

Externe Erweiterung

Weshalb extern erweitern? (Elektrische Anpassung / funktionelle Erweiterung)

Elektrische Probleme an den Schnittstellen zur Außenwelt

Pegel / Signalhub

Flanken / Anstiegszeiten

Zeitlicher Verlauf / Impulsbreiten

Ströme / Treibvermögen

offene Eingänge (z. B. steckbare Funktionseinheiten nicht angeschlossen, Drähte nicht angeklemmt)

Kurzschlüsse

ESD

EMV

Betriebsfall eingeschaltet / ausgeschaltet (Partial Power Down)

Metastabilität

Isolation (galvanische Trennung)

Einzweckanschlüsse

Bussysteme und Interfaces

Grundlagen

Elektrische Auslegungen

Auslegung paralleler Bussysteme

Ringstrukturen

Schieberegisterstrukturen

E-A-Erweiterung

Speichererweiterung

Welche Bauelemente einsetzen?

Wir reden grundsätzlich über vergleichsweise kleine Projekte – alles, was irgendwie kompliziert oder aufwendig aussieht, wollen wir schließlich der Rechenmaschine (vom Mikrocontroller bis zum Industrie-PC) überlassen und dort mit Software erledigen. Bei der Bauelementeauswahl sind heutzutage nicht nur die Kosten von Bedeutung, sondern auch Fragen der praktischen Handhabbarkeit:

- wie aufwendig ist ggf. der rechnergestützte Schaltungsentwurf (Software, Computer, Programmiergeräte usw.)?
- kann man mit einfachen Mitteln einen Laboraufbau zustande bringen?
- kann man die Leiterplatte mit gleichsam hausbackenen Mitteln entwerfen?*)
- läßt sich die Leiterplatte bei jedem x-beliebigen Dienstleister preisgünstig fertigen?
- passen Speisespannung und Signalpegel zum Rest des Systems oder sind in dieser Hinsicht Sonderaufwendungen erforderlich (z. B. zur Pegelwandlung)?

Es geht also nicht nur um die eigentliche Funktion, sondern auch um Gehäusebauformen, um Anschlußabstände und Lötverfahren, um die Ebenenzahl der Leiterplatte, um Speisespannung und Signalpegel (Stichworte: 5-V-Toleranz, 3,3-V-Toleranz) sowie um die anzuwendenden Entwurfs- und Programmierverfahren. Unter diesen Gesichtspunkten wird es sich ab und zu herausstellen, daß mancher – an sich sehr attraktive – Schaltkreis für unsere Zwecke nicht in Frage kommt (das betrifft vor allem ganz moderne Typen, die vor allem zum Einsatz in Mobiltelefonen u. dergl. entwickelt wurden).

*) : soll heißen: Placieren und Entflechten nach gängigen Faustregeln und gesundem Menschenverstand.

Herkömmliche Schaltkreise (Off the Shelf)

Naheliegender, wenn es um kleinere Vorhaben geht. Sind aber 20 Stück^{*)} und mehr erforderlich, sollte man sich nach Alternativen umsehen.

Programmierbare Logikschaltkreise

Das Mittel der Wahl für alle auch nur halbwegs komplizierten und/oder umfangreichen Digitalerschaltungen (Richtwert: zu erwägen, wenn der Entwurf ansonsten mehr als 5 DIL-Gehäuse (TTL/CMOS) erfordern würde). Der Einstieg erfordert aber einigen Aufwand, nämlich eine passende Entwicklungssoftware, einen hinreichend leistungsfähigen PC und ggf. ein Programmiergerät.

Praxistip:

Eine CPLD- oder FPGA-Familie mit Flash-Programmierung, die in der Anwendungsschaltung programmiert werden kann (In-System Programming (ISP)). Die Programmiervorkehrungen beschränken sich darauf, die zum Programmieren erforderlichen Signale auf geeignete Anschlüsse zu führen (Abb. 1.1). Die Preise der Entwicklungsumgebungen bzw. Starterkits liegen in erschwinglicher Größenordnung (Mindestausstattung: Entwurfssoftware + Download-Kabel). Einige Hersteller bieten kostenlose Entwurfssoftware an. Die einfachste Form der Entwurfseingabe: über Schaltplan. Typische Vorteile:

- viele Projekte kommen mit einem einzigen Schaltkreis aus (vgl. Abb. 1.1),
- man kann immer wieder ändern (= löschen und neu programmieren),
- man kann oft gleichsam naiv entwerfen, ohne sich um die Schaltungsoptimierung, um das Heraussuchen passender 74er oder 4000er Schaltkreise, um die Aufteilung der Funktionen auf die Schaltkreise usw. kümmern zu müssen (mit den DeMorganschen Regeln, mit Karnaugh-Plänen usw. müssen wir uns nicht mehr abgeben -- das erledigt die Entwurfssoftware).

*): Richtwert. Entspricht typischerweise einer mit DIL-Gehäusen voll bestückten Europakarte 100 @160 mm.

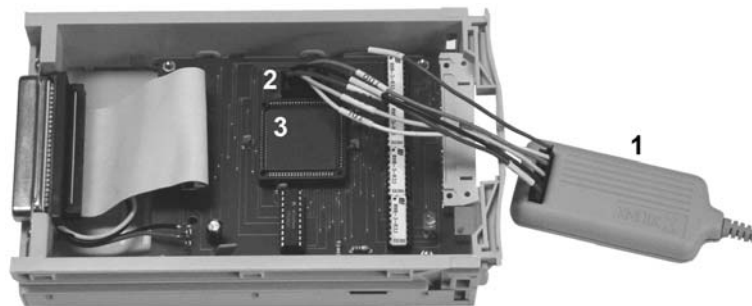


Abb. 1.1 Ein CPLD-Schaltkreis wird programmiert. 1 - Download-Kabel (an die Parallelschnittstelle des Entwicklungs-PCs angeschlossen). 2 - Programmier-Steckverbinder auf der Leiterplatte; 3 - der zu programmierende Schaltkreis

IP: fertige Entwürfe nutzen

IP = Intellectual Property. Es werden Schaltungsentwürfe angeboten, die man in eigene CPLDs und FPGA einbringen kann. Das ist zum einen eine kommerzielle Angelegenheit (kostet richtig Geld). Zum anderen gibt es eine „freie“ bzw. „offene“ Szene. Von derartigen Schaltungen kann man aber keineswegs erwarten, daß sie unter allen Umständen ohne eigenes Zutun auf Anhieb funktionieren – es kommt u. a. darauf an, für welche Schaltkreistypen der Entwurf ursprünglich vorgesehen war, in welcher Form (VHDL, Verilog, Netzliste, Programmierdaten usw.) er übergeben wird und welche weitere Unterstützung (Dokumentation, Testdaten usw.) es gibt.

Hochintegrierte Schaltkreise der Massenfertigung

Ehe man damit beginnt, eine komplizierte Schaltung selbst zu entwerfen, liegt es nahe, sich erst einmal danach umzusehen, was es so gibt. Für unsere Zwecke kommen vor allem E-A-Schaltkreise in Betracht:

- universelle Typen, z. B. der alte, aber immer noch gern verwendete 8255,
- die Super-IO-Schaltkreise der PCs,
- USB-Schnittstellenschaltkreise,
- PCI-Schnittstellenschaltkreise,
- Schaltkreise mit Einfachinterfaces (I²C, SPI o. dergl.),
- Leistungselektronikschaltkreise mit digitaler Schnittstelle (Logikteil + Leistungsteil), z. B. als Relaisreiber, zum Ansteuern von Schrittmotoren usw.

Mikrocontroller

Kleine Mikrocontroller sind gelegentlich eine Alternative zum programmierbaren Logikschaltkreis oder zum fertig bezogenen E-A-Schaltkreis. Der Mikrocontroller ist zu programmieren. Das kann ein Vorteil sein (mehr Freiheit, eigene Vorstellungen zu verwirklichen, Entlastung des PCs von bestimmten Funktionen), aber auch ein Nachteil (Arbeitszeit, Kosten).

Mikrocontroller oder CPLD?

Ein kleiner CPLD-Schaltkreis mit typischerweise 32...36 Makrozellen (= Flipflops) kostet ungefähr dasselbe wie ein PIC oder AVR oder 8051 usw. mit vergleichbarer Anzahl an Signalanschlüssen. Man hat also gelegentlich die Qual der Wahl. Ein naheliegendes Entscheidungskriterium ist die Reaktionszeit an den zu steuernden Schnittstellen:

- wenn es auf Mikrosekunden oder gar Nanosekunden ankommt: CPLD,
- wenn die Millisekunde kaum eine Rolle spielt: Mikrocontroller.

Hinweise:

1. Der Mikrocontroller bietet mehr fürs Geld, nämlich nahezu unbeschränkte Funktionsvielfalt (freie Programmierbarkeit), die CPLD hingegen nur das, was sich mit den wenigen Flipflops und einigen hundert Gattern realisieren läßt. Zudem haben viele Mikrocontroller eingebaute Schnittstellen (die man andernfalls selbst entwerfen müßte).
2. Die CPLD ist viel schneller. 100 MHz Takt heißt 10 ns Reaktionszeit vom Eingang zum Ausgang. Ein 100-MHz-Mikrocontroller hätte da bestenfalls einen einzigen Befehl ausgeführt -- und der leistet nicht eben viel...
3. Extrem schnelle Mikrocontroller (von 25 MHz an aufwärts) werden angeboten. Sie sind deutlich teurer als die Massenfabrikate (mit typischerweise 5...20 MHz). Trotz der hohen Taktfrequenz sind die Reaktionszeiten in nichttrivialen Anwendungen enttäuschend, ja manchmal geradezu lausig.

Elementare E-A-Schnittstellen

Eine hinreichende Anzahl an Ein- und Ausgabeports bildet die Grundlage der anwendungsseitigen Schnittstellenentwicklung. Das gilt auch für das Anschließen von Leitungstreibern, Analog-Digital-Wandlern, Zeitstufen usw. Steht das Anwendungsvorhaben fest, so ist klar, wieviele Ein- und Ausgänge zu unterstützen sind. Es liegt nahe, Mikrocontroller einzusetzen, die eine ausreichende Anzahl an Ein- und Ausgängen haben. Dem stehen aber oftmals hohe Kosten entgegen, und für manche Anforderungen gibt es gar nichts Passendes. Viele Entwicklungsaufgaben sind deshalb mit vergleichsweise wenigen Interfacesignalen zu lösen (z. B. mit kostengünstigen Mikrocontrollern, die typischerweise 4...32 E-A-Anschlüsse haben).

Ausgänge

Ausgänge sind typischerweise über Register bereitzustellen. Brauchen wir mehr Ausgänge, als das Interface Datensignale hat, bieten sich folgende Lösungen an:

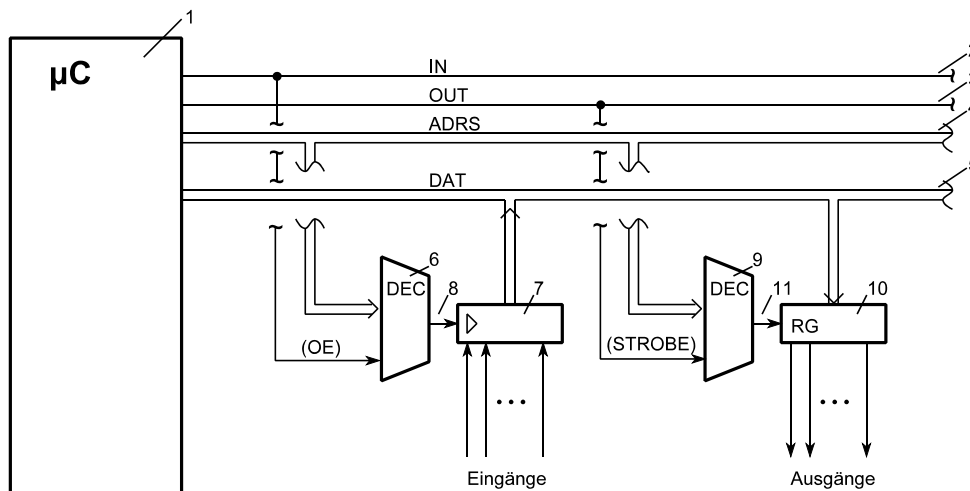
- paralleler Bus, an den mehrere einzeln ladbare Register angeschlossen sind (Abb. 1.2),
- Schieberegister.

Eingänge

Eingangsbelegungen sind auf die Datenleitungen des Interfaces aufzuschalten. Brauchen wir mehr Eingänge, als das Interface Datensignale hat, bieten sich folgende Lösungen an:

- paralleler Bus. Aufschaltung nach dem Tri-State-Prinzip (Abb. 1.2).
- Abfrage über Schieberegister,
- Abfrage über Multiplexer (Abb. 1.3).

Erfordert das Interface eine synchrone Signalaufschaltung, so sind die Eingänge zunächst einzutaktieren (Synchronisation). Vor der eigentlichen Aufschaltung ist also ggf. ein taktflankengesteuertes Register anzuordnen.



1 - Mikrocontroller; 2 - Eingabesteuersignal (Strobe, Takt); 3 - Ausgabesteuersignal (Output Enable); 4 - Adreßbus; 5 - Datenbus; 6 - Adreßdecoder; 7 - Koppelstufe für Eingänge; 8 - Aufschaltsignal; 9 - Adreßdecoder; 10 - Ausgaberegister; 11 - Übernahme-signal.

Abb.1.2 Anschluß mehrerer Einrichtungen an einen Port (Busprinzip)

Zunächst wird die jeweilige Adresse auf den Adreßbus 3 gelegt. Der weitere Ablauf hängt von der Art des Zugriffs ab:

Eingabe:

Der Mikrocontroller belegt den Datenbus und gibt einen IN-Impuls ab. Erkennt der Adreßdecoder 6 die betreffende Adresse, so werden über die Koppelstufe 7 die betreffenden Eingänge auf den Datenbus 5 geschaltet.

Ausgabe:

Der Mikrocontroller schalten den Datenbus auf Eingabe um (Bus wird hochohmig) und gibt einen OUT-Impuls ab. Erkennt der Adreßdecoder 9 die betreffende Adresse, so wird die Belegung des Datenbus 5 in das Ausgaberegister 11 übernommen. Die Belegung der angeschlossenen Ausgangsleitungen entspricht dann der besagten Datenbusbelegung.

Zu Position 10: Register oder Koppelstufe?

Das hängt davon ab, ob der Datenport des Mikrocontrollers metastabile Zustände verträgt oder nicht. Wenn nein, dann ein Register. Wenn ja, kann ggf. auch eine Koppelstufe gewählt werden.

Achtung: Beim Aufschalten von Bustreibern Buskonflikte vermeiden (Richtungssteuerung).

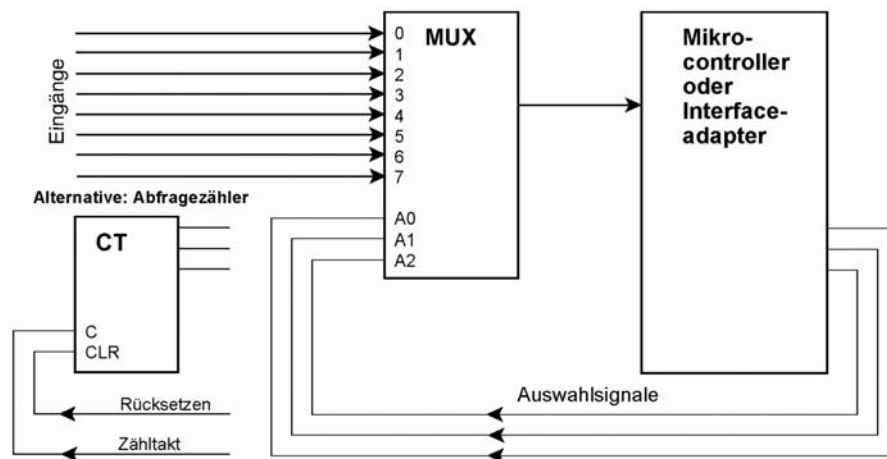


Abb. 1.3 Informationseingabe über Multiplexer

Eine Multiplexeranordnung zum Abfragen von n Bits braucht $\log_2 n$ Adreßeingänge, die von einem Ausgabeport zu treiben sind (Auswahlsignale). Alternative: ein externer Adreßzähler (Abfragezähler). Zu dessen Ansteuerung benötigen wir lediglich ein Takt- und ein Rücksetzsignal.

Wie aufwendig ist eine bestimmte Art der Schnittstellenerweiterung? – Wichtige Auswahlkriterien:

- die Anzahl der zur Ansteuerung erforderlichen Anschlüsse (Pin Count),
- die Zugriffszeiten,
- Anzahl und Kosten der erforderlichen Schaltkreise.

Die Wichtigkeit entspricht typischerweise der Reihenfolge unserer Aufzählung, denn meist denkt man über solche Erweiterungsmaßnahmen deshalb nach, um mit einem kleinen Mikrocontroller auszukommen. Die Anschlußzahl hat also oft Vorrang.

Tabelle 1.1 enthält eine Gegenüberstellung typischer Erweiterungsprinzipien. Dabei beziehen wir uns auf

naheliegende, einfache Auslegungen.

paralleler Bus (Ein- und Ausgabe)	Schieberegister (Ein- und Ausgabe)	Multiplexer (nur Eingabe)
Datenleitungen gemäß Zugriffsbreite, Adreßleitungen gemäß \lg Ports ¹⁾ , Lesesteuerleitung (RD), Schreibsteuerleitung (WR)	4 (Dateneingang, Datenausgang, Takt, Übernahme)	1 Dateneingang, Adreßleitungen gemäß \lg Bits ²⁾ Mit externem Abfragezähler: 3 (Dateneingang, Takt Rücksetzen)
Beispiel: der Zugriff auf 32 Bitpositionen erfordert...		
$8 \cdot \text{Daten} + 2 \cdot \text{Adresse} + 2 \text{ Steuerung} = 12$ Leitungen	4 Leitungen	$1 \cdot \text{Daten} + 5 \cdot \text{Adresse} = 6$ Leitungen. Mit externem Abfragezähler 3 Leitungen
Zeitbedarf (auf das Beispiel bezogen)		
1 Byte je Zeitintervall ³⁾ (4 Zeitintervalle)	1 Bit je Zeitintervall ³⁾ (32 Zeitintervalle)	1 Bit je Zeitintervall ³⁾ (32 Zeitintervalle)

1): Zweierlogarithmus der Portanzahl; 2): Zweierlogarithmus der Bitanzahl; 3): gemeint ist die Zeit, die ein einzelner programmseitiger Zugriff erfordert (das Ausgeben eines Bytes, das Schieben eines Bits usw.)

Tabelle 1.1 E-A-Schnittstellenerweiterungen im Vergleich

Sehr breite Anwendungsschnittstellen (Assembly/Disassembly)

Gelegentlich sind viele Signalleitungen anzusteuern oder abzufragen. Eine Grundfrage: ist es zulässig, das gleichsam stückweise zu erledigen (z. B. Byte für Byte) oder müssen alle Signalbelegungen auf einmal gestellt oder abgeholt werden? – Ist die stückweise Erledigung zulässig, genügen die bisher angesprochenen Erweiterungslösungen. Ansonsten müssen wir uns etwas einfallen lassen (Abb. 1.4 und 1.5).

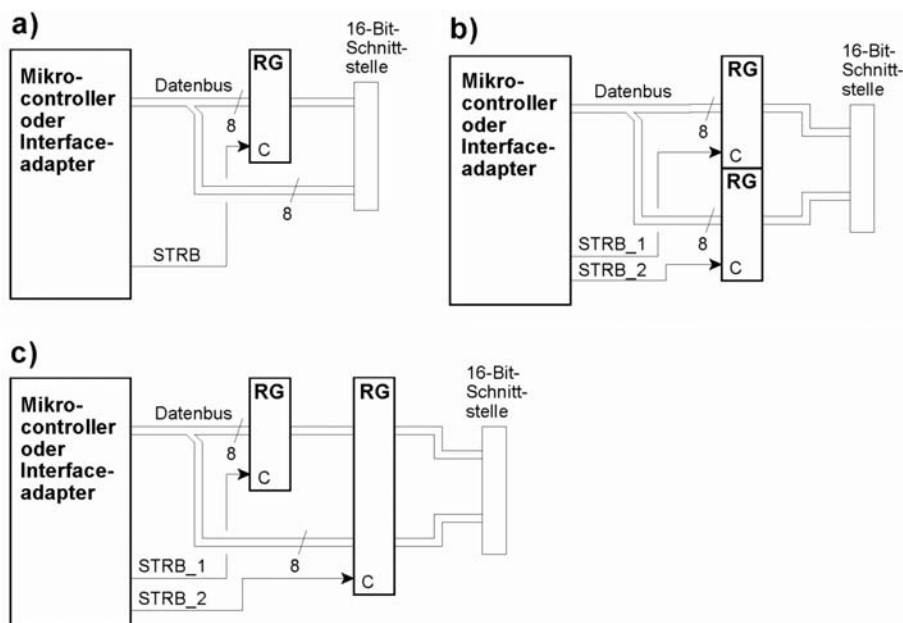


Abb.1.4 Porterweiterungen (1). Ausgabe (Beispiele)

- 16-Bit-Schnittstelle mit 8-Bit-Port. 8 Bits werden aus einem Pufferregister geliefert, 8 weitere Bits direkt vom Port. Laden des Registers über programmierbares Strobe-Signal. Problem: um das Register zu laden, muß der Port mit dem Registerinhalt belegt werden. Somit ändert sich die Belegung der verbleibenden 8 Bits.
- 16-Bit-Schnittstelle mit 8-Bit-Port. Zwei Register. Belegung bleibt außen stabil, erscheint aber in 8-Bit-Abschnitten mit zeitlichem Versatz.
- 16-Bit-Schnittstelle mit 8-Bit-Port und Assembly-Funktion. Außen erscheinen die 16-Bit-Belegungen stets auf einen Schlag (kein Zeitversatz). Nutzung: (1) das 8-Bit-Register laden, (2) die verbleibenden 8 Bits auf den Port geben und das 16-Bit-Register laden.

