Technische Grundlagen der Informatik 2

Rechnerorganisation

Prof. Dr. Ben Juurlink

Fachgebiet: Architektur eingebetteter Systeme
Institut für Technische Informatik und Mikroelektronik
Fak. IV – Elektrotechnik und Informatik

SS 2014

Zielgruppen

TechGI2: 2 VL + 2 UE (6 LP)

- Informatik, Wirtschaftsingenieurwesen

TechGI2**TI**: 2 VL + 2 UE + 2 Praktikum „Digitale Systeme“ (8LP)

- Technische Informatik

Prüfung

- **VL**: Klausur mit Zugangsvorraussetzung (01.08. + 22.09.)
  - 4 Hausaufgaben, pro HA 33% der Punkte, insg. 50%
  - Muss in dem Semester erbracht werden, in dem die Klausur geschrieben werden soll

- **PR**: Unbenotetes Testat
  - Protokollierte Leistungen + wöchentliche Rücksprachen
  - Muss einmalig erbracht werden

- **Anmeldung** über QISPOS oder Prüfungsamt (siehe geltende Studien- und Prüfungsordnung) nach Vorlesungsende Mitte Juli
- **Modulnote** = Note der Abschlussklausur

Übungen/Tutorien

- Papier- und Rechnerübungen in Kleingruppen
- **Anmeldung** über MOSES (www.moses.tu-berlin.de)
  - **Frist** endet ***morgen***, 16.04.2014, 18.00 Uhr
  - Auch fürs **Praktikum** erforderlich!
- Beginn in der 2. Vorlesungswoche (ab 22.04.), Praktikum ab 3. (28.04.)
- Abgabe + Benotung der Hausaufgaben durch zugeteilten Tutor
- Tutor 1. Ansprechpartner für alle Fragen + Probleme

Weitere Infos?

- Aktuelles und Organisation des Übungsbetriebs via ISIS (http://www.isis.tu-berlin.de/2.0/)
  - Übungsblätter und Hausaufgaben
  - Angaben zu Räumlichkeiten (Tutorien, Übungen am Rechner, Klausur)
  - Sprechstunden der Tutoren
  - Angaben zur Klausur

- Direkter Kontakt:
  - Tutorienbetrieb, Hausaufgaben, Klausur: *techgi2@aes.tu-berlin.de*
  - Praktikum: *techgi2pr@aes.tu-berlin.de*

Fragen, Probleme?

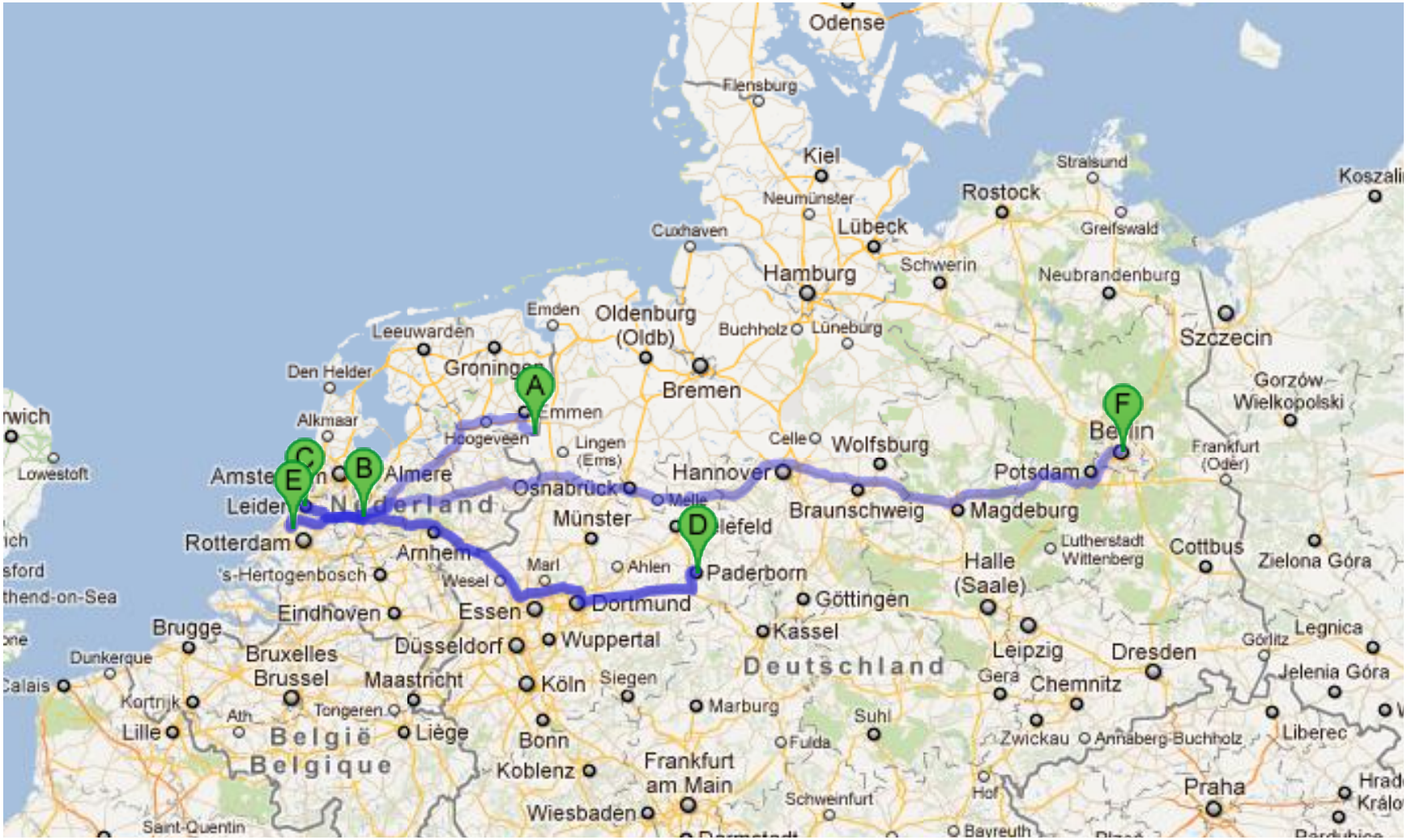
- Tutor
  - 2h Sprechstunde pro Woche in Raum E-N 630
  - Sprechzeiten auf ISIS-Seite
- Foren
  - Über ISIS-Seite
  - Moderierte peer-to-peer Diskussionen
- Sprechstunde
  - Di, 8.00-10.00 Uhr
  - Termin vereinbaren via techgi2@aes.tu-berlin.de

Wer bin ich, *who am I*?

