

Rechnertechnik

Mitschrift der Vorlesung
bei LBA Perner
im Wintersemester 2001/2002

Stand: 18. Januar 2002

Vorwort

Rechtliches

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.1 or any later version published by the Free Software Foundation; with no Invariant Sections, with no Front-Cover Texts, and with no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".¹

¹ Der genaue Text findet sich unter <http://www.fsf.org/copyleft/fdl.html>

Inhaltsverzeichnis

Kapitel 1

Historisches

Motiv: Vereinfachung des Zahlenrechnens

- Kugelrechenmaschinen
- 1623 Wilhelm Schickend: Addition und Subtraktion
- 1641 Blaise Pascal: Addition und Subtraktion
- 1670 Gottfried Wilhelm Leibnitz: Alle Grundrechenarten; Mechanik des Sprossenrads:

- Rechenschieber: Multiplikation und Division; basiert auf

$$\log(ab) = \log(a) + \log(b) \quad \log(a/b) = \log(a) - \log(b)$$

- 1940 Konrad Zuse (eigentlich Bauingenieur)
 - Digitaldarstellung von Zahlen (eigentlich alt)
 - Kombination mit elektrischer Verarbeitung (zunächst mit Relais)

Multiplikation kann als Addition aufgebaut werden, s.o.

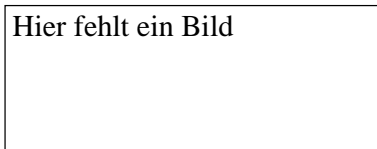


Abbildung 1.1: Grafik ueber Mechanik des Sprossenrads

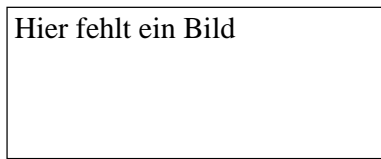


Abbildung 1.2: Beispiel Binär-Addition

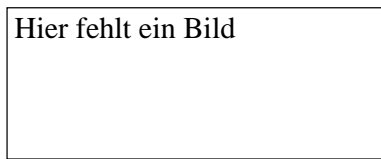


Abbildung 1.3: Beispiel Binär-Subtraktion

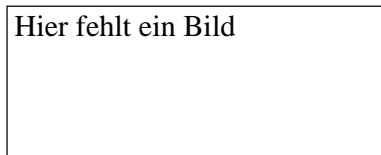


Abbildung 1.4: Beispiel Binär-Multiplikation

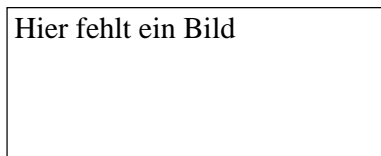


Abbildung 1.5: Beispiel Binär-Division

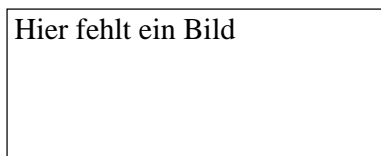


Abbildung 1.6: Addierer fuer 3 Stellen

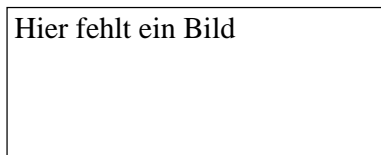


Abbildung 1.7: Wahrheitstafel fuer 1-bit-Volladdierer

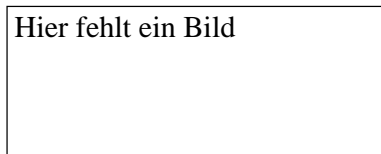


Abbildung 1.8: Gatterschaltung fuer 1-bit-Volladdierer, 7 AND, 2 OR

Kapitel 2

Digitale Schaltungen

2.1 Grundsaltungen

- AND-Gatter:
- Versorgungsspannung: 5 (auch 3) V:
 - "1" entspricht 3V-5V
 - "0" entspricht 0V-0.5V

- 1.Fall: $R=5k\Omega$, $R_d=200\Omega$, $R_s=10k\Omega$

$$R_{ges} = R + (1 / ((1/R_d) + (1/R_s))) = 5k\Omega + 100\Omega = 5.1k\Omega \quad I = U / R_{ges} = 0.98mA \quad U_y = I * R_s = 9.8V$$

- 2.Fall:

$$R_{ges} = 200 + (1 / ((1/5000) + (1/10000))) = 3533\Omega \quad I = U / R_{ges} = 0.0014A = 1.4mA \quad U_y = I * R_{ges} = 4.72V$$

- 3.Fall: analog zu 2. Fall

- 4.Fall:

$$I = 0 \implies U_y = U = 5V \implies Y = "1"$$

- OR-Gatter: $R=5k\Omega$, $R_d=200\Omega$, $R_s=10k\Omega$

– 1.Fall: $R_{ges} = +\infty \implies U_y = 0 \implies Y = "0"$

– 2.Fall:

$$R_{ges} = R_d + (1 / ((1/R) + (1/R_s))) = 3533\Omega \quad I = U / R_{ges} = 1.4mA \quad U_y = 3533 * I = 4.72V \implies Y = "1"$$

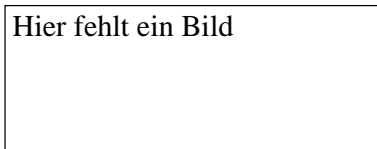


Abbildung 2.1: UND-Gatter, Wahrheitstafel $A \& B = Y$

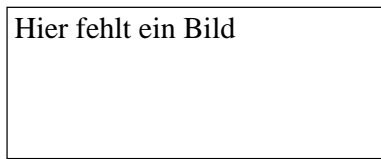


Abbildung 2.2: Schaltung UND-Gatter mit 2 Dioden, 1 R

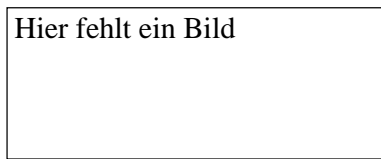


Abbildung 2.3: Schaltung AND-Gatter bei $A=B=0$

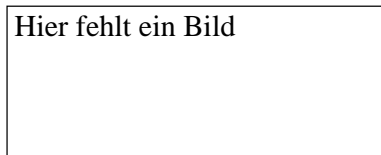


Abbildung 2.4: Schaltung AND-Gatter bei $A=B=0$ mit Widerständen

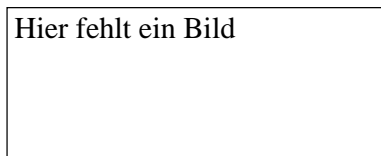


Abbildung 2.5: Schaltung AND-Gatter bei $A=1, B=0$

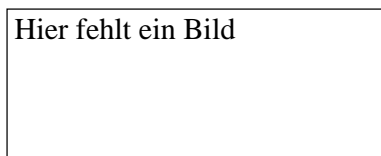


Abbildung 2.6: Schaltung AND-Gatter bei $A=1, B=0$ mit Widerständen

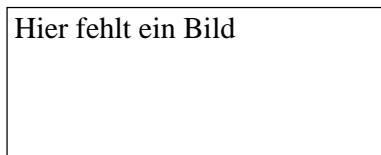


Abbildung 2.7: Schaltung AND-Gatter bei $A=B=1$

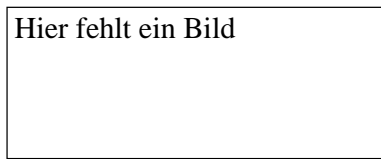


Abbildung 2.8: Grafik OR-Gatter, Wahrheitstafel $A-B=Y$

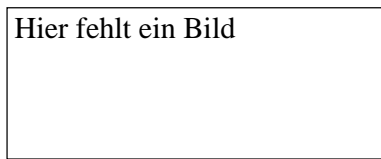


Abbildung 2.9: Schaltung OR-Gatter mit 2 Dioden, 1 R

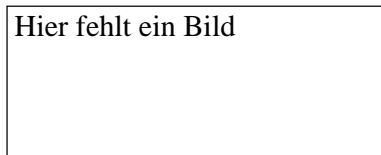


Abbildung 2.10: Schaltung OR-Gatter bei $A=B=0$

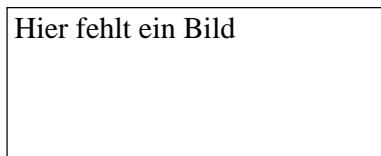


Abbildung 2.11: Schaltung OR-Gatter bei $A=1, B=0$

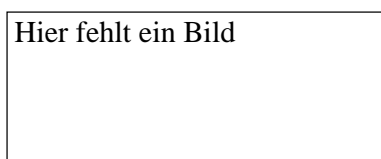


Abbildung 2.12: Schaltung OR-Gatter bei $A=1, B=0$ mit Widerständen

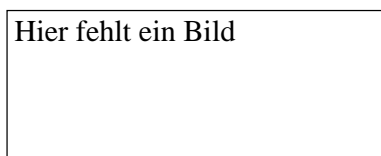


Abbildung 2.13: Schaltung OR-Gatter bei $A=B=1$

Hier fehlt ein Bild

Abbildung 2.14: Schaltung OR-Gatter bei A=B=1

Hier fehlt ein Bild

Abbildung 2.15: Blockschaltbild NOT-Gatter

- 3.Fall analog zu 2.Fall
- 4.Fall: $R_{ges}=5100$, $I=0.98mA$, $U_y=5k\Omega \cdot 0.98mA=4.90V \Rightarrow Y="1"$

- Inverter:

- 1.Fall:

$$R_{ges} = 15k\Omega \cdot I = \frac{5}{15} \cdot 10^{-3} mA = 0.33mA \quad U_y = IR = 0.33mA \cdot 10k\Omega = 3.3V \Rightarrow Y = "1"$$

(aber gerade noch, man wuerde dann in der Praxis andere Widerstände nehmen)

- 2.Fall: $R_{ges}=5200\Omega$, $I=5/5200=0.96mA$, $U_y=200\Omega \cdot 0.96mA=0.19V$,
 $\Rightarrow Y="0"$

2.2 Gatter-Ausgangsschaltungen

Sie dienen der Signalformung.

1. Interner Kollektor-Widerstand
2. Offener Kollektor \rightarrow externer Widerstand notwendig! z.B. 7416
3. Gegentakt-Ausgang Es muss sichergestellt sein, dass nur einer der beiden Transistoren durchschaltet, sonst Kurzschluss. Wenn beide Transistoren sperren, dann haben wir ein frei im Raum schwebendes Potential. Man nennt dies den "dritten Zustand".

Hier fehlt ein Bild

Abbildung 2.16: Wahrheitstabelle NOT-Gatter $A=Y$

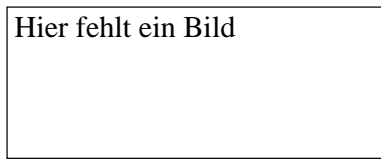


Abbildung 2.17: Schaltbild NOT-Gatter mit Transistor

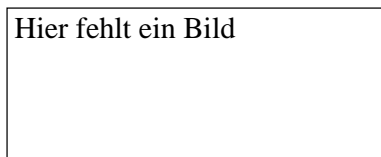


Abbildung 2.18: Ersatzschaltung NOT-Gatter mit Widerständen im Sperrfall

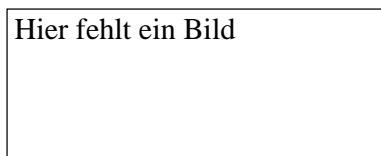


Abbildung 2.19: Ersatzschaltung NOT-Gatter mit Widerständen im Durchlassfall

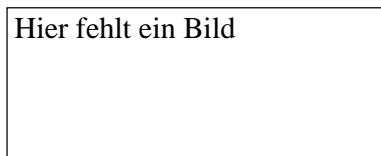


Abbildung 2.20: mit internem Koll-Widerstand

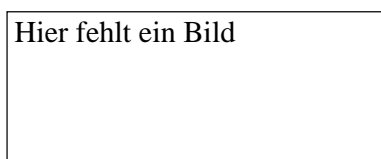


Abbildung 2.21: mit offenem Kollektor, R extern

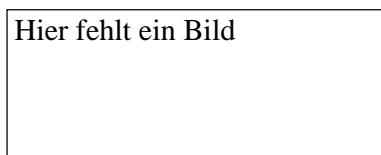


Abbildung 2.22: Gegentakt-Ausgang

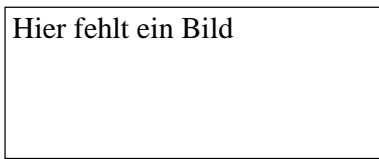


Abbildung 2.23: Tristate

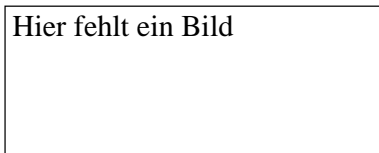


Abbildung 2.24: Oszillator

4. Tri-State Ausgang Der Block rechts funktioniert wie ein Schalter. Nötig für Bus-Systeme, wo nur ein Teilnehmer mit dem Bus sprechen darf.

2.3 Oszillatoren

ist ungenau, und wird hergenommen wenn Genauigkeit nicht so wichtig ist.

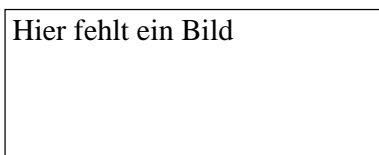


Abbildung 2.25: Oszi mit Quarz

Kapitel 3

Informationsspeicher

- Statische Speicher SRAM basieren auf Flipflops pro Bit Information
- Dynamische Speicher DRAM
- Festwertspeicher ROM/PROM
- Mehrfach beschreibbarer Festwertspeicher EPROM/EEPROM

Für den Einsatz in Rechnern werden Speicherschaltungen zu grösseren Einheiten mit 10^4 bis 10^6 Bytes zusammengefasst (sogenannten Speicherchips). Die Speicherung und Wiedergewinnung der Information erfolgt nach dem *Busprinzip*.

Ein Bus besteht aus:

- Addressleitungen (16, 32, ...): *Adressbus* zur Addressierung der Speicherplätze
- Datenleitungen (8, 16, ...): *Datenbus* über welche die Information gespeichert bzw. ausgelesen wird
- Steuerleitungen (3, ...): *Steuerbus*

z.B. Speicherbaustein 65kx8 (65000 Speicherplätze, 8 bit):

3.1 Addressbus (AB)

:

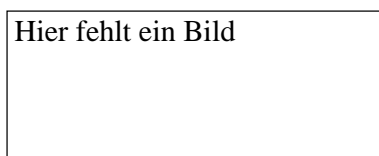


Abbildung 3.1: Chip mit 16 Addressleitungen, 8 Datenleitungen

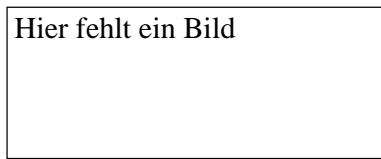


Abbildung 3.2: aus 4 Chips 256x4 einen Speicher 512x8 aufbauen

Jede Kombination (A0 ... A15) gleich 65000 Möglichkeiten adressiert einen 8 bit enthaltenden Speicherplatz. Die Speicherplätze werden entsprechend den Hexadezimal-Äquivalenten der A0 bis A15 bezeichnet. Der AB muß für seine Leitungen sein Potential von außen aufgeprägt bekommen (unidirektionaler Bus).

3.2 Datenbus (DB)

:

Jede Kombination (D0 ... D7) stellt eine Informationseinheit dar (256 Mögl.). Diese werden ebenfalls entsprechend ihren Hexadezimaläquivalenten bezeichnet. Die Leitungen des DB erhalten entweder ihr Potential von außen (schreiben IN den Speicher) oder von innen (Lesen AUS dem Speicher).

3.3 Steuerbus (SB)

:

Mit den SElect-Signal wird der Baustein aktiviert. Ist SEL nicht gesetzt, sind alle Datenausgänge unabhängig vom Zustand der anderen Steuersignale im Tri-State-Zustand. SEL ist meist Active-low ausgeprägt.

Mit dem Signal R/W wird festgelegt ob der Baustein im Lesezustand (R) oder im Schreibzustand arbeitet.

R/W=1 bedeutet: der durch AB adressierte Speicherplatz gibt seine Information an den internen Datenbus.

R/W=0: Die an den D0 bis D7 liegende Signalkombination wird an die durch A0 bis A15 bestimmte Position geschrieben. Mit dem Signal OW wird die Potentialkombination des internen Datenbusses nach außen durchgeschaltet (Tri-State-Zustand aufgehoben). OE ist ebenfalls meist Active-low ausgeprägt. SB ist ein unidirektionaler Datenbus.

Beispiel 3.1 Aus Bausteinen 256x4 soll ein Speicher 512x8 aufgebaut werden (nur vereinfachte Darstellung)

Beispiel 3.2 Aus Bausteinen 1024x8 soll Speicher 4096x8 aufgebaut werden

3.4 Ausschluß aktiver Bauelemente von einem Bus

(Addierer mit Ein/Ausgaberegister und Speicher)

Hier fehlt ein Bild

Abbildung 3.3: aus 4 Chips 1024x8 einen Speicher 4096x8, mit DEC

Hier fehlt ein Bild

Abbildung 3.4: Schaltung Addierer mit 3 Registern und 256x8-Speicher

Annahmen:

[03]=A, [07]=B $A+B=[0C]$

Schritte zur Durchführung der Addition

- Über den Adressbus wird die Speicherstelle 03 ausgewählt. Das Signal R/W wird auf "Lesen" gesetzt und OE aktiviert. Damit liefert der Speicher den Inhalt A der Speicherzelle 03 an den Datenbus. Gleichzeitig wird das Clock-Signal für R1 aktiviert wodurch der Inhalt des DB in das Register R1 geschrieben wird.
- Über den Adressbus wird die Speicherzelle 07 ausgewählt. Es wird über R/W in den Read-Modus geschaltet und OE gesetzt. Dadurch wird der Inhalt B der Speicherzelle 07 an den Datenbus geschaltet. Durch Clock auf R2 wird dieser Inhalt in R2 übertragen. Die Summe $A+B$ befindet sich nun in Register R3
- Über den Adressbus wird die Speicherstelle 0C ausgewählt. OE an R3 schaltet die Summe $A+B$ an den Datenbus durch. $R/W=0$ schaltet den Inhalt des DB an die ausgewählte Speicherstelle 0C

3.5 Register mit Speicher

- An den Adressbus wird die Signalkombination 00 gelegt, an den Datenbus 33 (hex). So wird 33h an die Speicherstelle 0 geschrieben.

