1. Einführung Architektur = Struktur+Funktionen

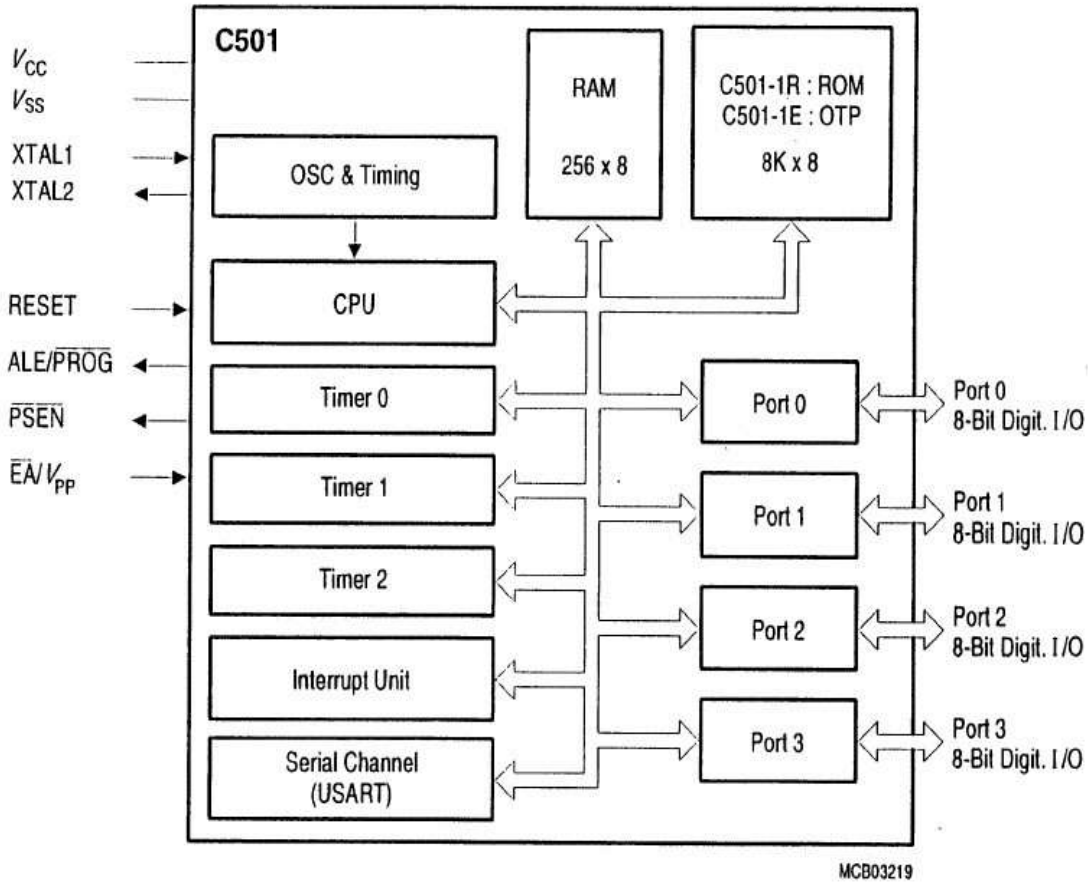
Spezialrechner	Universalrechner
angepaßt an bestimmte Anwendung	keine Anpassung
Anwendung is tbekannt	Anwenundung nicht bekannt
„so gut wie nötig“	„so gut wie möglich“
starke Peripherieorientierung	stärkere Bearbeitungsorientierung
Echtzeitforderungen	

2. Spezialrechner

einbettendes System – eingebettetes System [Einchipcontroller, MC, EMRechner, DSP]

2.1 Einchipcontroller

a) EMR der unteren Leistungsklasse



Bsp. C501 Infineon:

CPU: 8bit Datenbreite bei ca. 40MHz

- Interne Takte- und Reseterzeugung
- RAM: intern; klein; schnell wie Register
- oft kombiniert mit ProzRegistern, Ints schnell
- ROM: verschiedene Varianten
  - Maskenrom (Einmalige Anfertigung → Massenprodukt)
  - PROM, OTP programmable ROM
  - EPROM erasable PROM, löschbar
  - EEPROM electrically EPROM, Flash-EEPROM
  - „Bound-Out-Variante“ – Anschlüsse nach Außen