Prof. Dr. B.H.H. (Ben) Juurlink

- 1968: geboren in Nieuw Schoonebeek, Niederlande
- 1987-1992: MSc Computer Science, Utrecht University (NL)
- 1992-1996: PhD Computer Science, Leiden University (NL)
- 1997-1998: Postdoctoral fellow, Heinz Nixdorf Institute, Paderborn University (DE)
- 1998-2009: Assistant/associate professor, Delft University of Technology (NL)
- Seit 2010: Professor für Architektur eingebetter Systeme, TU Berlin

Werdegang

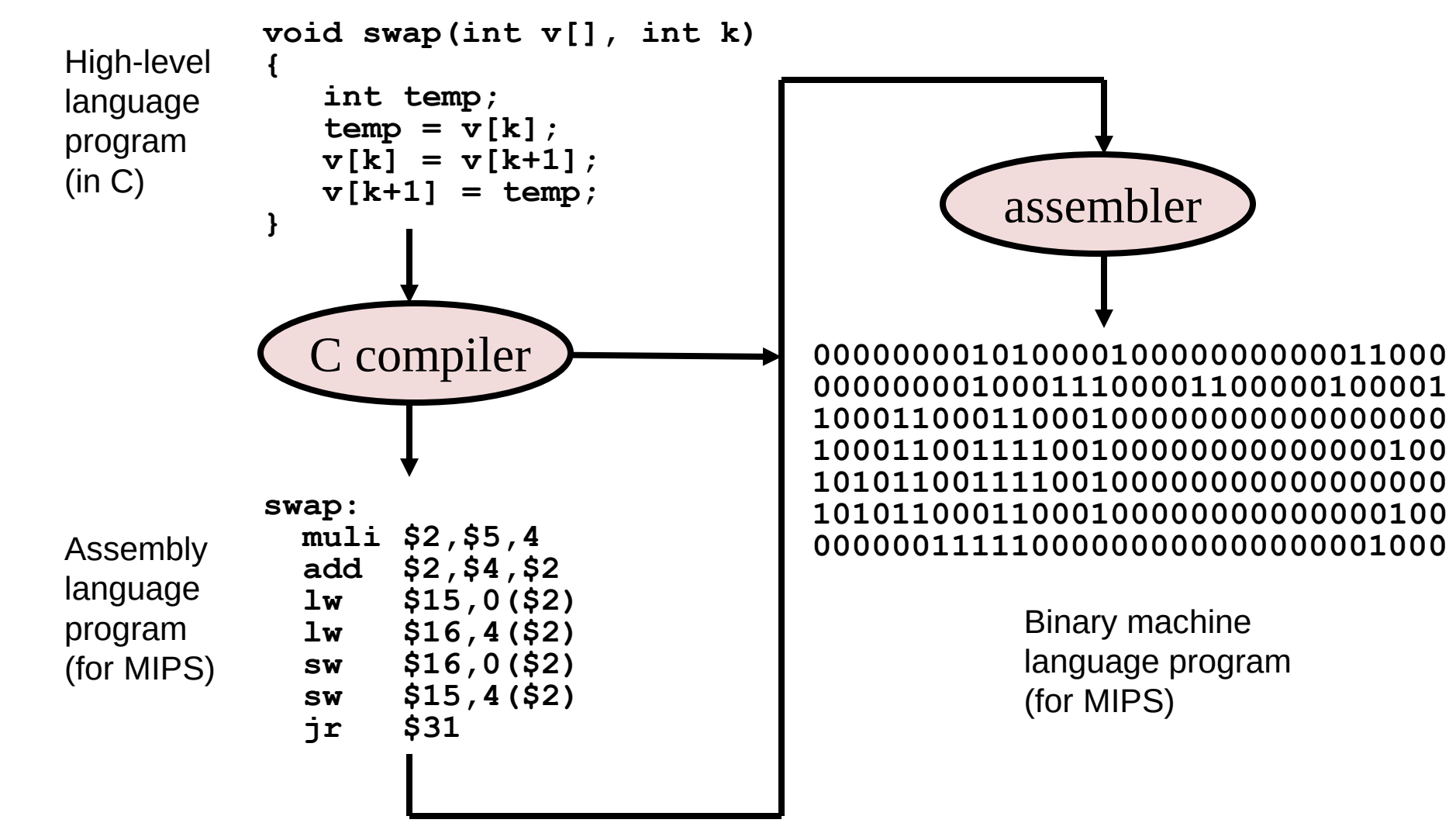

A
B
C
D
E
F
Odense
Flensburg
Kiel
Neumünster
Cuxhaven
Lübeck
Stralsund
Rostock
Greifswald
Koszali
Hamburg
Schwerin
Neubrandenburg
Szczecin
Emden
Oldenburg (Oldb)
Buchholz
Lüneburg
Leeuwarden
Groningen
Den Helder
Bremen
Gorzów Wielkopolski
Alkmaar
Emmen
Hoogeveen
Lingen (Ems)
Celle
Wolfsburg
Berlin
Frankfurt (Oder)
Lowestoft
Amsterdam
Almere
Hannover
Potsdam
Osnabrück
Leiden
Braunschweig
Magdeburg
Münster
Rotterdam
Arnhem
Lutherstadt Wittenberg
Cottbus
Zielona Góra
's-Hertogenbosch
Wesel
Marl
Ahlen
Paderborn
Halle (Saale)
Eindhoven
Essen
Dortmund
Göttingen
Brugge
Kassel
Legnica
Dunkerque
Düsseldorf
Wuppertal
Leipzig
Dresden
Görlitz
Bruxelles
Brussel
Maastricht
Köln
Siegen
Gera
Chemnitz
Jelenia Góra
Calais
Kortrijk
Marburg
Suhl
Tongeren
Lille
Ath
Liège
Bonn
Fulda
Zwickau
Annaberg-Buchholz
Liberec
België
Belgique
Koblenz
Frankfurt am Main
Schweinfurt
Hof
Praha
Wiesbaden
Saint-Quentin
Bayreuth

