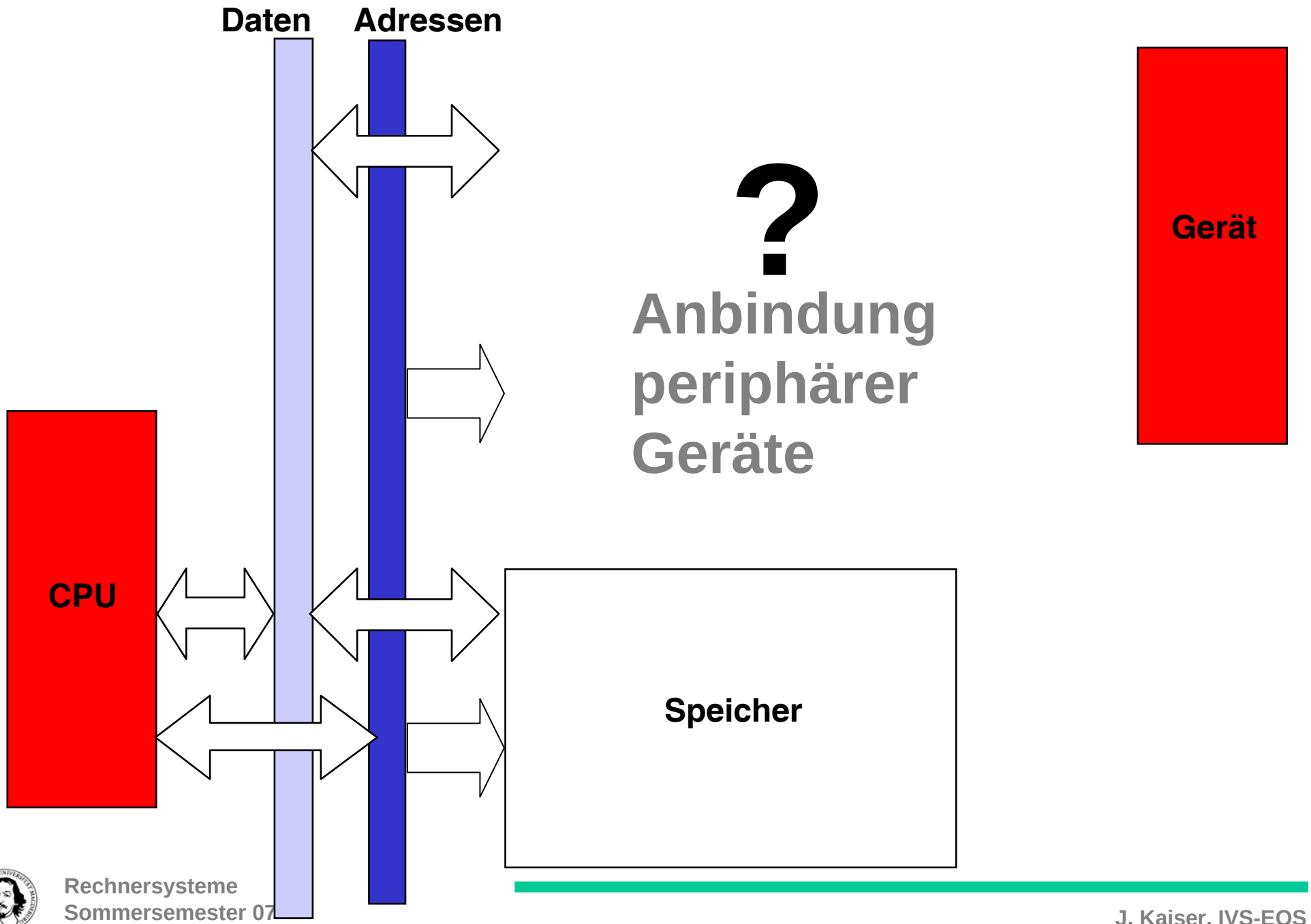
Eingabe & Ausgabe

- Generelle Struktur
- Parallele Ein- und Ausgabe
- Asynchrone, Bit-serielle Ein- und Ausgabe


Daten
Adressen
?
Anbindung
periphärer
Geräte
Gerät
CPU
Speicher

Anbindung periphärer Geräte

Problem:

Diskrepanz zwischen

- Datenrate periphärer Geräte
- Datenrate von Speicherbussen

Schritte bei der Ein-/Ausgabe

Selektion des peripheren Geräts (Adressierung)

Datentransfer zwischen peripherem Gerät und Prozessor

Charakteristik der Ein-/Ausgabegeräte

Datenrate:

- Schnelle Geräte
- Langsame Geräte

Datenübertragung

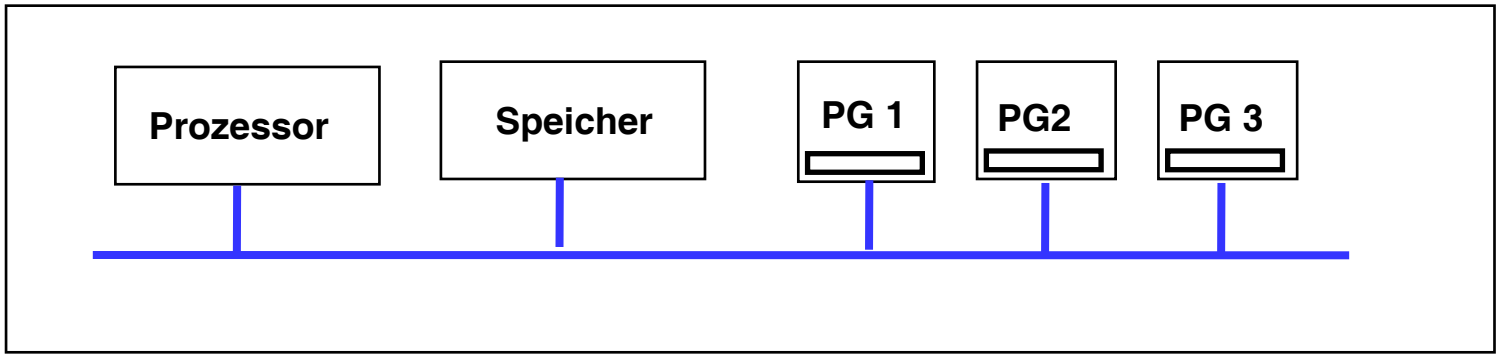
- Synchrones Protokoll
- Asynchrones Protokoll

Übertragungsart

- Bitseriell
- Byte-/Wortorientiert

Steuerung

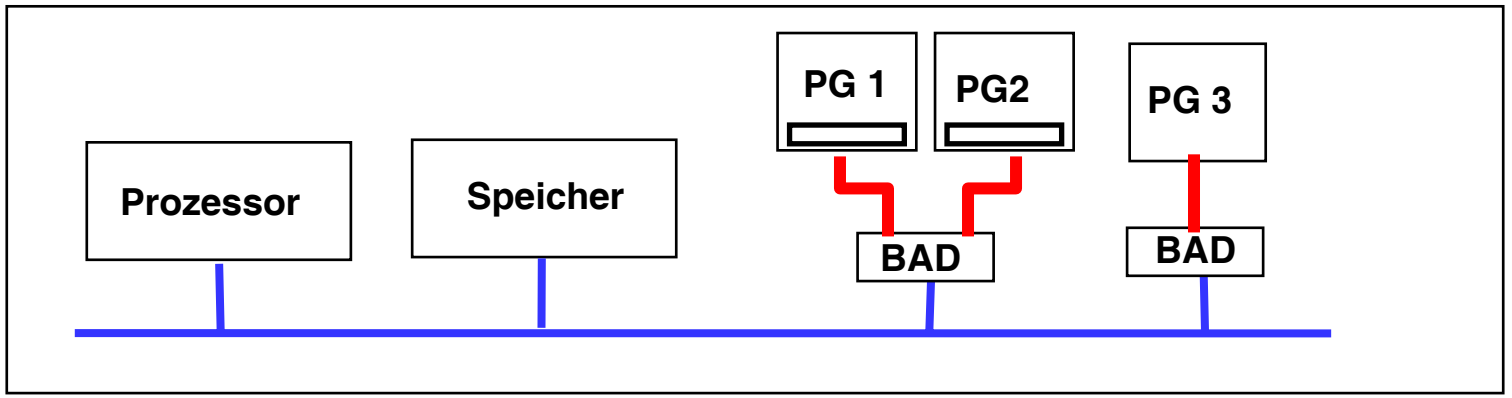
- Intelligenz im Gerät
- Intelligenz im Prozessor



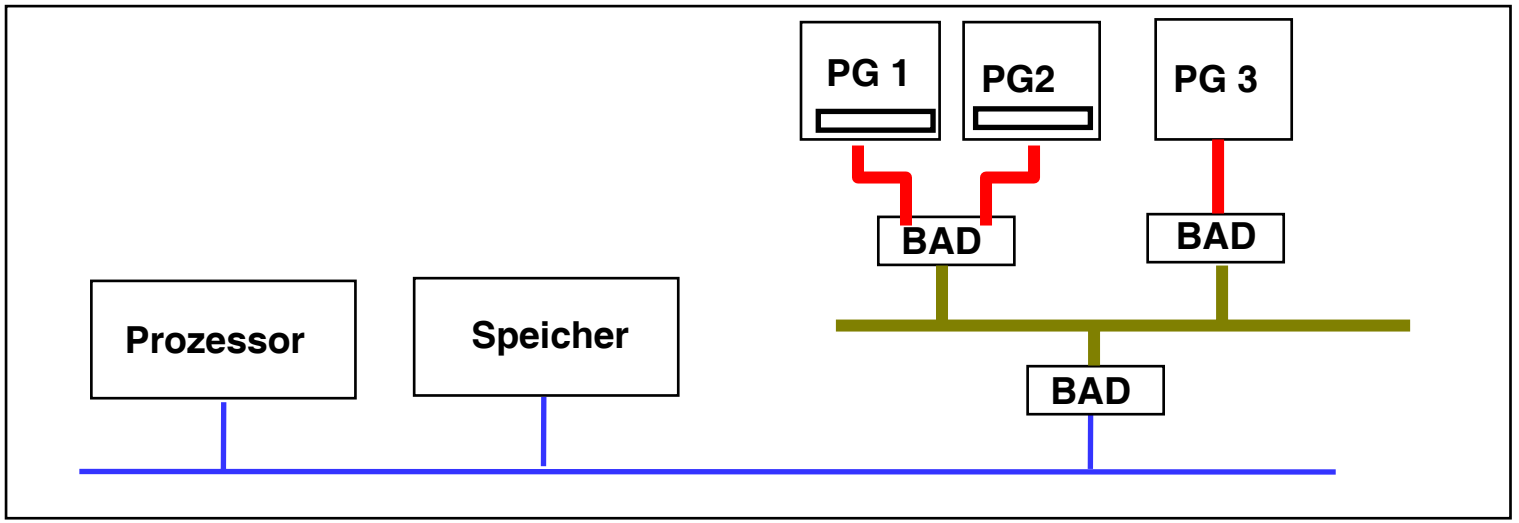
PG: Peripheres Gerät

BAD: Bus-Adapter

: Speicherbus (prozessorabhängig)