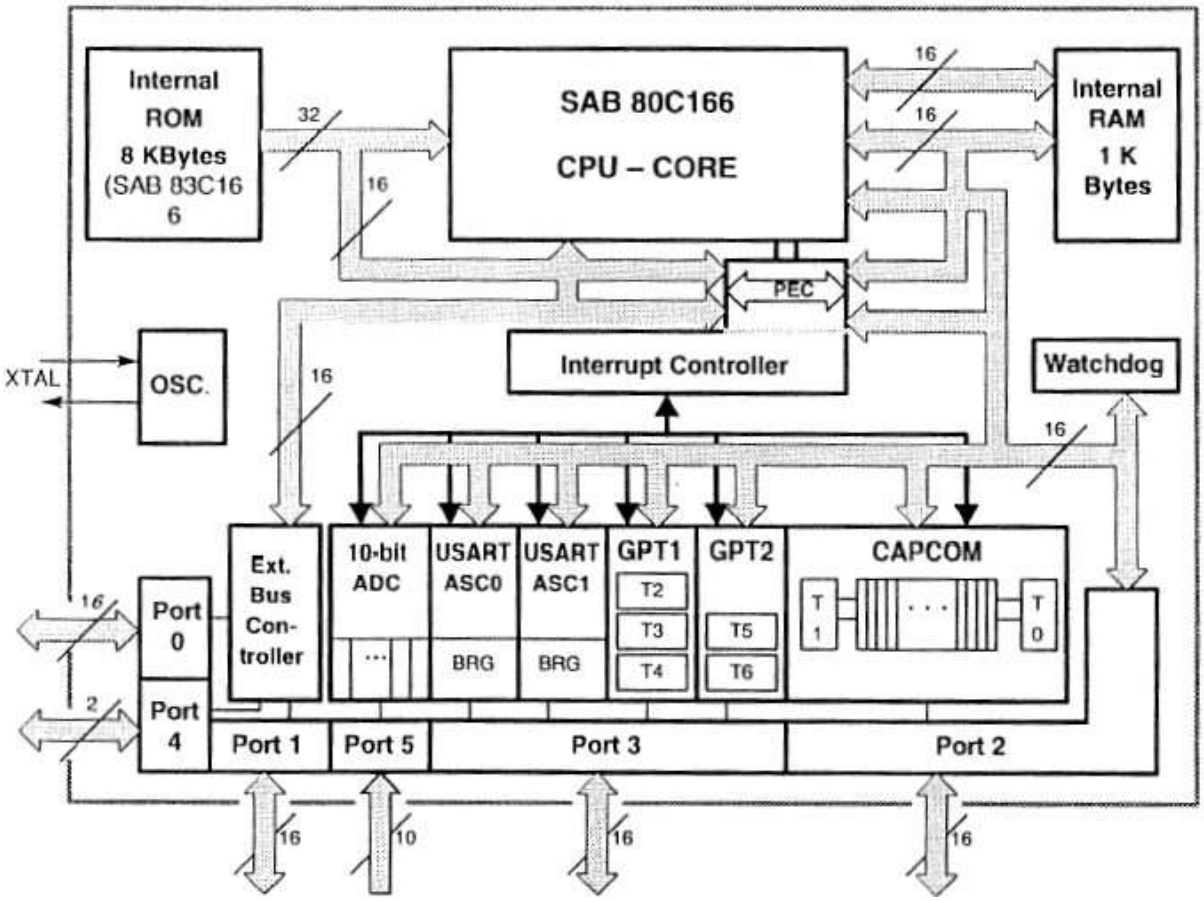
Peripherie: starke Konfigurierbarkeit

- 32Bit parallele digitale E/A (4x8)

- USART Universal/Asynchron/Receiver/Transmitter
- "Timer" → Zähler+Zeitgeber
    - z.B. Zeitmessung, zyklische Vorgänge, Ereigniszählung (Impulssensoren)
- Interruptsystem: Signalerfassung von Außen, Priorisierung

Bestandteile Controllerchip:
Timer (acuh für INTs), CPU, USART (Interface nach Außen), Ports (Tasten & Display)

2.2 Einchipmikrorechner höhere Leistungsfähigkeit
Beispiel: C166 (Infineon)



XTAL – chrystal, PLL – Phase blcoked loop (an äußeren Takt angepaßt)
Peripherial Event Controller – Vorpriorisierung der events von Außen

