Bus: Agenda

- Motivation für Busse
- ISA-Bus im PC/XT und PC/AT
- ISA Plug and Play
- EISA, MCA, VLB
- PCI-Bus
- AGP

Bus: Literatur

- allgemein, ISA:

  Tanenbaum, Computerarchitektur

  Hans Peter Messmer, PC-Hardwarebuch

  www.ieee.org IEEE Standards, z.B. VME-Bus, ISA-Bus, ...

- PCI und AGP:

  developer.intel.com

  www.pcisig.org

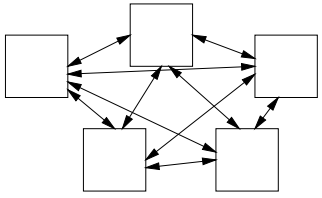
  www.webopedia.com/TERM/P/PCI.html

  www.techfest.com/hardware/

  www.pcguide.org/ref/mbsys/buses/

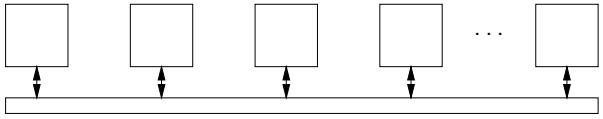
Bus: Motivation

Komponenten direkt verbinden:



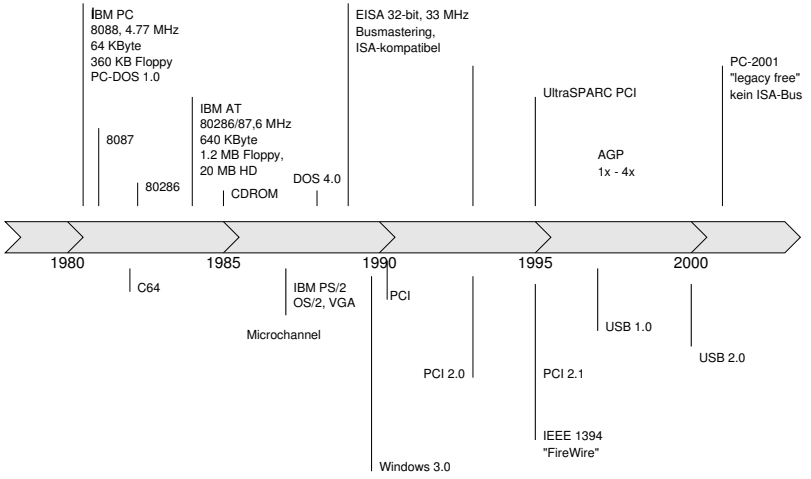
- viele Signale
- irreguläre Struktur

Bus:



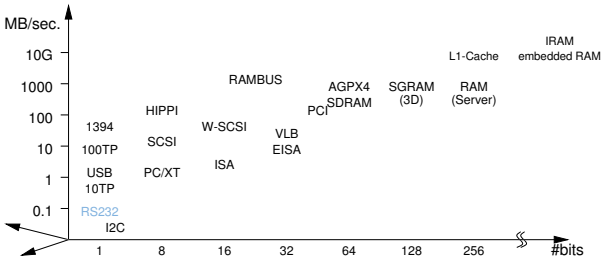
- n-Signale plus Steuersignale
- erfordert Arbitrierung

Bus: Timeline



Bus: Taxonomie

- Protokoll — Speicherzugriff vs. messages
- Datenübertragung — seriell / parallel (#bits)
- Datenrate, Bandbreite — # bit / sec.
- Anzahl der Geräte — # master, # slave
- Arbitrierung — single / multi master
  round-robin, ...



Leerseite

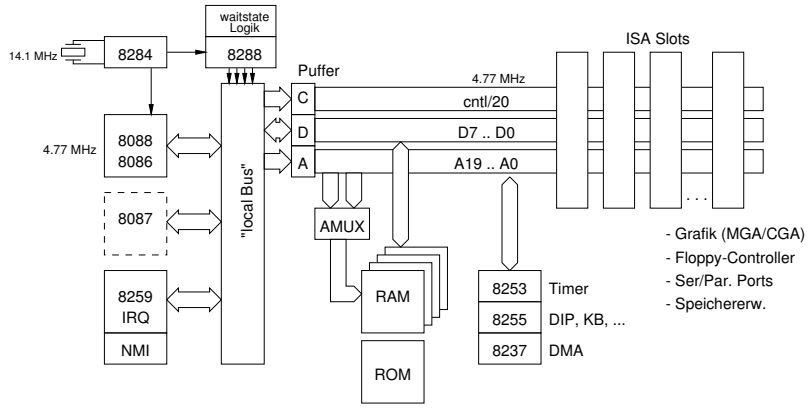
Bus: Peripherie

Anforderungen:	Bytes / sec.	Anschluss an:
• Speicher (PC800)	1.600.000.000	separat
• 3D (AGPx4)	1.000.000.000	AGP / PCI
• Video (800x600x16bx50)	48.000.000	AGP / PCI
• Festplatte	20.000.000	PCI (IDE/SCSI)
• Netzwerk (100TP)	10.000.000	PCI
• Audio (Synth 64x48Kx24b)	9.000.000	PCI
• Audio (CD: 2x44Kx16b)	176.000	PCI / ISA
• Floppy	100.000	ISA
• Modem	> 10.000	ISA / USB
• Terminal (25x80 Zeichen)	2.000	USB / ser.
• Maus	150	USB / ser.
• Tastatur	< 20	USB / ser.

