

Versuch: P1-64

Digitale Elektronik, Schaltlogik

- Vorbereitung -

Die Grundlage unserer modernen Welt liegt begründet in der digitalen Elektronik und der Schaltlogik - deshalb ist es als essentiell anzusehen, dass angehende Physiker, aber auch Lehrer, welche die moderne Welt der nächsten Generation erklären sollen, diese Vorgänge verstehen. Dieser Versuch soll die nötigen Kenntnisse dafür fördern.

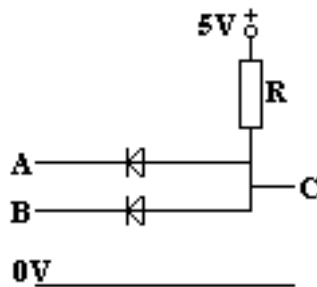
Inhaltsverzeichnis

1	Gatter aus diskreten Bauelementen	2
1.1	AND-Gatter	2
1.2	NOT und NAND-Gatter	3
1.3	OR-Gatter	4
2	Gatter, realisiert mit ICs	4
2.1	Inverter	4
2.2	EXOR	5
2.3	EXOR mit NAND-Gattern	5
3	Addierer	6
3.1	Halbaddierer	6
3.2	Volladdierer	6
3.3	Subtrahierer	7
4	Speicherelemente	8
4.1	RS-Flip-Flop	8
4.2	Getaktetes RS-Flip-Flop	9
4.3	JK-Master-Slave-Flip-Flop	10
5	Schieben, Multiplizieren, Rotieren	11
5.1	4-Bit-Schieberegister	11
5.2	Parallele Eingabe, Rotationsregister	11
6	Zähler	12
6.1	4-Bit-Asynchrnzähler	12
6.2	Asynchroner Dezimalzähler	13
6.3	4-Bit-Synchrnzähler	14
6.4	Synchrner Dezimalzähler	14
7	Digital-Analog-Wandlung	15

1 Gatter aus diskreten Bauelementen

1.1 AND-Gatter

Die Schaltung für das AND-Gatter wird folgendermaßen aufgebaut:



Die Punkte A und B können voneinander unabhängig mit einer äußeren Spannung $U_0 = 5V$ verbunden werden. Dieser Zustand entspräche *logisch 1*. Desweiteren können sie auch mit der 0V-Leitung verbunden werden, was dem Zustand *logisch 0* entspräche. Der Widerstand R wird so gewählt, dass er:

- klein ist gegen den Sperrwiderstand
- groß gegen den Durchlasswiderstand

ist. Sind nun beide Punkte A und B logisch 1, so sperren die Dioden und die Spannung U_0 liegt nun am Punkt C an, womit dieser den Wert logisch 1 einhält.

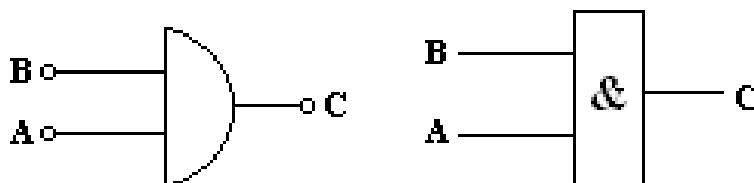
Verbindet man aber einen der beiden Punkte mit der Leitung mit $U = 0V$, so fließt dort ein Strom, der die Spannung am Punkt C im Vergleich zu logisch 1 sinken lässt, weshalb sich der Punkt dann im Zustand logisch 0 befindet. Schaltet man die andere Diode auch auf logisch 0, so sinkt die Spannung logischerweise weiter ab und der Wert für C bleibt bei logisch 0.

Wir können eine dementsprechende Wahrheitstabelle aufschreiben:

logischer Zustand von A	logischer Zustand von B	logischer Zustand von $A \wedge B$
0	0	0
0	1	0
1	0	0
1	1	1

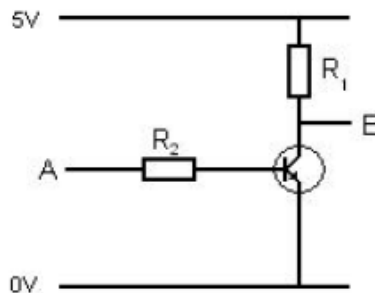
Der Punkt C ist also nur dann logisch 1, wenn beide, A UND (englisch: AND) B den Zustand logisch 1 haben.

Für diese Situation gibt es zwei verschiedene Schaltsymbole:



1.2 NOT und NAND-Gatter

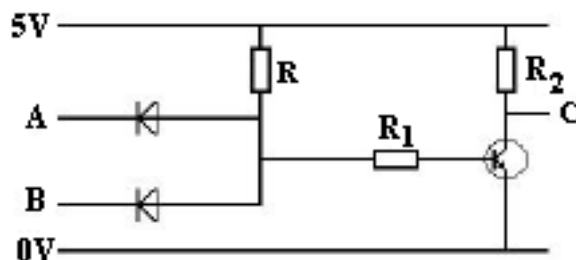
Eine sogenannte NOT-Schaltung hat folgendes Schaltbild:



Die NOT-Schaltung funktioniert folgendermaßen: Man kann wieder am Punkt A eine Spannung anlegen, einweder 0V (logisch 0) oder 5V (logisch 1). Im ersten Fall sperrt der Transistor und die gesamte Spannung fällt am Punkt B ab (für diesen: logisch 1). Im zweiten Fall, also beim Anlegen von 5V, schaltet der Transistor durch, wodurch dann allerdings der Punkt B auf logisch 0 steht. Er ergibt sich also die folgende Wahrheitstabelle:

A	B (= $\neg A$)
1	0
0	1

Das sogenannte NAND-Gatter ergibt sich schon vom Namen her aus der Hintereinanderschaltung eines NOT- und eines AND-Gatters, was folgendermaßen aussieht:

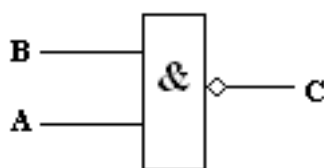


Die logische Struktur verhält sich analog zur Schaltung: „in Reihe“. Mit anderen Worten: zuerst stellen sich ganz normal die Werte bei A und B ein, wie bei der AND-Schaltung üblich. Durch die dahinter geschaltete NOT-Schaltung erfolgt die Negation, was folgende Wahrheitstabelle zur Konsequenz hat:

A	B	C = $\neg(A \wedge B)$
1	1	0
1	0	1
0	1	1
0	0	1

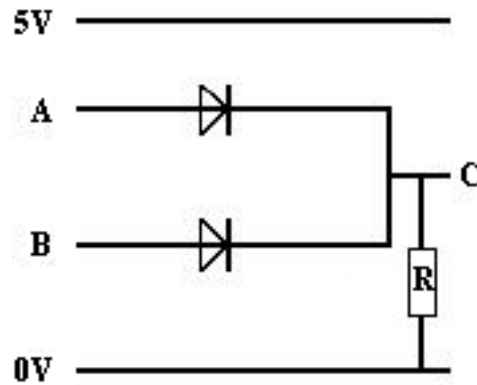
Da hier Transistor und Diode verwendet werden, bezeichnet man das NAND-Gatter auch als DTL-Baustein (Dioden-Transistor-Logik).

Schaltsymbol:



1.3 OR-Gatter

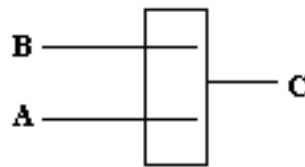
Dem OR-Gatter liegt folgende Schaltung zugrunde:



Hier wird klar: C befindet sich auf logisch 1, wenn entweder die A-Diode durchschaltet, die B-Diode durchschaltet oder beide durchschalten. Sind beide, A und B, an 0V angeschlossen, dann sperren die Dioden und C liegt auf logisch 0. Die Wahrheitstabelle lautet dementsprechend:

A	B	$C = A \vee B$
0	0	0
1	0	1
0	1	1
1	1	1

Das Schaltsymbol lautet:



Bei einem NOR-Gatter wird noch ein Kreis für die Negation (wie beim AND) eingefügt.

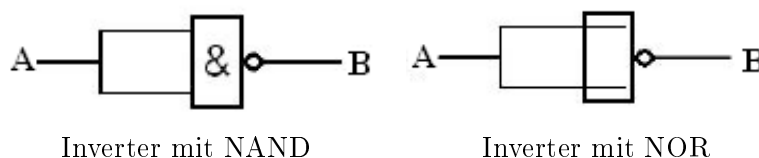
2 Gatter, realisiert mit ICs

Nun werden wir nicht mehr die oben in Aufgabe 1 verwendeten Schaltungen immer wieder einzeln aufbauen, sondern wir verwenden Bauteile, die diese Schaltungen bereits in sich vereinen. Diese Bauteile heißen *Integrated Circuits*, kurz ICs. Bei den ICs wirken freie Eingänge so, als seien sie auf logisch 1.

2.1 Inverter

Das hier aufzubauende NOT-Gatter soll ein digitaler Inverter sein. Aufgebaut werden soll er durch das Verbinden der beiden Eingänge eines NAND- oder eines NOR-Gatters. Da dadurch bei A und B das gleiche Potential sein muss, reduziert sich die Wahrheitstabelle auf:

A	$A \neg B$
0	1
1	0



Mit dieser Schaltung werden also binäre Werte negiert. Eine andere Möglichkeit der Schaltung wäre eine dauerhafte Schaltung eines Eingangs vom NAND-Gatter auf logisch 1, bzw. im NOR-Gatter auf 0.

2.2 EXOR

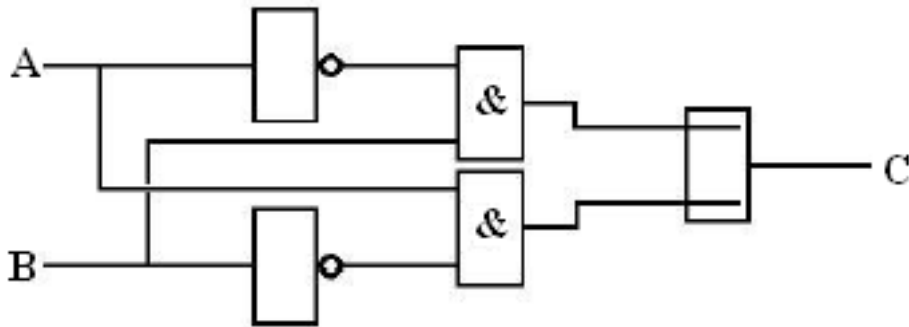
In der Vorbereitungsmappe wurde bereits die Wahrheitstabelle eines EXOR-Gatters gegeben:

a	b	f
0	0	0
0	1	1
1	0	1
1	1	0

Aus der Tabelle kann man die disjunktive Normalform der Schaltung ablesen, sie entspricht einem umgangssprachlichen „entweder oder“, wobei also nur eines der Argument wahr sein darf:

$$(a \wedge \bar{b}) \vee (\bar{a} \wedge b) \text{ mit } (\bar{a} = \neg a) \quad (1)$$