Hauptziele

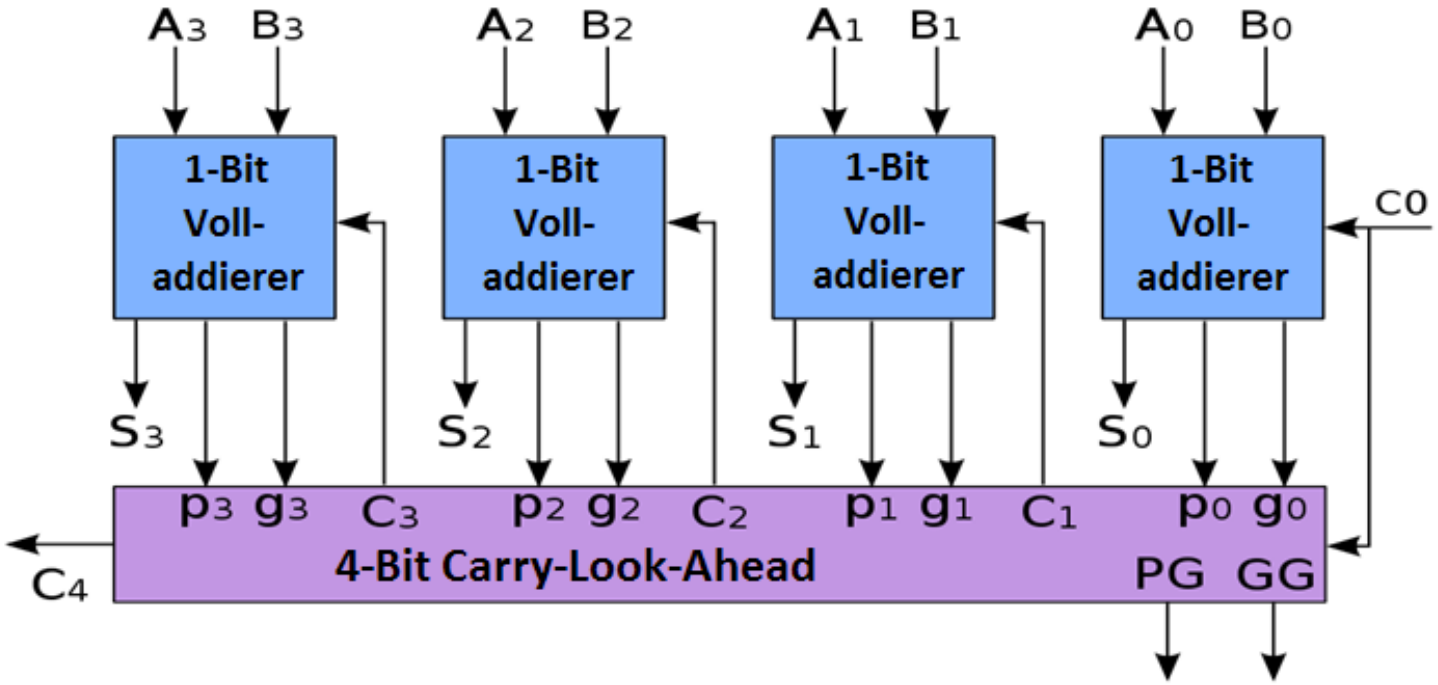
- Verstehen, wie ein Programm geschrieben in einer höheren Programmiersprache (z. B. C oder Java) in eine Maschinensprache übersetzt und von einem digitalen System ausgeführt wird
- Die mit der Bearbeitung der Maschinenbefehle einhergehenden logischen Abläufe in einem digitalen System auf der Register-transferebene verstehen und ggfs. erweitern
- Fähigkeit, die Systemfunktionalität in konstruktiver Weise mittels eines endlichen Automaten oder mittels Mikroprogrammierung festzulegen
- Kenntnisse in Zahlendarstellungen und in den für die arithmetischen Operationen zugrundeliegenden Mikroalgorithmen
- Kompetenzen im Aufbau digitaler Systeme, einschließlich Ein-/Ausgabe-organisation, und in den Strukturprinzipien von Rechnern

Themen

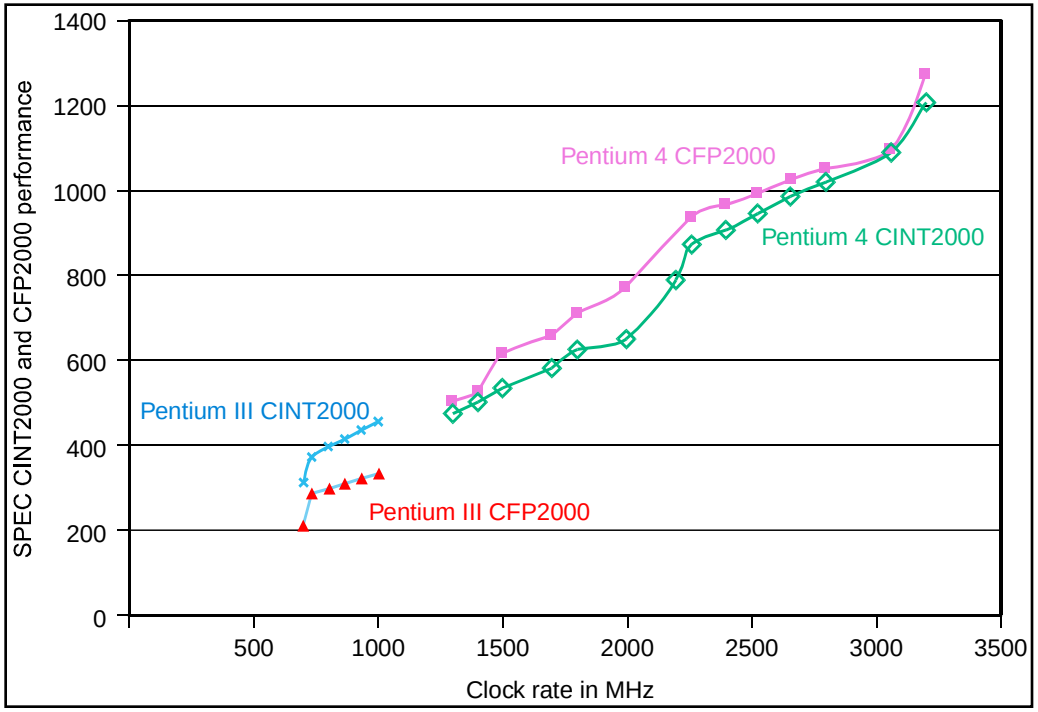
- Grundlegende Technologien und Komponenten einer Rechnerarchitektur
- Assemblerprogrammierung: Assemblersprache, Steuerkonstrukte, Adressierungsarten.
- Rechnerarithmetik: Zahlendarstellungen (Stellenwertsysteme, Fest- und Gleitpunktzahlen), Mikroalgorithmen für arithmetische Operationen.
- Codes (Ziffern- und Zeichencodes, Codesicherung)
- Rechenleistung verstehen und beurteilen (SPEC benchmarks, Amdahl's Law).
- Aufbau und Funktionsweise eines einfachen Von-Neumann-Rechners.
- Aufbau und Funktionsweise einer Mehrzyklenimplementierung.
- Fliessbandverarbeitung (Pipelining), Pipelinekonflikte und ihre Lösungen.
- Speicherhierarchie, Caches, virtueller Speicher.
- Ein-/Ausgabetechniken (Adressierung, Synchronisation, Direktspeicherzugriff).
- Merkmale moderner Prozessoren (Superskalarität, VLIW, Multi-Core).


