

Realzeit mit Multiprozessor- Systemen auf PC-Basis

Lehrstuhl für Realzeit-Computersysteme
Technische Universität München

Alexander von Bülow, Jürgen Stohr

- ❑ Das Projekt RECOMS
- ❑ Ausführungszeiten auf IA-32 Prozessoren
- ❑ Optimale Speicherorganisation
- ❑ Einfluss des PCI-Busses
- ❑ Zusammenfassung / Ausblick

RECOMS

REaltime with Commercial Off-the-shelf Multiprocessor Systems

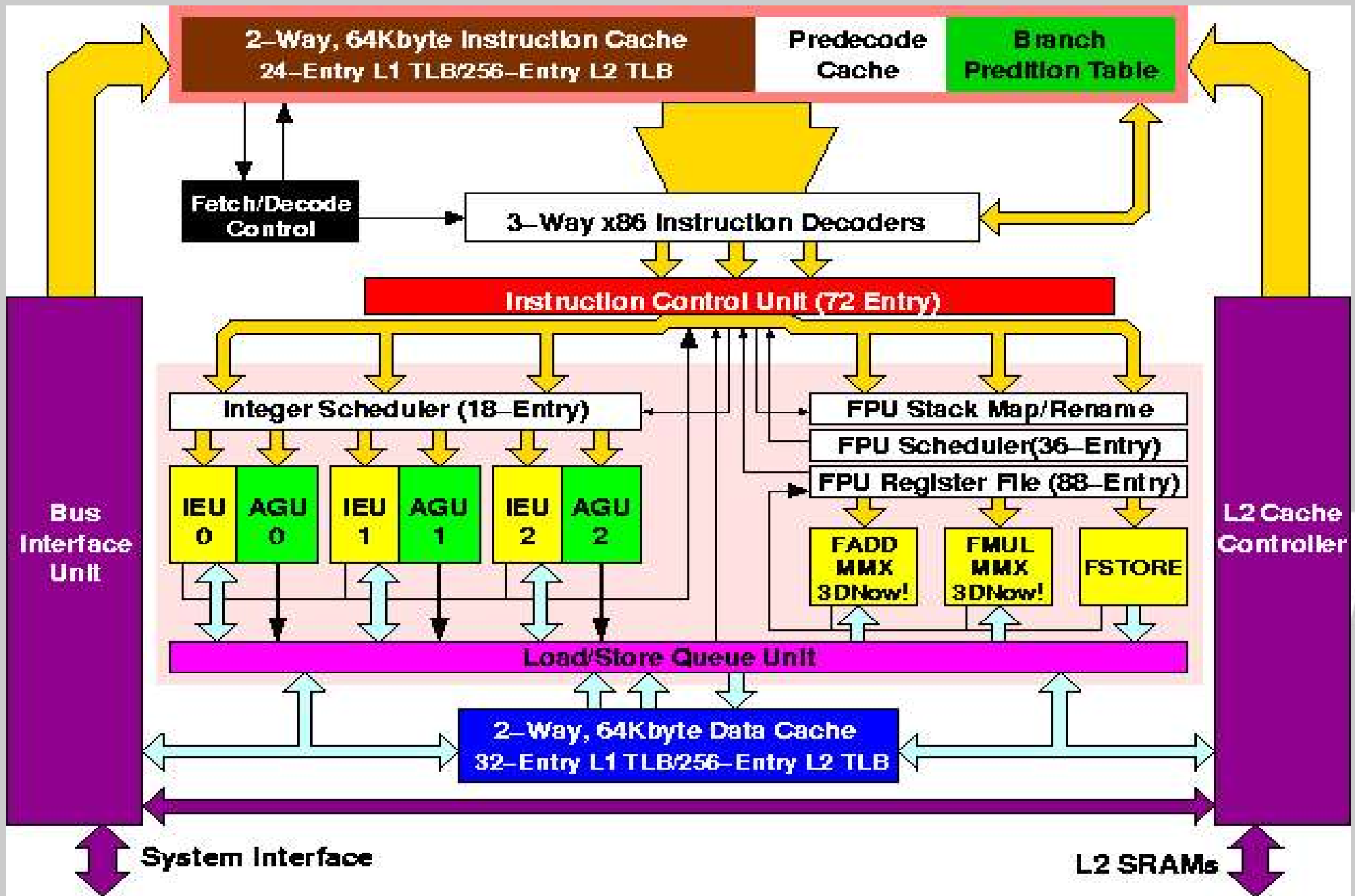
- ❑ Commercial Off-the-shelf
 - Günstige Hardwarepreise
 - Hohe Rechenleistung
 - Schnelle Weiterentwicklung mit Rückwärtskompatibilität
- ❑ Realtime
 - RTAI-Linux – OpenSource UNIX-Derivat mit Realzeiterweiterung
 - Große nutzbare Software- und Treiberbasis
 - Stetige Weiterentwicklung
- ❑ Multiprocessor Systems
 - Klare Trennung zwischen Standard- und Realzeittasks
 - Optimale Nutzung der CPU-Ressourcen für Realzeittasks möglich
 - Hohe Flexibilität beim Softwaredesign des Realzeitsystems