Die Schaltung könnte folgendermaßen aussehen:



2.3 EXOR mit NAND-Gattern

Wir gehen vom folgenden Term aus und versuchen einen „schöneren“ zu finden, d.h. der uns einer Realisierung auf dem Schaltbrett näher bringt (mit Kurzschreibweise $ab = a \wedge b$):

$$f = \overline{\overline{a \bar{a} b} \overline{b \bar{a} b}} \quad (2)$$

$$= \overline{[a \wedge (\overline{a \wedge b})] \wedge [b \wedge (\overline{a \wedge b})]} \quad (3)$$

$$= \overline{[a \wedge (\overline{a \wedge b})] \vee [b \wedge (\overline{a \wedge b})]} \quad (4)$$

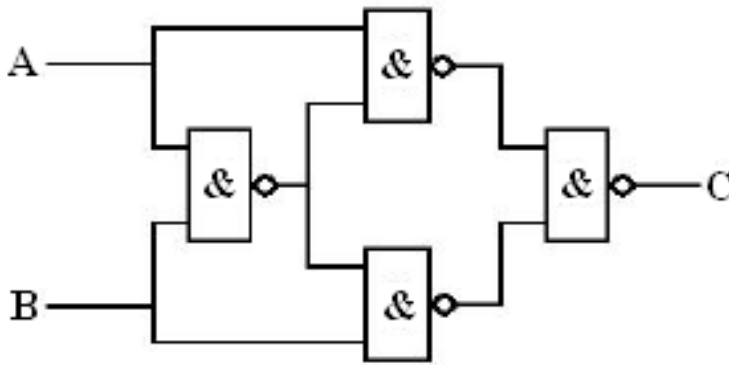
$$= [a \wedge (\overline{a \wedge b})] \vee [b \wedge (\overline{a \wedge b})] \quad (5)$$

$$= [a \wedge (\bar{a} \vee \bar{b})] \vee [b \wedge (\bar{a} \vee \bar{b})] \quad (6)$$

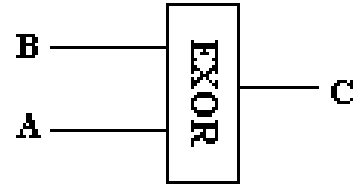
$$= (\bar{a} \wedge b) \vee (a \wedge \bar{b}) \vee (a \wedge \bar{a}) \vee (\bar{b} \wedge \bar{b}) \quad (7)$$

$$= (\bar{a} \wedge b) \vee (a \wedge \bar{b}) \quad (8)$$

Man könnte die EXOR-Schaltung also folgendermaßen aufbauen:



EXOR-Schaltung



Schaltsymbol EXOR

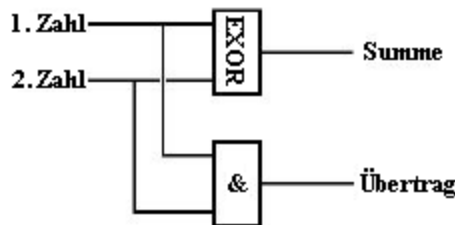
3 Addierer

3.1 Halbaddierer

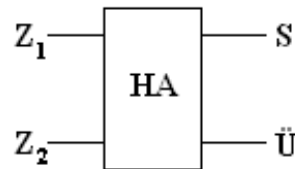
Der Halbaddierer soll zwei einstellige Dualzahlen addieren und er soll folgende Wahrheitstafel umsetzen können:

1. Zahl	2. Zahl	Summe	Übertrag
0	0	0	0
1	0	1	0
0	1	1	0
1	1	0	1

Scharfes Anschauen der Tabelle lässt bereits verwendeten Bauteile erkennen - nämlich EXOR für die Summe und AND für den Übertrag. Damit folgt für den Halbaddierer die folgende Schaltung:



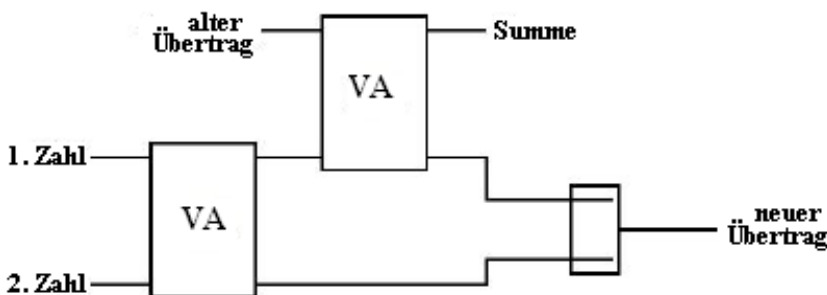
Aufbau Halbaddierer



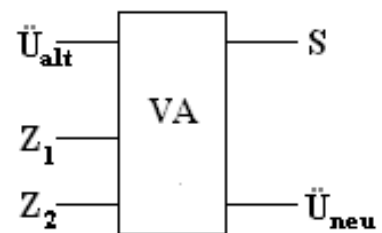
Schaltsymbol Halbaddierer

3.2 Volladdierer

Um aus der Beschränkung der Addierbarkeit von einstelligen Zahlen herauszukommen, schaltet man mehrere *Volladdier* in Reihe. Diese 1-Bit Addierer werden benötigt, um mehrstellige Dualzahlen zu addieren, für jede Stelle wird ein Addierer gebraucht. Somit kann ein 1-Bit Addierer auch „Addierer für drei einstellige Dualzahlen“ heißen. Somit wird der Volladdierer folgendermaßen realisiert:



Aufbau Volladdierer



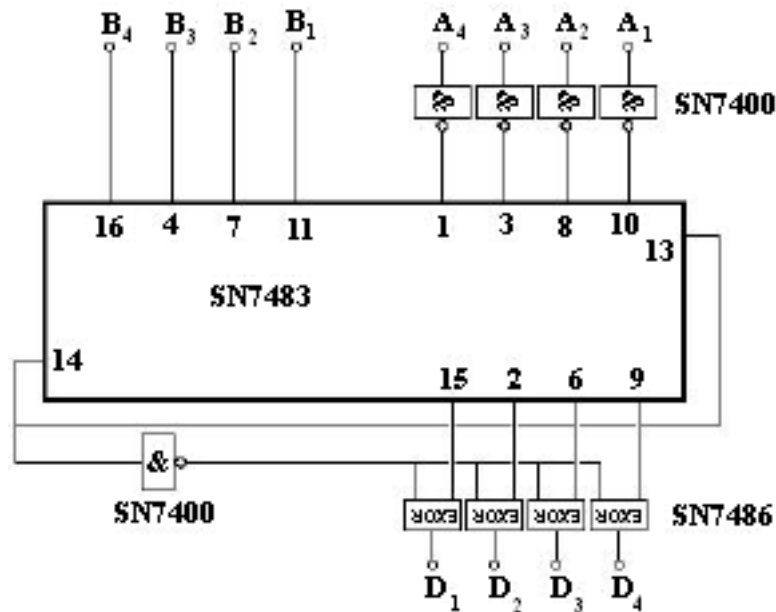
Schaltsymbol Volladdierer

Es ergibt sich folgende Wahrheitstafel:

1. Zahl	2. Zahl	alter Übertrag	Summe	neuer Übertrag
1	0	0	1	0
1	1	0	0	1
1	0	1	0	1
1	1	1	1	1
0	1	0	1	0
0	0	1	1	0
0	1	1	0	1
0	0	0	0	0

3.3 Subtrahierer

Der Subtrahierer wird wie in Abb.4 der Vorbereitungsmappe aufgebaut:



Um die Subtraktion durchzuführen wird schlicht die verwendet, dass $b - a = b + (-a)$ ist. Um $(-a)$ zu erhalten verwendet man $a + \bar{a} = 1111 = 10000 - 0001 \Rightarrow -a = a + 10000 - 0001$. Man verwendet eine 5.Ziffer als Ausgangsübertrag, um das Vorzeichen darstellen zu können. Man verwendet insgesamt 4 Volladdierer.

An den Eingängen B_1 bis B_4 gibt man die Zahl b direkt als Dualzahl ein. Entsprechend erfolgt die Eingabe von a an A_1 bis A_4 , wobei diese mittels NAND-Gatter zu \bar{a} geändert wird.

An der Ausgangsseite befinden sich EXOR-Gatter. Diese haben bei positivem Ergebnis des Ergebnisses der Subtraktion keine Bedeutung. Am Übertrag liegt logisch 1 an und wird auch als Eingangsübertrag wieder eingegeben.

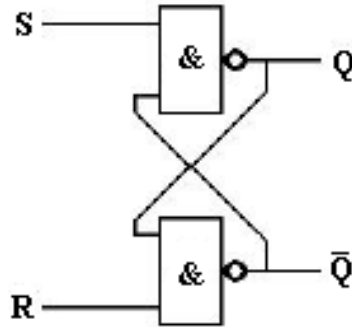
Wird das Ergebnis der Subtraktion aber negativ, so invertieren die EXOR-Gatter dieses Ergebnis, so dass der Betrag der Zahl ausgegeben wird. Daher muss der Ausgangsübertrag (jetzt logisch 0) mit dem Eingangsübertrag verbunden sein (eigentlich müsste man 0001 addieren, alternativ kann man aber auch 0001 vom Komplement subtrahieren). Zuletzt wird für die Ausgabe der SGN-Ausgang (von signum = Vorzeichen) invertiert, da üblicherweise negative Zahlen ein Vorzeichen haben und nicht positive, so wie es bisher der Fall ist.

4 Speicherelemente

Es werden im Folgenden einige Typen von Flip-Flops(FF) vorgestellt, welche als digitale Speicher dienen.

4.1 RS-Flip-Flop

Ein Rs-Flip-Flop besteht aus zwei miteinander verzahnten NAND-Gattern



wodurch sich folgende Schaltlogik ergibt:

$$Q = \bar{S} + RQ = \bar{S} \vee (R \wedge Q) \quad (9)$$

$$\bar{Q} = \bar{R} + S\bar{Q} = \bar{R} \vee (S \wedge \bar{Q}) \quad (10)$$

womit sich folgende mögliche Systemzustände ergeben:

α) $R = S = 1$: Es folgt aus (9) und (10):

$$Q = 0 \vee (1 \wedge Q) = Q$$

$$\bar{Q} = 0 \vee (1 \wedge \bar{Q}) = \bar{Q}$$

also sind beide Ausgänge stabil, sie behalten ihren Zustand bei und somit die Speicherfunktion des Elements bewirken.