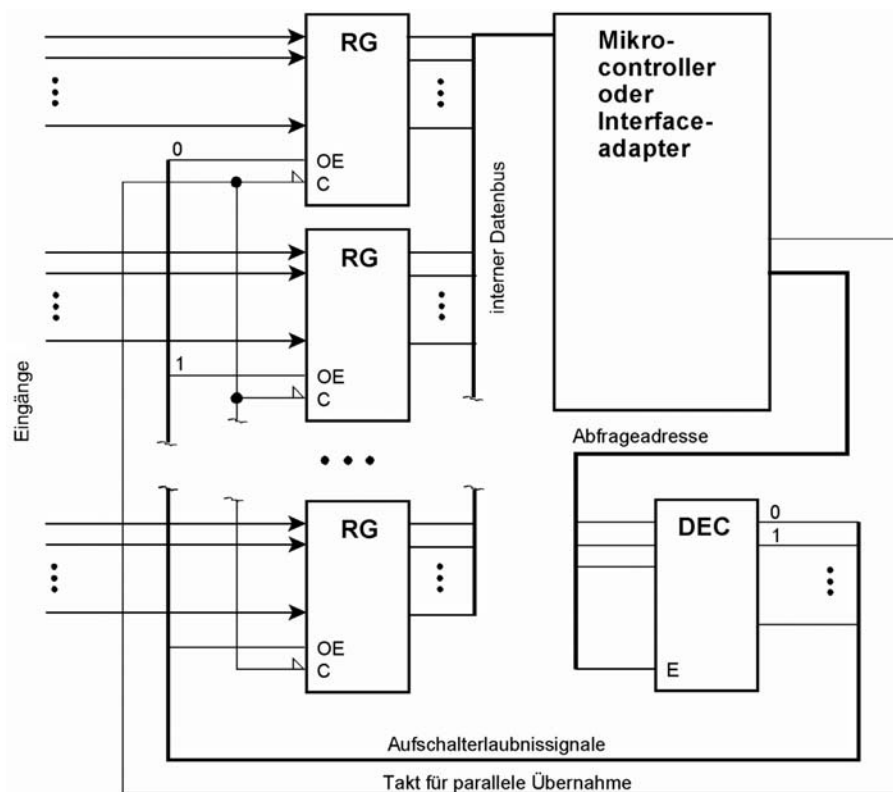
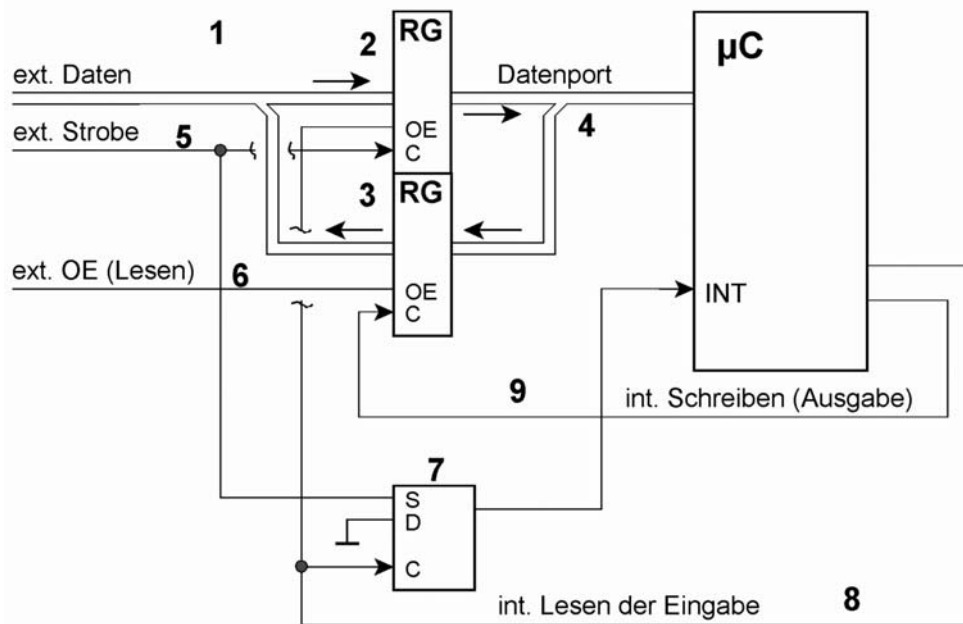


Abb.1.5 Porterweiterungen (2). Eingabe (Beispiel)

Die eingangsseitige Datenbelegung wird in alle Register parallel übernommen. Anschließend können die Registerinhalte einzeln über den internen Datenbus gelesen werden (Disassembly-Funktion).

Zugriffe von außen steuern (Slave-Ports)

Manchmal werden von anderen Einrichtungen Signalbelegungen angefordert oder geliefert, und zwar in einem Zeitraster, dem keine Software gewachsen ist (die entsprechenden Erlaubnis- oder Strobe-Signale sind vielleicht 20...200 ns lang aktiv). Beispiele: Anschluß an den ISA-Bus, an die Parallelschnittstelle oder an einen Prozessorbus. Eine entsprechende E-A-Schnittstelle muß sich gegenüber der Anwendungsseite so verhalten wie ein Register oder wie ein Speicherschaltkreis – mit anderen Worten: wie eine Slave-Einrichtung an einem üblichen Bussystem. Einige Mikrocontroller haben derartige Slave-Ports. Gibt es keine, müssen die Funktionen mit Buskoppelschaltkreisen (oder in einer CPLD) nachgebildet werden (Abb. 1.6).



1 - externer Datenbus (bidirektional); 2 - Eingaberegister; 3 - Ausgaberegister; 4 - interner Datenbus (bidirektional); 5 - externe Strobeleitung (Übernahmetakt); 6 - externe Leserlaubnisleitung (Datenaufschaltung); 7 - Interruptanforderungsflipflop; 8 - Eingabelesesignal; 9 - Ausgabeschreibsignal.

Abb.1.6 Ein Slave-Port

Eingabe:

Strobeleitung 5 bewirkt Datenübernahme nach Register 2. Dabei wird zugleich Flipflop 7 gesetzt. Das löst einen Interrupt im Mikrocontroller aus. Der Interruptbehandler liest den Inhalt des Registers 2, indem er den Bus 4 auf Eingabe schaltet und das Lesesignal 8 aktiviert. Hierdurch wird zugleich das Flipflop 7 gelöscht.

Ausgabe:

Der Mikrocontroller stellt die auszugebende Daten im Register 3 bereit (Bus 4 auf Ausgabe, Datenübernahme mittels Schreibsignal 9). Die angeschlossene Einrichtung liest die Daten durch Erregen der Leitung 6.

1.1 Elementare Buskoppelschaltkreise

Trotz des heutigen Standes der Schaltungsintegration werden Buskoppelschaltkreise nach wie vor in großen Stückzahlen eingesetzt. Typische Gründe dafür:

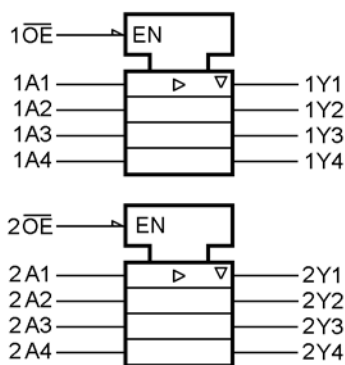
- besondere Forderungen an die Treibfähigkeit,
- besondere anwendungsseitige Forderungen. Beispiele: Ziehen/Stecken von Einrichtungen während des Betriebs (Hot Plugging), Schutz der funktionellen Schaltkreise, Pegelanpassung, vereinfachte Fehlerbeseitigung – ein über den Bus verursachter Ausfall betrifft nur den (kostengünstigen) Koppelschaltkreis, nicht aber die (teuren) funktionellen Schaltkreise.
- Zusammenfügen verschiedener Technologien, Aufbau der Einrichtung aus verschiedenartigen Bauelementen (wie bei kleinen bis mittleren Stückzahlen allgemein üblich).
- Kostensenkung.

Es liegt nahe, Typen zu bevorzugen, die bewährten Industriestandards entsprechen (Abb. 1.7 bis 1.9) und ggf. kleinere Nachteile (im Vergleich zu diesem oder jenem Exoten) in Kauf zu nehmen.

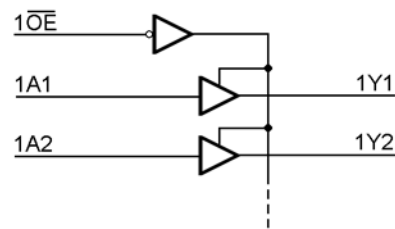
Woran erkennen wir, ob ein Schaltkreis ein „Industriestandard“ ist oder nicht? – Industriestandard-Typen

- gibt es seit einiger Zeit,
- werden in nahezu allen Baureihen angeboten,
- werden auch für neue Baureihen angekündigt,
- haben in den Typenliste der Hersteller keine einschränkenden Vermerke,
- werden in Massenprodukten eingesetzt (z. B. auf Speichermoduln),
- sind bisweilen auffallend preisgünstig.

a) Bustreiber (Puffer) '244



Der Schaltkreis enthält zwei Blöcke zu vier Tri-State-Stufen mit gemeinsamem Erlaubnissignal (Enable).



b) Transceiver '245

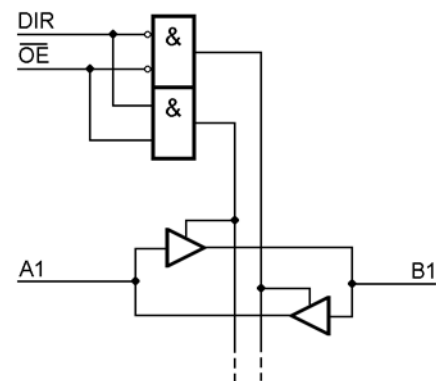
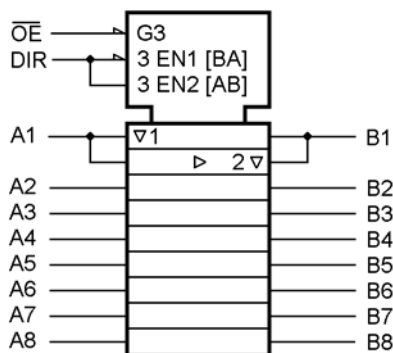


Abb.1.7 Herkömmliche Buskoppelschaltkreise (Beispiele)

- Bustreiber (Puffer). Eine Signalflußrichtung (von A nach Y). A - Dateneingänge; Y - Ausgänge (Busanschlüsse); OE bzw. EN - Erlaubniseingänge (betreffen je 4 Datensignale). Bei aktivem Erlaubnissignal werden die zugehörigen Datensignale zum Bus durchgeschaltet. Ist das Erlaubnissignal inaktiv, sind die zugehörigen Ausgänge (Y) hochohmig.
- bidirektionale Koppelstufe (Transceiver). Eine von zwei Signalflußrichtungen wählbar (Tabelle 1.2). Beide Seiten A, B, haben Tri-State-Anschlüsse. OE - Erlaubniseingang; DIR - Richtungssteuereingang (Direction Control Input).

DIR	OE	Wirkung
0	0	Richtung von B nach A; A-Ausgänge aktiv
0	1	Richtung von B nach A; A-Ausgänge hochohmig
1	0	Richtung von A nach B; B-Ausgänge aktiv
1	1	Richtung von A nach B; B-Ausgänge hochohmig

Tabelle 1.2 Zur Wirkungsweise der Steuersignale des Transceivers '245

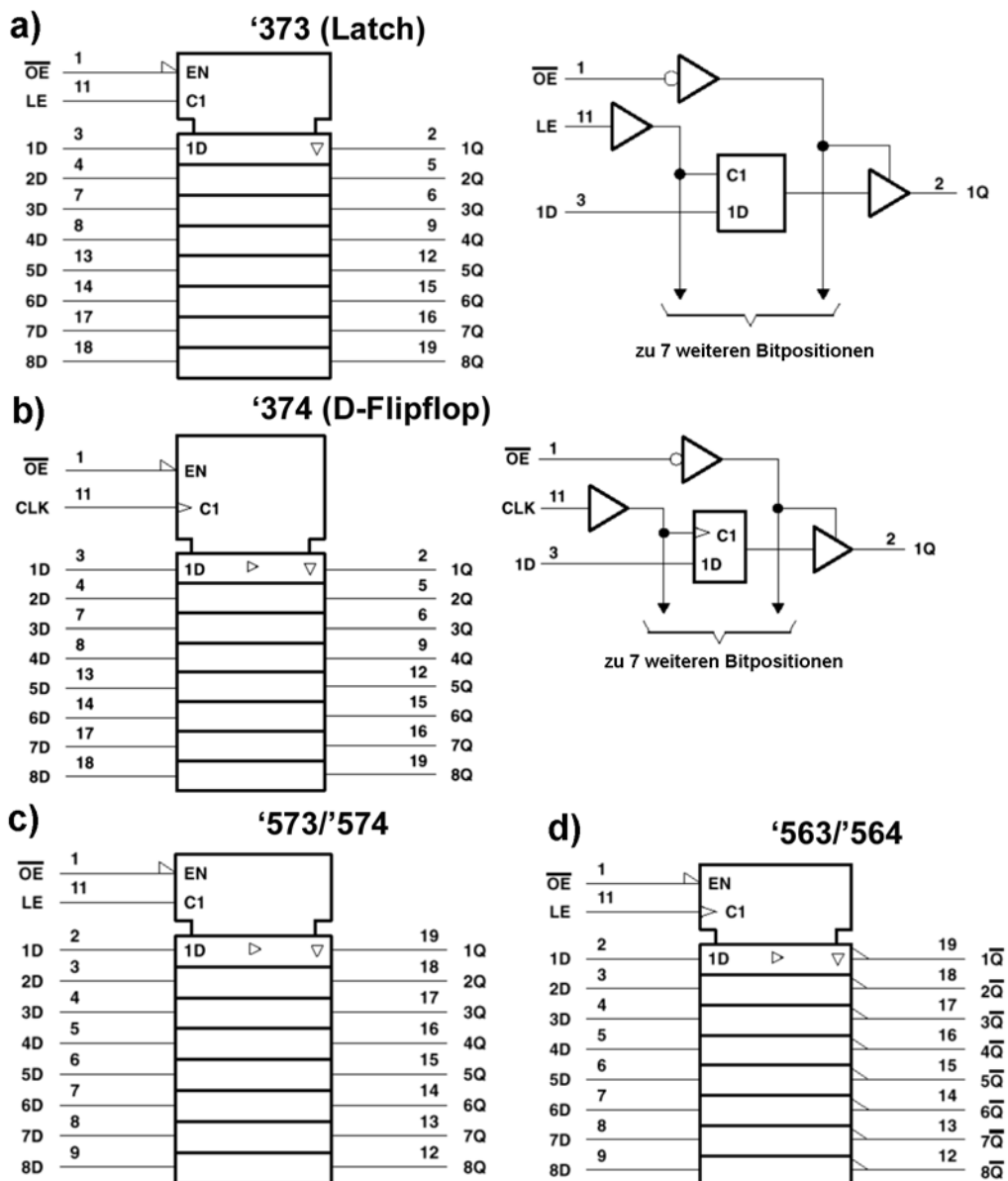


Abb. 1.8 Typische Register - eine kleine Auswahl. a) - Latch-Register; b) - D-Flipflop-Register; c) - Register mit anderer Anschlußbelegung; d) - Register mit negierten Ausgängen.

Die Schaltkreise haben Tri-State-Ausgänge. Braucht man zweiwertige Ausgänge, ist der Erlaubniseingang (OE) fest mit Masse zu verbinden. Gängige Breiten (Anzahl der Bitpositionen): 8, 16, 18, 20, 32, 36. Es gibt verschiedene Anschlußbelegungen (Pinouts). Bei a) und b) sind Ein- und Ausgänge gleichsam gemischt (Anschluß 2 = Ausgang, Anschluß 3 = Eingang usw.), bei c) und d) liegen sich hingegen Ein- und Ausgänge auf beiden Seiten gegenüber (1, 2 usw. sind Eingänge, 19, 18 usw. sind Ausgänge). Der Zweck: die Auslegung kostengünstiger Leiterplatten zu erleichtern¹⁾.

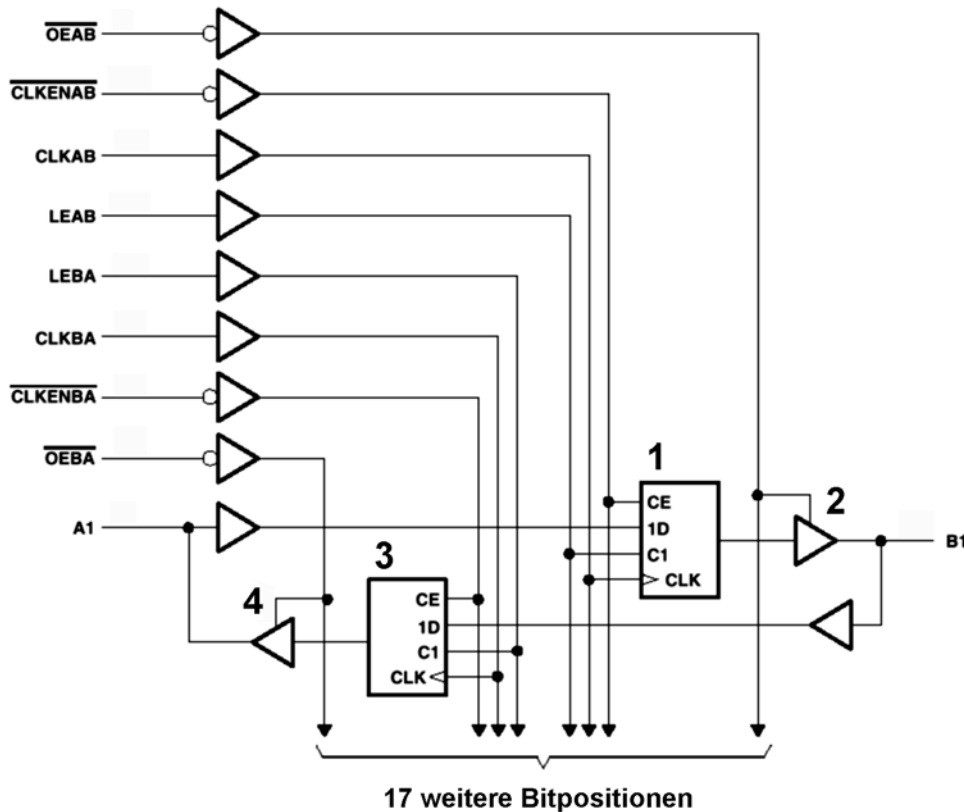


Abb.1.9 Eine Stufe eines Universal Bus Transceivers (74LVT16601; Texas Instruments)

Ein Schaltkreis des angegebenen Typs enthält insgesamt 18 solcher Stufen. Die Steuereingänge sind allen Stufen gemeinsam. 1 - Datenspeicher für Richtung A - B; 2 - Tri-State-Treiber für B-Anschluß; 3 - Datenspeicher für Richtung B - A; 4 - Tri-State-Treiber für A-Anschluß.

Die Funktionsweise wollen wir zunächst anhand der Signalflußrichtung von links nach rechts (von A nach B) beschreiben (A ist Eingang, B Ausgang). OEAB - Output Enable Richtung A-B; CLKENAB - Clock Enable Richtung A-B; CLKAB - Takt Richtung A-B; LEAB - Latch Enable Richtung A-B. Sinngemäß erklären sich die Steuersignale der Gegenrichtung (B-A)

Aktivieren und Deaktivieren des Ausgangs (B): mittels OEAB (Low: Ausgang aktiv, High: Ausgang hochohmig).

Verhindern, daß der Eingang (A) über den Weg B - A beeinflusst wird: OEBA auf High (deaktiviert Treiber 4).

1) einfache Strukturen – z. B. vom Mikrocontroller zum Adreß-Latch, vom Adreß-Latch zum Speicher (vgl. 8051) – passen bei Anschlußbelegung c) auf eine einzige Leiterplattenebene.

Steuerung der Betriebsweise des Datenspeichers 1: über LEAB:

- LEAB = Low: Datenspeicher wirkt als D-Flipflop. Übernimmt mit der Low-High-Flanke des Taktes CLKAB die Belegung des A-Eingangs (Registerfunktion). Damit CLKAB wirksam werden kann, muß CLKENAB = Low sein (Übernahmesteuerfunktion).
- LEAB = High: Datenspeicher wirkt als Latch oder als einfache Durchreiche.

Durchreiche (keine Speicherfunktion): Solange LEAB = High ist, wirkt der Datenspeicher als Durchreiche.

Latch-Funktion: Wird LEAB von High auf Low geschaltet, so wirkt der Datenspeicher als Speicherglied und hält die Eingangsbelegung zum Zeitpunkt der High-Low-Flanke von LEAB (Verhalten eines transparenten Latches). In dieser Betriebsart darf CLKAB bei LEAB = Low nicht wirksam werden (sonst: Informationsübernahme mit Low-High-Flanke).

Durch entsprechendes Beschalten der Steuereingänge können viele der gängigen Buskoppelstufen nachgebildet werden. Beispiele:

1. Verhalten ähnlich '244 (einfache Durchreiche mit Tri-State-Ausgängen)

OEAB wirkt als OE. LEAB und OEBA fest auf High. Restliche Steuersignale auf beliebige Festwerte.

2. Verhalten ähnlich '245 (bidirektionaler Treiber)

OEAB und OEBA wirken als Erlaubnissignale auf der B- bzw. auf der A-Seite. LEAB und LEBA fest auf High. Restliche Steuersignale auf beliebige Festwerte. Wird eine zum '245 kompatible Steuerung mit DIR und OE gewünscht, sind OEAB und OEBA über UND-Gatter gemäß Abb. 5.87b anzusteuern.

3. Verhalten ähnlich 373/573 (Latch-Register)

OEAB wirkt als OE. OEBA fest auf High. Datenübernahme mit LEAB (wirkt als Takt). Restliche Steuersignale auf beliebige Festwerte.

4. Verhalten ähnlich 374/574 (D-Flipflop-Register)

OEAB wirkt als OE. OEBA fest auf High. Datenübernahme mittels Takt an Takteingang CLKAB. Übernahmesteuerung muß aktiv sein. Dazu CLKENAB auf Low. CLKENAB kann als Erlaubnissignal verwendet werden (vgl. Eingang CE in Abb. 5.23b). LEAB auf Low. Restliche Steuersignale auf beliebige Festwerte.

1.2 Das Schieberegister als Universalinterface

Das Schieberegister hat einen bedeutsamen Vorteil: man kann es so lang auslegen wie man will und braucht, um Bits zu transportieren, nur einen Dateneingang, einen Datenausgang und einen Takt. Da man mit den übertragenen Bits natürlich noch etwas anfangen will, braucht man zusätzlich irgendwelche Steuersignale. Dafür genügt eine einzige Leitung: wird sie aktiviert, so wird die eingeschobene Information zwecks Ausgabe übernommen und durch einzugebende Information ersetzt, die dann nur noch herausgeschoben werden muß (Abb. 1.10).

Die einzelnen Einrichtungen enthalten Schieberegisteranordnungen mit parallelen Ein- und Ausgängen. Die auszugebende Information wird Bit für Bit durchgeschoben. Anschließend wird STROBE erregt. Dies bewirkt zum einen das Laden der Parallelregister mit den eingeschobenen Daten (Ausgabe) und zum anderen das Laden der Schieberegister mit den Daten der jeweiligen Umgebung (Eingabe). Mit weiteren Schiebetakten wird dann die übernommene Information Bit für Bit herausgeschoben.

