

Betriebssysteme (BS)

VL 7.1 – IA-32: Das Programmiermodell der Intel-Architektur – Überblick

Volkmar Sieh / Daniel Lohmann

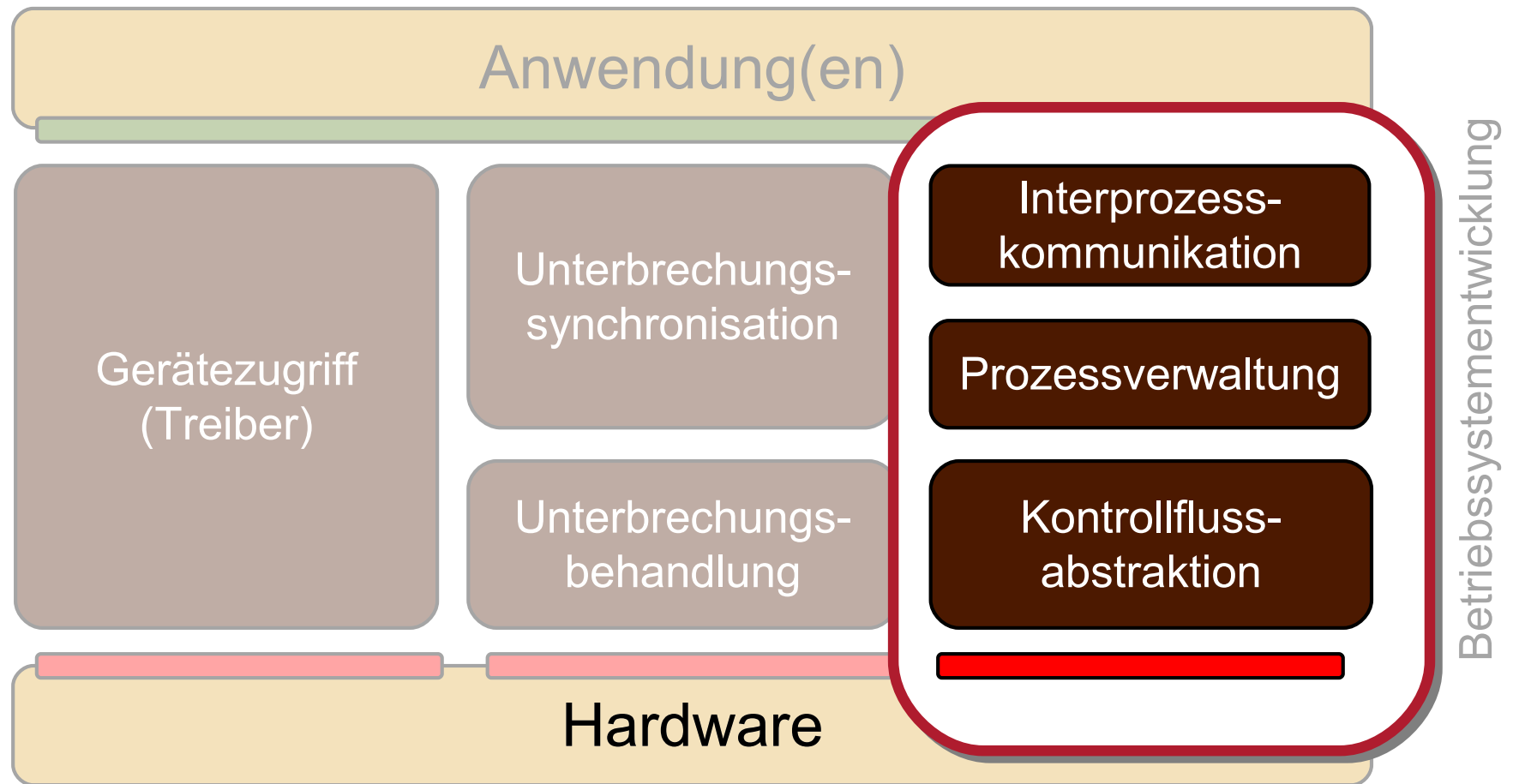
Lehrstuhl für Informatik 4
Verteilte Systeme und Betriebssysteme