β) $R = 1, S = 0$: In diesem Fall gilt:

$$Q = 1 \vee (1 \wedge Q) = 1$$

$$\bar{Q} = 0 \vee (0 \wedge \bar{Q}) = 0$$

Damit kann mittels $S = 0$ den Ausgang Q auf logisch 1 und \bar{Q} auf 0 setzen.

γ) $R = 0, S = 1$ In diesem Fall gilt:

$$Q = 0 \vee (0 \wedge Q) = 0$$

$$\bar{Q} = 1 \vee (1 \wedge \bar{Q}) = 1$$

Somit kann man mittels $R = 0$ und $S = 1$ das Gegenteil erreichen, d.h. Q auf logisch 0 und \bar{Q} auf 1 setzen.

δ) $R = 0, S = 0$ In diesem Fall würde gelten:

$$Q = 1 \vee (0 \wedge Q) = 1$$

$$\bar{Q} = 1 \vee (0 \wedge \bar{Q}) = 1$$

was allerdings hieße, dass beide Ausgänge $Q = \bar{Q} = 1$ sind - deshalb wird dieser Zustand auch als *verbotener Zustand* bezeichnet.

Also ist der FF stabil und kann speichern, wenn beide Eingänge R und S (stehend für *Reset* und *Set*) auf logisch 1 sind.

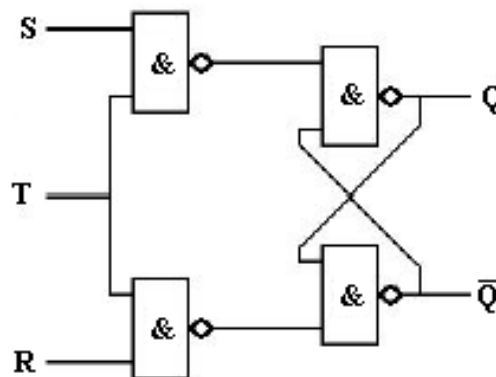
Entsprechend der Namen wird der Ausgang Q bei $R = 0$ resetet ($Q = 0$) und bei $S = 0$ gesetzt ($Q = 1$). Da \bar{Q} der inverse Zustand von Q ist, gilt für ihn das Umgekehrte.

Wir haben also folgende Funktionaltafel begründet:

S	R	Q	\bar{Q}	Funktion
1	1	Q	\bar{Q}	speichern
0	1	1	0	set
1	0	0	1	reset
0	0	1	1	verboten

4.2 Getaktetes RS-Flip-Flop

Ein getaktetes Flip-Flop (RST-FF) besteht aus einem ganz normalen Flip-Flop und davor geschalteten NAND-Gattern, welche dafür sorgen, dass der Flip-Flop erst reagiert, wenn die Taktung an T es erlaubt:

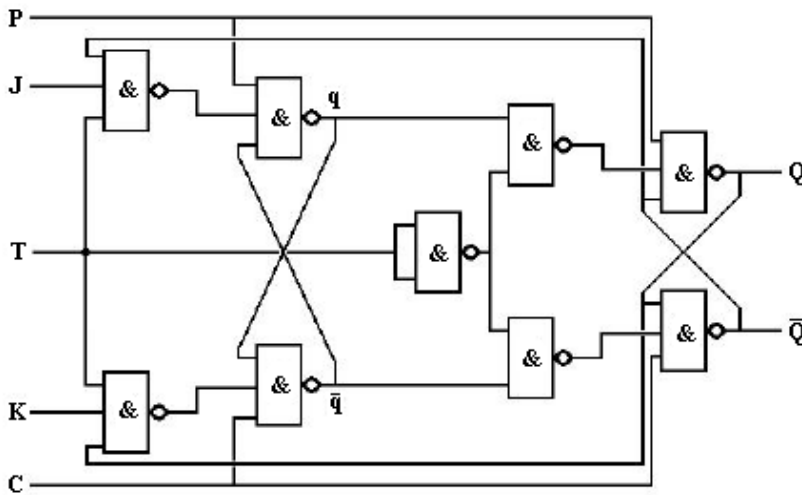


Wenn der Takt T auf logisch 0 liegt, dann sorgt die Ansteuerschaltung (s. Wahrheitstabelle für NAND), dass die Signale von R und S keinen Einfluss haben, auch wenn sie gestört sein sollten, oder ihren Wert änderten. Taktet man T auf 1, dann reagiert die Schaltung wie ein normales Flip-Flop und schaltet entsprechend den Zuständen von R und S . Es ergibt sich also folgende Wahrheitstabelle:

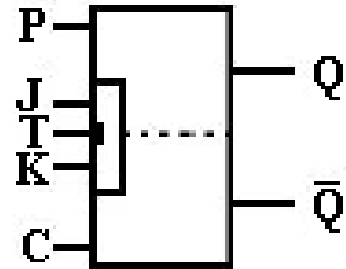
S	R	T	Q	\bar{Q}
0	0	0	Q	\bar{Q}
1	0	0	Q	\bar{Q}
0	1	0	Q	\bar{Q}
1	1	0	Q	\bar{Q}
0	0	1	Q	\bar{Q}
0	1	1	0	1
1	0	1	1	0
1	1	1	1	1

