

EMES: Eigenschaften mobiler und eingebetteter Systeme

Ausführungsvorhersage

Dipl. Inf. Jan Richling
Wintersemester 2005/2006



1. Einleitung
2. Theoretische Vorbetrachtungen
3. Woher kommt der Pessimismus?
4. Technische Grundlagen der Referenzarchitektur
5. Ein Beispielalgorithmus
6. Messungen an Prozessoren

- Schedulingverfahren basieren auf Kenntnis der Ausführungszeit einer Task
- Zeitliche Garantien können nur gegeben werden, wenn man Ausführungszeiten kennt

Problem: Wie mißt man eine Ausführungszeit?

- Stoppuhr? Logic-Analyzer?

Genügt das?

Worst Case Execution Time I

- Kenntnis **einer** Ausführungszeit genügt nicht, denn Laufzeit eines Programmes kann von den Eingaben abhängen
- Interessant ist die längstmögliche Ausführungszeit (WCET)
- WCET hängt ab von
 - Programmcode
 - Compiler, benutzte Bibliotheken
 - Pfade durch den Programmcode (abhängig von Eingaben)
 - der benutzten CPU
 - Parametern der Umgebung der CPU (Speicher, Caches)
 - sowie Wechselwirkungen zwischen diesen Parametern

Worst Case Execution Time II

Es gibt zwei generelle Probleme:

- Welches ist der längste Pfad durch ein Programm?
Unterproblem: Wie bewertet man einen Pfad?
→ **Highlevel Analyse**
- Wie berechnet man zu einem gegebenen Programmpfad die Ausführungszeit?
→ **Lowlevel Analyse**

Bestimmen der WCET einer Task

- Analysewerkzeug erstellt einen *Kontrollflussgraphen*
→ Kontrollflussgraph besteht aus *Basic Blocks*
- Ausführungszeit aller Basic Blocks wird ermittelt (Lowlevel-Analyse!)
- Berechnung des längsten Pfades
- Gegebenenfalls weitere Optimierungen (ist längster Pfad *feasible*?)

Basic Blocks sind Blöcke von Maschinenbefehlen, die genau einen Ein- und einen Austrittspunkt haben.

Highlevel-Analyse mit dem Timing-Schema

$$W(X) = \text{Zahl, wenn } X \text{ ein BB ist}$$

$$W(X; Y) = W(X) + W(Y)$$

$$W(\text{if } Z \text{ then } X \text{ else } Y) = W(Z) + \max \{W(X), W(Y)\}$$

$$W(\text{for } Z \text{ loop } X) = (n + 1) \cdot W(Z) + n \cdot W(X)$$

- Schleifendurchläufen sind durch n beschränkt
- Ausführungszeit eines BBs aus der Lowlevel-Analyse
- wird rekursiv für jedes Blatt des Syntaxbaumes ausgeführt
- einfachstes Timing-Schema