Friedrich-Alexander-Universität
Erlangen Nürnberg

WS 21 – 6. Dezember 2021



Überblick: Einordnung dieser VL



Agenda

Einordnung

Urvater: Der 8086

Die 32-Bit Intel-Architektur

Protected Mode

Multitasking

Zusammenfassung



Historie der Intel x86-Prozessoren

- 1978: **8086** *der Urvater des PC Prozessors*
- 1982: **80286** *Einführung des Protected Mode*
 - segmentbasierter Speicherschutz
- 1985: **80386** *erster IA-32 Prozessor*
 - seitenbasierter virtueller Speicher
 - Protected Mode
- 1989: **80486** *integrierte FPU, RISC Ansätze*
- 1993: **Pentium** *P5-Architektur*
 - superskalar, 64-Bit Datenbus
 - SMM, MMX, APIC, Dualprozessor-fähig



- 1995: **Pentium Pro** *P6-Architektur*
 - 36-Bit Adressbus (PAE)
 - Level 2 Cache on Chip, RISC-artige Mikroinstruktionen
- 1997: **Pentium II** *Pentium Pro + MMX*
 - Level 2 Cache wieder extern, dafür bessere 16-Bit Performance
- 1999: **Pentium III** *SSE, Pentium M (2003)*
- 2000: **P4** *Netburst-Architektur*
 - SSE2, optimiert für hohe Taktzahlen (angedacht bis zu 10 GHz)
- 2004: **P4 Prescott** *Erweiterte Netburst Architektur*
 - Hyperthreading, Vanderpool, EM64T, 31-stufige Pipeline!



- 2005: **Core** *Ende der Netburst Architektur*
 - geringerer Takt, weniger Strom, aber bessere Performance!
 - Architektur basiert auf P6-Architektur, kein Hyperthreading
- 2006: **Core 2** *Dual Core, Quad Core, 64 Bit*
- 2008: **Atom** *extrem stromsparend*
 - Architektur (wieder) CISC-lastiger, Ähnlichkeiten mit 486/Pentium
- 2009: **Core i7** *Nehalem-Architektur*
Sandy-Bridge-Architektur (2011)
Haswell-Architektur (2013)
Skylake-Architektur (2017)
 - Hyperthreading, Octa Core, Quick Path Interconnect, AVX
 - “Power Control Unit” (PCU) passt Takt der TDP an
- 2012: **Xeon Phi** *Larrabee-Architektur*
 - P54C Manycore (62 cores), SIMD Instruktionen



Agenda

Einordnung

Urvater: Der 8086

 Programmiermodell

 Speichermodell

Die 32-Bit Intel-Architektur

Protected Mode

Multitasking

Zusammenfassung



8086: Programmiermodell

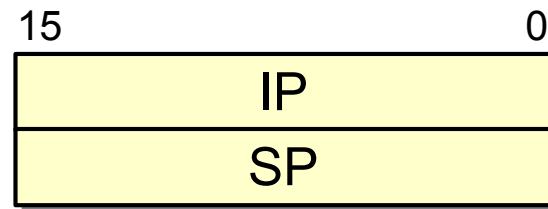
- 16-Bit Architektur, little-endian
- 20-Bit Adressbus, d.h. maximal 1 MiB Hauptspeicher
- wenige Register
 - (jedenfalls aus heutiger Sicht)
- 123 Befehle
 - kein orthogonaler Befehlssatz
- Befehlslängen von 1 bis 4 Byte
- segmentierter Speicher
- **noch immer aktuell**
 - obwohl von 1978 noch heute von jeder IA-32 CPU unterstützt
 - *Real Mode, Virtual 8086 Mode*