High-level language program (in C)
void swap(int v[], int k)
{
int temp;
temp = v[k];
v[k] = v[k+1];
v[k+1] = temp;
}
C compiler
swap:
muli $2,$5,4
add $2,$4,$2
lw $15,0($2)
lw $16,4($2)
sw $16,0($2)
sw $15,4($2)
jr $31
Assembly language program (for MIPS)
assembler
00000000101000010000000000011000
00000000100011100001100000100001
10001100011000100000000000000000
10001100111100100000000000000100
10101100111100100000000000000000
10101100011000100000000000000100
00000011111000000000000000001000
Binary machine language program (for MIPS)

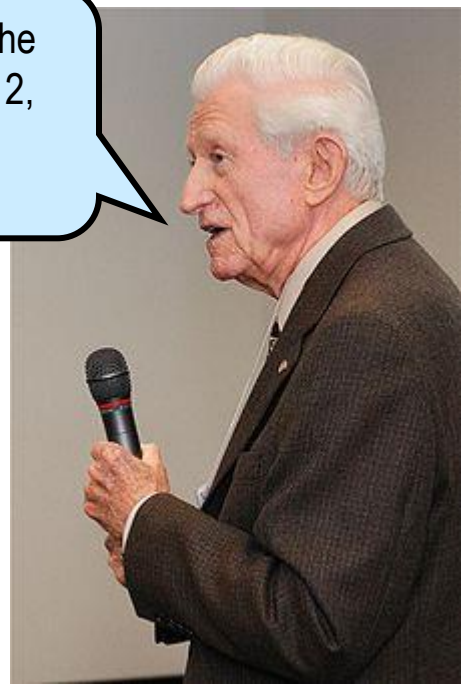
$$b_{31}b_{30}\ldots b_1b_0 = -b_{31}2^{31}+b_{30}2^{30}+\ldots+b_12^1+b_02^0$$



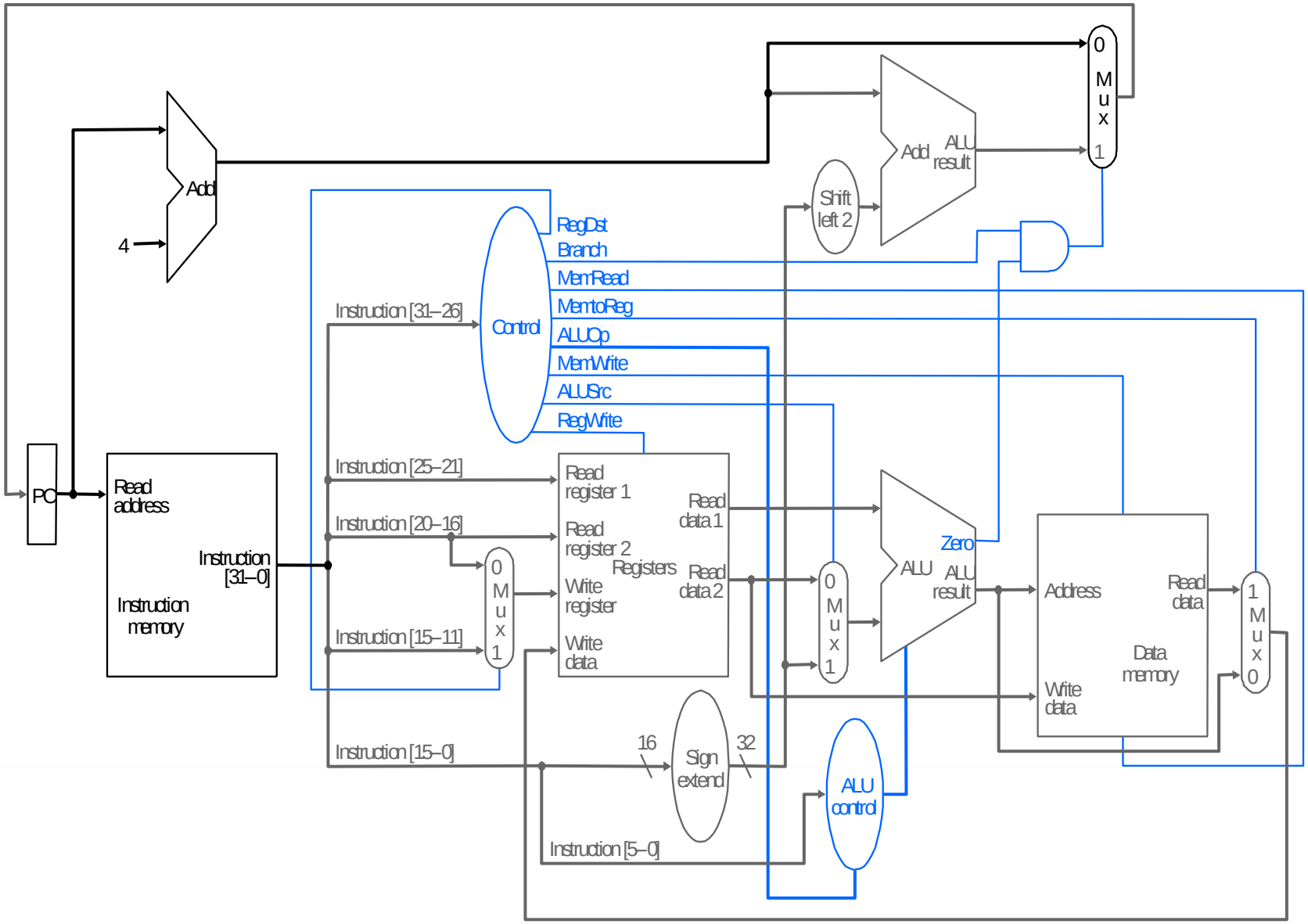
$$T = N_{instr} \cdot CPI \cdot t_{cycle} = \frac{N_{instr} \cdot CPI}{f}$$