=> Hierarchie von Bussen

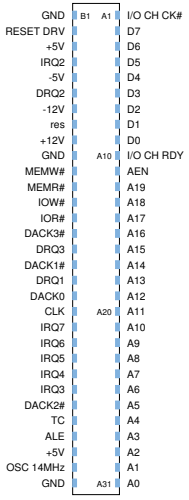
Leerseite

ISA: PC/XT



- x86 mit diversen Support-Chips
- ein gemeinsamer Bus für alle Komponenten

ISA: PC/XT Slot / Signale



RESET DRV	buffered reset
CLK/OSC	4.77 MHz / 14.1 MHz
ALE	"address latch enable"
AEN	DMA aktiv
DRQ1..3	DMA request
DACK1..3	DMA acknowledge
TC	DMA end (terminal count)
I/O CH CK#	error indication
I/O CH RDY	insert wait cycle
IOR#/IOW#	I/O cycle
MEMR#/MEMW#	memory cycle
	total: 62 Leitungen

ISA: PC/XT Eigenschaften

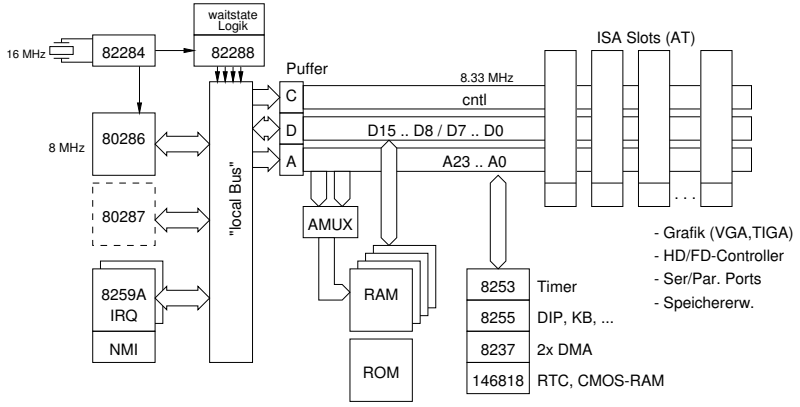
Original-IBM PC:

- Intel 8088, 4.77 MHz (Turbo-Versionen bis 10 MHz)
- real-mode, 1 MB Adressraum
- nutzt alle verfügbaren Support-Chips

- ein gemeinsamer Bus
- 20 bit Adressen (1MB), 8 bit Daten, diverse Steuerleitungen
- RAM / ROM mit am zentralen Bus
- RAM-Refresh über Timer und DMA
- 8 Interrupt-Quellen, 3 DMA-Kanäle frei

- weitere Peripherie (Grafik!) über Slots
- nur CPU und DMA als Busmaster

ISA: PC/AT



- 286 mit passenden Support-Chips
- gemeinsamer Bus, 8/16-bit Transfers

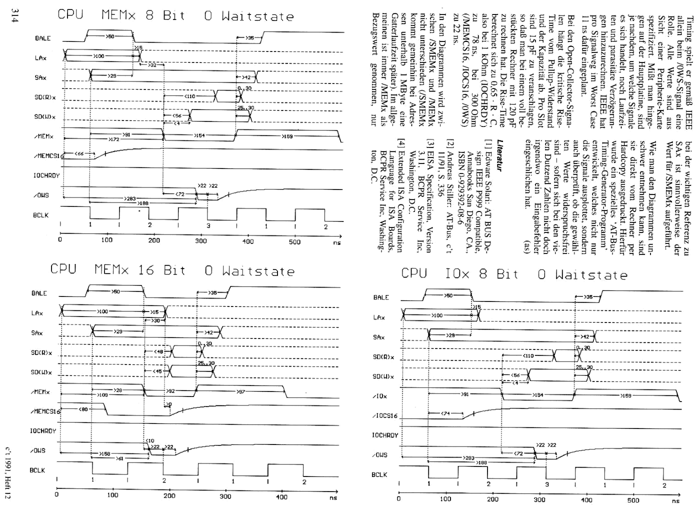
ISA: PC/AT Eigenschaften

- 80286/80287-Prozessor, plus passende Support-Chips
- 16-bit Daten, 24-bit Adressen
- real-mode oder protected-mode

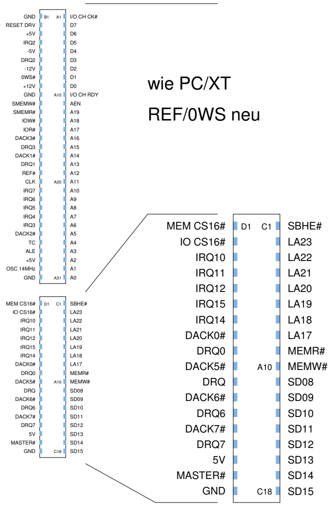
- neue Slots, abwärtskompatibel für 8-bit XT-Karten
- eingeschränktes Busmastering möglich
- max. Bustakt 8.33 MHz ("ISA Standard")