Aufwärtskompatibilität wird bei Intel groß geschrieben



8086: Registersatz

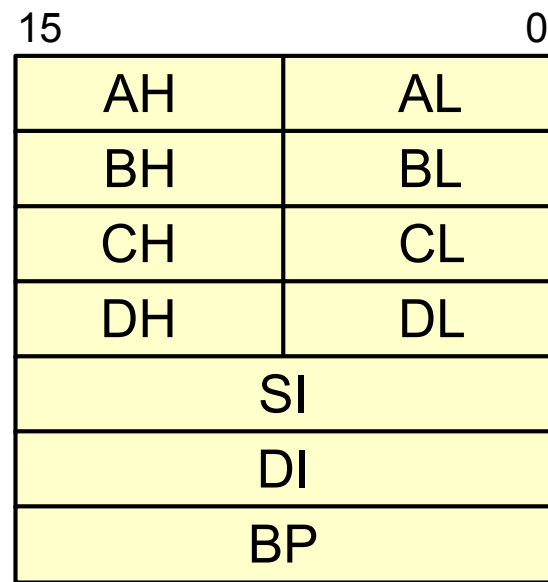
Befehls- und Stapelzeiger



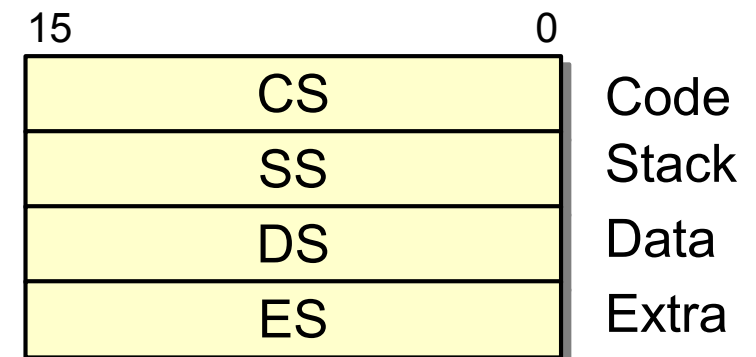
Flag Register



Vielzweckregister

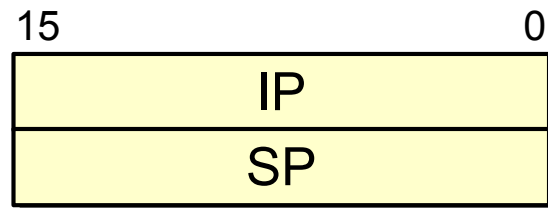


Segmentregister

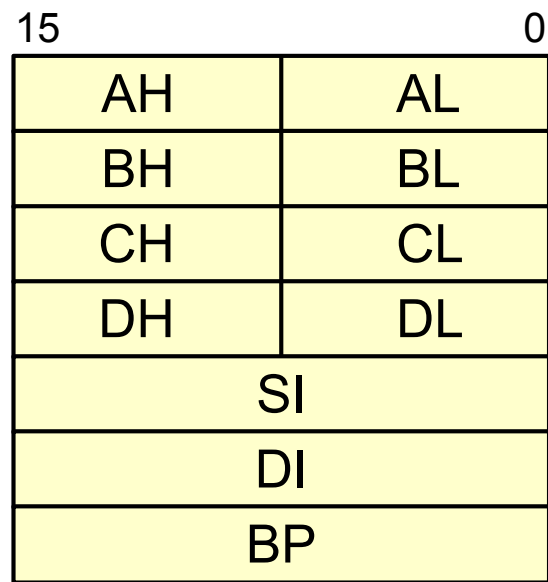


8086: Registersatz

Befehls- und Stapelzeiger



Vielzweckregister



AX: Accumulator Register

- arithmetisch-logische Operationen
- I/O
- kürzester Maschinencode

BX: Base Address Register

CX: Count Register

- für LOOP Befehl
- für *String* Operationen mit REP
- für Bit *Shift* und *Rotate*

DX: Data Register

- DX:AX sind 32 Bit für MUL/DIV
- Portnummer für IN und OUT

SI, DI: Index Register

- für Array-Zugriffe (Displacement)