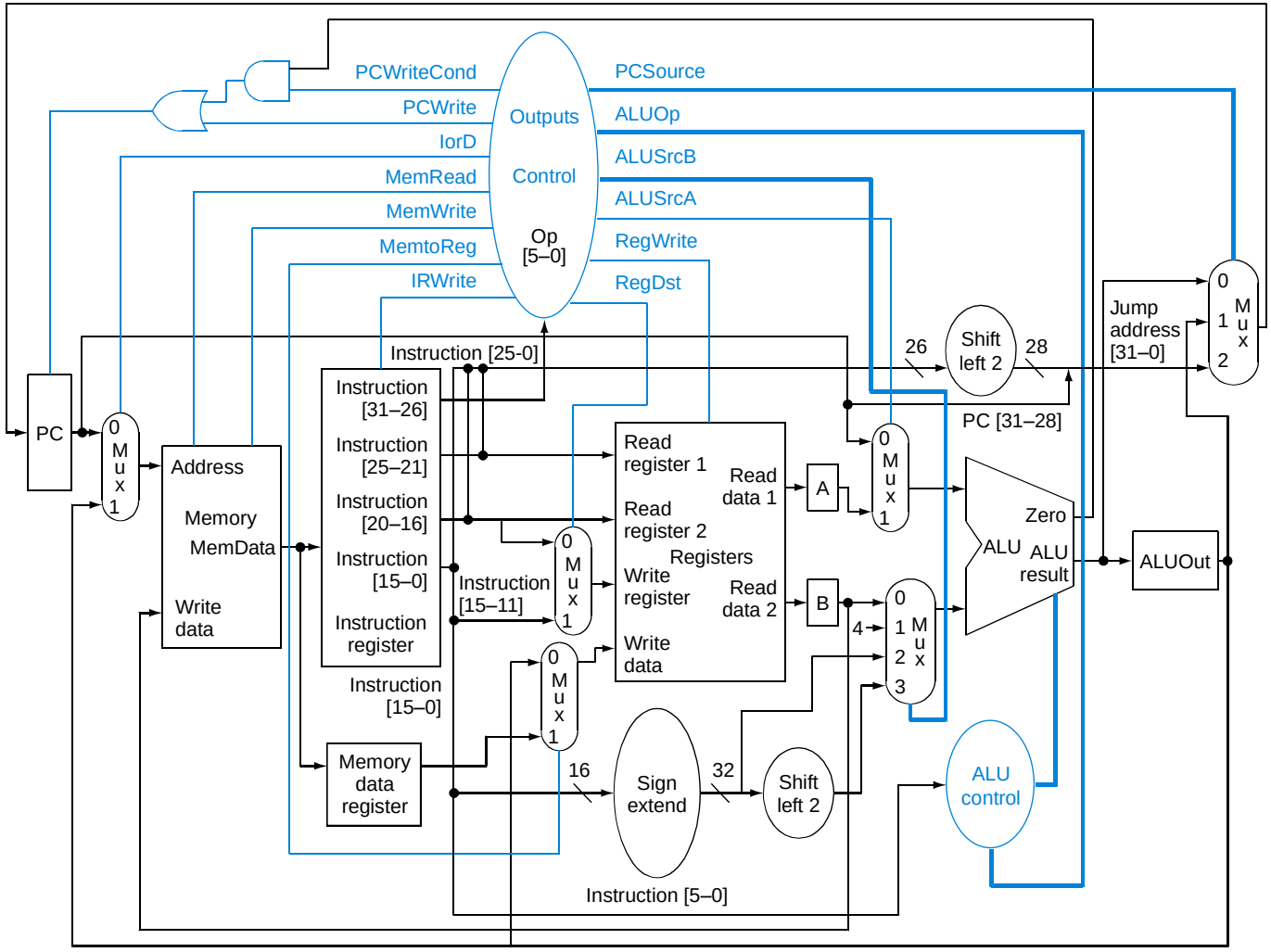


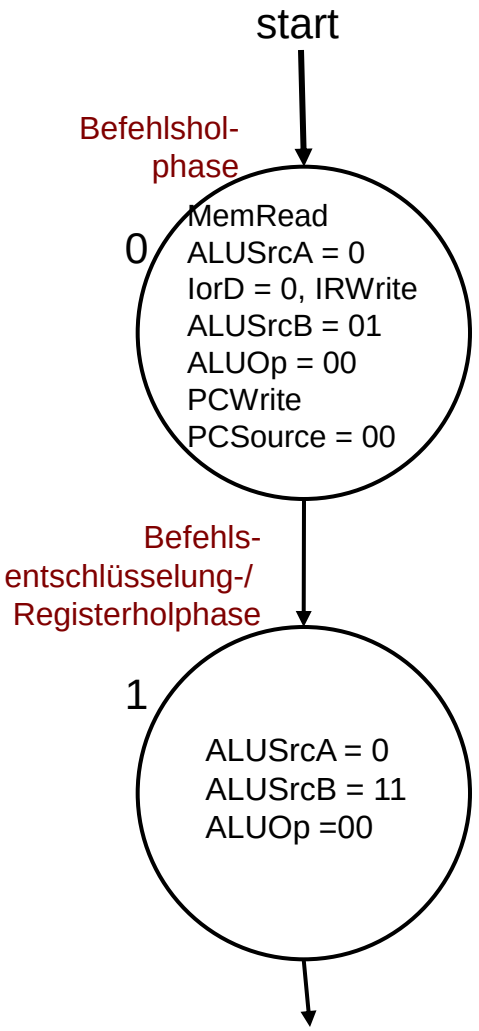
If 50% is sequential, the maximum speedup is 2, no matter how many cores you use