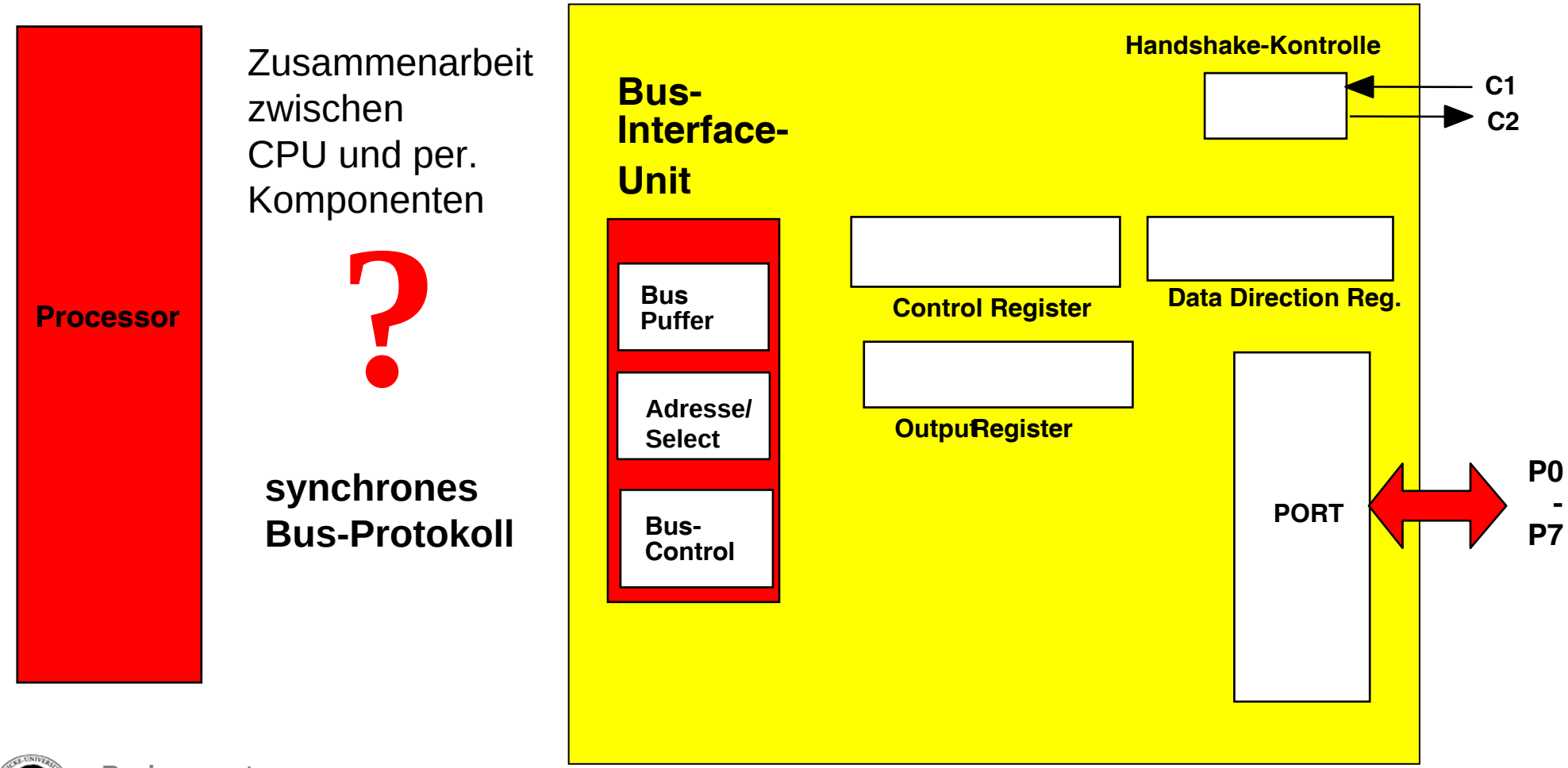


: I/O-Bus (z.B. SCSI, USB, Firewire),

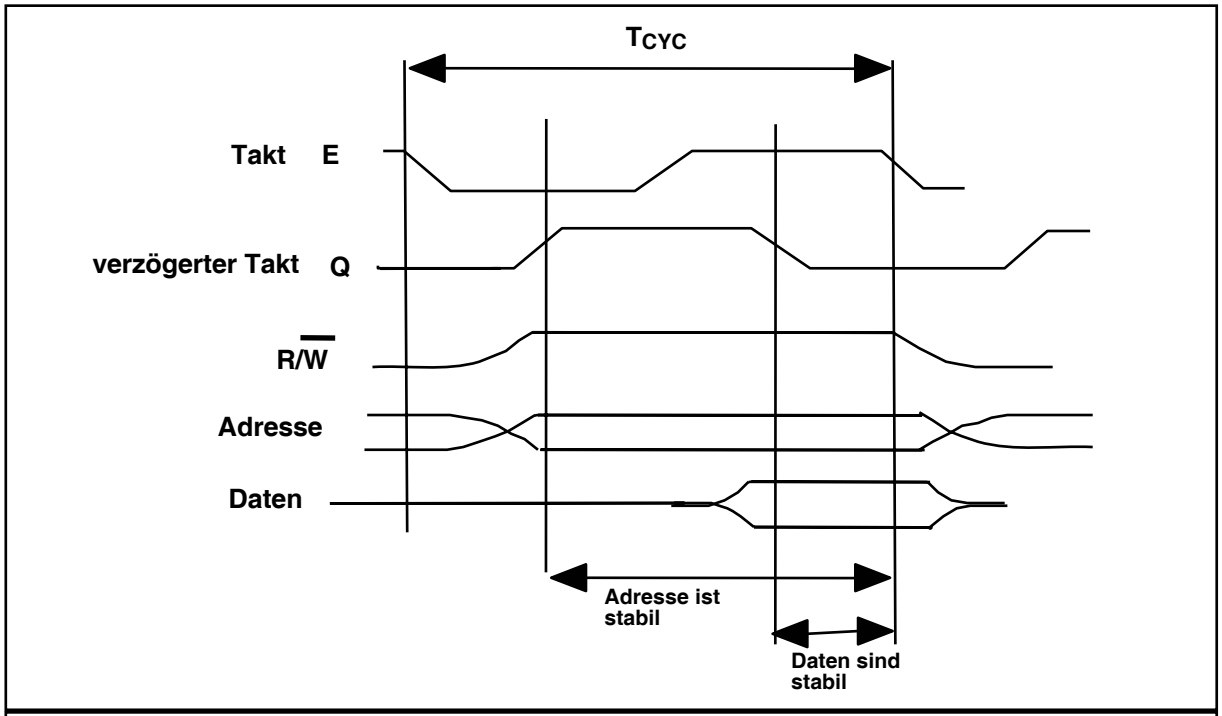


: Backplane-Bus (zB. PCI, ISA, VME, S-Bus)

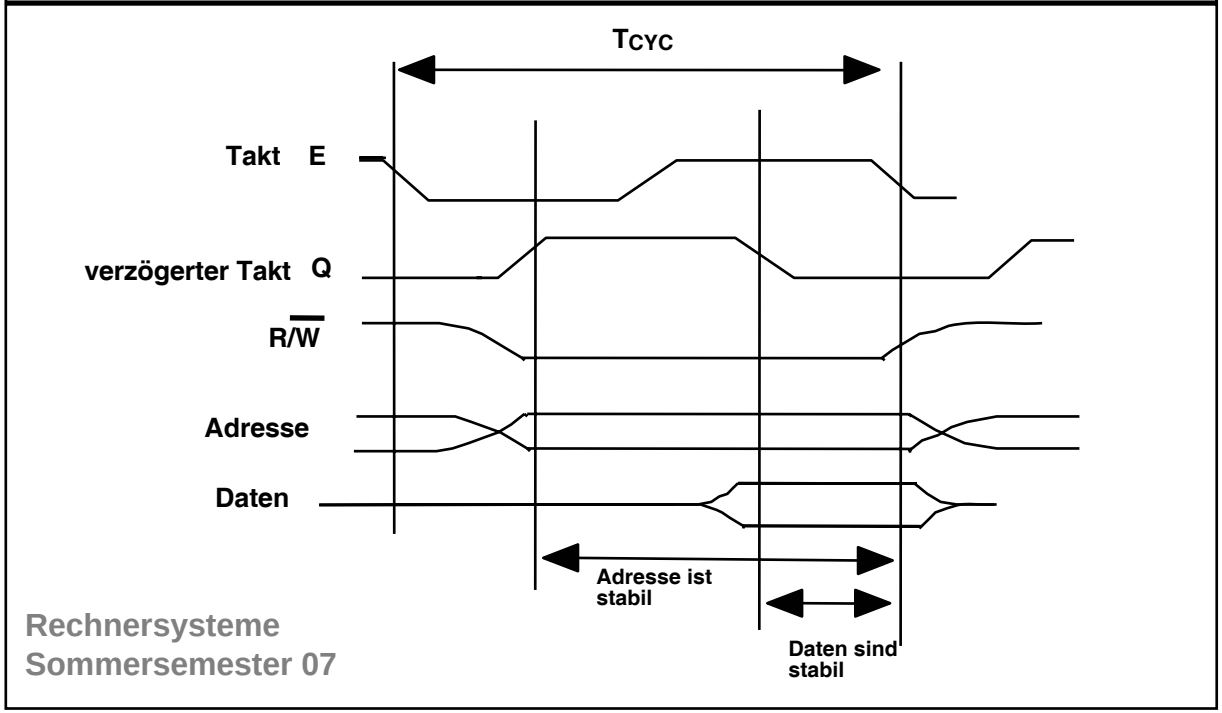
Eine einfache (Byte-parallele) Geräteschnittstelle



Einfaches, synchrones Busprotokoll des MC 6809



Lese-Zyklus



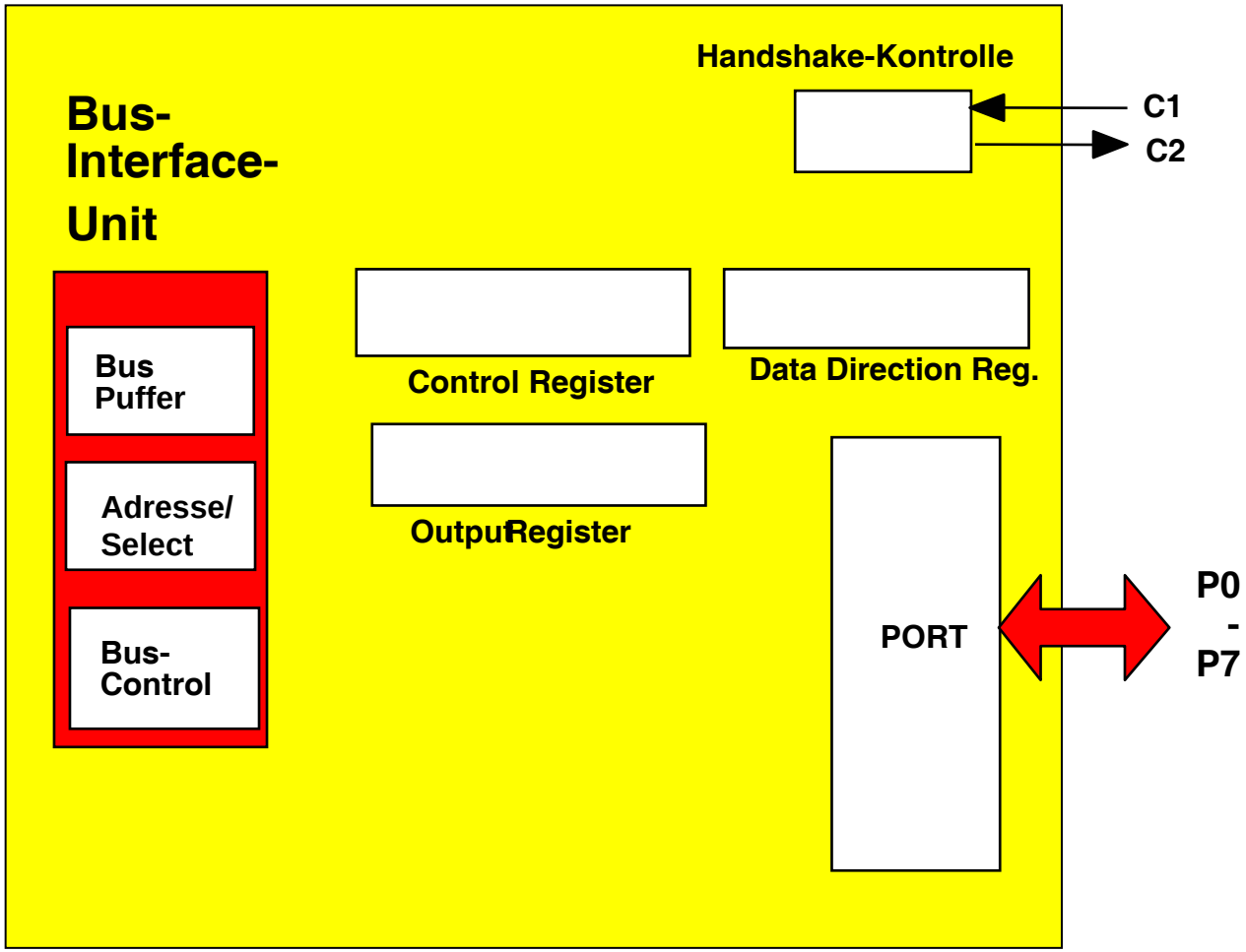
Schreib-Zyklus

Eine Byte-parallele Geräteschnittstelle

Processor

wie wird die per. Komponente vom Befehlssatz eingebunden

?

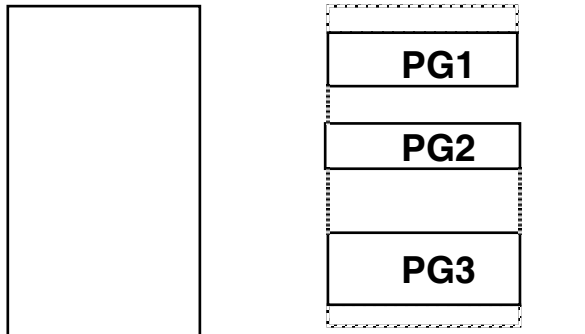


Einbindung der peripheren Gräte:

1.) Peripherie hat eigenen Adreßraum: Spezielle Ein/Ausgabe-Befehle z.B. Intel x86, OUT Befehle

2.) Adreßraum der Peripherie ist in den allgemeinen Adreßraum abgebildet: (Memory-Mapped I/O)

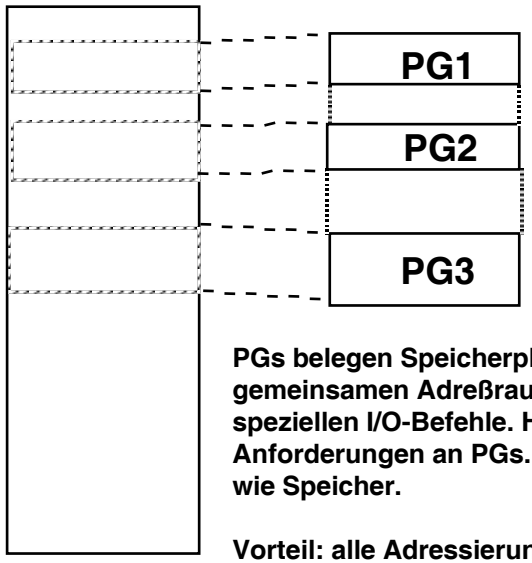
1.)



meist wesentlich kleiner als der normale Adreßraum in dem Programme ausgeführt werden. Spezielle I/O R/W-Signale zur Steuerung der PGs.

Vorteil: PGs belegen keinen allgemeinen Adreßraum, Es werden meist weniger Leitungen zur Adressierung benötigt, weil man den I/O-Adreß-raum nicht optimal nutzen muß.

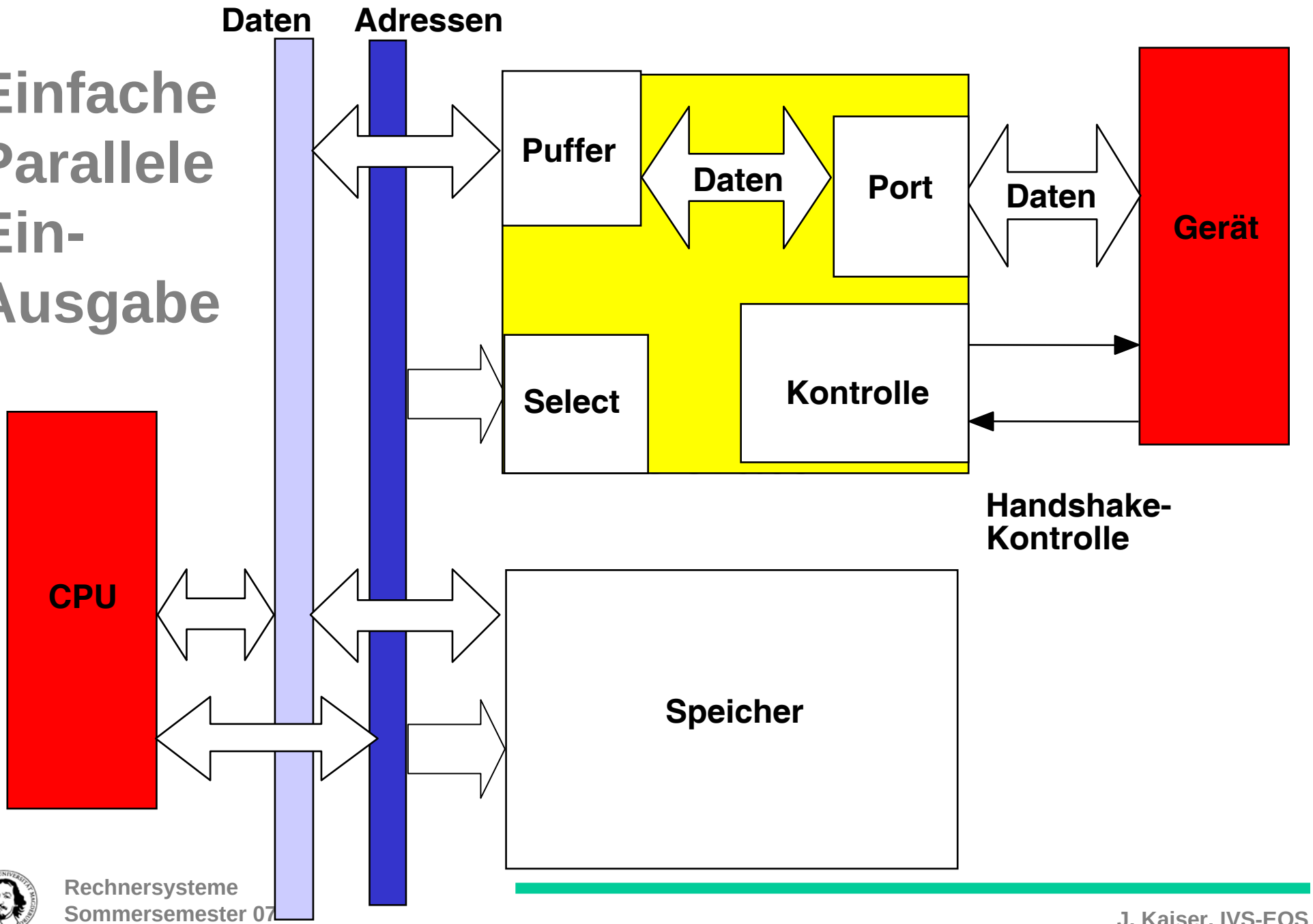
2.)