Lowlevel-Analyse – Voraussetzungen

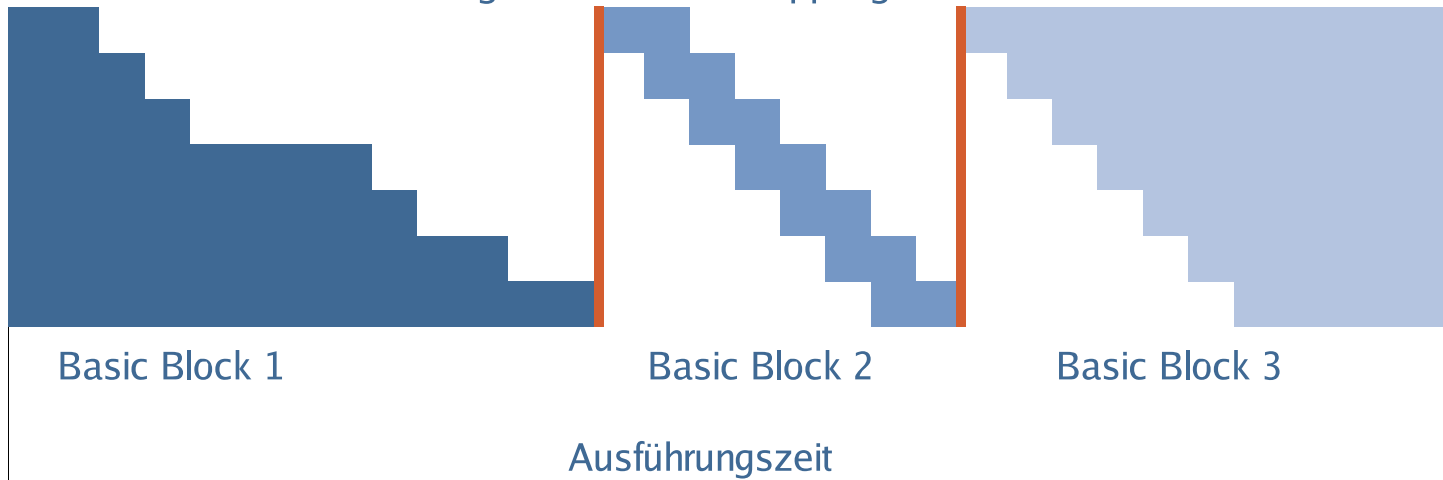
- Genaue Reihenfolge der Instruktionen muss bekannt sein, d.h.
 - keine (unvorhergesehenen) Interrupts
 - keine Exceptions
- Prozessor muss spezifikationsgetreu betrieben werden
- Prozessorzustand muss konstant bleiben
- DMA-Zugriffe sind verboten
- Detaillierte Beschreibung des gewünschten Prozessors/Systems muss vorliegen

Woher kommt der Pessimismus I

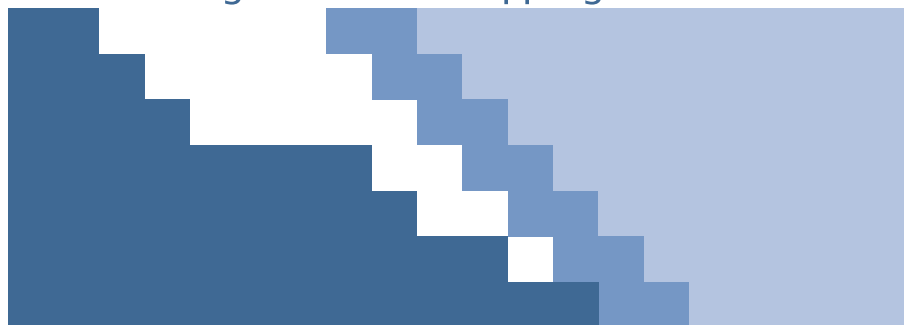
- **Vereinfachende Annahmen** zu
 - Synchronisationszeiten
 - Anzahl von Schleifendurchläufen
 - Zugriffszeiten
 - Cache/TLB/BHT
 - Prozessoreigenheiten
 - ...
- Unkenntnis des Systemzustandes
- Schleifen
- Haben wir auch wirklich den Worst-Case-Pfad gefunden?
- Ungenaue Lowlevel-Analyse (Überlappung)