- Datenbreit: 16bit
- Adreßraum: bis 2^{32}bit
- Interner Speicher: „größer"
- Leistungsfähiger Befehlssatz
- viele E/A Funktionen
- starke Konfigurierbarkeit
- teurer

Watchdog:

Zählerstand überschreitet bestimmten Wert → Reset

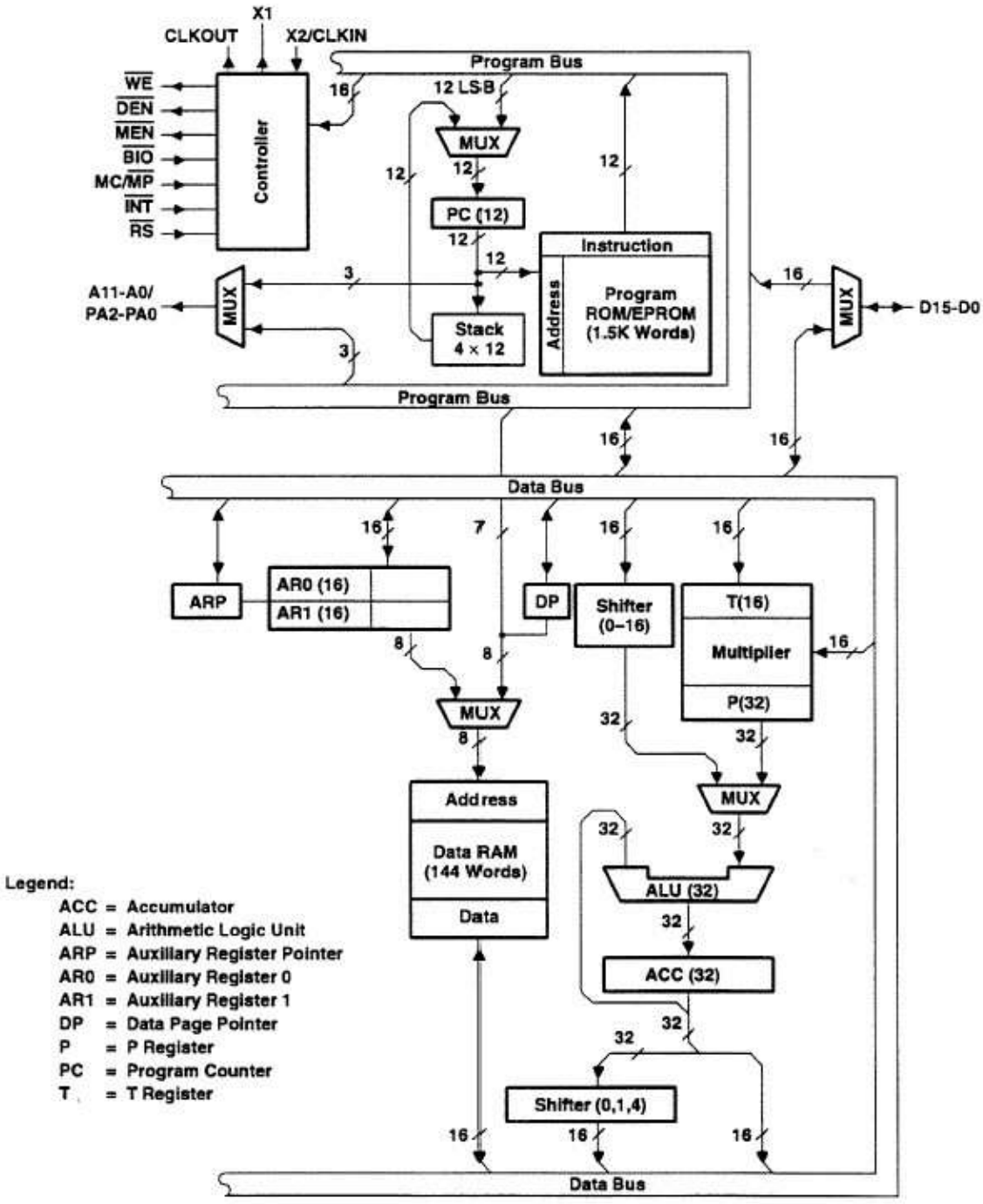
- ASC – asynchron/ asynchr. control
- GPT – gerneral purpose timer = Timer as usual
- BRG – Baud Rate Generator
- CAPCOM – capture compare → Anwendung z.B. PWM-Ausgang (PulseWidthModulation) [Blockbild: Compare(Zähler+Vergleichsregister)→out]