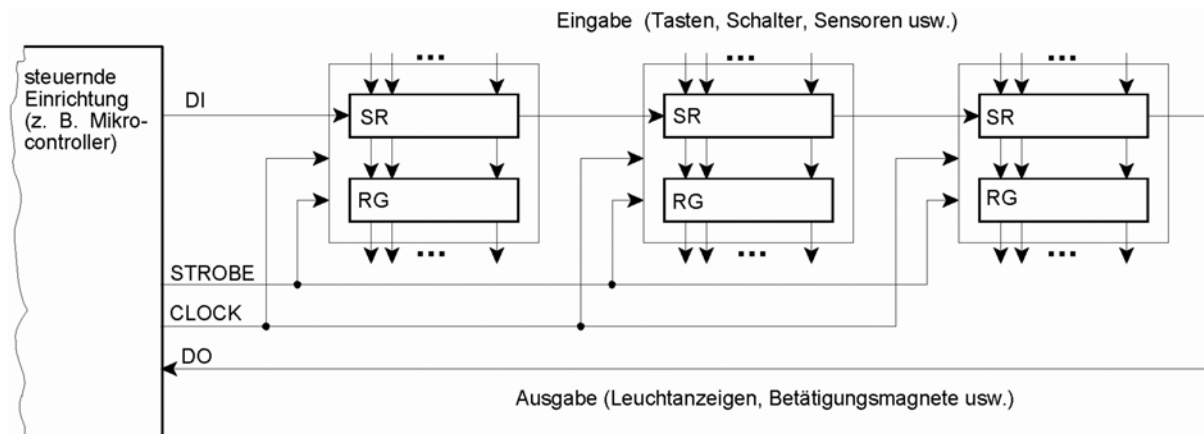


Abb. 1.10 Ein einfaches Schieberegister-Interface. SR - Schieberegister; RG - Parallelregister (Halteregister); DI - Dateneingang; DO - Datenausgang; CLOCK - Schieberegisterbetakt; STROBE - Parallelübernahmeimpuls

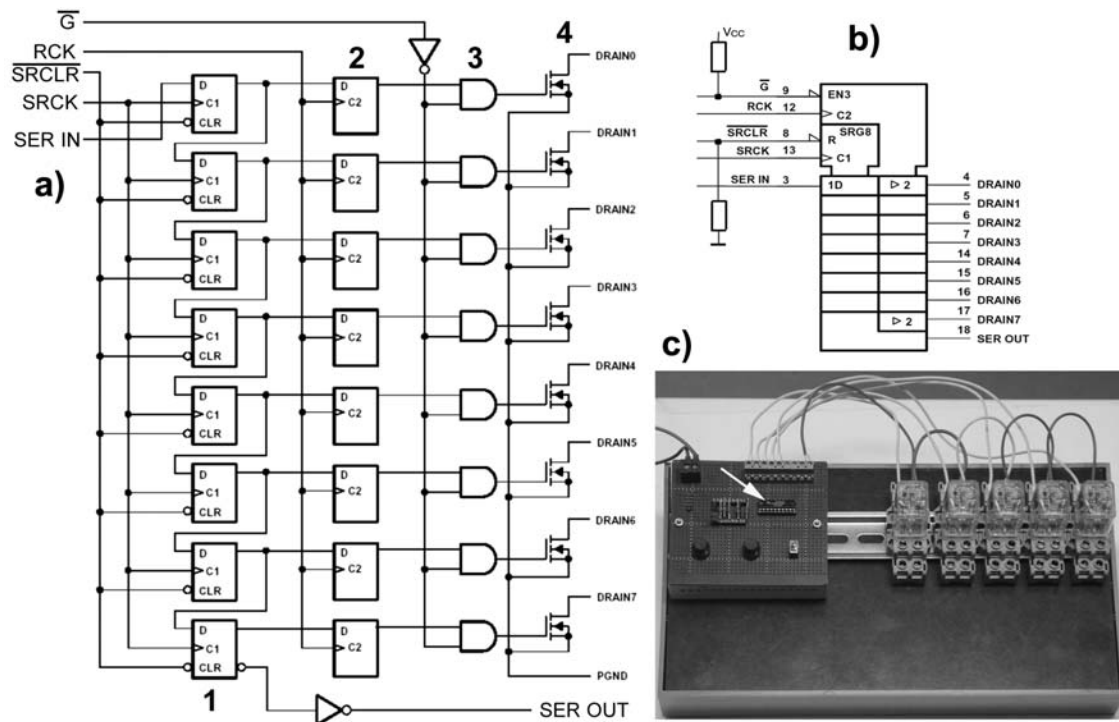
Es gibt viele Schaltkreise mit solchen Schnittstellen. Einige sind als Privatlösungen anzusprechen (Abb. 1.11, 1.12), einige haben den Charakter von Industriestandards. Die wichtigsten: I²C-Bus, Microwire und SPI (Tabelle 1.3, Abb. 1.13 bis 1.16).

Die Anordnung von Abb. 1.11 entspricht dem in Abb. 1.10 dargestellten Prinzip. Einschreiben der Bits über SER IN mit Schieberegisterbetakt SRCK. Parallelübernahme mit RCK. Es können mehrere Schaltkreise hintereinandergeschaltet werden (Verbindung SER OUT - SER IN). Anwendungspraktisch wichtig ist, daß kein Leistungs-FET unberechtigt aktiv werden darf (z. B. nach dem Rücksetzen). Deshalb gibt es eine direktwirkende Sperrmöglichkeit über die Gatter 4. Sperrsignal G wird über Pull-up-Widerstand auf High gezogen und erst nach vollständiger Initialisierung programmseitig aktiviert. Zudem kann das Schieberegister gelöscht werden (Eingang SCLR). Achtung: Das Halteregister 2 läßt sich nicht direkt löschen. Vor der Freigabe der Sperrgatter 4 ist zunächst der Inhalt des gelöschten Schieberegisters 1 ins Halteregister 2 zu übernehmen (Impuls auf RCK).

Es gibt verschiedene Abwandlungen des einfachen Schieberegisterprinzips von Abb. 1.10. Abb. 2.12 zeigt ein Beispiel. Das Ziel der Entwickler: mit nur zwei Signalleitungen (Takt- und Datenleitung) auszukommen. Hier hat man das Start-Stop-Prinzip der seriellen Schnittstelle aufgegriffen. Das erste gesendete Bit wird als Startbit interpretiert. Dann folgen die eigentlichen Informationsbits (hier: 35). Die Takte werden intern mitgezählt. Nach dem 36. Takt gelangt der Schaltkreis wieder in den Ausgangszustand (internes Rücksetzen) und wartet auf das nächste Startbit.

Hinweis:

Bei derartigen Interfaces auf die Impulsbreiten bzw. Taktfrequenzen achten (Datenblatt). Der Schaltkreis von Abb. 1.11 kommt noch mit 20 ns breiten Impulsen zurecht (entspricht einem Schieberegisterbetakt von 25 MHz), der Schaltkreis von Abb. 1.12 hingegen hat eine maximale Taktfrequenz von nur 500 kHz. Naiv programmierte Ausgabeschleifen sind typischerweise zu schnell (auch in „langsamen“ Mikrocontrollern).



a) - Schaltbild (nach Texas Instruments); b) - Schaltsymbol mit Widerstandsbeschaltung; c) - Anwendung als Relaisstreiber (Labormuster). 1 - Schieberegister; 2 - Haltereister; 3 - Sperrgatter; 4 - Leistungs-FETs.

Abb.1.11 Leistungschaltkreis mit Schieberegisterinterface

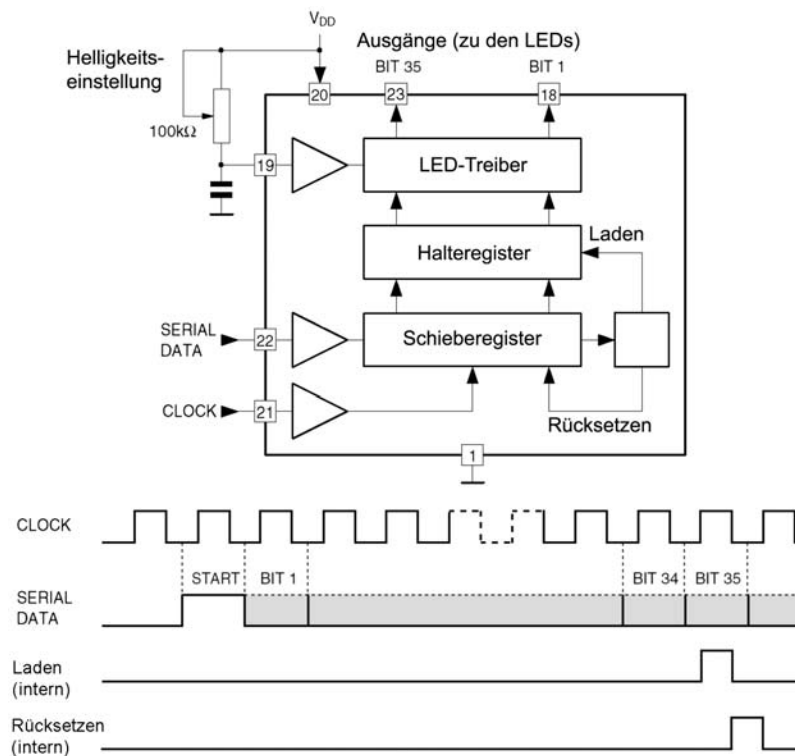
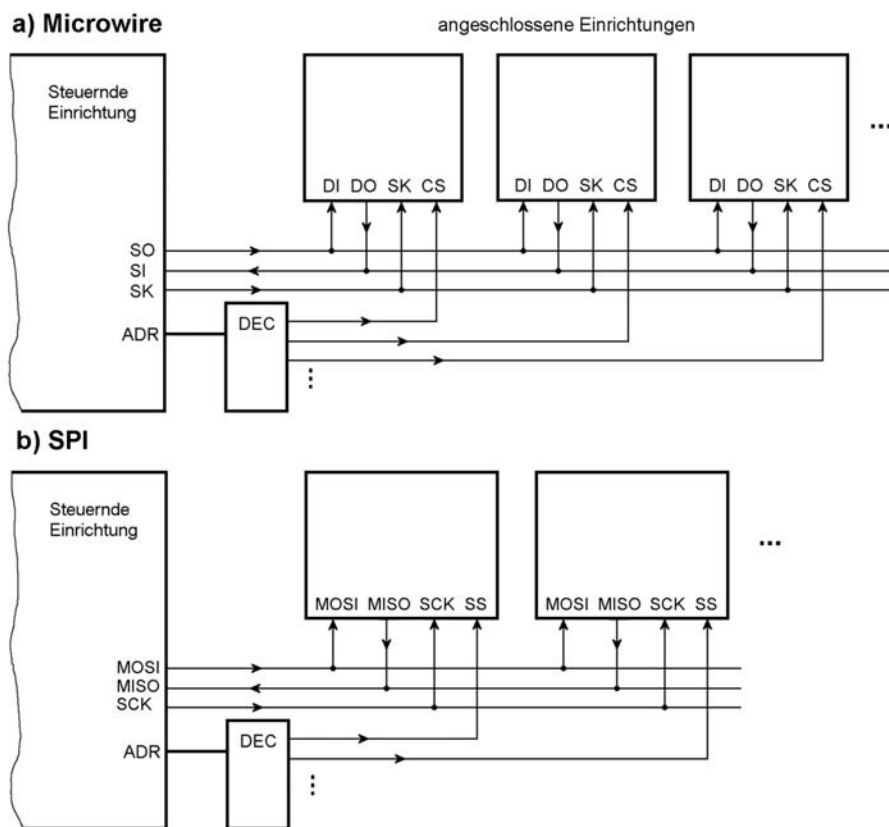


Abb.1.12 LED-Treiber MC4541 (nach ST Microelectronics)

	ÎC	Microwire	SPI
Entwicklungsfirma	Philips	National Semiconductor	Motorola
Anzahl der Signalleitungen	2	3 bzw. 4*)	3 bzw. 4*)
Architektur	Multimaster-Bus	Bus mit zentralem Master	Bus mit zentralem Master
Schaltkreisauswahl	über Adreßdecodierung	zentrale Auswahl über CS-Eingang	zentrale Auswahl über CS-Eingang
Taktfrequenz**) (Richtwerte)	100 bzw. 400 kBits/s, 3,4 MBits/s	1...3 MHz	1...5 MHz
Signalprotokolle	komplizierter	einfacher	einfacher

*) 3 Interfaceleitungen + 1 Schaltkreisauswahlsignal.**): entspricht der maximalen Datenrate in Bits/s. Betriebsspannungsabhängig (je geringer die Betriebsspannung, desto geringer die spezifizizierte maximale Taktfrequenz)

Tabelle 1.3 Industriestandards im Überblick



- a) Microwire: Dateneingang DI, Datenausgang DO, Schiebetakt SK und Schaltkreisauswahl CS. Es ist möglich, DI und DO zu einer bidirektionalen Datenleitung zusammenzuschalten.
- b) SPI: Dateneingang MOSI, Datenausgang MISO; Schiebetakt SCK, Schaltkreisauswahl SS. (MOSI = Master Out/Slave In; MISO = Master In/Slave Out; SCK = Shift Clock; SS = Slave Select.)

Abb.1.13 Microwire und SPI

Microwire

Die zentrale Steuerung wählt jeweils eine angeschlossene Einrichtung aus, indem sie die betreffende CS-Leitung aktiviert. Auszugebende Information wird über DO geliefert und in der ausgewählten Einrichtung mit der Vorderflanke der SK-Impulse übernommen; einzugebende Information wird über DI mit SK-Taktimpulsen Bit für Bit abgeholt. Das Schieben ist byteweise organisiert. Was die einzelnen Bytes bedeuten, ist schaltkreisspezifisch.

SPI

Der jeweilige Slave wird durch Aktivierung des Eingangs SS ausgewählt. Der Datentweg zwischen Master und ausgewähltem Slave ist eine Ringstruktur aus zwei Schieberegistern (Abb. 1.14, 1.15).

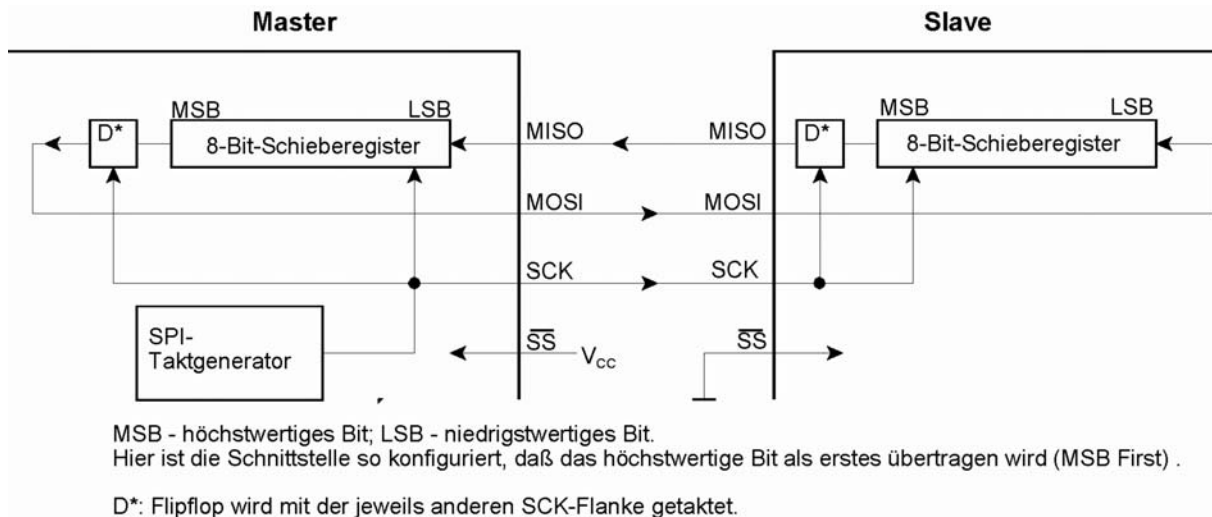


Abb. 1.14 SPI: der Datenweg zwischen Master und Slave

Abb. 1.14 zeigt eine Art Ersatzschaltung, um die typischen Besonderheiten zu veranschaulichen. Es wird byteweise geschoben. Das Byte im Schieberegister des Masters gelangt in den Slave, während gleichzeitig das im Schieberegister des Slaves stehende Byte in den Master geschafft wird. Dabei wird mit der einen Taktflanke der auf der jeweiligen Datenleitung anliegende Wert übernommen und in das jeweils empfangende Schieberegister eingeschoben, mit der anderen wird das jeweils nachfolgende Bit ausgeschoben. Diese Betriebsweise vermeidet Takttoleranzprobleme – deshalb kann SPI mit höheren Taktfrequenzen betrieben werden als Microwire.

Die SPI-Hardware in einem Mikrocontroller ist typischerweise als autonom arbeitende State Machine ausgelegt, die – von der Software gesteuert – wahlweise als Master oder als Slave betrieben werden kann. Die Konfigurationsmöglichkeiten sind schaltkreisspezifisch. Programmseitig einstellbar sind u. a.:

- das Übertragungsformat: Einzelbyteübertragung (nach Übertragung eines jeden Bytes wird SS wieder inaktiv) oder Mehrbyteübertragung (aufeinanderfolgende Bytes werden übertragen, während SS aktiv bleibt),
- die Polarität des SCK-Signals (aktiv High oder aktiv Low),
- die Schiebeordnung (höchstwertiges oder niedrigstwertiges Bit zuerst).

Multi-Master-Betrieb

SPI kann als echter Bus betrieben werden, auf den sich mehrere Einrichtungen wahlweise aufschalten können. Die Mastervermittlung muß aber gesondert verwirklicht werden. (Typischerweise kann man den SS-Anschluß im Master-Betrieb so konfigurieren, daß dessen Aktivierung das Freigeben des Busses veranlaßt.)

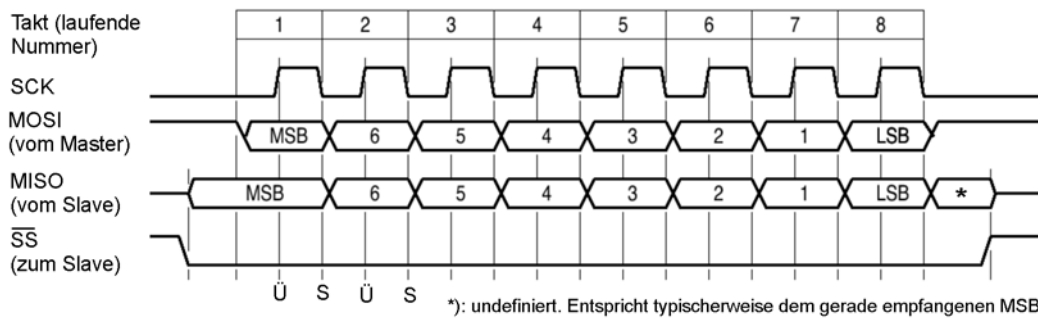


Abb.1.15 Datentransport über SPI (nach Motorola). Ü - Datenübernahme; S - Ausschieben

Wir zeigen hier nur einen der möglichen Betriebsfälle. Das Schieben eines Bytes erfordert 8 Takte. Zuvor ist SS zu aktivieren und nach dem Schieben wieder zu deaktivieren. Datenübernahme mit der Low-High-Flanke von SCK, Weiterschieben mit der High-Low-Flanke. Die letzte High-Low-Flanke bewirkt im Slave ein weiteres Schieben, in dessen Ergebnis MISO typischerweise mit dem ersten der zuvor eingeschobenen Bits belegt wird.

Der I²C-Bus

I²C = IIC = Inter Integrated Circuit. Das Bussystem wurde entwickelt, um Schaltkreise auf einer größeren Leiterplatte auf kostengünstige Weise miteinander zu verbinden. Der I²C-Bus verbindet mehrere Slave-Einrichtungen mit einer steuernden Einrichtung (dem Master). Es gibt nur zwei Leitungen:

- Serieller Takt SCL (Serial Clock). Diese Leitung wird ausschließlich vom Master angesteuert.
- Serielle Daten SDA (Serial Data). Diese Leitung ist bidirektional; je nach Übertragungsrichtung wird sie vom Master oder vom jeweils ausgewählten Slave angesteuert.

Treiberstufen

Die Treiberstufen des I²C-Bus sind Open-Drain-Ausgänge. Deshalb sind die Busleitungen mit Pull-up-Widerstände beschaltet (z. B. mit 4,7 kΩ). Typischer Treiberstrom: 3 mA.

Geschwindigkeit

Typischerweise liegt die maximale Taktfrequenz bei 100 kHz (praxisüblich: bis zu 80 kHz). Zudem gibt es zwei weitere Betriebsarten: (1) bis 400 kHz (Fast Mode), (2) bis 3,4 MHz (High Speed Mode). Der I²C-Bus ist voll statisch (also mit beliebig geringer Taktfrequenz) betreibbar.

Slave-Auswahl (1). 7-Bit-Adressierung

Das erste vom Master gesendete Byte (Steuerbyte) enthält die Slave-Adresse und die Zugriffskennung (Abb. 1.16). Mit den 7 Bits der Slave-Adresse könnte man theoretisch 128 Einrichtungen adressieren. Einige Belegungen sind aber reserviert, und für bestimmte Arten von Einrichtungen (z. B. für serielle EEPROMs) sind bestimmte Adreßformate festgelegt (Abb. 1.16b).

Slave-Auswahl (2). 10-Bit-Adressierung

Dem Steuerbyte folgt ein weiteres Adreßbyte nach. Die Bits 7...3 des Steuerbytes kennzeichnen die Adressierungsweise (Belegung mit Festwert 11110B), die Bits 2 und 1 enthalten zwei der 10 Adreßbits.

Signalfolgen und Zugriffsabläufe

Tabelle 1.4 und Abb. 1.17 veranschaulichen die elementaren Signalfolgen auf den beiden Busleitungen. Anhand von Abb. 1.18 wollen wir das Grundsätzliche der Zugriffsabläufe erläutern. Unser Beispiel: ein serieller EEPROM.

a) allgemeine Struktur

7	Slave-Adresse						1	0
							R/W	

b) Beispiel (serielle EEPROMs)

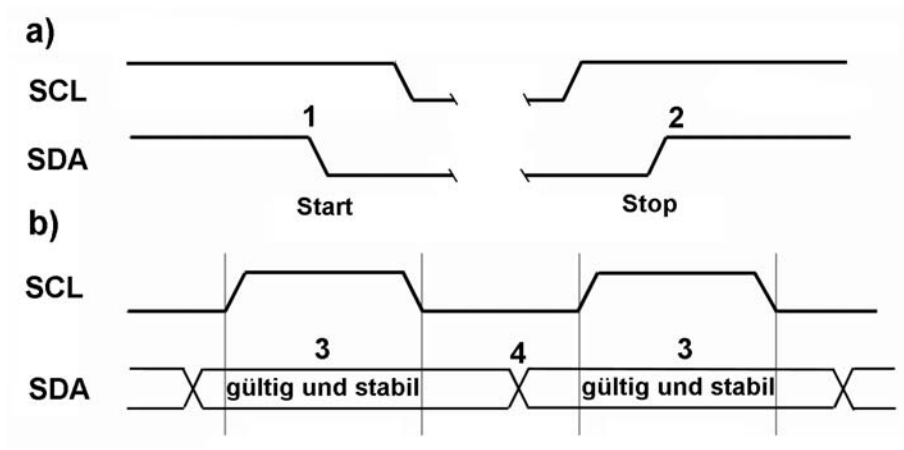
Typkennung				Slave-Adresse			R/W

Abb.1.16 Das Steuerbyte

Die Slave-Adresse wählt die Einrichtung (= einen Schaltkreis) aus, die Typkennung kennzeichnet die Art der Einrichtung, die Zugriffskennung (R/W) bestimmt die Richtung der Datenübertragung (0 = Schreiben, 1 = Lesen).