- ❑ Abhängig von der Prozessorarchitektur
 - ❑ Caches
 - ❑ Pipelines
 - ❑ Branch Prediction

- ❑ Abhängig von externer Hardware
 - ❑ Verbindung CPU-Chipsatz-Hauptspeicher
 - ❑ Kommunikation über Bussysteme (z.B. PCI)
 - ❑ Verhalten der Hardware bei Zugriffen (z.B. Festplatte, Netzwerkkarte)

- ❑ Zur Bestimmung der WCETs wird aufgrund der hohen Komplexität ein messtechnischer Ansatz verfolgt

- ❑ Vor jeder Messung muss daher der Worst Case für den Code hergestellt werden können



- ❑ Caches
 - ❑ Hierarchisch aufgebaut (L1,L2, zunehmend auch L3)
 - ❑ L1 Cache geteilt in Instruktionen- (oder Trace-) und Datencache
 - ❑ Organisationsmerkmale sind Setgröße, Assoziativität und Cachelinegröße
 - ❑ Sehr wichtig für die Bestimmung von Ausführungszeiten

- ❑ Pipeline
 - ❑ Schrittweise Befehlsabarbeitung
 - ❑ Nutzung von ILP (out-of-order execution -> Tomasulo Algorithmus)

- ❑ Branch Prediction
 - ❑ Verhindert Pipeline-Stalls bei bedingten Programmverzweigungen
 - ❑ Wichtig für Prozessoren mit langer Pipeline (z.B. Pentium 4)

- ❑ Belegung der Caches korrespondiert direkt mit der Hauptspeicherbelegung
- ❑ Jede Verdrängung aus dem Cache kostet relativ viel Zeit, abhängig davon, ob Daten zurückgeschrieben werden müssen oder nicht
- ❑ Auch die Anbindung des Speichers (Host Bridge) spielt eine große Rolle, ebenso die PCI-Bus Aktivität (Blockierungen)
- ❑ Resultierend daraus ist jeder Verdrängungsvorgang mit einem stark schwankenden Jitter behaftet