Beispiel: Überlappung

Gesamte Ausführungszeit ohne Überlappung der Bbs



Ausführungszeit mit Überlappung



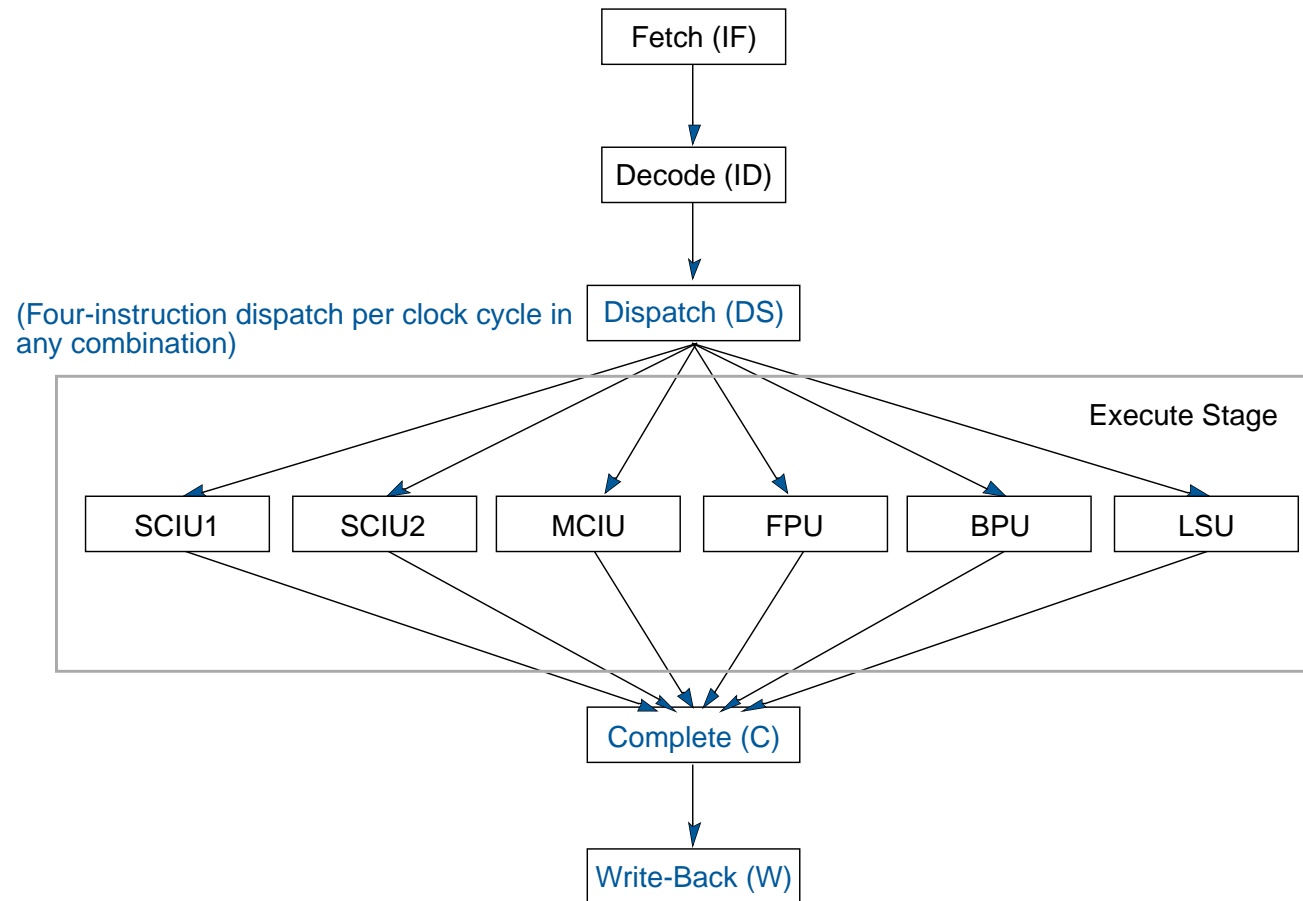
Wichtige Systemparameter

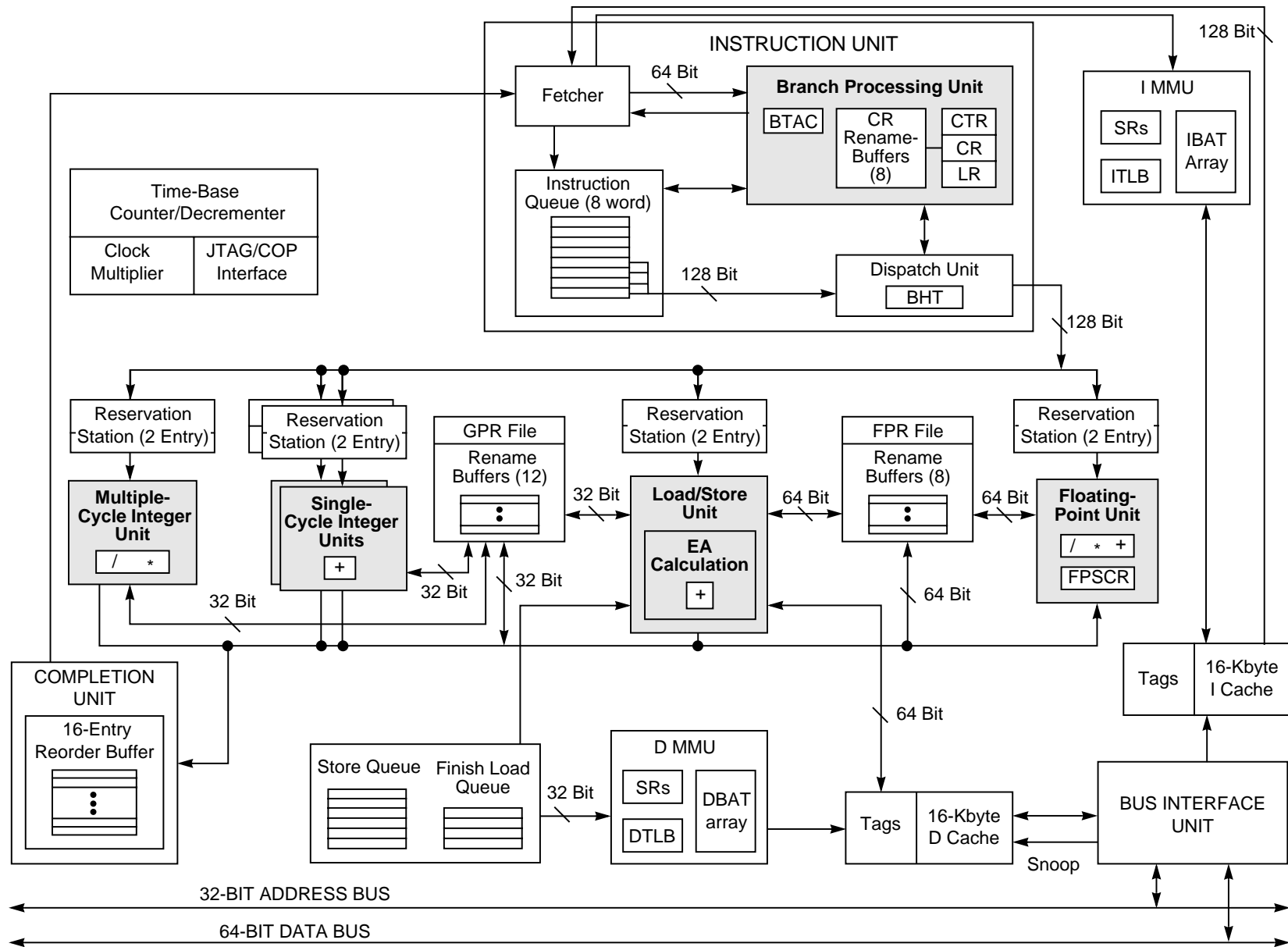
- Pipeline?
 - Funktionsweise der einzelnen Stufen
 - Länge
 - Organisation
- Funktionseinheiten
- Speicher
 - Aufbau
 - Struktur
 - Cachestruktur
 - Geschwindigkeit
 - Busorganisation
 - Burstmodi
 - Taktteiler Bus/CPU

Referenzarchitektur PowerPC 604 – Warum?

