Physikalisches Anfängerpraktikum
P1

Versuch:

P1-63,64,65

Schaltlogik

Schriftliche Ausarbeitung von Georg Fleig

Gruppe: Di-11

Datum der Versuchsdurchführung:

29.11.2011

Einführung

In den folgenden Versuchen lernen wir einfache nicht programmierbare Schaltungen kennen. Dabei sollen wir deren Aufbau und Logik verstehen. Die Zustände 0 und 1, die im Weiteren angesprochen werden, entsprechen einer Spannung von $0V$ und $+5V$.

Aufgabe 1: Gatter aus diskreten Bauelementen

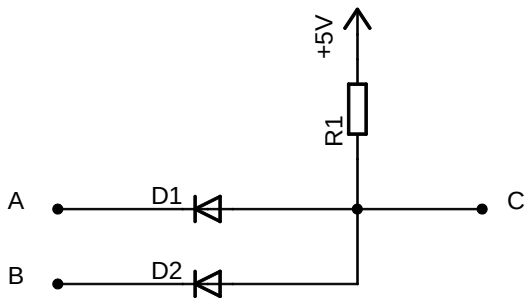
Es sollen einfache Grundschaltungen der Schaltlogik aufgebaut und untersucht werden. Diese lassen sich beispielsweise aus Dioden aufbauen und benötigen recht wenig Bauteile. Die Verkettung solcher Gatter führt jedoch wegen den Verlusten an den Dioden zu einem Verschwimmen der LOW (0) und der HIGH (1) Werte. Um das zu umgehen werden meistens CMOS- oder TTL-Gatter Verwendet.

Aufgabe 1.1 - AND-Gatter

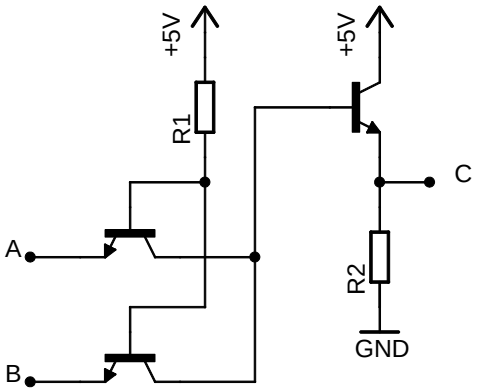
Das Logische AND ergibt nur dann eine 1, wenn beide Eingänge auf 1 liegen. In allen anderen Fällen wird eine 0 ausgegeben. Dies ergibt sich aus folgender Wahrheitstabelle:

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

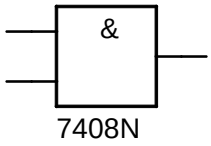
Die Schaltung wird entsprechend untenstehender Schaltskizze realisiert. Dabei Werden die Dioden so angeordnet, dass ihre Sperrrichtung in Richtung des Eingangs zeigt. Der Widerstand R_1 wird dabei so gewählt, dass er über dem Durchlasswiderstand der Dioden liegt, damit an ihm die gesamte Spannung abfällt. Nur wenn auf beiden Eingängen 1 liegt, fließt kein Strom und somit liegt am Ausgang C auch 1 an. Liegt entweder nur an einem Eingang oder an beiden Eingängen 0 an, fließt immer ein Strom durch eine bzw. beide Dioden ab und so liegt an C 0 an. Die Dioden verhindern einen Stromfluss zwischen den beiden Eingängen.



Als TTL-Gatter ließe sich die AND-Schaltung folgendermaßen realisieren:



Das AND-Gatter besitzt außerdem folgendes IC-Symbol:

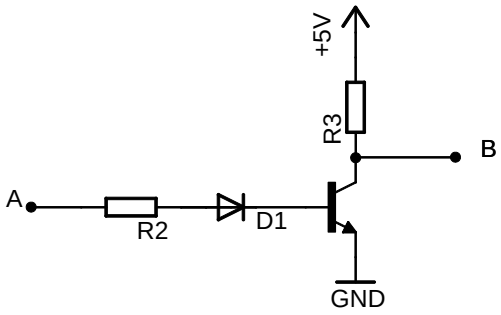


Aufgabe 1.2 - NOT- und NAND-Gatter

Es soll ein NOT-Gatter mittels einer Transistorschaltung realisiert werden und mit dem AND-Gatter zu einem NAND-Gatter verknüpft werden.
Die Wahrheitstabelle einer logischen NOT-Verknüpfung sieht folgendermaßen aus:

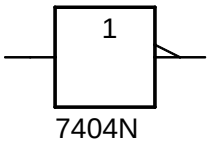
A	$B = \overline{A}$
0	1
1	0

Man erkennt leicht, dass das Eingangssignal immer negiert wird. Eine 1 wird also zur 0 und umgekehrt. Realisieren lässt sich ein solches Gatter entsprechend dieser Schaltskizze:



In diesem Schaltplan stellen R_3 und der Transistor einen Spannungsteiler dar. R_2 dient als Stabilisator und Begrenzung des Transistors, die Diode D_1 soll vernachlässigt werden.
Liegt der Eingang A auf 1, so ist die Kollektor-Emitter-Strecke des Transistors offen. Dabei fällt fast die gesamte Spannung an R_3 ab, da dieser Widerstand viel größer ist als der des Transistors in offenen Zustand. Am Ausgang B misst man daher ein Potential von $0V$.
Liegt A hingegen auf 0, so ist die Kollektor-Emitter-Strecke des Transistors geschlossen und er hat damit

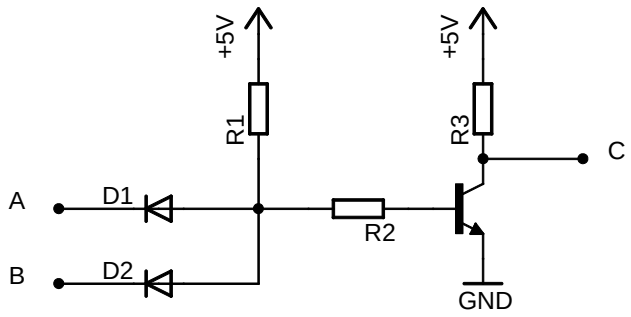
einen sehr großen Widerstand gegenüber R_3. Fast die gesamte Spannung fällt also am Transistor ab und der Ausgang B liegt daher auf 1.
Das IC-Symbol des NOT-Gatters ist:



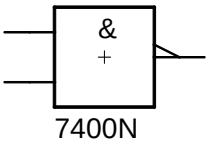
NAND-Gatter
Durch Anschließen des NOT-Gatters an den Ausgangs des AND-Gatters erhält man ein NAND-Gatter mit folgender Wahrheitstabelle:

A	B	$C = \overline{A \wedge B}$
0	0	1
0	1	1
1	0	1
1	1	0

Es ist leicht zu erkennen, dass die Ausgabe des AND-Gatters negiert wird.
Die Schaltung sieht dann folgendermaßen aus:



Mit dem zugehörigen IC-Symbol:

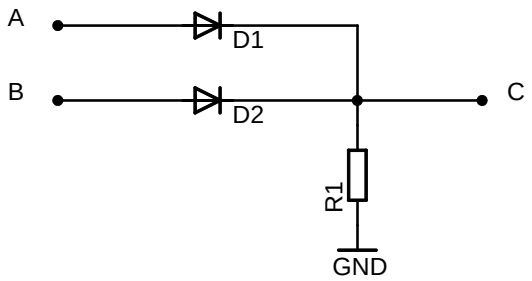


Aufgabe 1.3 - OR-Gatter

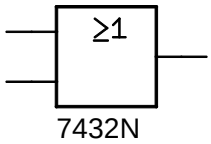
Die logische OR-Verknüpfung ergibt dann 1, wenn mindestens einer der beiden Eingänge auf 1 liegt.

