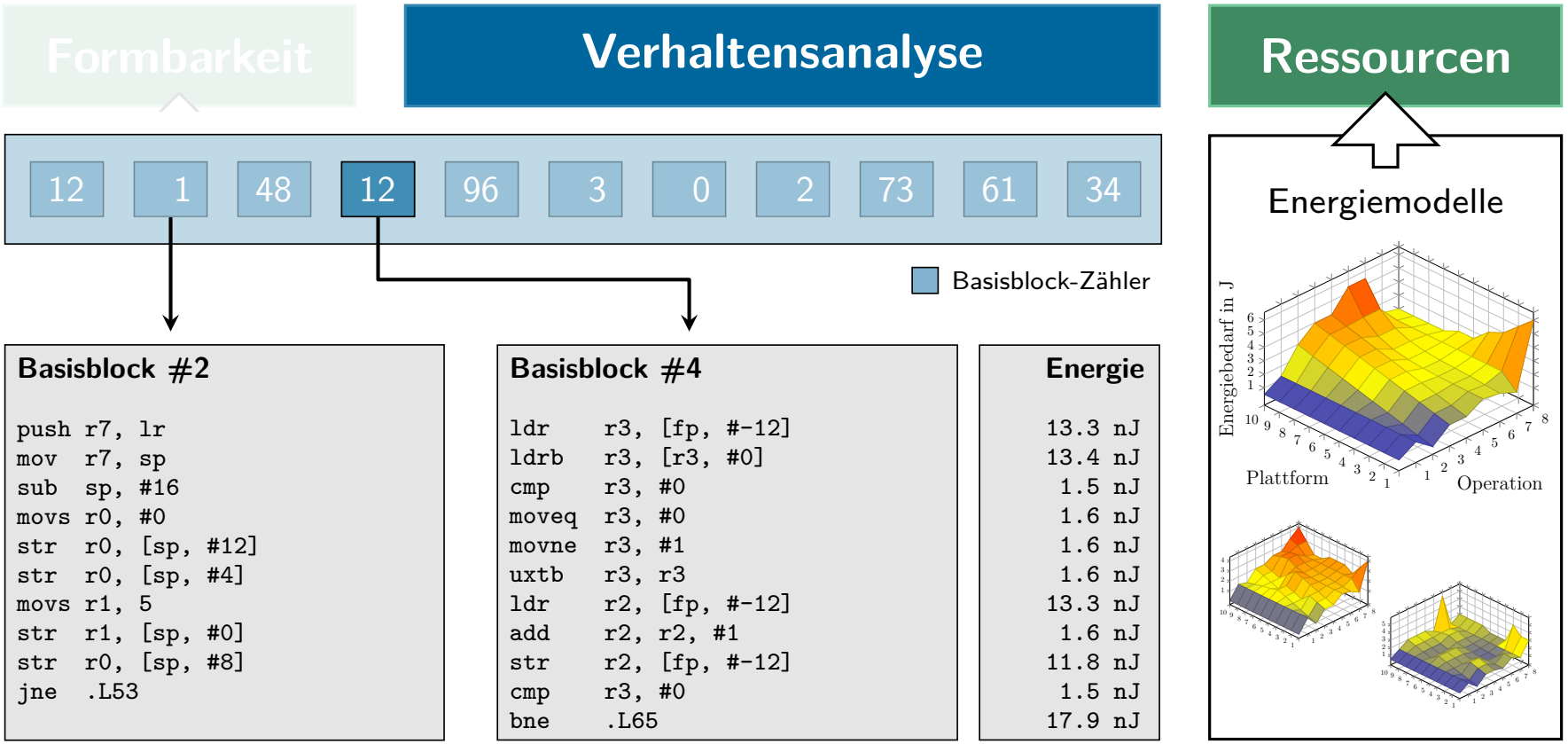

```
 1: if x > 0 then
 2:   if x > 4 then
 3:     ... /* Pfad A */
 4:   else
 5:     ... /* Pfad B */
 6:   end if
 7: else
 8:   if x < −16 then
 9:     ... /* Pfad C */
10:   else
11:     ... /* Pfad D */
12:   end if
13: end if
```



- Bestimmung der Programmpfade (symbolische Ausführung)
- Erzeugung allein lauffähiger Programme für die Pfadinstanzen

- Formbarkeitsfeststellung durch Programmanalyse
- Laufzeitanalyse mit Prozessausführung und -bewertung
- Ressourcenbedarfsanalyse anhand von Energiemodellen

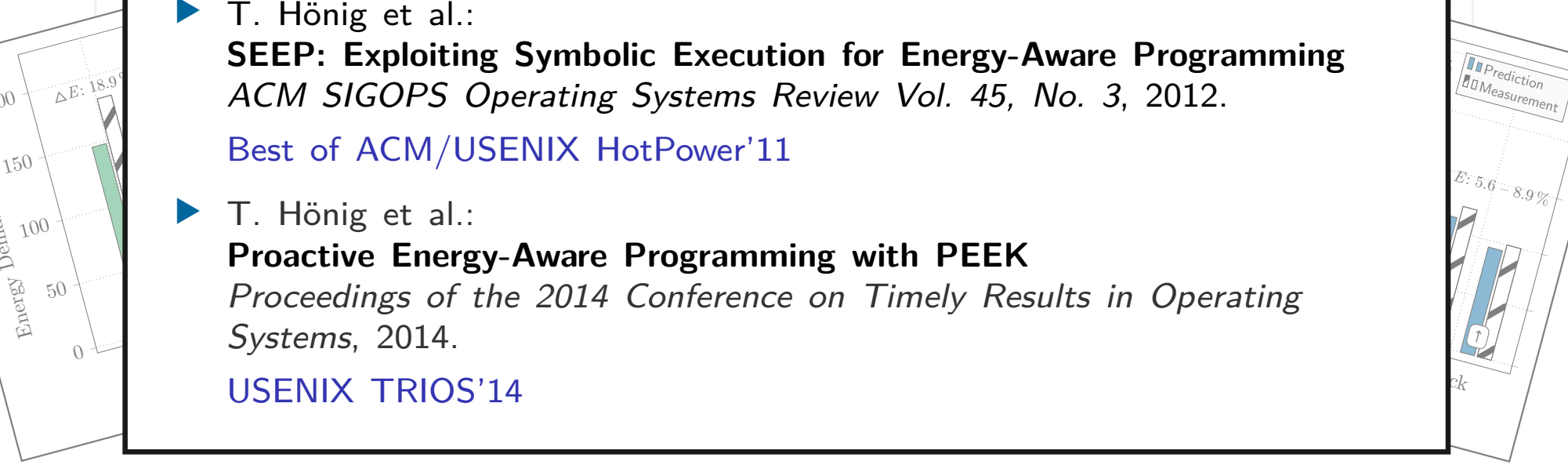
Architektur und Implementierung



- Formbarkeitsfeststellung durch Programmanalyse
- Laufzeitanalyse mit Prozessausführung und -bewertung
- Ressourcenbedarfsanalyse anhand von Energiemodellen

Proaktive energiebewusste Programmierung: Ergebnisse

- Energiebedarfsprognose weicht durchschnittlich um weniger als 9,1 % im Vergleich zu einer Energiemessung ab
- Die Evaluierung zeigt, dass der Energiebedarf funktional identischer Prozesse bis um das 3,9-fache abweicht



- T. Hönig et al.:
  SEEP: Exploiting Symbolic Execution for Energy-Aware Programming
  ACM SIGOPS Operating Systems Review Vol. 45, No. 3, 2012.
  Best of ACM/USENIX HotPower'11
- T. Hönig et al.:
  Proactive Energy-Aware Programming with PEEK
  Proceedings of the 2014 Conference on Timely Results in Operating Systems, 2014.
  USENIX TRIOS'14

Proaktive energiebewusste Programmierung: Ergebnisse

- ✔ Vergleich von (funktional identischen) Programmen, die unterschiedliche nicht-funktionale Eigenschaften besitzen
- ✔ An Entwicklungsprozess gekoppelte Bewertungsmethode zur Feststellung des Energiebedarfs von Software

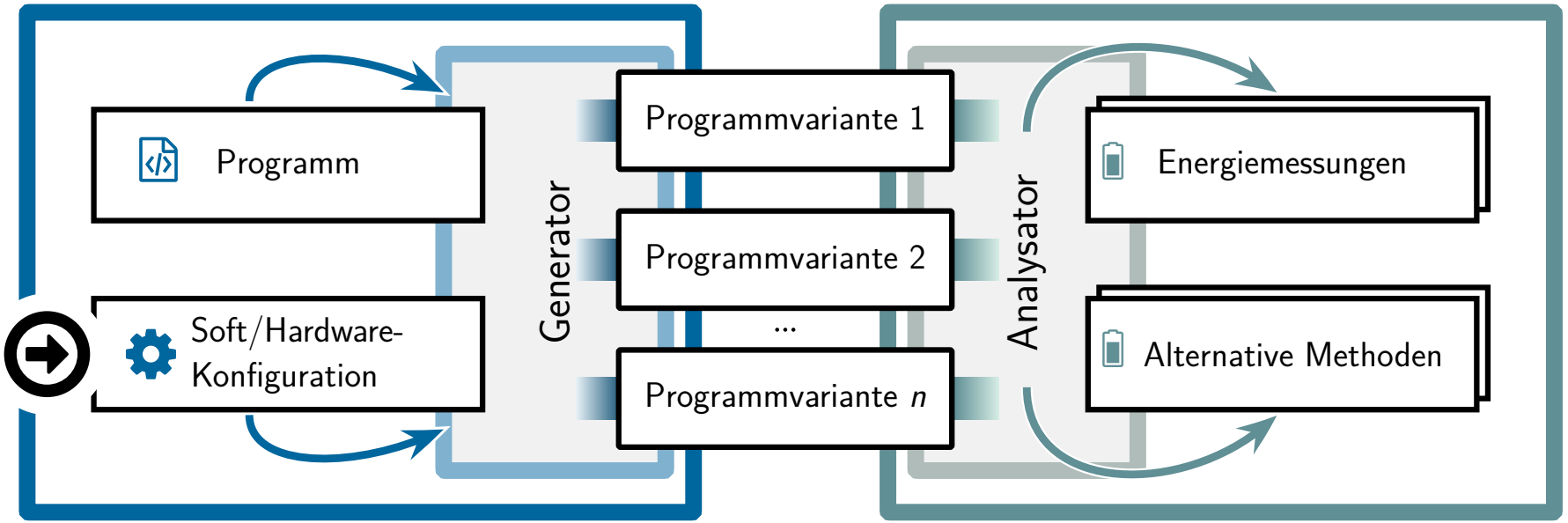
- T. Hönig et al.:
  SEEP: Exploiting Symbolic Execution for Energy-Aware Programming
  ACM SIGOPS Operating Systems Review Vol. 45, No. 3, 2012.
  Best of ACM/USENIX HotPower'11
- T. Hönig et al.:
  Proactive Energy-Aware Programming with PEEK
  Proceedings of the 2014 Conference on Timely Results in Operating Systems, 2014.
  USENIX TRIOS'14

Proaktive energiebewusste Programmierung: Ergebnisse

- ✖ Unausgeschöpftes Einsparpotential durch Vorabanalyse laufzeitabhängiger Energiesparmechanismen
- ✖ Fehlende und ungenaue Energiemodelle für Hardware-komponenten sind der Regelfall

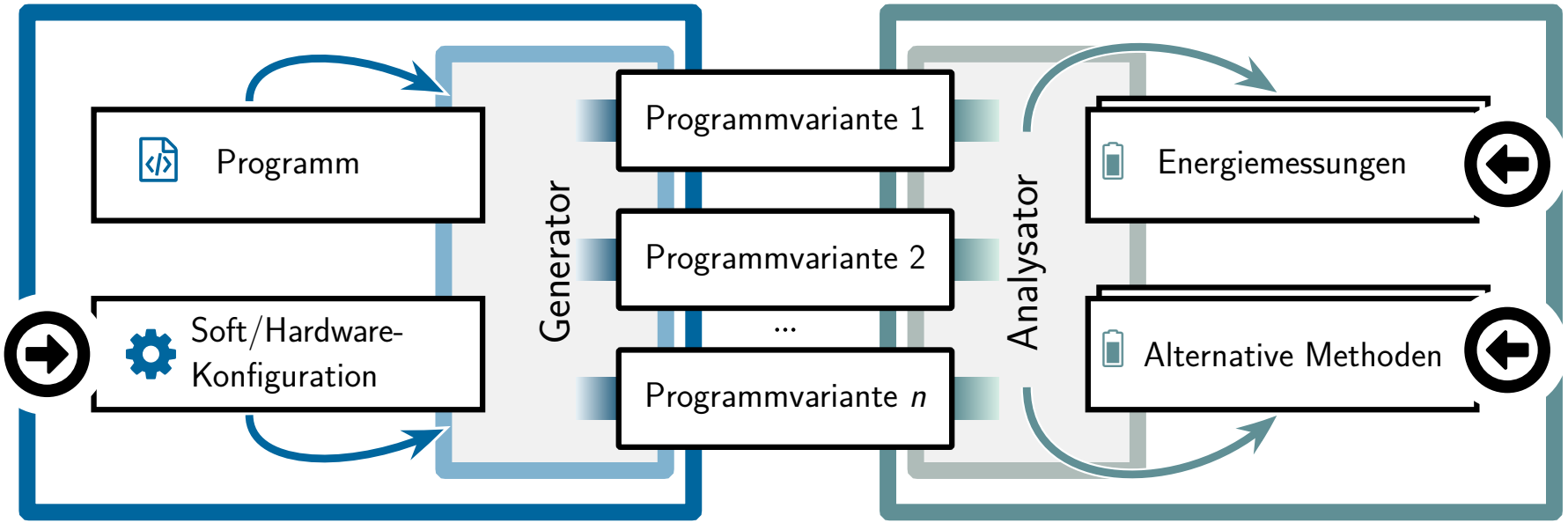
- T. Hönig et al.:
  SEEP: Exploiting Symbolic Execution for Energy-Aware Programming
  ACM SIGOPS Operating Systems Review Vol. 45, No. 3, 2012.
  Best of ACM/USENIX HotPower'11
- T. Hönig et al.:
  Proactive Energy-Aware Programming with PEEK
  Proceedings of the 2014 Conference on Timely Results in Operating Systems, 2014.
  USENIX TRIOS'14