Hier fehlt ein Bild

Abbildung 3.5: Schaltung Register mit Speicher

- An den Adressbus wird die Signalkombination 00 gelegt, der Datenbus bleibt offen. $RW=1$ und $OE=1$ (am Speicher) schalten den Inhalt der Speicherstelle 0 an den Datenbus. Gleichzeitig wird mit $Clock=1$ dieser Inhalt in das Register geschrieben. An den Adressbus wird nun 01 gelegt, der Datenbus bleibt offen. $OE=1$ vom Register steuert den Inhalt vom R an den Datenbus und dieser wird mit $R/W=0$ in den Speicher geschrieben.
- An den Adressbus wird 02 gelegt, der Datenbus bleibt offen. OE vom Register steuert den Inhalt von R an den DB und dieser wird mit $R/W=0$ in den Speicher geschrieben. Dieser Schritt kann mit unterschiedlichen Adressen wiederholt werden.

Das Anlegen der Signalkombinationen an AB, DB, SB kann automatisiert werden indem diese als Inhalte eines passenden Speichers in hintereinanderliegenden Speicherzellen abgelegt werden. Das Auslesen dieser Signalkombinationen erfolgt dann durch Anschließen eines Zählers an den Adressbus dieses Speichers mit $RW=1$ und $OE=1$. Da dieser Speicher die Anweisungen (oder Befehle) für die Schaltung enthält, heißt er Anweisungs- oder Befehlsspeicher im Gegensatz zum Datenspeicher.

In heutigen PCs sind diese Speicher zum Arbeitsspeicher zusammengefasst. Die Architektur mit den getrennten Speichern heißt Harvard-Architektur, die Architektur mit der Kombination der Speicher von-Neumann-Architektur. Vom Sicherheitsaspekt ist es sinnvoller, die Harvard-Architektur einzusetzen, man hat es aber mit der von-Neumann-Architektur einfacher und ist flexibler.

Kapitel 4

Register, Datenspeicher und Sprungbefehl

4.1 Bedingte Sprungbefehle

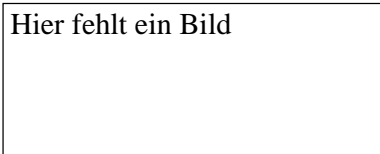
Diese werden abhängig vom Inhalt eines Spezialregisters mit unterschiedlichen Distanzen (Sprungweiten) ausgeführt.

Das Spezialregister wird durch eine arithmetische oder logische Operation geladen. (z.B. arithmetische: Überlauf bei Addition, Unterlauf bei Subtraktion, Division durch 0, logische: Prüfung auf (Un)Gleichheit, Prüfung auf größer/kleiner, usw.). Dieses Register wird auch Flag-Register genannt. Sodann wird dieser Inhalt zum aktuellen Stand des Mikro-Befehlszählers dazuaddiert, was bewirkt daß abhängig vom Inhalt des Spezialregisters der nächste oder der übernächste Befehl ausgeführt wird.

4.2 Memcopy-Befehl

4.3 Maschinenbefehle (Assemblerbefehle)

Für die Steuerung der Schaltung sind die Mikrobefehle erforderlich. Der Ablauf einer komplexeren Operation (z.B. Transfer eines Inhaltes von einem Speicherbereich in einen anderen) erfordert unter Umständen mehrere Mikrobefehle - man nennt dies eine Mikrobefehls-Sequenz. Mikrobefehls-Sequenzen werden durch Decodierung von Maschinenbefehlen erzeugt. Dies erfolgt im Befehls-Decoder.



Hier fehlt ein Bild

Abbildung 4.1: Schaltung mit Mem/Reg/BZ und Mini-Programmcode

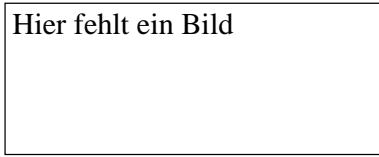


Abbildung 4.2: Schaltung mit Mem, RegA, zwei DP-Regs, Zähler und Mini-Progcode für Memcopy

Durch diese Zuordnung kann die unterschiedliche Länge der Maschinenbefehle auf die erforderliche Länge der Parameter reduziert werden.

Kapitel 5

CPU (Central Processing Unit)

Eine CPU ist eine Standardschaltung zur Ausführung von Maschinenbefehlen. Die Schaltung beinhaltet u.A.: ALU, Registereinheit, Befehls-Decoder mit MBZ, Steuereinheit, Adress-Buffer, Daten-Buffer, Steuer-Buffer, welche intern alle über einen AB/DB/SB verbunden sind.

Nach aussen geführt sind ein externer Adressbus, Datenbus, Steuerbus. Weitere Eingänge sind Takt, 5V und 0V.

5.1 Beschreibung der Funktionseinheiten

AB/DB/SB Stellt die Verbindung der internen Funktionseinheiten dar, und ermöglicht den Austausch von Daten und Signalen.

Adressbuffer Ist eine Schnittstelle der CPU nach aussen und wird vom internen Bus angesteuert. Im Adress-Puffer werden Adressen (Bit-Kombinationen) von Speicherzellen oder E/A-Registern zwischengespeichert. Die abgehenden Leitungen des Adress-Buffers bilden den externen Adressbus.

Datenbuffer Ist ein bidirektionaler Tristate-Puffer und ebenfalls eine Schnittstelle nach aussen. Im Datenpuffer werden Daten, die vom internen Bus oder vom externen Datenbus kommen, zwischengespeichert.

Steuerbuffer Ist ein bidirektionaler Puffer über welchen Steuersignale ausgetauscht werden. Es gibt Steuersignale nach aussen z.B. R/W: Fetch/MemR nach innen z.B. HOLD (wird z.B. für Single-Step-Betrieb benutzt), Memory ready, Interrupt

ALU Verknüpft zwei Operanden (Eingangsregister) zu einem Ergebnis (Ausgangsregister) und hinterlegt in einem Spezial-Register (Flag-Register) zusätzliche Informationen. Eine Standard-ALU kann arithmetische (+ - * /) und logische (& — = != < > <= >= inv) Operationen. Flag-Register: Überlauf, Unterlauf, Division durch 0, 0/1 bei logischen

Registereinheit Interner Speicher zur temporären Speicherung und Manipulation von Daten. Die Speicherplätze heissen Register (z.B. A,B,C,D,DP1,DP2,...,SP,...)

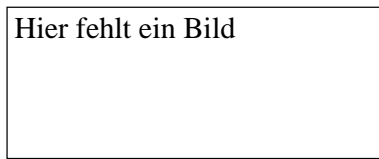


Abbildung 5.1: Vier unterschiedlich lange Befehls-Worte im Speicher; l_i = Länge des jeweiligen Maschinenbefehls in Bytes: 1,2,3,...,32

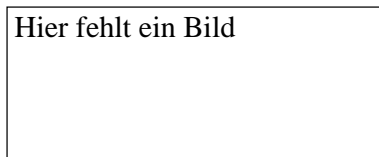


Abbildung 5.2: Grafik OP-Code danach Op1 bis Opn

Befehlsdecoder Ordnet jedem Maschinenbefehl einen oder eine Sequenz von Mikrobefehlen zu, welche im Mikroprogrammspeicher fest hinterlegt sind. Ausserdem wird im Falle einer Mikrobefehls-Sequenz der Mikrobefehls-Zähler mit der Anzahl der Befehle innerhalb der Sequenz geladen und gestartet.

Steuereinheit steuert die extern angeschlossenen Bauteile (Speicher, I/O-Geräte) entsprechend dem internen Arbeitsfortschritt der CPU. versorgt den Steuerpuffer mit den erforderlichen Signalen. Außerdem empfängt der Steuerpuffer die Signale von den extern angeschlossenen Bauteilen. Die Steuereinheit enthält außerdem den Maschinenbefehls-Zähler. Dieser enthält die Adresse des jeweils aktuell abzuarbeitenden Maschinenbefehls (Pointer auf den Anfang des Maschinenbefehls). Nach der vollständigen Abarbeitung des aktuellen Maschinenbefehls (Ende der Mikrobefehls-Sequenz) wird dieser Pointer um die Länge des aktuellen Maschinenbefehls weiterschaltet und zeigt dann auf den nächsten Maschinenbefehl.

Takt Gibt das Zeitraster für die sequentiell ablaufenden Arbeitsschritte vor

5.2 Maschinenbefehle (Assemblerbefehle)

Maschinenbefehle sind codierte Steueranweisungen für die CPU. Je nach Bauart der CPU sind die Maschinenbefehle unterschiedlich ausgeprägt (codiert). Diese Ausprägung nennt man Maschinencode.

Im Großrechner-Bereich (Mainframe) hat sich der IBM 360-Assembler durchgesetzt, im PC-Bereich als de-facto-Standard der INTEL-Assembler. Letzterer ist abwärtskompatibel, d.h. die neueren Prozessoren enthalten als echte Teilmenge den Maschinencode des Vorgängers.

5.2.1 Struktur eines Maschinenbefehls

Im OP-Code sind Art und Länge des Maschinenbefehls mit encodiert.

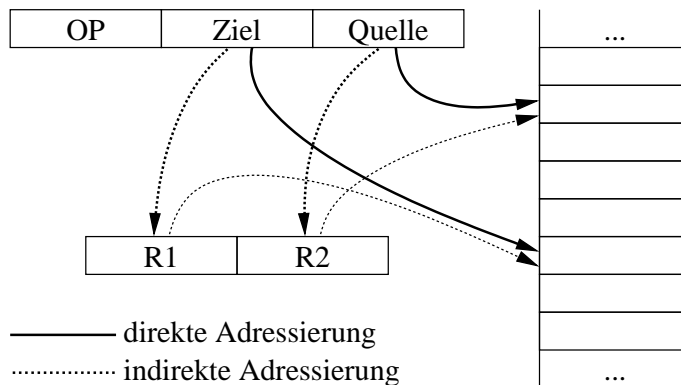


Abbildung 5.3: Links OPCode mit SRC und DST, rechts Speicherzeichnung. Kurze Erklärung von direkter (rot/fest) und indirekter (gelb/gestrichelt) Adressierung

5.2.2 Klassen von Maschinenbefehlen

Transferbefehle lesen Daten von einem Speicherplatz und schreiben diesen Inhalt an einen anderen Speicherplatz. Speicherplatz kann dabei ein Platz im Arbeitsspeicher (Memory) oder im CPU-Internen Registerspeicher sein.

- MEM $\bar{}$ MEM (MOVC, MOV)
- REG $\bar{}$ MEM (Load, LD)
- MEM $\bar{}$ REG (Store, ST)
- REG $\bar{}$ REG (Move, MOV)

Spezialfall: Datenwort steht im Befehl (IMMEDIATE: MVI, LI, SI) Adressierung vom MEM:

- Direkte Adressierung: Adresse steht im Befehl
- Indirekte Adressierung (Datenpointer): Im Befehl steht die Adresse eines Registers in dem sich die Adresse vom MEM befindet

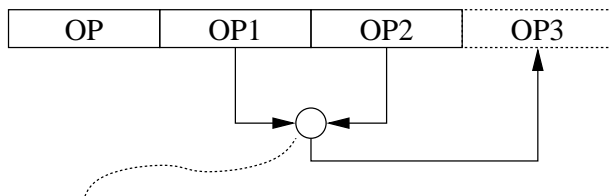
Verknüpfungsbefehle verknüpfen zwei Operanden nach dem im Operationscode angegebenen Verfahren und hinterlegen das Ergebnis in einem Speicherplatz oder Register. Außerdem wird, abhängig vom Ergebnis, das Flag-Register versorgt.

(Adressierungsarten wie bei den Transferbefehlen)

Oft ist es so, dass das Ergebnis einen Operanden überschreibt. Spezialfall: Bitmanipulationen

bedingte und unbedingte Sprünge Sprungbefehle ändern den Programmablauf in Abhängigkeit von Bedingungen (Inhalt des Spezial-Registers oder unbedingt).

- bedingt: Ist die Bedingung erfüllt, wird der Befehlszähler mit dem Sprungziel geladen, ansonsten kommt der nächste Befehl.



Operatoren: + - * / = >= > <= < | & ...

Abbildung 5.4: oben OPCode mit OP1 und OP2, optional (gestrichelt) OP3, alle miteinander verknüpft (OP1 X OP2) -, OP3; unten Liste der Operationen

JMP	Sprungziel
nächster Befehl	

Abbildung 5.5: OPCode mit Sprungziel und nächster Befehl

- unbedingt: es wird in jedem Fall der Befehlszähler mit dem Sprungziel geladen

Wie die Sprungbedingung ausgewertet wird, wird bei den bedingten Sprüngen im Operationscode angegeben.

Sprungbefehle:

- JMP
- JE, JNE
- JL, JH

Prozessorsteuerbefehle • EI, DI ermöglichen/ignorieren von Interrupts (deutsch Unterbrechungen)

- Power down
- PUSH, POP zur Bedienung des STACK-Speichers (LIFO)

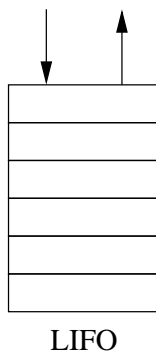


Abbildung 5.6: Stack

Hier fehlt ein Bild

Abbildung 5.7: kleiner Speicherauszug mit JMP 0000 an \$0000

Hier fehlt ein Bild

Abbildung 5.8: Links CPU mit Mem und RegA verbunden, rechts Idle-Schleife

5.3 Arbeitsweise und Zusammenwirken der Prozessoreinheiten

Beispiel: An der Adresse 0000 steht ein unbedingter Sprung mit Ziel 0000.

1. Reset-Signal: Befehlszähler wird mit 0000 geladen und gibt diese Adresse am Adressbus aus. Ausserdem wird über den Steuerbus das Signal FETCH ausgegeben.
2. Der Speicher wird dadurch an der Adresse 0000 selektiert und wegen FETCH der Inhalt dieser Speicherstelle auf dem Datenbus verfügbar gemacht.
3. in einem weiteren Schritt wird vom Befehlsdecoder der Sprungbefehl JMP erkannt und in zwei Takten die folgenden Bytes (beide 00) eingelesen.
4. Der Sprungbefehl wird ausgeführt indem diese beiden in den Befehlszähler geladen werden

5.4 Programmunterbrechung

Nach dem Einschalten und dem Reset-Signal ist eine CPU immer tätig. D.h., sie liest Befehle aus dem Befehlsspeicher und führt sie aus (z.B. die Idle-Schleife). Zur Kommunikation mit anderen Komponenten eines Rechners (Tastatur, Datenübertragung (mit Modem), Plattenspeicher usw.) ist es nötig, dies der CPU mitzuteilen. Dies geschieht durch zwei Arten:

1. Abfragebetrieb (Polling): Hierbei wird in die Idle-Schleife die Abfrage eines Schnittstellenregisters eingebaut und falls in diesem Schnittstellenregister eine Anforderung gesetzt ist, wird zu einer entsprechenden Service-Routine verzweigt. Mit einem Transferbefehl (meistens "IN") wird dieses Schnittstellenregister abgefragt

Vorteil: einfache Hardware; Nachteil: Zeitkonsum bei vielen Schnittstellenregistern

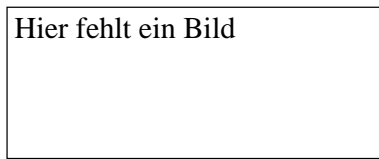


Abbildung 5.9: Interrupt-Service-Register: UR

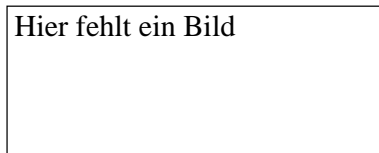


Abbildung 5.10: Grafik zur Funktionsweise Stack bei Auftreten eines Interrupts

2. Interrupt-Betrieb: Über ein spezielles Signal INT wird über den Steuerbus dem Prozessor das Vorliegen eines externen Ereignisses mitgeteilt. Von welcher Art das externe Ereignis ist, wird im Unterbrechungsregister durch eine Maske angekündigt.

Erkennt der Prozessor das Interrupt-Signal wird das Register gelesen und in Abhängigkeit von der Maske in die entsprechende Service-Routine verzweigt.

5.5 Der Stack-Speicher

Stack-Speicher ist Teil des Arbeitsspeichers und wird über ein Spezialregister (Stack-Pointer) adressiert.

Mit dem CALL-Befehl, der eine Kombination aus Push- und Sprungbefehl ist, wird an ein Sprungziel gesprungen gleichzeitig aber die Adresse des nächsten Befehles auf dem Stack-Speicher deponiert.

Das Gegenstück dazu ist der RET-Befehl. Er ist eine Kombination aus Pop- und Sprungbefehl. Er holt die am höchsten am Stack liegende Adresse und lädt diese in den Befehlszähler.

5.6 Unterprogrammtechnik

Hauptprogramm:

CALL: BZ -i STACK (PUSH)

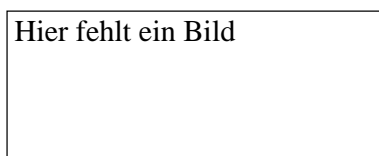


Abbildung 5.11: mehrfache Verzweigungen mit CALL und RET

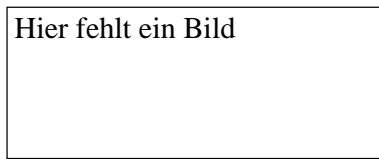


Abbildung 5.12: Links: Verknüpfung Segment/Offset, Rechts: in Segmente unterteilter Speicher

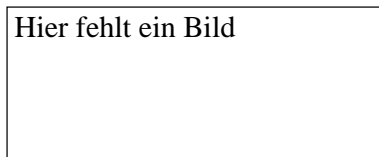


Abbildung 5.13: offset plus segment-lookup in Deskriptortabelle ergibt physikalische Adresse

Zieladresse $-i$ BZ (JMP)

Im Unterprogramm folgt als erstes ein PUSH für alle Register, die im Unterprogramm verwendet werden. An dessen Ende werden diese Register in umgekehrter Reihenfolge wieder zurückgePOPpt.

Dieser Mechanismus von Unterprogramm-Aufrufen wird von Funktionsdefinitionen in Hochsprachen wie C benutzt.

5.7 Die Adressierung des Arbeitsspeichers

Die physikalische Adresse ist die Bitkombination am Adressbus. Bildung der physikalischen Adresse erfolgt über Segmente und einen Offset.