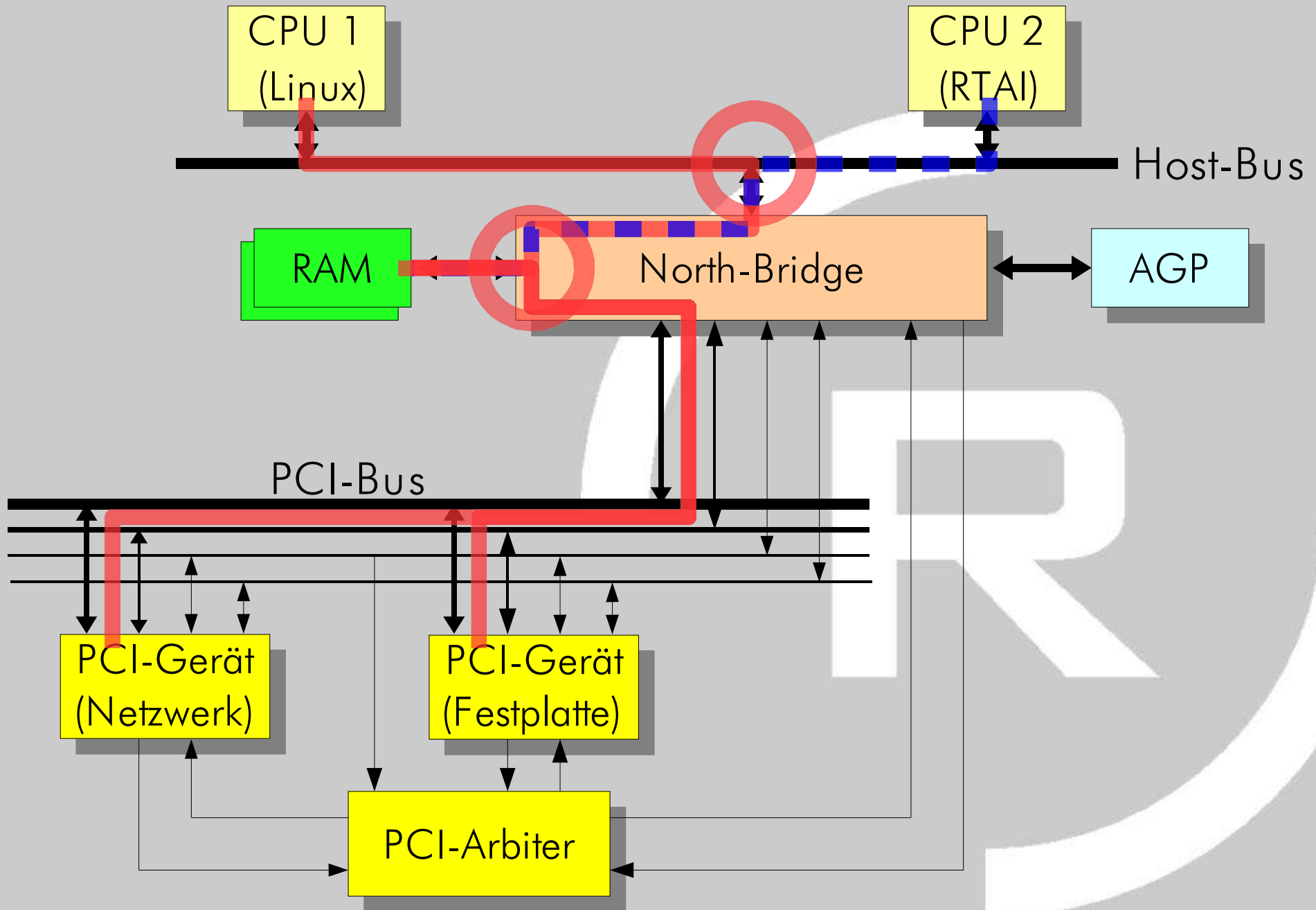
- ❑ Hauptziel ist, so wenig Verdrängungen wie möglich zu haben
- ❑ Kritische Code-/Daten Objekte sollen im Cache fixiert werden können, d.h. sie werden niemals verdrängt
- ❑ Die Caches mehrerer CPUs sollen genutzt werden können
- ❑ Vermeidung von Cache-Snooping Effekten
- ❑ Die Zahl der TLB-Misses soll minimiert werden
- ❑ Durch genaue Kenntnis der verbliebenen Verdrängungen bessere WCET Abschätzung möglich