Programmvarianten-Generator und -Analyse



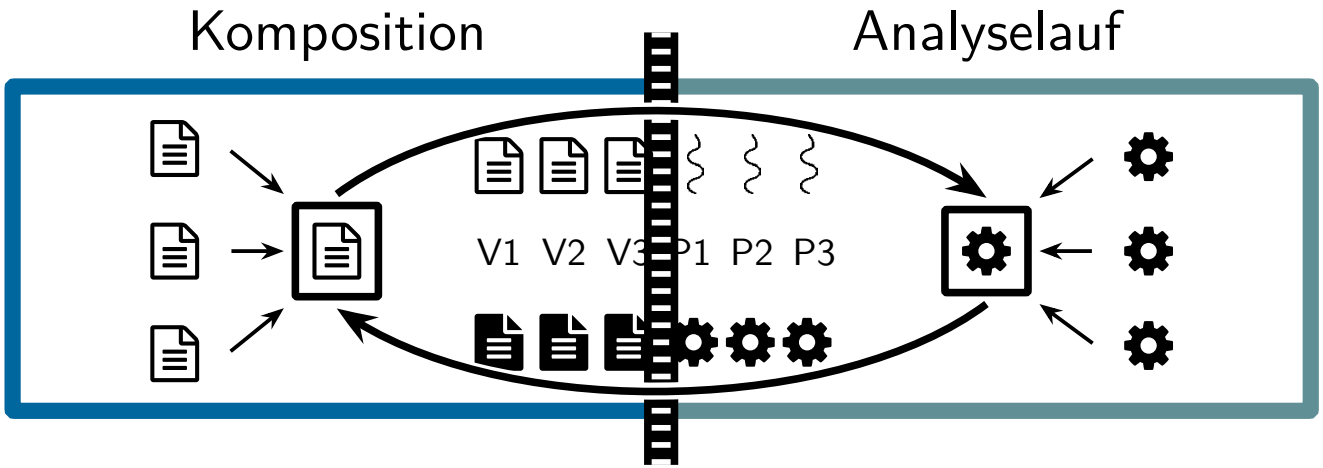
- Verknüpfung von Programmen mit unterschiedlichen Soft/Hardware-Konfigurationen zur Variantengeneration

Programmvarianten-Generator und -Analyse



- Verknüpfung von Programmen mit unterschiedlichen Soft/Hardware-Konfigurationen zur Variantengeneration
- Energiemessungen mit einer auf einem Stromspiegel basierenden Messschaltung zur Bestimmung des Energiebedarfs
- Energiemodelle mittels künstlicher neuronaler Netze

Programmvarianten: Experimente und Ergebnisse



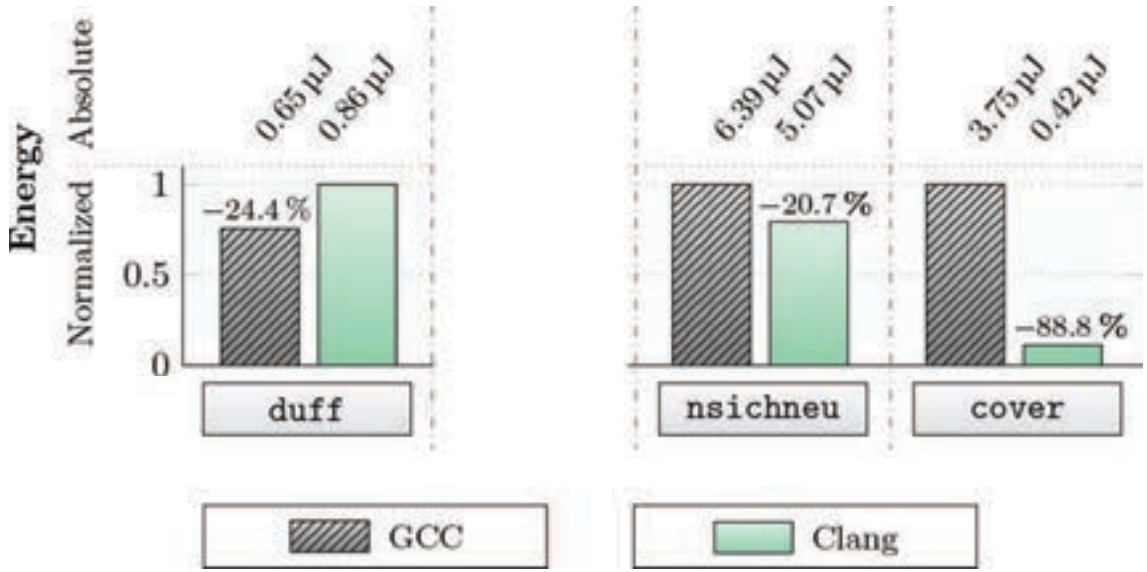
Erstes Experiment[1]: Vergleich schnittstellenkompatibler Übersetzer

GCC vs. Clang

- GCC generiert in 80 % der Fälle energieeffizientere Programmvarianten (bis zu einem Viertel niedrigerer Energiebedarf)
- Eine Programmvariante von Clang ist ca. 10 x energieeffizienter ist als die entsprechende Variante von GCC

[1] Software: GNU GCC 4.8, LLVM Clang 3.4, Hardware: ARM Cortex-M0+ (Kinetis KL02)

Programmvarianten: Experimente und Ergebnisse



GCC vs. Clang

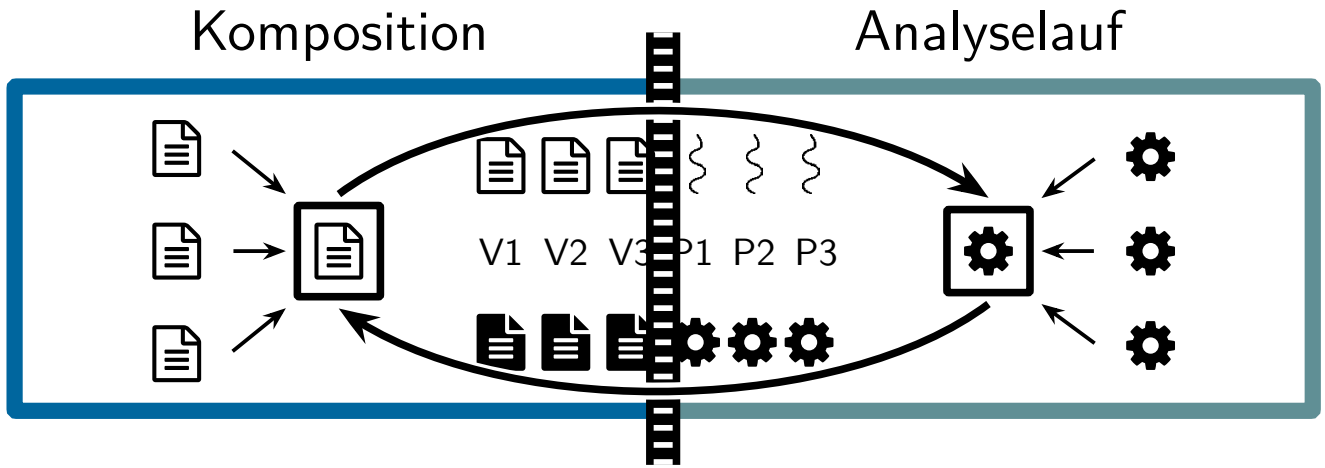
- GCC generiert in 80 % der Fälle energieeffizientere Programmvarianten (bis zu einem Viertel niedrigerer Energiebedarf)
- Eine Programmvariante von Clang ist ca. 10 x energieeffizienter ist als die entsprechende Variante von GCC

Energie vs. Zeit

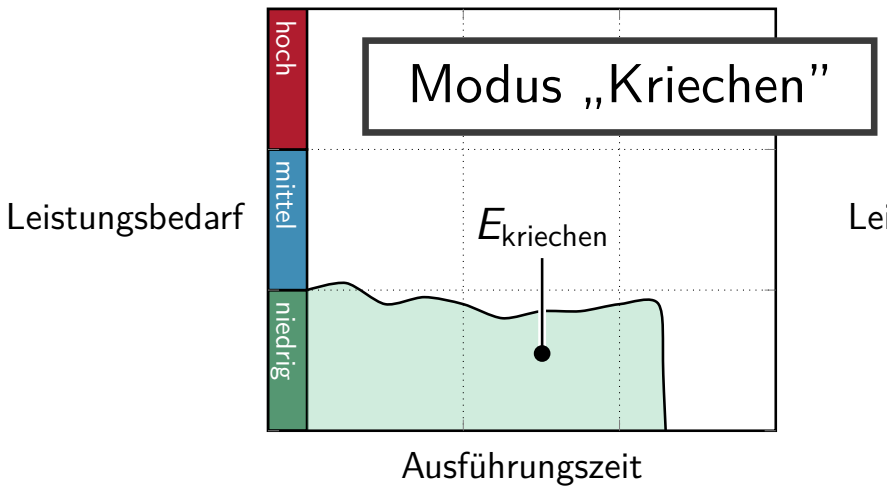
- Kein kausaler Zusammenhang von Energiebedarf und Prozesslaufzeit in 10 % der durchgeführten Programmanalysen

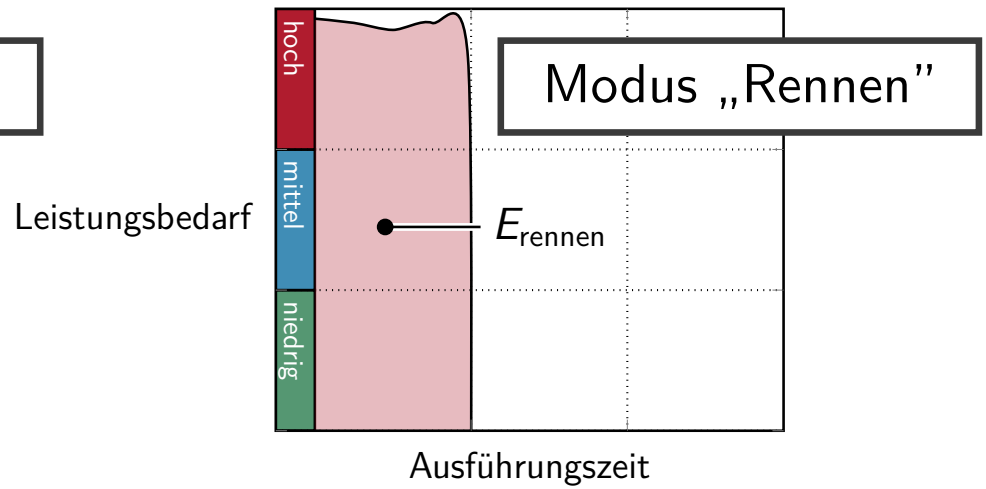
[1] Software: GNU GCC 4.8, LLVM Clang 3.4, Hardware: ARM Cortex-M0+ (Kinetis KL02)