Durch Unterteilung des Arbeitsspeichers in Segmente kann die Ausführung der Maschinenbefehle auf die Offset-Komponente reduziert werden (Verkürzung der Maschinenbefehle). Die Segmentadresse ist im Segment-Register hinterlegt und wird nur bei einem Segmentwechsel verändert. Die Daten- und Befehls-Adressierung erfolgt über das Offset-Register. Die Segmente ermöglichen eine logische Strukturierung des Speichers in Code- und Daten-Segmente, welche verschiedenen Programmen zugeordnet werden können.

5.7.1 Adressierung des Arbeitsspeichers im "protected" mode

Der Selektor-Anteil der Adresse enthält nicht eine physikalische Adress-Komponente sondern verweist auf die Deskriptor-Tabelle.

Der Deskriptor-Anteil der Adresse in den Einträgen der Deskriptor-Tabelle regelt den Zugriff auf die Speichersegmente. Unberechtigte Zugriffe auf ein Speichersegment werden abgewiesen.

Geschützt (protected) sind:

- Codesegmente eines Tasks vor dem Zugriff anderer Tasks



Abbildung 5.14: drei Matrizen Virtuell adressierbar, realer Arbeitsspeicher, Pagingbereich (Platte) mit Zusammenhängen. bm hat die Grafiken etwas zerlegt...

- Datensegmente, deren Zugriff nicht erlaubt ist (Datenbereiche des OS)
- Speichersegmente (Code/Data) die nur für privilegierte Tasks des OS zum Zugriff freigegeben sind

Privilegierungsstufen (Hierarchie):

- Betriebssystem (OS)
- Treiberprogramme
- Daten-Management-Systeme
- Anwenderprogramme

Jeder Privilegierungsstufe ist ein Berechtigungs-Deskriptor zugeordnet, bei jedem Speicherzugriff wird dieser mit dem Segment-Deskriptor verglichen. Stimmen diese nicht überein, wird der Zugriff abgewiesen.

5.7.2 Virtuelle Adressierung oder Paging

Falls der physikalische Speicher wesentlich kleiner ist als die durch Segmentanzahl und Segmentgröße adressierbare Speichergröße, werden nicht alle Segmente im Speicher gehalten, sondern auf den Plattenspeicher ausgelagert.

Die Programmausführung erfolgt im physikalischen Speicher. Wird eine Adresse angesprochen, die nicht in einem Segment des physikalischen Speichers liegt, so wird über eine Unterbrechung das entsprechende Segment vom Paging-Speicher in den physikalischen Speicher geladen. Dadurch "erscheint" der Arbeitsspeicher größer als er eigentlich ist.

5.8 Cache-Speicher

Cache-Speicher sind Zwischenspeicher für Daten und/oder Code. Sie sind als schnelle SRAMs ausgebildet und befinden sich entweder auf dem CPU-Chip (L1 on-chip Cache) oder ausserhalb der CPU als externe Komponente (L2 external Cache). Aufgabe der Caches: Beschleunigung des CPU-Zugriffs auf Daten & Code.

Hier fehlt ein Bild

Abbildung 5.15: links: Zusammenhang CPU, Mem bus contr./Cache Contr./SRAM Cache, DRAM. Als Teilmenge von SRAM dargestellt ist Rechts: Cachelines, n einträge a 33 Byte, jedes 1. Byte Index

5.8.1 Arbeitsweise des Cache-Speichers

- Der Datenverkehr zwischen DRAM und SRAM Cache erfolgt blockweise im *Burst Mode*.
- Beim *Burst Mode* (auch Blockmodus genannt) werden ausgehend von einer Beginn-Adresse n (z.B. 32) nacheinanderliegende Bytes gelesen. Dies ermöglicht eine verkürzte Adressbildung durch Addition um 1.
- Die Adressbildung beim Burst Mode erfolgt im Cache Controller.
- Dadurch wird die CPU beim Transfer einer Cache-Line vom DRAM in das SRAM nur wenig belastet.
- Dieser Vorgang heisst *Linefill*.

- Adressiert die CPU ein Byte im DRAM um zu lesen, wird vom Cache Controller überprüft, ob sich dieses Byte in einer Cacheline befindet.

Nein Cache-Miss. Nun wird vom Cache Controller ein Linefill durchgeführt, d.h. es werden 32 Bytes (welche das adressierte Byte enthalten) in das Cache geladen.

Ja Cache-Hit. Die CPU liest das Byte aus dem Cache.

- Adressiert die CPU ein Byte im DRAM um zu schreiben, wird ebenfalls vom Cache Controller geprüft, ob sich dieses Byte im Cache befindet.

Nein Der Cache Controller schaltet den Schreibzyklus auf das DRAM durch. (*Write-Through*)

Ja Cache-Hit. Die CPU schreibt in die entsprechende Cache-Line und der Cache-Controller setzt in der entsprechenden Cacheline einen Merker (modified). Die Daten im DRAM und im Cache sind nun *unterschiedlich*. Führt ein Linefill zum Überschreiben einer Cacheline dürfen Lines mit dem Merker m nicht überschrieben werden. Sie müssen vorher vom Cache Controller in das DRAM zurückgeschrieben (*Write back*) werden und mit dem Merker invalid. versehen werden. Nur Cache-Lines mit dem Merker i dürfen überschrieben werden.

- LRU: last recently used. Es werden diejenigen Cachelines vorrangig verdrängt, die "in letzter Zeit" am wenigsten benutzt wurden.

Hier fehlt ein Bild

Abbildung 5.16: MESI-Protokoll, n CPUs mit eigenem Cache und DRAM

Hier fehlt ein Bild

Abbildung 5.17: MESI Uebergangsgraph mit den verschiedenen Zuständen der Cachelines (Automatentheorie)

- Die Datenkonsistenz zwischen Hauptspeicher (DRAM) und einem oder mehrerer Cache-Speicher regelt das MESI-Protokoll. Im MESI-Protokoll hat jede Cacheline aufgrund des Merkers verschiedene Zustände. Das MESI-Protokoll ist ausgelegt für Mehrprozessor-Systeme, die auf einen gemeinsamen Hauptspeicher zugreifen.

5.8.2 Zustände der Cache-Lines

M = modified Die Cacheline befindet sich nur in einem Cache und wurde modifiziert. Der Hauptspeicher ist nicht mehr up to date. R/W ist möglich und führt zu keinem Speicherzyklus.

E = Exclusive Die Cacheline befindet sich nur in einem Cache und wurde nicht modifiziert. Im Hauptspeicher ist also eine gültige Kopie der Line. Read möglich, Write möglich, ändert aber den Zustand nach M.

S = shared Die Cacheline befindet sich auch in anderen Caches. Read möglich. Write führt zu einem Write-Through in das DRAM. Dies bemerken die anderen Cache Controller durch die Funktion des snoopings und setzen ihre entsprechende Cacheline auf invalid.

I = invalid Die Cacheline ist ungültig. Read führt zu einem Linefill, Write zu einem Write-Through.

Erklärung:

I (Invalid) Die angesprochene Cacheline ist invalid oder garnicht im Speicher.

- Read: Cache Miss führt zu Linefill, Zustand Exklusiv
- Write: führt zu Write-Through, Zustand bleibt unverändert.

E

- Read: Zustand unverändert.
- Write: führt zu Zustand modified.

- Read Snoop: vom Cache Controller wird erkannt, dass ein anderer Prozessor auf die DRAM Line zugreift, die im Cache ist. Zustandsübergang nach S.
- Write Snoop: Im Cache Controller wird erkannt, dass ein anderer Prozessor die DRAM Line ändert, die im eigenen Cache ist. Sie wird damit wertlos. Zustandsübergang nach I.

M (Modified) Wir sind immer noch Besitzer der Cacheline

- Write Back: eine modifizierte Cacheline wird ins DRAM geschrieben. Damit wird das Cache exklusiver Besitzer der Line.
- Write Snoop: Cache Controller erkennt, dass die Cacheline im DRAM verändert wurde. Damit wird sie wertlos \rightarrow Zustand I
- Read Snoop: Cache Controller erkennt, dass ein anderer diese Cacheline liest. \rightarrow Zustand S

S (Shared) Write führt zu Write-Through und Zustandsänderung nach E. Spontan wird dies ausgeführt als Folge des Uebergangs M nach S. (gelbe Linie)

Beispiel 5.1 • *Zwei CPUs, beide haben eine Zeile vom selben SpeicherBereich, beide Shared*

- *CPU 1 schreibt zurueck*
- *CPU 2 snoopt dies und geht auf Invalid*

5.9 Multitasking (Multiprogramming)

Mehrere Tasks werden von einem Prozessor bearbeitet. Prozessorzuteilung erfolgt über Algorithmen.

5.9.1 Priorität

Task mit höchster Priorität gibt während Wartezeiten den Prozessor für Tasks niederer Priorität frei. Task mit höchster Priorität erreicht seinen Aktivzustand erst, wenn der Task mit niederer Priorität sich in den Wartezustand schaltet. Nachteil: Prioritäten sind nur wirksam, wenn sich jeder Task genügend oft in den Wartezustand schaltet.

5.9.2 Zeitscheiben

Jeder Task wird nach Ablauf der ihm zugeordneten Zeitscheibe unterbrochen und der Prozessor dem nächsten Task zugeordnet.

5.10 Multiprozessing

Mehrere Prozessoren sind an der Abarbeitung eines Tasks beteiligt.

5.10.1 Serielles Multiprocessing

Für verschiedene Aufgaben innerhalb eines Tasks werden die bestens dafür geeigneten Prozessoren ausgewählt, z.B. ein Prozessor für E/A, einer für mathematische Operationen etc.

5.10.2 Paralleles Multiprocessing

Verschiedene Aufgaben eines Tasks werden parallel (zeitgleich) in verschiedenen Prozessoren durchgeführt. Dafür müssen in der Software Voraussetzungen geschaffen sein. Jeder Prozessor hat ein eigenes Cache, alle haben einen gemeinsamen Hauptspeicher.

5.11 Befehls-Pipelines

Unter einer Befehls-Pipeline versteht man eine Folge von Schaltungsinstanzen, denen ein auszuführender Befehl der Reihe nach zugeordnet wird. Der Beitrag der jeweils nächsten Schaltungsinstanz hängt vom Ergebnis der jeweils vorigen Instanz ab. Üblich sind 5 Instanzen:

Befehls-Prefetch: Der Befehl wird aus dem Cache in einen Prefetch-Puffer übertragen und es wird seine Länge aus dem Operationscode bestimmt. Die Längenbestimmung ist notwendig, um den Beginn des jeweils nächsten Befehls zu erkennen.

Befehls-Dekodierung: Der Befehl wird erkannt und es wird festgestellt, ob für die Ausführung die Integer-ALU oder die Floating Point-ALU oder nur die Registerbänke benötigt werden.

Adress-Dekodierung: In dieser Schaltungsinstanz werden die Operanden-Adressen bestimmt und bereitgestellt.

Befehls-Ausführung: Dekodierung aus 2 und Adressbereitstellung aus 3 werden in der Integer- oder Floating Point-ALU oder in der Registerbank ausgeführt.

Write Back: Das Resultat wird ins Cache oder in die Register geschrieben.

Zur Erhöhung der Rechnerleistung schaltet man mehrere Pipes parallel. Bei der parallelen Bearbeitung von Befehlen entstehen aber logische Kollisionen bei 3, 4 und 5. Bei CPUs mit mehreren Pipes müssen Vorkehrungen für den Kollisionsfall getroffen werden. Dies geschieht durch eine Rangordnung der Pipes derart, dass im Kollisionsfall nachgeordnete Pipes angehalten werden.

Die Wirksamkeit mehrerer paralleler Pipes hängt von der Kollisionswahrscheinlichkeit ab:

- Kollisionswahrscheinlichkeit: w_k
- Gutfallwahrscheinlichkeit: $1 - w_k$
 - 2 Pipes $(1 - w_k)$
 - 3 Pipes $(1 - w_k)^3$
 - 4 Pipes $(1 - w_k)^6$

$$- 5 \text{ Pipes } (1 - w_k)^{10}$$

Beispiel: $w_k = 10\%$, $(1 - w_k) = 90\%$

- 2 Pipes $(0.9)^1 = 0.90 \cdot 2 \Rightarrow$ Leistung 1,8
- 3 Pipes $(0.9)^3 = 0.73 \cdot 3 \Rightarrow$ Leistung 2,2
- 4 Pipes $(0.9)^6 = 0.53 \cdot 4 \Rightarrow$ Leistung 2,1
- 5 Pipes $(0.9)^{10} = 0.34 \cdot 5 \Rightarrow$ Leistung 1,7

5.12 Sprungvorhersage (Branch prediction)

Bei einem bedingten Sprung gibt es zwei Möglichkeiten für den Programmablauf: Entweder wird der Sprung ausgeführt (1, taken, es folgt Befehl Alpha) oder nicht (2, not taken, es folgt Befehl Beta). Bei CPUs mit mehreren Pipes besteht die Möglichkeit

1. Sowohl den Befehl Alpha als auch den Befehl Beta in je eine Pipe zu geben
2. als auch die Möglichkeit entweder Alpha oder Beta in eine Pipe zu geben.

Im Fall a wird ein Befehl umsonst vorbereitet, denn nach Abschluss des Sprungbefehls wird entweder Alpha oder Beta durchlaufen.

Je nach Treffer ist im Fall b der falsche Befehl in der Pipe und sie muss entleert und neu durchlaufen werden. Die Trefferquote kann durch Vorhersageverfahren erhöht werden. Vorhergesagt wird durch Auswertung der Vorgeschichte.

5.13 RISC/CISC-Prozessoren

5.13.1 RISC (Reduced Instruction set Computer)

- Einfache Befehle gleicher Länge
- wenig Befehle (ca. 80)
- große Registerspeicher
- 4-5 Pipes
- große Caches (64, 128 Byte)
- Beispiele: IBM RS 6000, Sun SPARC

5.13.2 CISC (Complex Instruction set Computer)

- Komplexe Befehle unterschiedliche Länge
- deutlich mehr Befehle (bis zu 250)
- Umfangreichen Mikrocode zur Befehlsausführung
- Beispiele: Intel Pentium, IBM Mainframe Maschinen, usw.

Typischer CISC-Befehl: MOV mit Länge (R1 Pointer A, R2 Pointer B, R3 Länge). Bei RISC existiert nur der MOV für ein Maschinenwort, man müsste diese Schleife selber programmieren - kann dafür aber je nach Situation optimieren.

In der Regel kann man aber keinen deutlichen Vorteil einer RISC- gegenüber einer CISC-Maschine (und umgekehrt) angeben.

Ziel von RISC: Instruktionsausführung in einem Prozessortakt, schnelle Befehlsdekodierung weil wenig Befehle.

Entwicklung heute: Rückkehr zum CISC, grossem Cache und vielen Pipelines.

Kapitel 6

BUS-Systeme

Ein Bus-System in einem Rechner dient dazu, Daten von einer Quelle zu einer Senke zu transportieren. Quelle und Senke sind dabei nur für die Dauer eines Datentransports festgelegt.

- interne Bussysteme (on chip, on board)
- externe Bussysteme (ISA, PCI)

Quellen: Tastatur, ROM, CD-Laufwerk, RAM, CPU (Talker)

Senken: Bildschirm, Drucker, RAM, CPU (Listener)

Physikalisch besteht ein Bussystem aus Leitungen und den Interfaces der Bus-teilnehmer.

6.1 Einteilung der Bussysteme

1. nach der Logik:
 - unidirektionale / bidirektionale
 - single master, multi slave / multi master, multi slave
2. nach der Breite: 8, 16, 32 oder 64 Bit parallel
3. nach der Betriebsart:
 - synchrone / asynchrone
 - Adressen und Daten getrennt / Adressen und Daten gemultiplext

Hier fehlt ein Bild

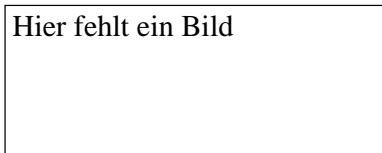


Abbildung 6.1: Blockschaltbild CPU, RAM, Drucker, Bildschirm, Interface, alle an einem Bus, jeweils mit Bustreiber

Hier fehlt ein Bild



Abbildung 6.2: 1 Master, n Slaves, alle über Daten- Adress- und Kontrollbus verbunden, jeder Bus eine Linie

Hier fehlt ein Bild




Abbildung 6.3: Zeitdiagramm Fall 1

6.2 Funktionsweise eines Bus-Systems

6.2.1 Busleitungen

- Adressleitungen
- Datenleitungen
- Steuerleitungen

6.2.2 Single Master, Multi Slave

6.2.3 Master talker, Slave listener

1. Master sendet über Adressbus die Adresse des zu schreibenden Datums und die Kennung des Slaves (Slave Select)
2. Master sendet das zu übertragende Datum über die Datenleitungen aus.
3. Master sendet über die Steuerleitungen das Signal WRITE. Mit diesem Signal übernimmt der selektierte Slave das Datum auf die Adresse.

6.2.4 Master listener, Slave talker

1. Master sendet über Adressbus die Adresse des zu lesenden Datums und die Kennung des Slaves (Slave Select)
2. Master sendet über die Steuerleitungen das Signal READ.
3. der selektierte Slave sendet daraufhin das Datum der angegebenen Adresse über die Datenleitung.

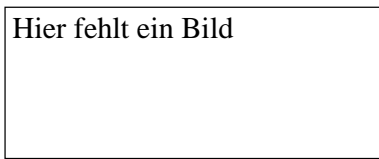


Abbildung 6.4: Zeitdiagramm Fall 2

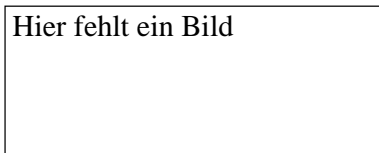


Abbildung 6.5: n Master, n Slaves, 1 Arbiter schematisch

6.2.5 Zeitdiagramme

Die Daten werden hier mit der fallenden Flanke des WRITE-Signals bzw. der steigenden Flanke des READ-Signals aktiviert. Zu diesem Zeitpunkt müssen die Signale stabil sein. Adressierung der Slaves:

Slave-Speicher enthalten viele zu adressierenden Speicherplätze, dementsprechend viele Adressleitungen werden benötigt. Slave Ein/Ausgabegeräte enthalten nur wenig zu adressierende Speicherplätze. Speicher werden über Adressen selektiert, I/O-Geräte über spezielle Steuerleitungen. Der Datenaustausch zwischen Slaves ist nur über den Master möglich.