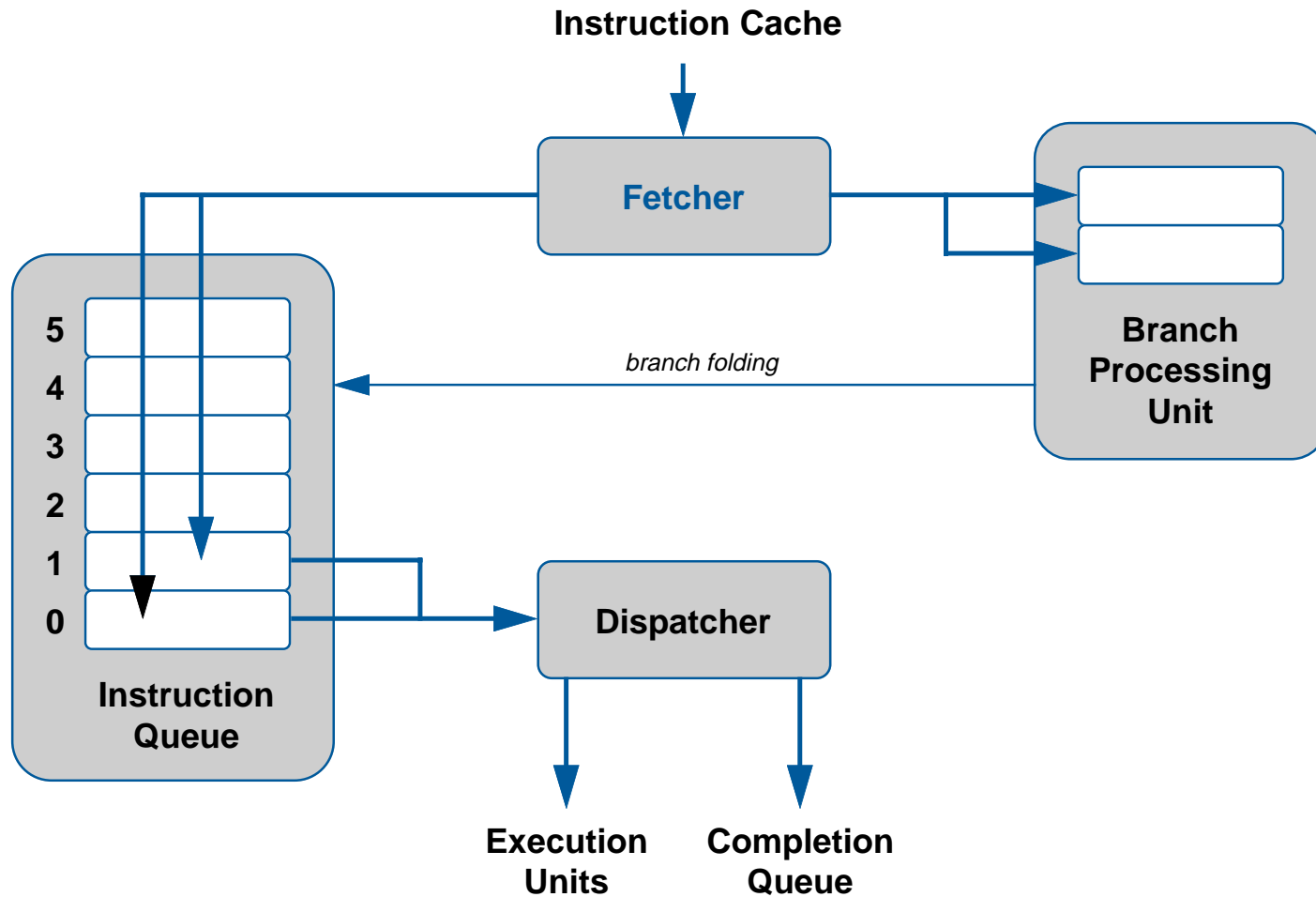
- RISC - Architektur
- Mitglied einer ganzen Familie von Prozessoren
- Komplexität ist noch handhabbar
- Existiert als Hostprozessor und als eingebettetes Gerät
- Preiswert
- Implementiert „Performance Enhancements“
 - Pipeline
 - Große Caches
 - Sprungvorhersage
 - Spekulative Ausführung
 - Out-of-Order-Execution
 - ...