4.3 JK-Master-Slave-Flip-Flop

Prinzipiell besteht der JK-Master-Slave-Flip-Flop aus zwei RST-Flip-Flops:



Aufbau JK-M-S-FF



Schaltsymbol JK-M-S-FF

Das erste Flip-Flop (Master) dient als Zwischenspeicher. Beim Takt werden die Werte von R und S dort gespeichert, wobei das zweite Flip-Flop (Slave) diese nicht erhält. Erst bei einem weiteren Takt werden die Werte an das zweite Flip-Flop übermittelt und dort gespeichert. Der Vorteil des Master-Slave-Systems ist, dass der unbestimmte Zustand ausgeschlossen ist.

Wird der Eingang C (Clear) auf logisch 0 gesetzt, dann werden beim nächsten 1-0 Durchgang des Taktsignals die Speicher gelöscht. Es sollte sich folgendes ergeben:

Taktwechsel	J	K	q	\bar{q}	Q	\bar{Q}	Aktion
0-1	0	0	q	\bar{q}	Q	\bar{Q}	speichern
1-0	0	0	\bar{q}	q	q	\bar{q}	speichern
0-1	1	1	\bar{Q}	Q	Q	\bar{Q}	sperrern
1-0	1	1	q	\bar{q}	q	\bar{q}	sperrern
0-1	0	1	0	1	Q	\bar{Q}	set
1-0	0	1	q	\bar{q}	q	\bar{q}	set
0-1	1	0	1	0	Q	\bar{Q}	reset
1-0	1	0	q	\bar{q}	q	\bar{q}	reset

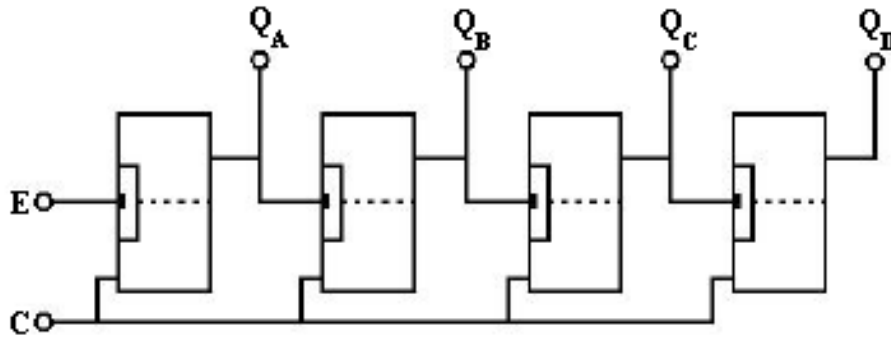
Wobei mit $C \vee P \neq 1$ folgt, dass das Ergebnis unabhängig von J, K, T ist:

P	C	Q	\bar{Q}
0	0	1	1
1	0	1	0
0	1	0	1

6 Zähler

6.1 4-Bit-Asynchrnzähler

Der Aufbau erfolgt gemäß der Vorbereitungsmappe:



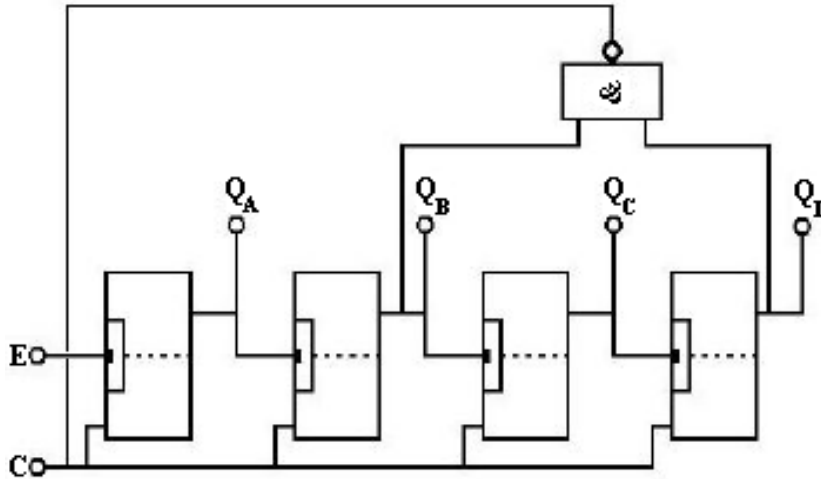
Es werden also 4 JK-M-S-FFs geschaltet. Der so entstandene 4-Bit-Asynchrnzähler kann insgesamt von 0 (Bitfolge 0000) bis 15 (Bitfolge 1111) zählen. Dies geschieht, indem man Impulse auf die Taktleitung gibt, wobei es einen Taktzyklus lang dauert, bis der Wert abgelesen werden kann. Der Zähler heißt asynchron, weil immer ein Taktzyklus gewartet werden muss, bis abgelesen werden kann (J, K frei, d.h. auf 1, deshalb kippen der Schaltung bei jedem Taktgang).