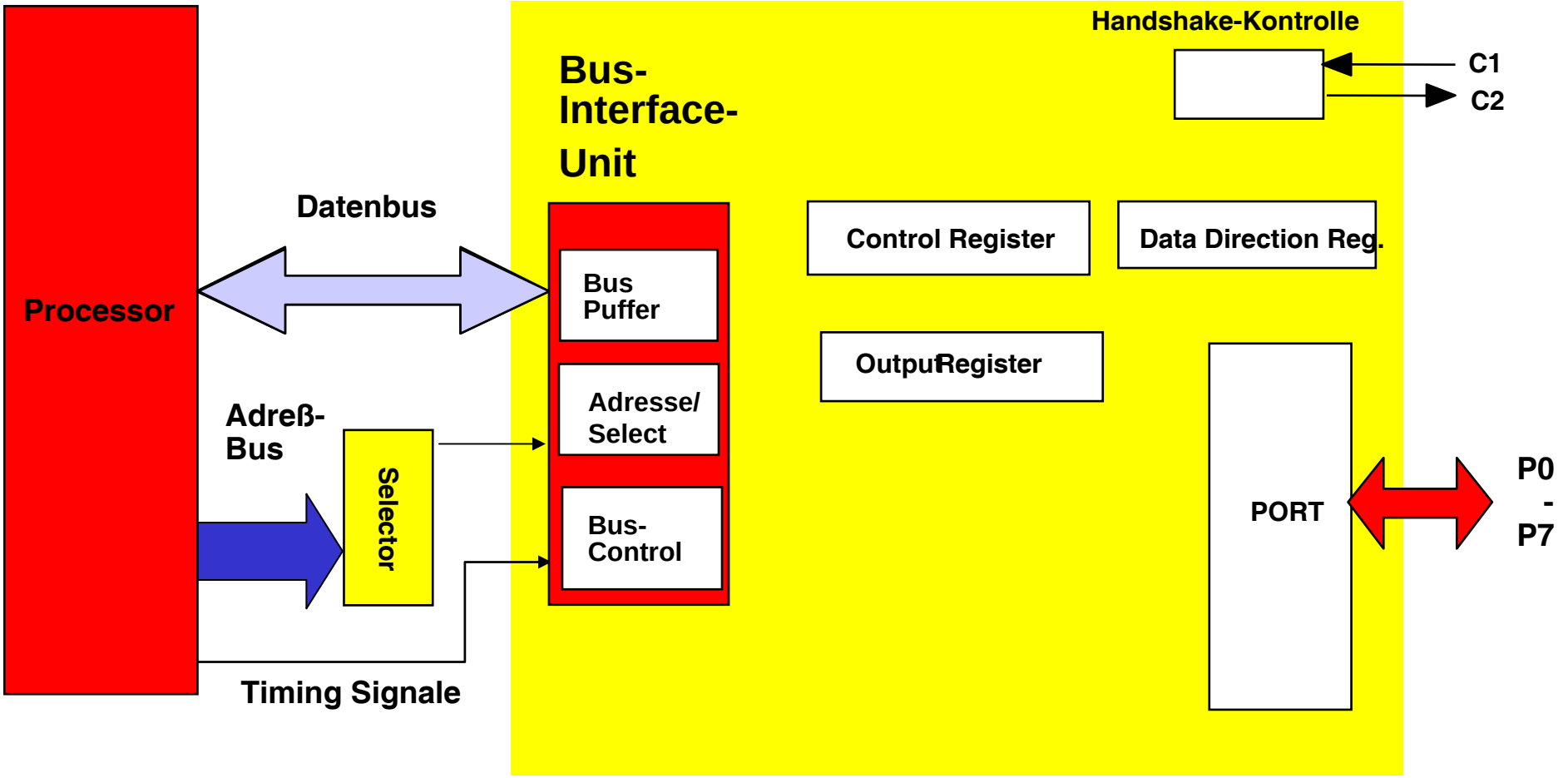
PGs belegen Speicherplatz in einem gemeinsamen Adreßraum. Keine speziellen I/O-Befehle. Höhere Anforderungen an PGs. PG vehält sich wie Speicher.

Vorteil: alle Adressierungsarten können genutzt werden, um PG zu adressieren - Befehle wie AND, OR oder TST können direkt auf den Registern der PGs operieren.