Signalfolge	Leitung SCL	Leitung SDA
Start	High	Flanke High => Low
Stop	High	Flanke Low => High
Datenübertragung	Flanke Low => High	Datenbit (muß während der Low-High-Flanke des Taktes stabil anliegen)
Bestätigung (Acknowledge)	Low-High-Flanke (9. Takt der Datenübertragung)	wird von der jeweils empfangenden Einrichtung (Master oder Slave) auf Low gezogen

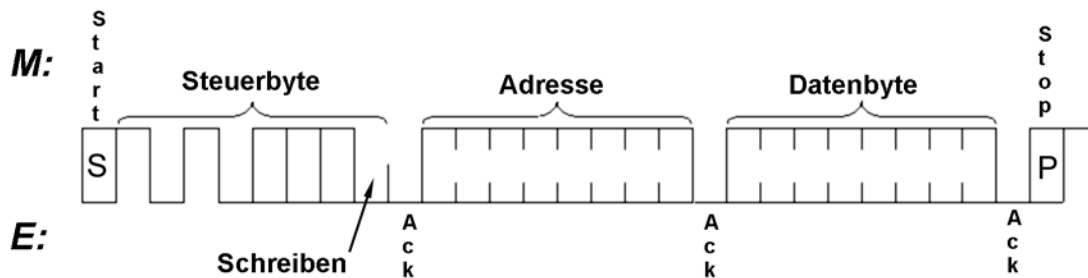
Tabelle 1.4 Elementare Signalfolgen des I²C-Bus



a) - Start- und Stopsignalisierung; b) - Gültigkeit der Datenbelegung in Bezug auf den Takt. 1 - Start = SDA-Flanke von High nach Low bei SCL = low ; 2 - Stop: SDA-Flanke von Low nach High bei SCL = Low. 3 - Datenbelegung liegt stabil an, wenn SCL = High; 4 - Datenbelegung darf sich nur dann ändern, wenn SCL = Low.

Abb.1.17 Elementare Signalfolgen des I²C-Bus

a) Schreiben



b) Lesen

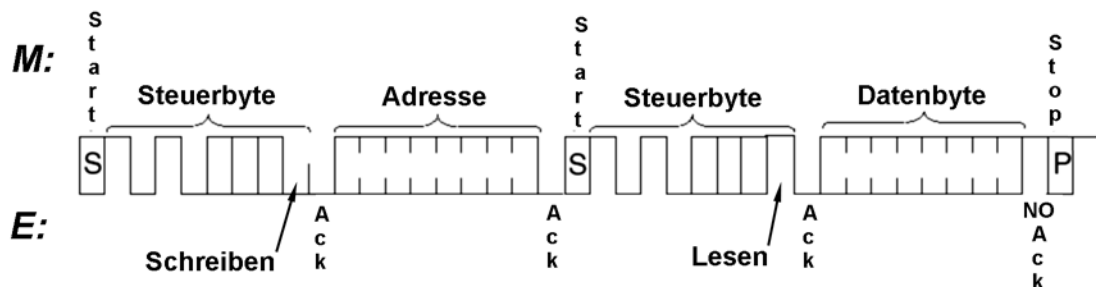


Abb.1.18 Elementare Zugriffsabläufe am Beispiel eines seriellen EEPROMs (nach Microchip)

Es sind die einzelnen Bitpositionen des seriellen Datenstroms auf der Leitung SDA dargestellt. Jeder Bitposition entspricht ein Taktimpuls auf der Leitung SCL. M - Aktivitäten des Masters; E - Aktivitäten des EEPROMs; Start (S) - Startbedingung; Ack - Bestätigung (Acknowledge); No Ack - keine Bestätigung; Stop (P) - Stopbedingung.

a) Schreiben

Nach der anfänglichen Startbedingung kommt das Steuerbyte. $R/W = 0$ zeigt einen Schreibzugriff an. Das Steuerbyte (Schreibkommando) wird von EEPROM bestätigt (Acknowledge). Darauf folgt die Adresse. Bei entsprechend geringer Speicherkapazität genügt ein Adreßbyte (EEPROMs mit größerer Speicherkapazität erfordern 2 Adreßbytes). Der Empfang des Adreßbytes wird vom EEPROM bestätigt (Acknowledge). Dann folgt das Datenbyte. Nach dessen Bestätigung (Acknowledge) sendet der Master eine Stopbedingung. Daraufhin wird im EEPROM der Programmiervorgang ausgelöst.

b) Lesen

Das Lesen beginnt mit einem Schreibkommando, um die Adresse in den EEPROM zu schaffen. Nach der Bestätigung (Acknowledge) der Adreßübertragung sendet der Master eine weitere Startbedingung und ein Steuerbyte mit $R/W = 1$ (Lesekommando). Nach dessen Bestätigung beginnt der EEPROM, das Datenbyte zu senden. Es gibt zwei Ablaufvarianten:

- Lesen eines einzelnen Bytes (Random Read). Der Master bestätigt den Empfang des Datenbytes *nicht* (No Acknowledge), hält also während der betreffenden Taktperiode SDA auf High, und sendet dann eine Stopbedingung (vgl. Abb. 1.17b).
- Fortlaufendes Lesen (Sequential Read). Der Master bestätigt den Empfang des ersten Datenbytes (Acknowledge). Darauf folgt sofort das zweite usw. Jede weitere Bestätigung führt zum Liefern eines weiteren Datenbytes. Um das Lesen zu beenden, muß der Master auf das jeweils letzte Datenbyte so antworten, wie in Abb. 1.17b gezeigt (keine Bestätigung, dann Stopbedingung). Die EEPROMs haben Adreßzähler, die den gesamten Speicher fortlaufend adressieren (man könnte also den vollständigen Speicherinhalt im Rahmen eines einzigen Lesekommandos abholen).

Schieberegisterinterfaces selbstgebaut

Das Nachbauen von SPI, I²C, IEEE 1149 usw. lohnt sich meist nicht (zu kompliziert). Die typische Eigenentwicklung beruht meist auf dem einfachen Prinzip von Abb. 1.10. Manchmal sind weitere Vereinfachungen möglich.

Schieberegisterschaltkreise der Logikbaureihen

Das herkömmliche Sortiment ist reichhaltig (Tabelle 1.5). Schieberegister sind heutzutage allerdings eher Exoten (oft teurer, nur in wenigen Baureihen erhältlich). Manche Typen (vor allem der LS-Baureihe) sind für Neuentwicklungen nicht zu empfehlen (veraltet, keine garantierte Verfügbarkeit).

Typ	Ausführung	Typ	Ausführung
74x164	8 Bits, parallele Ausgänge	74x195	4 Bits, parallele Eingänge, parallele Ausgänge
74x165	8 Bits, parallele Eingänge	74x595	8 Bits, parallele Ausgangs-Latches
74x166	8 Bits, parallele Eingänge	74x597	8 Bits, parallele Eingangs-Latches
74x194	4 Bits, parallele Eingänge, parallele Ausgänge	74LS671/ 672	4 Bit parallele Eingänge, paralleles Ausgangsregister

Tabelle 1.5 Schieberegisterschaltkreise (Auswahl)

Schieberegister mit Buskoppelschaltkreisen

Eine Alternative: wir verwenden „Industriestandard“-Buskoppelschaltkreise mit D-Flipflop-Registern und bilden den Schiebeweg auf der Leiterplatte nach (Abb. 1.19 bis 1.21).

Parallele Ausgabe

Kein Problem, da alle Ausgänge zugänglich sind. Ggf. wird ein weiteres Register als paralleles Ausgaberegister angeschlossen.

Parallele Eingabe

Es liegt nahe, 2-zu-1-Multiplexer einzusetzen (Umschaltung zwischen Schieben und Parallelübernahme). Das erfordert aber vergleichsweise viele Schaltkreise. Womöglich günstiger: Tri-State-Koppelstufen oder entsprechende Register.

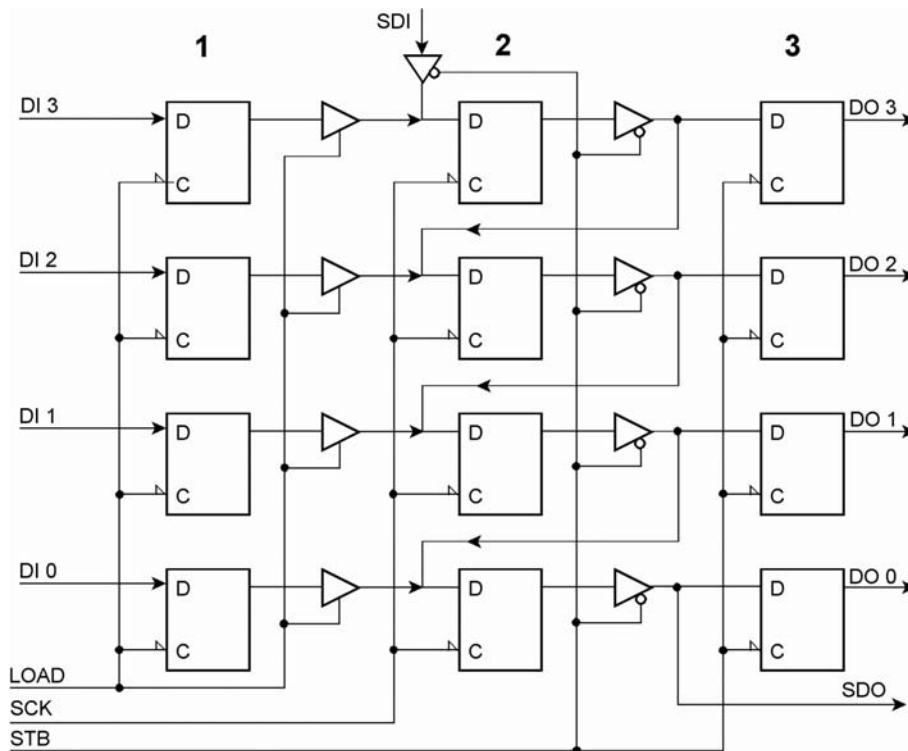
Schieberegister in CPLDs

Ein Flipflop kostet eine ganze Makrozelle. Die Parallelübernahme von Eingangssignalen bedeutet praktisch keinen Zusatzaufwand, ein paralleles Ausgangsregister erfordert hingegen eine weitere Makrozelle je Bitposition (Abb. 1.22). Wichtig: soviel wie möglich mit einem gemeinsamem Takt erledigen (ähnlich JTAG 1149, nur einfacher)¹⁾.

Auf das parallele Ausgangsregister verzichten

Das ist dann möglich, wenn es nicht stört, daß die nachgeordneten Schaltungen den Schiebeporgang mitbekommen. Anwendungsbeispiel: LED-Anzeigen mit Einzelansteuerung (kein Multiplexbetrieb). Voraussetzungen: (1) es wird immer nur dann eine neue Belegung eingeschoben, wenn sich die Anzeige ändert (Sache der Anwendungsprogrammierung), (2) die Änderungen kommen nicht allzu oft vor (das ist z. B. dann der Fall, wenn sich Änderungen der Anzeige auf Grund von Bedienhandlungen ergeben).

1) individuelle Takte belegen zumeist weitere Ressourcen in den Makrozellen.



1 - Eingaberegister; 2 - Schieberegister; 3 - Ausgaberegister. SCK = Schiebetak; LOAD = Parallelübernahme der Eingangsbelegung; STB = Parallelübernahme der Ausgangsbelegung.

Abb.1.19 Schieberegisteranordnung mit Ein- und Ausgaberegistern

Um die Schaltung einfach zu halten, werden zwei Steuerleitungen verwendet. Der Schiebepuls ist über Tri-State-Stufen geführt. Beim Schieben sind LOAD und STB = Low. Somit sind die Tri-State-Treiber des Schieberegisters 2 aktiv. Nach dem Schieben schalten wir STB auf High. Damit wird die Schieberegisterbelegung ins Ausgaberegister 3 übernommen, und die Tri-State-Stufen des Schieberegisters 2 werden freigegeben. LOAD bewirkt, daß die Eingangsbelegung ins Eingaberegister 1 übernommen wird und daß dessen Tri-State-Koppelstufen aktiviert werden. Ein Impuls auf SCK veranlaßt die Übernahme ins Schieberegister 2. Abschließend wird zunächst LOAD und dann STB auf Low gebracht. Nun kann ein neuer Schiebepuls beginnen.

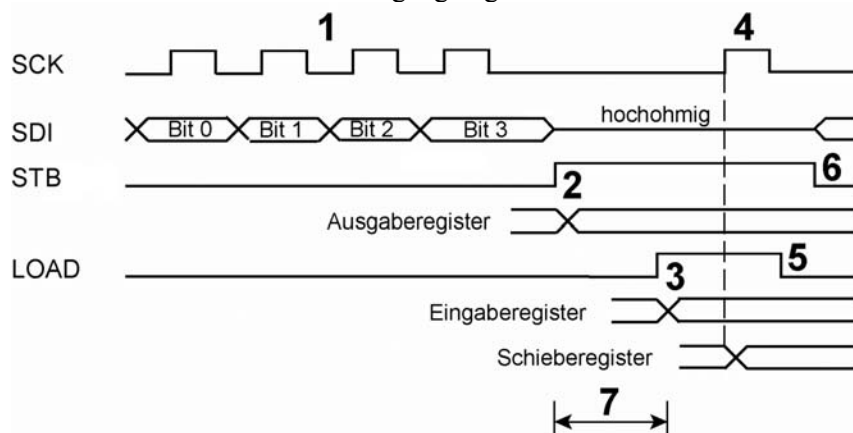


Abb.1.20 Ein Schiebepuls

1 - Einschreiben von vier Bits; 2 - Übernahme ins Ausgaberegister. Tri-State-Koppelstufen des Schieberegisters werden hochohmig. 3 - Übernahme der Eingangsbelegung ins Eingaberegister. Dessen Tri-State-Koppelstufen werden aktiv. 4 - Übernahme der Eingangsbelegung ins Schieberegister. 5 - Deaktivieren der Tri-State-Koppelstufen des Eingangsregisters. 6 - der Schiebeweg wird wieder aktiviert. 7 - wir bringen erst die Ausgänge zur Wirkung (2) und holen die Eingänge später ab (3). Somit entspricht die Eingangsbelegung bereits der Reaktion auf die soeben eingestellte Ausgangsbelegung. Wir können sie also schon im folgenden Schiebeprozess abholen. Um diesen Effekt auszunutzen, muß ein hinreichend langer Abstand zwischen STROBE und LOAD vorgesehen werden.

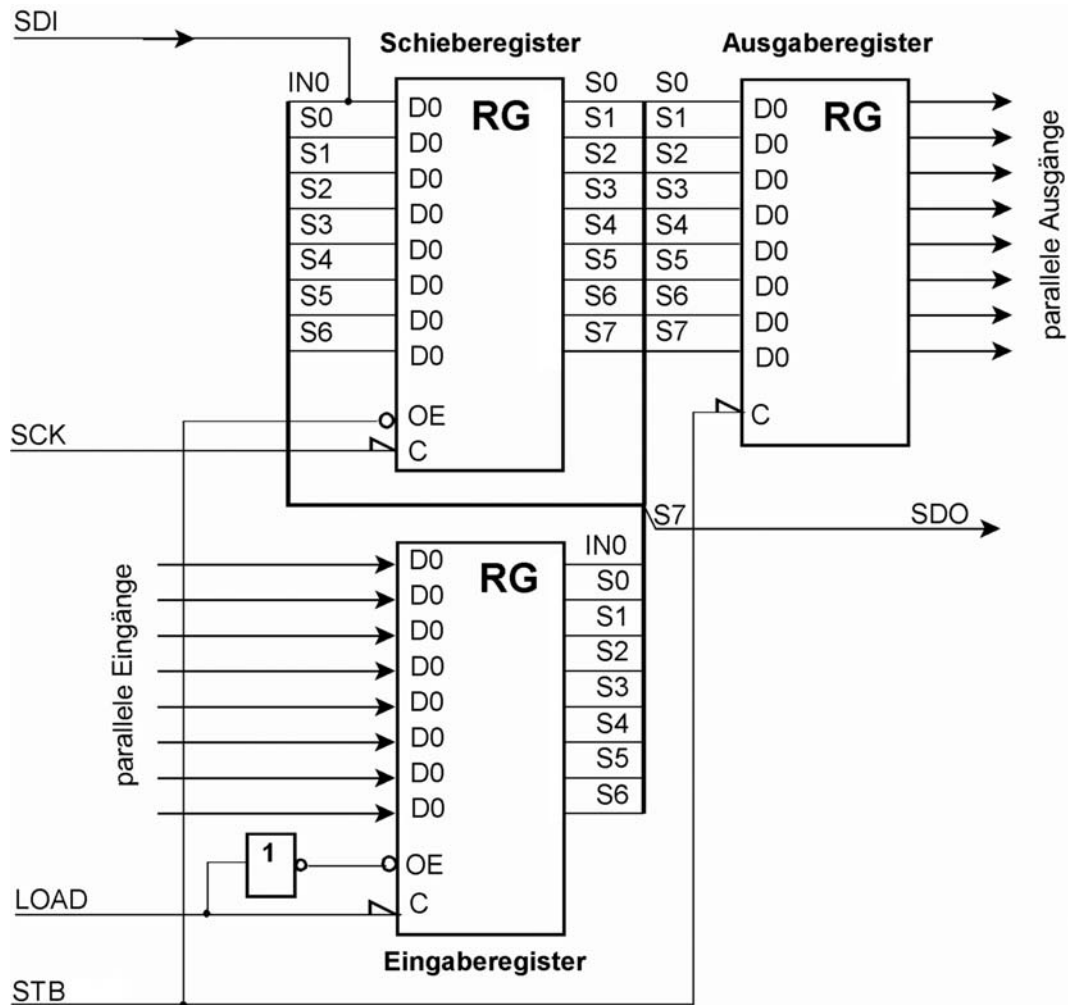


Abb.1.21 Schieberegisteranordnung mit drei universellen Registerschaltkreisen

Das Prinzip von Abb. 1.19 wurde hier mit drei D-Flipflop-Registern (z. B. 74x374) verwirklicht. Mit modernen universellen Bustreibern lassen sich bei gleicher Schaltkreiszahl mehr E-A-Anschlüsse (z. B. 18) vorsehen (vgl. Abb. 1.9). Im Vergleich zu „echten“ Schieberegistern brauchen wir zwar mehr Schaltkreise, es sind aber Typen aus der Massenfertigung, die es in nahezu allen Logikbaureihen gibt – auch in den ganz modernen. Somit lassen sich ggf. Probleme der Pegelwandlung, des Partial Power Down usw. durch Auswählen entsprechender Typen lösen. Hinweis: In der Schaltung von Abb. 1.21 muß der steuernde Mikrocontroller dafür sorgen, daß die Leitung SDI hochohmig geschaltet wird (vgl. die Positionen 2 und 6 in Abb. 1.19) – es sei denn, man nutzt das erste (an SDI angeschlossene) Flipflop des Schieberegisters nicht zu Eingabezwecken.

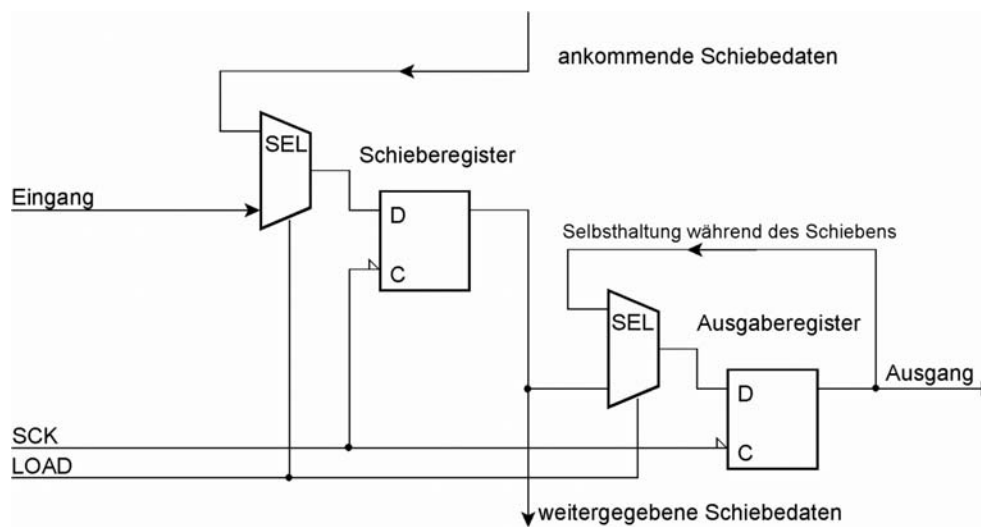


Abb.1.22 Eine Registerposition in einem CPLD

SCK ist das einzige Taktsignal. Bei $LOAD = Low$ ist der Schiebeweg aktiv, und die Belegung des Ausgaberegisters wird über die Rückführung gehalten. Ist $LOAD = high$, so bewirkt ein Impuls auf SCK die Übernahme der Schieberegisterbelegung ins Ausgaberegister und die Übernahme der Eingangsbelegung ins Schieberegister. Hier wird – wie bei Boundary Scan – die jeweils vorhergehende Eingangsbelegung übernommen, so daß, um die Reaktion auf die aktuelle Ausgabe abzuholen, ein weiterer Schiebeablauf erforderlich ist.

1.3 Interfaceanschlüsse trickreich ausnutzen

Oft geht es darum, mehrere Einrichtungen anzuschließen, dabei aber mit einer möglichst geringen Zahl an Interfacesignalen auszukommen. In den Applikationsschriften der Mikrocontroller-Hersteller, in einschlägigen Zeitschriften usw. finden wir hierzu eine Vielzahl von Anregungen. Eine naheliegende systematische Vorgehensweise:

- wir schließen zunächst die „dickste“ Einrichtung – jene mit den meisten Interfacesignalen – an,
- wir überlegen, wie wir diese Signale anderweitig ausnutzen können. Die Mehrfachnutzung beruht auf der Tatsache, daß es verschiedene Arten von Interfacesignalen gibt (Abb. 1.23).

In vielen Fällen können wir folgende Arten von Interfacesignalen erkennen:

- Datensignale,
- Adreß- und Auswahlsignale,
- echte Steuersignale. Das sind solche, die tatsächlich irgendwelche Vorgänge (Informationsübernahme, Aufschaltung usw.) auslösen (typische Bezeichnungen: Clock, Enable, Strobe, Write, Chip Select usw.).
- Rückmeldungen (Zustands- und Fehleranzeigen, Interruptsignale usw.).

Das Prinzip der Mehrfachnutzung: sind die echten Steuersignale inaktiv, so dürfen die anderen Signale beliebig schalten. Abb. 1.24 zeigt ein entsprechendes Beispiel.

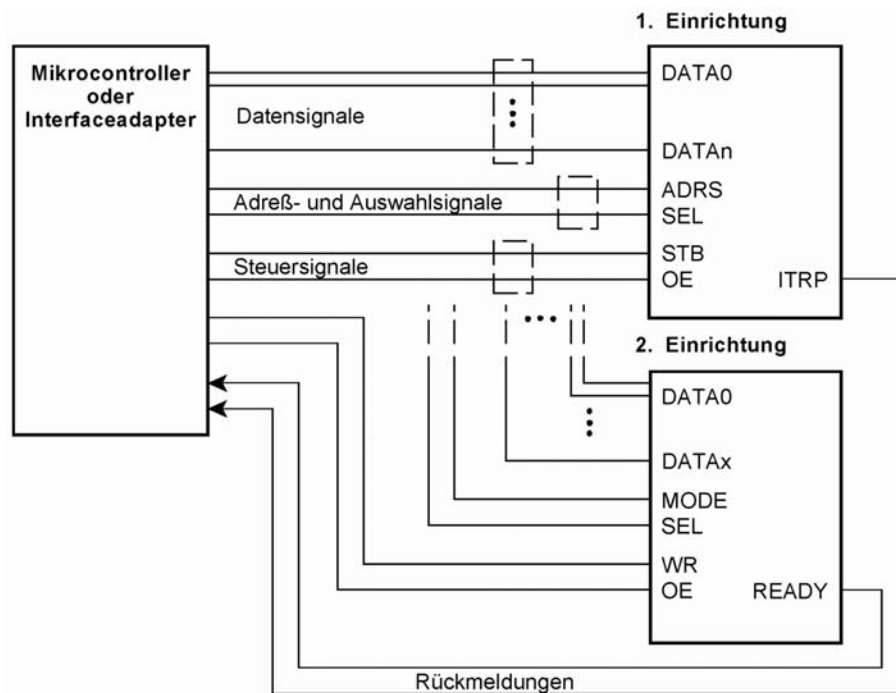


Abb.1.23 Verschiedene Arten von Interfacesignalen

Praxistips:

1. Datenwege gemeinsam nutzen. Eine Art Privatbus aufbauen.
2. Adreß- und Auswahlanschlüsse mehrfach nutzen.
3. Nur die echten Steuersignale müssen für jede Einrichtung gesondert vorgesehen werden.