Programmvarianten: Experimente und Ergebnisse



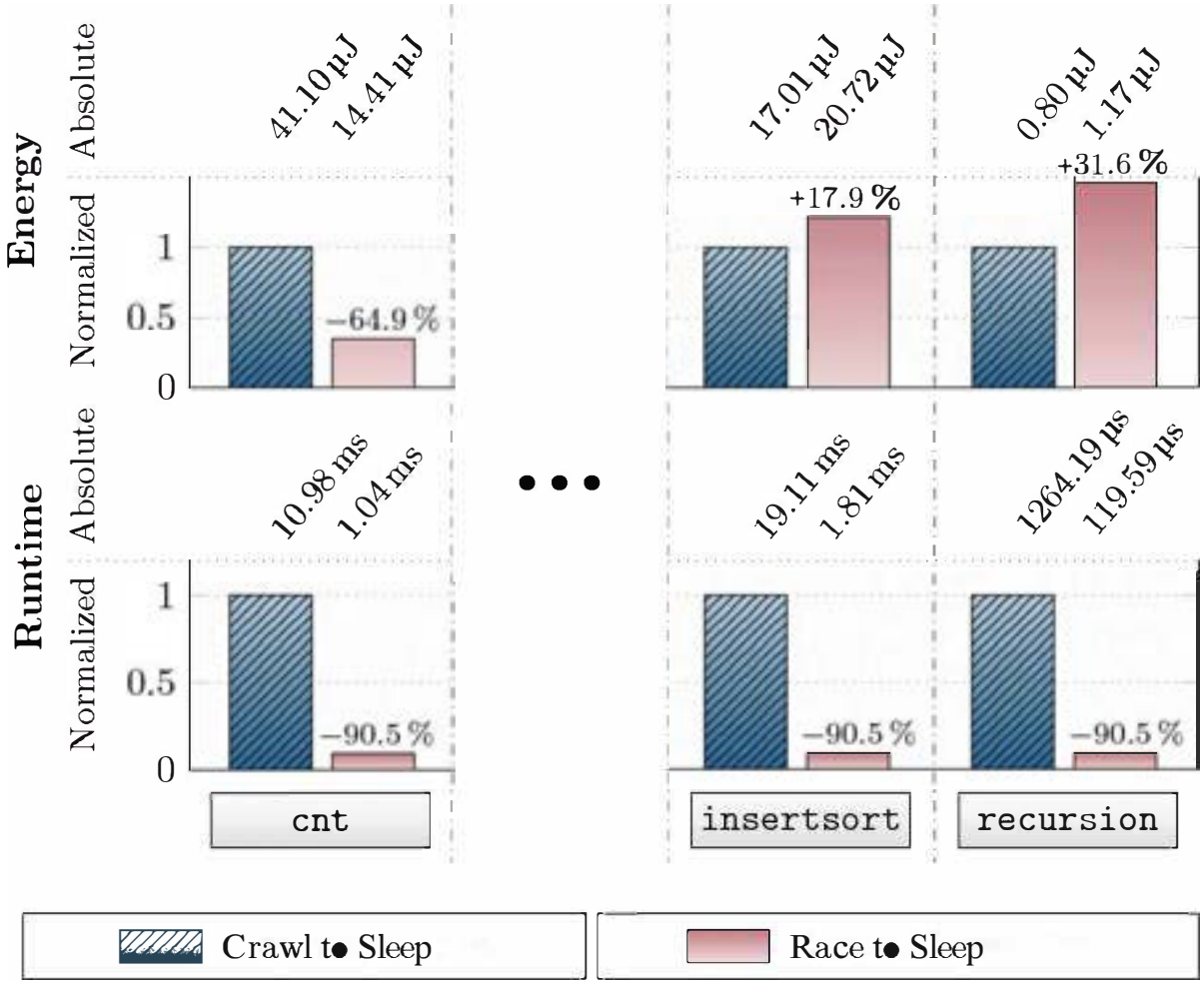
Zweites Experiment[2]: Skalierung von Betriebsspannung und Taktfrequenz



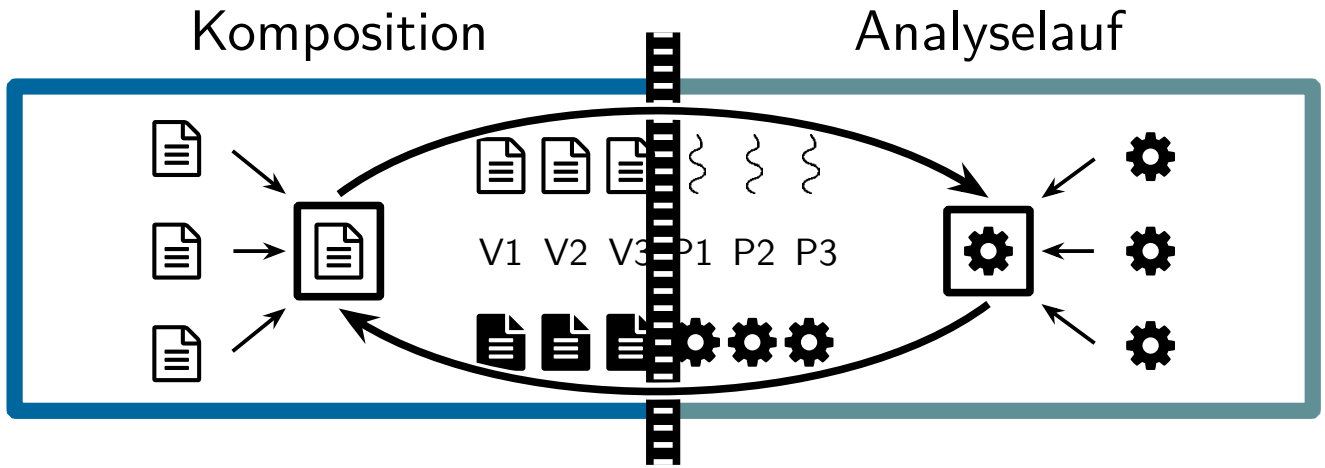


[2]Software: GNU GCC 4.8, Hardware: ARM Cortex-M0+ (Kinetis KL02, RUN/VLPR)

Programmvarianten: Experimente und Ergebnisse



Programmvarianten: Experimente und Ergebnisse



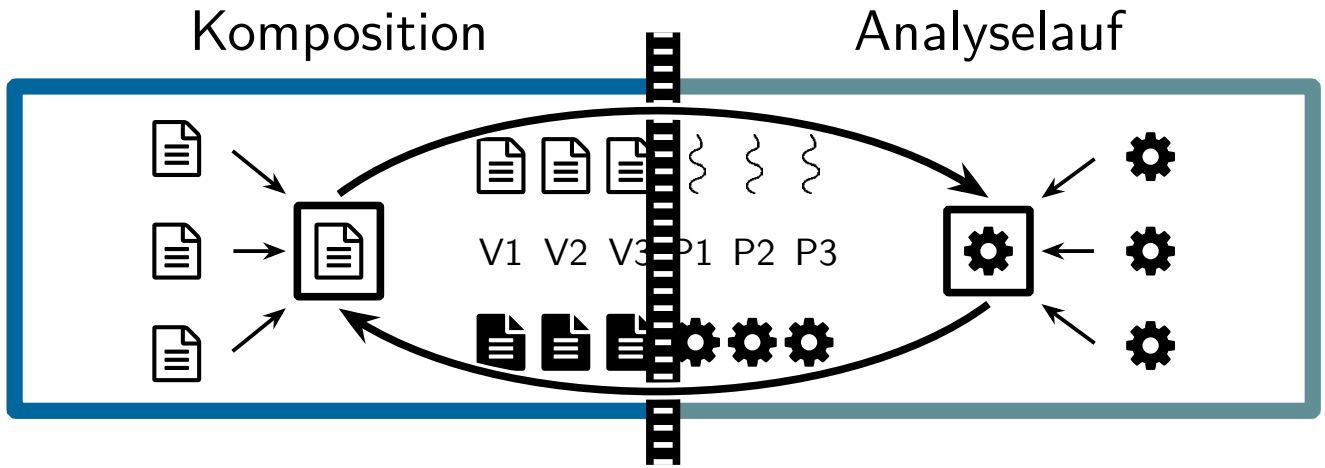
Zweites Experiment[2]: Skalierung von Betriebsspannung und Taktfrequenz

Rennen vs. Kriechen

- Modus „Rennen" wird gewöhnlich bevorzugt, um die Leerlaufzeit zu maximieren (→ Nutzung von Schlafzuständen)
- Zu erwartende Performancesteigerung tritt in allen Testfällen ein → Reduktion der Ausführungszeit

[2] Software: GNU GCC 4.8, Hardware: ARM Cortex-M0+ (Kinetis KL02, RUN/VLPR)

Programmvarianten: Experimente und Ergebnisse



Zweites Experiment[2]: Skalierung von Betriebsspannung und Taktfrequenz

Rennen vs. Kriechen

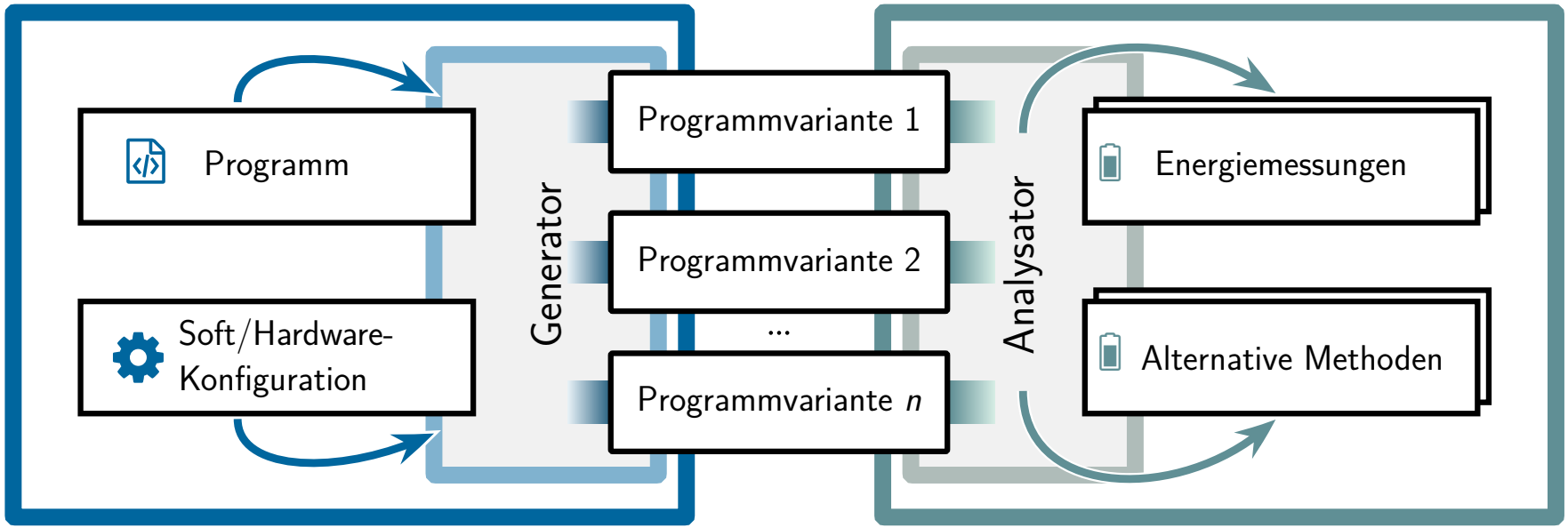
- Modus „Rennen" wird gewöhnlich bevorzugt, um die Leerlaufzeit zu maximieren (→ Nutzung von Schlafzuständen)
- Zu erwartende Performancesteigerung tritt in allen Testfällen ein → Reduktion der Ausführungszeit

Energie vs. Zeit

- **Weniger ist mehr:** In 20 % der Programmanalysen führt „Kriechen" zu *höherer* Energieeinsparung

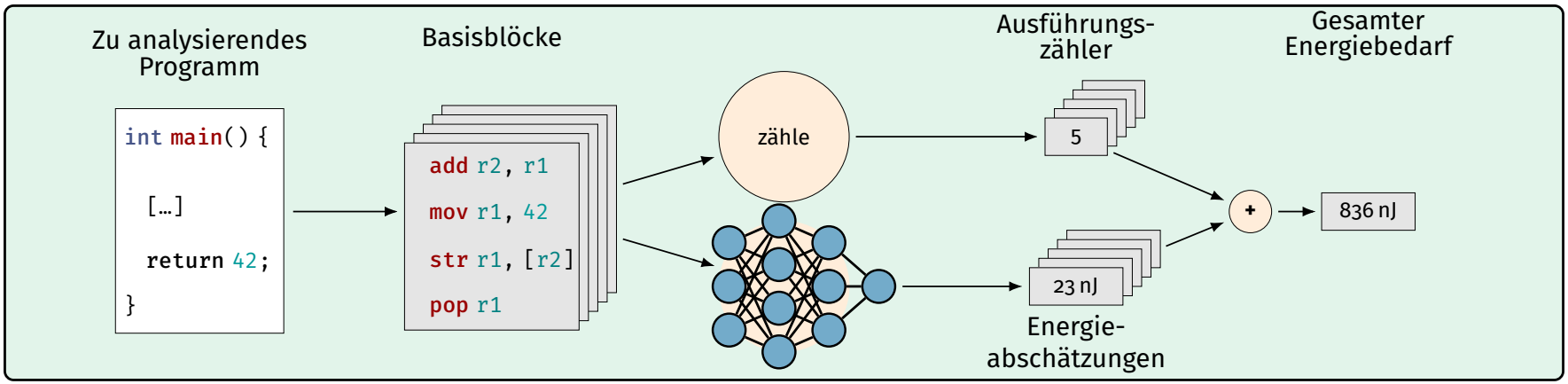
[2] Software: GNU GCC 4.8, Hardware: ARM Cortex-M0+ (Kinetis KL02, RUN/VLPR)