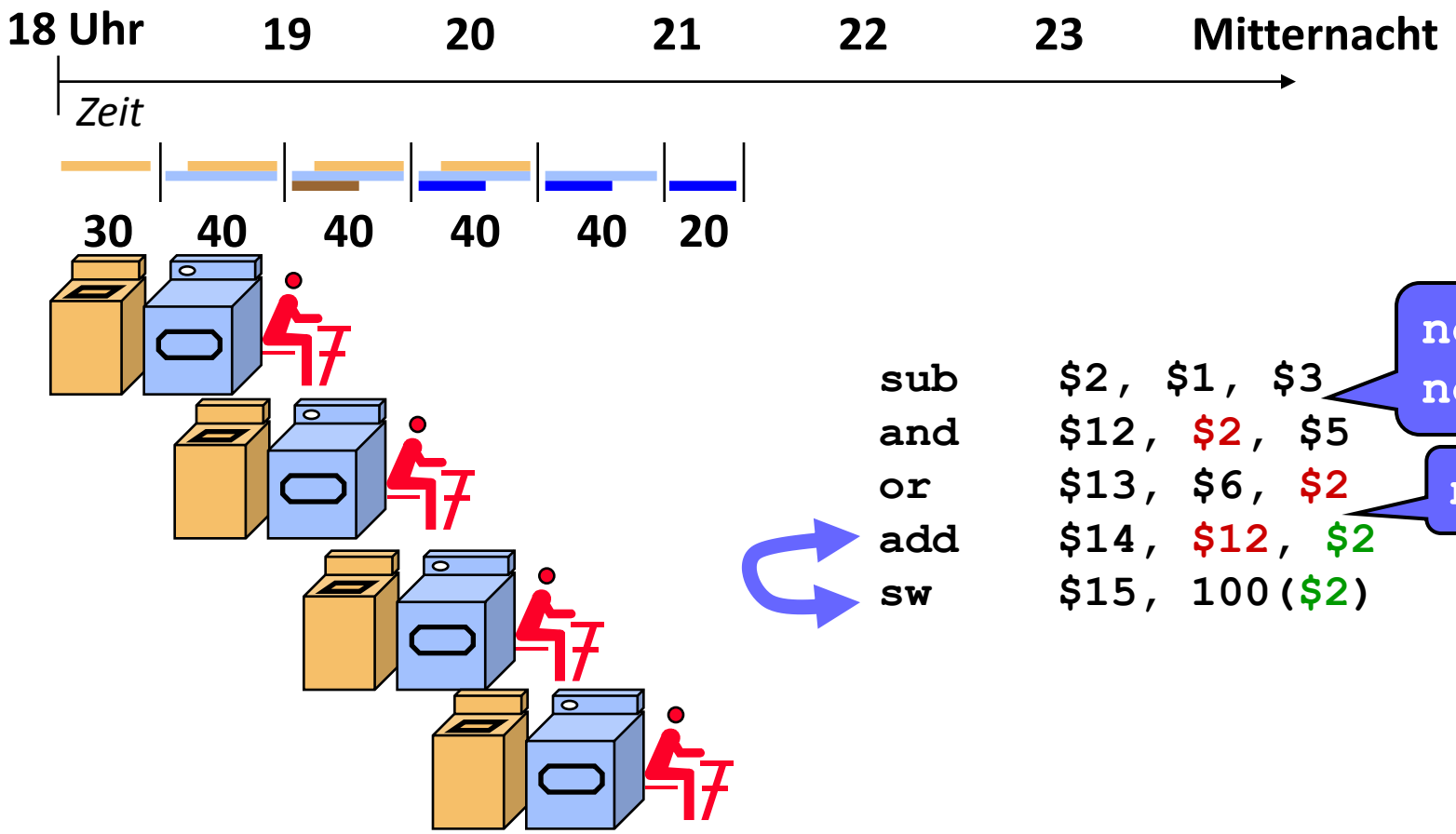


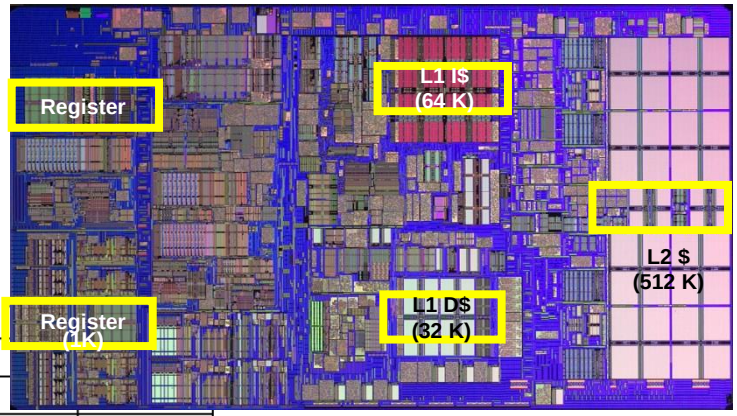
Thema 4: Der Eintaktprozessor

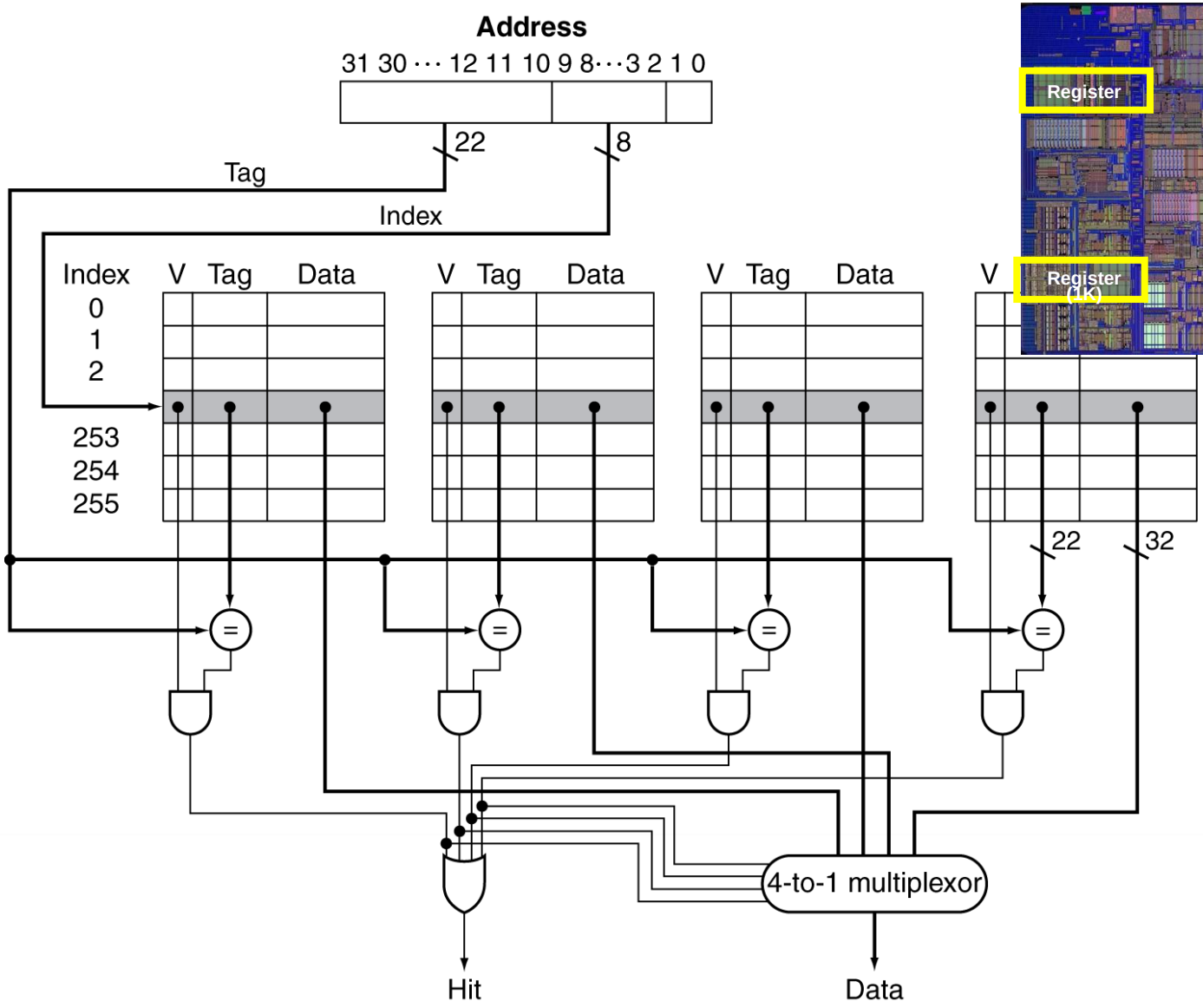


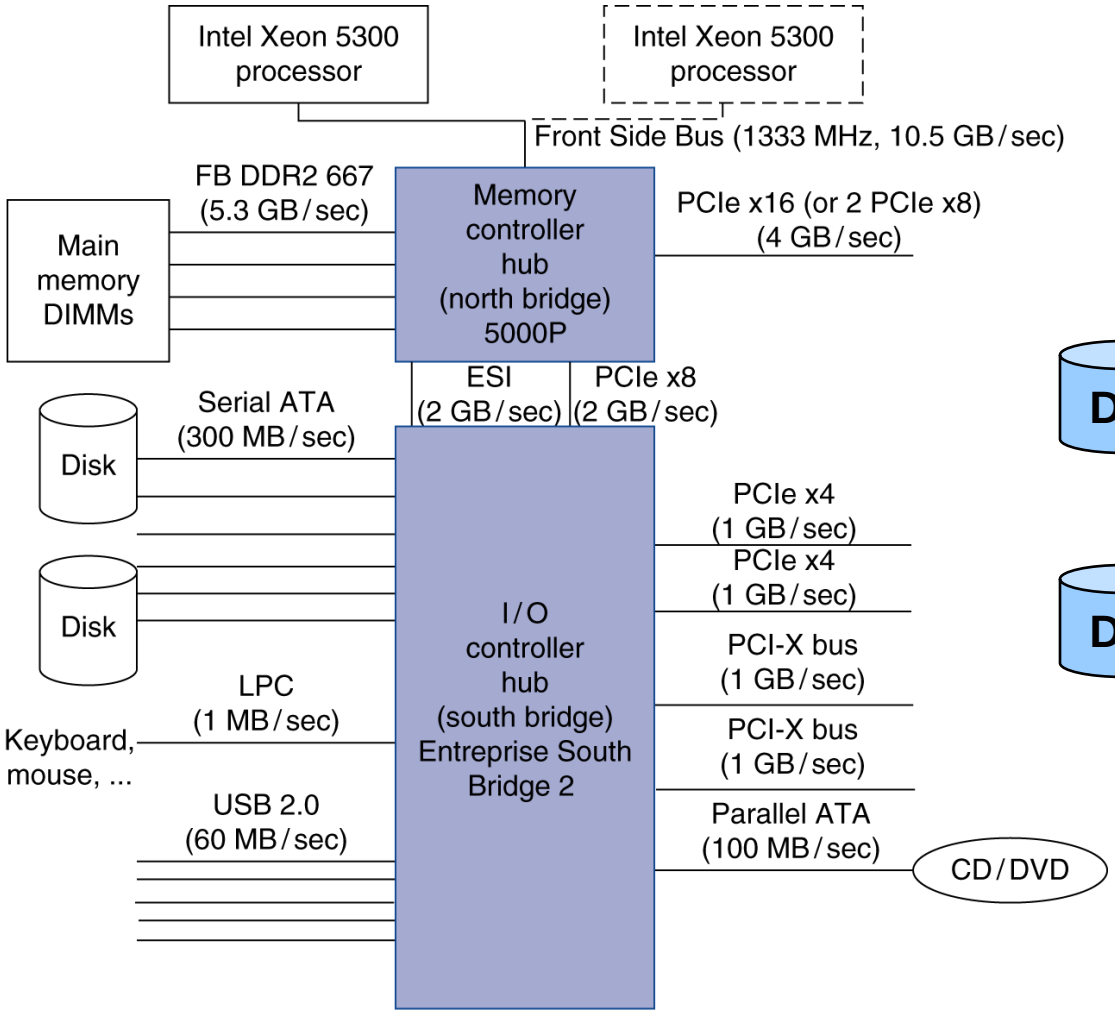

PCWriteCond
PCWrite
IorD
MemRead
MemWrite
MemtoReg
IRWrite
Outputs
Control
Op [5–0]
PCSource
ALUOp
ALUSrcB
ALUSrcA
RegWrite
RegDst
PC
0 M u x 1
Address
Memory
MemData
Write data
Instruction [31–26]
Instruction [25–21]
Instruction [20–16]
Instruction [15–0]
Instruction register
Instruction [25-0]
Instruction [15–0]
Instruction [15–11]
Memory data register
Read register 1
Read register 2
Registers
Write register