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

Ein OR-Gatter wird folgendermaßen realisiert:



R_1 wir hier wieder so gewählt, dass er sehr viel größer als der Durchlasswiderstand der Dioden ist. Die Kombination der Dioden und des Widerstandes wird als Spannungsteiler angesehen. Liegt nun einer der beiden Eingänge oder beide auf 1, so fließt ein Strom durch die Dioden. Da R_3 ziemlich groß ist, fällt an ihm ein Großteil der Spannung ab und damit liegt der Ausgang C auf 1. Durch die Dioden wird hier verhindert, dass Strom vom einen Eingang zum anderen fließen kann, falls nur einer der beiden auf 1 liegt.
Liegt weder an A, noch an B eine Spannung an, so fließt logischerweise überhaupt kein Strom und C liegt auf 0.
Eine Kombination mehrere OR-Gatter ist nicht sinnvoll, da die Dioden einen Durchlasswiderstand haben und so die HIGH-Schwelle am Ausgang immer weiter absinkt und sich kaum mehr von einer LOW-Schwelle unterscheiden lässt. Das zugehörige IC-Symbol des OR-Gatters ist:



Aufgabe 2: Integrated Circuit Gatter

Die nachfolgenden Schaltungen werden mit sogenannten IC-Bausteinen realisiert. Sie vereinfachen den Aufbau sowie die Schaltskizzen.

Aufgabe 2.1 - NOT-Gatter

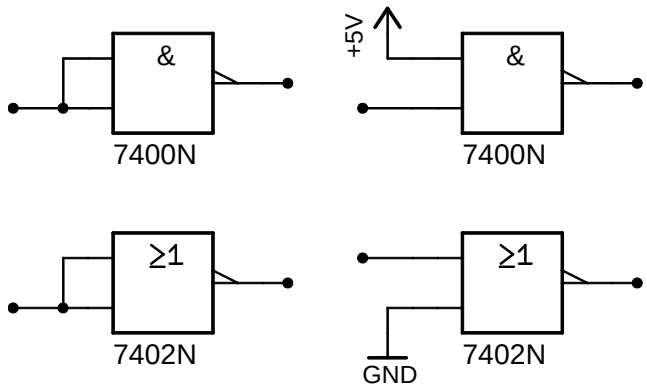
Es soll ein NOT-Gatter aus einem NAND-Gatter oder einem NOR-Gatter erzeugt werden. Durch Betrachten der beiden Wahrheitstabellen

	A	B	$C = \overline{A \wedge B}$
$\star$	0	0	1
	0	1	1
	1	0	1
$\star$	1	1	0

	A	B	$C = \overline{A \vee B}$
$\star$	0	0	1
	0	1	0
	1	0	0
$\star$	1	1	0

erkennt man direkt (entsprechende Zeilen mit $\star$ markiert), dass bei beiden Gattern jeweils die Eingänge verbunden werden müssen, um ein negiertes Signal und damit ein NOT-Gatter zu erhalten. Eine Alternative dazu wäre beim NAND-Gatter einen Eingang konstannt auf 1 zu legen (entspricht z.B. Zeile 3

und 4 für Eingang A) oder beim NOR-Gatter einen der Eingänge konstant auf 0 zu legen (entspricht z.B. Zeile 1 und 2 für Eingang A). Auch so erhält man ein negiertes Signal. Hier sind die entsprechenden Schaltungen mit den ICs dargestellt:



Aufgabe 2.2 - XOR-Gatter

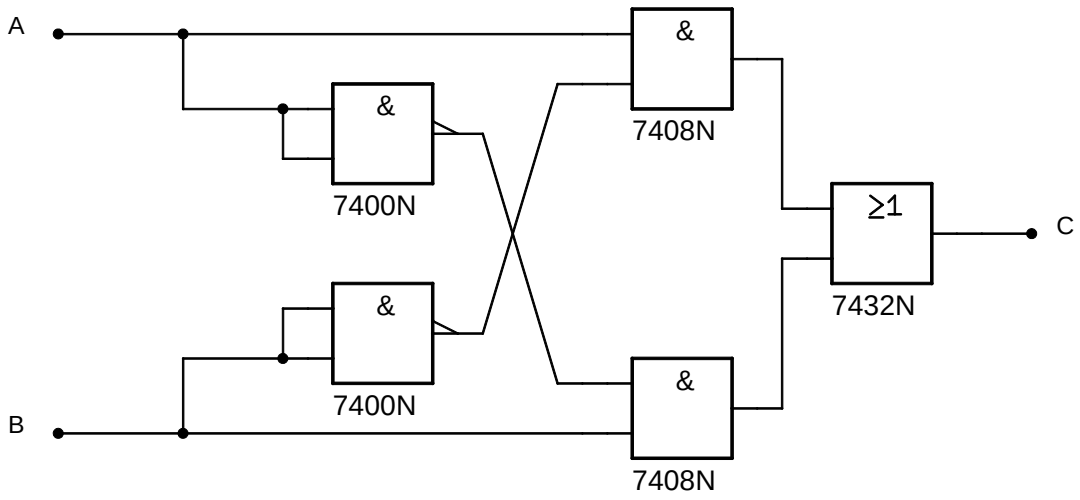
Die XOR-Verknüpfung gibt nur 1, wenn genau einer der beiden Eingänge auf 1 liegt. Dies entspricht folgender Wahrheitstabelle:

A	B	$C = A \underline{\vee} B$
0	0	0
0	1	1
1	0	1
1	1	0

Für C ergibt sich daher:

$$C = (\overline{A} \wedge B) \vee (A \wedge \overline{B})$$

Hieraus lesen wir ab, dass zur Realisierung des XOR-Gatters ein OR-, zwei AND- und zwei NOT-Gatter benötigt werden. Dabei werden die NOT-Gatter entsprechend wie in der vorigen Aufgabe dargestellt durch NAND-Gatter erzeugt. Es ergibt sich folgender Schaltplan:

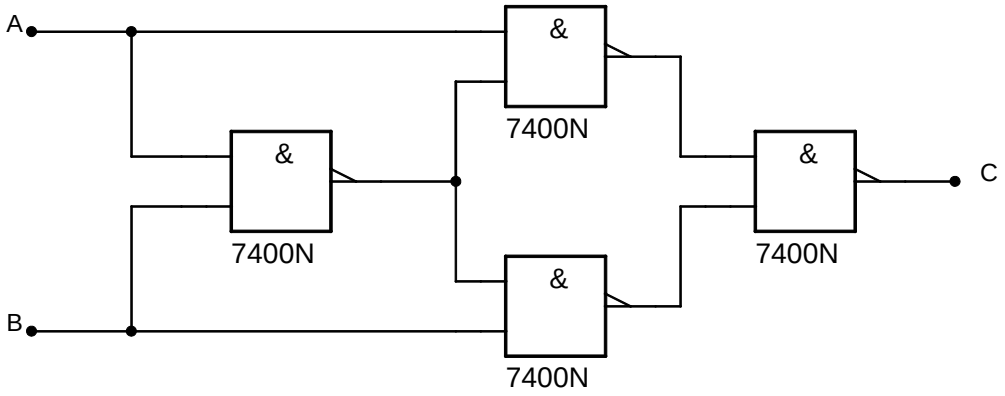


Aufgabe 2.3 - XOR aus NAND-Gattern

Bei dieser Aufgabe soll nun ein XOR-Gatter ausschließlich aus NAND-Gattern erzeugt werden. Dazu wird die disjunkte Normalform aus Aufgabe 2.2 mehrfach umgeformt:

$$\begin{aligned}
C &= \left(A \wedge \overline{B}\right) \vee \left(\overline{A} \wedge B\right) \\
&= \left(A \wedge \overline{B}\right) \vee \left(A \wedge \overline{A}\right) \vee \left(\overline{A} \wedge B\right) \vee \left(B \wedge \overline{B}\right) \\
&= A \wedge \left(\overline{A} \vee \overline{B}\right) \vee B \wedge \left(\overline{A} \vee \overline{B}\right) \\
&= A \wedge \left(\overline{\overline{\overline{A} \vee \overline{B}}}\right) \vee B \wedge \left(\overline{\overline{\overline{A} \vee \overline{B}}}\right) \\
&= A \wedge \left(\overline{A \wedge B}\right) \vee B \wedge \left(\overline{A \wedge B}\right) \\
&= \overline{\overline{A \wedge \left(\overline{A \wedge B}\right) \vee B \wedge \left(\overline{A \wedge B}\right)}} \\
&= \overline{\overline{A \wedge \left(\overline{A \wedge B}\right)} \wedge \overline{B \wedge \left(\overline{A \wedge B}\right)}}
\end{aligned}$$

Es lässt sich also aus vier NAND-Gattern ein XOR-Gatter erzeugen:



Der Vorteil dieser XOR-Gatter Realisierung gegenüber der aus Aufgabe 2.2 liegt darin, dass her nur Bauteile desselben Typs gebraucht werden. In der Praxis können so Produktionskosten eingespart werden.

Aufgabe 3: Addierer

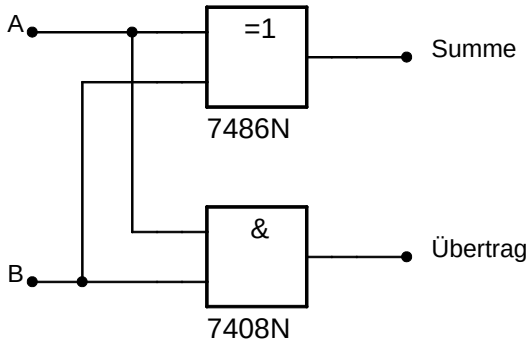
In den folgenden Aufgabenteilen sollen verschiedene Gatter-Typen so kombiniert werden, dass mit ihnen einfache Addition und Subtraktion von Dualzahlen durchgeführt werden können.

Aufgabe 3.1 - Halbaddierer

Mit einem Halbaddierer lassen sich zwei einstellige Dualzahlen addieren. Zunächst wird eine Wahrheitstabelle mit Summe und Übertrag aufgestellt:

A	B	S	Ü
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Hieraus lässt sich direkt erkennen, dass für die Summe ein XOR-Gatter und für den Übertrag ein AND-Gatter benötigt wird.



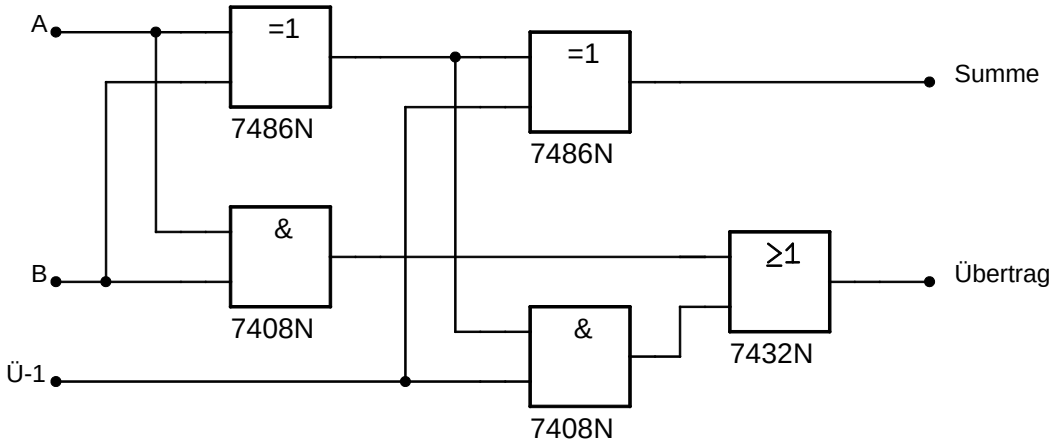
Der Einfachheit halber wird hier direkt ein fertiger XOR-Baustein anstelle der komplizierteren Schaltungen aus Aufgabe 2 verwendet.

Aufgabe 3.2 - Volladdierer

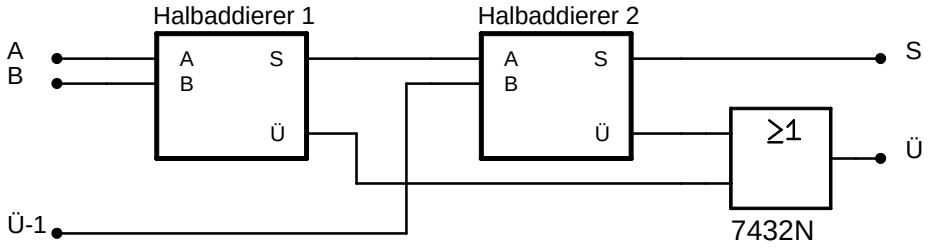
Es soll ein 1-Bit-Volladdierer realisiert werden, welcher im Gegensatz zum Halbaddierer auch den Übertrag aus der vorigen Stufe berücksichtigt. Durch geschickte Kombination zweier Halbaddierer erhält man einen Volladdierer, der drei Dualzahlen addieren kann. Es ergibt sich folgende Wahrheitstabelle:

A	B	C	S	Ü
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

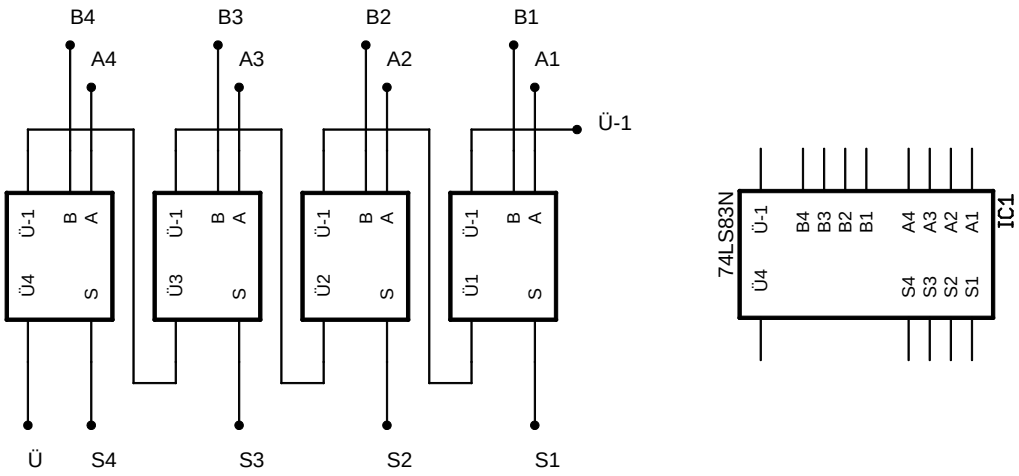
A und B sind dabei normale Eingänge und C liefert den Übertrag Ü_{-1} aus der vorangegangenen Stufe. Einen Volladdierer erhält man, indem ein Halbaddierer A und B addiert und die Summe S von diesem im zweiten Halbaddierer auf den Übertrag der vorigen Spalte Ü_{-1} bzw. auf eine weitere Zahl C addiert wird. Die Summe daraus ist die Gesamtsumme, den Übertrag erhält man durch ein OR-Gatter, das an die Überträge der beiden Halbaddierer angeschlossen wird. Dargestellt ist dies in folgender Schaltung:



Oder etwas vereinfacht dargestellt:

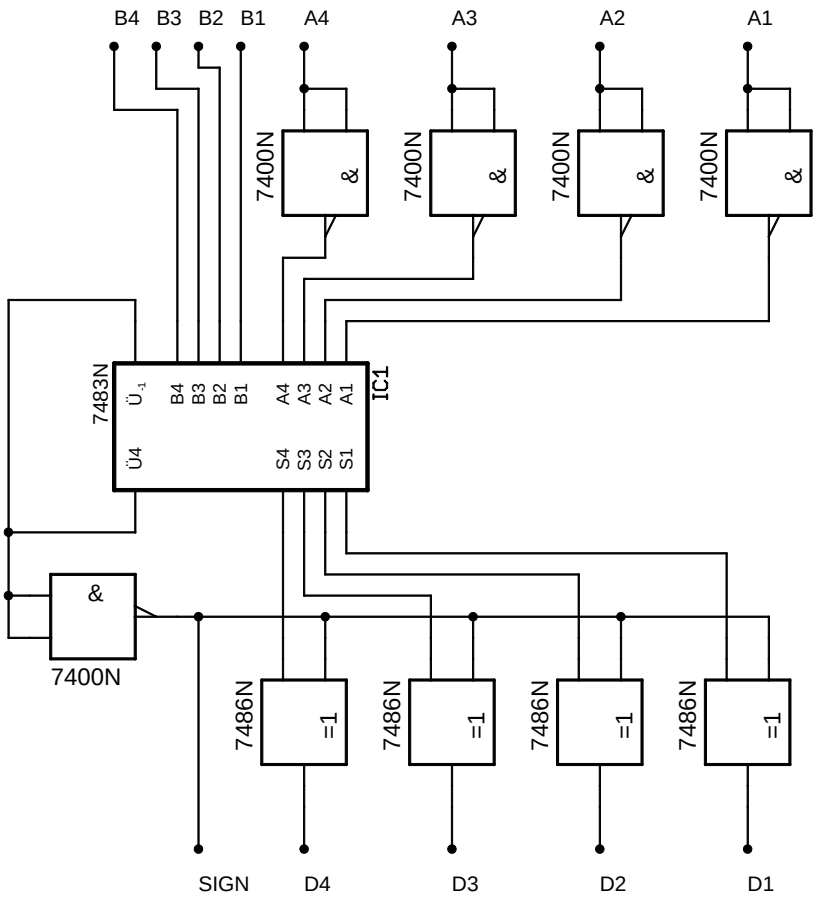


Schließt man mehrere solcher Volladdierer zusammen, können größere Zahlen miteinander addiert werden. Hier ist beispielsweise ein 4-Bit-Addierer und sein zugehöriges IC-Symbol dargestellt:



Aufgabe 3.3 - Subtrahierer

Es soll ein Subtrahierer gebaut werden, welcher mit 4-Bit-Zahlen rechnen kann. Dazu benötigen wir einen 4-Bit-Addierer in Kombination mit 5 NOT-Gattern und 4 XOR-Gattern. Der Subtrahierer soll entsprechend der Schaltskizze aufgebaut werden:



Der Subtrahierer rechnet folgendermaßen: Die abzuziehende Zahl A wird durch die zwischengeschalteten NOT-Gatter zu $\overline{A}$ invertiert, dabei ergibt sich bei einer 4-Bit Zahl:

$$\overline{A} = 15 - A$$

und somit für die Summe:

$$\begin{aligned} S &= B + \overline{A} + \ddot{\mathrm{U}}_{-1} \\ &= B - A + 15 + \ddot{\mathrm{U}}_{-1} \end{aligned}$$

Die Zahl B wird direkt an den 4-Bit-Addierer als Binärzahl angelegt. Vor das tatsächliche Ausgangssignal sind XOR-Gatter geschaltet . Der Übertrag $\ddot{\mathrm{U}}_{-1}$ ist mit dem letzten Übertrag $\ddot{\mathrm{U}}_4$ verbunden, um ihn in der Rechnung mit zu berücksichtigen.
Es gilt nun drei Fälle zu unterscheiden:

1. Fall: $B > A$
   Hier entsteht ein Übertrag von 1 bei $\ddot{\mathrm{U}}_4$, da die Summe der Zahl sich nicht mit 4 Bit darstellen lässt. Das Vorzeichen an SIGN ist wegen des vorgeschalteten NOT-Gatters negiert und somit 0, was einem + entspricht. Die vor die Ausgabebits geschalteten XOR-Gatter ändern ihren Wert nicht, da an einem Eingang nun immer 0 anliegt. Zahlen größer als 15 (größte Zahl, die sich mit 4 Bits

darstellen lässt) werden also abgeschnitten. Für S ergibt sich:

$$\begin{aligned} S &= B + \overline{A} + \ddot{\mathrm{U}}_{-1} \\ &= B - A + 15 + 1 \\ &= B - A + 16 \end{aligned}$$

Das "Abschneiden" der Zahl ergibt:

$$\begin{aligned} D &= S \bmod 16 \\ D &= S - 16 \\ &= (B - A + 16) - 16 \\ &= B - A \end{aligned}$$

Das entspricht genau unserem gewünschten Ergebnis.

2. Fall: $B < A$
   Hier ist das Ergebnis mit 4 Bits direkt darstellbar. Der Übertrag an $\ddot{\mathrm{U}}_4$ ist 0, die Negation 1 und entspricht damit einem negativen Vorzeichen (-). Außerdem werden alle Ausgabebits invertiert, was zu dem Ergebnis führt:

$$\begin{aligned} D &= (-1) \cdot \overline{S} \\ &= (-1) \cdot (15 - S) \\ &= (-1) \cdot (15 - (B - A + 15)) \\ &= B - A \end{aligned}$$

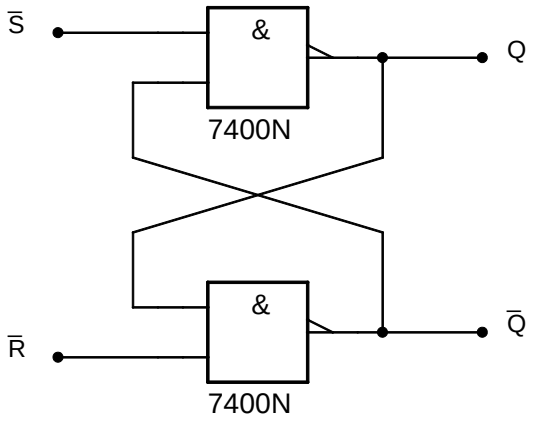
3. Fall: $B = A$
   Das Ergebnis hängt davon ab, ob $\ddot{\mathrm{U}}_{-1}$ 0 oder 1 gewesen ist. Es kommt in beiden Fällen 0 als Gesamtergebnis raus, jedoch ein mal mit negativem Vorzeichen. Da letzteres mathematisch keinen Sinn ergibt, muss das Vorzeichen vernachlässigt werden.

Aufgabe 4: Speicherelemente

Flip-Flops sind statische Speicher, die einen binären Zustand (1 oder 0) speichern können. In den folgenden Versuchen soll die Funktionsweise solcher Speicher genauer untersucht werden.