Programmvarianten-Generator und -Analyse



- Verknüpfung von Programmen mit unterschiedlichen Soft/Hardware-Konfigurationen zur Variantengeneration ✓
- Energiemessungen mit einer auf einem Stromspiegel basierenden Messschaltung zur Bestimmung des Energiebedarfs ✓
- Energiemodelle mittels künstlicher neuronaler Netze

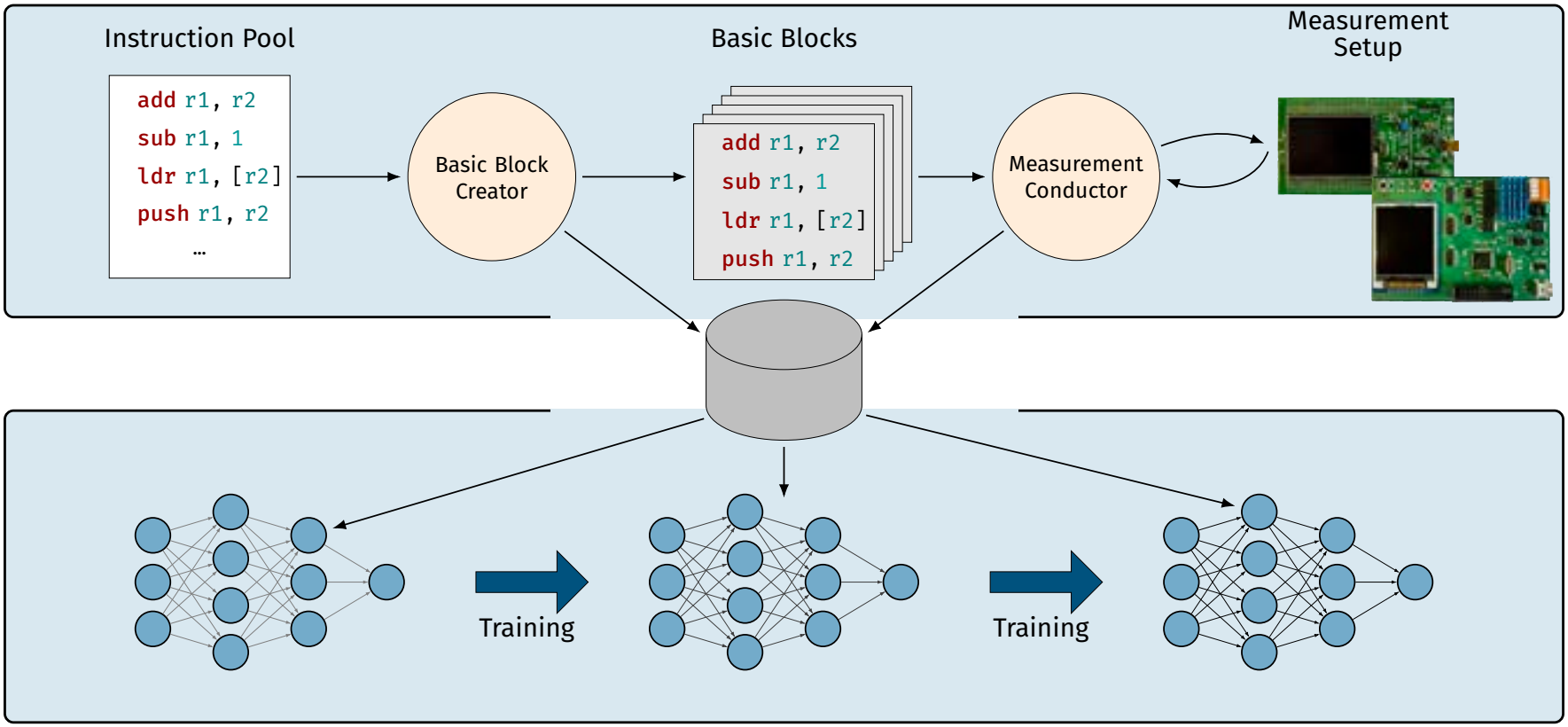
Energiemodelle mittels künstlicher neuronaler Netze



- Energieabschätzung für individuelle Programmpfade
- Zusammenführen der Ausführungszähler und Energieabschätzungen
- Energieabschätzung durch neuronales Netz

T. Hönig et al.:
Energy-Demand Estimation of Embedded Devices Using Deep Artificial Neural Networks
Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing, 2019.
ACM SAC'19

Energiemodelle mittels künstlicher neuronaler Netze

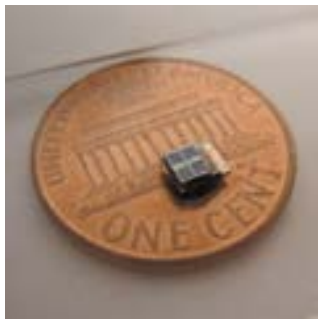


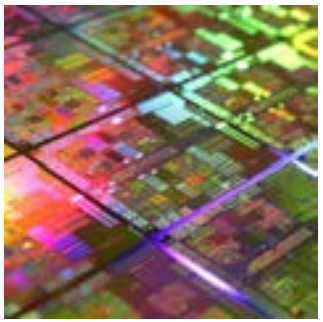
T. Hönig et al.:
Energy-Demand Estimation of Embedded Devices Using Deep Artificial Neural Networks
Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing, 2019.

ACM SAC'19

Betriebssystem für energieeffiziente HPC-Systeme

- elektrische Energie ist wichtigste Betriebsressource für Computer




Embedded —— Laptop/Desktop —— HPC/Mainframe

✓ ✓ ↑

- Albatross OS: Betriebssystem für energieeffiziente HPC-Systeme
- Motivation: Ausnutzung variabler Elektrizitätspreise und Leistungsbedarf heterogener Verarbeitungseinheiten

Agenda

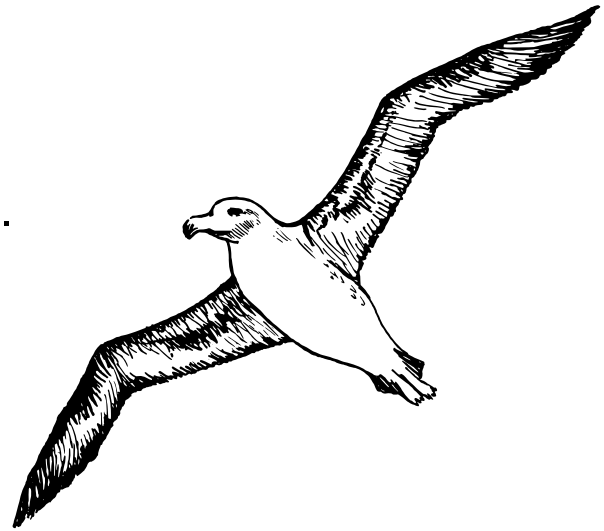
1. Motivation ✔
2. Energieeffiziente Systemsoftware ✔
3. Energiebewusste Programmierung ✔
4. Albatross OS
5. Ausblick: Aktuelle Forschung
6. Zusammenfassung und Schlussfolgerung

Offshore-Windpark Gode Wind: 582 MW

Nordsee, Europa

Aufbau eines Betriebssystems für heterogene HPC-Cluster

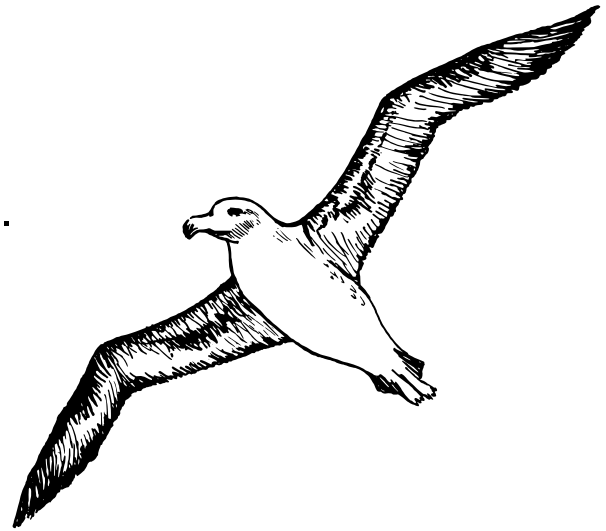
- Erneuerbare Energiequellen haben einen großen Einfluss auf das Elektrizitätsnetz.



- T. Hönig et al.: **How to Make Profit: Exploiting Fluctuating Electricity Prices with Albatross, A Runtime System for Heterogeneous HPC Clusters.**
  International Workshop on Runtime and Operating Systems for Supercomputers (ROSS'18)
- T. Hönig et al.: **Bridging the Gap: Energy-efficient Execution of Software Workloads on Heterogeneous Hardware Components**
  ACM International Conference on Future Energy Systems, Poster (e-Energy'19)

Aufbau eines Betriebssystems für heterogene HPC-Cluster

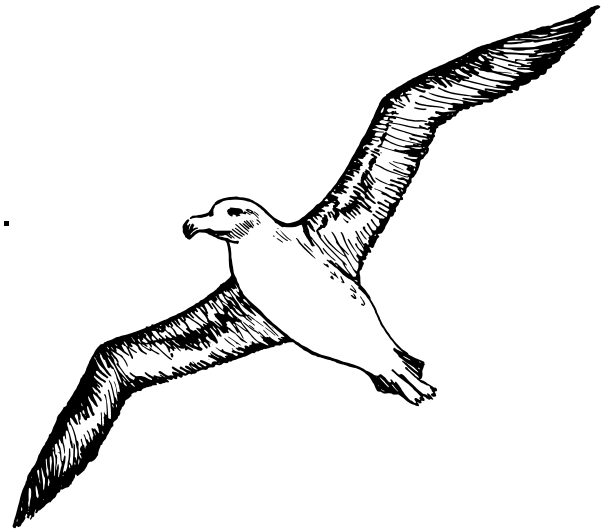
- Erneuerbare Energiequellen haben einen großen Einfluss auf das Elektrizitätsnetz.
- Verfügbarkeit erneuerbarer Energien führt zu schwankenden Strompreisen.



- T. Hönig et al.: **How to Make Profit: Exploiting Fluctuating Electricity Prices with Albatross, A Runtime System for Heterogeneous HPC Clusters.**
  International Workshop on Runtime and Operating Systems for Supercomputers (ROSS'18)
- T. Hönig et al.: **Bridging the Gap: Energy-efficient Execution of Software Workloads on Heterogeneous Hardware Components**
  ACM International Conference on Future Energy Systems, Poster (e-Energy'19)

Aufbau eines Betriebssystems für heterogene HPC-Cluster

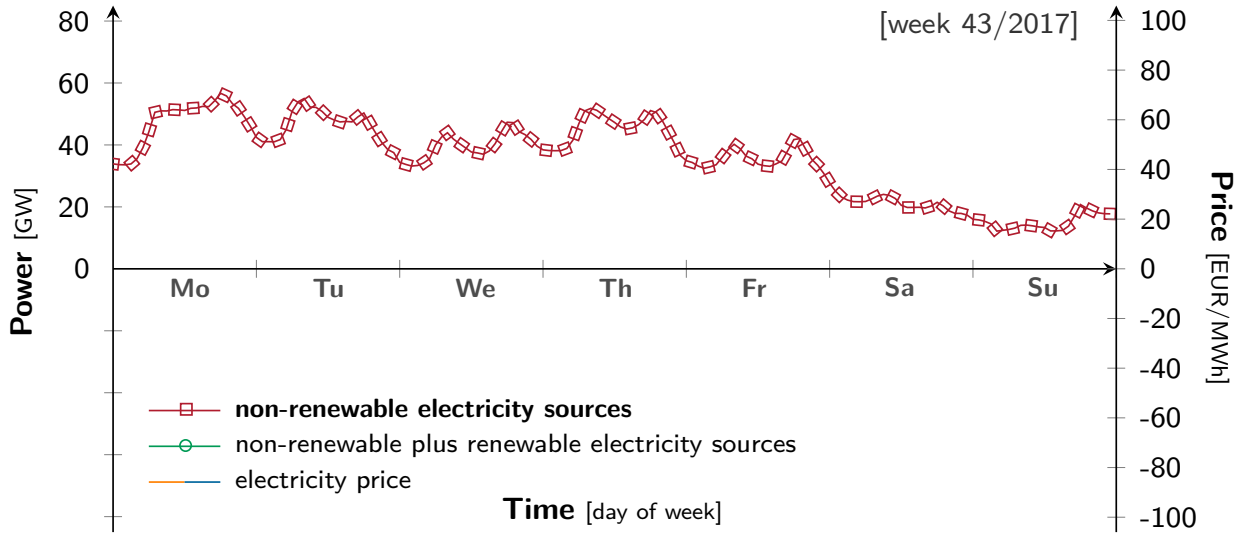
- Erneuerbare Energiequellen haben einen großen Einfluss auf das Elektrizitätsnetz.
- Verfügbarkeit erneuerbarer Energien führt zu schwankenden Strompreisen.