Write data
Read data 1
Read data 2
A
B
4
ALU
Zero
ALU result
ALUOut
26
Shift left 2
28
PC [31–28]
Jump address [31–0]
0 1 2 M u x
0 1 2 3 M u x
16
Sign extend
32
Shift left 2
ALU control
Instruction [5–0]

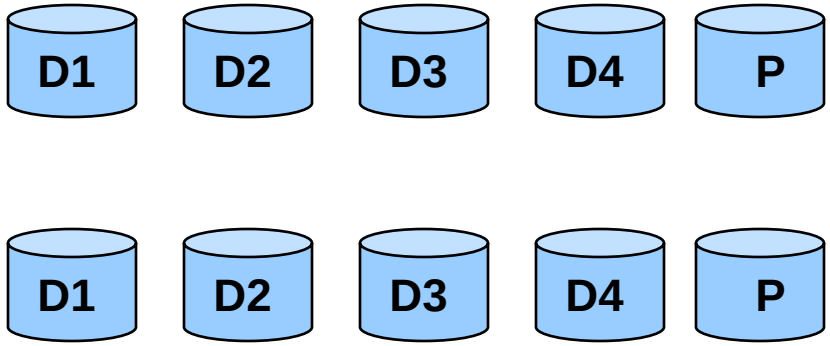

start
Befehlshol-phase
0
MemRead
ALUSrcA = 0
IorD = 0, IRWrite
ALUSrcB = 01
ALUOp = 00
PCWrite
PCSource = 00
Befehls-entschlüsselung-/ Registerholphase
1
ALUSrcA = 0
ALUSrcB = 11
ALUOp =00


18 Uhr
19
20
21
22
23
Mitternacht
Zeit
30
40
40
40
40
20
Arbeitsreihenfolge
A
B
C
D
sub $2, $1, $3
and $12, $2, $5
or $13, $6, $2
add $14, $12, $2
sw $15, 100($2)
nop
nop
nop