Rückmeldungen

Im Prinzip ist ein Zusammenfassen über Auswahlanschlüssen (Multiplexer) oder eine Art Bussystem möglich. Es kommt aber darauf an, wie sich die Signale verhalten und wie wir sie auswerten (direkte Wirkung auf die Hardware¹⁾, Auslösung von Interrupts, programmseitige Abfrage).

Praxistips:

1. Soll der Mikrocontroller oder Interfaceadapter mit mehreren Einrichtungen gleichzeitig arbeiten, müssen jene Signale, die direkt auf die steuernde Hardware einwirken (Wartezustände, Handshaking, Interruptauslösung usw.), einzeln angeschlossen werden.
2. Arbeitet der Mikrocontroller oder Interfaceadapter zu einer Zeit nur mit einer einzigen Einrichtung, wäre eine Zusammenfassung möglich.
3. Bei rein programmseitiger Abfrage (Polling) ist die Zusammenfassung stets möglich.
4. Das Zusammenfassen lohnt sich nur dann, wenn hierdurch tatsächlich weniger Anschlüsse benötigt werden (an die erforderlichen Auswahlanschlüsse denken...)

1) z. B. zum Hervorrufen von Wartezuständen oder in Handshaking-Signalspielen.

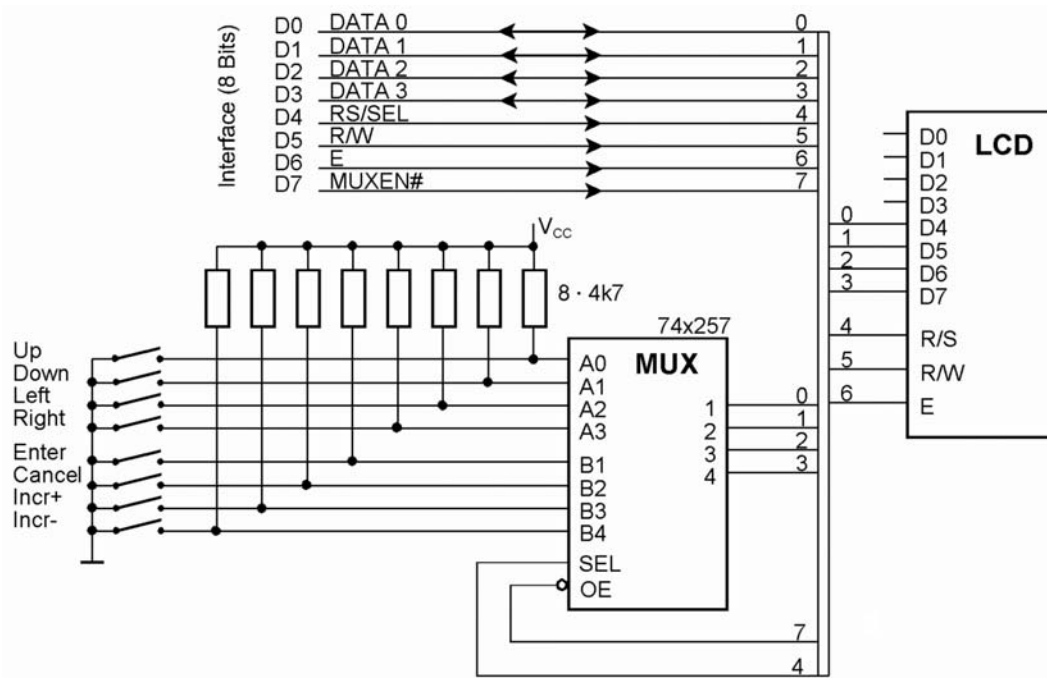


Abb.1.24 Praxisbeispiel: eine Bedien- und Anzeigetafel

Die Bedien- und Anzeigetafel enthält sechs Tasten, einen Incrementalgeber und eine LCD-Punktmatrixanzeige. Diese Einrichtungen werden über einen einzigen universellen 8-Bit-Port angeschlossen. Vier Bits (D3...0) bilden einen bidirektionalen Datenbus. Bit D6 treibt das Steuersignal (E) der LCD-Anzeige, Bit D7(MUXEN#) aktiviert den Multiplexer. Ausgabe auf die LCD-Anzeige: Bit D7 auf High. Bit D6 wird bei jedem Zugriff aktiviert. Abfrage der Bedienelemente: Bit D6 bleibt inaktiv (Low). Bit 7 auf Low. Der Multiplexer schaltet jeweils vier Kontaktbelegungen auf den Datenbus auf. Auswahl über Bit D4. Ist die LCD-Anzeige inaktiv, so dürfen die Bits D4 und D5 beliebig schalten. Nutzt man beide Bits aus, können insgesamt $4 \cdot 4 = 16$ Kontakte oder andere Eingangssignale abgefragt werden.

Elektrische Kopplungsprobleme

Die im folgenden kurz beschriebenen Probleme ergeben sich, weil es nicht um Verbindungen innerhalb einer einzigen Schaltungsanordnung handelt, sondern um Verbindungen zwischen verschiedenen Einrichtungen oder zwischen elektronischer Einrichtung und Außenwelt. Die Vernachlässigung dieser Probleme äußert sich zumeist in überdurchschnittlich vielen Funktionsfehlern, die oft den Charakter von immer wieder auftretenden flüchtigen Fehlern (Aussetzfehlern) haben.

Infolge der Verbindung mit der Außenwelt ist Schnittstellen-Hardware in besonderem Maße fehleranfällig. Eine Unachtsamkeit – z. B. das Stecken eines fehlerhaften Kabels – kann ausreichen, die Hardware zu entschärfen. Wichtige Gefahrenpunkte im Überblick:

- elektrostatische Entladung (ESD) z. B. beim Berühren von Kontaktstiften. Moderne Hardware sollte so etwas wegstecken (EMV-Vorschriften). Trotzdem Vorsicht. Zudem: die Gesetzgebung betrifft nur komplette Geräte und damit nur die ESD-Festigkeit der außen zugänglichen Anschlüsse. Für Busanschlüsse von Steckkarten und für interne Interfaces gilt sie nicht!
- Stecken bei anliegender Speisespannung (Hot Plugging). Serielle Schnittstellen müssen das aushalten, desgleichen alle Interfaces, die nach neueren Standards ausgelegt sind.
- Kurzschlüsse. Kritisch sind vor allem Kurzschlüsse zwischen Speisespannung und Masse. Hiervon können alle Interfaces mit herausgeführter Speisespannung betroffen sein*).
- Kurzschlüsse zwischen Signalleitungen. Die meisten Interfaces dürften diese überleben (bitte nicht allzu sehr darauf verlassen...). *Serielle Schnittstellen* (RS-232) sind grundsätzlich kurzschlußfest.

*) : moderne Interfacestandards schreiben typischerweise den Einsatz selbststrückstellender Sicherungen vor.

Hot Plugging, Hot Docking, Hot Swapping

Die Begriffe werden häufig verwendet, aber nicht immer einheitlich. Es ist wichtig, zwei Merkmale genau zu unterscheiden:

- die Schnittstelle hält das Stecken und Trennen bei anliegender Betriebsspannung aus, geht also – als Hardware – davon nicht kaputt. Das System wird aber typischerweise abstürzen (beim Trennen) oder ein neu angestecktes Gerät nicht erkennen.
- das Stecken und Trennen bei laufendem Betrieb wird auch vom System unterstützt; ein abgetrenntes Gerät wird aus dem System gleichsam verabschiedet, ein neu angestecktes in das System aufgenommen (weitere Fachbegriffe mit gleicher Bedeutung: Hot Insertion, Dynamic Attach/Deattach o. dergl.).

Hinweise:

1. Fehlt die Hot-Docking-Unterstützung, so wird die Plug&Play-Software nicht automatisch durchlaufen. Somit wird das Hinzufügen oder Abtrennen von Geräten nicht erkannt. Solche Systeme sind, wenn an den betreffenden Schnittstellen herumgestöpselt wird, stets herunterzufahren und neu zu starten.
2. Im PC-Bereich werden manche Schnittstellen traditionell unterstützt – also auch von älteren Systemversionen (z. B. das PC-Card-Interface). Oft funktioniert das Hot Docking aber erst mit zeitgemäßen Betriebssystemen wirklich befriedigend (Beispiel: die Hot-Docking-Unterstützung in Windows ab 2000/XP).
3. Hot Docking in Eigenentwicklungen: von Grund auf kaum durchführbar (Entwicklungsaufwand, Kosten). Behelf: die eigene Lösung auf entsprechende Angebote der einschlägigen Industrie stützen, z. B. auf USB-Interfacewandler oder auf PC-Karten.

4. Viele Embedded Systems sind gleichsam abgeschlossene Anwendungslösungen, in denen niemand wahllos herumstöpselt. Hier ist Hot Docking nicht erforderlich.

Pegel / Signalhub

Signalhub zu niedrig: Transistorschaltstufe, Komparator

Signalhub zu hoch: Spannungsteiler, Begrenzer- oder Klammerschaltung, Komparator

Unterschiedliche Bezugspegel (Massepotentiale): Spannungsteiler, Komparator, rechnende Operationsverstärkerschaltungen, Differenzmeßverstärker (Instrumentation Amplifier), Optokoppler, transformatorische oder kapazitive Kopplung.

Gar kein gemeinsamer Bezugspegel: Wechsellspannungskopplung (transformatorisch oder kapazitiv), Gleichstromwiederherstellung, Optokoppler

Flanken / Anstiegszeiten

Flanken zu steil: Flankenverschleifung (Kondensatoren, RC-Glieder)

Flanken nicht steil genug: Schmitt-Trigger, Komparator

Zeitlicher Verlauf / Impulsbreiten

Kurze (Stör-) Impulse ausblenden: digitale Signalbewertung, Tiefpaßfilter

Kurze Impulse erkennen und auswerten: Impulsdehnung (z. B. mit monostabilem Multivibrator), Fangschaltungen.

Ströme / Treibvermögen

Eingangsströme zu gering: Verstärker

Hohe Ausgangsströme erforderlich: Treiberstufen, Leistungsstufen, Relais.

Zur Treibfähigkeit der E-A-Ports:

- TTL-Baureihe, herkömmliche Controller usw.: Treibvermögen nach Low viel größer als nach High (Standard TTL 16 mA vs. 0,4 mA, Atmel AVR: 20 mA vs. 3 mA).
- neuere Logikbaureihen (z. B. AC / ACT) und Controller: Treibvermögen nach beiden Seiten gleich groß (z. B. 25 mA (AC /ACT) oder 20 mA (Atmel MegaAT)).

Achtung: Gesamtstrom nicht größer als maximaler Strom durch die Masse- und Speisespannungsanschlüsse.

Offene Eingänge

(z. B. steckbare Funktionseinheiten nicht angeschlossen, Drähte nicht angeklemt)

So etwas darf eigentlich nicht vorkommen – jeder Schaltkreiseingang sollte stets mit einem definierten Signalpegel belegt sein (das betrifft vor allem CMOS-Schaltkreise). Wann kann es zu offenen Eingängen kommen?

- wenn der Eingang nicht genutzt wird,
- wenn an ein Interface nichts angeschlossen ist (Kabel abgezogen o. dergl.),
- wenn es sich um Busleitungen handelt und wenn der Bus gerade von keiner der angeschlossenen Einrichtungen getrieben wird.

Abhilfe:

- ungenutzte Eingänge: fest beschalten, so daß sich definierte Pegel ergeben.
- Kabelanschlüsse: Widerstandsbeschaltung (Pull-up, Pull-down, Spannungsteiler).
- Busleitungen: Widerstandsbeschaltung oder Bus parken oder Bushalteschaltkreise vorsehen (halten die letzte Belegung auf dem Bus) .

Hinweise:

1. Pegel ggf. so festlegen, daß der betreffenden Eingang nicht aktiviert wird (z. B. keine Interruptauslösung).
2. Pegel so festlegen, daß der Betriebszustand als Fehler erkannt wird.
3. Manche Schaltkreise (Mikrocontroller, CPLDs usw.) haben an ihren Anschlüssen eigens durch Programmierung schaltbare Pull-up-Widerstände). Ausnutzen!

Kurzschlüsse

Kurzschlüsse zwischen Speisespannung und Masse: Strombegrenzung, Sicherungselemente (z. B.s selbstrückstellende Halbleitersicherungen), Erkennung des Betriebsspannungsabfalls (Brownout), softwareseitiger Wiederanlauf (Recovery).

Kurzschlüsse zwischen Signalleitungen: Strombegrenzung, kurzschlußfeste Treiberstufen (z. B. Open Collector statt Gegentakt), diagnostische Rückführungen, um diesen Betriebsfall als Fehler zu erkennen

Schutz gegen elektrostatische Entladung (ESD)

Daß die Hardware elektrostatische Entladungen unbeschadet überlebt, ist eine Grundforderung der EMV-Gesetzgebung. Signalwege, die über Steckverbinder mit berührbaren Kontakten verlaufen, sind besonders gefährdet. Abhilfe: Schutzbeschaltung, Berührungsschutz.

Schutzbeschaltung:

- mit Metalloxid-Varistoren (MOVs),
- Suppressordioden (Transient Voltage Suppressors, TSVs) oder
- Suppressorbauelemente auf Polymerbasis.

Jede Signalleitung wird mit einem solchen Bauelement beschaltet.

Varistoren sind kostengünstig und höher belastbar, schalten aber langsamer. Sie stellen eine vergleichsweise große kapazitive Last dar (von einigen hundert pf bis zu mehreren nF). TSVs schalten in kürzester Zeit (Größenordnung: einige ps... ca. 5 ns). Sie können hohe impulsförmige Ströme aushalten (Größenordnung: < 10... > 100 A). Die Dauerbelastbarkeit ist aber deutlich geringer (einige zehn mA...wenige A). Ist ein Schutz gegen länger dauernde Überlastung gewünscht, müssen Bauelemente vorgeschaltet werden, die den Stromweg trennen (Schmelzsicherung, PolySwitch-Sicherung oder ähnliches). Suppressorbauelemente auf Polymerbasis haben eine sehr geringe Kapazität (unter 1 pF), vertragen aber nur die typischen ESD-Entladungen und keine anderweitig eingekoppelten Störungen.

Schaltkreise ohne Zusatzbeschaltung

Man kann typischerweise mit einer ESD-Festigkeit von 2000 V rechnen. Aber schon beim Gang über einen Teppich können sich bis zu 15 000 V aufbauen...

Es gibt Schnittstellenschaltkreise mit garantierter ESD-Festigkeit. Typische Werte: $\pm 2 \text{ kV}$, $\pm 5 \text{ kV}$, $\pm 10 \text{ kV}$, $\pm 15 \text{ kV}$ (jeweils gemäß dem sog. Human Body Model (HBM) der ESD-Entladung).

In der Massenfabrikation ist man bestrebt, die Anforderungen der EMV-Gesetzgebung mit geringstem Aufwand zu erfüllen (also möglichst ohne Zusatzbeschaltung). Man bewegt sich hierbei oft scharf an der Grenze (Motto: nicht mehr tun, als unbedingt nötig...). Wer wirkliche Qualität anbieten will, sollte aber nicht um jeden Preis sparen (Abb. 1.25).

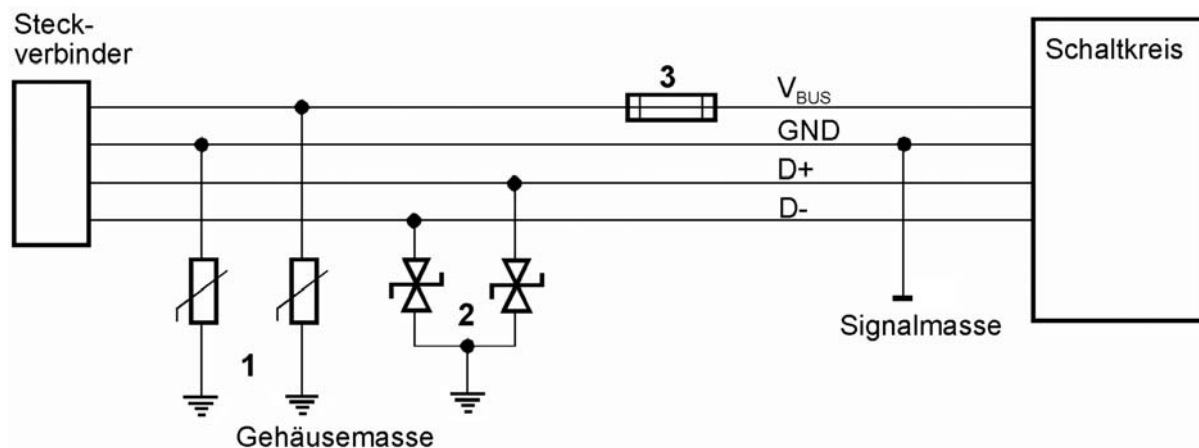


Abb. 1.25 Eine typische Schutzbeschaltung am Beispiel eines USB-Anschlusses (nach Littlefuse)

1 - Schutz der Stromversorgungsleitungen durch Mehrschichtvaristoren. Die vergleichsweise hohe Kapazität ist hier sogar erwünscht (Glättungswirkung). 2 - Schutz der Datenleitungen durch bidirektionale Suppressordioden (USB 2: Suppressorbauelemente auf Polymerbasis – bei der hohen Datenrate kommt es auf geringste kapazitive Belastung an). 3 - selbstrückstellende Halbleitersicherung im Stromweg (Vorkehrung gegen Kurzschluß).

EMV

Filterschaltungen, Abschirmung

ESD + EMV: Überwachung des Betriebszustandes (z. B. Watchdog) und softwareseitiger Wiederanlauf (Recovery).

Betriebsfall eingeschaltet / ausgeschaltet (Partial Power Down)

Wir betrachten miteinander verbundene Einrichtungen (PC und Interfaceadapter, Interfaceadapter und Anwendungshardware). Werden alle Einrichtungen aus der gleichen Spannungsquelle gespeist (gemeinsame Batterie oder gemeinsames Netzteil), gibt es das hier in Rede stehende Problem nicht. Ansonsten ist mit einem besonderen Betriebsfall zu rechnen: nämlich damit, daß die eine von zwei miteinander verbundene Einrichtungen eingeschaltet und die andere ausgeschaltet ist (im Fach-Englisch: Partial Power Down). In einem solchen Fall muß zweierlei gewährleistet sein: (1) die ausgeschaltete Hardware darf den Betrieb der eingeschalteten nicht beeinträchtigen, (2) die ausgeschaltete Hardware darf nicht durch Erregen seitens der eingeschalteten Schaden nehmen. Das Problem ist gelöst, wenn die ausgeschaltete Hardware gegenüber der eingeschalteten eine hohe Impedanz darstellt.

Um festzustellen, ob ein Anschluß eines ausgeschalteten Schaltkreises hoch- oder niederohmig ist, kann man sich die jeweilige Innenschaltung ansehen und annehmen, daß bei ausgeschalteter Betriebsspannung ($V_{CC}=0$) ein Kurzschluß zwischen Masse (GND) und Betriebsspannung (V_{CC}) besteht. Auf dieser Grundlage ist dann zu untersuchen, welche Stromwege sich bilden können, wenn an den

Schaltkreisanschluß eine Spannung gelegt wird. Wesentlichen Einfluß hierauf haben die Diodenstrukturen an den Ein- und Ausgängen der Schaltkreise (Abb. 1.26, 1.27 und Tabelle 1.6).

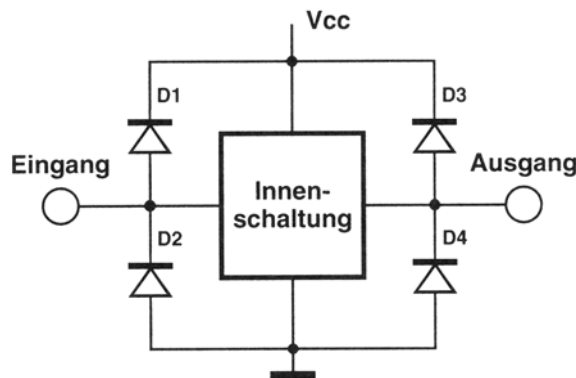


Abb.1.26 Diodenstrukturen an den Ein- und Ausgängen von Schaltkreisen (nach Texas Instruments)

Diode	Erklärung	vorhanden in...	nicht in...
D1	ESD-Schutz (Ableitung positiver Spannungsspitzen)	CMOS	TTL und BiMOS
D2	parasitäre Struktur; bipolare Schaltkreise haben parallel dazu eine absichtlich eingebaute Diode, um negative Spitzen (Unterschwinger) zu kappen	CMOS, TTL, BiMOS	
D3	parasitäre Struktur*); manche CMOS-Typen haben parallel dazu eine absichtlich eingebaute Diode als ESD-Schutz	bipolaren Gegentaktausgängen und CMOS	bipolaren Open-Collector- und Tri-State-Ausgängen
D4	parasitäre Struktur; manche Schaltkreise haben parallel dazu eine absichtlich eingebaute Diode als ESD-Schutz oder zum Kappen negativer Spitzen	CMOS, TTL, BiMOS	

*) in CMOS-Schaltungen bildet der "obere" p-Kanal-FET der Ausgangsstufe eine parasitäre Diode, die in Flußrichtung geschaltet ist, sobald die Ausgangsspannung größer ist als V_{CC}

Tabelle 1.6 Erläuterungen zu den Diodenstrukturen in Abb. 1.3

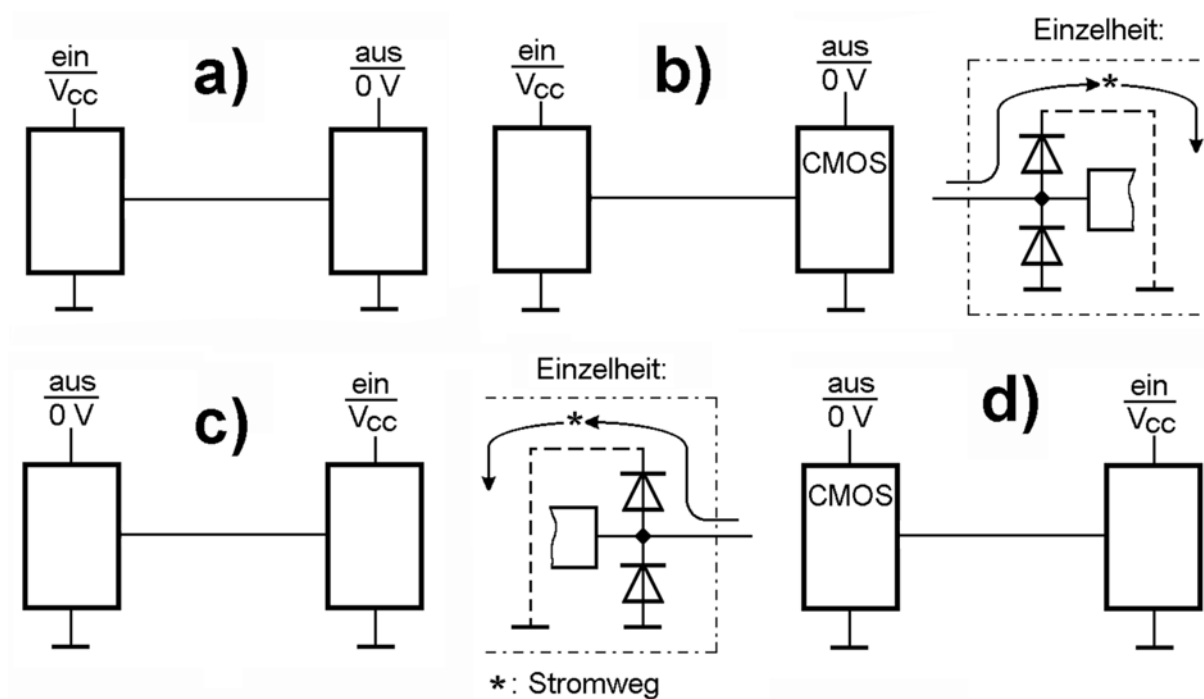


Abb. 1.27 Betriebsfälle in teilweise ausgeschalteter Hardware (Partial Power Down)

- ein aktiver Ausgang wirkt auf einen inaktiven bipolaren Eingang. Unproblematisch, da bipolare Eingangsstufen bei ausgeschalteter Speisespannung hinreichend hochohmig sind.
- ein aktiver Ausgang wirkt auf einen inaktiven CMOS-Eingang. Ein inaktiver CMOS-Eingang ist -- infolge der Dioden D1, D2 (vgl. Abb. 1.3) – grundsätzlich niederohmig. Es ist an sich der ungünstigste Betriebsfall: die gegen V_{CC} geschaltete Diode D1 am Eingang liegt bei $V_{CC} = 0$ in Flußrichtung, und es gibt nichts, was den Strom begrenzt. Der inaktive Schaltkreis kann somit zerstört werden.
- ein inaktiver bipolarer Ausgang wirkt auf einen aktiven Eingang. Bipolare Tri-State-Ausgänge (z. B. jene von Bustreibern) sind im ausgeschalteten Zustand hochohmig, normale bipolare Ausgangsstufen hingegen niederohmig.
- ein inaktiver CMOS-Ausgang wirkt auf einen aktiven Eingang. Ein inaktiver CMOS-Ausgang ist – infolge der Dioden D3, D4 (vgl. Abb. 1.3) – niederohmig. CMOS-Stufen mit nach V_{CC} geschalteten Dioden können also nicht an Schnittstellen verwendet werden, die Hochohmigkeit erfordern.