- **Wie können wir ein HPC-Laufzeitsystem entwerfen, um schwankende Strompreise zu nutzen?**

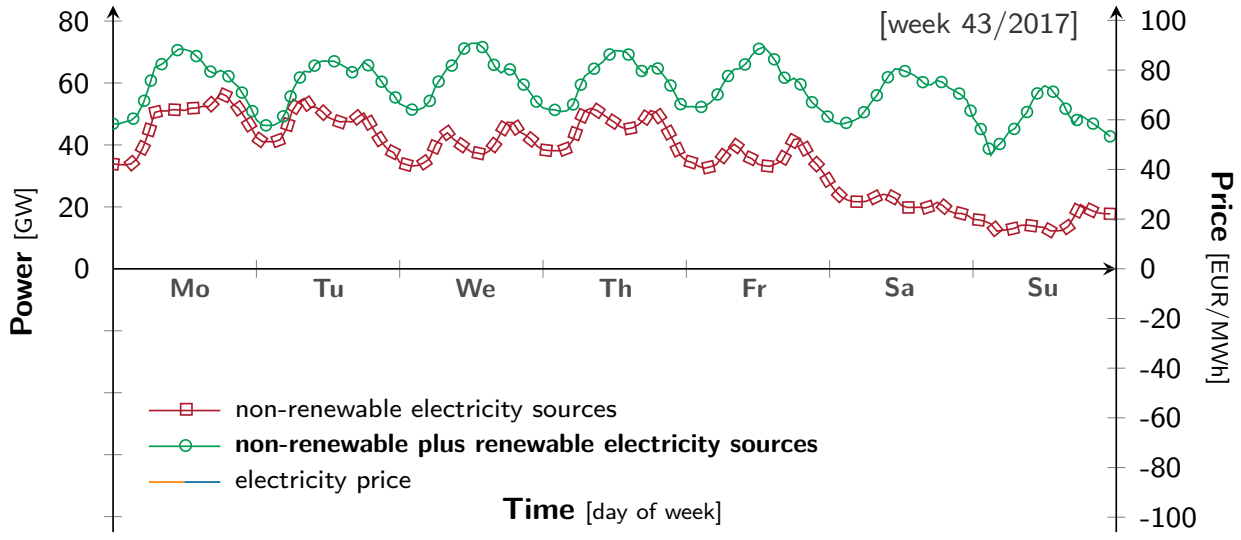
- T. Hönig et al.: **How to Make Profit: Exploiting Fluctuating Electricity Prices with Albatross, A Runtime System for Heterogeneous HPC Clusters.**
  International Workshop on Runtime and Operating Systems for Supercomputers (ROSS'18)
- T. Hönig et al.: **Bridging the Gap: Energy-efficient Execution of Software Workloads on Heterogeneous Hardware Components**
  ACM International Conference on Future Energy Systems, Poster (e-Energy'19)

Schwankungen der Strompreise



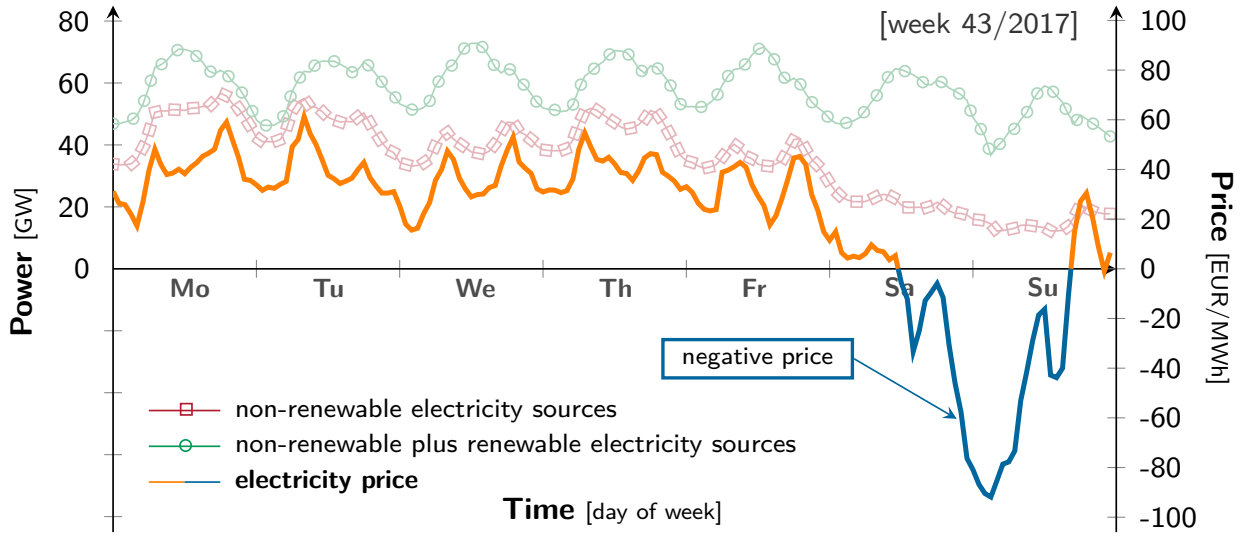
- nicht erneuerbare Energien sind **planbar**...

Schwankungen der Strompreise



- nicht erneuerbare Energien sind **planbar**...
- ...aber erneuerbare Energien sind häufig **unvorhersehbar**

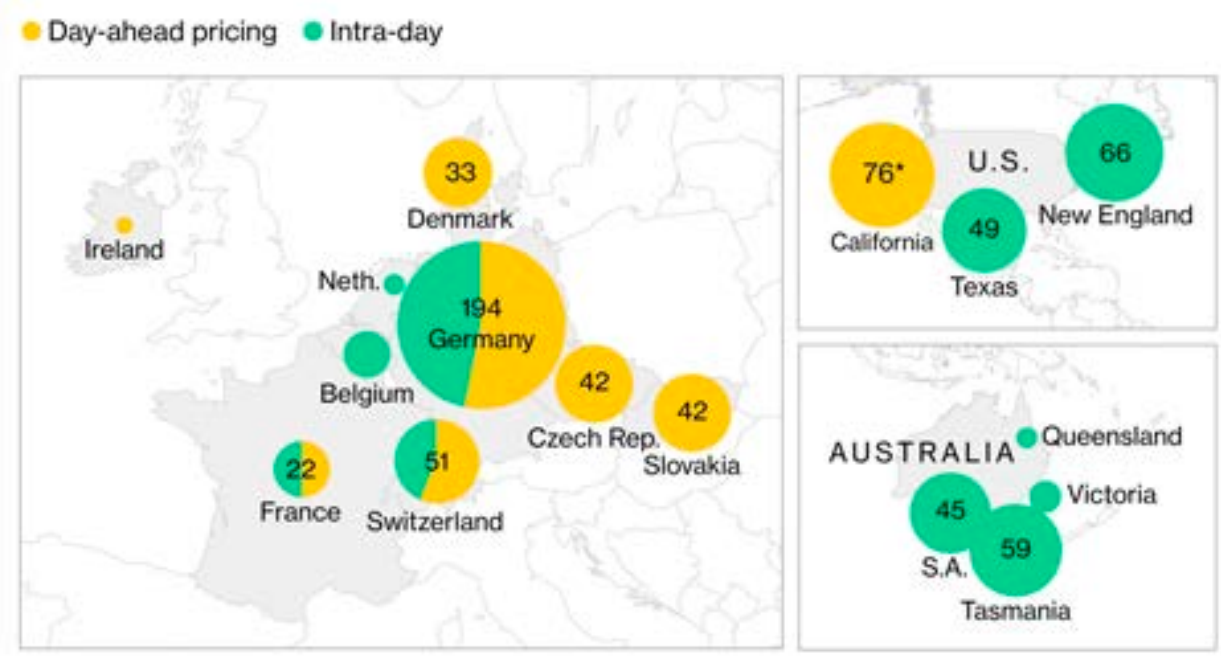
Schwankungen der Strompreise



- Folge: **Starke Strompreisschwankungen**
- bei extremem **Ungleichgewicht von Angebot und Nachfrage**

➜ negative Preise

Schwankungen der Strompreise

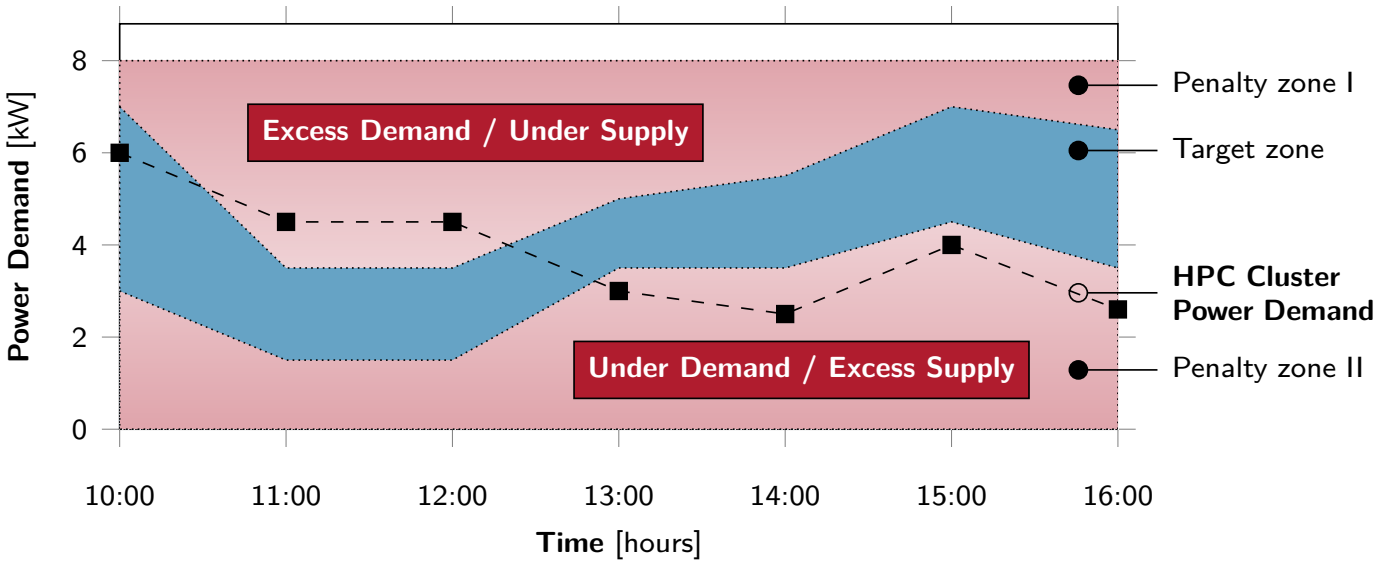


Anzahl der Stunden mit negativen Strompreisen (Q1-Q3 2018, Quelle: Bloomberg)

- Folge: **Starke Strompreisschwankungen**
- bei extremem **Ungleichgewicht von Angebot und Nachfrage**

➔ negative Preise

Aufbau eines Betriebssystems für heterogene HPC-Cluster



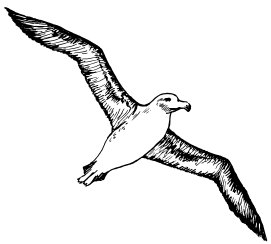
- Herausforderung: Nutzung von niedrigen und negativen Preisen
- Berücksichtigung von Verträgen (→ *penalties*)
- Integration in bestehende HPC-Cluster-Infrastruktur

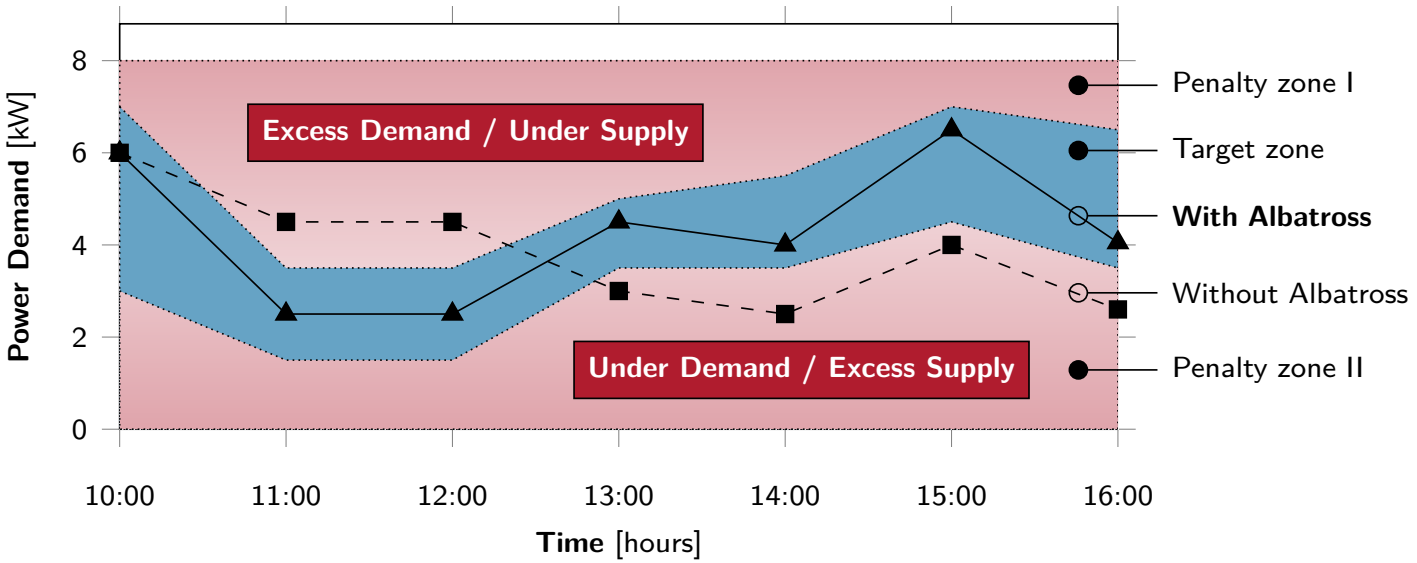
Aufbau eines Betriebssystems für heterogene HPC-Cluster