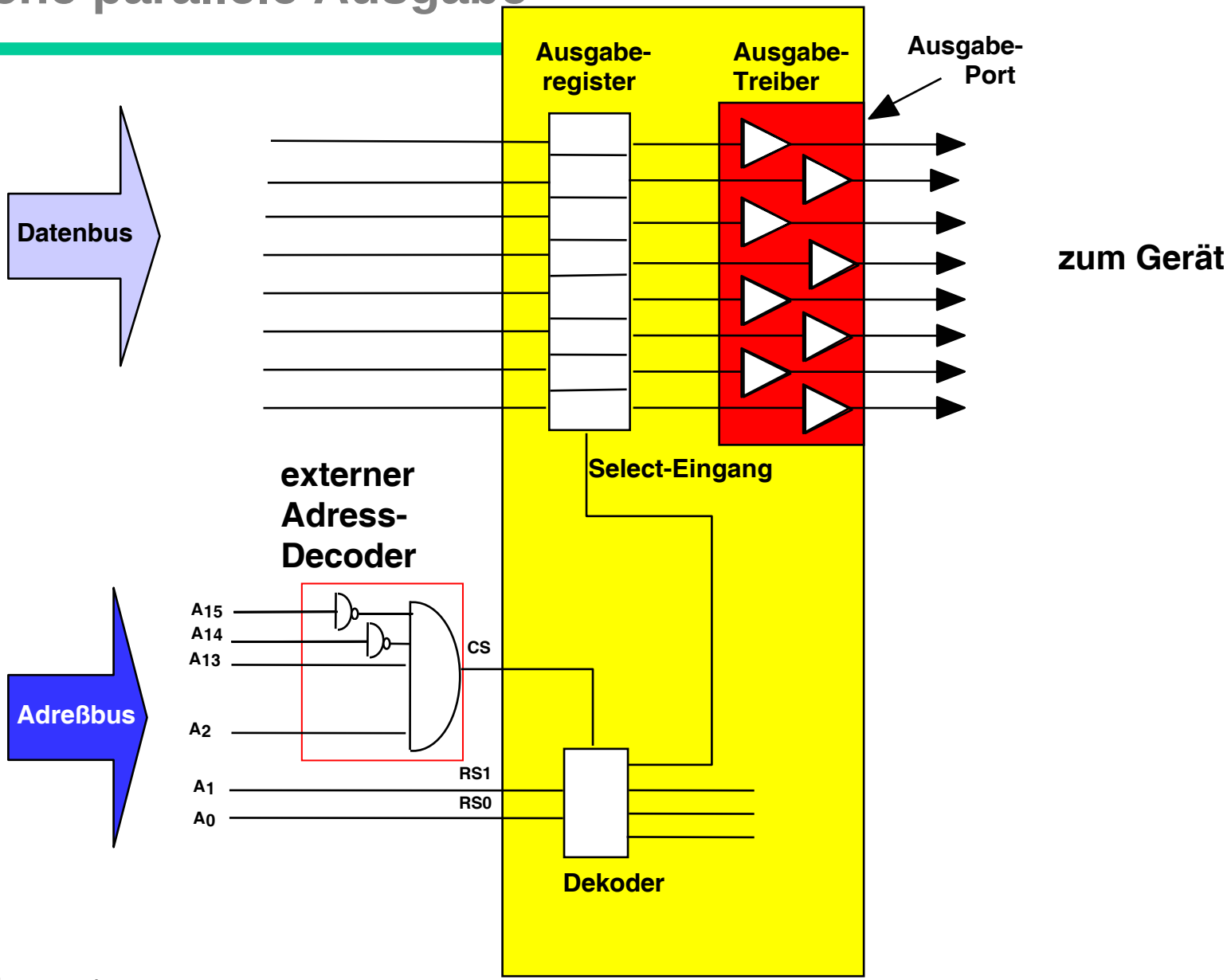
Einfache Parallele Ein-Ausgabe

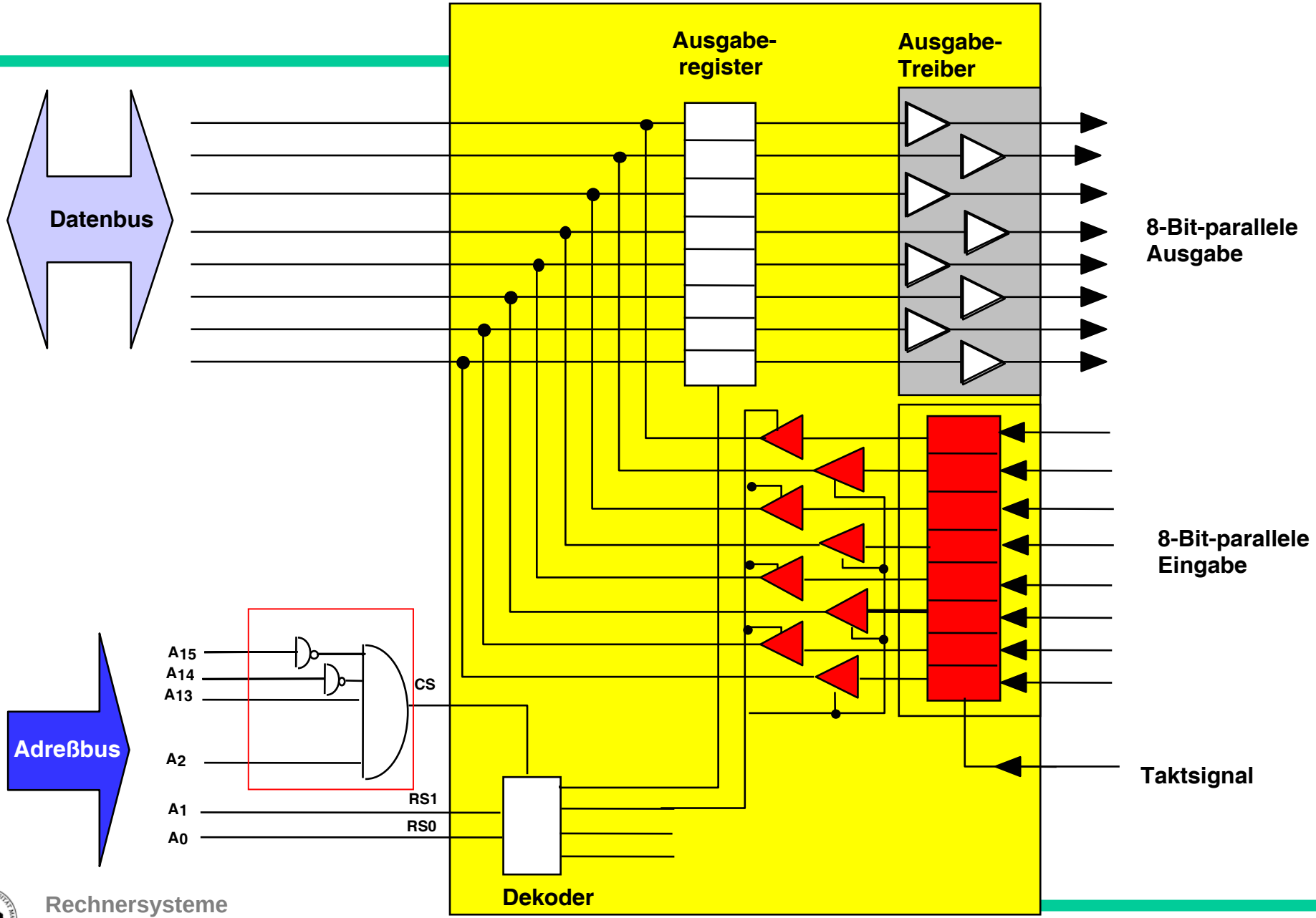


Eine Byte-parallele Geräteschnittstelle

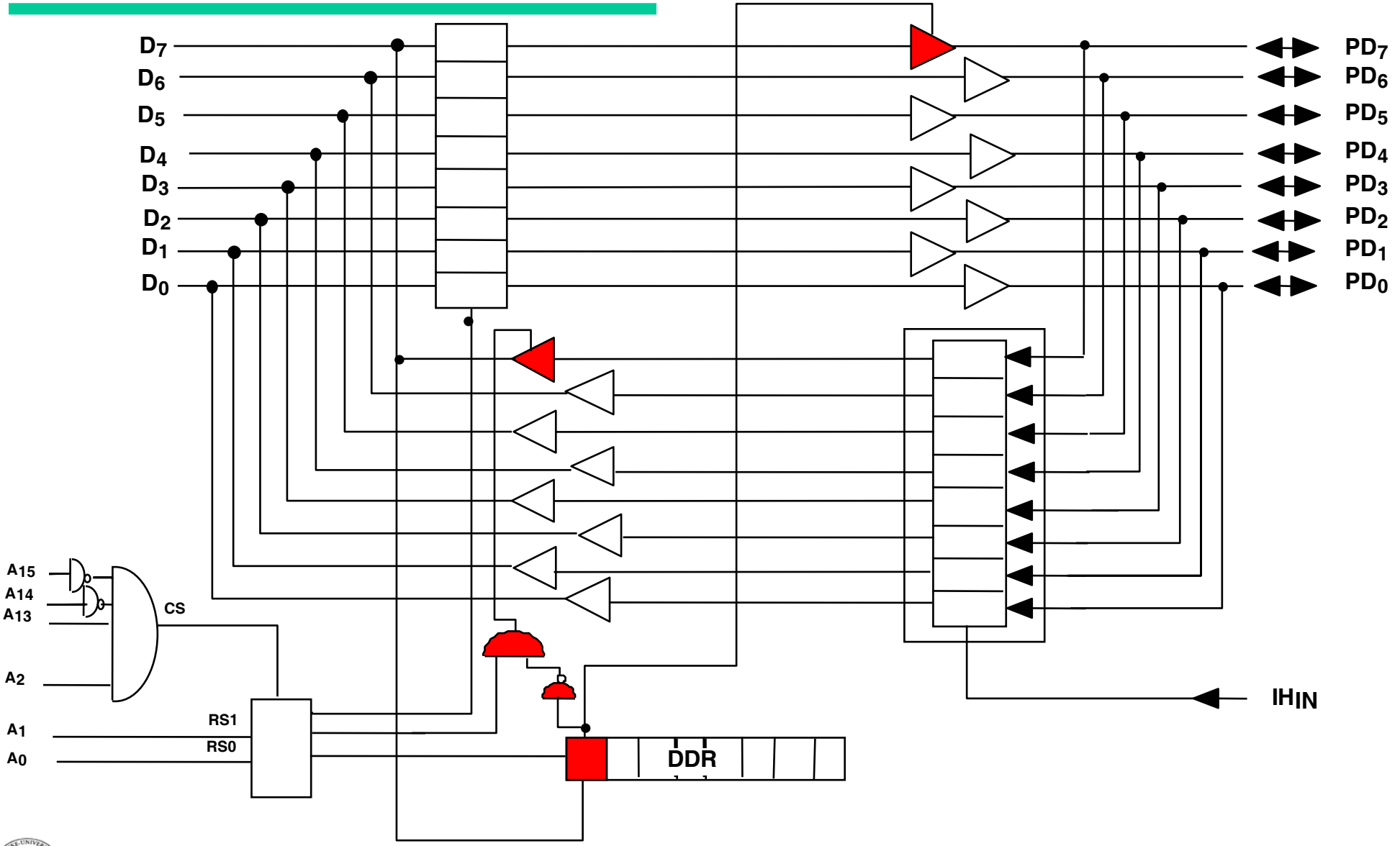


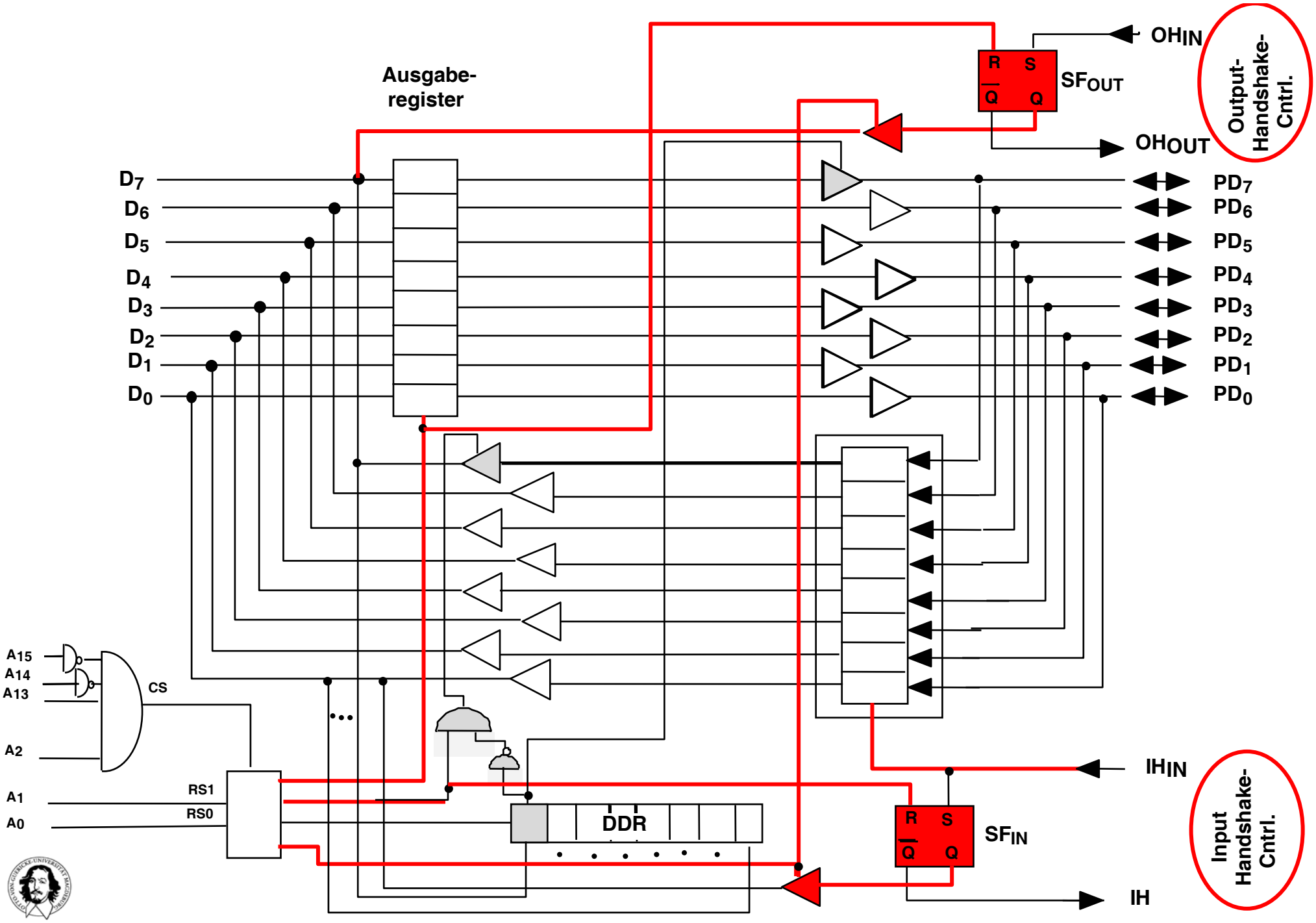
Einfache parallele Ausgabe



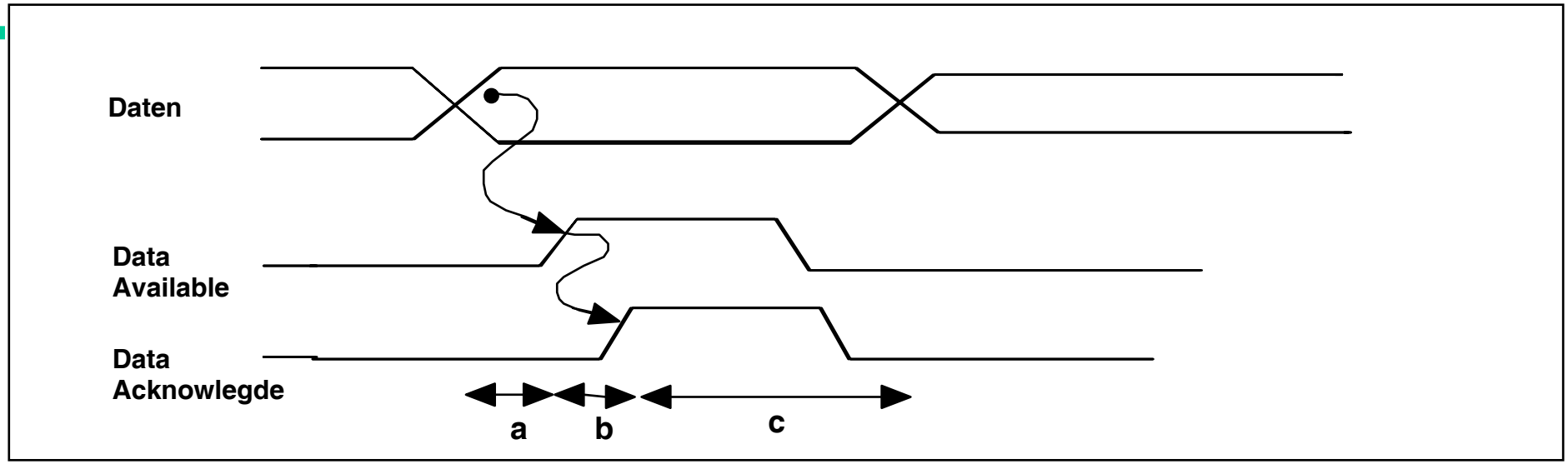

Ausgabe-
register
Ausgabe-
Treiber
Datenbus
8-Bit-parallele
Ausgabe
8-Bit-parallele
Eingabe
Taktsignal
Adreßbus
A15
A14
A13
A2
A1
A0
CS
RS1
RS0
Dekoder

Bitweise Auswahl der Richtung: Das Data Direction Register (DDR)

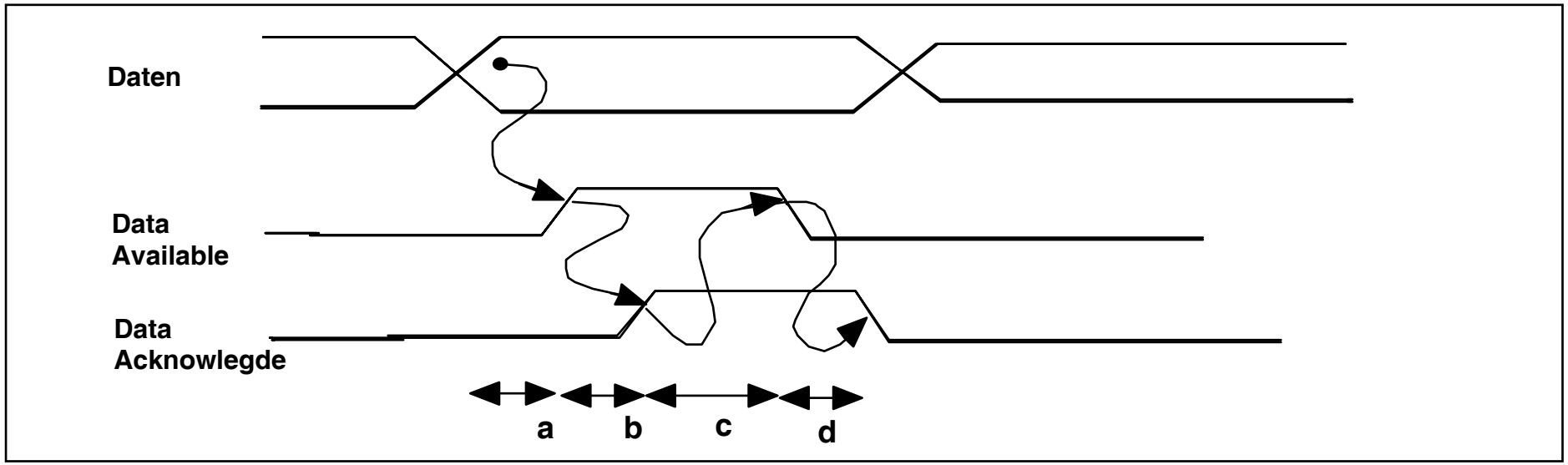



Ausgabe-
register
D7
D6
D5
D4
D3
D2
D1
D0
R
S
Q
Q
SFOUT
OHIN
OHOUT
Output-
Handshake-
Cntrl.
PD7
PD6
PD5
PD4
PD3
PD2
PD1
PD0
A15
A14
A13
CS
A2
A1
A0
RS1
RS0
DDR
SFIN
IHIN
IH
Input
Handshake-
Cntrl.