Periode bleibt gleich (low+high zusammen); Verhältnis low:high ändert sich aber

2.3 DSPs – zw. AD und DA-Wandler
Zweckbestimmung: Durchsatz, auf höhere Rechenleistung ausgelegt
→ erhöhte Datentransportfähigkeit
Vorteile digital: analog zu tolerant, weniger Verschleiß, idealere Eigenschaften, niedrigere Frequenzen

Wert- und Zeitdiskretisierung, typische Anwendungen
Tonsignalverarbeitung, Bildverarbeitung, Digitale Regler, Radartechnik, Arithmetikeinheit, eingebettete Systeme (mit hohem arithm. Durchsatz)

z.B. 1. Generation



0-Adr-Maschine („Stackmaschine“)
1-Adr-Maschine („Akkumulatormaschine“)
2-Adr-Maschine (z.B. x86)
→ Anzahl der Operanden in tyischem Befehl

Kombinierte Befehle: z.B.: Rechnen/Transport; ADD/MULT [MAC], beliebige Befehle

4. Generation – 3-Adr-Maschine
Merkmale: ausgebaute Transportstruktur auf Chip; Durchsatz durch Gleichzeitigkeit

ERR (R0-R11) – Genauigkiet, Gleitkomma intern 40bit, dargestellt werden 32bit
DISP, IR0, IR1 – Adreßberechnungen

3. Entwicklung der Rchnerarchitekturen

3.1 Einführung

Beschleunigung des Ablaufs:

- höhere Taktfrequenzen
- innere Parallelisierung („Mikroparallelität“)

3.2 CISC – Architekturen z.B. x86, IBM360, Z80

Complex Instruction Set Computing

- mächtige Maschinenbefhle

3.3 RISC – Architekturen z.B. IA-64, PowerPC, SPARC, MIPS

Reduce Instruction Set Computing

- einfache Maschinenbefehle von einheitlicher Mächtigkeit (Anz. kann trotzdem hoch sein)
- Load/StoreArchitektur

CISC	RISC
einfache & complexe Befehle	nur einache Befehle (einheitlich)
heterogener Befehlssatz	Orthogonaler Befehlssatz (systematisch aufgebaut)
Ausnahmen (best. Registerkombinationen nicht verwendbar)	z.B. ADD op1 op2 alle ops und Adr. einsetzbar
verschiedene Taktzahlen pro Befehl	meist 1 Takt pro Befehl
viele Befehlscodeformate mit unterschiedlicher Länge	weniger Formate, einheitliche Länge
Mikroprogrammierwerk (Steuerwerk, jeder Befehl löst Ablauf eines Unterprogrammes aus)	Direktverdrahtung (1 Schritt, keine Verzweigung)
Vermischung von Verarbeitungs- & Speicherbefehlen	Trennung Verarbeitung und Speicherbefehle (Abläufe getrennt von Abläufen des Speichers)

Gründe für CISC: vorhandene Software;
Konvergenz RISC+CISC z.B. Prozessor intern RISC extern CISC

4. Fortschritte im Speichersystem

- Normalfall: 32 Bit breite Wörter übereinander
- mit Interleave: Bänke ansprechen („modulo“)
- Burstmode