- Aufbau eines flexiblen Betriebssystems, das Daten über die Strompreise in seine Betriebsentscheidungen integriert
- Reduzierung der Energienachfrage bei hohen Preisen
- Erhöhung der Energienachfrage bei niedrigen (oder negativen) Preisen

- Herausforderung: Nutzung von niedrigen und negativen Preisen
- Berücksichtigung von Verträgen (→ *penalties*)
- Integration in bestehende HPC-Cluster-Infrastruktur

Aufbau eines Betriebssystems für heterogene HPC-Cluster



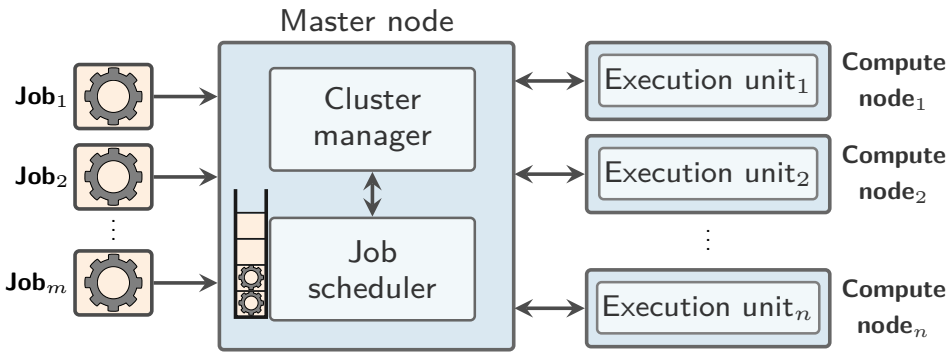


Das vorgeschlagene Betriebssystems Albatross basiert auf Linux und...

- implementiert Betriebsarten mit niedrigem und hohem Leistungsbedarf
- berücksichtigt Heterogenitätsaspekte der Clusterhardware
- respektiert QoS und nicht-funktionale Anforderungen der Arbeitslasten

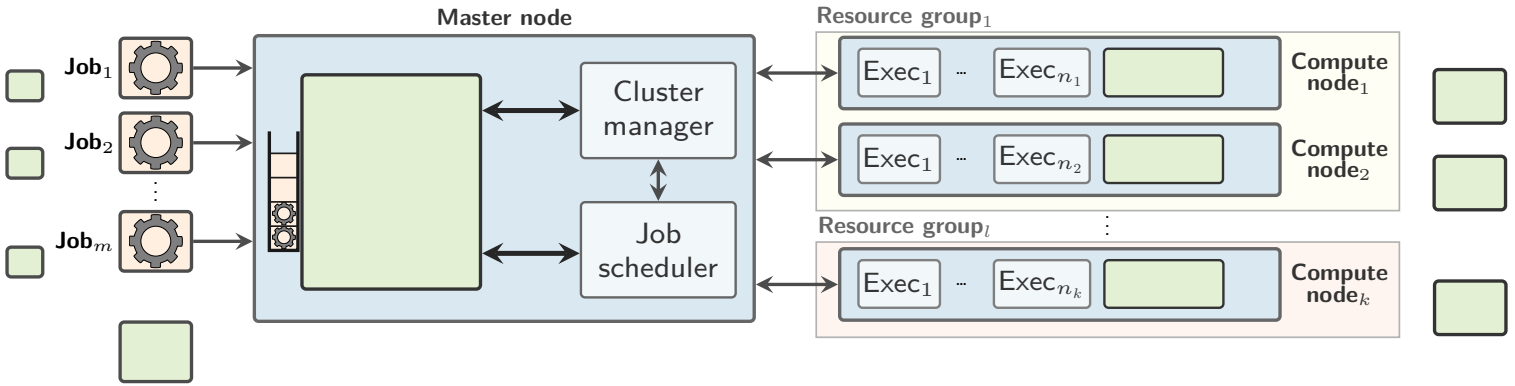
Albatross: Konzept und Implementierung

- generisches Systemmodell eines HPC-Clusters ohne Albatross

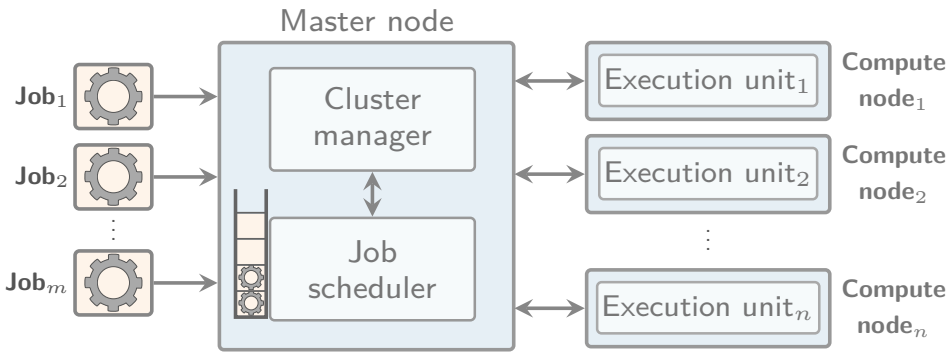


Albatross: Konzept und Implementierung

- erweitertes Systemmodell eines HPC-Clusters mit Albatross

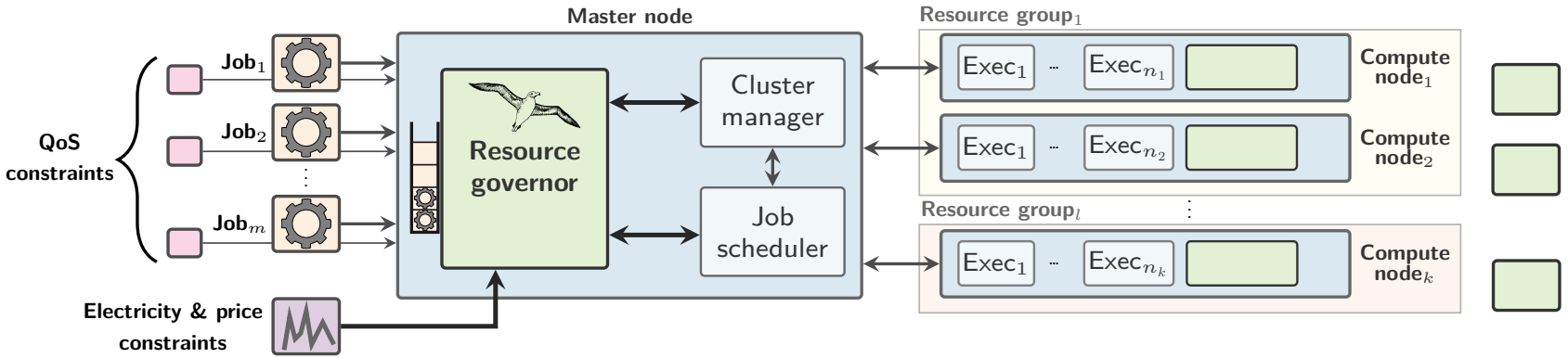


- generisches Systemmodell eines HPC-Clusters ohne Albatross



Albatross: Konzept und Implementierung

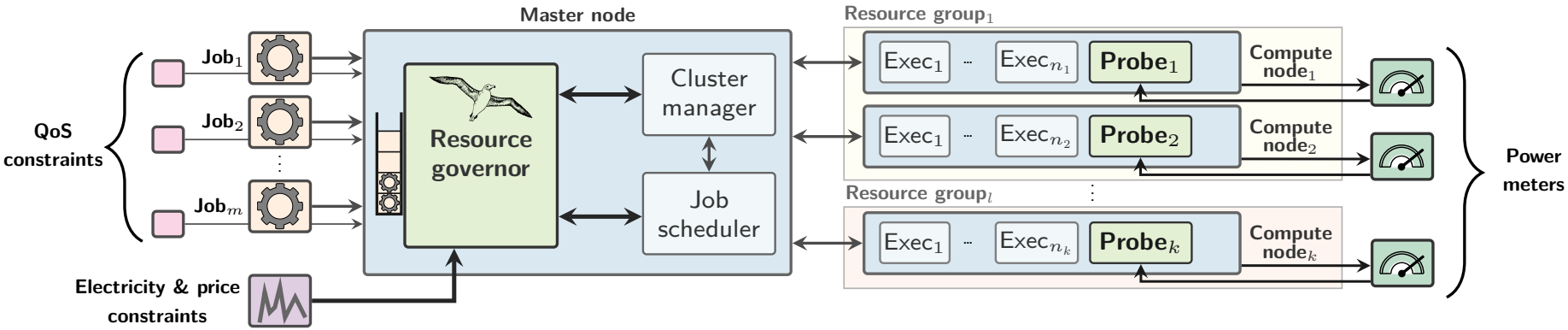
- erweitertes Systemmodell eines HPC-Clusters mit Albatross



- Eingabeschnittstellen für die Auftragsübertragung und -steuerung
  - Eingabeschnittstelle für Strompreisdaten
  - Echtzeitdaten gemäß der Bereitstellung durch den Netzbetreiber
  - detaillierte Auftragsbeschreibung, zusätzliche QoS-Beschränkungen

Albatross: Konzept und Implementierung

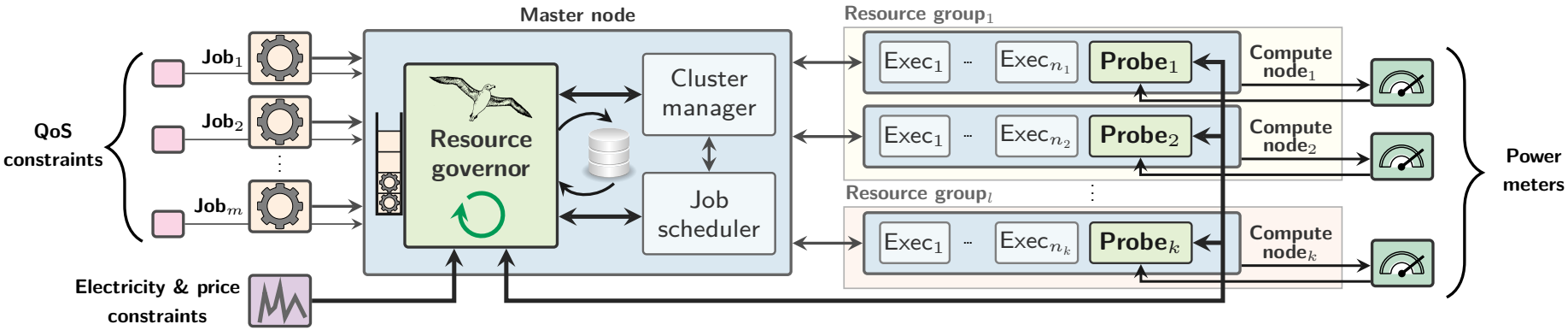
- erweitertes Systemmodell eines HPC-Clusters mit Albatross



- Energieüberwachungs- und Steuerungsschnittstellen
  - „Probes" bedienen Power-Management-Funktionen (z.B. Power-Capping)
  - Hardware-spezifische Messungen (z.B., RAPL, Messgeräte)
  - Bereitstellung von Basisdaten für die Feedbackschleife an den Resource-Governor

Albatross: Konzept und Implementierung

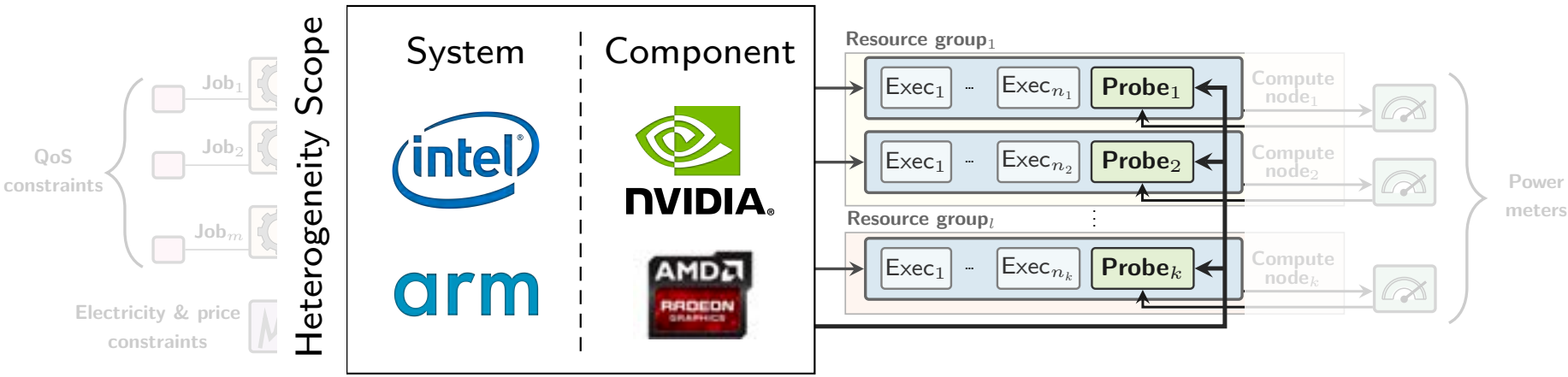
- erweitertes Systemmodell eines HPC-Clusters mit Albatross