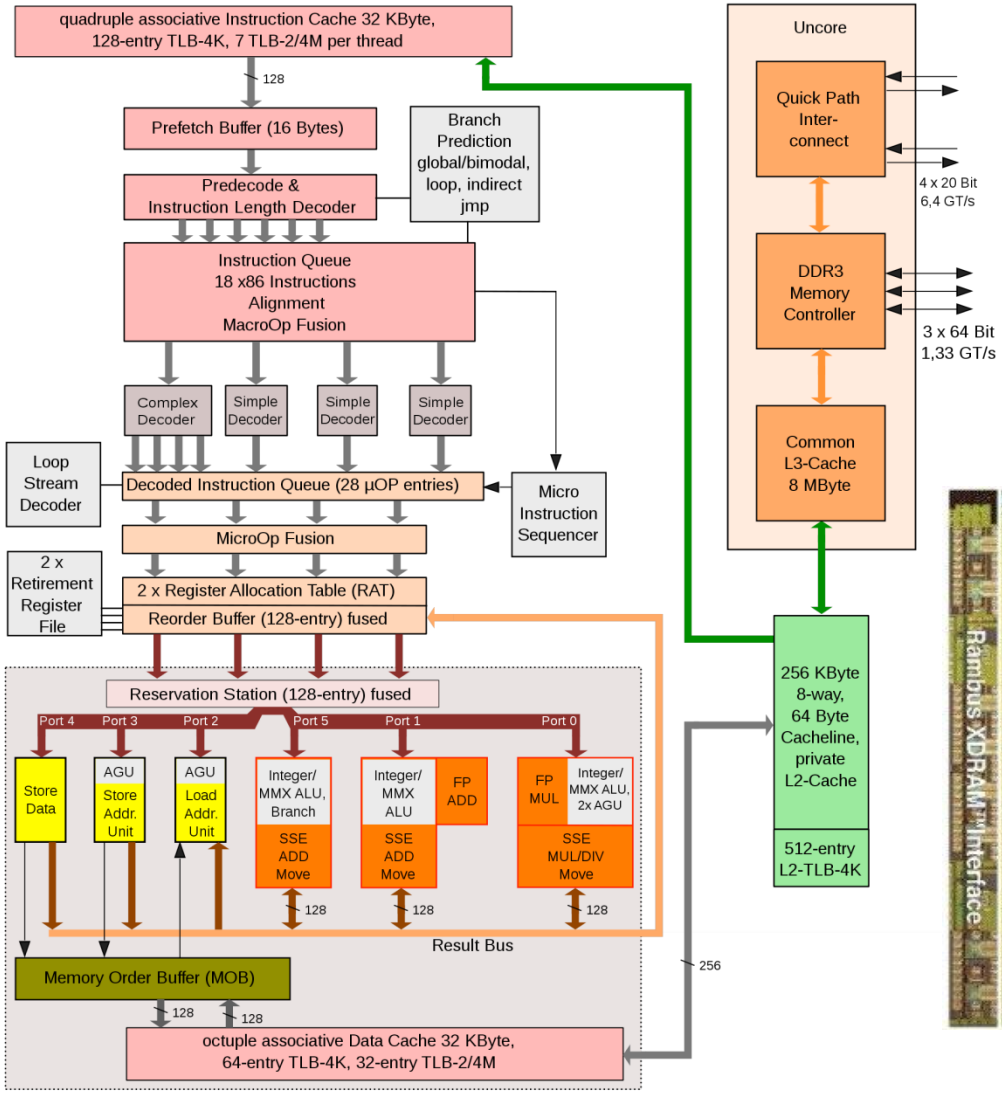

Register
L1 I$
(64 K)
L2 $
(512 K)
Register
(1K)
L1 D$
(32 K)

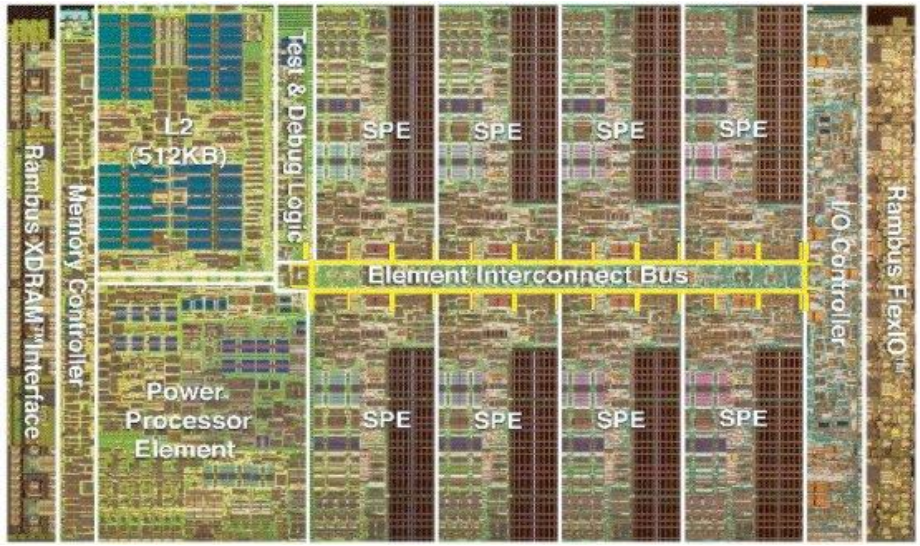

Address
31 30 ··· 12 11 10 9 8···3 2 1 0
22
8
Tag
Index
Index
V
Tag
Data
0
1
2
253
254
255
22
32
=
4-to-1 multiplexor
Hit
Data


Intel Xeon 5300 processor
Intel Xeon 5300 processor
Front Side Bus (1333 MHz, 10.5 GB/sec)
FB DDR2 667 (5.3 GB/sec)
Main memory DIMMs
Memory controller hub (north bridge) 5000P
PCIe x16 (or 2 PCIe x8) (4 GB/sec)
ESI (2 GB/sec)
PCIe x8 (2 GB/sec)
Serial ATA (300 MB/sec)
Disk
Disk
I/O controller hub (south bridge) Entreprise South Bridge 2
PCIe x4 (1 GB/sec)
PCIe x4 (1 GB/sec)
PCI-X bus (1 GB/sec)
PCI-X bus (1 GB/sec)
Parallel ATA (100 MB/sec)
CD/DVD
LPC (1 MB/sec)
Keyboard, mouse, ...
USB 2.0 (60 MB/sec)


D1
D2
D3
D4
P
D1
D2
D3
D4
P

Intel Nehalem microarchitecture





GT/s: gigatransfers per second

Buch

- Rechnerorganisation und -entwurf – Die Hardware/ Software-Schnittstelle
- David A. Patterson and John L. Hennessy
- Übersetzt von Arndt Bode, Wolfgang Karl und Theo Ungerer
- 3. Auflage im UB
  - 4. Auflage auch i. O. und jetzt auch auf Deutsch

Viel Spaß und frohes Schaffen!