- 15 Interrupt-Kanäle
- insgesamt 7 DMA-Kanäle, davon 4x 8-bit, 3x 16-bit

ISA: PC/AT read/write, 0 wait



ISA: PC/AT Slot / Signale



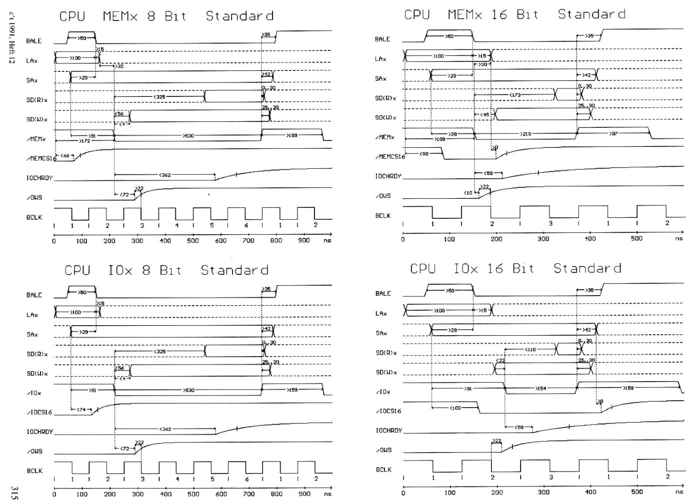
wie PC/XT
REF/0WS neu

AT-Bus: XT-Bus + 36 neue Kontakte
XT Karten passen weiterhin

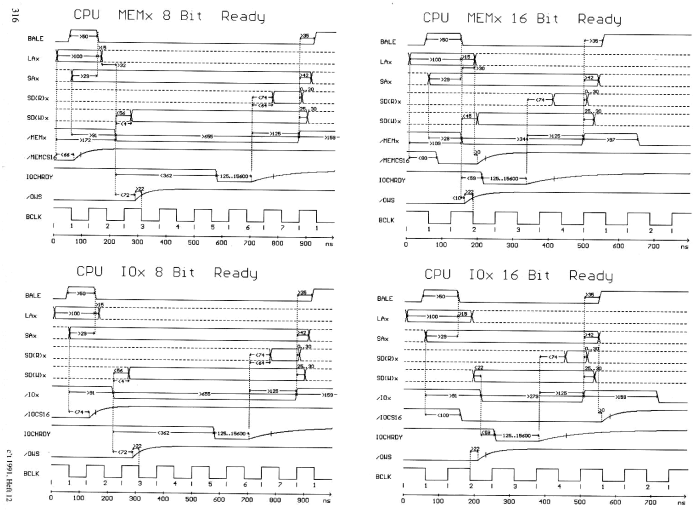
obere 8 Datenleitungen SD08..SD15
weitere Adressleitungen
neue DMA/IRQ-Leitungen

0WS (B8):	0 Wartezyklen nötig
REF (B19):	Speicherrefresh (dack0)
SBHE:	system bus high enable
IRQ1x:	Interrupt-Inputs
DRQ/DACK:	DMA request / acknoledge
Master:	Busmaster-Anforderugn
SMEM/MEM:	Zugriffe 0..1M, 1M ..16M

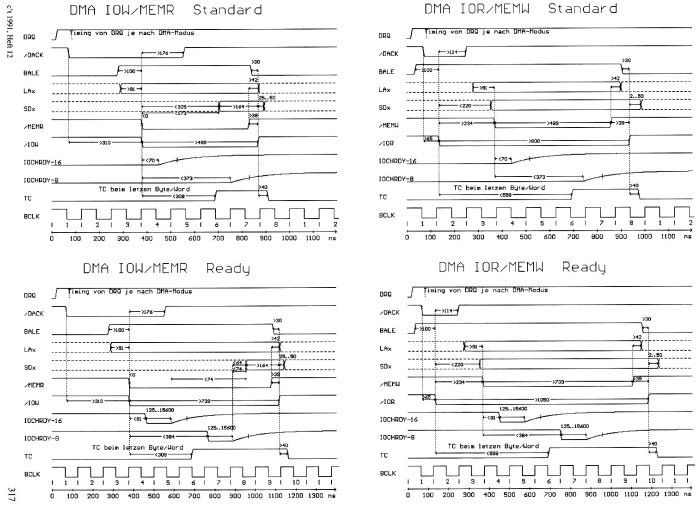
ISA: PC/AT read/write



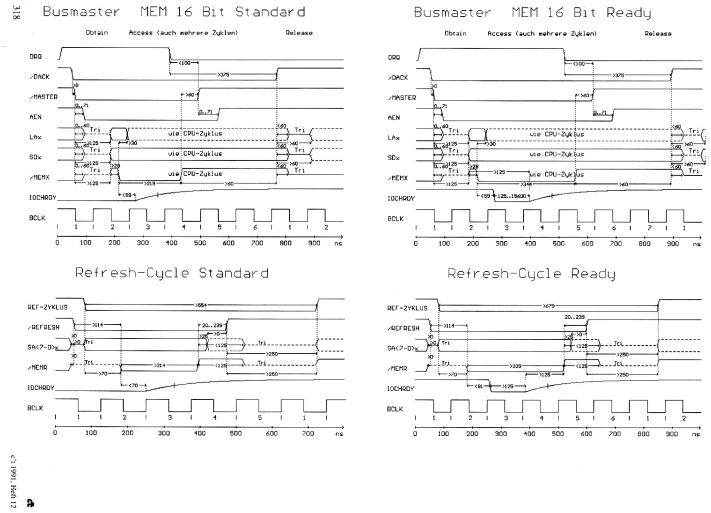
ISA: PC/AT read/write (ready)



ISA: PC/AT DMA-Zyklen



ISA: PC/AT Busmaster-Zyklen



Leerseite

ISA: Plug and Play

- ISA-Karten benutzerkonfiguriert: Jumper
- viele Hersteller, keine einheitliche Konfiguration
- beschränkte Ressourcen (I/O, IRQ, DMA)