6.3 Multi-Master / Multi-Slave

Beim Multi-Master-Bus können Konflikte auftreten wenn mehr als 1 Master auf den Bus zugreifen möchte. Der Bus-Arbiter entscheidet im Konfliktfall, wer zugreifen darf.

6.4 Funktionsweise des BUS-Arbiter nach Prioritäten

Bemerkungen: Wenn der Bus-Arbiter einem Master das Enable-Signal hat zukommen lassen muss dieses Enable-Signal während des Bus-Transfers erhalten

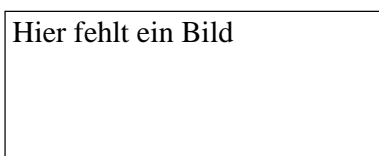


Abbildung 6.6: Funktionsweise Bus-Arbiter



Hier fehlt ein Bild

Abbildung 6.7: Cache bei synchronem Bus

bleiben. Zu diesem Zweck sind die rechten FFs da, sie werden gesperrt wenn ein anderer Master das Enable-Signal anliegen hat.

Jeder Bus-Master setzt an den Arbitrer ein eigenes Request-Signal Rq_n . Er erhält das Enable-Signal, falls kein höherpriorer Master einen Request gesetzt hat, und kein anderer Master ein Enable-Signal zugeteilt bekommen hat. Es ist jeweils nur ein Enable-Signal aktiv.

Nach Beendigung des Bus-Transfers, welchen der Master auf Grund eines Enable-Signals abwickelt, meldet der Master die Beendigung über ein RS_n -Signal. Somit können gleichzeitig anliegende Request-Signale prioritätsgerecht nacheinander abgearbeitet werden.

6.5 Synchron/asynchrone Busse

Die Betriebsart (synchron/asynchron) wird durch den Busmaster bestimmt.

6.5.1 Asynchron

Der Takt für den Datentransfer über den Bus wird vom Busmaster in Abhängigkeit vom Bedarf bestimmt. Taktlücken können bei Unterbrechungen der CPU oder bei Abarbeitung komplexer Befehle entstehen.

6.5.2 Synchron

Der Takt für den Datentransfer ist gleichmässig. Es ist aber ein Cache als Puffer nötig.

Der Datenaustausch erfolgt paketweise synchron zwischen Cache und Speicher bzw. I/O-Geräten und asynchron nach Bedarf zwischen CPU und Cache. Synchron Busse können bei gleicher Geometrie mit höherer Taktrate betrieben werden.

In Konfigurationen mit Cache und synchronem Bus erfolgen die Bus-Transfers in Paketen, d.h. die Daten werden aus aufeinanderfolgenden Speicherplätzen geholt und in aufeinanderfolgenden Speicherplätze geschrieben. Es muß also nur die Anfangsadresse angegeben werden, die Folgeadressen können im Slave und Master implizit erzeugt werden. Da hier die explizite Adressierung relativ selten vorkommt wird diese über den Datenbus vorgenommen.

6.5.3 Funktionsweise

Zu Beginn des Transfers wird über die Multiplexer-Steuerung der Bus als Adress-Bus genutzt. Es wird die Anfangsadresse übertragen. Sodann wird über die

Multiplexer-Steuerung der Bus als Datenbus benutzt. Die Adressen werden sowohl im Master als auch im Slave mit jedem Transfertakt um eins hochgezählt. (Burst-Betrieb) Vorteil: Man braucht weniger Busleitungen

6.5.4 genormte Busse

ISA (auch IBM/AT-Bus) 32 Adresse, 16 Daten, 2 Kontaktfelder, max. 20 MByte/s

EISA 64 Adressen, 32 Daten, max. 100 MByte/s, Modes: small/large

MCA Normversuch von IBM

PCI 64 Leitungen multiplex, 20 Steuerleitungen max. 264 MByte/s

Kapitel 7

Ein/Ausgabesysteme

Ein/Ausgabesysteme braucht man, um die Peripherie eines Computers zu betreiben: Platten, Floppy, Band, Drucker, Tastatur, Leitungsanschlüsse, Bildschirm

Zeichenbetrieb: Drucker, Tastatur, Leitungen über Modem

Blockbetrieb: Platte, FD, (Band), Leitungen (LAN)

Die Datenübergabe von/zum Ein/Ausgabesysteme erfolgt über Schnittstellenregister. Diese sind an das Bussystem angeschlossen. Die Adressierung der Schnittstellenregister erfolgt nach zwei Methoden:

1. Memory-mapped I/O Die Schnittstellenregister werden wie Speicherplätze adressiert. Dabei sind für Dateneingabe und Datenausgabe eigene Schnittstellenregister vorgesehen. Die Adressierung dieser Schnittstellenregister erfolgt über Adressen die außerhalb des vorhandenen Arbeitsspeichers liegen. Die Unterscheidung der I/O-Register erfolgt über die Kontrollsignale Read und Write.
2. Selective I/O: Die Schnittstellenregister werden wie eigene, unabhängige Speicher über die niederwertigen Adressbits des Adressbusses adressiert und über eigene I/O-Signale des Control-Busses selektiert. Die Adressierung der ... erfolgt wieder über Read/Write des Control-Busses.

7.1 Zeichenbetrieb

Es wird jeweils nur ein Zeichen übertragen und dann auf ein Quittungs-Signal gewartet bevor das nächste Zeichen übertragen wird (Quittung abhängig vom Parity-Bit).

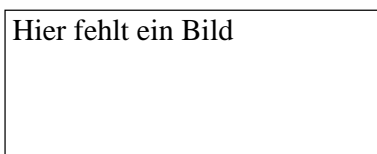


Abbildung 7.1: Blockbild Adressierbarer Adressraum j vorhandener Speicher

Hier fehlt ein Bild



Abbildung 7.2: Nullzustand 1, Startbit 0 mit startflanke, anschliessend n (8?) Schritte auf einer Zeitachse, zur Mitte des Abschnitts 'Potentialabtastung im Zeitraster'. Am Ende das/die Stopbits.

7.2 Blockbetrieb

Es wird ein Datenblock vorgesehener Länge übertragen und dann auf Quittung gewartet bevor der nächste Block übertragen wird. Quittung abhängig von der Blockcheck-Information.

7.3 Parallel/Seriell Datenconversion

Die I/O-Register werden vom Rechner stets bit-parallel bedient. Der weitere Datentransfer vom Schnittstellenregister zu einem Gerät kann sowohl bit-parallel (z.B. LPT-Schnittstelle für Drucker) als auch bit-seriell (z.B. Tastatur) erfolgen. Der Transfer über Leitungen erfolgt immer bit-seriell.

Prinzip der parallel/seriell-Datenconversion ist der gemeinsame Zeittakt - zumindest für die Dauer eines Bytes müssen die Takte in Sender und Empfänger einen Gleichlauf haben.

Für den einwandfreien Empfang des seriellen Bitstroms durch den Empfänger muss die Abtastung des Signals möglichst in der Mitte des jeweiligen Bits erfolgen. Dazu muss der Empfänger wissen:

- Den Beginn der Übertragung
- die Dauer eines Bits (Baudrate)
- Den Zeichenrahmen (Länge des Zeichens: 7, 8, oder 9 Bit)
- das Ende der Übertragung

Baudrate und Zeichenrahmen müssen vom Sender und Empfänger vor der Datenübertragung vereinbart werden. (Einstellungen). Nach der Art, wie Beginn und Ende signalisiert werden, unterscheiden wir zwischen asynchroner Übertragung und synchroner Übertragung.

7.3.1 Asynchrone Übertragung

Der Gleichlauf zwischen Sender und Empfänger wird jeweils nur fuer ein Zeichen hergestellt. Der Beginn wird durch die Startflanke eines Startbits signalisiert, das Ende durch 1, 1.5 oder 2 Stopbits.

Hier fehlt ein Bild

Abbildung 7.3: Grafik für synchrone Übertragung, SYN, STX, Daten, ETX von rNI

Hier fehlt ein Bild

Abbildung 7.4: Uebergangsgraph, Zustände 0,1,2

Ablauf der Übertragung

Der Empfänger erkennt die Startflanke, wartet daraufhin eineinhalb Bitlängen, und tastet nun 8mal im zeitlichen Abstand einer Bitlänge das Potential ab. Die dabei erkannten Signalpegel werden als die übertragenen Bits bewertet und im Schieberegister des Empfängers aufgesammelt. Nach der letzten Bewertung wartet der Empfänger eine halbe Bitlänge und prüft dann, ob 1, 1.5 oder 2 Stopbits auf der Leitung sind.

7.3.2 Synchrone Übertragung

Der Gleichlauf zwischen Sender und Empfänger wird für die Übertragung von Zeichenpaketen bis zu mehreren 1000 Zeichen hergestellt. Dies geschieht durch die Aussendung von Synchronisationszeichen (SYN) vor der eigentlichen Übertragung. Das Übertragungspaket selbst wird durch Sonderzeichen die nicht im Übertragungspaket vorkommen dürfen (STX=Start of Text und ETX=End of Text), eingeschlossen.

Erkennt ein Empfänger ein SYN-Zeichen, wird sein Zeittakt für die Abtastung die Übertragungsleitung. Trifft nun das STX-Zeichen ein, wird das Eingangssignal im Zeittakt abgetastet, die eintreffenden Bits in 8er-Portionen aufgesammelt und als Zeichen (z.B. an die CPU) übergeben. Das Erkennen des ETX-Zeichens beendet die Übertragung.

USART = universal synchronous/asynchronous receiver/transmitter

7.3.3 Anwendung asynchroner Betrieb

Zeichenweiser Betrieb, z.B. für Tastatur und Maus. Einfache Schaltung und langsame Übertragungsraten.

7.3.4 Anwendung Synchroner Betrieb

Paketbetrieb, z.B. für LAN/WAN, genauer Zeittakt erforderlich, Hohe bis höchste Übertragungsraten.

Hier fehlt ein Bild



Abbildung 7.5: einfache Schaltung 4bit D/A-Wandler

Hier fehlt ein Bild



Abbildung 7.6: Funktionsweise A/D-Wandler

7.4 Digital/Analog-Datenconversion

7.4.1 D/A

Ein Digitalwert wird in Bezug auf einen Referenzwert zugeordnet. Die durch den Digitalwert dargestellte Dualzahl ist also proportional dem Analogwert. Der maximale Digitalwert (alle Bits 1) entspricht dem Referenzwert. Anwendungen:

- Signalgeneratoren (genauer Sinus, genauer Sägezahn, genaues Dreieck)
- Sprach/Geräuschausgabe
- Analoge stellgrößen digital vorgeben

7.4.2 A/D

Einem Analogwert wird in Bezug auf einen Referenzwert ein Digitalwert zugeordnet. Der Digitalwert repräsentiert die Dualzahl, die am nächsten dem Analogwert in Bezug auf den Referenzwert proportional ist.

7.4.3 Arbeitsweise

Im Approximationsregister werden nach einem Verfahren Digitalwerte ermittelt. Diese werden über den D/A-Wandler in eine analoge Spannung umgewandelt, und diese schliesslich im Komparator mit der Eingangs-Spannung An_{In} verglichen.

Je nachdem, ob der aus dem Digitalwert gewonnene Analogwert zu gross oder zu klein ist, wird im Approximationsregister ein neuer, nun aber besserer Digitalwert erzeugt.

7.4.4 Approximationsverfahren

- hochzählen von 0 aus

- Intervallteilungsverfahren: Es wird mit der Hälfte des maximalen Digitalwertes begonnen. Je nachdem, ob zu gross oder zu klein wird das obere oder das untere Intervall weiter halbiert.

Kapitel 8

Peripheriegeräte

8.1 Der VGA-Bildschirm (Video Graphics Array)

8.1.1 Prinzip ist die Braun'sche Röhre

(Ferdinand Braun, dt. Physiker)

An der Strecke d zwischen Glühkathode und Steuerelektrode liegt eine hohe Spannung (ca. 10 kV). Die Elektronen werden auf dieser Strecke beschleunigt und treten durch die Lochblende in der Anode als freier Elektronenstrahl aus. Dieser durchläuft nun zwei aufeinander senkrecht stehende Magnetfelder H und H' . Je nach Intensität dieser Magnetfelder wird der Elektronenstrahl in H oder H' -Richtung ausgelenkt. Danach trifft der Elektronenstrahl auf die mit Phosphor beschichtete Innenseite des Bildschirms und erzeugt dort einen Lichtpunkt. Je nach Art der Phosphor-Beschichtung kann der Lichtpunkt rot, grün oder blau gemacht werden. Ausserdem leuchtet der Lichtpunkt kurze Zeit nach. Die Intensität des Lichtpunktes wird durch die an der Strecke d angelegte Spannung beeinflusst.

8.1.2 Bildaufbau

Der Bildschirm wird in Pixel (Picture Elements) eingeteilt (z.B. 1024 Pixel für die Breite und 768 Pixel für die Höhe, also 786432 insgesamt), welche der Reihe nach alle vom Elektronenstrahl abgetastet werden. Dies geschieht nach zwei Verfahren:

1. Rasterverfahren: Die Pixel werden Zeilenweise in dichter Folge abgetastet. Nach Beendigung der letzten Zeile wird wieder mit der ersten Zeile begonnen, ähnlich wie am Fernseher. Für die Ablenkung des Elektronenstrahls müssen folgende Spannungen bereitgestellt werden:
 - Horizontalablenkung: Sägezahn
 - Vertikalablenkung: Für die Periodendauer einer Horizontalablenkung fest, zwischen zwei Perioden kleiner Schritt nach unten. Nach der letzten Zeile Sprung nach oben.

2. Zeilensprungverfahren (Interlacing): Die Pixel werden nur für jede zweite Zeile geschrieben, nach der letzten Zeile kommen die ausgelassenen Zeilen. Dies ist das beim Fernsehen verwendete Prinzip. Farbbildpunkte entstehen dadurch, dass 3 Elektronenstrahlen parallel über das Bild geführt werden. Durch unterschiedliche Stärke der 3 Elektronenstrahlen leuchten die 3 Grundfarben mit unterschiedlicher Intensität.

8.1.3 Der Bildspeicher

Jedem Pixel des Bildschirms ist ein Speicherbereich im Bildspeicher zugeordnet, in dem die Intensität der Schwarz/Weiß-Information oder die Intensität der drei Grundfarb-Anteile kodiert ist.

Während eines Bilddurchlaufes (ca 1/100 s) wird der gesamte Bildspeicher ausgelesen. Die Kodierung der drei Grundfarbanteile von Digital nach Analog gewandelt. Die so entstehenden Analogspannungen werden an die drei Beschleunigungsstrecken geführt.

Von der CPU her erfolgt der Bildaufbau im Bildspeicher. Um das Auslesen des Bildspeichers durch Schreibvorgänge der CPU nicht zu verzögern, werden auf der VGA-Karte zwei Bildspeicher, ein aktueller und ein Schattenspeicher gehalten. Vom aktuellen Bildspeicher wird für den Bildaufbau ausgelesen und ein Duplikat im Schattenspeicher angelegt.

Bei Bildänderungen schreibt die CPU in den Schattenspeicher welcher nach Abschluß der Änderungen zum aktuellen Bildspeicher wird. Die beiden Bildspeicher arbeiten im Tandem-Betrieb. Für hochwertige Bewegtbild-Darstellungen werden auch mehrere Schattenspeicher benützt.

8.1.4 LCD Bildschirme (Liquid Crystal Displays)

Physikalisches Prinzip ist die Drehung der Polarisationssebene von polarisiertem Licht beim Durchgang durch einen Flüssig-Kristall, an den eine Spannung angelegt wird. Zugrunde liegt der Kerr-Effekt.

Unpolarisiertes Licht ist eine transversale elektromagnetische Schwingung, deren Amplitudenvektoren gleichmässig nach allen Richtungen verteilt sind (A). Beim Durchgang durch Polarisationsfilter 1 werden nur jene Schwingungsanteile durchgelassen, deren Amplitudenvektor in der Polarisationssebene liegt (B). Beim Durchgang dieses polarisierten Lichtes durch den Flüssigkristall wird in Abhängigkeit von der angelegten Spannung die Polarisationssebene um den Winkel α gedreht (C). Keine Spannung: $\alpha=0$, bestimmte Spannung: $\alpha=90$ Beim durchgang des gedrehten polarisierten Lichts durch den Polarisationsfilter 2 werden nur jene Amplitudenanteile durchgelassen deren Amplitudenvektor in der nun wagrechten Polarisationssebene liegt (D).

Diese Anordnung ermöglicht es also, die Intensität eines Lichtstrahles (ähnlich wie bei der Braun'schen Röhre) durch eine Spannung praktisch verzögerungsfrei zu beeinflussen. Bei $\alpha=0$ wird der Lichtstrahl abgedunkelt, bei $\alpha=90$ gelangt er ungehindert durch die Vorrichtung nach D.

Beim LCD-Bildschirm wird diese Anordnung für jeden Bildpunkt mit einem Mehrschicht-Herstellungs- Prozess (ähnlich wie bei der Chipherstellung) auf eine Platte aufgebracht. Farb-LCD's werden hergestellt indem die obige Anordnung für jeden Bildpunkt verdreifacht wird, um die drei Grundfarben Rot,

Blau und Grün mit steuerbarer Intensität auf die Glasplatte zu werfen und dadurch den Farbeffekt zu erzeugen. Die Farbfilter werden zwischen der Flüssigkristallschicht und dem zweiten Polarisationsfilter eingebaut. Die Ansteuerung der Flüssigkristall-Matrix erfolgt wie beim VGA-Bildschirm über einen D/A-Wandler und einen Bildspeicher.

8.1.5 Vergleich VGA / LCD

Leuchtkraft, Kontrast und Farbintensität ist beim VGA größer als beim LCD. Der LCD-Bildschirm ist aber absolut ruhig, da keine Helligkeitsschwankungen der Bildpunkte vorkommen.

8.2 Plattenspeicher

Jede Spur ist in Sektoren eingeteilt, jeder Sektor beginnt mit einer eigenen Sektorkennung. Anzahl und Größe der Sektoren werden bei der Platteninitialisierung festgelegt. Gelesen und geschrieben wird immer Sektorweise. Mit Rücksicht auf die Datenstruktur (große/kleine Portionen) gibt es eine optimal Sektorgröße in Bezug auf die Zugriffszeit.