Aufgabe 4.1 - Reset-Set-Flip-Flop (RS-FF)

Das RS-FF ist das einfachste Flip-Flop mit einem Set und einem Reset Eingang. Es wird wie im Schaltplan zu sehen ist aus zwei NAND-Gattern aufgebaut:



Die Eingänge R und S werden negiert um sie später direkt mit dem getakteten RS-Flip-Flop vergleichen zu können. Außerdem ist zu beachten, dass der Ausgang des NAND-Gatters jeweils mit dem Eingang des anderen NAND-Gatters verbunden ist. Es ergibt sich folgende Wahrheitstabelle:

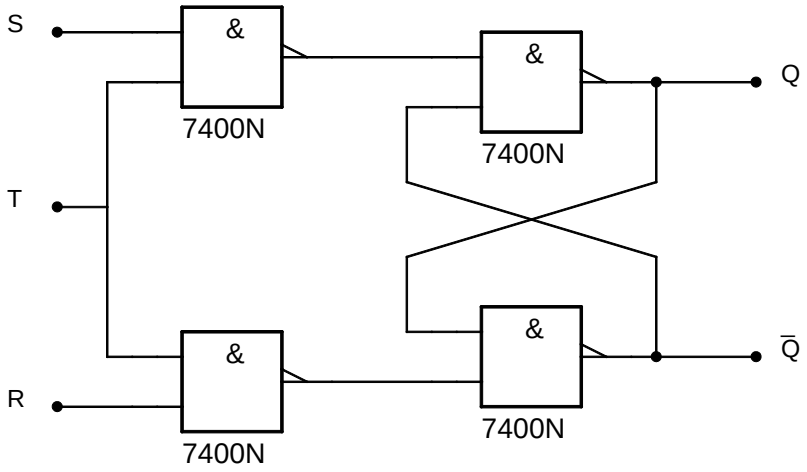
S	R	Q_n	$\overline{Q}_n$	
0	0	Q_{n-1}	$\overline{Q}_{n-1}$	keine Änderung (speichern)
0	1	0	1	setze 0
1	0	1	0	setze 1
1	1	1	1	$Q = \overline{Q}$: verboten, falsche Aussage

Aufgabe 4.2 - Getaktetes RS-Flip-Flop (RST-FF)

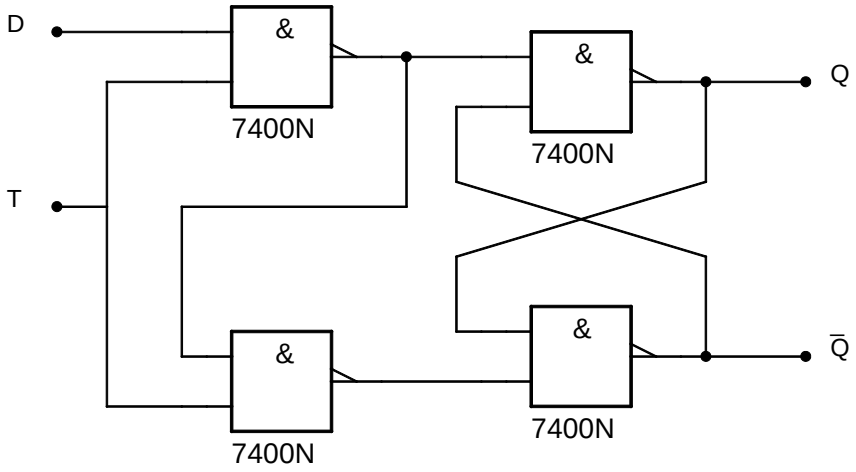
Ein RST-FF arbeitet prinzipiell wie ein RS-FF nur mit dem Unterschied, dass es einen zusätzlichen Takteingang gibt, der Zustandsänderungen des Flip-Flops nur bei einem anliegenden Takt von 1 erlaubt. Realisiert wird das durch eine jeweilige Verknüpfung der S und R Eingänge mit dem Takteingang T über ein NAND-Gatter. Da das RS-FF bereits mit negierten Eingängen beschrieben wurde, sind die Wahrheitstabellen dieser beiden Flip-Flops identisch, bis auf den zusätzlichen Takteingang:

T	S	R	Q_{n+1}	$\overline{Q}_{n+1}$	
0	0	0	Q_n	$\overline{Q}_n$	kein Takt $\Rightarrow$ keine Zustandsänderung
0	0	1	Q_n	$\overline{Q}_n$	kein Takt $\Rightarrow$ keine Zustandsänderung
0	1	0	Q_n	$\overline{Q}_n$	kein Takt $\Rightarrow$ keine Zustandsänderung
0	1	1	Q_n	$\overline{Q}_n$	kein Takt $\Rightarrow$ keine Zustandsänderung
1	0	0	Q_n	$\overline{Q}_n$	keine Änderung (speichern)
1	0	1	0	1	setze 0
1	1	0	1	0	setze 1
1	1	1	1	1	$Q = \overline{Q}$: verboten, falsche Aussage

Die Schaltskizze sieht folgendermaßen aus:



Da auch hier wieder der "verbotene Zustand" $Q = \overline{Q}$ erzeugt werden kann, soll nun eine Möglichkeit gefunden werden, dies zu verhindern. Aus der Wahrheitstabelle lässt sich entnehmen, dass S und R nie beide den Wert 1 haben dürfen. Realisieren kann man dies, indem R als die Negation von S gewählt wird, also der Eingang R hinter das NAND-Gatter von S geschaltet wird. Das sähe als Schaltplan dann so aus:



Man hat nun nur noch einen Eingang D um Informationen anzulegen. für $D = 1$ und $T = 1$ wird also 1 gespeichert, entsprechend für $D = 0$ und $T = 1$ eine 0. Wird der Takt auf 0 gelegt, ändert sich unabhängig von Eingang D nichts. Dieses Flip-Flop wird auch als Data-Flip-Flop (D-FF) bezeichnet.

Aufgabe 4.3 - Jump-Kill-Master-Slave-Flip-Flop (JK-MS-FF)

Beim JK-MS-FF handelt es sich um ein taktflankengesteuertes Flip-Flop. Daher soll hier kurz auf den Unterschied zwischen diesen und taktzustandgesteuerten Flip-Flops eingegangen werden.

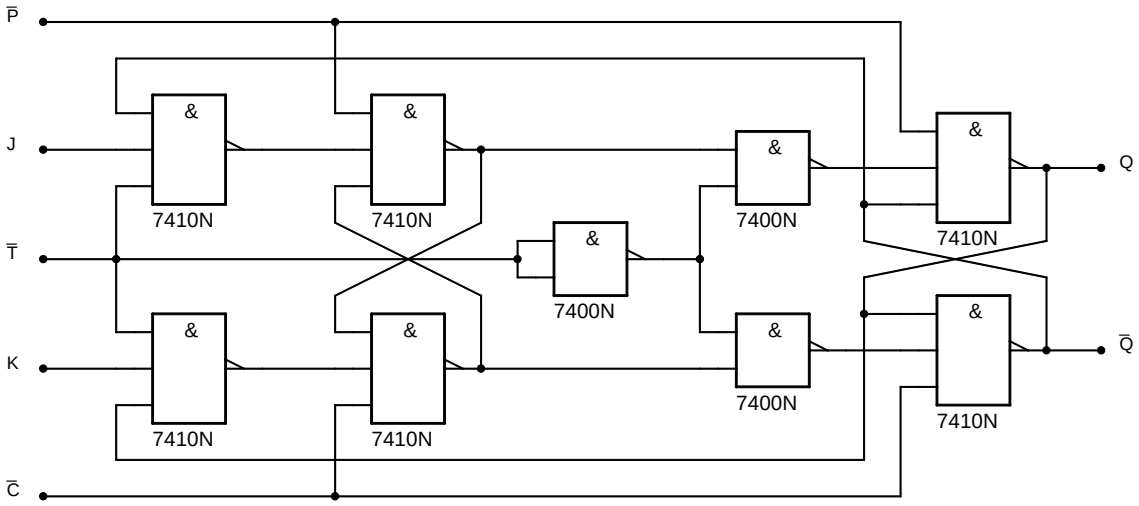
Taktzustandgesteuerte Flip-Flops

Diese Flip-Flops (z.B. RST-FF, D-FF) erlauben Zustandsänderungen über den gesamten Zeitraum in dem der Takt auf 1 liegt. Es können also mehrfach Änderungen an den Eingängen vorgenommen werden, die immer direkt an die Ausgänge weitergegeben werden.