=> häufig Probleme durch Konflikte
=> Abhilfe durch Autokonfiguration der Karten

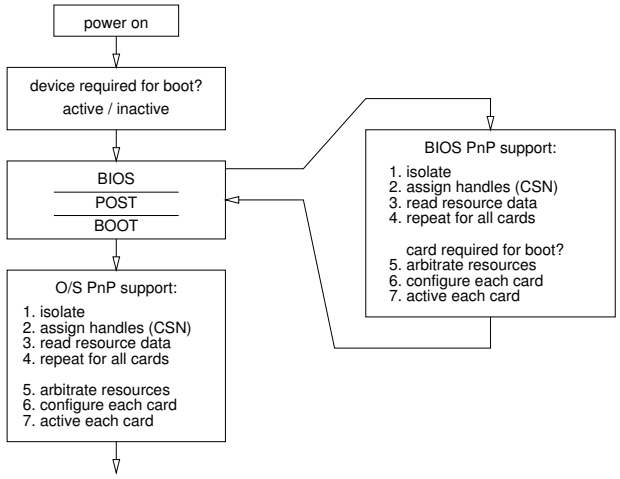
Plug and Play ISA Specification:

- automatische Erkennung von PnP-Karten
- Konfiguration durch PnP-BIOS und PnP-OS (=Windows 9x)
- entwickelt von Microsoft und Intel, 1993-1994

www.roestock.demon.co.uk/isapnptools/

www.microsoft.com/hwdev/respec/pnpspecs.htm

ISA: Plug and Play: Sequenz



ISA: Plug and Play

essentielle Funktionen des ISA-PnP:

- Erkennung der PnP-Karten (isolate)
- Auslesen der Konfigurationsdaten (identify)
- Setzen gültiger Konfigurationsdaten
- Laden geeigneter Kartentreiber

Vorteile:

- PnP-Karten sind kompatibel zu alten ISA-Karten
- keine Änderung des PCs / Motherboards notwendig
- System voll autokonfigurierbar, solange nur PnP-Karten
- "ease of use for the end user"

PnP: I/O-Ports

Three 8-bit ports are used by the software to access the configuration space on each Plug and Play ISA card. The ports are listed in table 1. The configuration space is implemented as a set of 8-bit registers. These registers are used by the Plug and Play software to issue commands, check status, access the resource data information, and configure the Plug and Play hardware.

The ports have been chosen so as to avoid conflicts in the installed base of ISA functions, while at the same time minimizing the number of ports needed in the ISA I/O space.

Table 1. Auto-configuration Ports

Port Name	Location	Type
ADDRESS	0x0279 (Printer status port)	Write-only
WRITE_DATA	0x0A79 (Printer status port + 0x0800)	Write-only
READ_DATA	Relocatable in range 0x0203 to 0x03FF	Read-only

The ADDRESS and WRITE_DATA ports are located at fixed addresses. The WRITE_DATA port is located at an address alias of the ADDRESS port. All three auto-configuration ports use a 12-bit ISA address decode.

The READ_DATA port is relocatable within the I/O range from 0x0203h to 0x03FFh. This is the only readable auto-configuration port.

ISA: Plug and Play Isolate

Isolate: Erkennen der PnP-Karten auf dem ISA-Bus

- drei I/O-Adressen:

ADDRESS	0x0279	write
WRITE_DATA	0x0A79	write
READ_DATA	0x0203 .. 0x03FF	read

- initiation:
  write 0x00, 0x00
  write initiation key (LFSR)
- isolation:
  repeat
  isolate one card
  if (ISA conflict) relocate READ_DATA
  until (no more cards found)
- identify

PnP: Hardware-Protocol

3.3.1. Hardware Protocol

The isolation protocol can be invoked by the Plug and Play software at any time. The initiation key, described earlier, puts all cards into configuration mode. The hardware on each card expects 72 pairs of I/O read accesses to the READ_DATA port. The card's response to these reads depends on the value of each bit of the serial identifier which is being examined one bit at a time, in the sequence shown in figure 5.

If the current bit of the serial identifier is a „1“, then the card will drive the data bus to 0x55 to complete the first I/O read cycle. If the bit is „0“, then the card puts its data bus driver into high impedance. All cards in high impedance will check the data bus during the I/O read cycle to sense if another card is driving D[1:0] to „01.“ During the second I/O read, the card(s) that drove the 0x55, will now drive a 0xAA. All high impedance cards will check the data bus to sense if another card is driving D[1:0] to „10.“

If a high impedance card sensed another card driving the data bus with the appropriate data during both cycles, then that card ceases to participate in the current iteration of card isolation. Such cards, which lose out, will participate in future iterations of the isolation protocol.

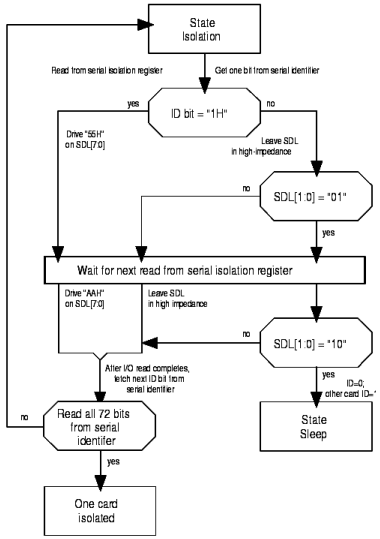
NOTE:	*During each read cycle, the Plug and Play hardware drives the entire 8-bit data bus, but only checks the lower 2 bits.*

If a card was driving the bus or if the card was in high impedance and did not sense another card driving the bus, then it should prepare for the next pair of I/O reads. The card shifts the serial identifier by one bit and uses the shifted bit to decide its response.