Referenzarchitektur PowerPC 604 – Pipeline





Referenzarchitektur PowerPC 604 – Branch-Unit



Die Datenstruktur

IF	1	2	3	4								
DEC		1	2	3	4							
DIS			1	2	3	4						
EX				1	2	3	4	4	4			
WB					1	2	3			4		
CPL						1	2	3			4	

Die Datenstruktur II

Reservation Table

- Erlaubt Behandlung von Pipelines
- Aufbau repräsentiert Merkmale der Architektur
- Zusammensetzen mehrerer Tabellen ist durch Architektur eingeschränkt
- Einfaches Ablesen der Laufzeit (Spaltenanzahl)
- Intuitives Konzept und leicht graphisch darstellbar

Angepasste Reservation Table - Eine Spalte

				Instruction Fetch I
				Branch Prediction Unit (BPU)
				Instruction Fetch II
				Decode
				Dispatch
				Single Cycle Integer Unit (SCIU)
				Multiple Cycle Integer Unit (MCIU)
				Load Store Unit (LSU)
				Floating Point Unit (FPU)
				Complete
				Write Back
				Registers {R} {W}

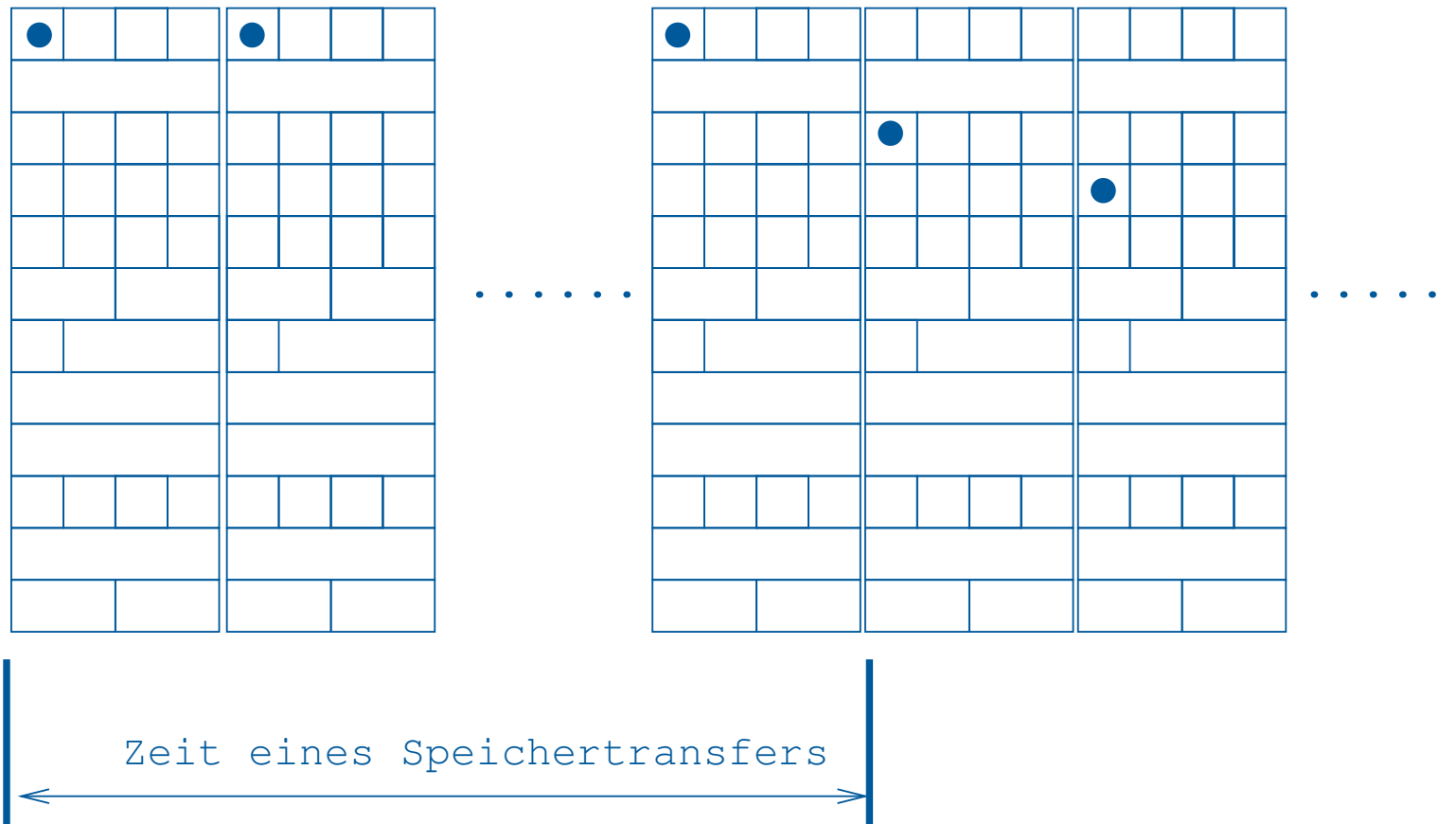
Repräsentation eines OR - Befehls

4: or 27, 11, 13											
IF_0	4										
BPU											
IF_1		4									
DEC				4							
DISP						4					
REG						11,13	27	11,13	27		
SCIU									4		
MCIU											
LSU											
FPU											
COMP											4
WB											4

Repräsentation eines Verzweigungsbefehls

C: b loop															
IF_0				C											
BPU					C										
IF_1															
DEC															
DISP									C						
REG															
SCIU															