- Laufzeitkontrolle des HPC-Systems durch den Resource-Governor
  - Auftragserteilung auf Basis von QoS-Daten und aktuellen Strompreisen
  - Kontrolle und Messung des Energiebedarfs für einzelne Arbeitslasten
  - Berücksichtigung von Heterogenitätsaspekten (System, Komponente und Konfiguration)

Albatross: Konzept und Implementierung

- erweitertes Systemmodell eines HPC-Clusters mit Albatross



- Laufzeitkontrolle des HPC-Systems durch den Resource-Governor
  - Auftragserteilung auf Basis von QoS-Daten und aktuellen Strompreisen
  - Kontrolle und Messung des Energiebedarfs für einzelne Arbeitslasten
  - Berücksichtigung von Heterogenitätsaspekten (System, Komponente und Konfiguration)

Albatross: Konzept und Implementierung

- Albatross basiert auf Linux und Slurm
- auftragsbezogene Zuordnungsstrategie

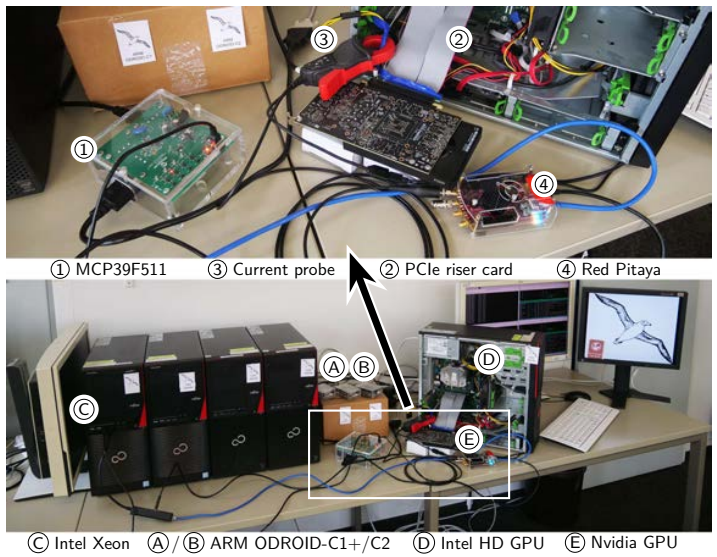
Feature	Moab	Tivoli	Univa	Slurm	
Job-Pinning	○	○	✓	✓	✓
CPU-Zuteilung	✓	✓	○	✓	✓
Quality-of-Service	✓	○	✓	✓	✓
Generische Ressourcen	✓	○	✓	✓	✓
Cluster-Status	✓	✓	✓	✓	✓
Heterogenität	✓	○	✓	○	✓
Energie-/Preisbewusstsein	○	○	○	○	✓

Tabelle: Feature-Vergleich von Albatross mit vergleichbaren Systemen.

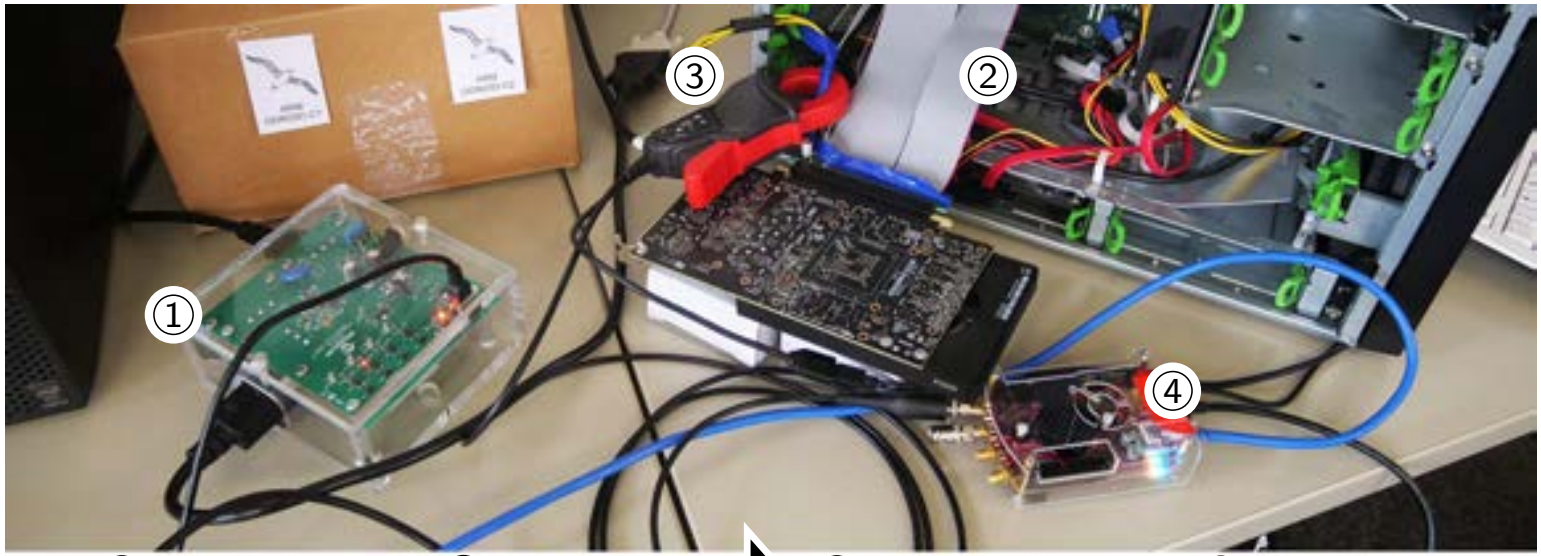
Evaluation: Überblick

- Evaluationsszenario
  - </> NAS Parallel Benchmarks (NPB) Suite, Albatross OS und Laufzeitsystem
  - heterogener Cluster mit Intel & ARM CPUs, Nvidia & Intel GPUs und unterschiedlichen Leistungsmessgeräten

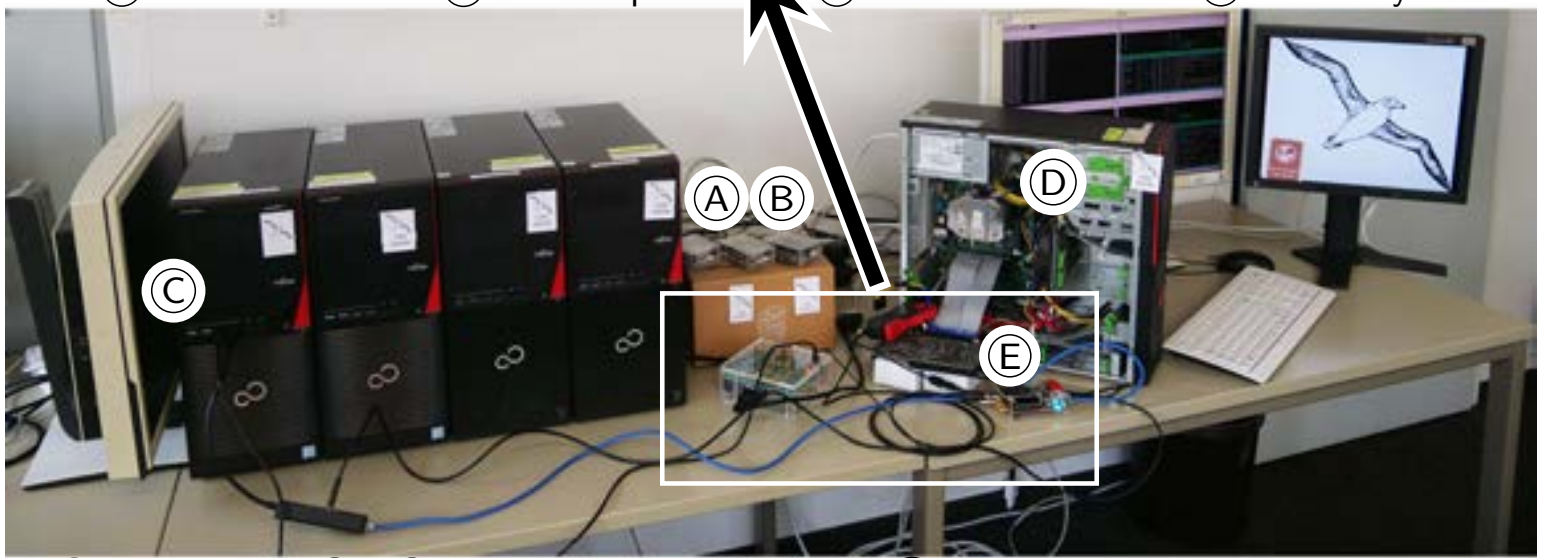
- Evaluationsexperimente und -ziele
  1. Auswirkungen der Systemkonfiguration auf Energiebedarf und Ausführungszeiten
  2. Einfluss der Heterogenität auf „Energy-Delay-Product" und Leistungsbedarf
  3. Kombination von Energie- und Preisbewusstsein zur Nutzung dynamischer Strompreise



Evaluation: Überblick



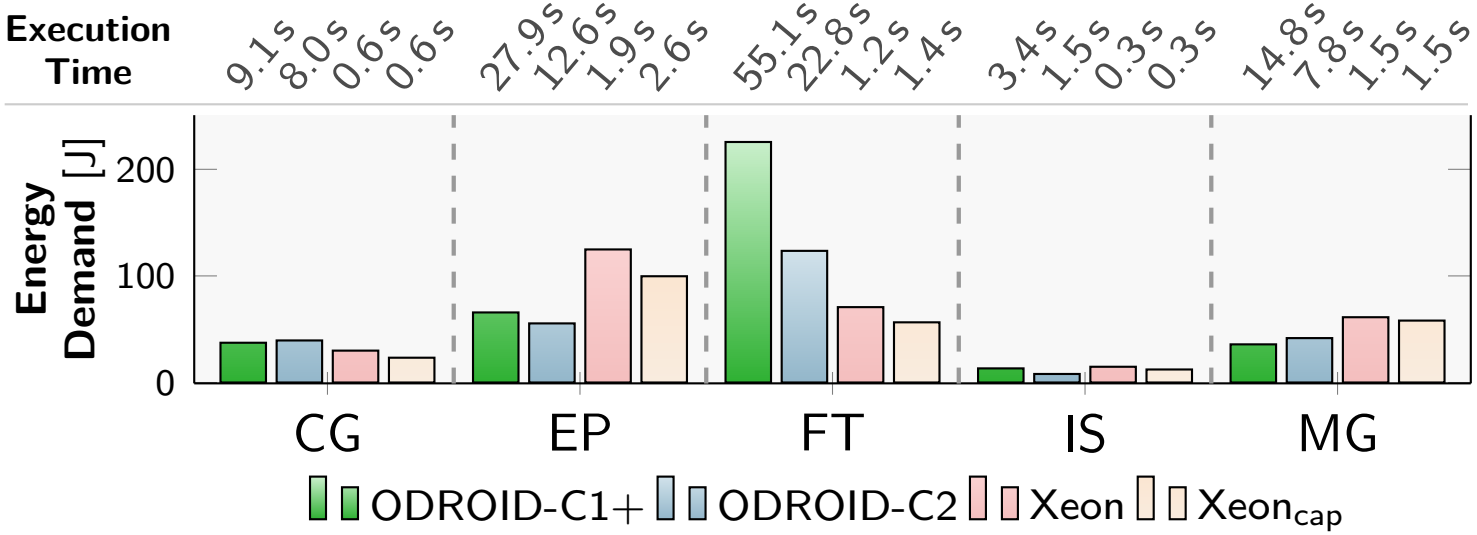
① MCP39F511 ③ Current probe ② PCIe riser card ④ Red Pitaya



Ⓒ Intel Xeon Ⓐ / Ⓑ ARM ODROID-C1+/C2 Ⓓ Intel HD GPU Ⓔ Nvidia GPU

Experiment 1: Energiebedarf und Ausführungszeiten

- Auswirkungen der Heterogenität (System, Konfiguration)



- Xeon und Xeon$_{cap}$ dominieren mit Performance
- Energiebedarf für ARM-Plattformen ist in einigen Fällen geringer
- nur gelegentlicher Zusammenhang zwischen Prozessenergiebedarf und Ausführungszeit