Schaltungslösungen:

Gemeinsame Spannungsversorgung. Damit werden Betriebsfälle gemäß Abb. 1.26 sicher vermieden. Die Lösung ergibt sich bei entsprechend kompakter Auslegung des gesamten Systems (alles in einem Gehäuse, alles aus einem gemeinsamen Netzteil gespeist).

Bestückung der Schnittstellen mit geeigneten Schaltkreisen. Diese müssen die kritische Betriebsfälle ausschließen oder aushalten. Beispiele: bipolare und BiMOS-Digitalschaltkreise, CMOS-Schaltkreise mit entsprechenden Vorkehrungen (LVT, ALVT, LVC, AVC usw.), bipolare Comparatoren, fehlergeschützte (failure protected) Analogschalter und -multiplexer.

Isolation durch FET-Schalterbauelemente. Solche Schaltkreise gewährleisten die erforderliche Isolation

zwischen ein- und ausgeschalteter Hardware (Abb. 1.28).

Geeignete Mikrocontroller und CPLDs. Es gibt Typen, die ausdrücklich so entworfen wurden, daß sie den Betriebsfall aushalten (z. B. durch eingebaute FET-Schalter ähnlich Abb. 1.28).

Zusatzbeschaltung mit Widerständen und Dioden. Solche Lösungen aus der Frühzeit der integrierten Digitalschaltkreise sind auch heutzutage noch zweckmäßig -- vor allem dann, wenn es um nur wenige Anschlüsse geht, die vor dem Betriebsfall Partial Power Down zu schützen sind (Abb. 1.29).

Hinweise:

1. Kann die gemeinsame Speisung aller Funktionseinheiten gewährleistet werden, tritt das Problem nicht auf.
2. Nutzt man die externen Schnittstellen der PCs und setzt man auf der PC-Seite des Interfaceadapters jeweils standardkonforme Schaltkreise ein, löst sich das Problem dort von selbst (Schaltkreise sind für Partial Power Down ausgelegt).
3. Auf der Anwendungsseite tritt das Problem nicht auf, wenn Anwendungshardware und Interfaceadapter eine gemeinsame Stromversorgung haben.

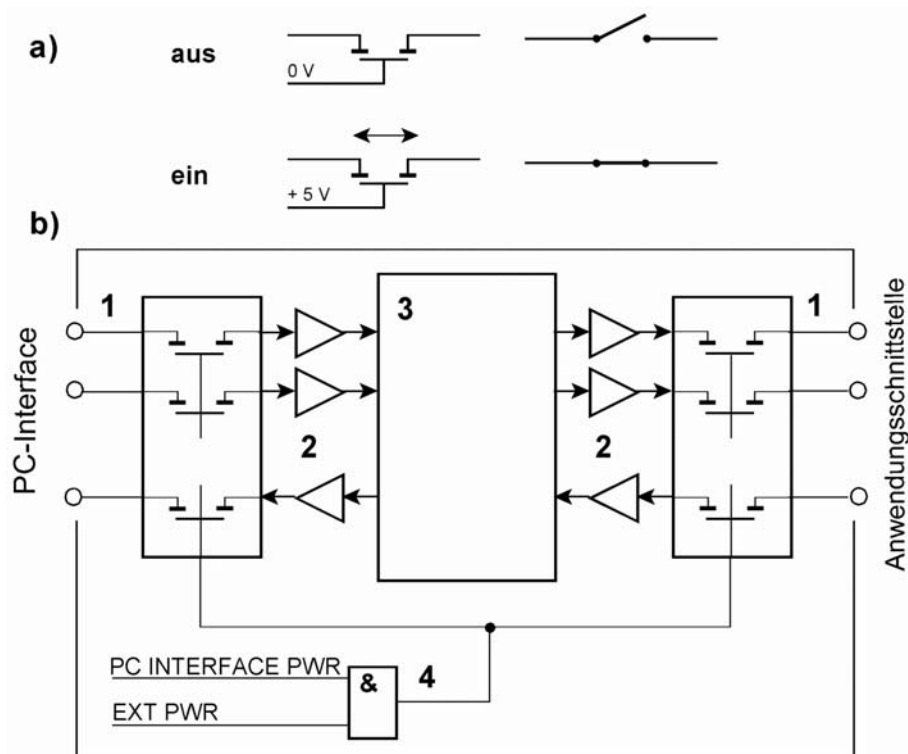


Abb.1.28 Isolation durch FET-Schalterbauelemente

- a) Prinzip des FET-Schalters. Ist das Gate nicht geladen (0 V), so ist der FET gesperrt (offener Schalter). Liegt eine positive Spannung am Gate, so ist der FET leitend (geschlossener Schalter; Durchgangswiderstand typisch 5Ω , Verzögerungszeit typisch $< 250 \text{ ps}$). Der Schalter wirkt in beiden Richtungen (bidirektional).
- b) Aufbau eines Interfaceadapters. 1 - FET-Schalterbauelemente; 2 - Treiber- und Empfängerstufen; 3 - die interne Logik; 4 - Steuerung der Schalterbauelemente 1. Hier werden die Schalter

geschlossen, wenn beide Versorgungsspannungen (PC- und Anwendungsschnittstelle) anliegen.

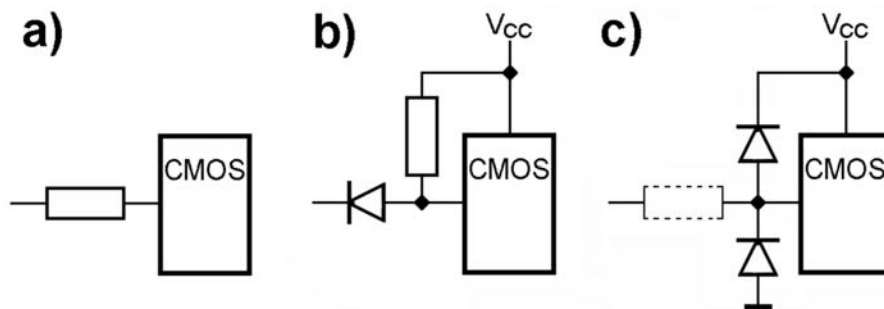


Abb. 1.29 Zusatzbeschaltung mit Widerständen und Dioden

- Strombegrenzung durch Serienwiderstand. Der Widerstand ist so zu dimensionieren, daß bei größter Spannungsdifferenz nur ein unschädlicher Strom (Richtwert: wenige mA) fließen kann.
- Verhindern unerwünschter Stromflüsse durch Einfügen von Sperrdioden. Bei ausgeschalteter Speisespannung liegt die Diode in Sperrichtung, so daß praktisch kein Strom fließen kann. Es ist allerdings ein Pull-up-Widerstand nötig, um den High-Pegel am Eingang zu halten. Somit steigt der Strom bei Low. Zudem wird der Low-Pegel infolge der Flußspannung der Diode angehoben (Schottky-Dioden einsetzen).
- Externe Schutzdioden. Die Schottky-Dioden kappen zu hohe positive oder negative Spannungen auf einen Wert $V_{CC} + 0,5 \text{ V}$ bzw. Massepotential $- 0,5 \text{ V}$. Die Schaltkreise werden damit geschützt; die Kurzschlußströme fließen nun aber durch die externen Dioden, so daß erforderlichenfalls für eine Strombegrenzung zu sorgen ist. Diodenanordnungen gemäß Abb. 1.29c sind marktgängig. Sie halten relativ hohe Ströme aus (Richtwert: 50 mA), so daß Strombegrenzungswiderstände niederohmiger dimensioniert werden können.

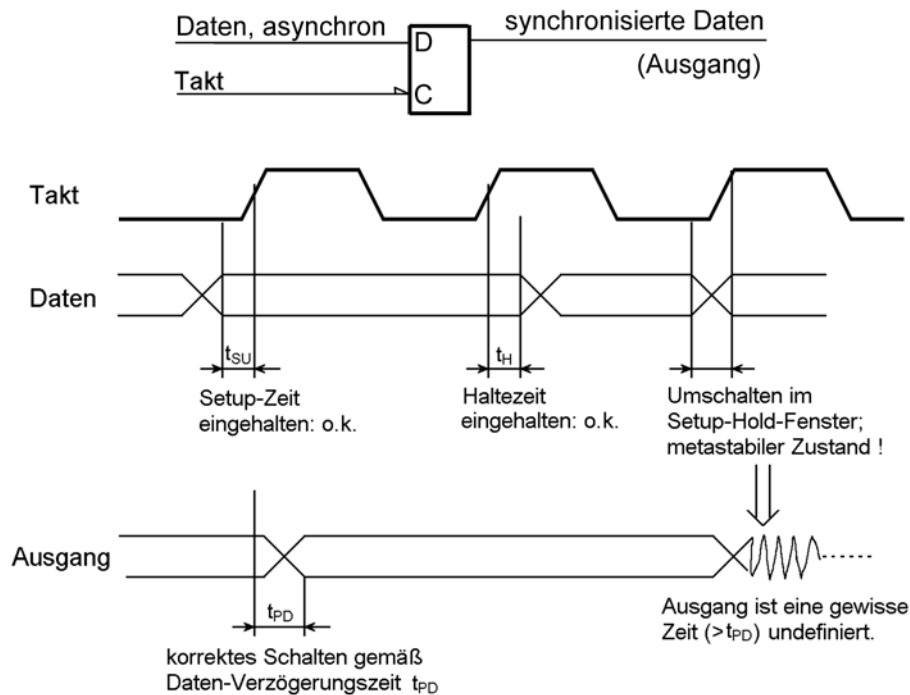
Metastabilität

Moderne Computer sind taktgesteuerte (synchrone) Schaltwerke. Schnittstellen zur Außenwelt hingegen sind naturgemäß immer asynchron (Schalter, Tasten, Sensoren, Interfaces anderer Funktionseinheiten usw. können nun einmal nichts vom Takt im Computer wissen). An diesen Schnittstellen ist also stets das Problem der *Synchronisation* (Eintaktierung) zu lösen. Die naheliegende Lösung: das asynchrone Signal an den Dateneingang eines D-Flipflops legen, der mit dem jeweils passenden Takt angesteuert wird.

Wenn eine Taktflanke mit einer Schaltflanke des asynchronen Signals zusammentrifft, kann es zu sog. metastabilen Zuständen kommen. Kennzeichnend für einen metastabilen Zustand ist, daß das Flipflop Ausgangssignale abgibt, die nicht eindeutig einem der beiden Logikpegel entsprechen (Abb. 1.30). Die Wahrscheinlichkeit hierfür hängt von der Schalthäufigkeit des asynchronen Signals und von der Taktfrequenz ab. Mit Ereignissen, die zu metastabilen Zuständen führen können, ist typischerweise in Abständen von Sekunden...Minuten zu rechnen.

Metastabile Zustände sind grundsätzlich unvermeidbar. Was man lediglich tun kann: von der übernehmenden Taktflanke an eine gewisse Zeit warten – in der Hoffnung, daß dann ein metastabiler Zustand, der sich womöglich eingestellt hat, wieder abgeklungen sein wird. Der Abklingvorgang verläuft gemäß einer Exponentialfunktion. Demgemäß nimmt mit wachsender Abklingzeit die Wahrscheinlichkeit dafür, daß sich das Flipflop immer noch im metastabilen Zustand befindet, exponentiell ab.

Eine einfache Synchronisationsschaltung:



Beispiele von Signalverläufen:



1 - korrekt; 2, 3 - metastabile Zustände

Abb.1.30 Das metastabile Verhalten eines Flipflops

Gegenmaßnahmen:

- Flipflops auswählen, die von Natur aus nur selten in metastabile Zustände übergehen.
- Ausgänge solcher Synchronisations-Flipflops dürfen nach der schaltenden Taktflanke erst dann ausgewertet werden, wenn mögliche metastabile Zustände abgeklungen sind. Je länger die Wartezeit, um so höher die Funktionssicherheit.
- durch Hintereinanderschalten von zwei Flipflops wird die Wartezeit gleichsam automatisch gewährleistet (Doppelsynchronisation; Abb. 1.31). Das Signal wird dadurch um eine weitere Taktperiode verzögert.

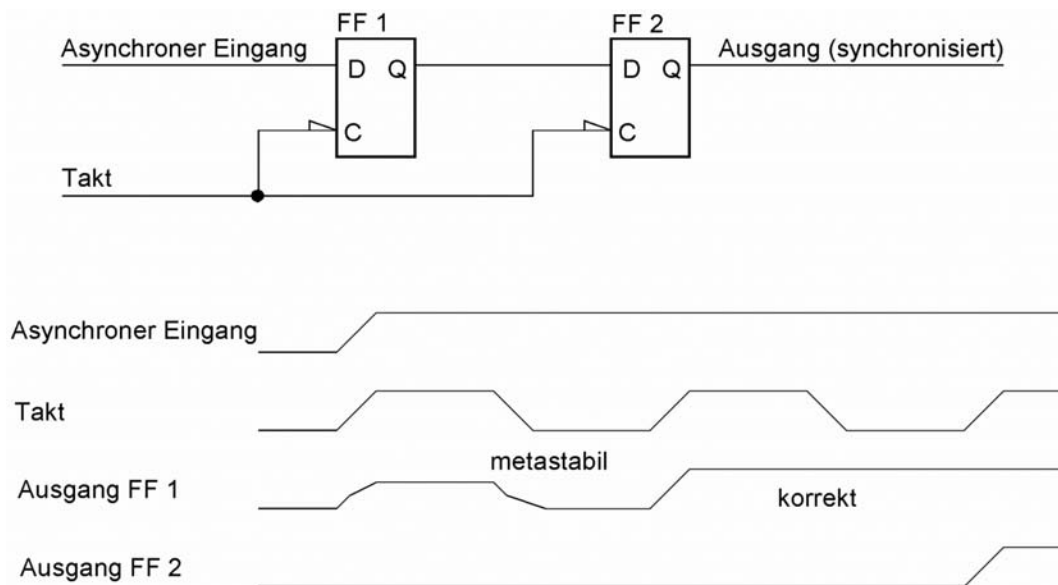


Abb. 1.31 Doppelsynchronisation (nach Texas Instruments)

Welche Vorkehrungen gibt es?

- in Off-the-Shelf-Logik (LSI, MSI): nichts; der Entwickler muß aufpassen,
- in CPLDs und FPGAs: desgleichen,
- in Mikrocontrollern: die E-A-Ports enthalten Synchronisationsvorkehrungen (Abb. 1.32). Anschließen asynchroner Signale unbedenklich möglich.
- in Hochleistungsprozessoren: es gibt nur wenige Eingänge, die mit asynchronen Signalen beschaltet werden dürfen (z. B. Reset, Interrupt, NMI). Ansonsten sind die Setup- und Haltezeitvorgaben einzuhalten.
- im PC: externe Interfaces müssen Synchronisationsvorkehrungen enthalten, interne (ISA, PCI, IDE/ATA usw.) hingegen nicht. Die Synchronisation mit den Takt- oder Gültigkeitssignalen des jeweiligen Interfaces ist Sache der angeschlossenen Hardware.

Metastabile Zustände an typischen Schnittstellen:

- alle Schnittstellen mit serieller Datenübertragung (RS-232, USB, Ethernet usw.): kein Problem. Wird in den Steuerschaltkreisen erledigt.
- alle Schnittstellen mit paralleler Übertragung (Bussysteme im PC, IDE/ATA, parallele Schnittstelle, IEEE-488 usw.): aufpassen. Die Setup- und Haltezeitvorgaben der Datenbusbelegung in Bezug auf die jeweiligen Takt-, Strobe- oder Handshake-Signale sind einzuhalten. Einzugebende Daten sind hierzu ggf. in Halteregeistern zwischenzupuffern. Hierzu D-Flipflops nehmen, keine Latches.

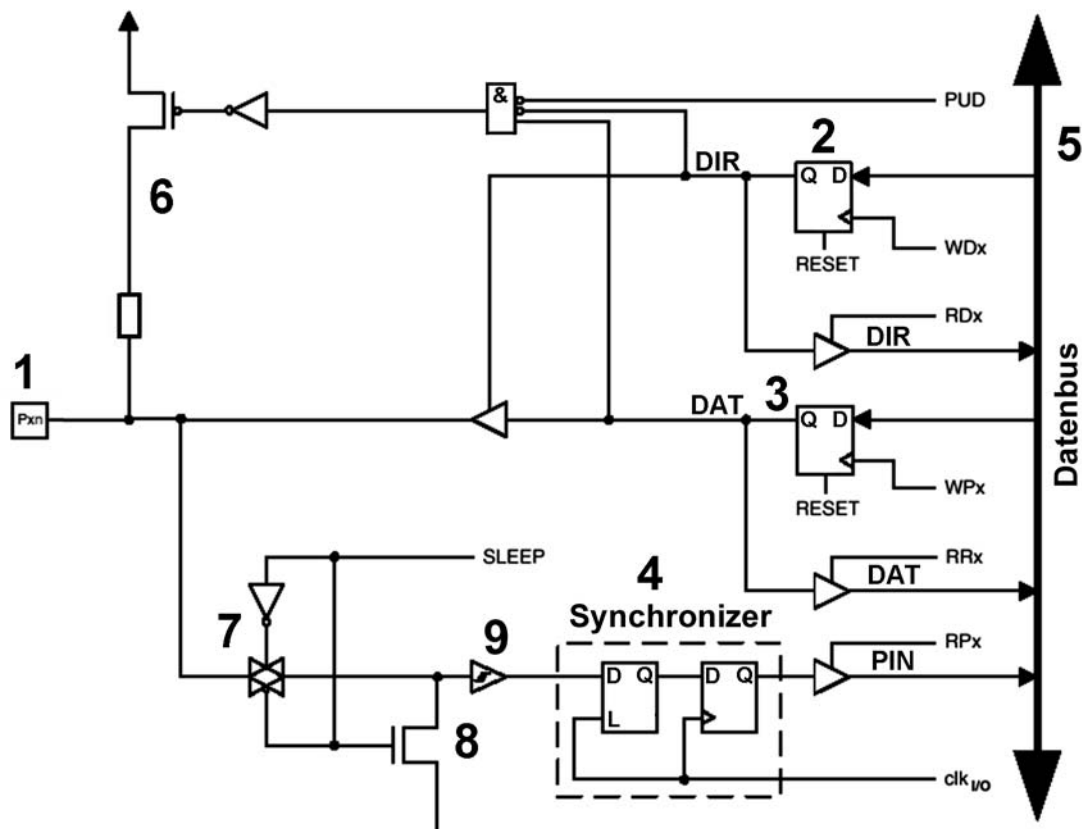


Abb. 1.32 Mikrocontroller-Port. Eine einzelne Bitposition (nach Atmel)

Die Abbildung veranschaulicht mehrere der zuvor erläuterten Schaltungsvorkehrungen. 1- E-A-Anschluß (Pin); 2 - Richtungsregister; 3 - Datenregister; 4 - Synchronisationsstufe^{*)} mit Latch und D-Flipflop; 5 - der interne Datenbus; 6 - schaltbarer Pull-up-Widerstand. Kann aktiviert werden, wenn die Bitposition als Eingang geschaltet, aber nicht belegt ist (offener Eingang). 7- Schalterelement; 8 - Pull-down-Transistor; 9 - Eingangspuffer. In Schlafzuständen (SLEEP) wird der Anschluß von der Innenschaltung abgetrennt (7) und der Eingangspuffer 9 mit dem Festwert Low belegt (8).

*) : in vielen Datenblättern ist diese Stufe nicht dargestellt.

Nicht vergessen: das Zurücksetzen

Nach dem Einschalten muß die gesamte Elektronik in einen definierten Anfangszustand gelangen -- sonst ergeben sich womöglich sehr unschöne Nebenwirkungen. Beispiel: wir haben ein Register, daran angeschlossen einen Treibertransistor mit Relais. Dieses steuert einen Elektromotor. Wenn nichts getan wird, steht die Chance 50:50, daß das betreffende Flipflop nach dem Einschalten in die Stellung „1“ kippt (und so das Relais zum Anziehen bringt...).

Wenn das Initialisieren zu lange dauert

Einfache Mikrocontroller-Anwendungen verzeihen manche Sünden in Hinsicht auf eine saubere Initialisierung, weil nach dem Einschalten sofort das Anwendungsprogramm gestartet wird, die Unsicherheit also schlimmstenfalls nur wenige Millisekunden dauert. Das Hochfahren der PCs zieht sich aber hin (Sekunden...Minuten). Das gilt sinngemäß für andere Systeme, die auf eine komplexe Betriebssoftware angewiesen sind (Linux ist übrigens um keinen Deut besser...). Der Anfangszustand muß somit auf direktem Wege erzwungen werden (also ohne auf Software angewiesen zu sein). Typische Maßnahmen:

- eigene Rücksetzsaltungen im Interfaceadapter (Abb. 1.33),
- Einsatz von Flipflops und Registern mit Rücksetzeingang,
- Anordnung von Pull-up- und Pull-down-Widerständen,
- Nutzung moderner Leistungselektronik-Schaltkreise, die entsprechende Vorkehrungen haben.