8.2.1 FD-Speicher

Beim Floppy-Speicher ist das magnetische Trägermaterial auf einer flexiblen Plastikscheibe aufgebracht.

8.2.2 CD-Speicher

Die Oberfläche der Compact Disk hat unterschiedliche Reflexionseigenschaften für einen abtastenden Laserstrahl. Die unterschiedlichen Reflexionseigenschaften werden durch einen irreversiblen Brennvorgang eingebrannt.

8.3 Drucker

8.3.1 Druckbildaufbau

Mechanische Schriftdrucker

- Typenraddrucker
- Kettendrucker (Siemens)

Elektronischer Druckbildaufbau

Dabei werden Druckeranweisungen die vom Drucker kommen in ein Druckbild verwandelt.

- BitmapVerfahren: Position, Punktraster Mit einer Anweisung wird eine Position und mit weiteren Anweisungen eine Punktmatrix festgelegt.

Hier fehlt ein Bild

Abbildung 8.1: Bitmap-Funktionsweise

Die Zuordnung zwischen der Punktmatrix-Anweisung und der eigentlichen Punktmatrix bezeichnet man als Bitmap Zeichensatz. Dieser Bitmap-Zeichensatz kann entweder im Drucker fest eingestellt sein oder in diesen geladen werden.

Dieses Verfahren ist einfach: Eine Änderung der Schrift, z.B. Höhe, Breite der Buchstaben oder Fettdruck ist nur möglich indem ein neuer Zeichensatz geladen wird.

- Vektorverfahren (outline Font, Proportionalverfahren): Die Position wird wie beim Bitmap-Verfahren angegeben, das Zeichen jedoch als Anweisung in den Variablen Höhe, Breite, Strichstärke.

Das Vektorverfahren ermöglicht es, die Buchstabengröße und die Strichstärke proportional über die Variablen b , h und Strichstärke zu ändern. Jedem Zeichen ist so ein Anweisungs-Satz zugeordnet.

Dieses Verfahren ist aufwendig (Rechenintensiv), ermöglicht aber die Erzeugung hochwertiger Schriftbilder mit mehreren Schriftarten und Fettdruck. Im Drucker muss allerdings der PostScript-Zeichensatz als Firmware vorhanden sein, damit proportionale Zeichen erzeugt werden können. Hilfsweise kann der Postscript-Zeichensatz auch im Rechner gehalten und in den Drucker geladen werden. Als dritte Möglichkeit kann im Rechner der Postscript-Zeichensatz in Bitmap Anweisungen übersetzt werden (langsamer Druck).

Postscript-Eigenschaften werden aber nicht nur für die Erzeugung von Zeichen genutzt sondern auch als Seitenbeschreibungssprache für Grafiken.

Das Bild wird mittels Anweisungen, die nur die Variablen enthalten, beschrieben. Damit ist Vergrößerung und Verkleinerung möglich.

Beispiel: A4-Text-Seite hat als Pixelbild z.B. 2MB, als Postscript 30 KB

8.3.2 Druckbildumsetzung auf Papier

- Matrixdrucker: Der Druckkopf wird zeilenweise über das Papier geführt und gleichzeitig der aufgebaute Druckspeicher schritthaltend abgetastet. Je nach abgetasteter Information werden die Nadeln des Druckkopfes durch die Magneten angedrückt oder nicht.
- Laserdrucker: Kernstück des Laserdruckers ist die Photoleitertrommel. Diese hat eine Oberfläche aus besonderem Material, welches durch Laserbestrahlung lokal elektrisch entladen (Ladungsverdrängung). Auf diese Trommel wird das Bild mit Hilfe eines Lasertrahls zeilenweise geschrieben. Dabei entsteht ein Ladungsbild des Druckbilds. Je nachdem, ob das

Ladungsbild positiv oder negativ bestimmt man von Schwarz- oder Weiß-Schreiben. Bei Berührung der Trommel mit dem Toner wird an den geladenen Stellen der Trommel der Toner elektrostatisch angezogen. Auf der Trommel entsteht so das Druckbild als Tonerbild. Nun wird das Papierblatt durch eine erhitzte Walze an die Photoleitertrommel gepresst. Die schwarze Tonerfarbe färbt das Papier und die Hitze läßt das Tonerwachs schmelzen und fixiert so das Druckbild auf dem Papier.

- Tintenstrahldrucker Prinzip wie beim Matrixdrucker, die Matrix besteht jedoch aus Tintenstrahl-Düsen. An der Rückseite der Tintendüse befindet sich ein elektrischer Widerstand der durch einen Stromimpuls von einigen Mikrosekunden auf 500C erhitzt wird. Diese Dampfblase schleudert einen Tintentropfen mit hoher Geschwindigkeit aus der Düse und auf das davor befindliche Papierblatt. Farb-Tintenstrahl-Drucker haben drei Tintendüsen mit den Grundfarben.
- Farb-Thermo-Drucker: Die Druckseite wird fix auf eine Rolle gespannt und an einem mitlaufenden Farbband vorbeigeführt. Der Druckkopf geht über die ganze Seitenbreite und besteht aus punktförmig erhitzbaren Heizelementen. An einem heissen Punkt des Druckkopfes wird das Wachs des Farbbandes geschmolzen und durch die Anpresswalze mit der Farbe auf das Papier gedrückt. Nachdem die Seite für den Druck einer Farbe einmal durchgelaufen ist kommt auf dem Farbband der nächste Farbbereich. Nachdem die Druckseite für alle Farben durchlaufen ist wird sie von der Trommel abgelöst und ausgeworfen.
- Farbsublimationsdrucker: Zur Verbesserung des Farbeindrucks werden die Heizelemente des Druckkopfes unterschiedlich stark erhitzt, dadurch entstehen grosse und kleine Farbpunkte. Der Mischeindruck der Farben verbessert sich dadurch weiter. Farbsublimationsdrucker erreichen Fotoqualität.

8.4 Scanner

Beim Scannen wird eine lichtempfindliche Diodenzeile über das beleuchtete Bild geführt. Das unterschiedlich stark reflektierte Licht bewirkt in den Dioden unterschiedliche Spannungen. Diese Spannungen werden digitalisiert und als Bildinformation gespeichert.

8.5 Lokalnetze

In Lokalnetzen¹ sind Entfernungen um die 10km und Übertragungsgeschwindigkeiten um 20 MBit/sec möglich.

Zeitvergleich:

4800 Bd	600 Bytes/sec	100 Bytes in 0.16 sec
20 MBd	2.5 MBytes/sec	100 Bytes in 0.00004 sec

Im letzteren Fall sinkt also die Kollisionswahrscheinlichkeit.

¹local area networks, LAN's

8.5.1 CSMA - Bus

CS vor der Sendung abhören ob der Bus belegt ist

MA mehrere greifen unabhängig voneinander auf den Bus zu. Die Stationen sind gleichberechtigt.

CD Es muß eine Maximalentfernung sowie eine minimale Sendungsdauer vorgeschrieben sein, um Zusammenstöße zu erkennen.

Nach CD (Collision Detection) wird JAM (HF-Signal) gesendet zur gegenseitigen Verständigung über einen Zusammenstoß.

Vorteile:

- einfach
- Keine Störung bei Stationsausfall

Nachteil:

- zunehmende Zusammenstöße bei steigender Last

8.5.2 Token Ring

Stationen sind ebenfalls gleichberechtigt. Eine Monitorstation hat jedoch in bestimmten Phasen Sonderaufgaben:

- Generieren eines freien Tokens
- Eingriffe im Fehlerfall

Nachrichten werden in jeder Station empfangen, gespeichert und weitergesendet. Betriebsweise:

- Nachdem ein freies Token vom Monitor erzeugt wurde, kreist dieses
- eine sendewillige Station verändert das freie Token in ein belegtes und hängt seine Nachricht an.
- Die Empfangsstation merkt sich die Nachricht und markiert das Token mit "Nachricht erhalten"
- Die Sendestation löscht die Nachricht und markiert das Token als frei

Vorteile:

- Durchsatzgarantie im Hochlastbereich
- erweiterbar

Nachteile:

- überflüssige Wartezeiten
- eine gestörte Station legt den Ring lahm

Hub Netze (hub = Nabe) (auch Sternnetz) Der Datentransfer von einem Teilnehmer zum anderen erfolgt in Blöcken, z.B. A sendet nach C. Der erste Block wird erst dann quittiert wenn dieser erfolgreich an C weitergegeben ist. Es können mehrere Übertragungen gleichzeitig laufen.

Repeater Das Leitungssignal wird aufgefrischt ohne Veränderung der Daten.

Bridge verbindet LAN-Netze mit gleichartigem Übertragungsverfahren. Es erfolgt eine Zwischenspeicherung der Daten. Aus der Sicht eines jeden LAN-Netzes ist die Bridge ein Teilnehmer im eigenen Netz.

Gateway funktioniert wie eine Bridge, jedoch sind Protokolltransformationen möglich.

Router speichert Datenpakete temporär und leitet sie anhand von Adressen die im Datenpaket enthalten sind weiter.

Kapitel 9

BUS-Systeme

Ein Bus-System in einem Rechner dient dazu, Daten von einer Quelle zu einer Senke zu transportieren. Quelle und Senke sind dabei nur für die Dauer eines Datentransports festgelegt.

- interne Bussysteme (on chip, on board)
- externe Bussysteme (ISA, PCI)

Quellen: Tastatur, ROM, CD-Laufwerk, RAM, CPU (Talker)

Senken: Bildschirm, Drucker, RAM, CPU (Listener)

Physikalisch besteht ein Bussystem aus Leitungen und den Interfaces der Bus-teilnehmer.

9.1 Einteilung der Bussysteme

1. nach der Logik:
 - unidirektionale / bidirektionale
 - single master, multi slave / multi master, multi slave
2. nach der Breite: 8, 16, 32 oder 64 Bit parallel
3. nach der Betriebsart:
 - synchrone / asynchrone
 - Adressen und Daten getrennt / Adressen und Daten gemultiplext

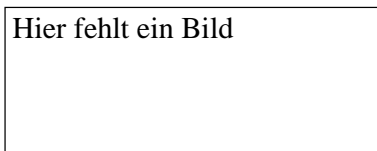


Abbildung 9.1: Blockschaltbild CPU, RAM, Drucker, Bildschirm, Interface, alle an einem Bus, jeweils mit Bustreiber

Hier fehlt ein Bild



Abbildung 9.2: 1 Master, n Slaves, alle über Daten- Adress- und Kontrollbus verbunden, jeder Bus eine Linie

Hier fehlt ein Bild



Abbildung 9.3: Zeitdiagramm Fall 1

9.2 Funktionsweise eines Bus-Systems

9.2.1 Busleitungen

- Adressleitungen
- Datenleitungen
- Steuerleitungen

9.2.2 Single Master, Multi Slave

9.2.3 Master talker, Slave listener

1. Master sendet über Adressbus die Adresse des zu schreibenden Datums und die Kennung des Slaves (Slave Select)
2. Master sendet das zu übertragende Datum über die Datenleitungen aus.
3. Master sendet über die Steuerleitungen das Signal WRITE. Mit diesem Signal übernimmt der selektierte Slave das Datum auf die Adresse.

9.2.4 Master listener, Slave talker

1. Master sendet über Adressbus die Adresse des zu lesenden Datums und die Kennung des Slaves (Slave Select)
2. Master sendet über die Steuerleitungen das Signal READ.
3. der selektierte Slave sendet daraufhin das Datum der angegebenen Adresse über die Datenleitung.

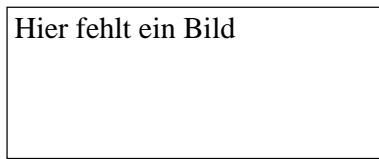


Abbildung 9.4: Zeitdiagramm Fall 2

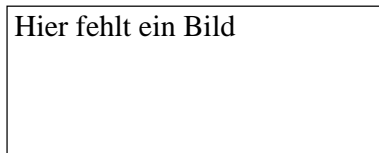


Abbildung 9.5: n Master, n Slaves, 1 Arbiter schematisch

9.2.5 Zeitdiagramme

Die Daten werden hier mit der fallenden Flanke des WRITE-Signals bzw. der steigenden Flanke des READ-Signals aktiviert. Zu diesem Zeitpunkt müssen die Signale stabil sein. Adressierung der Slaves:

Slave-Speicher enthalten viele zu adressierenden Speicherplätze, dementsprechend viele Adressleitungen werden benötigt. Slave Ein/Ausgabegeräte enthalten nur wenig zu adressierende Speicherplätze. Speicher werden über Adressen selektiert, I/O-Geräte über spezielle Steuerleitungen. Der Datenaustausch zwischen Slaves ist nur über den Master möglich.

9.3 Multi-Master / Multi-Slave

Beim Multi-Master-Bus können Konflikte auftreten wenn mehr als 1 Master auf den Bus zugreifen möchte. Der Bus-Arbiter entscheidet im Konfliktfall, wer zugreifen darf.

9.4 Funktionsweise des BUS-Arbiter nach Prioritäten

Bemerkungen: Wenn der Bus-Arbiter einem Master das Enable-Signal hat zukommen lassen muss dieses Enable-Signal während des Bus-Transfers erhalten

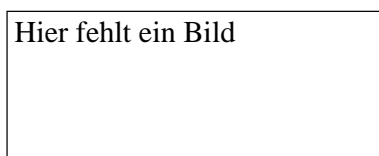


Abbildung 9.6: Funktionsweise Bus-Arbiter



Hier fehlt ein Bild

Abbildung 9.7: Cache bei synchronem Bus

bleiben. Zu diesem Zweck sind die rechten FFs da, sie werden gesperrt wenn ein anderer Master das Enable-Signal anliegen hat.

Jeder Bus-Master setzt an den Arbiter ein eigenes Request-Signal Rq_n . Er erhält das Enable-Signal, falls kein höherpriorer Master einen Request gesetzt hat, und kein anderer Master ein Enable-Signal zugeteilt bekommen hat. Es ist jeweils nur ein Enable-Signal aktiv.

Nach Beendigung des Bus-Transfers, welchen der Master auf Grund eines Enable-Signals abwickelt, meldet der Master die Beendigung über ein RS_n -Signal. Somit können gleichzeitig anliegende Request-Signale prioritätsgerecht nacheinander abgearbeitet werden.

9.5 Synchron/asynchrone Busse

Die Betriebsart (synchron/asynchron) wird durch den Busmaster bestimmt.

9.5.1 Asynchron

Der Takt für den Datentransfer über den Bus wird vom Busmaster in Abhängigkeit vom Bedarf bestimmt. Taktlücken können bei Unterbrechungen der CPU oder bei Abarbeitung komplexer Befehle entstehen.

9.5.2 Synchron

Der Takt für den Datentransfer ist gleichmässig. Es ist aber ein Cache als Puffer nötig.

Der Datenaustausch erfolgt paketweise synchron zwischen Cache und Speicher bzw. I/O-Geräten und asynchron nach Bedarf zwischen CPU und Cache. Synchron Busse können bei gleicher Geometrie mit höherer Taktrate betrieben werden.

In Konfigurationen mit Cache und synchronem Bus erfolgen die Bus-Transfers in Paketen, d.h. die Daten werden aus aufeinanderfolgenden Speicherplätzen geholt und in aufeinanderfolgenden Speicherplätze geschrieben. Es muß also nur die Anfangsadresse angegeben werden, die Folgeadressen können im Slave und Master implizit erzeugt werden. Da hier die explizite Adressierung relativ selten vorkommt wird diese über den Datenbus vorgenommen.

9.5.3 Funktionsweise

Zu Beginn des Transfers wird über die Multiplexer-Steuerung der Bus als Adress-Bus genutzt. Es wird die Anfangsadresse übertragen. Sodann wird über die

Multiplexer-Steuerung der Bus als Datenbus benutzt. Die Adressen werden sowohl im Master als auch im Slave mit jedem Transfertakt um eins hochgezählt. (Burst-Betrieb) Vorteil: Man braucht weniger Busleitungen

9.5.4 genormte Busse

ISA (auch IBM/AT-Bus) 32 Adresse, 16 Daten, 2 Kontaktfelder, max. 20 MByte/s

EISA 64 Adressen, 32 Daten, max. 100 MByte/s, Modes: small/large

MCA Normversuch von IBM

PCI 64 Leitungen multiplex, 20 Steuerleitungen max. 264 MByte/s

Kapitel 10

Ein/Ausgabesysteme

Ein/Ausgabesysteme braucht man, um die Peripherie eines Computers zu betreiben: Platten, Floppy, Band, Drucker, Tastatur, Leitungsanschlüsse, Bildschirm

Zeichenbetrieb: Drucker, Tastatur, Leitungen über Modem

Blockbetrieb: Platte, FD, (Band), Leitungen (LAN)

Die Datenübergabe von/zum Ein/Ausgabesysteme erfolgt über Schnittstellenregister. Diese sind an das Bussystem angeschlossen. Die Adressierung der Schnittstellenregister erfolgt nach zwei Methoden:

1. Memory-mapped I/O Die Schnittstellenregister werden wie Speicherplätze adressiert. Dabei sind für Dateneingabe und Datenausgabe eigene Schnittstellenregister vorgesehen. Die Adressierung dieser Schnittstellenregister erfolgt über Adressen die außerhalb des vorhandenen Arbeitsspeichers liegen. Die Unterscheidung der I/O-Register erfolgt über die Kontrollsignale Read und Write.
2. Selective I/O: Die Schnittstellenregister werden wie eigene, unabhängige Speicher über die niederwertigen Adressbits des Adressbusses adressiert und über eigene I/O-Signale des Control-Busses selektiert. Die Adressierung der ... erfolgt wieder über Read/Write des Control-Busses.

10.1 Zeichenbetrieb

Es wird jeweils nur ein Zeichen übertragen und dann auf ein Quittungs-Signal gewartet bevor das nächste Zeichen übertragen wird (Quittung abhängig vom Parity-Bit).

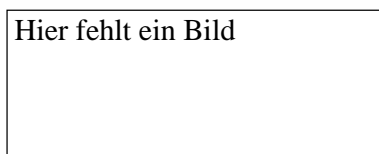


Abbildung 10.1: Blockbild Adressierbarer Adressraum j vorhandener Speicher



Hier fehlt ein Bild

Abbildung 10.2: Nullzustand 1, Startbit 0 mit startflanke, anschliessend n (8?) Schritte auf einer Zeitachse, zur Mitte des Abschnitts 'Potentialabtastung im Zeitraster'. Am Ende das/die Stopbits.