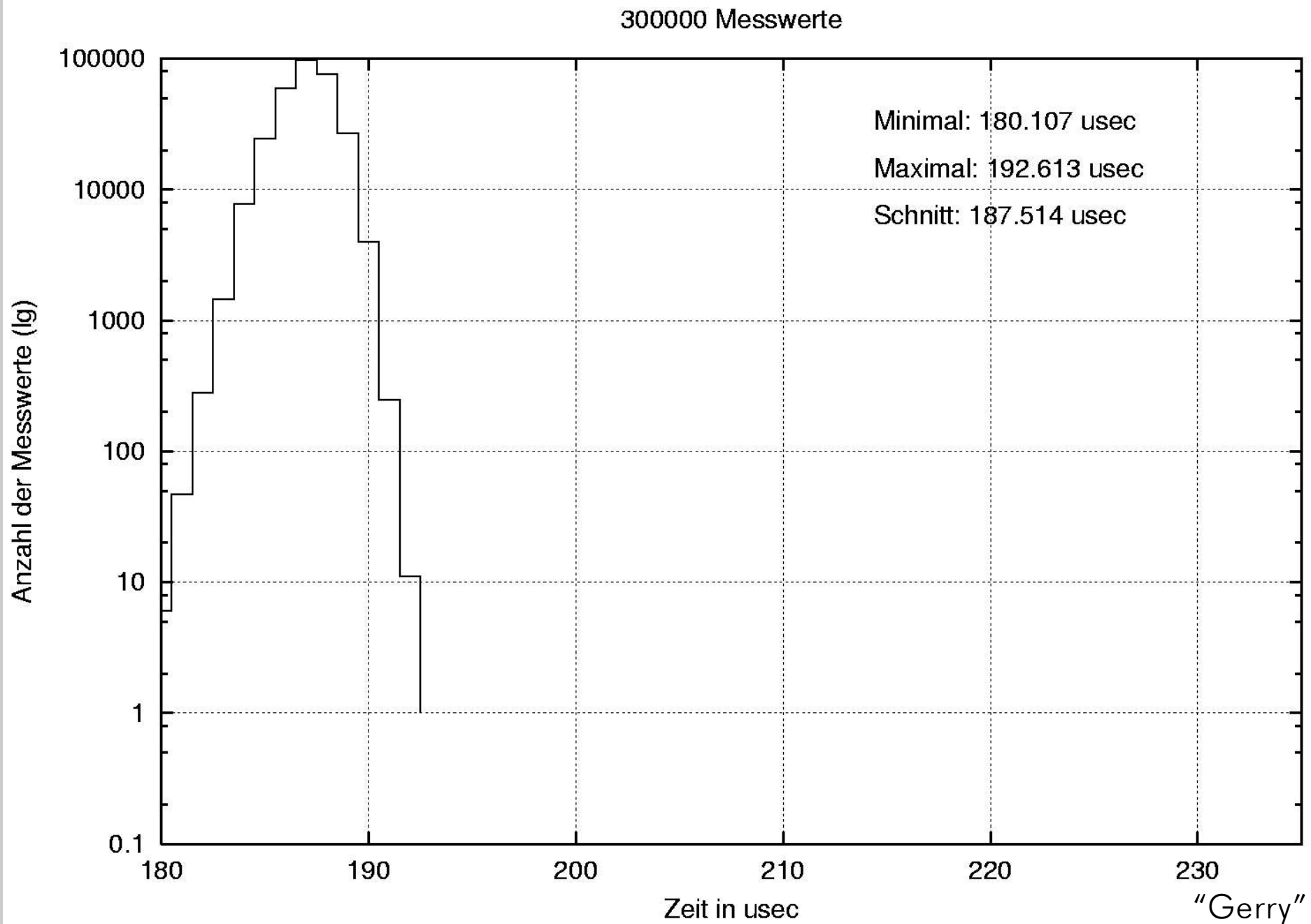
- ❑ Code wird jeweils in kompletten Funktionen angeordnet
- ❑ Alle Daten bzw. Datenstrukturen werden getrennt betrachtet
- ❑ Jede Realzeittask besitzt einen Stack, welcher jeweils extra behandelt wird
- ❑ Jedes Objekt (Code, Daten oder Stack) kann eines von drei Attributen besitzen:
 - ❑ FIXED: Darf nicht verdrängt werden
 - ❑ READ_WRITE: Objekte können sich ändern
 - ❑ READ_ONLY: Objekte können sich nicht ändern
- ❑ Pro Set immer nur Cachelines mit gleichen Attributen

- ❑ Es werden *alle* Funktionen/Daten des Realzeitsystems betrachtet, also RT-Applikationen, RTOS, ISRs
- ❑ Die Größe der Code-/Daten Objekte in Bytes ermitteln, Attribute vergeben:
 - ❑ Code ist immer READ_ONLY
 - ❑ Stack ist immer READ_WRITE
 - ❑ Für die restliche Datenobjekte müssen die Attribute entsprechend ihrer Sections ermittelt werden
 - ❑ Ggf. einzelne Objekte "FIXED" machen (z.B. kleine ISRs)
 - ❑ "FIXED" im L1-Cache oder L1- und L2-Cache
- ❑ Um nicht verschiebbare Funktionen (z.B. Linux-Kernelfunktionen) muss "herumoptimiert" werden