Experiment 1: Energiebedarf und Ausführungszeiten

- Auswirkungen der Heterogenität (System, Konfiguration)

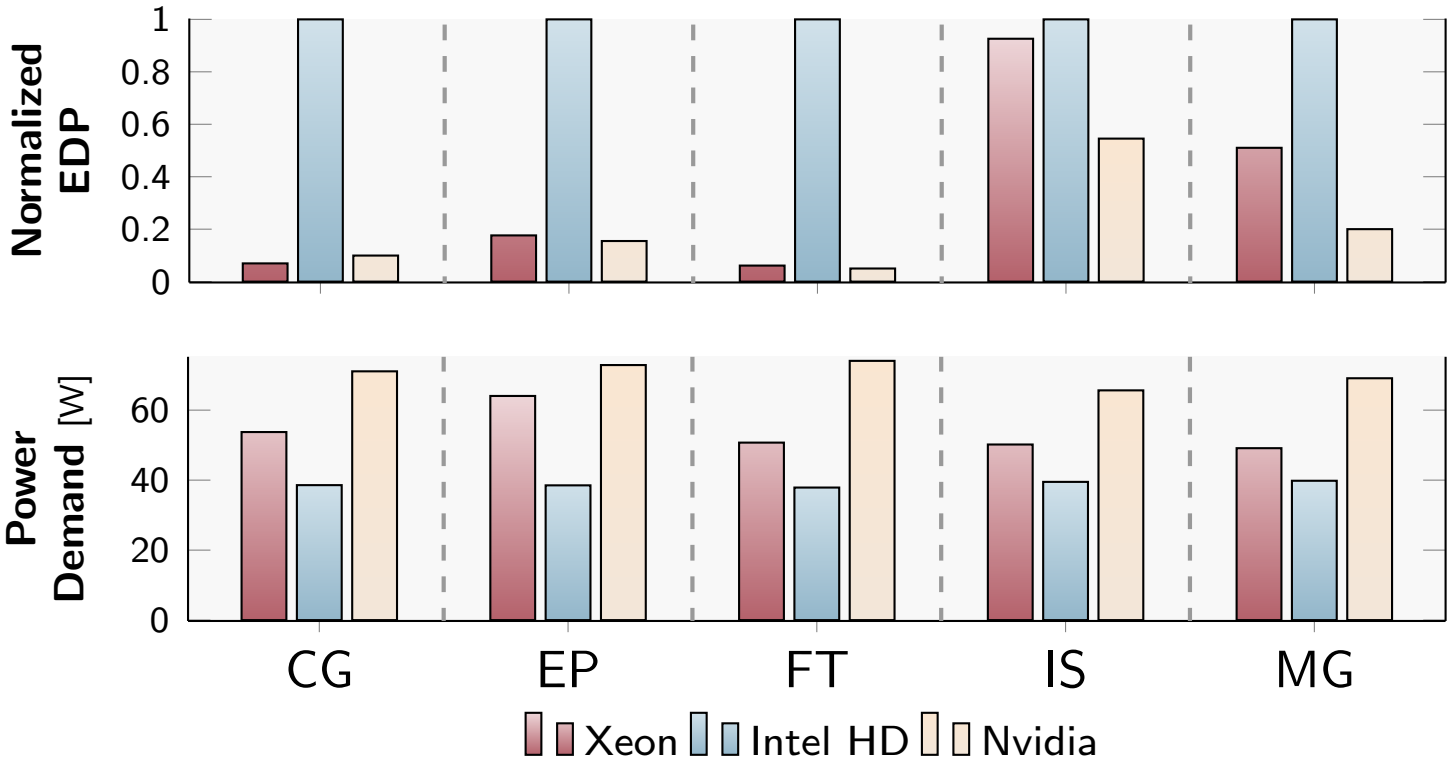
Erkenntnisse von Experiment 1

- Modus „Rennen" reduziert nicht immer den Energiebedarf
- Kenntnisse über die Auswirkungen der ausführenden Systemknoten sind für den Betrieb unerlässlich und erfordern Messungen

- Energiebedarf für ARM-Plattformen ist in einigen Fällen geringer
- nur gelegentlicher Zusammenhang zwischen Prozessenergiebedarf und Ausführungszeit

Experiment 2: Energy-Delay-Product und Leistungsbedarf

- Auswirkungen der Heterogenität (Komponenten)



- Nvidia GPU erreicht meistens das beste EDP-Ergebnis
- Intel HD erreicht die schlechtesten EDP-Ergebnisse, bietet aber den niedrigsten mittleren Leistungsbedarf

Experiment 2: Energy-Delay-Product und Leistungsbedarf

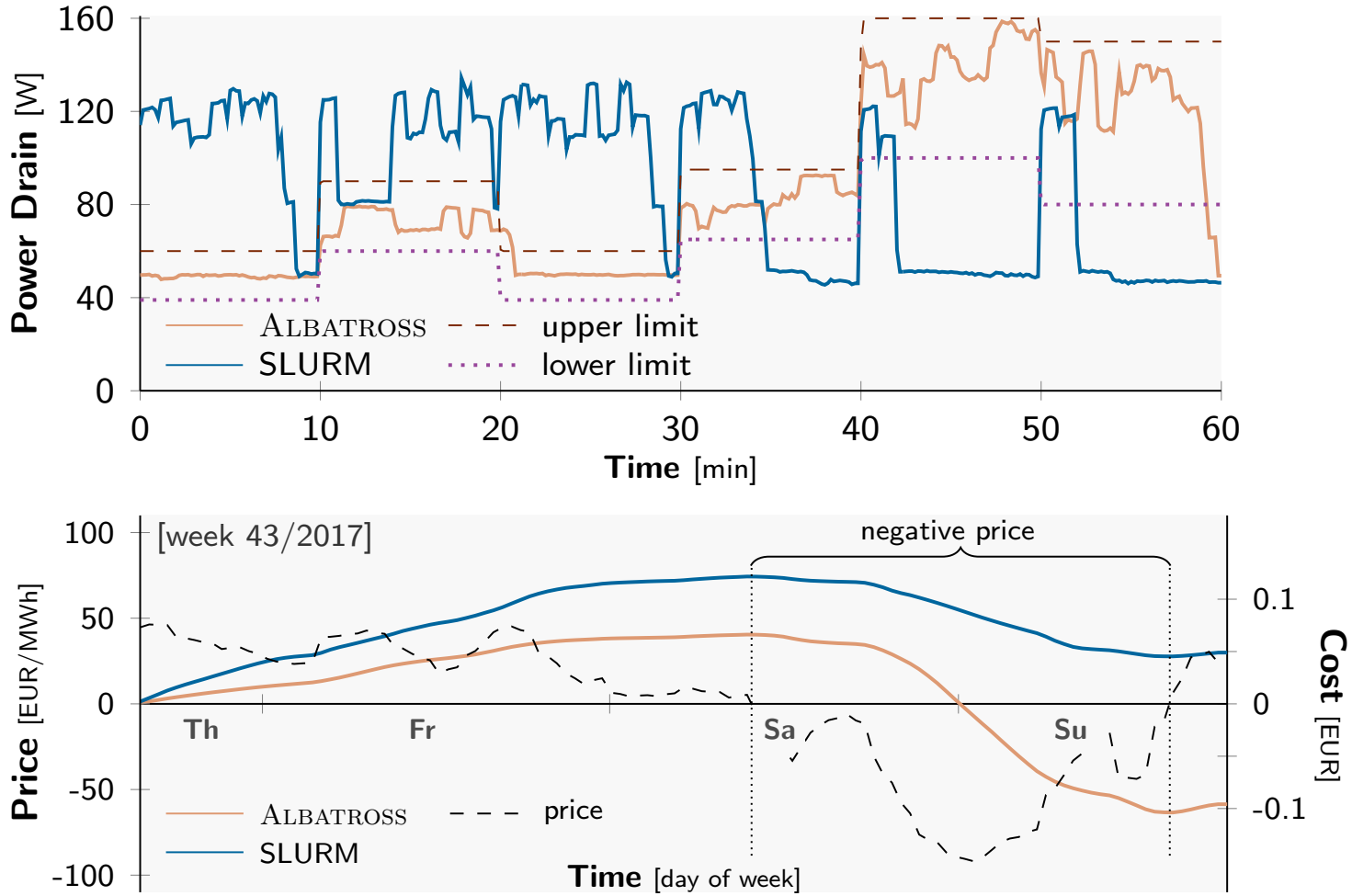
- Auswirkungen der Heterogenität (Komponenten)

Erkenntnisse von Experiment 2

- Sowohl EDP als auch der mittlere Leistungsbedarf muss berücksichtigt werden → Operationsmodi
- Um den Stromverbrauch des Clusters zu regulieren, wägt Albatross durchschnittlichen Leistungsbedarf und EDP ab

- Nvidia GPU erreicht meistens das beste EDP-Ergebnis
- Intel HD erreicht die schlechtesten EDP-Ergebnisse, bietet aber den niedrigsten mittleren Leistungsbedarf

Experiment 3: Kombination von Energie- und Preisbewusstsein

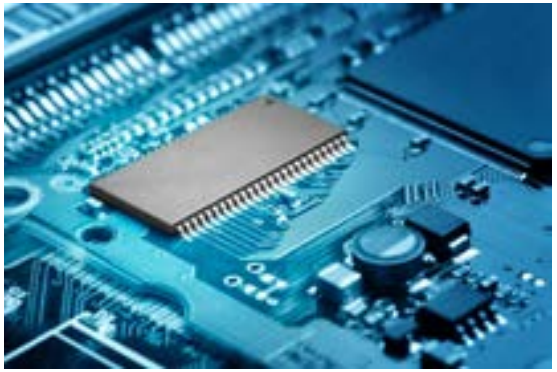


Agenda

1. Motivation ✔
2. Energieeffiziente Systemsoftware ✔
3. Energiebewusste Programmierung ✔
4. Albatross OS ✔
5. Ausblick: Aktuelle Forschung
6. Zusammenfassung und Schlussfolgerung

Ausblick: Aktuelle Forschung

- Ressourceneffiziente eingebettete Systeme
  - Cyber-Physical Networking: Latenzreduktion und Energieeffizienz
    - → Assoziierte Arbeiten mit DFG SPP 1914
    - → Industrie-Kooperation mit Nokia Networks (optische Netze, 5G)
  - Maßgeschneiderte Systemsoftware mit niedrigem Ressourcenbedarf



A. Ziegler, T. Hönig et al.:
Honey, I Shrunk the ELFs: Lightweight Binary Tailoring of Shared Libraries.
ACM SIGBED International Conference on Embedded Software (EMSOFT'19)
Note: to appear.

Ausblick: Aktuelle Forschung

- Systemsoftware für Many-Core-Systeme
  - Betriebssystemkonzepte für massiv parallele Many-Core-Systeme
    - → Principal Investigator (PI) SFB/TR 89 „Invasive Computing"
    - → Power-Aware Critical Sections (DFG)
  - Jitterarme Laufzeitumgebungen zum Erreichen optimaler Performance

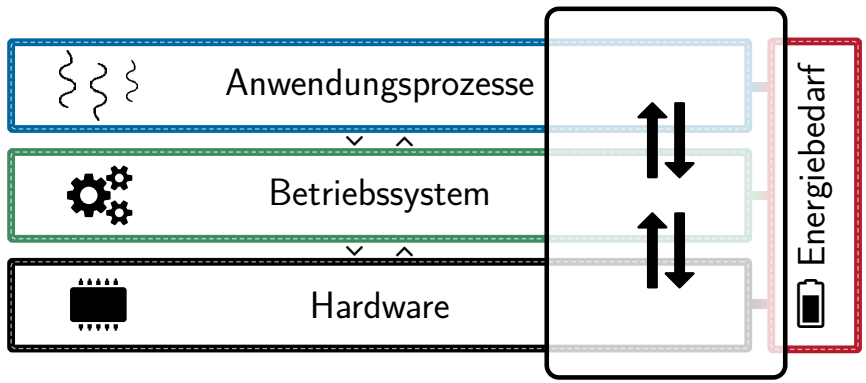
A. Ziegler, T. Hönig et al.:
Honey, I Shrunk the ELFs: Lightweight Binary Tailoring of Shared Libraries.
ACM SIGBED International Conference on Embedded Software (EMSOFT'19)
Note: to appear.

S. Maier, T. Hönig et al.:
Asynchronous Abstract Machines:
Anti-noise System Software for Many-core Processors.
ACM International Workshop on
Runtime and Operating Systems for Supercomputers (ROSS'19)

Zusammenfassung und Schlussfolgerung

- Systemsoftware für energieeffiziente Systeme
  - Software ist zentrale Einfluss- und Steuergröße für den Energiebedarf
  - Optimale Abbildung von Soft- auf Hardware ist essentiell: Weniger ist oft mehr!

- Schichtenübergreifende Analyse
  - Erschließung unausgeschöpfter Einsparpotentiale
  - Werkzeugkette bietet aktive Programmierunterstützung



- Phasenübergreifender Ansatz
  - Statische Verbesserung vor und
  - erweiterte dynamische Adaption zur Laufzeit