10.2 Blockbetrieb

Es wird ein Datenblock vorgesehener Länge übertragen und dann auf Quittung gewartet bevor der nächste Block übertragen wird. Quittung abhängig von der Blockcheck-Information.

10.3 Parallel/Seriell Datenconversion

Die I/O-Register werden vom Rechner stets bit-parallel bedient. Der weitere Datentransfer vom Schnittstellenregister zu einem Gerät kann sowohl bit-parallel (z.B. LPT-Schnittstelle für Drucker) als auch bit-seriell (z.B. Tastatur) erfolgen. Der Transfer über Leitungen erfolgt immer bit-seriell.

Prinzip der parallel/seriell-Datenconversion ist der gemeinsame Zeittakt - zumindest für die Dauer eines Bytes müssen die Takte in Sender und Empfänger einen Gleichlauf haben.

Für den einwandfreien Empfang des seriellen Bitstroms durch den Empfänger muss die Abtastung des Signals möglichst in der Mitte des jeweiligen Bits erfolgen. Dazu muss der Empfänger wissen:

- Den Beginn der Übertragung
- die Dauer eines Bits (Baudrate)
- Den Zeichenrahmen (Länge des Zeichens: 7, 8, oder 9 Bit)
- das Ende der Übertragung

Baudrate und Zeichenrahmen müssen vom Sender und Empfänger vor der Datenübertragung vereinbart werden. (Einstellungen). Nach der Art, wie Beginn und Ende signalisiert werden, unterscheiden wir zwischen asynchroner Übertragung und synchroner Übertragung.

10.3.1 Asynchrone Übertragung

Der Gleichlauf zwischen Sender und Empfänger wird jeweils nur fuer ein Zeichen hergestellt. Der Beginn wird durch die Startflanke eines Startbits signalisiert, das Ende durch 1, 1.5 oder 2 Stopbits.

Hier fehlt ein Bild

Abbildung 10.3: Grafik für synchrone Übertragung, SYN, STX, Daten, ETX von rNI

Hier fehlt ein Bild

Abbildung 10.4: Uebergangsgraph, Zustände 0,1,2

Ablauf der Übertragung

Der Empfänger erkennt die Startflanke, wartet daraufhin eineinhalb Bitlängen, und tastet nun 8mal im zeitlichen Abstand einer Bitlänge das Potential ab. Die dabei erkannten Signalpegel werden als die übertragenen Bits bewertet und im Schieberegister des Empfängers aufgesammelt. Nach der letzten Bewertung wartet der Empfänger eine halbe Bitlänge und prüft dann, ob 1, 1.5 oder 2 Stopbits auf der Leitung sind.

10.3.2 Synchrone Übertragung

Der Gleichlauf zwischen Sender und Empfänger wird für die Übertragung von Zeichenpaketen bis zu mehreren 1000 Zeichen hergestellt. Dies geschieht durch die Aussendung von Synchronisationszeichen (SYN) vor der eigentlichen Übertragung. Das Übertragungspaket selbst wird durch Sonderzeichen die nicht im Übertragungspaket vorkommen dürfen (STX=Start of Text und ETX=End of Text), eingeschlossen.

Erkennt ein Empfänger ein SYN-Zeichen, wird sein Zeittakt für die Abtastung die Übertragungsleitung. Trifft nun das STX-Zeichen ein, wird das Eingangssignal im Zeittakt abgetastet, die eintreffenden Bits in 8er-Portionen aufgesammelt und als Zeichen (z.B. an die CPU) übergeben. Das Erkennen des ETX-Zeichens beendet die Übertragung.

USART = universal synchronous/asynchronous receiver/transmitter

10.3.3 Anwendung asynchroner Betrieb

Zeichenweiser Betrieb, z.B. für Tastatur und Maus. Einfache Schaltung und langsame Übertragungsraten.

10.3.4 Anwendung Synchroner Betrieb

Paketbetrieb, z.B. für LAN/WAN, genauer Zeittakt erforderlich, Hohe bis höchste Übertragungsraten.

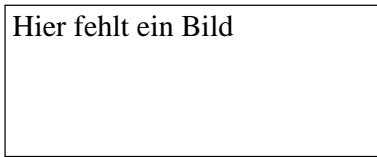


Abbildung 10.5: einfache Schaltung 4bit D/A-Wandler

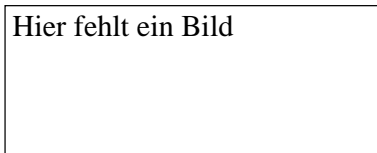


Abbildung 10.6: Funktionsweise A/D-Wandler

10.4 Digital/Analog-Datenconversion

10.4.1 D/A

Ein Digitalwert wird in Bezug auf einen Referenzwert zugeordnet. Die durch den Digitalwert dargestellte Dualzahl ist also proportional dem Analogwert. Der maximale Digitalwert (alle Bits 1) entspricht dem Referenzwert. Anwendungen:

- Signalgeneratoren (genauer Sinus, genauer Sägezahn, genaues Dreieck)
- Sprach/Geräuschausgabe
- Analoge stellgrößen digital vorgeben

10.4.2 A/D

Einem Analogwert wird in Bezug auf einen Referenzwert ein Digitalwert zugeordnet. Der Digitalwert repräsentiert die Dualzahl, die am nächsten dem Analogwert in Bezug auf den Referenzwert proportional ist.

10.4.3 Arbeitsweise

Im Approximationsregister werden nach einem Verfahren Digitalwerte ermittelt. Diese werden über den D/A-Wandler in eine analoge Spannung umgewandelt, und diese schliesslich im Komparator mit der Eingangs-Spannung $AnIn$ verglichen.

Je nachdem, ob der aus dem Digitalwert gewonnene Analogwert zu gross oder zu klein ist, wird im Approximationsregister ein neuer, nun aber besserer Digitalwert erzeugt.

10.4.4 Approximationsverfahren

- hochzählen von 0 aus

- Intervallteilungsverfahren: Es wird mit der Hälfte des maximalen Digitalwertes begonnen. Je nachdem, ob zu gross oder zu klein wird das obere oder das untere Intervall weiter halbiert.

Kapitel 11

Peripheriegeräte

11.1 Der VGA-Bildschirm (Video Graphics Array)

11.1.1 Prinzip ist die Braun'sche Röhre

(Ferdinand Braun, dt. Physiker)

An der Strecke d zwischen Glühkathode und Steuerelektrode liegt eine hohe Spannung (ca. 10 kV). Die Elektronen werden auf dieser Strecke beschleunigt und treten durch die Lochblende in der Anode als freier Elektronenstrahl aus. Dieser durchläuft nun zwei aufeinander senkrecht stehende Magnetfelder H und H' . Je nach Intensität dieser Magnetfelder wird der Elektronenstrahl in H oder H' -Richtung ausgelenkt. Danach trifft der Elektronenstrahl auf die mit Phosphor beschichtete Innenseite des Bildschirms und erzeugt dort einen Lichtpunkt. Je nach Art der Phosphor-Beschichtung kann der Lichtpunkt rot, grün oder blau gemacht werden. Ausserdem leuchtet der Lichtpunkt kurze Zeit nach. Die Intensität des Lichtpunktes wird durch die an der Strecke d angelegte Spannung beeinflusst.

11.1.2 Bildaufbau

Der Bildschirm wird in Pixel (Picture Elements) eingeteilt (z.B. 1024 Pixel für die Breite und 768 Pixel für die Höhe, also 786432 insgesamt), welche der Reihe nach alle vom Elektronenstrahl abgetastet werden. Dies geschieht nach zwei Verfahren:

1. Rasterverfahren: Die Pixel werden Zeilenweise in dichter Folge abgetastet. Nach Beendigung der letzten Zeile wird wieder mit der ersten Zeile begonnen, ähnlich wie am Fernseher. Für die Ablenkung des Elektronenstrahls müssen folgende Spannungen bereitgestellt werden:
 - Horizontalablenkung: Sägezahn
 - Vertikalablenkung: Für die Periodendauer einer Horizontalablenkung fest, zwischen zwei Perioden kleiner Schritt nach unten. Nach der letzten Zeile Sprung nach oben.

2. Zeilensprungverfahren (Interlacing): Die Pixel werden nur für jede zweite Zeile geschrieben, nach der letzten Zeile kommen die ausgelassenen Zeilen. Dies ist das beim Fernsehen verwendete Prinzip. Farbbildpunkte entstehen dadurch, dass 3 Elektronenstrahlen parallel über das Bild geführt werden. Durch unterschiedliche Stärke der 3 Elektronenstrahlen leuchten die 3 Grundfarben mit unterschiedlicher Intensität.

11.1.3 Der Bildspeicher

Jedem Pixel des Bildschirms ist ein Speicherbereich im Bildspeicher zugeordnet, in dem die Intensität der Schwarz/Weiß-Information oder die Intensität der drei Grundfarb-Anteile kodiert ist.

Während eines Bilddurchlaufes (ca 1/100 s) wird der gesamte Bildspeicher ausgelesen. Die Kodierung der drei Grundfarbanteile von Digital nach Analog gewandelt. Die so entstehenden Analogspannungen werden an die drei Beschleunigungsstrecken geführt.

Von der CPU her erfolgt der Bildaufbau im Bildspeicher. Um das Auslesen des Bildspeichers durch Schreibvorgänge der CPU nicht zu verzögern, werden auf der VGA-Karte zwei Bildspeicher, ein aktueller und ein Schattenspeicher gehalten. Vom aktuellen Bildspeicher wird für den Bildaufbau ausgelesen und ein Duplikat im Schattenspeicher angelegt.

Bei Bildänderungen schreibt die CPU in den Schattenspeicher welcher nach Abschluß der Änderungen zum aktuellen Bildspeicher wird. Die beiden Bildspeicher arbeiten im Tandem-Betrieb. Für hochwertige Bewegtbild-Darstellungen werden auch mehrere Schattenspeicher benützt.

11.1.4 LCD Bildschirme (Liquid Crystal Displays)

Physikalisches Prinzip ist die Drehung der Polarisationssebene von polarisiertem Licht beim Durchgang durch einen Flüssig-Kristall, an den eine Spannung angelegt wird. Zugrunde liegt der Kerr-Effekt.

Unpolarisiertes Licht ist eine transversale elektromagnetische Schwingung, deren Amplitudenvektoren gleichmäßig nach allen Richtungen verteilt sind (A). Beim Durchgang durch Polarisationsfilter 1 werden nur jene Schwingungsanteile durchgelassen, deren Amplitudenvektor in der Polarisationssebene liegt (B). Beim Durchgang dieses polarisierten Lichtes durch den Flüssigkristall wird in Abhängigkeit von der angelegten Spannung die Polarisationssebene um den Winkel α gedreht (C). Keine Spannung: $\alpha=0$, bestimmte Spannung: $\alpha=90$ Beim durchgang des gedrehten polarisierten Lichts durch den Polarisationsfilter 2 werden nur jene Amplitudenanteile durchgelassen deren Amplitudenvektor in der nun wagrechten Polarisationssebene liegt (D).

Diese Anordnung ermöglicht es also, die Intensität eines Lichtstrahles (ähnlich wie bei der Braun'schen Röhre) durch eine Spannung praktisch verzögerungsfrei zu beeinflussen. Bei $\alpha=0$ wird der Lichtstrahl abgedunkelt, bei $\alpha=90$ gelangt er ungehindert durch die Vorrichtung nach D.

Beim LCD-Bildschirm wird diese Anordnung für jeden Bildpunkt mit einem Mehrschicht-Herstellungs- Prozess (ähnlich wie bei der Chipherstellung) auf eine Platte aufgebracht. Farb-LCD's werden hergestellt indem die obige Anordnung für jeden Bildpunkt verdreifacht wird, um die drei Grundfarben Rot,

Blau und Grün mit steuerbarer Intensität auf die Glasplatte zu werfen und dadurch den Farbeffekt zu erzeugen. Die Farbfilter werden zwischen der Flüssigkristallschicht und dem zweiten Polarisationsfilter eingebaut. Die Ansteuerung der Flüssigkristall-Matrix erfolgt wie beim VGA-Bildschirm über einen D/A-Wandler und einen Bildspeicher.

11.1.5 Vergleich VGA / LCD

Leuchtkraft, Kontrast und Farbintensität ist beim VGA größer als beim LCD. Der LCD-Bildschirm ist aber absolut ruhig, da keine Helligkeitsschwankungen der Bildpunkte vorkommen.

11.2 Plattenspeicher

Jede Spur ist in Sektoren eingeteilt, jeder Sektor beginnt mit einer eigenen Sektorkennung. Anzahl und Größe der Sektoren werden bei der Platteninitialisierung festgelegt. Gelesen und geschrieben wird immer Sektorweise. Mit Rücksicht auf die Datenstruktur (große/kleine Portionen) gibt es eine optimal Sektorgröße in Bezug auf die Zugriffszeit.

11.2.1 FD-Speicher

Beim Floppy-Speicher ist das magnetische Trägermaterial auf einer flexiblen Plastikscheibe aufgebracht.

11.2.2 CD-Speicher

Die Oberfläche der Compact Disk hat unterschiedliche Reflexionseigenschaften für einen abtastenden Laserstrahl. Die unterschiedlichen Reflexionseigenschaften werden durch einen irreversiblen Brennvorgang eingebrannt.

11.3 Drucker

11.3.1 Druckbildaufbau

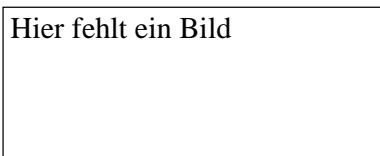
Mechanische Schriftdrucker

- Typenraddrucker
- Kettendrucker (Siemens)

Elektronischer Druckbildaufbau

Dabei werden Druckeranweisungen die vom Drucker kommen in ein Druckbild verwandelt.

- BitmapVerfahren: Position, Punktraster Mit einer Anweisung wird eine Position und mit weiteren Anweisungen eine Punktmatrix festgelegt.



Hier fehlt ein Bild

Abbildung 11.1: Bitmap-Funktionsweise

Die Zuordnung zwischen der Punktmatrix-Anweisung und der eigentlichen Punktmatrix bezeichnet man als Bitmap Zeichensatz. Dieser Bitmap-Zeichensatz kann entweder im Drucker fest eingestellt sein oder in diesen geladen werden.

Dieses Verfahren ist einfach: Eine Änderung der Schrift, z.B. Höhe, Breite der Buchstaben oder Fettdruck ist nur möglich indem ein neuer Zeichensatz geladen wird.

- Vektorverfahren (outline Font, Proportionalverfahren): Die Position wird wie beim Bitmap-Verfahren angegeben, das Zeichen jedoch als Anweisung in den Variablen Höhe, Breite, Strichstärke.

Das Vektorverfahren ermöglicht es, die Buchstabengröße und die Strichstärke proportional über die Variablen b , h und Strichstärke zu ändern. Jedem Zeichen ist so ein Anweisungs-Satz zugeordnet.

Dieses Verfahren ist aufwendig (Rechenintensiv), ermöglicht aber die Erzeugung hochwertiger Schriftbilder mit mehreren Schriftarten und Fettdruck. Im Drucker muss allerdings der PostScript-Zeichensatz als Firmware vorhanden sein, damit proportionale Zeichen erzeugt werden können. Hilfsweise kann der Postscript-Zeichensatz auch im Rechner gehalten und in den Drucker geladen werden. Als dritte Möglichkeit kann im Rechner der Postscript-Zeichensatz in Bitmap Anweisungen übersetzt werden (langsamer Druck).

Postscript-Eigenschaften werden aber nicht nur für die Erzeugung von Zeichen genutzt sondern auch als Seitenbeschreibungssprache für Grafiken.

Das Bild wird mittels Anweisungen, die nur die Variablen enthalten, beschrieben. Damit ist Vergrößerung und Verkleinerung möglich.

Beispiel: A4-Text-Seite hat als Pixelbild z.B. 2MB, als Postscript 30 KB

11.3.2 Druckbildumsetzung auf Papier

- Matrixdrucker: Der Druckkopf wird zeilenweise über das Papier geführt und gleichzeitig der aufgebaute Druckspeicher schritthaltend abgetastet. Je nach abgetasteter Information werden die Nadeln des Druckkopfes durch die Magneten angedrückt oder nicht.
- Laserdrucker: Kernstück des Laserdruckers ist die Photoleitertrommel. Diese hat eine Oberfläche aus besonderem Material, welches durch Laserbestrahlung lokal elektrisch entladen (Ladungsverdrängung). Auf diese Trommel wird das Bild mit Hilfe eines Lasertrahls zeilenweise geschrieben. Dabei entsteht ein Ladungsbild des Druckbilds. Je nachdem, ob das

Ladungsbild positiv oder negativ bestimmt man von Schwarz- oder Weiß-Schreiben. Bei Berührung der Trommel mit dem Toner wird an den geladenen Stellen der Trommel der Toner elektrostatisch angezogen. Auf der Trommel entsteht so das Druckbild als Tonerbild. Nun wird das Papierblatt durch eine erhitzte Walze an die Photoleitertrommel gepresst. Die schwarze Tonerfarbe färbt das Papier und die Hitze läßt das Tonerwachs schmelzen und fixiert so das Druckbild auf dem Papier.

- Tintenstrahldrucker Prinzip wie beim Matrixdrucker, die Matrix besteht jedoch aus Tintenstrahl-Düsen. An der Rückseite der Tintendüse befindet sich ein elektrischer Widerstand der durch einen Stromimpuls von einigen Mikrosekunden auf 500C erhitzt wird. Diese Dampfblase schleudert einen Tintentropfen mit hoher Geschwindigkeit aus der Düse und auf das davor befindliche Papierblatt. Farb-Tintenstrahl-Drucker haben drei Tintendüsen mit den Grundfarben.
- Farb-Thermo-Drucker: Die Druckseite wird fix auf eine Rolle gespannt und an einem mitlaufenden Farbband vorbeigeführt. Der Druckkopf geht über die ganze Seitenbreite und besteht aus punktförmig erhitzbaren Heizelementen. An einem heissen Punkt des Druckkopfes wird das Wachs des Farbbandes geschmolzen und durch die Anpresswalze mit der Farbe auf das Papier gedrückt. Nachdem die Seite für den Druck einer Farbe einmal durchgelaufen ist kommt auf dem Farbband der nächste Farbbereich. Nachdem die Druckseite für alle Farben durchlaufen ist wird sie von der Trommel abgelöst und ausgeworfen.
- Farbsublimationsdrucker: Zur Verbesserung des Farbeindrucks werden die Heizelemente des Druckkopfes unterschiedlich stark erhitzt, dadurch entstehen grosse und kleine Farbpunkte. Der Mischeindruck der Farben verbessert sich dadurch weiter. Farbsublimationsdrucker erreichen Fotoqualität.