RAS/CAS – Row/Column Adreßselection

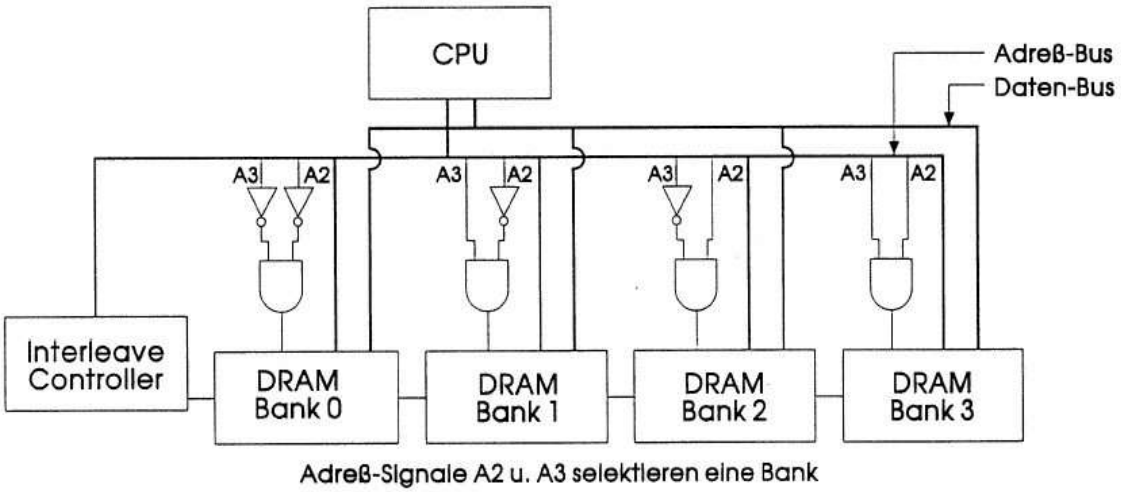


Abb. 4.20 DRAM-Struktur mit vier Speicherbänken

Speicherfamilien mit Adreßpipelining, Burst Mode und Banking:
dynamisch: SDRAM(SDR,DDR,QDR) & RDRAM
statisch: SBSRAM (synchr. – burst-static)

Speicherhierarchie: CPU-Cache-Hauptspeicher
groß – langsam & klein – schnell
Ziel: Nutzer möchte Haupspeicher schnell wie Cache →

Lokalität von Speicherzugriffen:
- temporal: mehrmaliger Zugriff auf dieselbe Speicherzelle in begrenzter Zeit
- spatial: Zugriff auf benachbarte Adressen in begrenzter Zeit

Trefferrate (Hitrate) eines Caches:
T=Anzahl der im Cache vorgefundenen Befehle/Gesamtanzahl der Speicherzugriffe

resultierende durchschnittliche Zykluszeit:
tr = (1-T)ta+Ttc
ta…Zykluszeit des Arbeitsspeichers pro Zugriff
tc…Zykluszeit des Cachespeichers pro Zugriff

Konsistenz: Übereinstimmung Cache und Speicher
Kohärenz: Übereinstimmung zw. verschiedenen caches

4.3 Bus Hierarchie
PCI – Peripheral Component Interconnect
EISA – Extended Industry Standard Architec.
PCEB – PCI-EISA-Bridge (Zeitverhalten voneinander entkoppeln)
ESC – EISA – System component
AGP – kein Bus, da nur ein Teilnehmer
MPI – Mltiprozessor Interconnect

5. Beispielarchitekturen (out of order eigentlich bei RISC)

PPro: Konverter CISC→RISC

- einfache Befehle: z.B. ADD AX,BX → je ein RISC-Befehl
- mittelkomplexe Befehle z.B. ADD [EAX],BX →mehrere RISC-Befehle
- hochkomplexe Befehle z.B. „ENTER“ → Mikroprogramm

PowerPC – Prozessor Chip
Power – Performance Optimized with enhanced RISC

VLIW…very large instruction word (superskalar), inorder, RISC-Befehlssatz
Prädikatierung – jeder einzelne Befehl kann bedingt sein
EPIC – Explicitly Parallel Instruction Computing

6. Leistungsbewertung von Architekturen

Ziel: Vergleich, Entwurfsprozeß, Verkaufsargument

6.1 Taktfrequenzen „P-Rating“

CPI = Taktanzahl/Befehlsanzahl clocks per instruction (kleiner ist besser)

mittlere Operationszeit – Berechnung aus Befehlskategorien:
z.B. Load/Store, Verzweigung/Sprung, Integer, FP, Logik/Schieben, sonstiges…
→ CPI = Summe über CPI mal P der n…Befehlsgruppen
skalare Architekturen habe CPI < 1

6.2 Leistung

„Befehl pro Zeit“
MIPS: „million instructions per second“
L[MIPS]=Befehlsanzahl/Zeit in µs
L[MIPS]=f[MHz]/CPI

MIPS ist ungerecht:

	einfacher CISC	einfacher RISC

Testprogramm	ADD[R2],R1	MOV R3,[R2]
		ADD R3,R1
		MOV [R2],R3

Taktzahl	3	2+1+2=5
Taktfrequenz	1500MHz	2000MHz
Ausfühungszeit	2ns	2,5ns
→ MIPS	500	1200

Deshalb: Relative MIPS (Vergleichsmaschine festlegen)
Klassische Referenzmaschine: VAX 11/7 80 („1-MIPS-Maschine“)
Lrel=Lref*(refZeit/aktZeit)