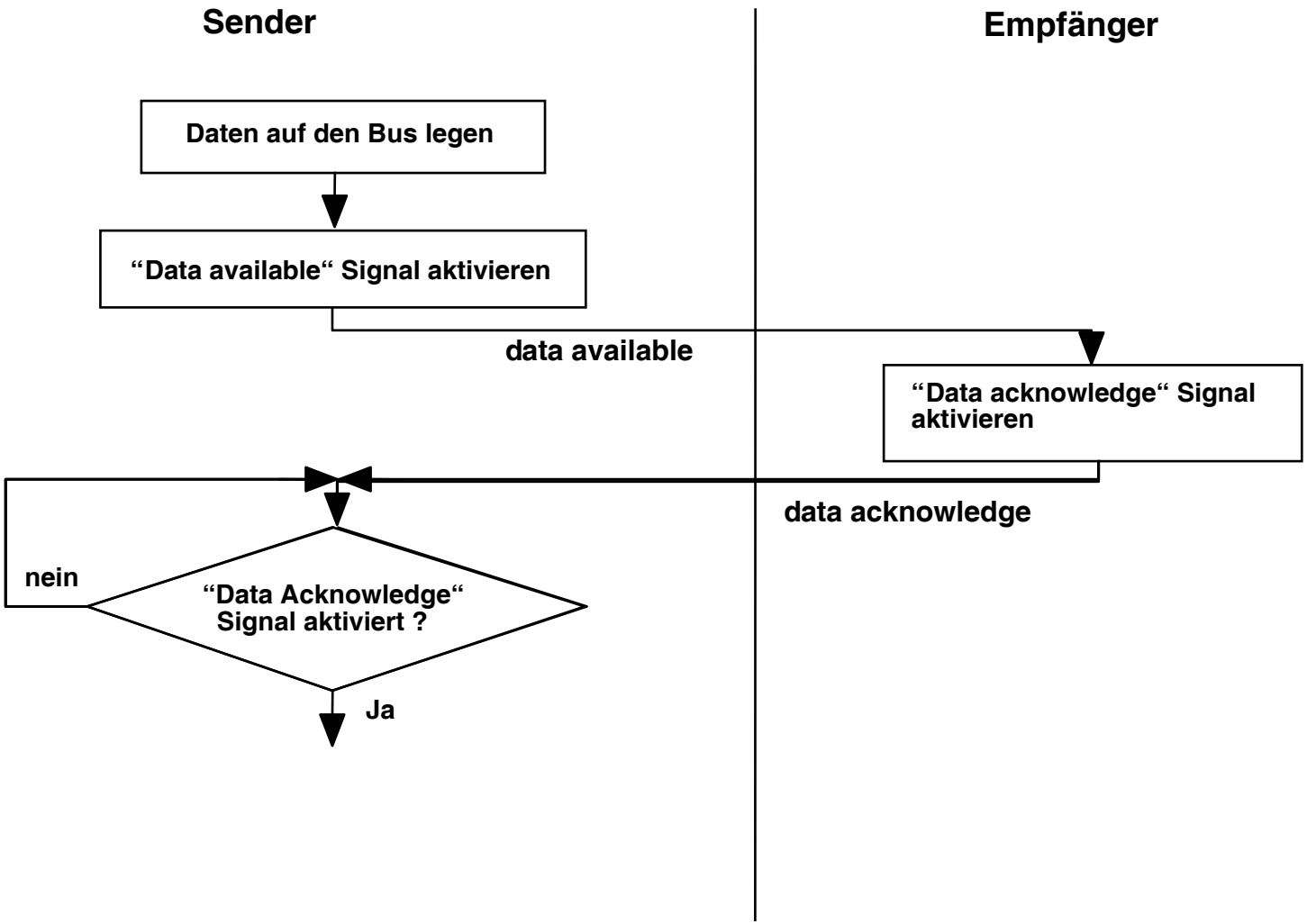
Einfacher Handshake



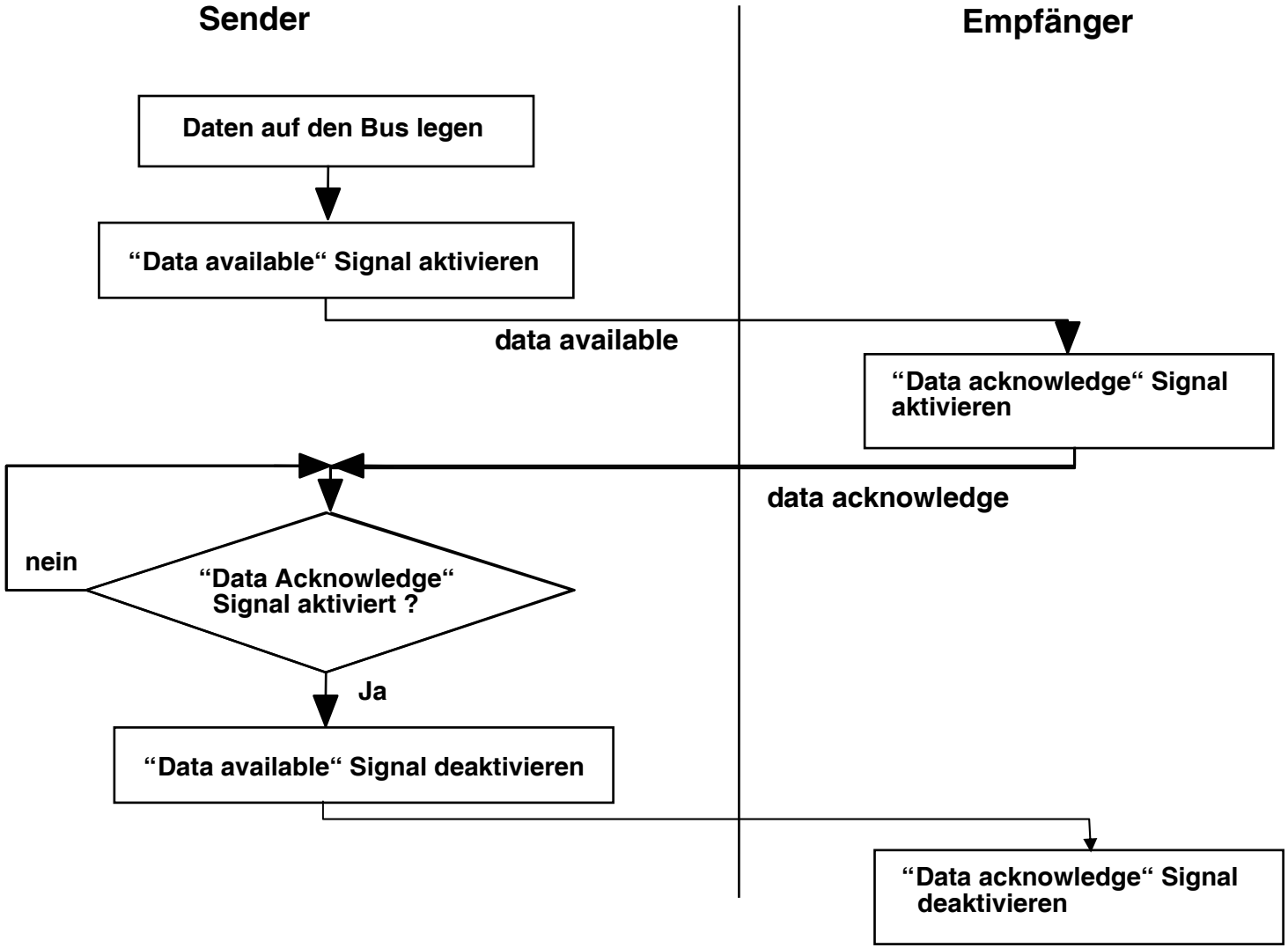
Vollständig verschränkter Handshake (fully interlocked handshake)