11.4 Scanner

Beim Scannen wird eine lichtempfindliche Diodenzeile über das beleuchtete Bild geführt. Das unterschiedlich stark reflektierte Licht bewirkt in den Dioden unterschiedliche Spannungen. Diese Spannungen werden digitalisiert und als Bildinformation gespeichert.

11.5 Lokalnetze

In Lokalnetzen¹ sind Entfernungen um die 10km und Übertragungsgeschwindigkeiten um 20 MBit/sec möglich.

Zeitvergleich:

4800 Bd	600 Bytes/sec	100 Bytes in 0.16 sec
20 MBd	2.5 MBytes/sec	100 Bytes in 0.00004 sec

Im letzteren Fall sinkt also die Kollisionswahrscheinlichkeit.

¹local area networks, LAN's

11.5.1 CSMA - Bus

CS vor der Sendung abhören ob der Bus belegt ist

MA mehrere greifen unabhängig voneinander auf den Bus zu. Die Stationen sind gleichberechtigt.

CD Es muß eine Maximalentfernung sowie eine minimale Sendungsdauer vorgeschrieben sein, um Zusammenstöße zu erkennen.

Nach CD (Collision Detection) wird JAM (HF-Signal) gesendet zur gegenseitigen Verständigung über einen Zusammenstoß.

Vorteile:

- einfach
- Keine Störung bei Stationsausfall

Nachteil:

- zunehmende Zusammenstöße bei steigender Last

11.5.2 Token Ring

Stationen sind ebenfalls gleichberechtigt. Eine Monitorstation hat jedoch in bestimmten Phasen Sonderaufgaben:

- Generieren eines freien Tokens
- Eingriffe im Fehlerfall

Nachrichten werden in jeder Station empfangen, gespeichert und weitergesendet. Betriebsweise:

- Nachdem ein freies Token vom Monitor erzeugt wurde, kreist dieses
- eine sendewillige Station verändert das freie Token in ein belegtes und hängt seine Nachricht an.
- Die Empfangsstation merkt sich die Nachricht und markiert das Token mit "Nachricht erhalten"
- Die Sendestation löscht die Nachricht und markiert das Token als frei

Vorteile:

- Durchsatzgarantie im Hochlastbereich
- erweiterbar

Nachteile:

- überflüssige Wartezeiten
- eine gestörte Station legt den Ring lahm

Hub Netze (hub = Nabe) (auch Sternnetz) Der Datentransfer von einem Teilnehmer zum anderen erfolgt in Blöcken, z.B. A sendet nach C. Der erste Block wird erst dann quittiert wenn dieser erfolgreich an C weitergegeben ist. Es können mehrere Übertragungen gleichzeitig laufen.

Repeater Das Leitungssignal wird aufgefrischt ohne Veränderung der Daten.

Bridge verbindet LAN-Netze mit gleichartigem Übertragungsverfahren. Es erfolgt eine Zwischenspeicherung der Daten. Aus der Sicht eines jeden LAN-Netzes ist die Bridge ein Teilnehmer im eigenen Netz.

Gateway funktioniert wie eine Bridge, jedoch sind Protokolltransformationen möglich.

Router speichert Datenpakete temporär und leitet sie anhand von Adressen die im Datenpaket enthalten sind weiter.

Kapitel 12

Der 8051

12.1 Blockschaltbild des 8051

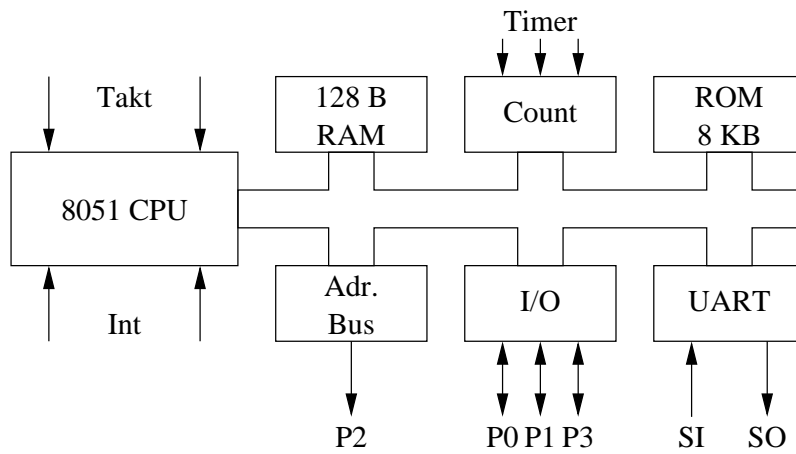


Abbildung 12.1: Blockschaltbild des 8051

12.1.1 Anschlüsse des 8051

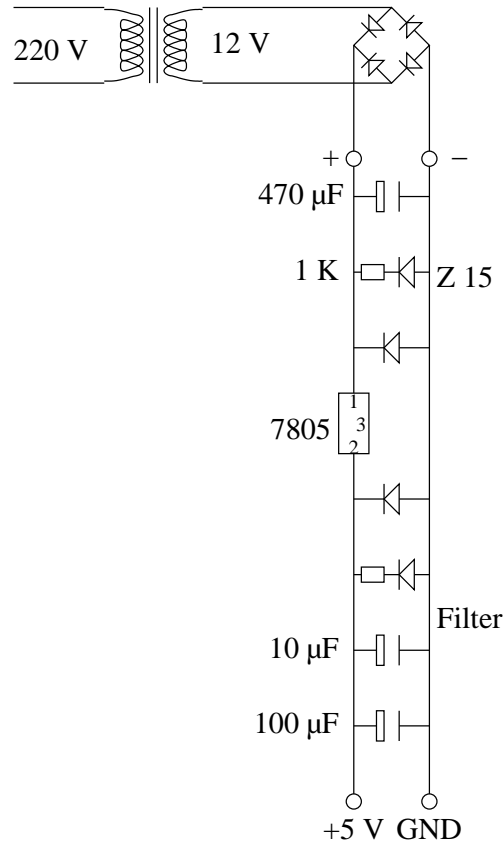


Abbildung 12.2: Anschlüsse des 8051

12.1.2 XTAL 1 / XTAL 2 Prozessortakt

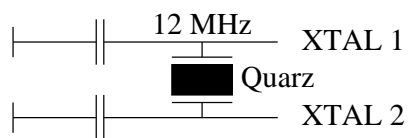


Abbildung 12.3: XTAL 1 / XTAL 2 Prozessortakt

12.1.3 Taktgenerator

12.1.4 Reset

(*C* und *R* gut gegen Kontakt prellen)

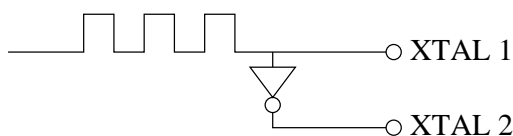


Abbildung 12.4: Taktgenerator

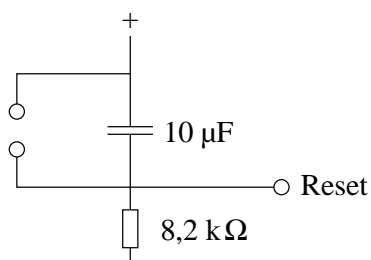
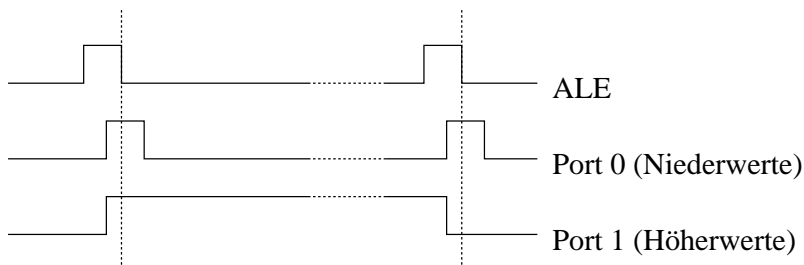


Abbildung 12.5: Gegen Kontaktprellen

Das Resetsignal bewirkt das zurücksetzen des 8051 in einen definierten Startzustand. Dauer (aktiv high) mehr als 24 Oszilator Perioden. Das sind bei 12 MHz 2s. Nach Reset liest der 8051 den ersten Befehl von der Adresse: 0000Hex ALE (address latch enable) Ale ist ein Steuersignal zur Speicherung der niederwertigen 8 Adreßbits bei externem Speicherzugriff. Diese Adreßbits sind mit der fallenden Flanke von ALE am Port 0 verfügbar. Die höherwertigen 8 Bits der Adresse werden am Port 2 auf Dauer ausgegeben.



Um die gesammte 2 Byte Adresse (low und high) für den ersten Zugriff verfügbar zu machen, müssen die niederwertigen Bits in einem Latch zwischengespeichert werden.

LE (Latchenable) mit H - Signal (high signal) werden die Eingangsdaten in das Latch übernommen und mit der fallenden Flanke fixiert.

OE (Output enable) aktive low d.h. die Ausgänge sind immer durchgeschaltet.

P0.0 - P0.7 (Port 0, 8 Bit)

Read Latch (aktiv): Der Inhalt des Latch wird an den internen Bus gelegt.

Read Pin (aktiv): Das Potential des Pin Anschlusses wird an den internen Bus gelegt.

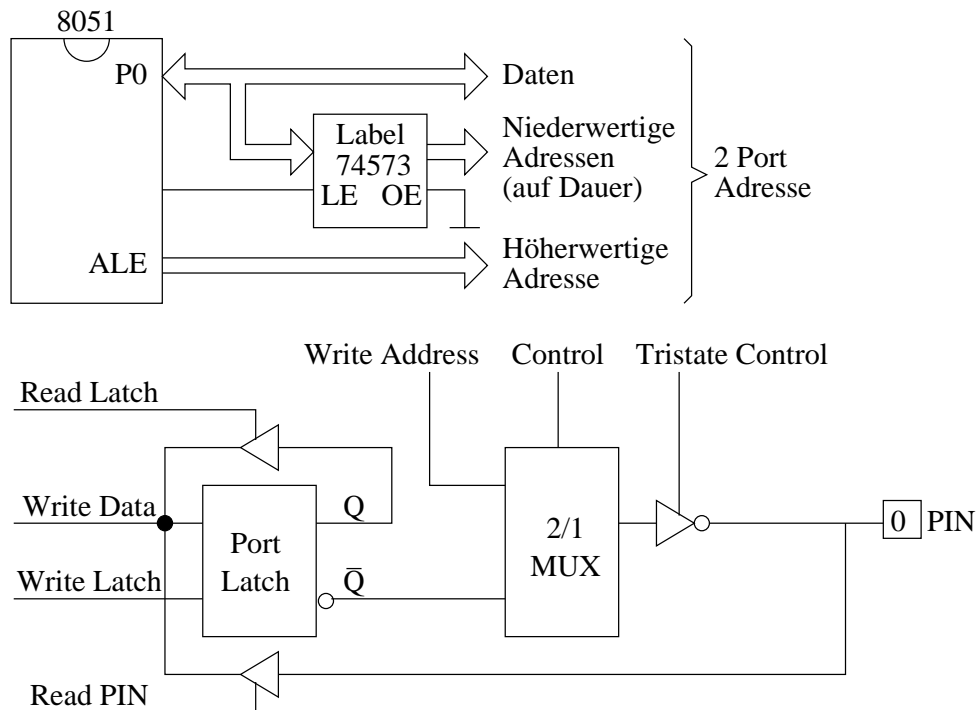


Abbildung 12.6: Bidirektionaler Tristate Port

Write Latch (aktiv): Das Potential des internen Busses wird mit write Latch in das Latch geschrieben über \bar{Q} invertiert und über den Multiplexer an den 3 state Ausgangstreiber gegeben, dort nochmal invertiert gelangt das Signal (positiv) an das Pin.

Write Adress (aktiv): Über den Multiplexer wird das invertierte Adresssignal eingekoppelt und über den Ausgangstreiber invertiert an das Pin gegeben. Write Latch und write Adress wird also im Multiplexbetrieb über den Port 0 ausgegeben.

P 2.0 - P 2.7 (Port 2, 8 Bit)

Bidirektionaler Port mit "pullup" Widerstand

Funktionen wie Port 0 jedoch nur geringere Ausgangsleistung am PIN möglich. Pineingang nur "frei" wenn Ausgang auf high liegt.

P 1.0 - P 1.7 (Port 1, 8 Bit)

Bidirektionaler Port mit "pullup" Widerstand

Read Latch/PIN: wie Port0 und 2.

Write Data: Das Potential des internen Busses wird mit *Write latch* in das latch geschrieben, über Q (\bar{Q}) invertiert und über den Ausgangstreiber abermals invertiert (positiv) an das PIN gegeben.

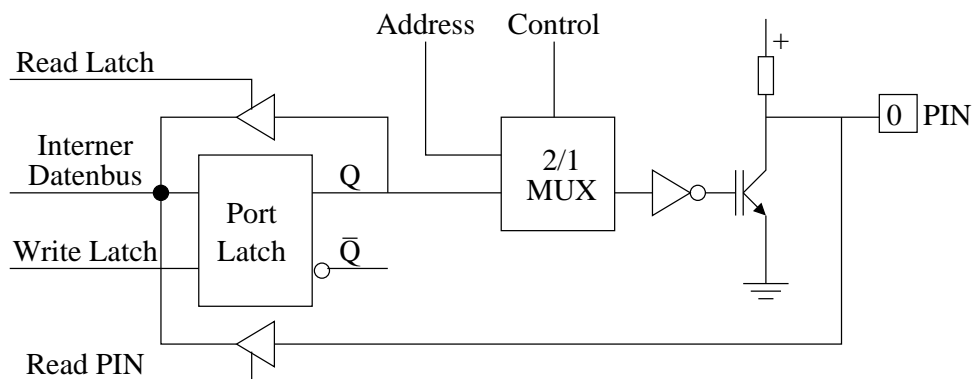


Abbildung 12.7: Schaltung pro Bit

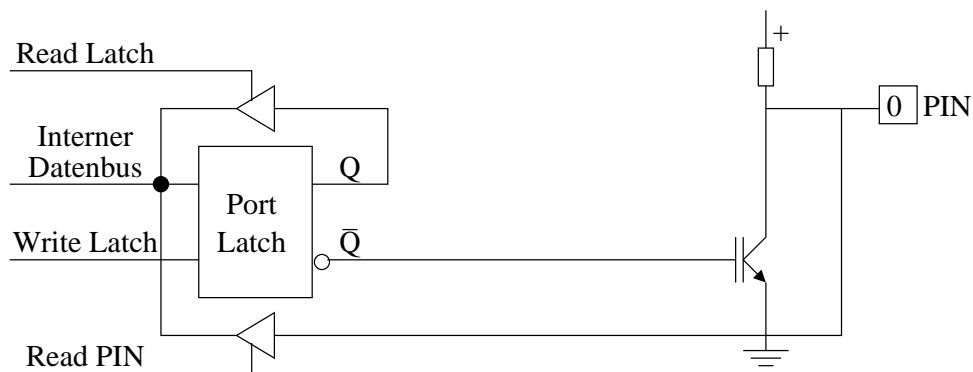


Abbildung 12.8: Schaltung pro Bit

Hinweise: Kein multiplexen mit zweiter Datenquelle.

Für Read Pin ist der Eingang nur frei, wenn der Ausgang auf high liegt (wie Port 2).

P 3.0 - P 3.7 (Port 3, 8 Bit)

Bidirektionaler Port mit "pullup" Widerstand
Read latch und Read Pin wie Port 0, 1 und 2.

Write Data Das Potential des internen Buses wird mit Write latch in das latch geschrieben über das invertierende UND Gatter (alternate output source high) invertiert und über den Ausgangstreiber abermals invertiert (positiv) an das PIN gegeben.

Write alternate output Das latch muss auf $Q = \text{high}$ stehen, die *alternate output source* wird über das invertierende UND Gatter am Ausgangstreiber abermals invertiert (positiv) an das PIN gegeben.

Read alternate in: Falls das PIN auf high liegt kann das *alternate input source* Potential über Read PIN gelesen werden.

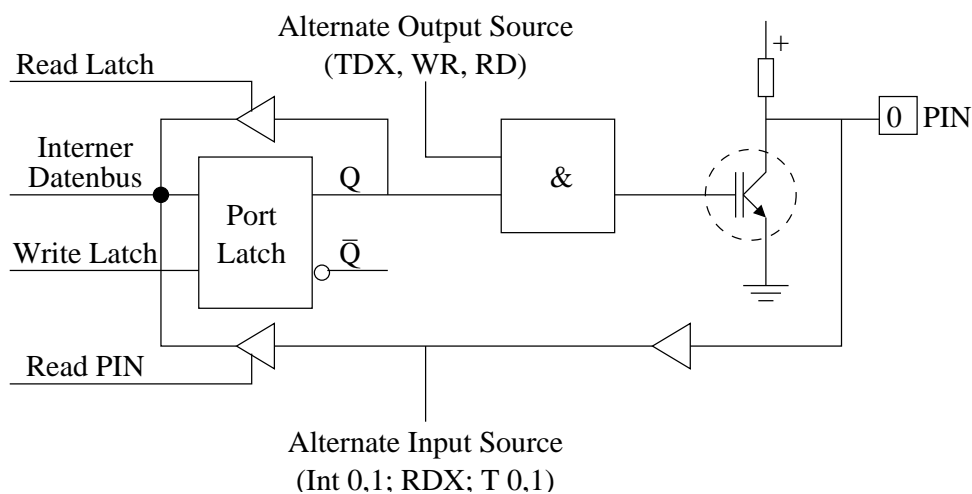


Abbildung 12.9: Schaltung pro Bit

Wann wird das PIN gelesen und wann das latch? Das hängt vom Befehl ab! Wird der Portzustand gelesen und invertiert dann gilt PIN Potential z.B. move, add, sub Befehl. Wird der Portzustand gelesen bearbeitet und zurückgegeben dann gilt das latch Potential. Bsp. logisches OR, UND

Der Wert des PIN wird nur dann richtig gelesen, wenn das zugehörige latch eine logische Eins enthält (frei). Die *alternativen output sourcen* des Port 3 werden direkt vom Prozessor geschrieben. Das zugehörige latch muss auf 1 stehen (frei). Zum Beispiel: TDX, WR, RD

12.1.5 Speicher am 8051

Der 8051 unterscheidet zwischen Programmspeicher welcher nur gelesen werden kann (Zugriffssignal PSEN = Program Storage Enable) und den Datenspeicher welcher sowohl gelesen (Zugriffssignal RD = Read Data) als auch beschrieben werden kann (Zugriffssignal WR = Write).