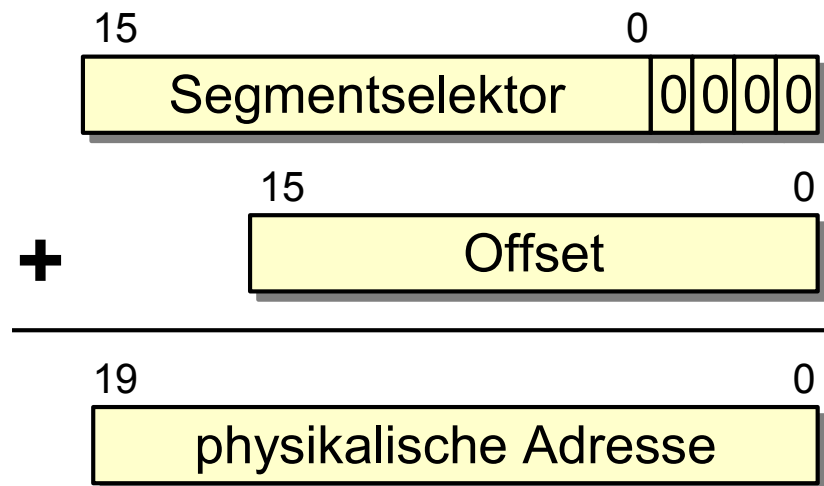
BP: Base Pointer

Jedes „Vielzweckregister“ erfüllt seinen speziellen Zweck



8086: Segmentierter Speicher

- logische Adressen bestehen beim 8086 aus
 - Segmentselektor (i.d.R. der Inhalt eines Segmentregisters)
 - Offset (i.d.R. aus einem Vielzweckregister oder dem Befehl)
- Berechnung der physikalischen Adresse:



die 16 Bit Konkurrenz konnte i.d.R. nur 64KB adressieren



8086: Segmentierter Speicher

- logische Adressen bestehen beim 8086 aus
 - Segmentselektor (i.d.R. der Inhalt eines Segmentregisters)
 - Offset (i.d.R. aus einem Vielzweckregister oder dem Befehl)

■ Bei

„640K ought to be enough for anybody“

angeblich ein Zitat von Bill Gates, 1981

physikalische Adresse

die 16 Bit Konkurrenz konnte i.d.R. nur 64KB adressieren



8086: Speichermodelle

- Programme können Adressen unterschiedlich bilden. Das Ergebnis waren unterschiedliche Speichermodelle:
 - **Tiny**
 - Code-, Daten- und Stacksegment sind identisch: 64K insgesamt
 - **Small**
 - Trennung des Codes von Daten und Stack: 64K + 64K
 - **Medium**
 - 32(20) Bit Zeiger für Code, Daten- und Stapelseg. aber fest (64K)
 - **Compact**
 - Festes Code Segment (64K), 32(20) Bit Zeiger für Daten und Stack
 - **Large**
 - „far“ Zeiger für alles: 1MB komplett nutzbar
 - **Huge**
 - wie „Large“, aber mit normalisierten Zeigern



- Urvater der PC-Prozessoren
 - bildete den Kern der ersten PCs
 - noch heute sind IA32-Prozessoren kompatibel
- Segmentregister brachten Vorteile
 - trotz 16-Bit-Architektur 1 MB Speicher
 - Trennung von logischen Modulen im Hauptspeicher
- Programm- und Übersetzerentwicklung ist aber vergleichsweise schwierig
 - verschiedene Speichermodelle
 - nicht orthogonaler Befehlssatz