Taktflankengesteuerte Flip-Flops

Solche Flip-Flops (wie z.B. das JK-MS-FF) erlauben Zustandsänderungen nur gerade in einem Zeitpunkt, wenn ein Taktwechsel vorliegt. Es wird dann genau das Signal der Eingänge verwendet, das zu diesem Zeitpunkt anliegt. Änderungen an den Eingängen während eines anliegenden Taktes von z.B. 1 werden nicht weiter verarbeitet. Dies wird im Folgenden genauer erläutert.

Beim zu untersuchenden JK-MS-FF wird der Taktübergang von 0 zu 1 als positive Taktflanke (+) bezeichnet, das Umgekehrte entsprechend als negative Taktflanke (-). Das Flip-Flop besteht aus einer Hintereinanderschaltung zweier RST-FFs, wobei die Taktleitung zwischen diesen beiden FFs durch ein NAND-Gatter negiert wird. Daher ist immer nur eines der beiden RST-FFs durch den Takt "aktiviert". Das vordere wird als Master und das hintere als Slave bezeichnet. Das JK-MS-FF wird entsprechend untenstehendem Schaltplan aufgebaut:



Die Eingänge $\overline{P}$reset und $\overline{C}$lear sollen während dem Versuch auf 1 gelegt werden, um nicht direkt das Ergebnis an Q und $\overline{Q}$ ohne Takt festzulegen. Dies kann man erreichen, indem sie einfach nicht angeschlossen werden und damit auf logisch 1 liegen.

Der JK-MS-FF arbeitet nun folgendermaßen:
Liegt T auf 0 und damit $\overline{T}$ auf 1 ist nur der Master durch den Takt aktiviert und verarbeitet die Informationen, die an den Eingängen J und K anliegen, und speichert sie entsprechend wie ein einzelner RST-FF. Wird nun der Takt auf 1 gelegt, ist der Master deaktiviert und der Slave wird aktiviert. Als Eingang liegen an ihm die Ausgänge Q_{Master} und $\overline{Q_{Master}}$ an. Er ließt quasi die Information des Masters ein und verarbeitet sie weiter, bis sie am Ende an Q und $\overline{Q}$ abgerufen werden können.
Um also Informationen an den Eingängen J und K komplett zu verarbeiten, wird zuerst der Taktzustand 0 und danach 1 benötigt. Dabei ist auch die Reihenfolge wichtig, da zuerst der Master und danach der Slave aktiviert werden muss. Nun ist auch klar, warum das alleinige vorgeben eines Taktes nicht zum Verarbeiten der Information ausreicht, sondern eine positive Taktflanke benötigt wird. Man spricht daher auch von einem positiv taktflankengesteuertem Flip-Flop
Der "verbotene Zustand" wird dadurch umgangen, dass die Ausgänge des Slave-FF gekreuzt in die

Eingänge des Masters rückgekoppelt werden.

Die zugehörige Wahrheitstabelle ist im Folgenden dargestellt:

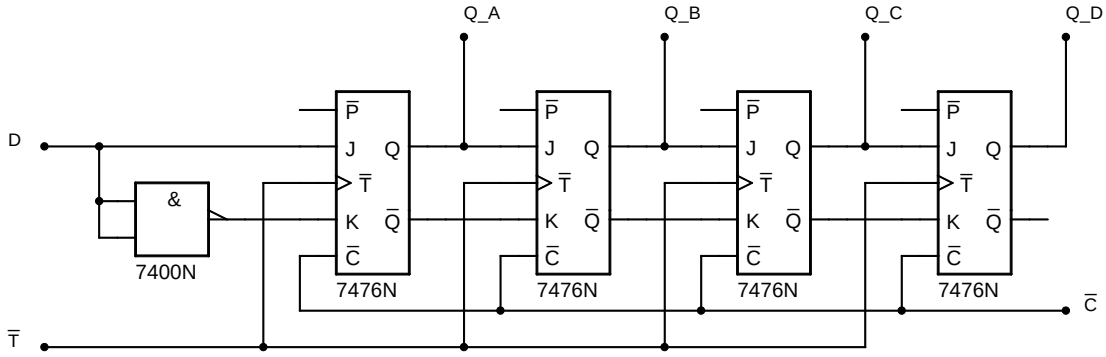
J	K	Taktflanke	Q_{n+1}	$\overline{Q}_{n+1}$	
0	0	$+$	Q_n	$\overline{Q_n}$	keine Änderung (speichern)
0	0	$-$	Q_n	$\overline{Q_n}$	falsche Taktflanke $\Rightarrow$ keine Zustandsänderung
0	1	$+$	0	1	setze 0
0	1	$-$	Q_n	$\overline{Q_n}$	falsche Taktflanke $\Rightarrow$ keine Zustandsänderung
1	0	$+$	1	0	setze 1
1	0	$-$	Q_n	$\overline{Q_n}$	falsche Taktflanke $\Rightarrow$ keine Zustandsänderung
1	1	$+$	$\overline{Q}_n$	Q_n	toggeln
1	1	$-$	Q_n	$\overline{Q_n}$	falsche Taktflanke $\Rightarrow$ keine Zustandsänderung

Aufgabe 6: Schieben, Multiplizieren und Rotieren

Um große Datenmengen zu verarbeiten, werden diese normalerweise in Blöcke zerlegt. Diese können dann entweder parallel über verschiedene Kanäle oder seriell, also nacheinander über einen Kanal, verarbeitet werden. Zum Wechseln zwischen den beiden Verfahren gibt es Parallel-Seriell-Wandler und Seriell-Parallel-Wandler.

Aufgabe 5.1 - 4-Bit-Schieberegister

Hier soll ein Seriell-Parallel-Wandler betrachtet werden, welcher seriell Daten aufnimmt und diese parallel an LEDs anzeigt. Realisiert wird dieser, indem vier JK-MS-FFs in Reihe geschaltet werden, wobei immer J_m mit Q_{m-1} und K_m mit $\overline{Q}_{m-1}$ verbunden wird. Die Schaltskizze sieht folgendermaßen aus:



Um die Signale einfacher einzuspeisen, wird der Kill Eingang des ersten Flip-Flops mit dessen Jump Eingang verbunden, ähnlich wie beim D-FF. Nun kann mittels eines entprellten Schalters oder des Taktgebers des Boards ein Eingangssignal generiert werden. Bei jeder positiven Taktflanke werden die gespeicherten Informationen einen JK-MM-FF weiter geschoben und können parallel an den LEDs beobachtet werden.