MCIU															
LSU															
FPU															
COMP														C	
WB														C	

Darstellung eines Cachesmiss



1. Erstelle zu jeder Instruktion eine Tabellenrepräsentation
2. Verbinde die Tabellen unter Berücksichtigung architektureller Gegebenheiten
3. Lese die Anzahl der konsumieren Takte ab

Was noch fehlt

- Die Caches und Puffer innerhalb der CPU müssen vom Algorithmus verwaltet werden:
 - Daten-/Instruktionscache, TLB
 - Statusregister
 - BTAC, BHT
 - Reorder Buffer
 - Store Queue, Finish Load Queue

Messungen am Prozessor

- PowerPC 604 hat Performance Monitor Register
- Vielfältige Monitoringmöglichkeiten
 - konsumierte CPU-Takte
 - Cachemisses
 - Statistiken zu Instruktionen
 - fehlerhaft vorhergesagte Sprünge
 - Speicherzugriffsverzögerung

aber: PMC laufen nur im Supervisor Mode

- Tiefgreifende Modifikationen (Interrupts / Cache ausschalten) ebenfalls nur im Supervisor Mode möglich → Programmierung eines Kerneltreibers



Expandierende For – Schleife

- n mal eine Zahl um eins erhöhen
- mehrmalige Ausführung
- Veränderung des Verhältnisses von Schleifenlänge und Sprunghäufigkeit
- Halbierung der Schleifenlänge bei Verdopplung der Sprunghäufigkeit
- Ausführung mit und ohne Cache
- Ausführung mit Registervariablen (wenige Speichertransfers, kurzer Code)
- Ausführung mit Variablen im Hauptspeicher (viele Speichertransfers, längerer Code)