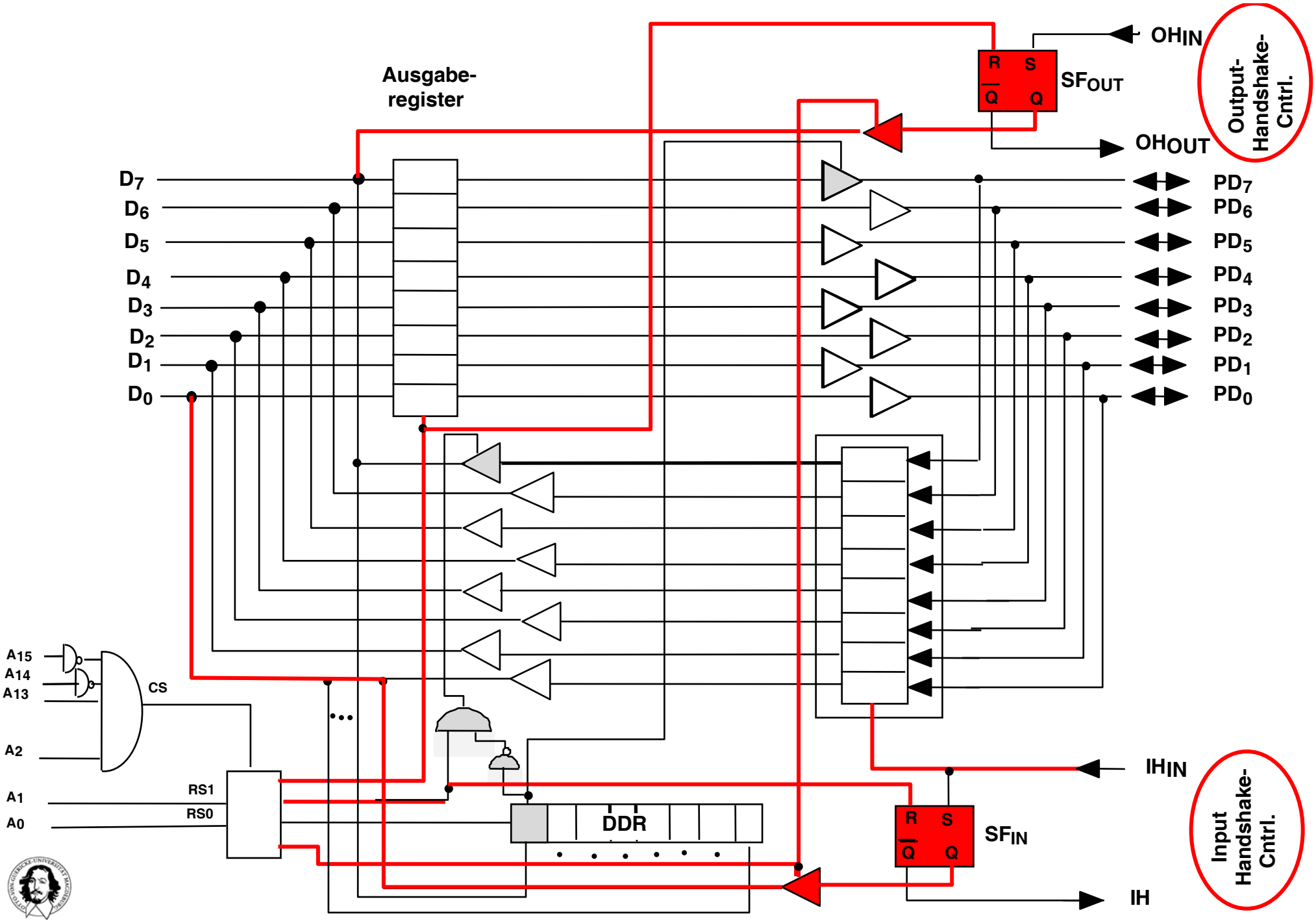


Einfacher Handshake

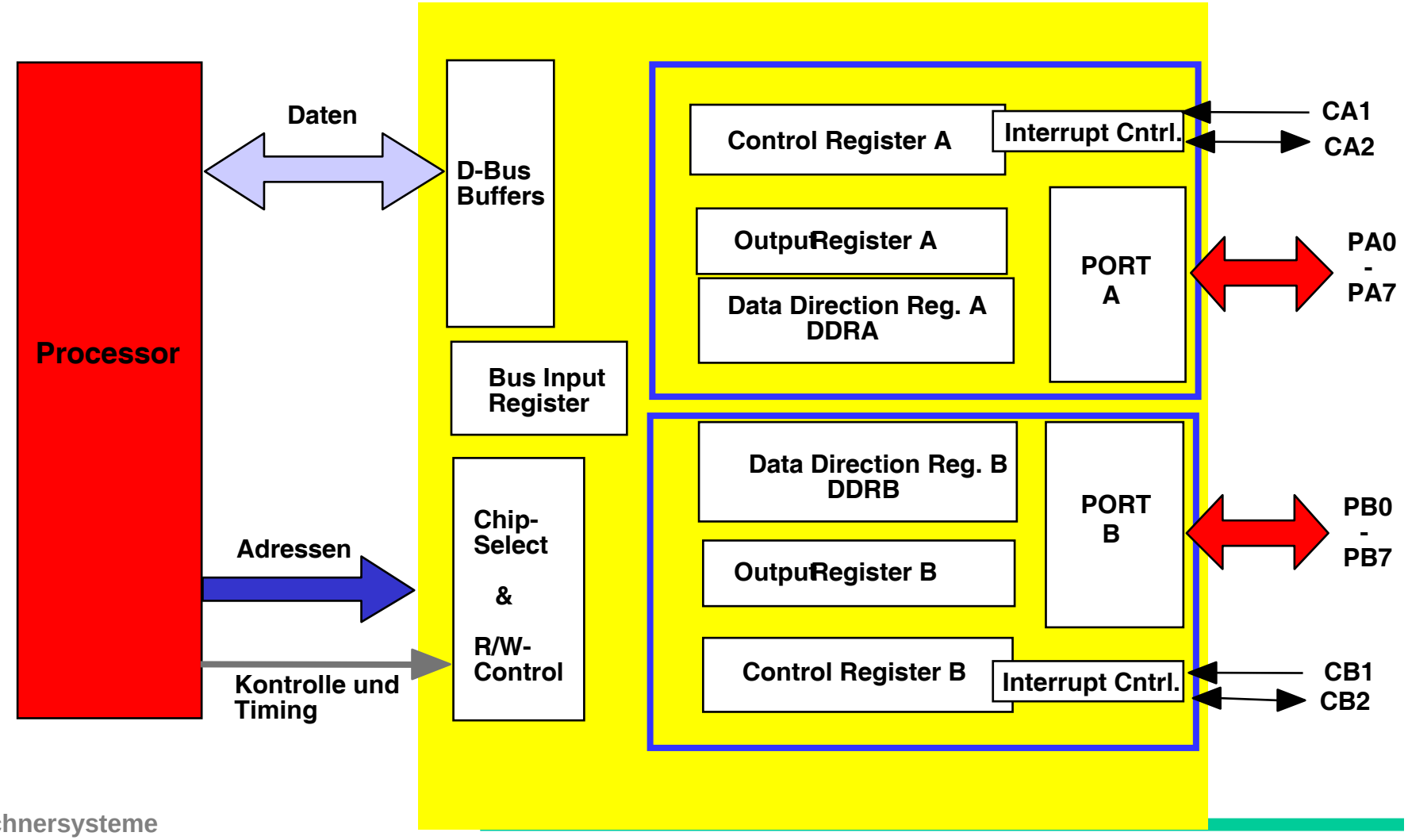


Vollständig verschränkter Handshake

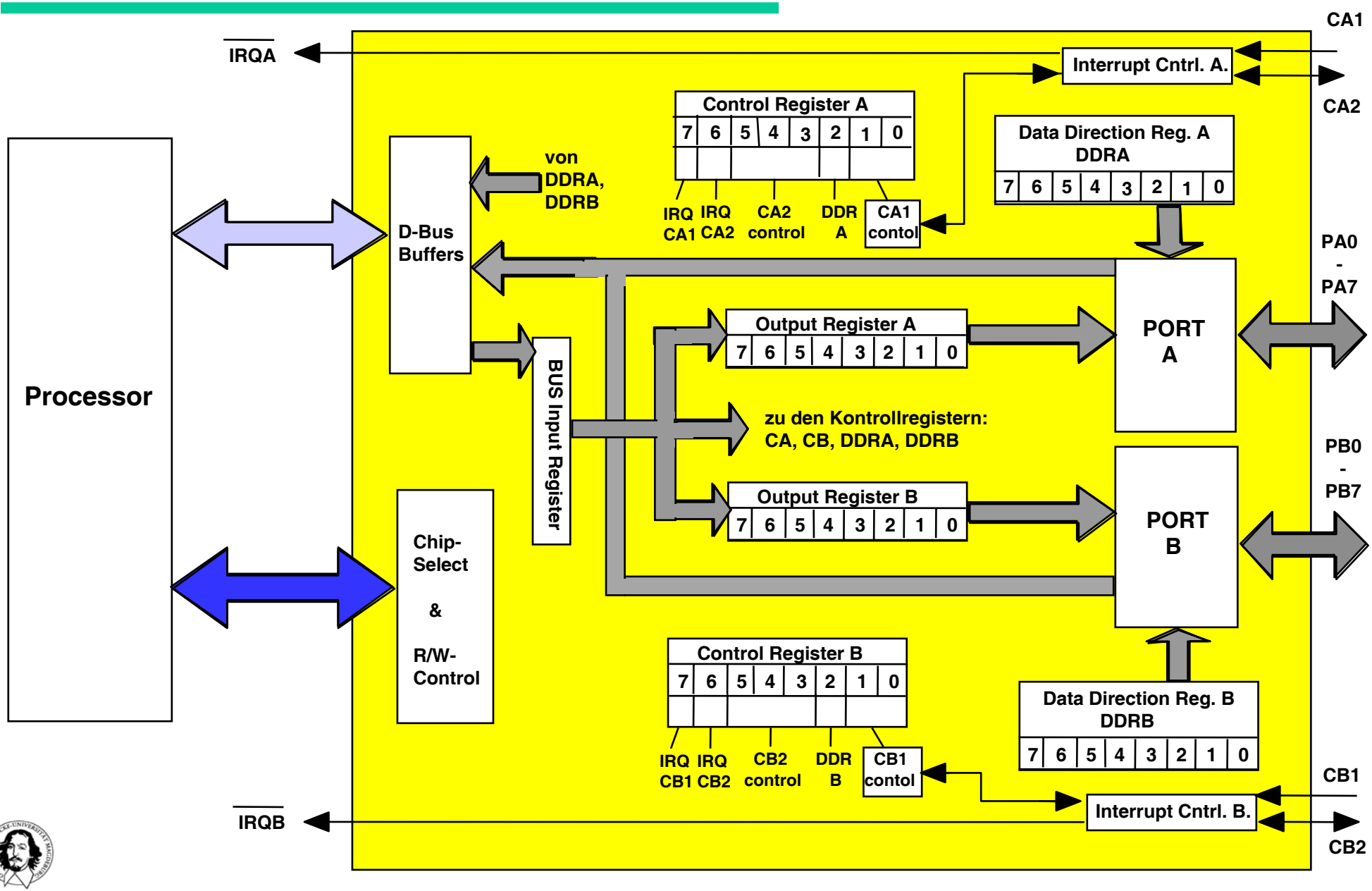



Ausgabe-
register
OHIN
SFOUT
R
S
Q
Q
OHOUT
Output-
Handshake-
Cntrl.
D7
D6
D5
D4
D3
D2
D1
D0
PD7
PD6
PD5
PD4
PD3
PD2
PD1
PD0
A15
A14
A13
CS
A2
A1
A0
RS1
RS0
DDR
IHIN
SFIN
IH
Input
Handshake-
Cntrl.

Eine Byte-parallele Geräteschnittstelle (PIA 6821)



PIA 6821


IRQA
CA1
Interrupt Cntrl. A.
CA2
Control Register A
7 6 5 4 3 2 1 0
IRQ CA1
IRQ CA2
CA2 control
DDR A
CA1 contol
Data Direction Reg. A
DDRA
7 6 5 4 3 2 1 0
von DDRA, DDRB
D-Bus Buffers
Processor
PA0 - PA7
Output Register A
7 6 5 4 3 2 1 0
PORT A
BUS Input Register
zu den Kontrollregistern:
CA, CB, DDRA, DDRB
PB0 - PB7
Output Register B
7 6 5 4 3 2 1 0
PORT B
Chip-Select & R/W-Control
Control Register B
7 6 5 4 3 2 1 0
IRQ CB1
IRQ CB2
CB2 control
DDR B
CB1 contol
Data Direction Reg. B
DDRB
7 6 5 4 3 2 1 0
CB1
Interrupt Cntrl. B.
IRQB
CB2

MOTOROLA
SEMICONDUCTORS
3501 ED BLUESTEIN BLVD., AUSTIN, TEXAS 78721

MC6821

6821
Data Sheet

FIGURE 18 — CONTROL WORD FORMAT

Determine Active CA1 (CB1) Transition for Setting Interrupt Flag IRQA(B)1 — (bit 7)
b1=0: IRQA(B)1 set by high-to-low transition on CA1 (CB1)
b1=1: IRQA(B)1 set by low-to-high transition on CA1 (CB1).

CA1 (CB1) Interrupt Request Enable/Disable
b0=0: Disables IRQA(B) MPU Interrupt by CA1 (CB1) active transition.[1]
b0=1: Enable IRQA(B) MPU Interrupt by CA1 (CB1) active transition.
1. IRQA(B) will occur on next (MPU generated) positive transition of b0 if CA1 (CB1) active transition occurred while interrupt was disabled.

IRQA(B) 1 Interrupt Flag (bit 7)
Goes high on active transition of CA1 (CB1); Automatically cleared by MPU Read of Output Register A(B). May also be cleared by hardware Reset.

Control Register

b7	b6	b5	b4	b3	b2	b1	b0
IRQA(B)1 Flag	IRQA(B)2 Flag	CA2 (CB2) Control			DDR Access	CA1 (CB1) Control	

IRQA(B)2 Interrupt Flag (bit 6)
When CA2 (CB2) is an input, IRQA(B) goes high on active transition CA2 (CB2); Automatically cleared by MPU Read of Output Register A(B). May also be cleared by hardware Reset.
CA2 (CB2) Established as Output (b5=1): IRQA(B) 2=0, not affected by CA2 (CB2) transitions.

Determines Whether Data Direction Register Or Output Register is Addressed
b2=0: Data Direction Register selected.
b2=1: Output Register selected.

CA2 (CB2) Established as Output by b5=1
(Note that operation of CA2 and CB2 output functions are not identical)

b5 b4 b3 → CA2

1 0

b3=0: **Read Strobe with CA1 Restore**
CA2 goes low on first high-to-low E transition following an MPU read of Output Register A; returned high by next active CA1 transition, as specified by bit 1.

b3=1: **Read Strobe with E Restore**
CA2 goes low on first high-to-low E transition following an MPU read of Output Register A; returned high by next high-to-low E transition during a deselect.

b5 b4 b3 → CB2

b3=0: **Write Strobe with CB1 Restore**
CB2 goes low on first low-to-high E transition following an MPU write into Output Register B; returned high by the next active CB1 transition as specified by bit 1. CRB-b7 must first be cleared by a read of data.

b3=1: **Write Strobe with E Restore**
CB2 goes low on first low-to-high E transition following an MPU write into Output Register B; returned high by the next low-to-high E transition following an E pulse which occurred while the part was deselected.

b5 b4 b3

1 1 → **Set/Reset CA2 (CB2)**
CA2 (CB2) goes low as MPU writes b3=0 into Control Register.
CA2 (CB2) goes high as MPU writes b3=1 into Control Register.

CA2 (CB2) Established as Input by b5=0

b5 b4 b3

0 → **CA2 (CB2) Interrupt Request Enable/Disable**
b3=0: Disables IRQA(B) MPU Interrupt by CA2 (CB2) active transition.*
b3=1: Enables IRQA(B) MPU Interrupt by CA2 (CB2) active transition.
*IRQA(B) will occur on next (MPU generated) positive transition of b3 if CA2 (CB2) active transition occurred while interrupt was disabled.

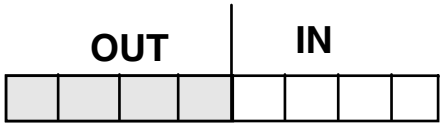
→ **Determines Active CA2 (CB2) Transition for Setting Interrupt Flag IRQA(B)2 — (Bit b6)**
b4=0: IRQA(B)2 set by high-to-low transition on CA2 (CB2).
b4=1: IRQA(B)2 set by low-to-high transition on CA2 (CB2).

MOTOROLA Semiconductor Products Inc.

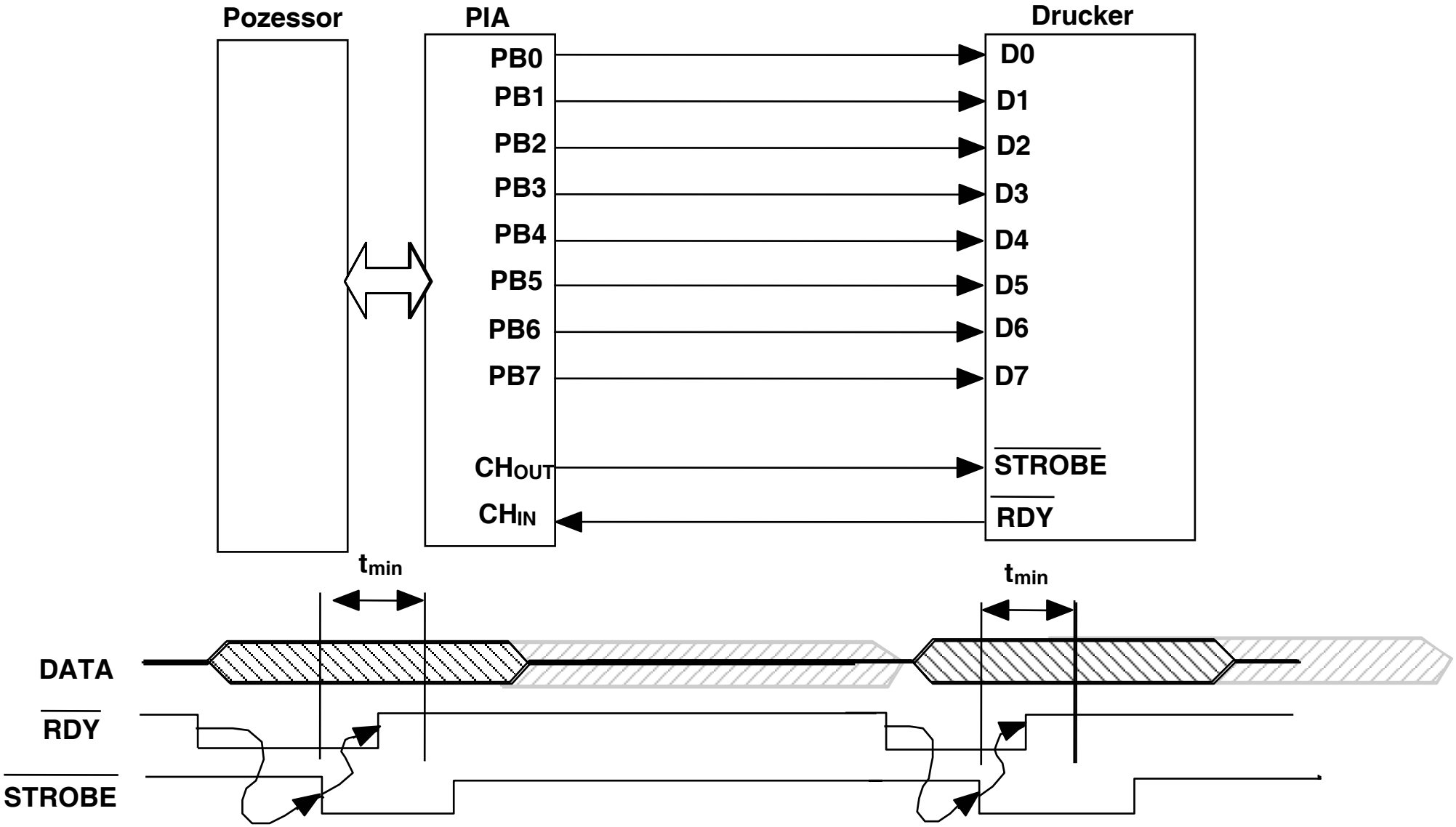
Konfigurations- und Ausgabeprogramm

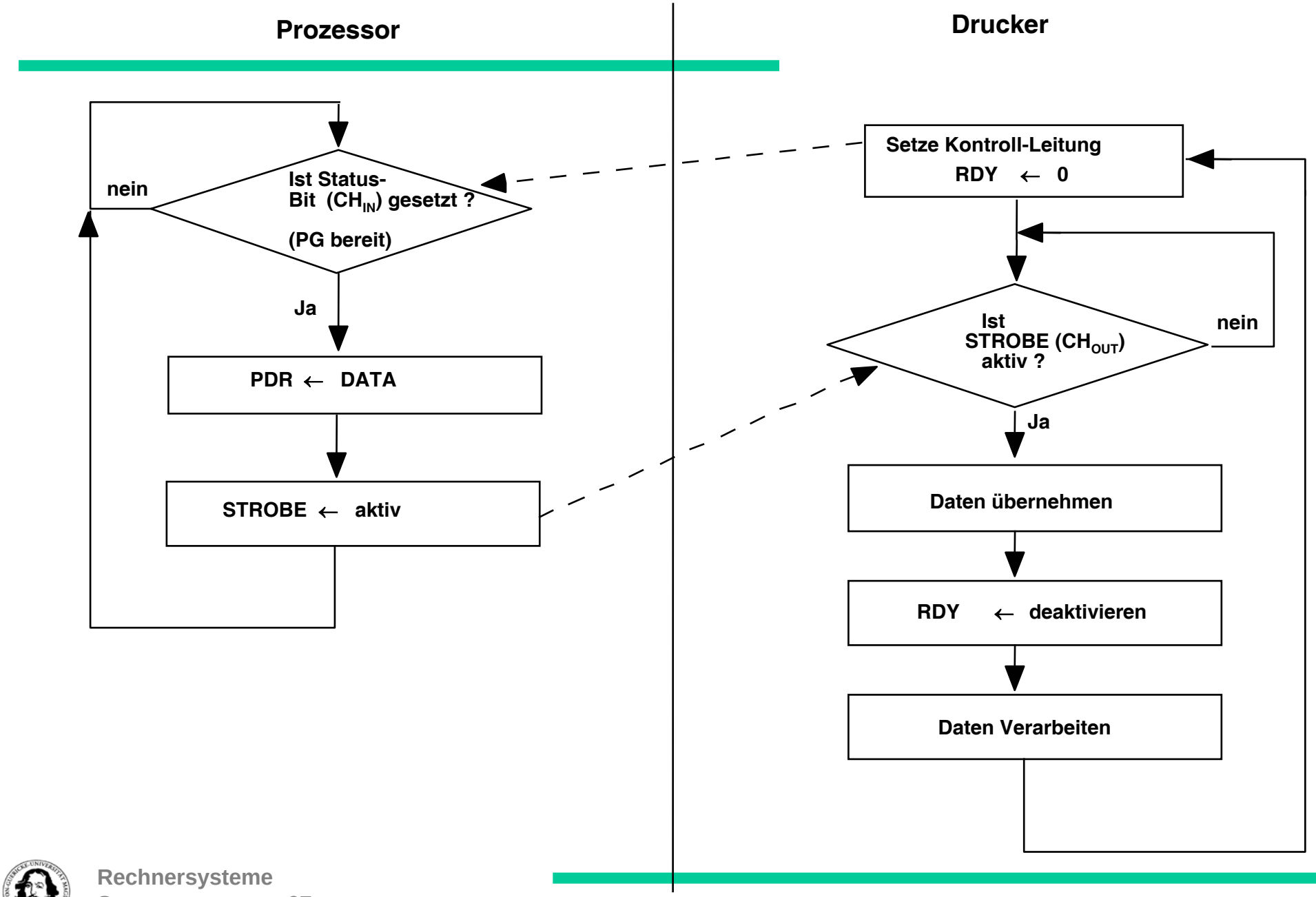
t

```
PIABAS      EQU     Adr.
PDR         EQU     Adr + offset1     peripheres Datenregister
DDR         EQU     Adr + offset2     Data  Direction  Register
CR          EQU     Adr + offset3     PIA Kontroll Register

Konfigurieren der PIA

                    LDA     #%1111 0000     4 Ausgangs und 4 Eingangsleitungen
                    STA     DDR
                    LDA     # Konf. -Muster z.B. Polarität der Steuersignale etc.
                    STA     CR
                    LDA     PDR             Dummy Read setzt Status Bit zurück

Ploop               Berechne Adresse des Ausgabedat.
                    LDA     Data            Prozessor lädt Ausgabedaten in Reg. A
Aloop               TST     CR              Höchstwertiges Bit des CR wird getestet
                    BPL     Aloop           Falls nicht gesetzt, warten
                    STA     PDR             Status-Bit gesetzt, Daten werden ins  PDR geschrieben
                    BRA     Ploop           Nächstes Datum  wird geholt
```