The above sequence is repeated for the entire 72-bit serial identifier.

At the end of this process, one card remains. This card is assigned a handle referred to as the *Card Select Number* (CSN) that will be used later to select the card. Cards which have been assigned a CSN will not participate in subsequent iterations of the isolation protocol. Cards must be assigned a CSN before they will respond to the other commands defined in the specification.

PnP: Isolation-Protocol



findet ein pnp-Gerät pro Iteration

(höchste ID zuerst)

entsprechend oft wiederholen

erkennt Konflikte mit nicht-pnp Karten

bei Konflikt den Read-Port wechseln

PnP: Software-Protocol

3.3.2. Software Protocol

The Plug and Play software sends the initiation key to all Plug and Play cards to place them into configuration mode. The software is then ready to perform the isolation protocol.

The Plug and Play software generates 72 pairs of I/O read cycles from the READ_DATA port. The software checks the data returned from each pair of I/O reads for the 0x55 and 0xAA driven by the hardware. If both 0x55 and 0xAA are read back, then the software assumes that the hardware had a „1“ bit in that position. All other results are assumed to be a „0.“

During the first 64 bits, software generates a checksum using the received data. The checksum is compared with the checksum read back in the last 8 bits of the sequence.

There are two other special considerations for the software protocol. During an iteration, it is possible that the 0x55 and 0xAA combination is never detected. It is also possible that the checksum does not match. If either of these cases occur on the first iteration, it must be assumed that the READ_DATA port is in conflict. If a conflict is detected, then the READ_DATA port is relocated. The above process is repeated until a non-conflicting location for the READ_DATA port is found. The entire range between 0x200 and 0x3FF is available, however in practice it is expected that only a few locations will be tried before software determines that no Plug and Play cards are present.

During subsequent iterations, the occurrence of either of these two special cases should be interpreted as the absence of any further Plug and Play cards (i.e. the last card was found in the previous iteration). This terminates the isolation protocol.

PnP: Architektur der Steckkarten

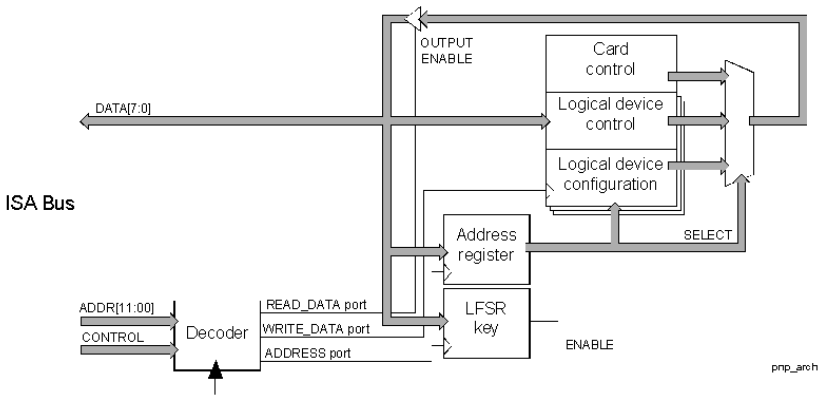


Figure 3. Logic Flow for Auto-configuration

- diverse Konfigurationsregister, spezielles LFSR
- ansonsten "normale" ISA-Karte

Leerseite

PnP: Configuration space

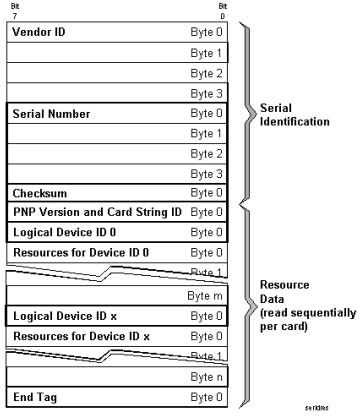


Figure 7. Serial Identifier and Resource Data

- Details siehe ISAPNP-Spezification

Leerseite

PCI: Motivation

ISA-Bus:

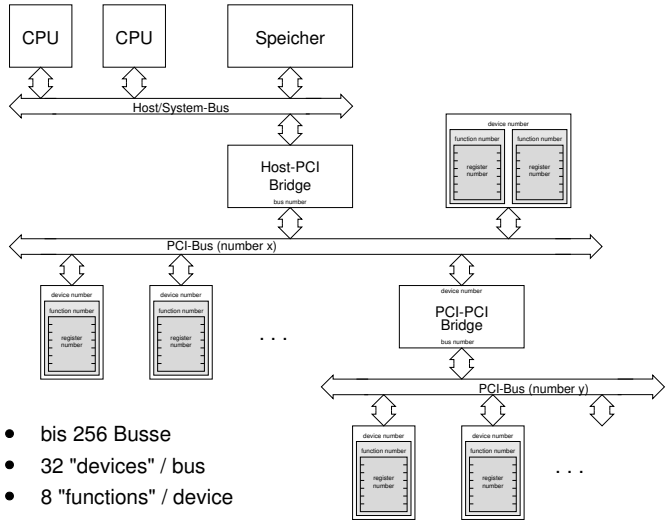
- zu langsam für Video und 3D
- Autokonfiguration problematisch
- kein effizientes Busmastering

=> neuer Bus erforderlich
=> aber Platzbedarf und Kosten ähnlich wie ISA

PCI: "Peripheral Component Interconnect", 1992:

- 32-bit Bus mit Option auf 64-bit
- Bandbreite 132 MB/sec @ 33 MHz
- weitgehende Autokonfiguration
- entwickelt, patentiert, freigegeben von Intel
- PCI Special Interest Group, www.pcisig.org

PCI: Hierarchie mit mehreren Bussen



- bis 256 Busse
- 32 "devices" / bus
- 8 "functions" / device

PCI: Übersicht

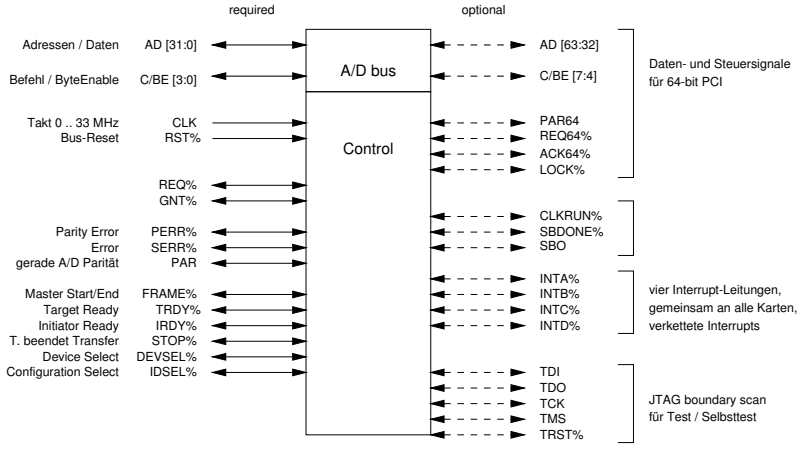
- universelles Bussystem
- ausreichend schnell für fast alle Anwendungen
- diverse Varianten (33/66 MHz, 32/64 Bit, 3.3/5 V, PCI-X)

- volle Autokonfiguration aller Devices
- flexibles Busmaster- und Interrupt-Konzept
- Burst-Transfers für hohe Bandbreite
- kaskadierbar über PCI-PCI-Bridges

- seit ca. 1994 in allen PC-Chipsätzen integriert
- teilweise direkt als Prozessorkomponente, etwa UltraSPARC IIi
- erstaunlicher Markterfolg, über 800 Hersteller
- hat fast alle proprietären Busse verdrängt

- eingeführt 1992, Version 2.0 seit 1993, derzeit 2.1

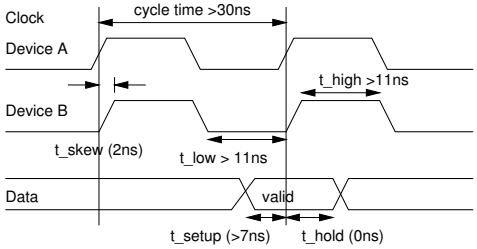
PCI: Signale



- A/D multiplex, insgesamt 120 / 184 Pins

PCI: Takt

- Taktfrequenz: 33 MHz (Standard) / 66 MHz (Option)

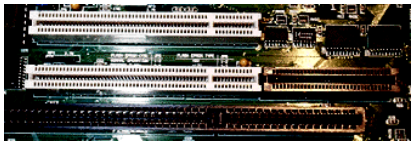


- jeweils ein Takt für Adress/Daten/Wait-Zyklus
- niedrigere Frequenzen sind erlaubt
- PC-Motherboards: PCI-Takt meistens an Systemtakt gekoppelt
- Übertakten problematisch

PCI: Stecker

- Varianten:

3.3V / 5V Signale	Position des Stegs im Stecker
33 MHz / 66 MHz	Signal M66EN auf GND/VCC
32 Bit / 64 Bit	einfacher / langer Stecker



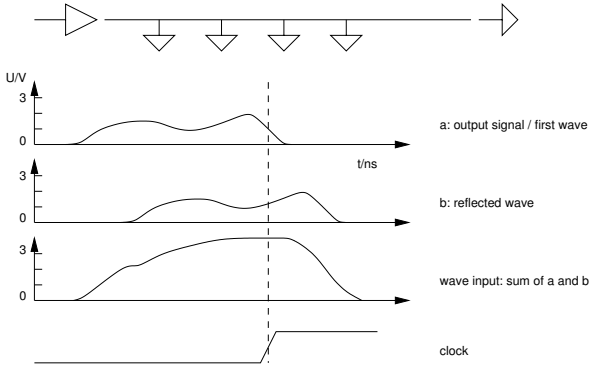
PCI, 5V, 32 bit

PCI, 5V, 64 bit

ISA, 16 bit

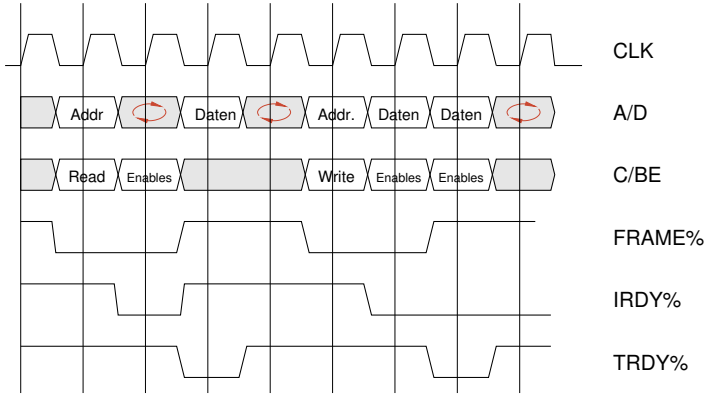
- 66 MHz nur dann, wenn alle Karten es erlauben
- narrensicher