Folgende Tabelle veranschaulicht die Vorgänge:

Taktimpuls	Q_A	Q_B	Q_C	Q_D
0.	0	0	0	0
1.	1	0	0	0
2.	0	1	0	0
3.	1	1	0	0
4.	0	0	1	0
5.	1	0	1	0
6.	0	1	1	0
7.	1	1	1	0
8.	0	0	0	1
9.	1	0	0	1
10.	0	1	0	1
11.	1	1	0	1
12.	0	0	1	1
13.	1	0	1	1
14.	0	1	1	1
15.	1	1	1	1
16.	0	0	0	0

6.2 Asynchroner Dezimalzähler

Um einen Dezimalzähler zu erhalten, muss der Zähler nach der Zahl 9 (wegen des Zehnerübertrags, also dezimal 10, bzw. $Q_A Q_B Q_C Q_D = 0101$) wieder auf 0 (bzw. 0000) umschalten. Um das zu realisieren werden die Ausgänge von Q_B und Q_D mit einem NAND-Gatter, dessen Ausgang am Eingang C (Clear) angeschlossen. Sobald also $Q_B = Q_D = 1$ liefert das NAND-Gatter eine 0 (s. Wahrheitstabelle bei NAND), die einen reset des Zählers bewirkt. Die Schaltung wird folgendermaßen aufgebaut:

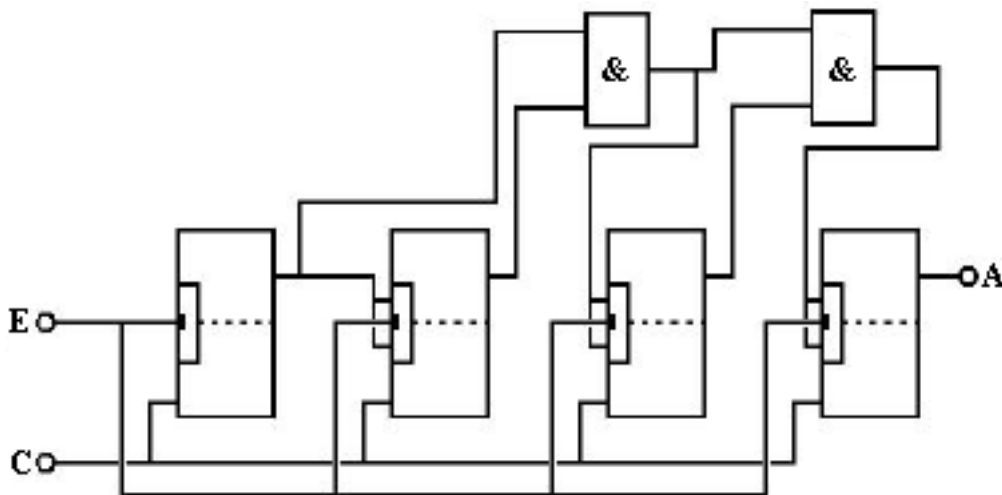


und die Wertetabelle ergibt sich wie beschrieben:

Taktimpuls	Q_A	Q_B	Q_C	Q_D
0.	0	0	0	0
1.	1	0	0	0
2.	0	1	0	0
3.	1	1	0	0
4.	0	0	1	0
5.	1	0	1	0
6.	0	1	1	0
7.	1	1	1	0
8.	0	0	0	1
9.	1	0	0	1
10.	0	0	0	0

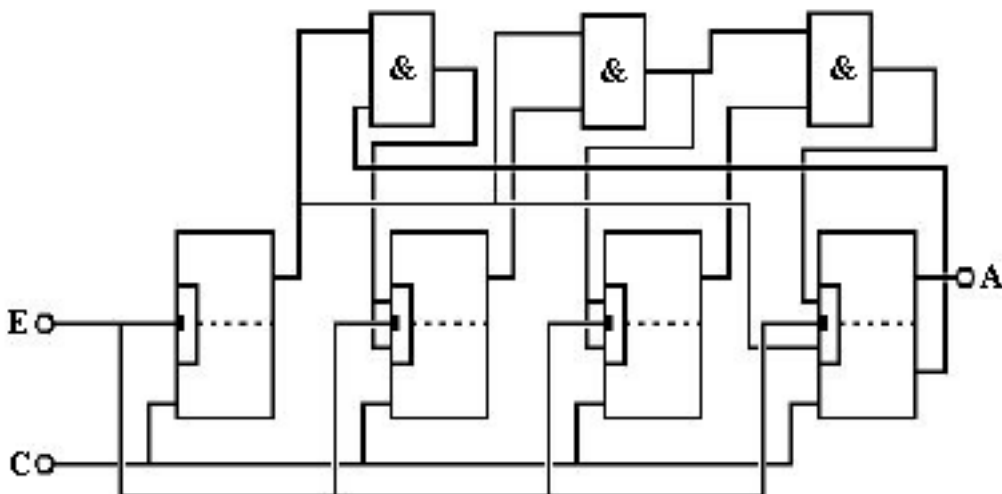
6.3 4-Bit-Synchronzähler

Beim synchronen Zähler benutzt man die Tatsache, dass ein JK-MS-FF genau dann bei einem Zählzyklus 0-1-0 kippt, wenn an den beiden Eingängen J und K eine 1 anliegt. Da alle Zählleitungen verbunden sind, kann das Umschalten nur gleichzeitig (beim anlegen des Taktsignals an E) erfolgen. Damit hängt das Umschalten nur vom Zustand der Eingänge J und K ab. Diese werden daher an die Ausgänge des vorherigen Moduls angeschlossen. Das dritte Modul darf aber nur kippen, wenn an den beiden Vorgängern eine 1 anliegt (011 wird zu 100), daher schliesst man die beiden Ausgänge der ersten beiden Module über ein AND-Gatter an den Eingang des dritten Moduls an. Analog verfährt man beim vierten Modul. Es darf nur kippen, wenn alle vorherigen Module eine 1 anliegen haben (d.h. man verbindet den Ausgang des dritten Moduls und den Ausgang des AND-Gatters von vorhin mit einem zweiten AND-Gatter und das wiederum mit dem Eingang des vierten Moduls. Die Wertetafel ist analog zu der des asynchronen Zählers. Der Aufbau erfolgt wie beschrieben:



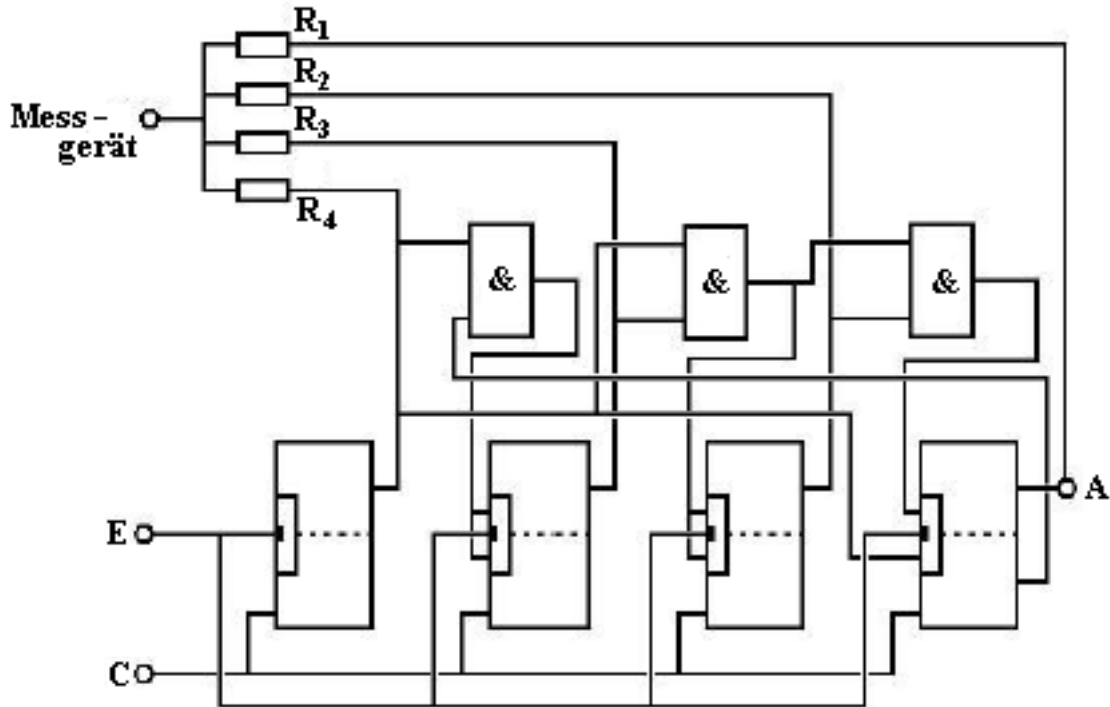
6.4 Synchroner Dezimalzähler

Hier soll wiederum der Zehnerübertrag realisiert werden, d.h. das Umschalten von 9 (1001) auf 0, was durch ein zusätzliches AND-Modul an der Schaltung von 6.3 erreicht wird. Die Wertetabelle ergibt sich analog zum asynchronen Zählmodus. Der Aufbau ist also klar:



7 Digital-Analog-Wandlung

Der Digital-Analog-Wandler wird realisiert, indem man an jedem Ausgang ein Signal abgreift und es über einen Widerstand (für jeden Ausgang anderen) von einem Messgerät messen lässt. Die Ströme dort addieren sich zum Gesamtstrom. Die Schaltung sieht so aus:



Es bleibt die Bestimmung der Widerstände. Die Schaltung soll so gebaut sein, dass jeder Zählschritt 10% Ausschlag des Zeigers bewirken. Es folgt:

Zahl dual	Zahl dezimal	benötigter Strom
0001	1	$10 \mu A$
0010	2	$20 \mu A$
0100	4	$40 \mu A$
1000	8	$80 \mu A$

Da der Maximale Strom $100 \mu A$ ist. Man wählt nun die Widerstände entsprechend (R_4 ist am Ausgang des ersten Moduls, zeigt also die Ziffer mit der Wertigkeit 1 an und muss daher auch den kleinsten Strom haben, d.h. größten Widerstand):

$$R_1 = \frac{U_0}{80 \mu A} = \frac{4V}{80 \mu A} = 50 k\Omega \quad (11)$$

$$R_2 = \frac{U_0}{40 \mu A} = \frac{4V}{40 \mu A} = 100 k\Omega \quad (12)$$

$$R_3 = \frac{U_0}{20 \mu A} = \frac{4V}{20 \mu A} = 200 k\Omega \quad (13)$$

$$R_4 = \frac{U_0}{10 \mu A} = \frac{4V}{10 \mu A} = 400 k\Omega \quad (14)$$

Somit haben wir den Digitalen-Analog-Wandler realisiert.