Agenda

Einordnung

Urvater: Der 8086

Die 32-Bit Intel-Architektur

Erweiterungen

A20-Gate

Protected Mode

Multitasking

Zusammenfassung



IA-32 – die 32 Bit Intel Architektur

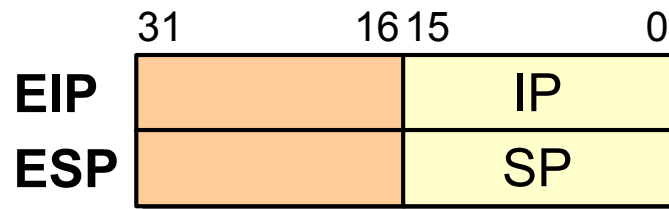
- die erste IA-32 CPU war der **80386**
 - wobei der Begriff „IA-32“ erst sehr viel später eingeführt wurde
- 32 Bit Technologie: Register, Daten- und Adressbus
 - ab Pentium Pro: 64 Bit Daten und 36 Bit Adressbus
- zusätzliche Register
- komplexe Schutz- und Multitaskingunterstützung
 - *Protected Mode*
 - ursprünglich schon mit dem 80286 (16-Bit) eingeführt
- Kompatibilität
 - mit älteren Betriebssystemen durch den *Real Mode*
 - mit älteren Anwendungen durch den *Virtual 8086 Mode*
- segmentbasiertes Programmiermodell
- seitenbasierte MMU



80386: Registersatz (Erweiterungen)

- erweiterte Register heißen aus Kompatibilitätsgründen E...

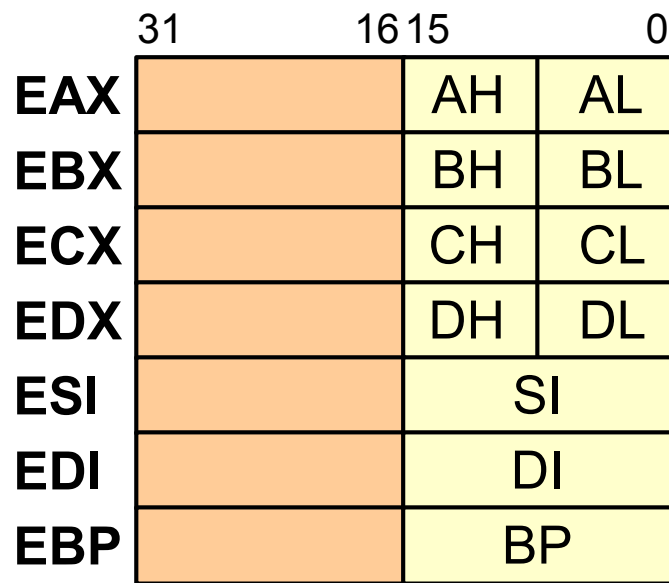
Befehls- und Stapelzeiger



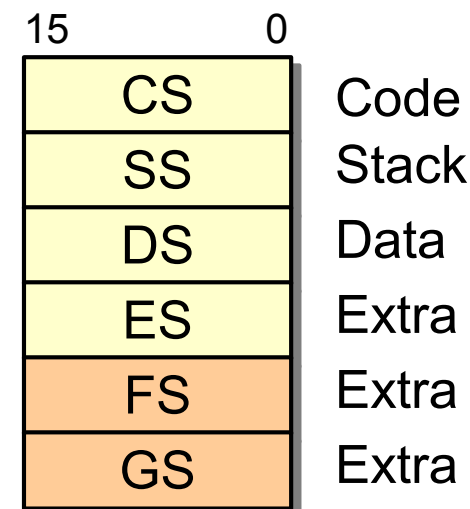
Flag Register



Vielzweckregister



Segmentregister



 Erweiterung zum 8086



80386: Registersatz (neue Register)

Speicherverwaltungsregister

	15	0 31	0 19	0
TR	TSS-Sel.	TSS-Basisadresse	TSS-Limit	
LDTR	LDT-Sel.	LDT-Basisadresse	LDT-Limit	
IDTR		IDT-Basisadresse	IDT-Limit	
GDTR		GDT-Basisadresse	GDT-Limit	

Erläuterungen
folgen ...