Unterscheidung: Peak Performance (Spitzenleistung); Sustained Performance (Dauerleistung)

Leistung bei Gleitkomma: MFLOPS
Lgk[MFLOPS]=Anzahl GK-Befehle/Zeit in µs (bei GFLOPS in ns)

Normierung um Ungerechtigkeiten für spezielle Befehle auszugleichen.
(ADD,SUB mit 1; DIV,SQRT mit 4; EXP,LOG mit 8)

6.3 Einfluß der Speicherzugriffe
Formel nach Sieworek,Bell u. Newell
Lsbn=1/CPI T[s]+S ts
S[bit]…Speicherbedarf pro Durchschnittbefehl
ts [s/bit]…zusätzliche Speicherzugriffszeit

erforderliche Speicherbandbreite für ein verzögerungsfreies Speichersystem:
Speicherbandbreite[Bit/s]=S/CPI T

6.4 Programme zur Leistungsbewertung
Ziele: realistisch, vergleichbare, reproduzierbare Leistungsbewertung
berücksichtigt: Taktfrequenz,CPU-Architekt, Speichersystem einschl. Cache, Compilereinfluß

Arten von Benchmarks:

a) Reale Programme: eingeschränkt, Interaktion
b) Kernels: extrahierte Kernbefehlsfolgen, ohne Interaktion
c) Toys – SpieleBenchmarks
d) Synthetische Benchmarks – Befehlsaufreihung; z.B. Dhrystone (Festkommanweisungen, paßt in Cache)

7. Parallele Rechnerarchitekturen

hier: MakroParallelität

Enge Kopplung z.B. Zweitorspeicher
Leistung bei Gleitkomma:

- **MFLOPS** – mega floating point operations per second
- **GFLOPS** – giga floating point operations per second
- **TFLOPS** – tera floating point operations per second

LGK *[MFLOPS]* = Anzahl GK-Befehle / Zeit *[µs]* = Leistung

Reale GK-Operationen	**„Normalisierte“ GK-Operationen**
ADD, SUB, MUL, COMP	1
DIV, SQRT	4
EXP, LOG, SIN, ...	8

Beispiele:

Prozessor	**MIPS**	**GFLOPS**
8086 (5 MHz)	0.4	---
Pentium 4 (3GHz)	6000	8
Earth Simulator	---	35000

6.3 Einfluss der Speicherzugriffe

Formel nach Sieworek, Bell und Newell

LSBN *[1/s]* = 1 / CPI * T *[s]* + S * ts

S *[bit]* = Speicherbedarf pro Durchschnittsbefehl
CPI = Takte pro Befehl
T = Taktzeit
ts *[s/bit]* = zusätzliche Speicherzugriffszeit

Erforderliche Speicherbandbreite für ein verzögerungsfreies Speichersystem:

Speicherbandbreite *[bit/s]* = S / CPI * T

6.4 Programme zur Leistungsbewertung

Ziele: realistische, vergleichbare, reproduzierbare Leistungsbewertung

Berücksichtigt:
- Taktfrequenz
- CPU – Architektur
- Speichersystem einschließlich Cache
- Compiler
- …

→ Benchmarkprogramme = standardisierte Testprogramme zur Leistungsbestimmung

Hier: Benchmark für die Rechenleistung
Arten von Benchmarks:

1. Reale Programme
   - Eingeschränkt
   - Evtl. mit Interaktion oder I/O

2. Kernel
   - Extrahierte Kernbefehlsfolgen
   - Ohne Interaktion und I/O
   - z.B.: „linpack“, „livermore loops“, „SPEC – system performance evaluation cooperative“

3. Toys („Spiel Benchmarks“)
   - Einfache, kurze Algorithmen
   - Leicht portierbar
   - Begrenzte Aussagekraft
   - z.B.: „sieve“, „quicksort“, „puzzle“

4. synthetische Benchmarks
   - Befehlsfolgen ohne „Sinn“ (Algorithmus), aber korrekte Befehlshäufigkeit
   - z.B.: Whetstone, Dhrystone

7 Parallele Rechnerarchitekturen10

bisher: Mikro – Parallelität (im Inneren der CPU)
hier: Makro – Parallelität

Klassifikation nach Flynn:

SISD	MISD
SIMD	MIMD

single } instruction , single } data
multiple multiple