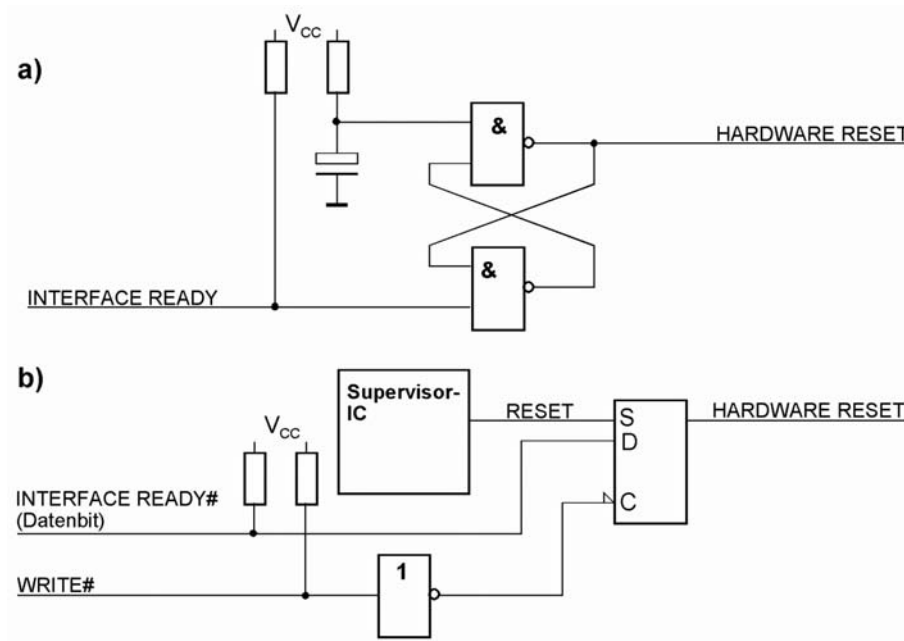


Abb.1.33 Bewährte Rücksetzsaltungen

a) - eine uralte Praxisschaltung. Erzeugen des Rücksetzimpulses über ein RC-Glied. b) - es gibt eigene Schaltkreise, die das Ansteigen und Abfallen der Speisespannung auswerten, um entsprechende Rücksetzimpulse zu bilden (Voltage Supervisors). Beide Schaltungen erzeugen ein Signal zum Hardware-Rücksetzen, das vom Einschalten an solange aktiv bleibt, bis über das Interface eine entsprechende Bereitschaftsmeldung (INTERFACE READY) kommt. Die Abbildung zeigt zwei Auslegungen: a) - mit eigenem Signal zur Bereitschaftsanzeige; b) - mit Übernahme einer Datensignalbelegung (das Rücksetz-Flipflop wird wie ein Ausgaberegister über einen Datenbus angesprochen).

Manche Interfaces haben eine Pegelzuordnung, die es ermöglicht, den Betriebszustand zu erkennen (Beispiel: USB). Gelegentlich kann man Ähnliches selbst verwirklichen, z. B. eine Signalleitung mit einem Pull-up-Widerstand beschalten und den ersten Low-Impuls als Bereitschaftsanzeige definieren.

Übers Interface zurücksetzen

Interfaces mit höherentwickelten Signalprotokollen erlauben es, die angeschlossenen Geräte softwareseitig zurückzusetzen oder bestimmte Initialisierungs- und Wiederanlaufvorgänge auszulösen. Es ist sinnvoll, auch in privaten Protokollen und Kommandosätzen so etwas vorzusehen.

Pegelwandlung usw.

– Nur zur Information –

(Komm nicht dran.)

Isolation (galvanische Trennung)

Optokoppler. Kann auch Gleichspannungssignale übertragen. Rückwirkungsfrei. Keine Leistungsübertragung.

Transformator

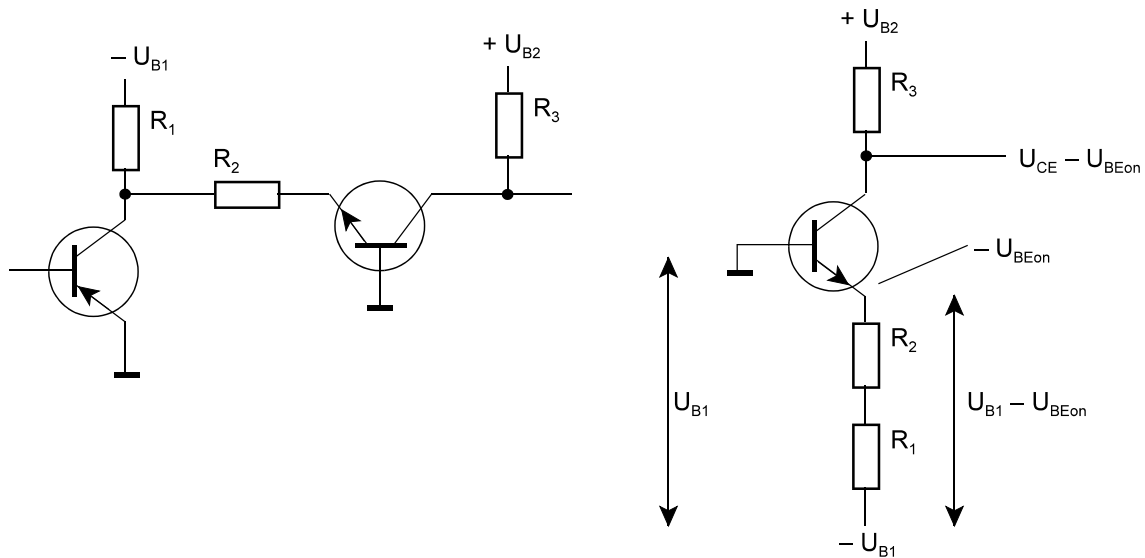
Kondensator

Input

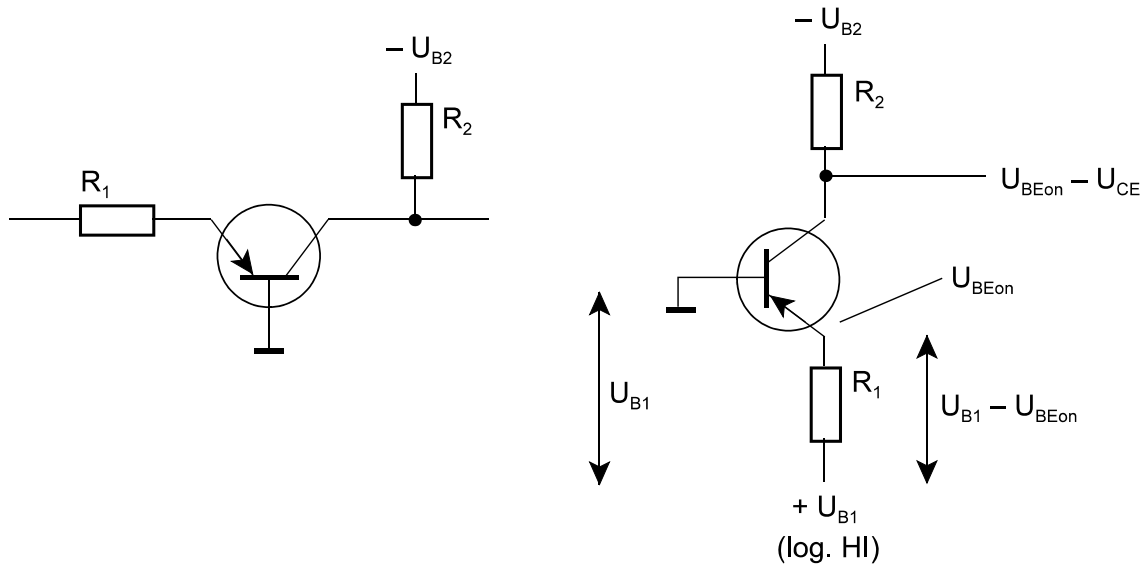
- höhere Pegel
- niedrigere pegel
- negative Pegel
- Isolation

Output

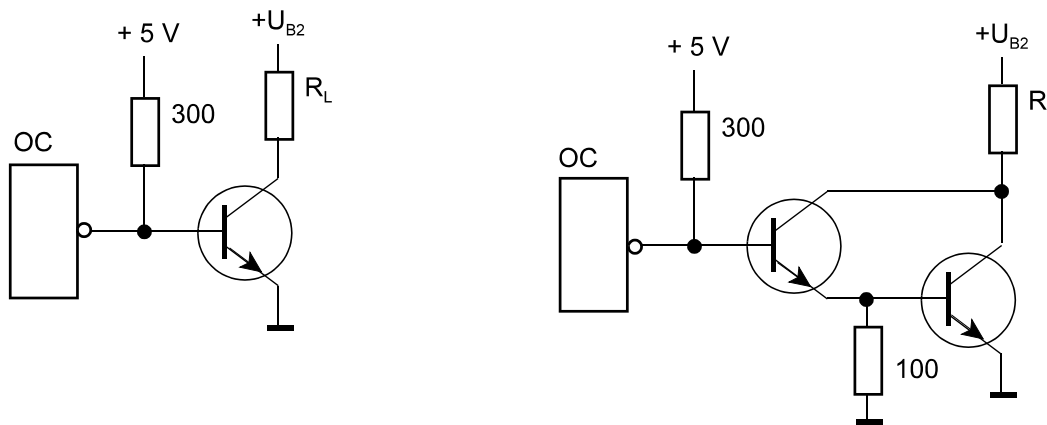
- höhere Pegel
- niedrigere Pegel
- negative pegel
- Isolation



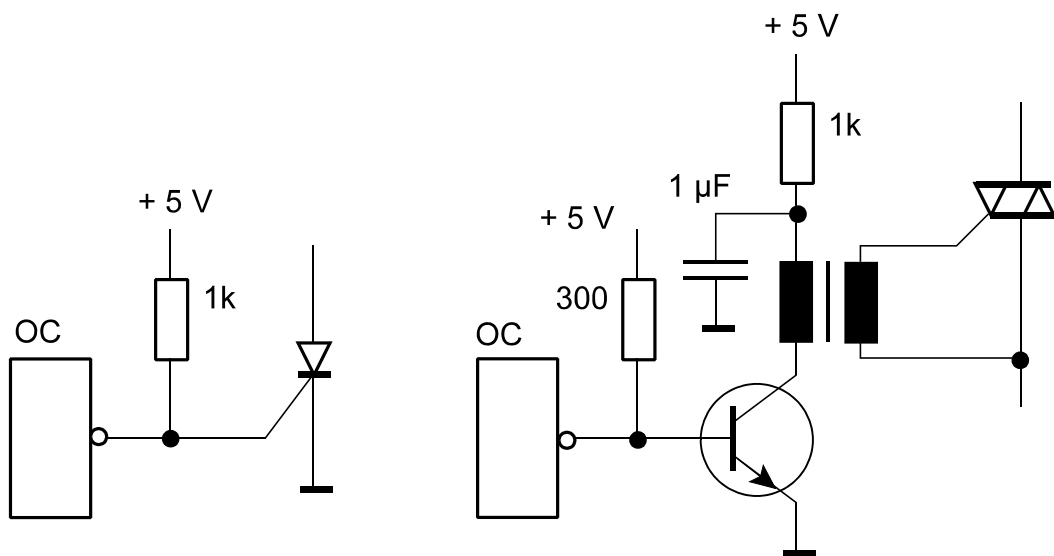
Wandlung negativ => positiv.



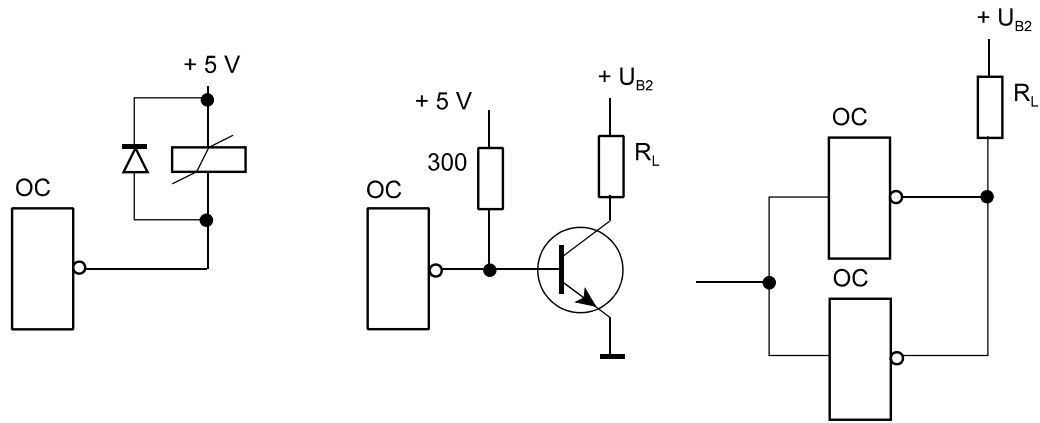
Wandlung positiv => negativ.



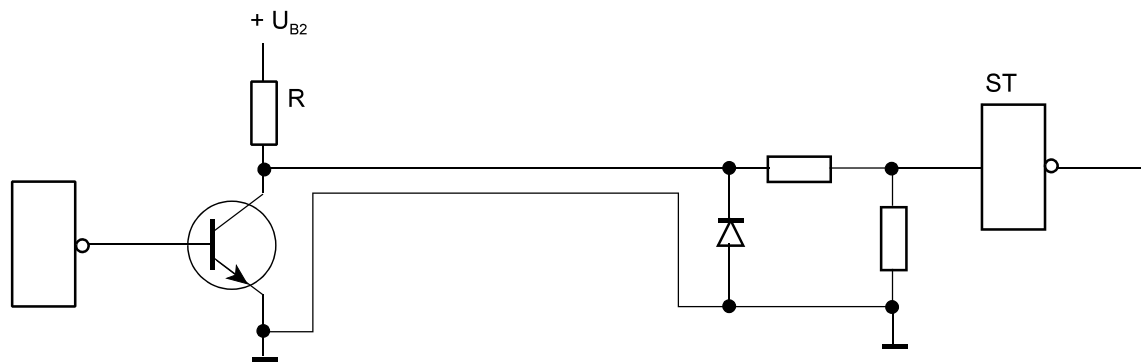
Treiber mit Open-Collector-Schaltkreis (1).



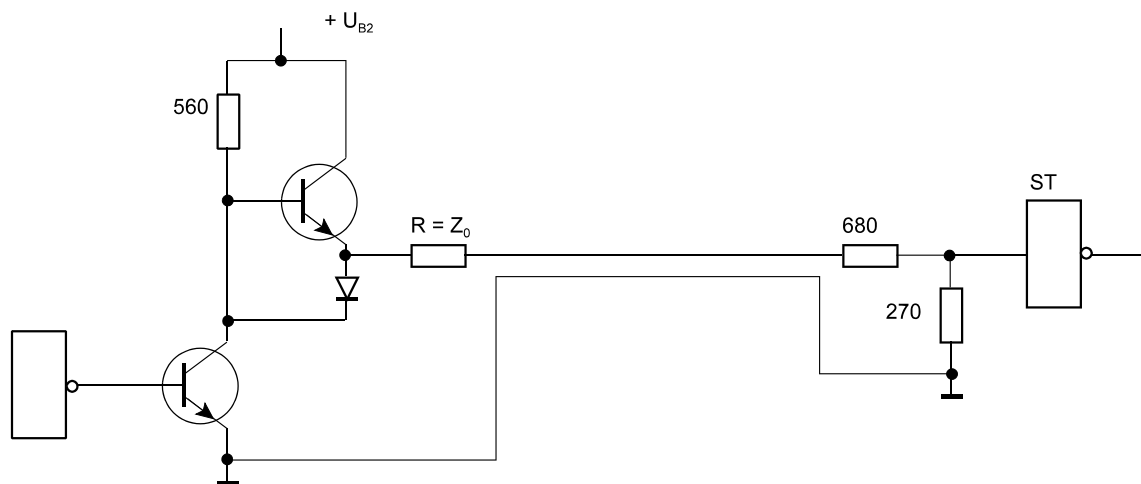
Treiber mit Open-Collector-Schaltkreis (2).



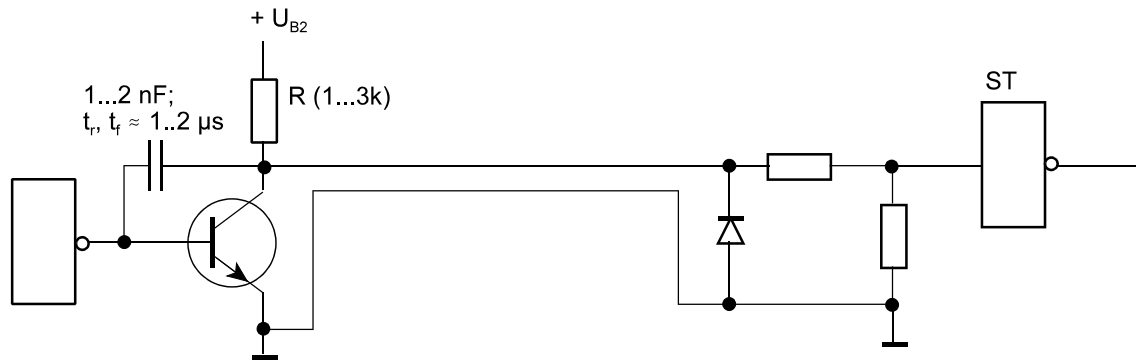
Treiber mit Open-Collector-Schaltkreis (3).



Treiben einer längeren Signalleitung (1).



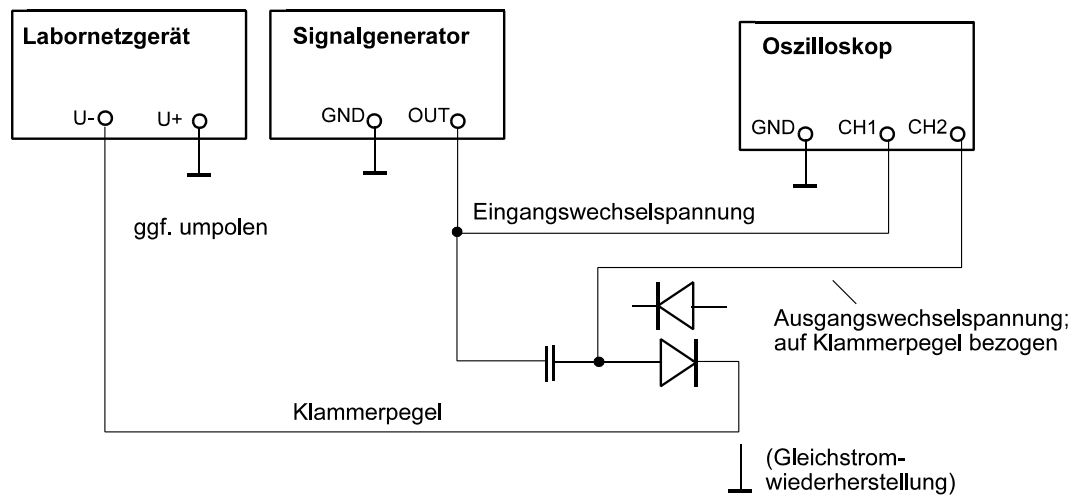
Treiben einer längeren Signalleitung (2).



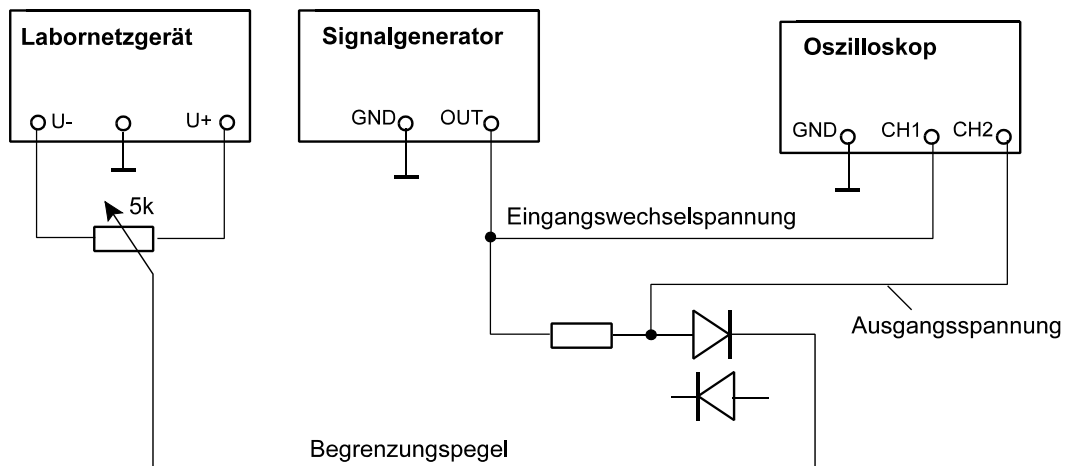
Treiben einer längeren Signalleitung (3).

Typische Logikpegel 24 V:

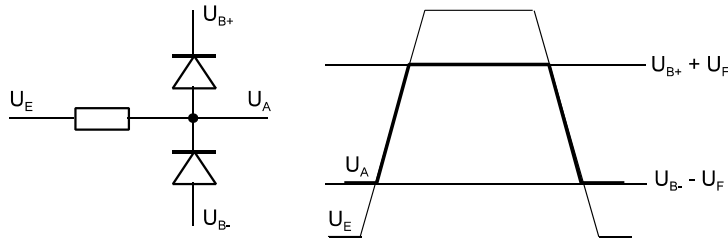
	min.	typ.	max.
Low-Pegel	-0,5 V		1,5 V
Schwellspannung		6,0 V	
High-Pegel	15 V		35 V



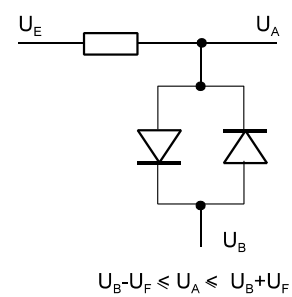
Gleichstromwiederherstellung mit Diode (Versuchsaufbau).



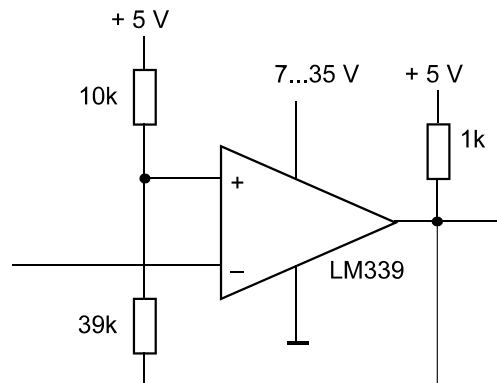
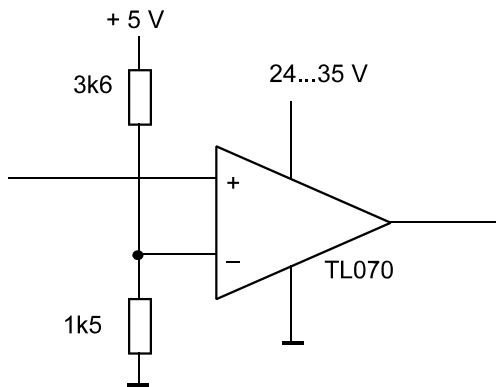
a)



b)



Signalbegrenzung mit Diode (Versuchsaufbau).