Dadurch entstehen zwei getrennte Adressräume von je 64 KByte (16 Bit Adressraum) die unabhängig voneinander genutzt werden können.

Durch Kombination der Signale PSEN und RD kann aber auch erreicht werden das Programme und Daten in einem gemeinsamen Speicher, der dann allerdings nur 64 KByte insgesamt gross ist abgelegt werden.

Im Programmspeicherbereich gibt es ausgezeichnete Adressen, die nach RESET bzw. nach Unterbrechungen, interrupts gelesen werden. An diesen Adressen müssen Befehle stehen welche auf die Ereignisse (INT, RESET) entsprechend reagieren. Für den RESET an der Adresse 0000 stehen 3 Bytes zur Verfügung, damit ein Sprungbefehl in die Startroutine realisiert werden. Für die Interrupts stehen jeweils 8 Bytes zur Verfügung in denen ebenfalls ein Sprung in eine Bedienroutine untergebracht werden kann oder die Routine steht in den 8 Bytes.

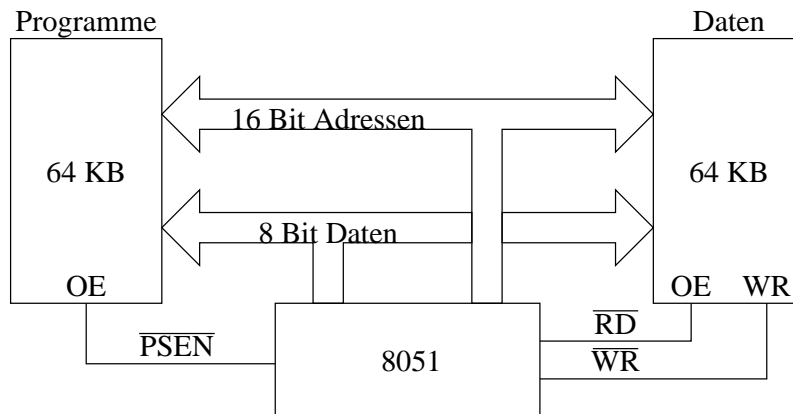


Abbildung 12.10: Trennung von Programm- und Datenspeicher (Harward)

12.2 Interner Programmspeicher

Abhängig vom Prozessortyp gibt es einen Internen Programmspeicher, welcher über EA=1 aktiviert werden kann.

Neben dem externen, nicht zum 8051 gehörenden Speicher für Programme und daten besitzt der 8051 aber auch einen 256/...Bte großen internen RAM Speicher für Daten, welcher über Befehle angesprochen werden kann. Dieser Speicher enthält die Register und Spezialregister sowie die Port-Adressen (memory mapped) sowie auch einen Bit-Adressierbaren und einen Byte-Adressierbaren Bereich zur beliebigen Verwendung. Die unteren 128 Byte des internen Datenspeichers können direkt und indirekt adressiert werden. Die oberen 128/.. Byte nur indirekt mit Hilfe der Register R0 und R1.

12.2.1 Die Bereiche des Internen RAM's

Byte 0 - Byte 1F (32 Byte)

Diese Bytes sind als 4 jeweils 8 Register umfassende Registerbänke organisiert. Die Register jeder Bank heißen R0, R1, ... R7. Welche Registerbank ausgewählt ist hängt von Bit 3 und Bit 4 des PSW (progamm status word) ab. Das PSW ist auf die Adresse B0 des internen RAM's abgebildet (s.Blatt 18.10.01)

Byte 20 - Byte 2F (16 Byte)

Dieses ist ein Bit Adressierbarer Bereich auf dem mit speziellen Befehlen zugegriffen werdn kann. Das niederwertigste Bit des Bereichs hat die Adresse 00 Hex, das höchstwertigste Bit des Bereichs hat die Adresse 7F Hex (128 Bit)

Byte 30 - Byte 7F (80 Byte)

Dieses ist ein Byte Adressierbarer Bereich auf den mit dem allgemeinen Befehlssatz zugegriffen werden kann. Er ist direkt und indirekt adressierbar. Das niederwertigste Byte hat die Byteadresse 30 Hex und das höchstwertigste die Byteadresse 7F hex

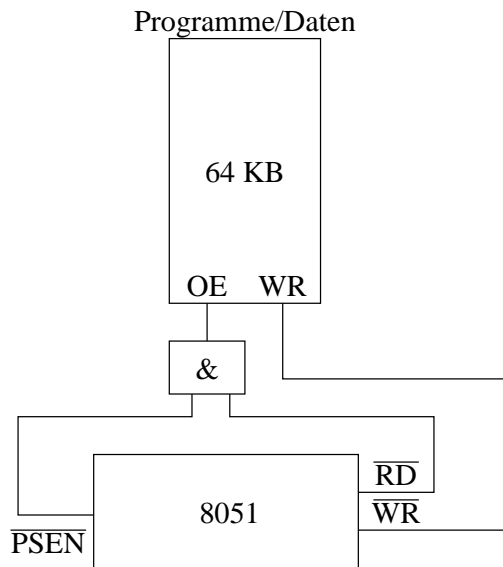


Abbildung 12.11: Programme und Daten

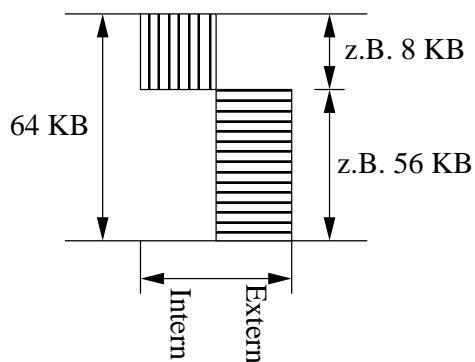


Abbildung 12.12: Großer Typ 8 KB

Byte 80: Port 0

Dieses ist die Adresse auf die die 8 Bit des Port 0 abgebildet werden. (memory mapped)

Byte 81: SP (stack pointer)

Enthält den Stackpointer. Dieser verweist auf eine Adresse im Byte-Adressierbaren Bereich (30 - 7F hex)

Byte 82/83: DP (data pointer)

Diese enthalten eine 16 Bit große externe Speicheradresse (DP = Datapointer, 82: DPL = DPlow, 83 DPH = DPhigh)

Byte 87: PCON (power control register)

SMOD	-	-	-	0	0	PD	IDL
------	---	---	---	---	---	----	-----

SMOD=1: Verdoppelung der Band Rate am Timer=1

PD/IDL: Bewirkt ein Power down/IDL Status (sleep) des Prozessors Das Ziel sind Stromsparmodi

Byte 88: TCON (Timer controll)

ermöglicht die Steuerung und Abfrage der TIMER. Das TIMERCONTROLL Register ist Bit adressierbar

TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0
-----	-----	-----	-----	-----	-----	-----	-----

TF1 = 1: TIMER 1 ist abgelaufen. Dieses Bit wird vom 8051 gesetzt und wird mit der Bedienung des Interrupts rückgesetzt.

TR1 = 1: TIMER 1 aktiviert

TR1 = 0: TIMER 1 inaktiv

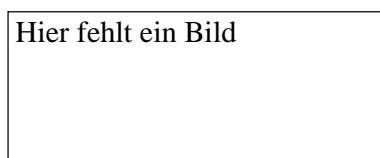
TF0 und TR0: analog nur für TIMER 0

IE1: wird vom 8051 gesetzt wenn der externe Interrupt 1 erkannt wird

IT1 = 1: Flankentriggerung

IT1 = 0: Leveltriggerung

IE0 und IT0: analog nur für Interrupt 0

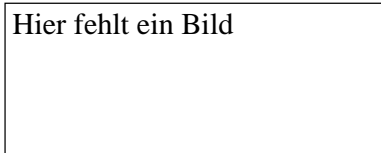
Byte 89

Timer oder Counter Betrieb:

$C/\bar{T} = 0$ Timer Btrieb THx und TLx T1/12 Oszilatorfrequenz. Die Register THx und TLx werden mit jedem Maschienenzyklus (Befehl) erhöht mit einem 1/12 Oszilatorfrequenz.

$C/\bar{T} = 1$ Counter Betrib THx und TLx (Pfeil hoch, 1 0 Übergang) Tx. Die Register THx und Tlx werden mit jedem 1 0 Übergang am Eingang tx erhöht.

Gate Wenn TRx gesetzt ist (Timer/Counter läuft) werden die Register Thx und Tlx erhöht, wenn Gate = 0. Ist Gate = 1 werden sie nur erhöht, wenn das zugehörige int x Pin 1 ist.



M1/M0	Definieren die Zählweise	des Timers/Counters	
0 0	(Mode 0) TH1 8 Bit	TLx 5 Bit -i	13 Bit Zähler
0 1	(Mode 1) TH1 8 Bit	Tlx 8 Bit -i	16 Bit Zähler
1 0	(Mode 2) Auto reload	TLx 8 Bit Zähler	Unterbrechungssequenz
		THx Ladewert nach Ablauf	

Byte 8A

TL0 (Low Byte Register für Timer /Counter 0)

Byte 8B

TL1 (Low Byte Register für Timer/ Counter 1)

Byte 8C

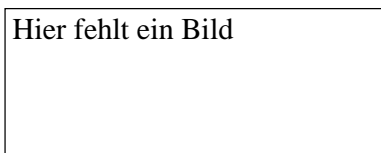
TH0 (high Byte Register für Timer/Counter 0)

Byte 8D

TH1 (High Byte REgister für Timer/Counter 1)

Byte 98: (SCON)

Steuerregister für die serielle Schnittstelle



SM0/SM1: definieren die Betriebsart

Die variable Baudrate kann mit Timer 1 im *autoreload mode* erzeugt werden, zum Beispiel der Prozessortakt von 11,059 MHz.

SM2 = 0 (constant)

REN = 1 Datenempfang über RDx möglich

0 0	shift Register mit fixer baudrate ($f_{osz}/12$) (RDx Pin, TDx Pin)
0 1	8 Bit UART mit variabler Baudrate aus Timer 1
1 0	9 Bit UART mit fixer Baudrate ($f_{oaz}/32$)
1 1	9 Bit UART mit variabler Baudrate aus Timer 1

TIMER 1					
Baudrate	SMODE	$C/\overline{T} = 0$	Mode	Reload Value	
19,2 KB	1	0	1 0	FD	
9,6 KB	0	0	1 0	FD	→ 3
4,8 KB	0	0	1 0	FA	→ 6

REN = 0 Datenempfang über RDx ausgeschaltet

TB8 9. Bit im Betriebsmodus 3 zur Sendung

RB8 9. Bit im Betriebsmodus 3 bei Empfang

T1 Transmit Interrupt Flag wird vom Prozessor nach Aussendung von 8 oder 9 Bits gesetzt

R1 Recieve Interrupt Flag wird vom Prozessor nach Empfang von 8 oder 9 Bits gesetzt

Ablauf eines Sendevorgangs

- Vorbereitung des Clocktacktes über Timer 1
 - TR1 = 1 Timer 1 aktivieren (TCON)
 - GATE = 0, $C/\overline{T} = 0$, Mode = 1 0 (TMOD)
 - TH1 = TD(z.B.9,6 KB)
- Vorbereitung der seriellen Schnittstelle
 - MOD = 0 1 (8 Bit UART variable Baudrate Timer 1)(SCON)
 - SM2 = 0, REN = 0, TB8 = 0, RB8 = 0 (SCON)
 - ES = 1 enable seriel interrupt (wird später erklärt)
- Aussendung eines Bytes
 - Byte - i SBUF (Byte 99) - i Bits gehen über TDx raus

Warteschleife im Programm, warten auf Ende interrupts der seriellen Byteausgabe Interrupt - i Start am 0023 - i neues Byte nach SBUF (und zurück zu Warteschleife)

Anhang A

Übungsaufgaben

Fragen

1. Durch welche digitalen Grundschaltungen werden die logischen Bauteile und/oder Inverter realisiert? Beschreiben Sie die Wirkungsweise der Schaltungen!
2. Welche internen Informationsspeicher (Chips) finden in Digitalrechner Verwendung?
3. Beschreiben Sie das Bus-Prinzip und die Funktion der verschiedenen Leitungsgruppen eines Buses!
4. Zeichnen Sie eine Schaltung, bei der ein Speicherbaustein über einen Bus mit einem Latch verbunden ist! Durch welche Folge von Signalkombinationen am Bus kann der Inhalt des Latches in die Speicherzelle 0 transportiert werden und dann der Inhalt der Speicherzelle 3 in das Latch?
5. Zeichnen Sie das Blockschaltbild einer CPU und beschreiben Sie die Aufgaben der einzelnen Funktionseinheiten!
6. In welche Klassen werden Maschinenbefehle eingeteilt und welche Befehle sind in den einzelnen Klassen?
7. Was bewirkt eine Programmunterbrechung und welche Aufgaben hat die zugehörige Service-Routine?
8. Wie ist der Stack-Speicher organisiert und wozu dient er?
9. Was sind die Unterschiede zwischen Real-Mode und Protected-Mode bei der Adressierung des Arbeitsspeichers?
10. Zeichnen Sie den Übergangsggraph des MESI Protokolls und beschreiben Sie die Zustände!
11. Erklären Sie Multitasking und Multiprocessing!
12. Wie funktionieren Befehlspipelines? Welcher Zusammenhang ist zwischen Kollisionswahrscheinlichkeit und Wirksamkeit zusätzlicher Pipes?

13. Wie funktioniert die Sprungvorhersage?
14. Wann ist ein Busarbiter n ötig und welche Aufgaben hat er?
15. Wie arbeitet ein gemultiplexer Bus?
16. Wie werden Ein/Ausgabesysteme angeschlossen und welche Betriebsweisen sind möglich?
17. Wie funktioniert die parallele/serielle Datenkonversion?
18. Wie funktioniert die digitale/analoge Datenkonversion?
19. Arbeitsweise und Bildaufbau beim VGA-Bildschirm?
20. Arbeitsweise und Bildaufbau beim LCD-Bildschirm?
21. Wie funktionieren das Bitmap und das Vektorverfahren beim Druckbildaufbau?
22. Wie funktionieren Matrix- und Laserdrucker?
23. Wie funktionieren CSMA und Token Ring bei lokalen Netzen?
24. Entwickeln sie eine Schaltung, welche in einer Mikroprozessorkonfiguration (SA) das Schreibsignal \overline{WR} für den externen RAM-Speicher immer dann unterdrückt, wenn Adressen $\geq 8000H$ und $< A000H$ angesprochen werden.
25. Entwickeln sie eine Schaltung, welche in einer Mikroprozessorkonfiguration (SA) deren verfügbarer externer RAM-Speicher von 64K auf 128K erweitert wird. Als zusätzliches Adressbit soll Bit 0 des Port 3 benutzt werden (P3.0).
26. Der Bus einer Mikroprozessorkonfiguration (SA) enthält 16 Adresspins, 8 Datenpins und die Signale \overline{WR} , \overline{RD} und \overline{PSEN} . Entwickeln sie die Schaltung, einer Erweiterungsplatine zu je 8 Bits Latch enthält und die memory-mapped über die Adressen $0x0001$, $0x0002$ und $0x0003$ angesprochen werden.

Antworten

Und hier die ersten Antworten, die per Email bei mir eingetrudelt sind:

1. FIXME1
2. In Rechnern finden sich sowohl flüchtige als auch nicht flüchtige Speicher. Dies wären einerseits die dynamischen und statischen RAM-Bausteine und auf der anderen Seite ROM, PROM, EPROM, EEPROM und Flash-Bausteine.
3. FIXME3
4. FIXME4

5. FIXME5
6. Arithmetische Befehle: ADD, SUB, INC, CMP
Bitoperatorbefehle: XOR, AND, NOT
Lade/Speicherbef. MOV, PUSH, POP
Programmflußbefehle: JMP, CALL, HLT
Sonstige Befehle: NOT,...
7. FIXME7
8. FIXME8
9. FIXME9
10. FIXME10
11. FIXME11
12. Die Verarbeitung von Befehlen benötigt in aller Regel einige Taktzyklen, wobei durch diesen Befehl mehrere voneinander abhängige Operationen ausgelöst werden. Dabei ruhen $n - 1$ Funktionseinheiten, während 1 Einheit arbeitet. Da ist es naheliegend, die durchlaufenen Einheiten gleich mit neuen Befehlen zu füllen, so daß pro Takt ein Befehl ausgeführt wird. Problematisch sind dabei bedingte Sprünge, da sie ganze Teile der Pipeline ungültig machen können, wenn sie doch anders Verzweigen als vorhergesagt. Dies ist auch gleich der Knackpunkt, warum längere Pipelines eher schlechter sind.
13. Sprungvorhersage funktioniert im Einfachsten Fall so, daß bei den klassischen Schleifenbefehlen (JNZ...) proforma so getan wird als würde gesprungen. In komplizierteren Fällen wird eine Statistik für die Sprünge geführt, und diese berücksichtigt. Gute Sprungvorhersage ist deshalb so wichtig, weil damit leertäufe einer u.U 20-stufigen Pipeline vermieden werden können.
14. FIXME14
15. FIXME15
16. Schreibfehler! E/A.Bausteine können am Daten/Adressbus als memory-maped E/A oder über dedizierte Ports / E/A-Busse angesprochen werden. Dabei können sie in der Betriebsart Interrupt oder Polling arbeiten.
17. Gatter
18. Widerstandsnetzwerk
19. FIXME19
20. FIXME20
21. FIXME21
22. FIXME22
23. FIXME23

24. FIXME24

25. FIXME25

26. FIXME26