PCI: "reflected waves"



- nutzt Reflektionen am Ende des Busses ...
- Platinenlayout kritisch, kurze Leitungen
- besondere Eingangsschaltungen in den PCI-Geräten
- kein statischer Stromverbrauch (anders als SCSI)

PCI: Transfer



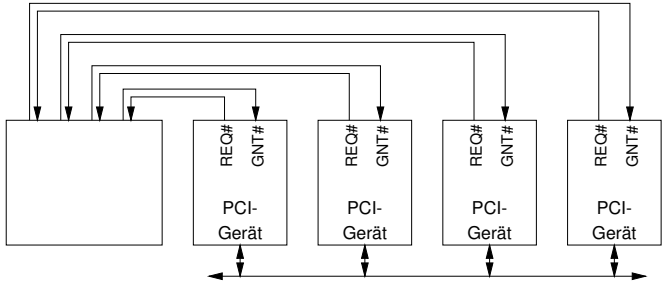
- FRAME signaliert Start und Ende eines Transfers
- je ein Warte-Takt ($) zur Umschaltung Read/Write
- Transfer beendet wenn TRDY & IRDY

PCI: Befehle / Transfers

C/BE 3	2	1	0	
0	0	0	0	Interrupt Acknowledge
0	0	0	1	Special Cycle (broadcast, data contains message type)
0	0	1	0	I/O Read
0	0	1	1	I/O Write
0	1	0	0	reserved
0	1	0	1	reserved
0	1	1	0	Memory Read
0	1	1	1	Memory Write
1	0	0	0	reserved
1	0	0	1	reserved
1	0	1	0	Configuration Read
1	0	1	1	Configuration Write
1	1	0	0	Memory Read Multiple
1	1	0	1	Dual Address Cycle
1	1	1	0	Memory Read Line
1	1	1	1	Memory Write And Invalidate

- I/O: Byte-Adressen, keine Bedingungen (Nebenwirkungen)
- Memory: 4-Byte Adressen, Lesen liefert zuletzt geschriebenen Wert

PCI: Arbitrierung



- ein zentraler Arbiter (normalerweise im Chipsatz)
- Anforderung mit REQ, Erlaubnis mit GNT
- gilt für eine Transaktion beliebiger Burst-Länge
- Arbiter kann GNT entziehen, um Transaktion zu beenden

PCI: PCI-BIOS

- PCI definiert auch Standard-Programmierschnittstelle
- als Menge von x86 Software-Interrupts
- als "Gast" (Erweiterung) des Uhren-Interrupts 1Ah
- PCI 2.1 definiert 12 Funktionen, Übergabe per AL-Register

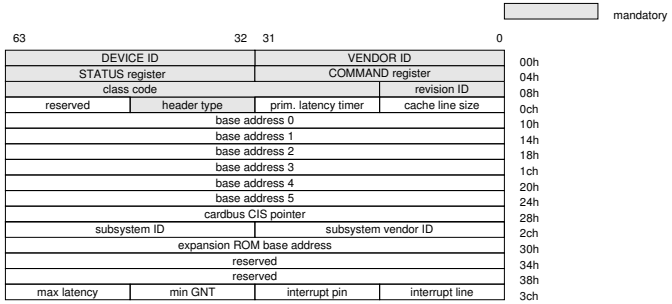
```
boolean PCI_BIOS_Present() {
  int AX = 0xB101; // "magic" PCI 2.1 defined function number
  interrrupt( 0x1A );
  boolean pci_bios_present = (AH==0) && (DX==0x4350);
  int number_of_busses     = CL;
  int pci_version          = BX;
  int config_version       = AL;

  return pci_bios_present;
}
```

PCI: PCI-BIOS Funktionen

- PCI definiert auch Standard-Programmierschnittstelle
- Beispielprogramme (Turbo-Pascal) z.B. in [ct 02-96-266ff]

```
PCI_BIOS_present
find_PCI_device
find_PCI_class_code
generate_special_cycle                (shutdown, halt, x86 )
read_configuration_xxx
write_configuration_xxx
get_PCI_interrupt_routing_options
set_PCI_IRQ
. . .
```

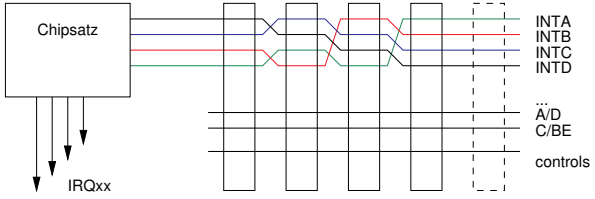
PCI: Configuration Space



256 Byte (64 DWORD) Configuration Space pro PCI Gerät:

- eindeutige Hersteller / Geräte-ID
- Status- und Befehlsregister
- Identifikation der Karten erlaubt Autokonfiguration

PCI: Interrupts



- Chipsatz setzt INTA .. INTD auf IRQxx um
- jede Karte darf alle Interrupts benutzen

Interrupt-Handler "chaining"

- Handler sucht nach auslösendem Gerät
- nicht immer perfekt implementiert :-(
- oft nur INTA benutzt: "krumme" Verdrahtung = weniger Konflikte
- manchmal hilft "Karten umstecken"