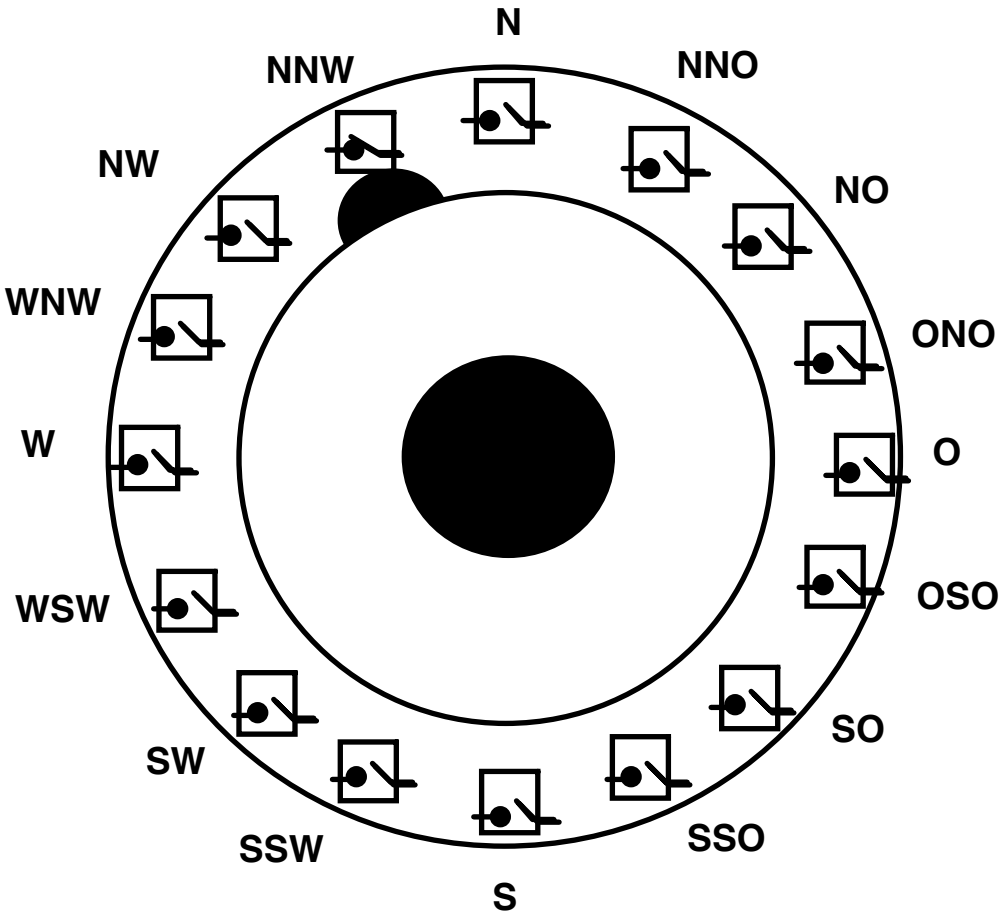


Parallele Druckerschnittstelle (Centronics)

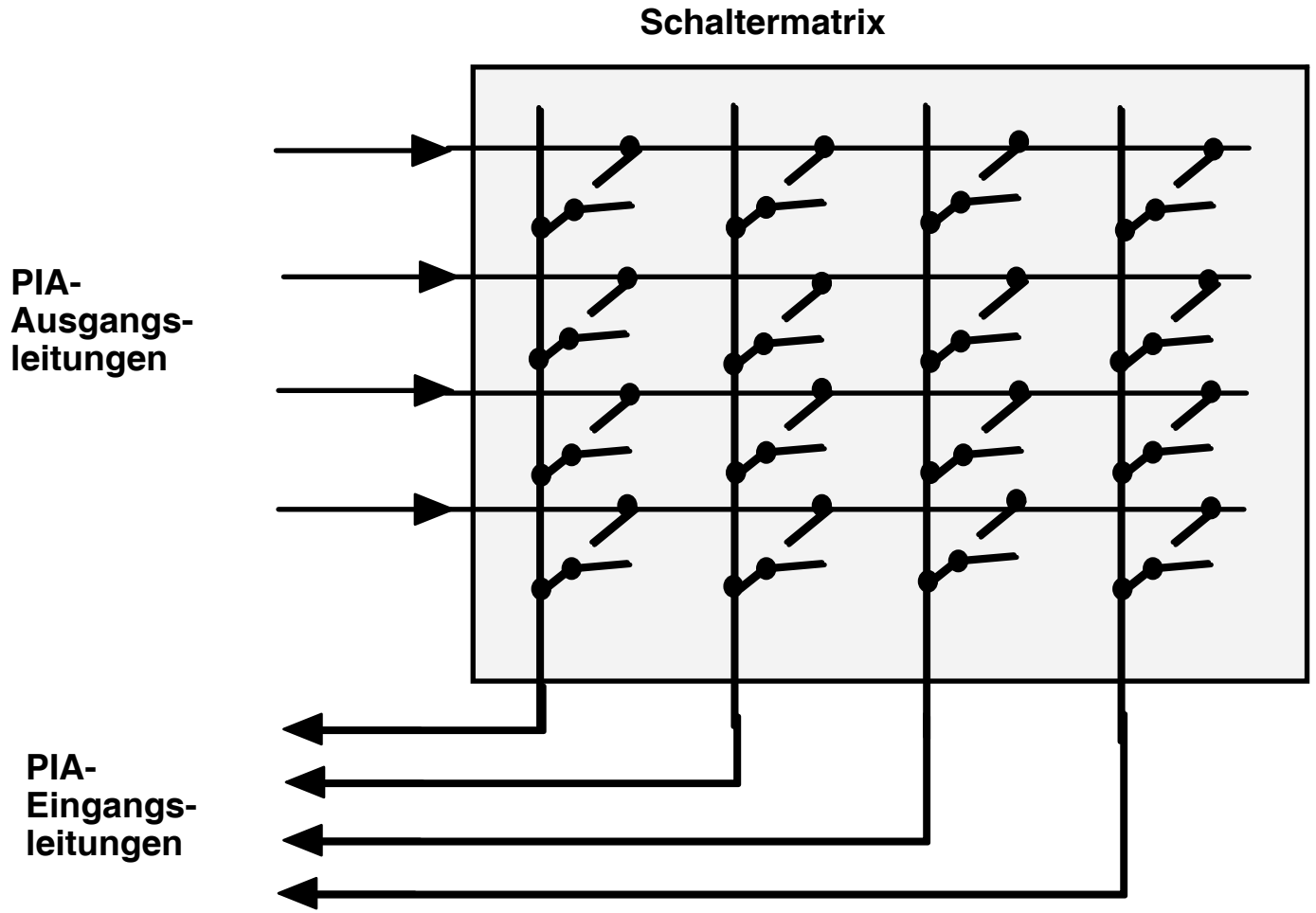



Prozessor
Drucker
Setze Kontroll-Leitung
RDY ← 0
nein
Ist Status-
Bit (CH_IN) gesetzt ?
(PG bereit)
Ja
PDR ← DATA
STROBE ← aktiv
Ist
STROBE (CH_OUT)
aktiv ?
nein
Ja
Daten übernehmen
RDY ← deaktivieren
Daten Verarbeiten