Steuerregister

	31	16 15	0
CR3			
CR2			
CR1			
CR0			

Testregister

	31	16 15	0
TR7			
TR6			

Debugregister

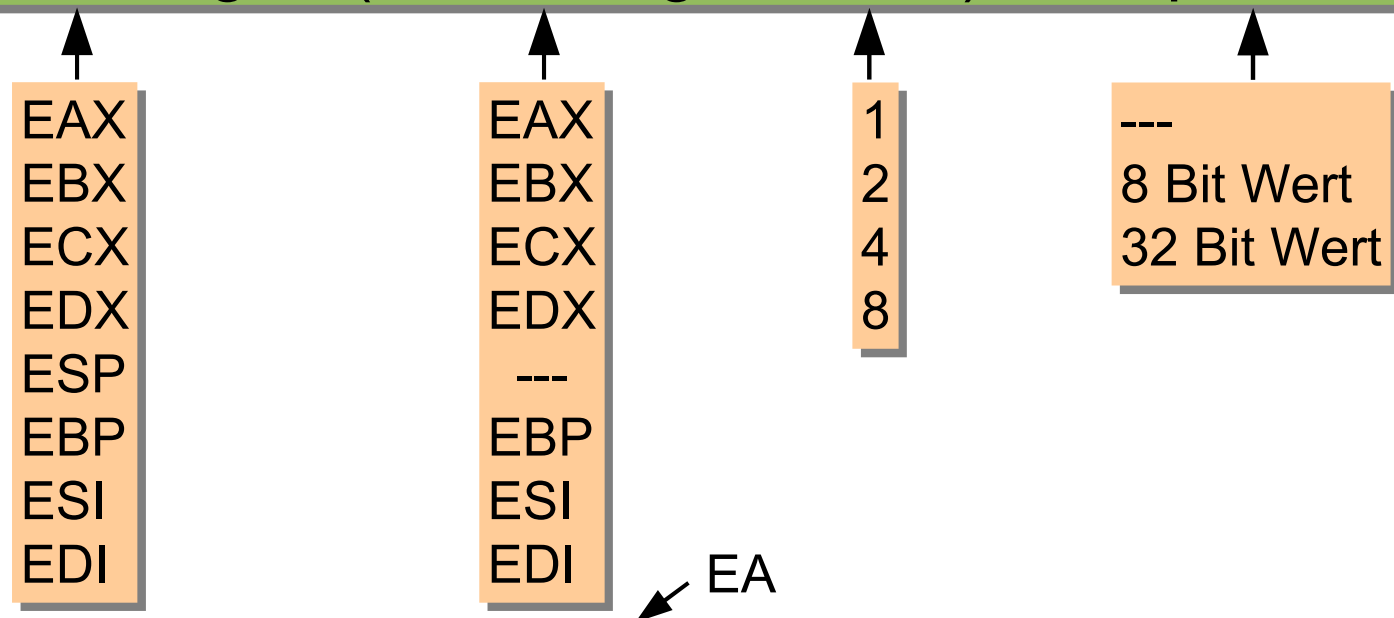
	31	16 15	0



IA-32: Adressierungsarten

- Effektive Adressen (EA) werden nach einem einfachen Schema gebildet
 - alle Vielzweckregister können dabei gleichwertig verwendet werden

$$\text{EA} = \text{Basis-Reg.} + (\text{Index-Reg.} * \text{Scale}) + \text{Displacement}$$



- Beispiel: `MOV EAX, Feld[ESI * 4]`
 - Lesen aus Feld mit 4 Byte großen Elementen und ESI als Index



Das A20-Gate

- ... ist ein Relikt aus der Zeit der 80286 Systeme (IBM AT)
- beim IBM XT (8086) konnte es bei der Adressberechnung zu einem Überlauf kommen. Im Maximalfall:

$$\begin{array}{rcl} & 0\text{x}\text{ffff0} & \leftarrow \text{Segment} * 16 \\ & + 0\text{x}0\text{ffff} & \leftarrow \text{Offset} \\ & \quad \quad \quad 1\ 1\ 1\ 1 \\ \hline 0\text{x}0\text{ffef} & \leftarrow \text{nur 20 Bit!} & 0\text{x}\textcolor{red}{1}0\text{ffef} \leftarrow \text{phys. Adresse} \end{array}$$

- MS-DOS (und andere Systeme) verwenden diesen „Trick“.
- Aus Kompatibilitätsgründen wurde im IBM AT die A20-Leitung über das „A20 Gate“ (Register im **Tastaturcontroller**) maskiert.
 - A20 muss explizit freigeschaltet werden, um Speicher > 1 MiB zu adressieren
- Ab dem 486 hat Intel das A20-Gate in die CPU integriert!