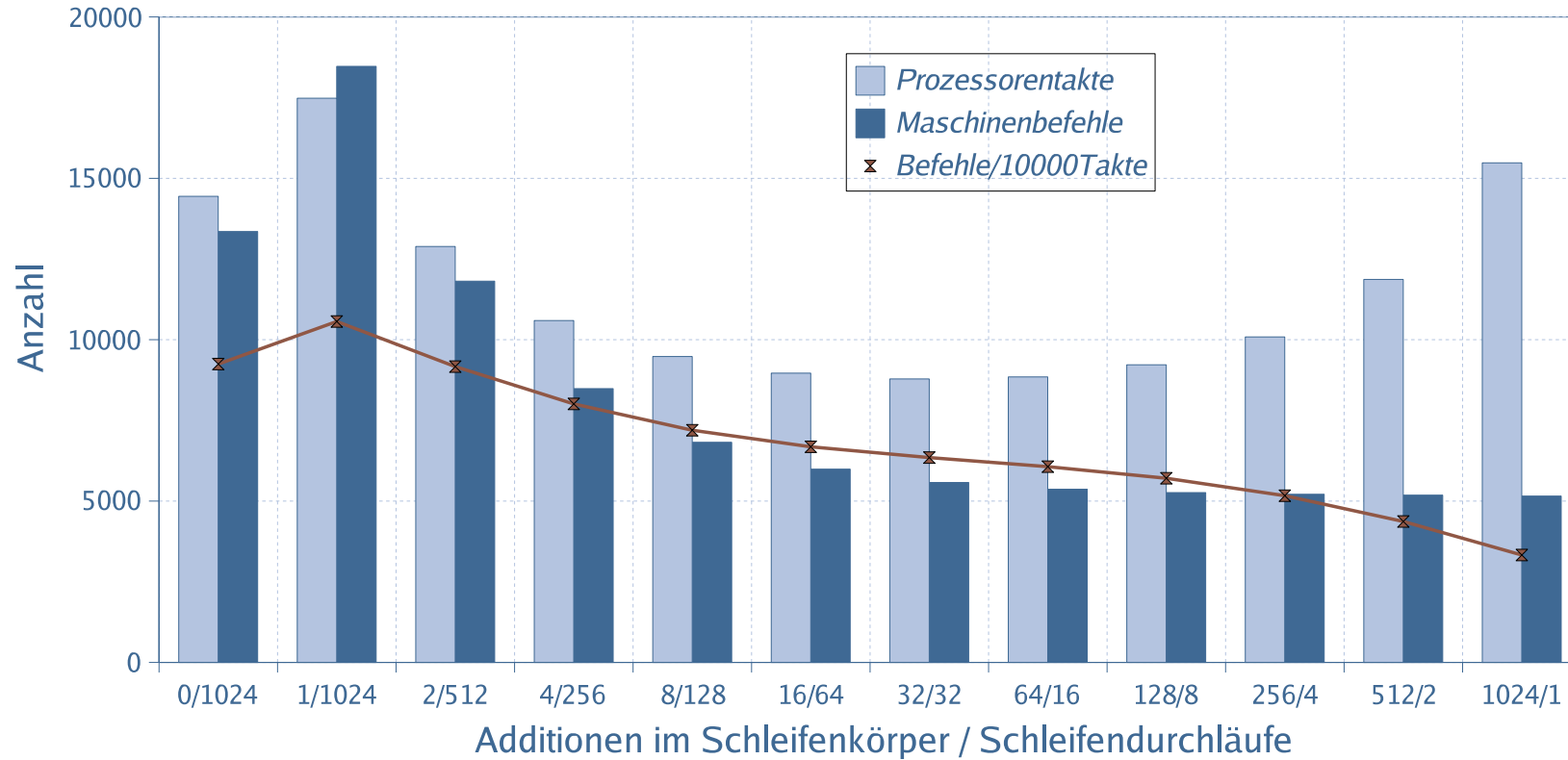
Ausgabe des Treibers

```
Interrupts are:           Enabled
Disable Interrupts:      [   OK   ]
ICache is:              Enabled [   OK   ]
DCache is:              Enabled [   OK   ]
Serialization is:       Disabled [   OK   ]
Saving Settings         [   OK   ]
```

```
PMC1_CYCLES=[14442 17482 12888 10593 9480 8964 8784 8847 9223 10086 11869]
PMC2_LOAD_MISS_PENALTY=[14 0 0 0 0 0 14 0 0 0 0 0 ]
PMC1_RESERVATIONS_REQ=[0 0 0 0 0 0 0 0 0 0 0 0 ]
PMC2_INSN_DISPACHED=[13352 18469 11815 8485 6821 5989 5575 5367 5263 5211]
PMC1_ICACHE_MISS=[5 3 4 4 8 13 23 42 82 162 322 645 ]
PMC2_DCACHE_MISS=[2 0 0 0 0 0 1 0 0 0 0 0 ]
PMC1_DTLB_MISS=[0 0 0 0 0 0 0 0 0 0 0 0 ]
PMC2_ITLB_MISS=[0 0 0 0 0 0 0 0 0 0 0 0 ]
PMC1_BRANCH_MISPREDICT=[1 1 1 1 1 1 1 1 1 1 1 1 ]
PMC2_BPU_OUT=[2052 2052 1028 516 260 132 68 36 20 12 8 6 ]
```



Messergebnisse



Vergleich von Messungen und Vorhersage