Beipielanwendung: Windrichtungsanzeige

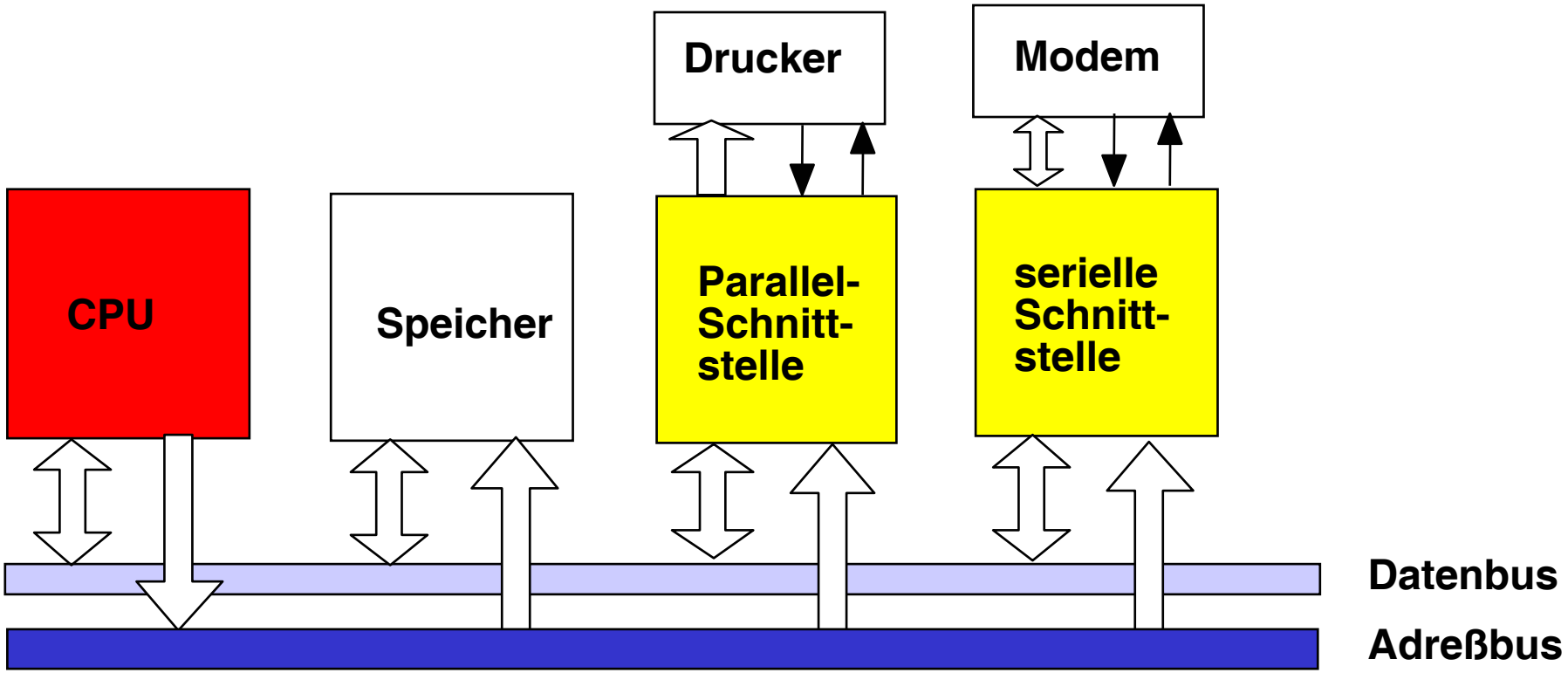


Beipielanwendung: Windrichtungsanzeige

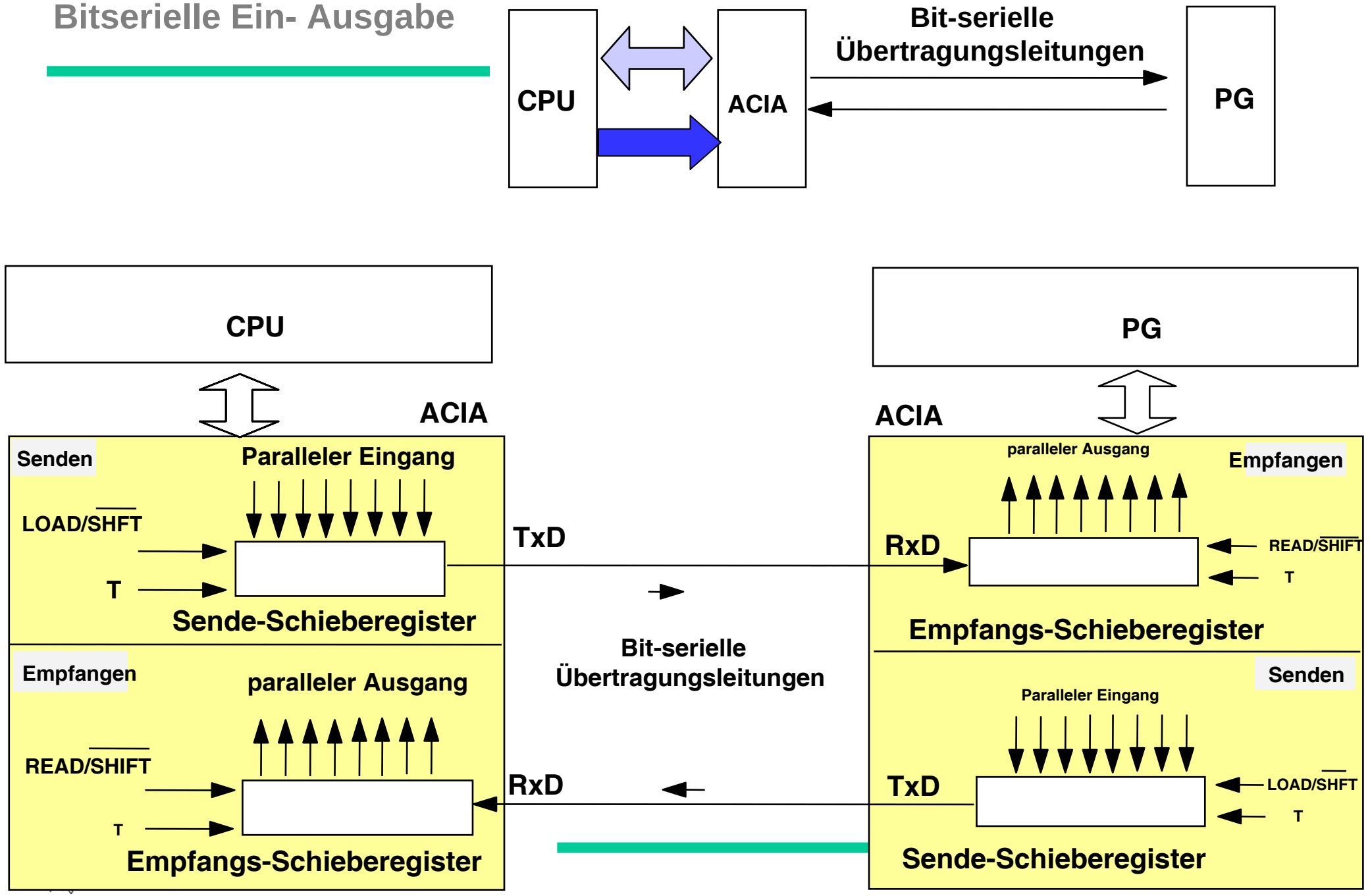


Assembler Fragment zur Dekodierung der Anzeige

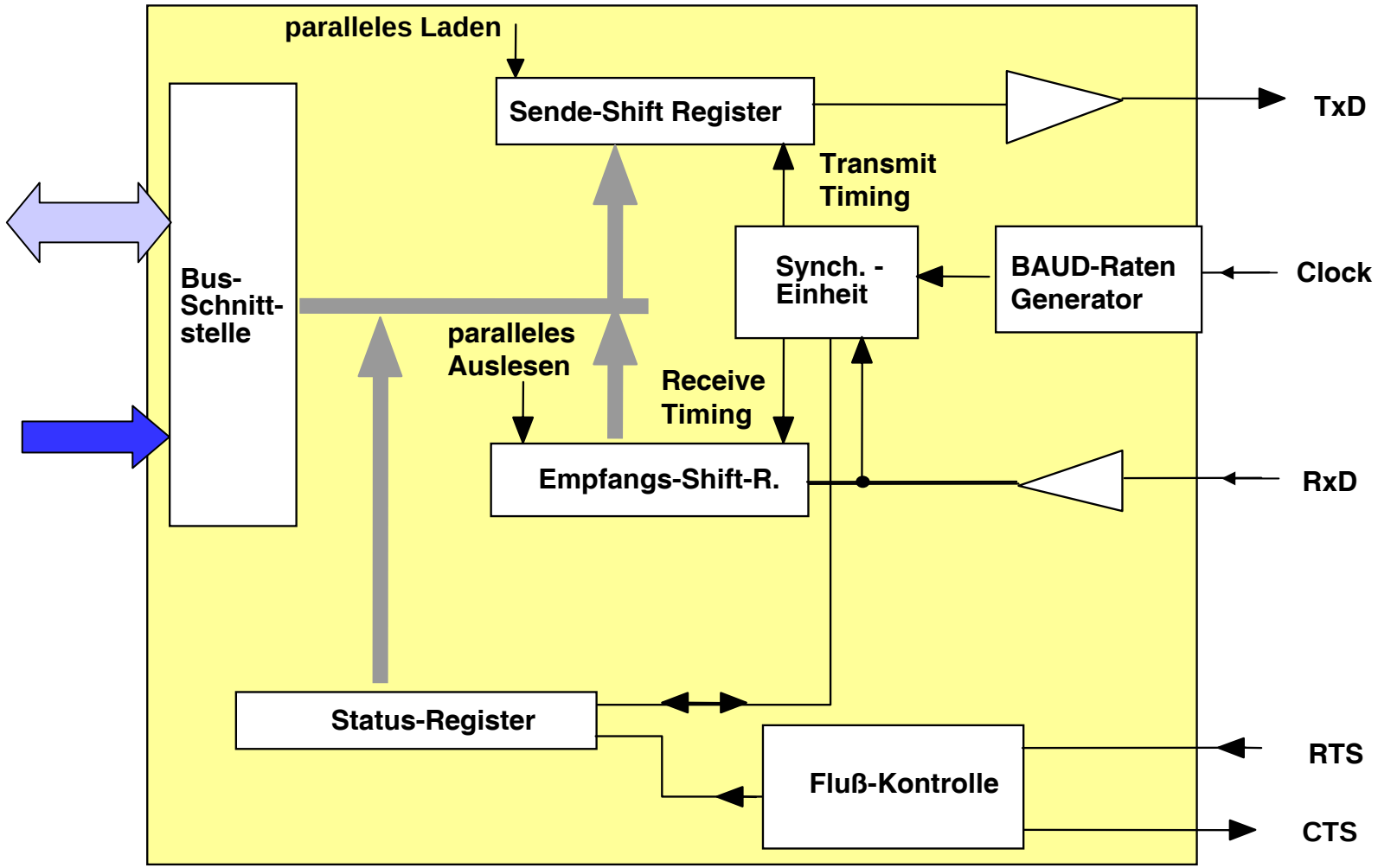
```
                ORG     $8000

PIABSE  EQU     xxxx            Festlegen der PIA-Basisadresse
DDR     EQU     PIABSE+3        DDR wird mit Distanz 3 adressiert
ORA     EQU     PIABSE+0        I/O- Register A
IOPAT   EQU     %1111 0000      Muster zur Konfiguration des DDR
INPAT1  EQU     %0001 0000      Muster für Out-Leitung 1
INPAT2  EQU     %0010 0000      Muster für Out-Leitung 2
INPAT3  EQU     %0100 0000      Muster für Out-Leitung 3
INPAT4  EQU     %1000 0000      Muster für Out-Leitung 4

                LDA     #IOPAT  Konfigurieren des
                STA     DDR     Data Direction Register

ROW1    LDA     #INPAT1
                STA     ORA     Aktiviere Leitung 1
                LDA     ORA     Lies Input
                TSTA            Ist Schalter aktiviert?
                BEQ     EVAL1   Wenn ja, berechne Windrichtung
                nächste Leitung ....
                .
                .

EVAL1   Indiziere mit Register A die Windrichtung
```

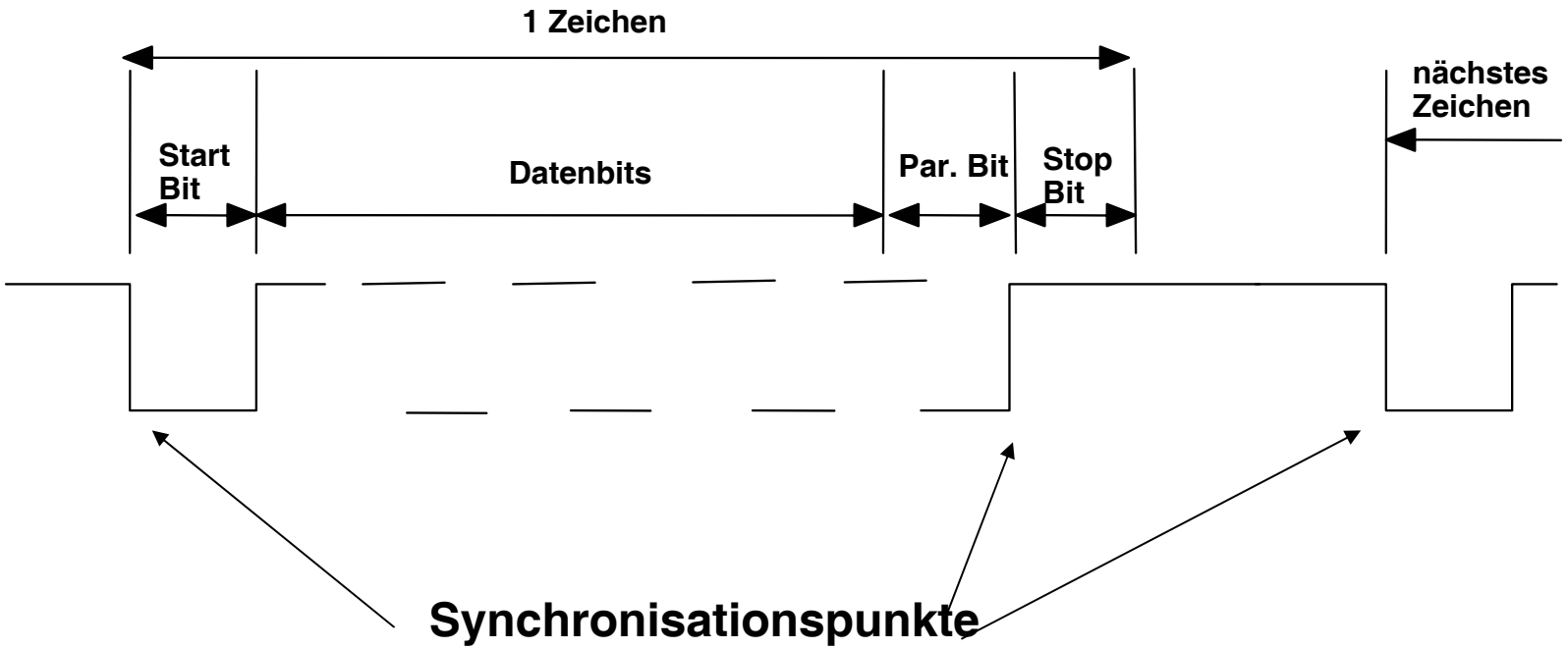

Drucker
Modem
CPU
Speicher
Parallel-
Schnitt-
stelle
serielle
Schnitt-
stelle
Datenbus
Adreßbus

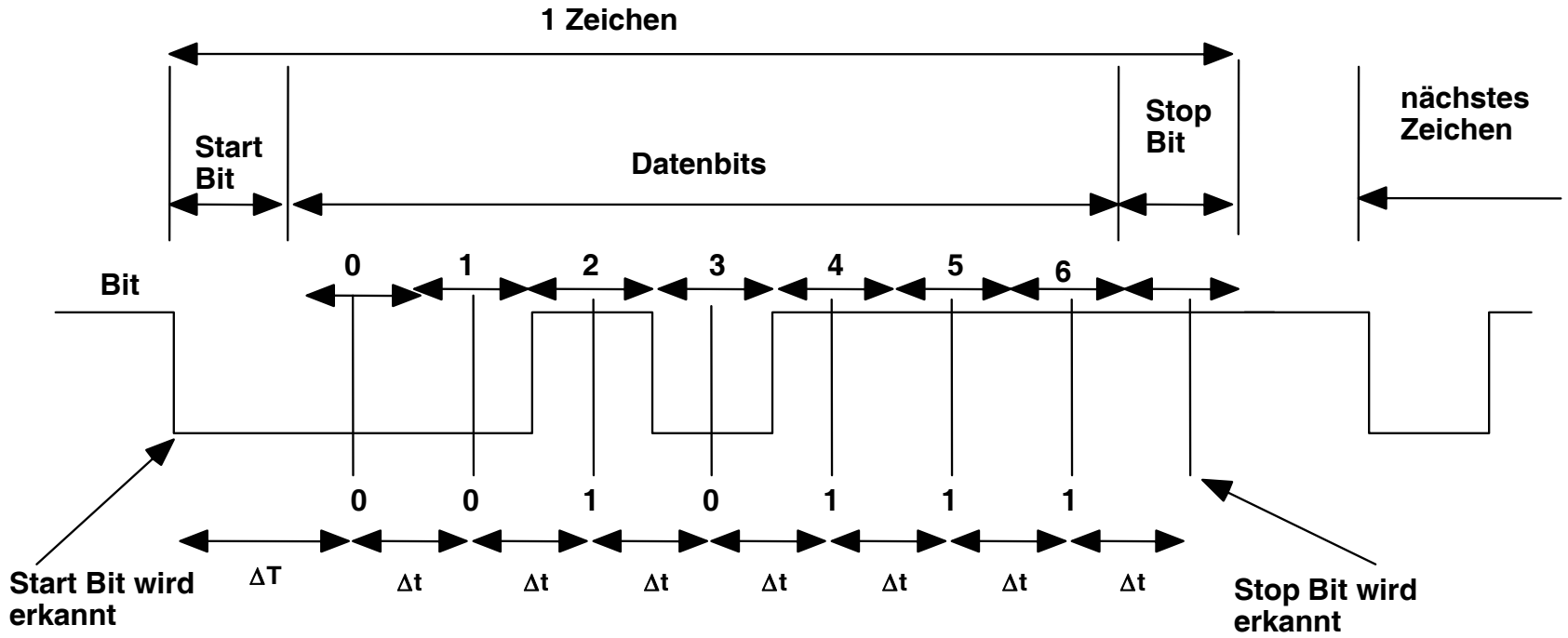
Bitserielle Ein- Ausgabe



Bitserielle Ein- Ausgabe




1 Zeichen
nächstes Zeichen
Start Bit
Datenbits
Par. Bit
Stop Bit
Synchronisationspunkte


1 Zeichen
Start Bit
Datenbits
Stop Bit
nächstes Zeichen
Bit
0
1
2
3
4
5
6
0
0
1
0
1
1
1
ΔT
Δt
Δt
Δt
Δt
Δt
Δt
Δt
Start Bit wird erkannt
Stop Bit wird erkannt

Lernziele

Kenntnis der unterschiedlichen Möglichkeiten periphäre Geräte einzubinden und anzusprechen.

Aufbau und Funktionsweise einer einfachen parallelen Schnittstelle

Koordination zwischen CPU, Schnittstellenkomponente und periphärem Gerät

Konfiguration und Programmierung der Schnittstellenkomponente

Funktionsweise der asynchronen seriellen Schnittstelle