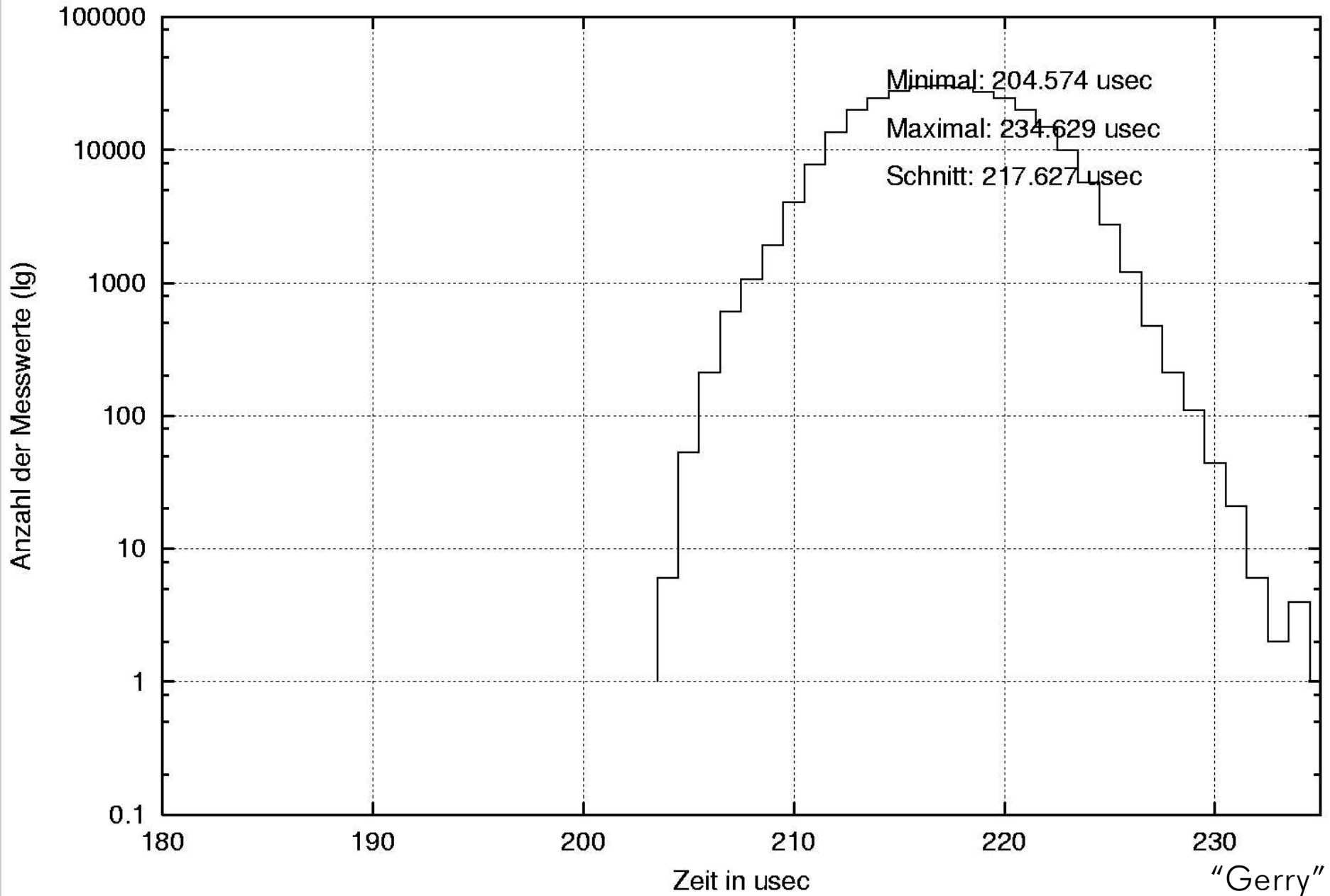
- ❑ Standardbus in heutigen PC-Systemen
- ❑ Hoher Datendurchsatz
 - ❑ 32/64 Bit, 33/66 Mhz
 - ❑ Burst-Transfers
- ❑ Prozessorunabhängigkeit
 - ❑ Plattformunabhängigkeit
 - ❑ Rückwärts- und Vorwärtskompatibilität
 - ❑ Multi-Master-Fähigkeit
- ❑ Standardisierte Konfigurationsschnittstelle
 - ❑ Definierte Informations- und Konfigurationsregister
 - ❑ Konfliktlose Ressourcenvergabe

- ❑ PCI-Geräte können Daten direkt in den Hauptspeicher transportieren (DMA)
- ❑ Ein PCI-Zyklus besteht aus einer Adress- und mehrerer Datenphasen
- ❑ Während eines solchen Zykluses kann die CPU nicht auf den Hauptspeicher zugreifen -> Blockierung!
- ❑ PCI-Bus muss vor jedem Zugriff arbitriert werden -> Wartezeiten beim Zugriff!
- ❑ Mögliche Lösung: Temporäre Abschaltung einzelner nicht benötigter Geräte (Master-Enable Bit löschen)





300000 Messwerte



"Gerry"

- ❑ Für die Bestimmung von Laufzeiten müssen sowohl die Prozessorarchitektur als auch die Anbindung externer Hardware berücksichtigt werden
- ❑ Durch gezielte Anordnung von Code und Daten im Hauptspeicher können die Caches optimal genutzt werden
- ❑ Multiprozessorsysteme bieten hier klare Vorteile gegenüber Uniprozessorsystemen
- ❑ Der Einfluss externer Bussysteme (PCI-Bus) muss berücksichtigt werden
- ❑ Für eine zuverlässige WCET-Abschätzung muss die Worst Case Situation erzwingbar sein

Vielen Dank für Ihre Aufmerksamkeit!

Für Fragen stehe ich gerne zur Verfügung!