PCI: Interrupt Chaining

```
/* irq handler with irq chaining - from AMCC S5933 databook */

void interrupt handler( void )
{
  byte  status;

  /* read AMCC S5933 chip status */
  status = inportb( REG_BASE_ADDR + AMCC_REG_INTCSR + 2 );

  if ((status & ANY_S5933_INT) != 0) { /* handle interrupt */
    /* disable bus mastering */
    outportb( REG_BASE_ADDR + AMCC_REG_MCSR+1, 0x11 );

    /* identify interrupt source(s) */
    if ((status & READ_TC_INT) != 0)
       /* read TC interrupt code goes here */
    if ((status & WRITE_TC_INT) != 0)
       /* write TC interrupt code goes here */
    ...

    /* clear interrupt enables */
    outportb( REG_BASE_ADDR + AMCC_REG_INTCSR+1, 0 );
    outportb( REG_BASE_ADDR + AMCC_REG_INTCSR+2, status );
  }
  else { /* not an S5933 interrupt, dispatch to next handler */
    chain_intr( oldhandler );
  }

  /* end of interrupt handler: clear in-service bits */
  ...
  outportb( 0x20, 0x60|(interrupt_line&0x07) );
}
```


AGP:

"Accelerated Graphics Port"

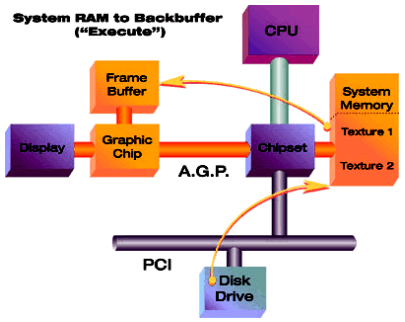
- basiert auf der 66 MHz PCI-Spezifikation
- nur zwei Geräte, Punkt-zu-Punkt Verbindung statt Bus

- pipeline-read/write Zugriffe
- daher keine Blockierung wegen Speicherlatenz
- spez. Transfermodi (Direktzugriff auf Hauptspeicher)
- separate Daten/Adressleitungen (sideband addressing)

- AGP 1x Transfers wie PCI-66
- AGP 2x "double data rate" Transfers
- AGP 4x niedrigere Spannungen, vierfache Datenrate
  Bandbreite bis 1 GB/s

- developer.intel.com

AGP: Konzept



- Chipsatz unterstützt sowohl AGP als auch PCI
- spezielle AGP-Transfers für Direktzugriff auf Hauptspeicher
- dazu Adressumsetzung im Chipsatz
- sehr komplizierte Details (siehe AGP specification)

AGP: AGP-4x Transfer

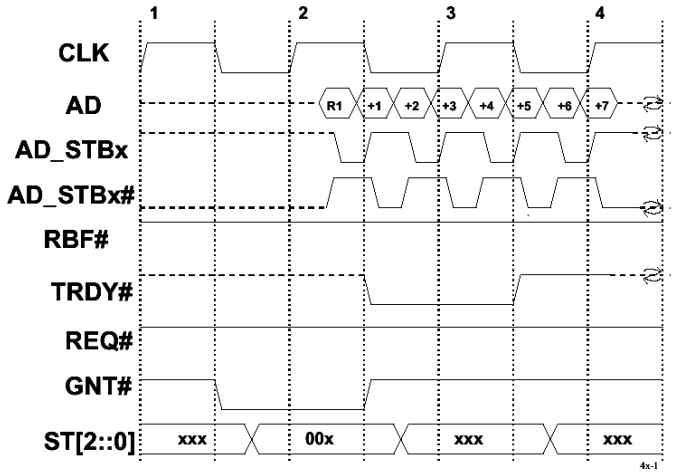
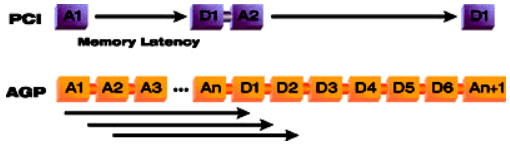


Figure 3-31: 4x Read Data - No Delay

AGP: vs. PCI



- PCI-Bus erlaubt keine "split-transactions":
- jede Transaktion muß Latenzzeiten voll abwarten

- AGP definiert Dutzende spezieller Transfermodi
- Hostrechner oder Graphikkarte als Initiator
- split-transactions zum Verdecken der Speicher-Latenzzeiten

AGP: Performance, vs. PCI

Vergleich PCI- und AGP-Grafikkarten

	3D Mark99 MAX, Racing	3D Mark99 MAX, 1st Person	Expendable/16	Expendable/32	Quake II, Crusher /16	Quake II, Crusher /32	Quake 3, q3demo1 /16	Quake 3, q3demo1 /32
	besser ▶	besser ▶	besser ▶	besser ▶	besser ▶	besser ▶	besser ▶	besser ▶
TNT-PCI-Grafik	38	44	35	20	27	21	18	14
TNT-AGP-Grafik	42	46	35	23	31	23	21	14
TNT2-PCI-Grafik	50	45	45	39	36	29	33	25
TNT2-AGP-Grafik[1]	50	44	45	39	36	–[2]	–[2]	–[2]

Alle Tests liefen unter DirectX6.1, bei 1024 × 768 Bildpunkten, 75 Hz Bildwiederholrate, 'Wait for VSync': on, Audio ein, alle Werte in Frames per Second (fps), /16: HiColor 16 Bit, /32: TrueColor 32 Bit

[1] Vergleichswerte Asus V3800 TVR mit Vsync = Off aus [7], [2] keine Vergleichswerte verfügbar

- achtfache Bandbreite von AGP4x gegenüber Standard-PCI
- Texturdaten im Hauptspeicher möglich

- wird aber von aktuellen Apps/Spielen noch nicht genutzt
- Vorteile nur bei Benchmarks / großen Texturen