Links: 5 V auf 24 V, rechts 24 V auf 5 V.

TL70: unkompensierter Operationsverstärker (schneller, kurzschlußfest).

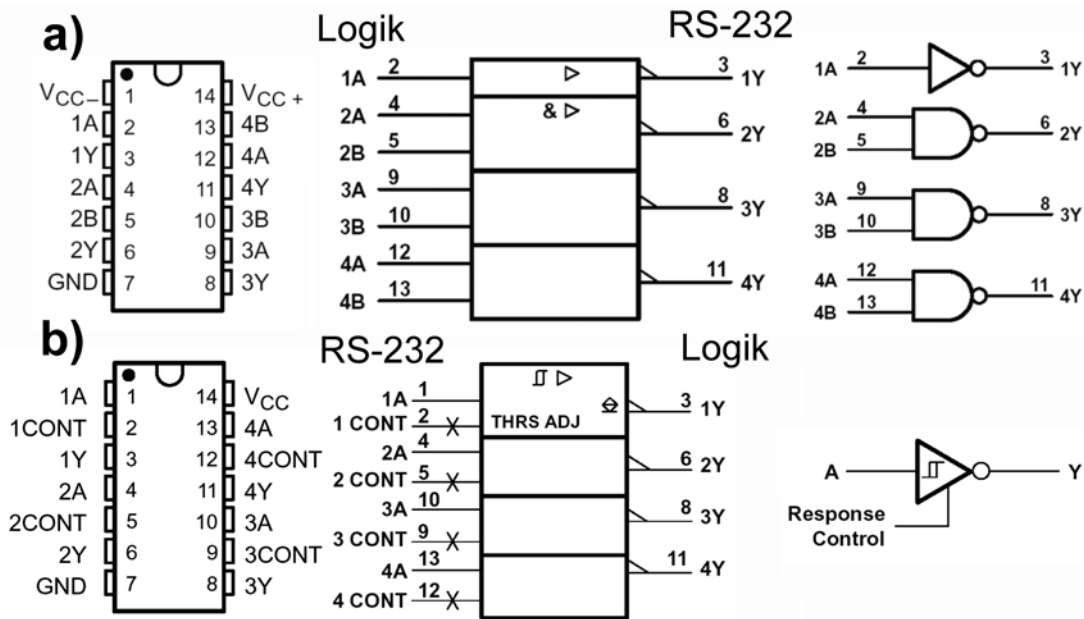


Abb.1.34 Herkömmliche Treiber und Empfänger (Texas Instruments)

a) - Treiber MC1488; b) - Empfänger MC1489. Jeder Schaltkreis enthält 4 Baustufen. Arbeitsweise: positive Logik (Low \times Space, High \times Mark). Es gibt Ausführungen in verschiedenen Technologien, die sich hinsichtlich der Speisespannungen und der Stromaufnahme voneinander unterscheiden (andere Typenbezeichnungen: z. B. 75188/189). Typische Speisespannungen:

- Treiber: V_{CC+} : bis + 15 V; V_{CC-} : bis - 15 V (Datenblatt: Mindestwerte $\pm 7,5$ V, Nennwerte ± 9 V, Höchstwerte ± 15 V), Kurzschlußstrom: $\pm 6...12$ mA (typisch ± 9 mA),
- Empfänger: $V_{CC} = + 5$ V.

Die Empfänger haben Steuereingänge (CONT/Response Control) zum Beeinflussen der Schaltschwelle bzw. der Störimpfindlichkeit (Filterwirkung). Typische Beschaltungen:

- keine Sonderwirkung: offen,
- Schaltschwellenbeeinflussung: Widerstand nach Masse,
- Filterwirkung: Kondensator nach Masse. Beispiel: 330 pF unterdrücken Störimpulse von 3 V Amplituden und 300 ns Dauer.

Zur seriellen Schnittstelle

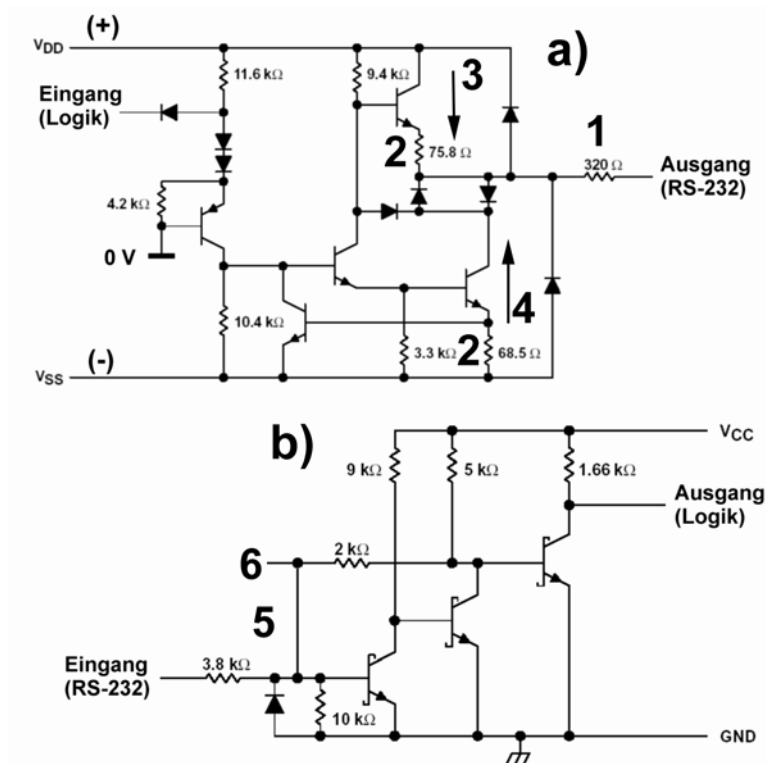


Abb.1.35 Typische Innenschaltungen (Texas Instruments)

a) - Treiber; b) - Empfänger; 1, 2 - Strombegrenzungswiderstände; 3 - Stromweg bei Space; 4 - Stromweg bei Mark; 5 - Eingangsspannungsteiler; 6 - in manchen Schaltkreisen als Steuereingang herausgeführt (vgl. Abbildung 8.27b): V_{DD} - positive Speisespannung (typisch + 12 V oder + 9 V), V_{SS} - negative Speisespannung (typisch - 12 V oder - 9 V), V_{CC} - Logikspeisespannung (typisch + 5 V).

Zum Treiber:

Der Kurzschlußstrom wird durch die Widerstände 1, 2 begrenzt. Sehen wir die Transistoren als ideale Schalter an, so ergibt sich der Innenwiderstand des Treiberausgangs aus der Reihenschaltung von R1 und R2. Das sind hier rund 400 Ohm. Somit beträgt der Kurzschlußstrom bei 12 V etwa 33 mA (in der Praxis deutlich weniger, da auch die Transistoren den Stromfluß begrenzen). *Anwendung in der Fehlersuchpraxis:* bei einigen hundert Ohm Innenwiderstand führen die typischen Lasten im Bereich mehrerer kOhm zu einem deutlich meßbaren Spannungsabfall. Beispiel: bei 12 V und 400 Ohm Treiberinnenwiderstand messen wir über einer Last von 3 kOhm (Empfängereingangswiderstand) eine Spannung von 10,5 V.

Zum Empfänger

Der Serienwiderstand am Eingangsspannungsteiler sorgt für den standardgemäßen Eingangswiderstand (3...7 kOhm). Ein offener Empfängereingang führt nahezu Massepegel (≈ 0 V). Der Empfänger ist kein Comparator, sondern ein simpler Schwellwertschalter, der zudem nur auf positive Eingangsspannungen reagieren kann (eine negative Eingangsspannung wird grundsätzlich als Mark interpretiert und führt zu einem High-Pegel am Ausgang). Er liefert ausgangsseitig nur Nullen oder Einsen, zeigt aber nicht an, daß das Schnittstellensignal im verbotenen Bereich liegt.

- Schaltschwelle in positiver Richtung (von Mark nach Space): typisch um 2 V,

- Schaltschwelle in negativer Richtung (von Space nach Mark): typisch um 1 V,
- Hysterese: typisch um 1 V,
- Ausgangspegel bei offenem oder mit Masse verbundenem Eingang: High.

Zur Spannungsversorgung:

- Empfänger werden mit der jeweils typischen Logik-Speisespannung (+ 5 V, + 3,3 V usw.) versorgt. Das ist im Grunde eine Sparlösung. Damit sie funktioniert, legt man die Schaltschwellen (wie soeben beschrieben) in den positiven Spannungsbereich (> 0 V).
- Treiber brauchen eine positive und eine negative Versorgungsspannung. Hierfür gibt es folgende Auslegungen:
 - C beide Versorgungsspannungen (z. B. + 12 V und - 12 V) werden von außen zugeführt,
 - C es wird eine positive Versorgungsspannung zugeführt (z. B. + 5 V). Diese wird für den Space-Pegel verwendet. Die negative Spannung (für den Mark-Pegel) wird mit einer Ladungspumpe (Inverter) erzeugt.
 - C es wird eine positive Spannung zugeführt (z. B. 2,5 oder 3,3 V). Die Spannungen für beide Pegel (Space und Mark) werden mit Ladungspumpen erzeugt. Eine typische Konfiguration: die positive Spannung wird durch Verdopplung erzeugt, die negative durch einen nachgeschalteten Inverter.

Verbindungen über serielle Schnittstellen funktionieren unter folgenden Bedingungen weitgehend problemlos:

- Kabel nicht länger als 15...20 m (Richtwert);
- beide Einrichtungen am selben Netzstromkreis oder wenigstens am selben Stromkreisverteiler.

Das betrifft typische Arbeitsplatzverkabelungen, Serverschränke und auch Installationen innerhalb eines nicht allzu großem Büroraums.

Da es sich um eine Signalübertragung gegen Masse handelt, ist die Schnittstelle auf gleiches Massepotential an beiden Enden angewiesen. Wird RS-232 verwendet, um größere Entfernungen zu überbrücken, ist diese Bedingung nicht immer erfüllt. Typische Störeinflüsse:

- Erdschleifen durch unterschiedliche Massepotentiale (z. B. beim Anschluß der Geräte an unterschiedliche Netzstromkreise bzw. Stromkreisverteiler),
- eingekoppelte Störungen, z. B. von in der Nähe liegenden Starkstromkabeln,
- Gleichtaktstörungen durch zeitweilige Anhebung eines Massepotentials. Typische Ursachen:
 - C das Einleiten von Über- und Fehlerspannungen in Erdverbindungen (Abbildung 8.50),
 - C die Anhebung des Erdpotentials durch Blitzschlag (Abbildung 8.51),
 - C die allgemein übliche Nutzung des Schutzleiters zum Verbinden von Signalmasse und Erdpotential. Das geht solange gut, bis der Schutzleiter einmal seine eigentliche Aufgabe zu erfüllen hat (das Fließenlassen eines Kurzschlußstromes, um die Sicherung im Netzstromkreis auszulösen; vgl. Abbildung 8.15).

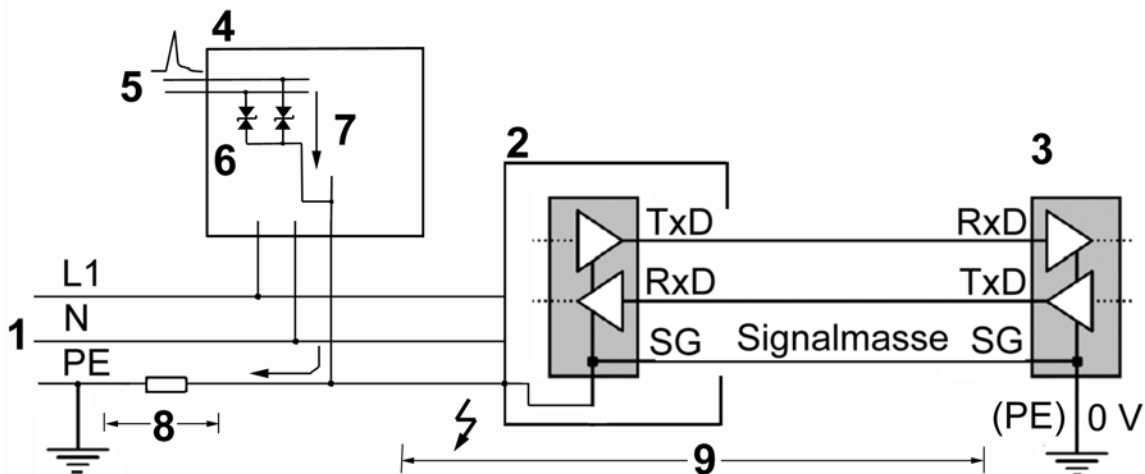


Abb.1.36 Eingeleitete Über- und Fehlerspannungen heben das Erdpotential an

1 - ein 230-V-Netzanschluß; 2, 3 - miteinander verbundene RS-232-Geräte. Die Geräte 2, 3 sind an verschiedene Netzstromkreise angeschlossen. Die Signalmasse ist über den jeweiligen Schutzleiter mit dem Erdpotential verbunden. 4 - ein weiteres (an sich beliebiges) Gerät am Netzstromkreis 1; 5 - ein weiteres (an sich beliebiges) Interface; 6 - Überspannungsableiter (z. B. Suppressordioden); 7 - Störungen am Interface 5 werden von den Überspannungsableitern 6 über den Schutzleiter zur Erde hin abgeführt; 8 - dieser Stromfluß bewirkt einen Spannungsabfall über den Schutzleiter (dessen Durchgangswiderstand ist hier symbolisch angedeutet); 9 - hierdurch wird das Erdpotential des Gerätes 2 gegenüber dem des Gerätes 3 kurzzeitig angehoben.

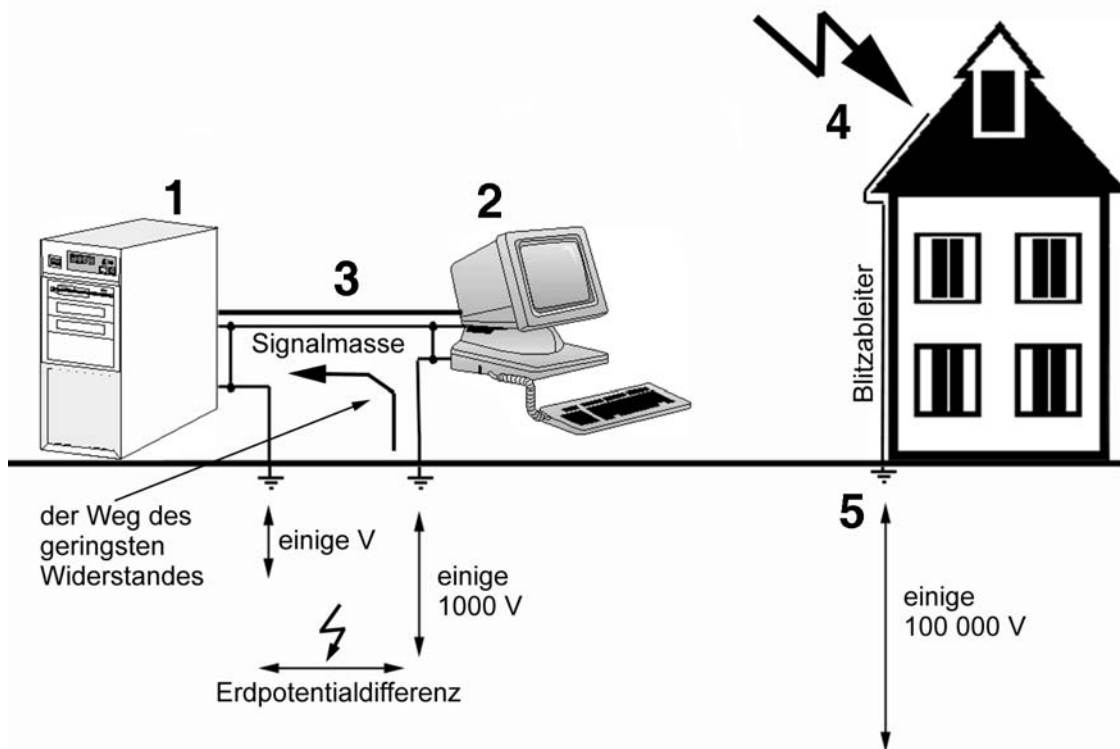


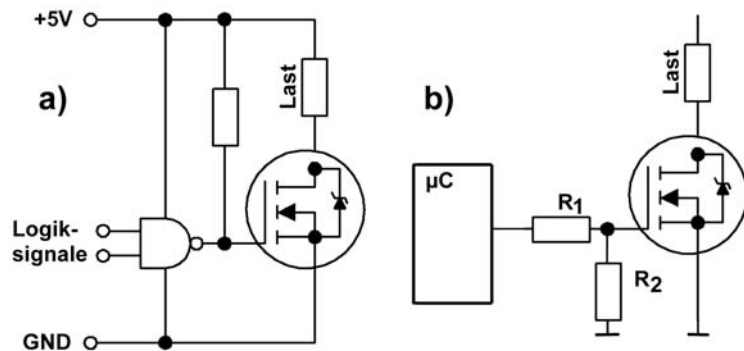
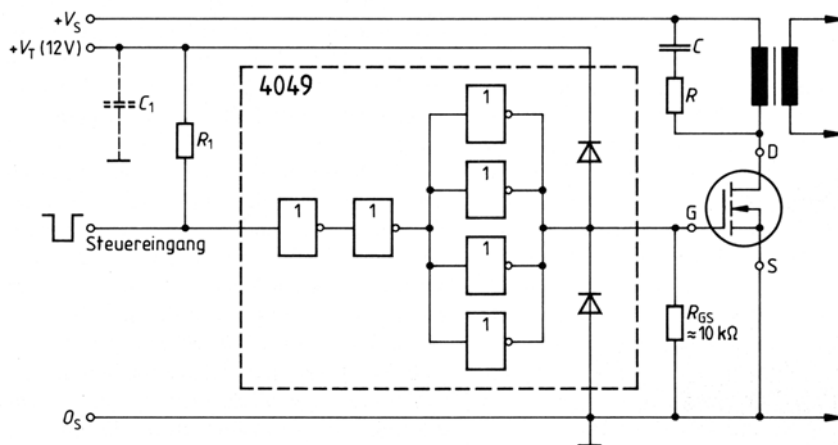
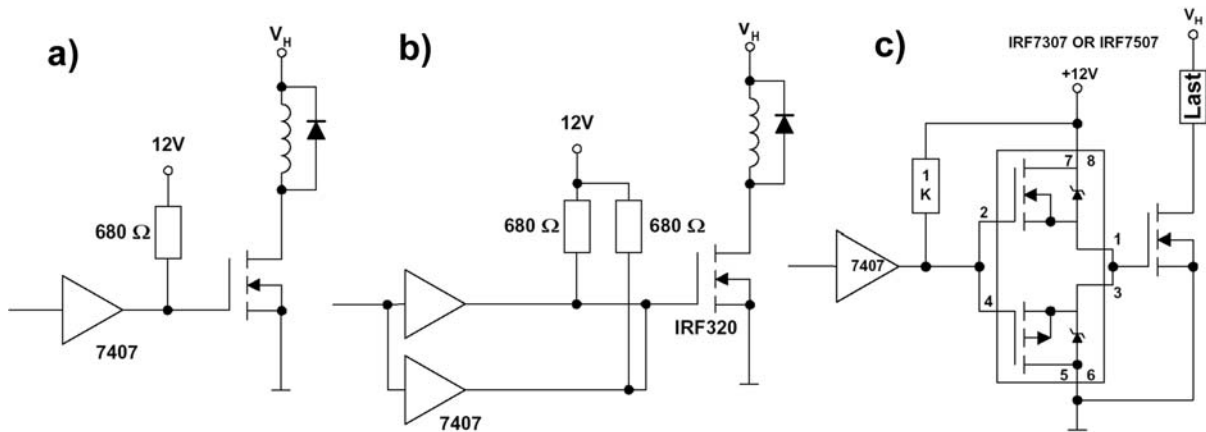
Abb.1.37 Anhebung des Erdpotentials durch Blitzschlag

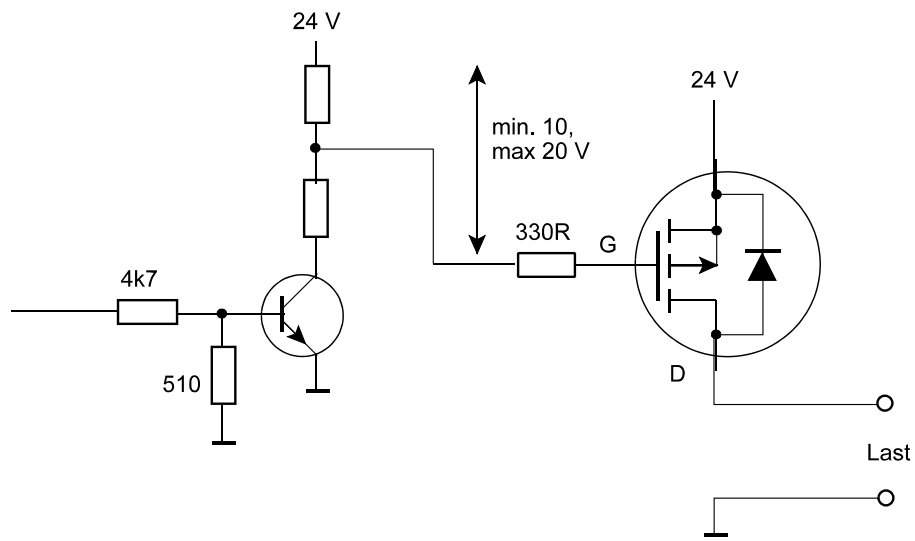
1, 2 - miteinander verbundene Geräte; 3 - Interfacekabel. Die Geräte 1, 2 haben unterschiedliche Erdverbindungen (z. B. über verschiedene Betriebsrder in den jeweiligen Gebäuden). 4 - Blitzeinschlag in ein weiteres Gebäude (in der Nähe). 5 - die Energie des Blitzes wird in die Erde abgeleitet. Am Ort des

Einschlags ist die Spannung am höchsten. Sie nimmt mit wachsender Entfernung ab. Es kann sein, daß das Erdpotential am Betriebserder des Gerätes 2 kurzzeitig auf einige tausend Volt ansteigt. An sich schadet das nicht, weil alles, was auf dem selben Fleck Erde steht, auf das gleiche Potential angehoben wird. Es gibt aber eine niederohmige Verbindung*) zum noch weiter entfernten Gerät 1, dessen Erdpotential nur wenig von 0 V abweicht - das Interfacekabel 3. Die Erdpotentialdifferenz zwischen den Geräten 1, 2 bewirkt somit einen Stromfluß über das Kabel (= über den Weg des geringsten Widerstands). Somit kommt an der Schnittstelle des Gerätes 1 ein Impuls mit einigen tausend Volt Amplitude an...

*) : Kabel haben typischerweise nur wenige Ohm Durchgangswiderstand.

Einige Treiberschaltungen für Leistungs-FETs





Typische Schutzmaßnahmen im Überblick (nach Vishay):