Dieser Schieber kann zudem zum Multiplizieren mit 2 oder Dividieren durch 2 benutzt werden. Dabei werden die Informationen entweder nach links oder nach rechts (von der Schaltskizze aus betrachtet)

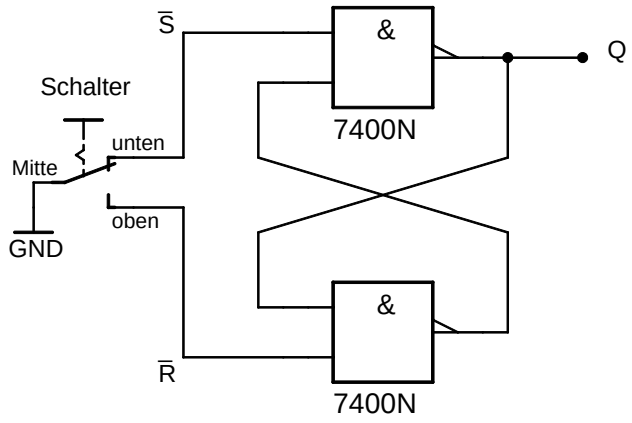
verschoben. Bei uns ist nur das Verschieben nach rechts möglich, also eine Division.

Entprellen des Eingangssignals

Beim Verwenden eines Schalters als Eingabeinstrument kann es Prellen kommen, wenn sich der hergestellte Kontakt direkt nach dem Umschalten kurzzeitig wieder löst ("prellt"). Daher kann es bei taktflankengesteuerten Schaltungen fälschlicherweise zur Registrierung unerwünschter Taktflanken kommen, weil jedes Abprallen als eine solche erkannt werden kann. Es kann ein RS-FF zwischengeschaltet werden um die korrekte Funktionsweise des Schiebers zu gewährleisten. Dabei wird der mittlere Ausgang des Schalters auf 0 gelegt, der obere mit $\overline{R}$ und der untere mit $\overline{S}$ des RS-FF verbunden. Der Ausgang Q des RS-FF gibt nun ein entprelltes Signal aus, dass an den Eingang D des 4-Bit-Schiebers angeschlossen werden kann.

Prellt der Schalter nun, so liegen für einen kurzen Moment beide Anschlüsse auf 1, weil sie offen sind. Wie in Aufgabe 4.1 dargestellt wurde, wird dadurch der gespeicherte Zustand des RS-FFs nicht verändert. Das Signal ist also entprellt. Für einen Wechsel des Signals wäre ein komplettes umkippen des Schalters nötig.

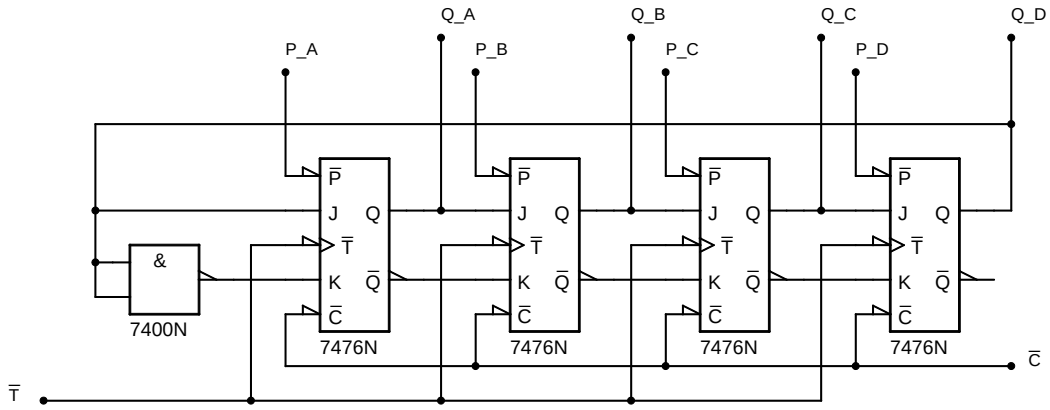
Eine solche Entprellschaltung ist hier dargestellt:



Aufgabe 5.2 - 4-Bit-Rotationsregister

Durch Verbinden des Ausgangs Q_D des letzten JK-MS-FFs mit dem Eingang D des ersten, erhält man ein Rotationsregister. Bei einer positiven Taktflanke werden die Informationen wie vorher auch eine Position nach rechts verrückt, die im letzten JK-MS-FF wird allerdings an den ersten übergeben. So wandern die Informationen quasi im Kreis. Verwendung findet dieses Register z.B. bei Lauflichtern.

Es soll entsprechend diesem Schaltplan aufgebaut werden:



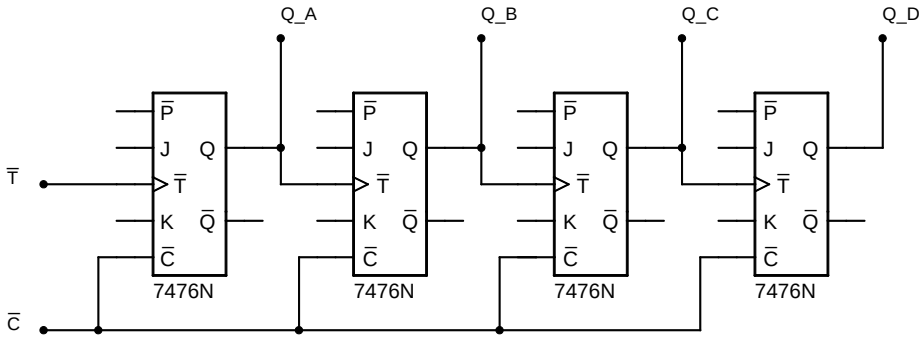
Über die Preset-Eingänge lassen sich nun parallel Werte für Q_A bis Q_D setzen.

Aufgabe 6: Zähler

Elektronische Zähler werden zur Zählung von Ereignissen, die entweder statistischer oder periodischer Natur sind, verwendet. In den folgenden Versuchen sollen verschiedene Zähler aus den IC-Bausteinen realisiert werden.

Aufgabe 6.1 - 4-Bit-Asynchronzähler

Es soll ein 4-Bit-Asynchronzähler entsprechend dieser Schaltskizze aufgebaut werden:



Wie zu sehen ist, besteht dieser aus vier JK-MS-FFs. Dabei ist immer der Ausgang Q am Takt-Eingang $\overline{T}$ des nächsten JK-MS-FF angeschlossen. Die J und K Eingänge sind offen und liegen damit auf 1. Über den $\overline{C}$ Eingang kann der Zähler auf 0000 zurückgesetzt werden.

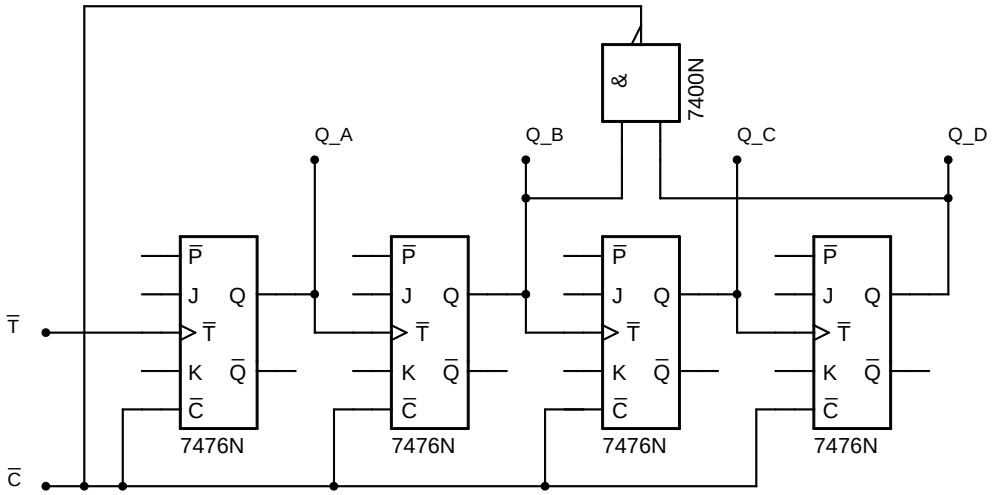
Folgende Tabelle stellt die zu zählenden Dezimalzahlen als Binärzahlen dar:

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

Warum der Zähler als asynchroner Zähler bezeichnet wird, soll folgendes Beispiel verdeutlichen: Betrachtet man das erste FF, schaltet dieses bei einer Taktflanke $\overline{T}$ $1 \rightarrow 0$. Am Ausgang Q_A liegt nun 1 an, dies entspricht einer Taktflanke $\overline{T}$ $0 \rightarrow 1$ für das folgende FF. Dieses zweite FF reagiert aber auch nur auf Taktflanken von $\overline{T}$ $1 \rightarrow 0$. Daher schaltet es erst wieder, wenn das erste FF zum zweiten Mal umschaltet. Der Takt des zweiten FF ist also gegenüber dem des ersten FF halbiert. Führt man dies für alle weiteren FFs fort, erkennt man, dass jedes FF immer mit dem halben Takt des vorigen FFs arbeitet. Daher nennt man diese Schaltung einen asynchronen Zähler. Das Umschalten des Zählers ist deswegen bei der Verwendung vieler Stufen zeitverzögert und damit unpraktikabel.

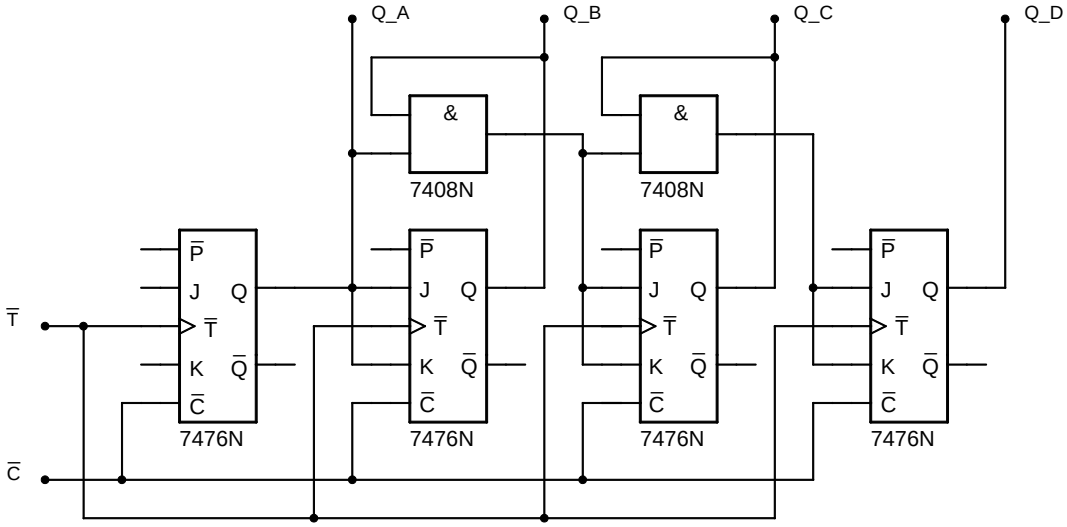
Aufgabe 6.2 - Asynchroner Dezimalzähler

Es soll ein asynchroner Dezimalzähler aufgebaut werden, der von 0 bis 9 zählt und bei der 10 auf 0 zurückgesetzt wird. Es wird also ein 4-Bit-Asynchronzähler benötigt, da man mit einer 3 Bit Anzeige lediglich bis 7 zählen könnte. Betrachtet man die Tabelle, die die Dezimalzahlen mit den Binärzahlen verknüpft, erkennt man direkt, dass es ausreicht ein NAND-Gatter an Q_D und Q_B anzuschließen, das beim erreichen der 10 über den an am Ausgang angeschlossenen Kanal $\overline{C}$ den Zähler zurücksetzt. Das entspricht folgendem Schaltplan:



Aufgabe 6.3 - 4-Bit-Synchronzähler

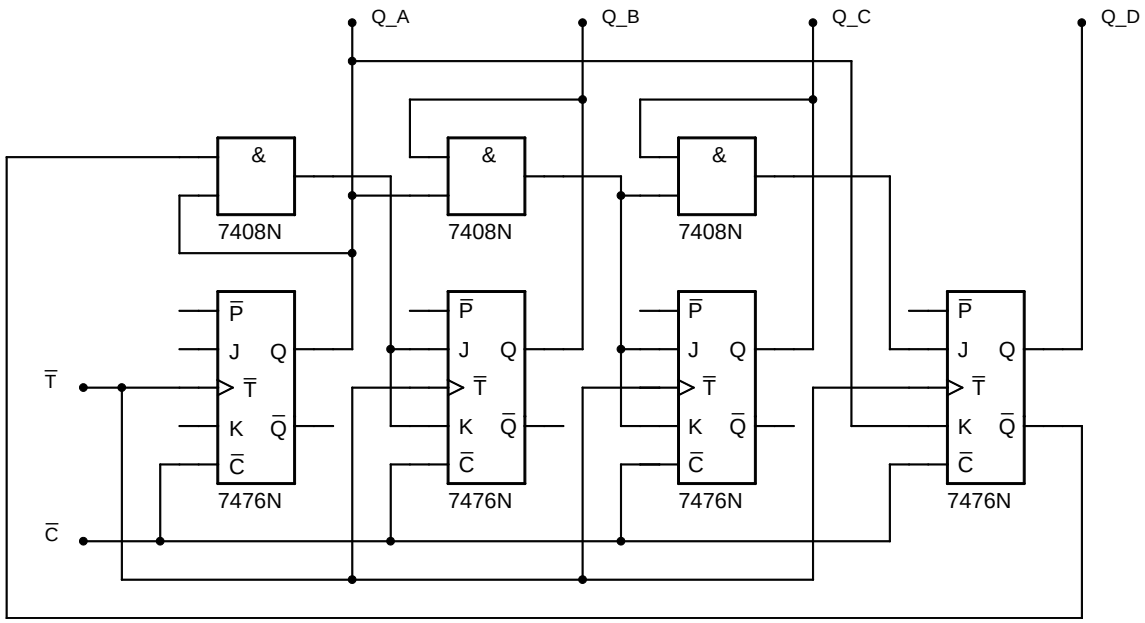
Da der asynchrone Zähler wie eben beschrieben in der Praxis meistens nicht einsetzbar ist, soll nun noch ein synchroner 4-Bit-Zähler nach folgendem Schaltplan aufgebaut werden:



Wie leicht zu erkennen ist, sind nun alle Takteingänge miteinander verbunden, der Zähler läuft also synchron. Um nun zu verhindern, dass jedes JK-MS-FF bei jedem Takt toggelt, werden an jeder Stufe die Ausgänge der vorangegangenen Stufen mit einem AND-Gatter verknüpft und an die J- und K-Eingänge des nächste Stufe angelegt. Dadurch schaltet eine Stufe erst, wenn alle vorangegangenen Stufen auf 1 liegen. Bei dem zu untersuchenden 4-Bit-Zähler reicht es, diese Verknüpfung nur bei der letzten und vorletzten Stufe einzufügen.

Aufgabe 6.4 - Synchroner Dezimalzähler

Entsprechend wie beim Asynchronzähler soll nun noch ein synchroner Dezimalzähler aufgebaut werden. Dies erreicht man wieder durch eine Verknüpfung von Q_D und Q_B mittels eines NAND-Gatters.
Eine Alternative dazu stellt folgender Schaltplan dar, der hier allerdings nicht weiter erläutert werden soll:



Aufgabe 7: Digital-Analog-Wandlung

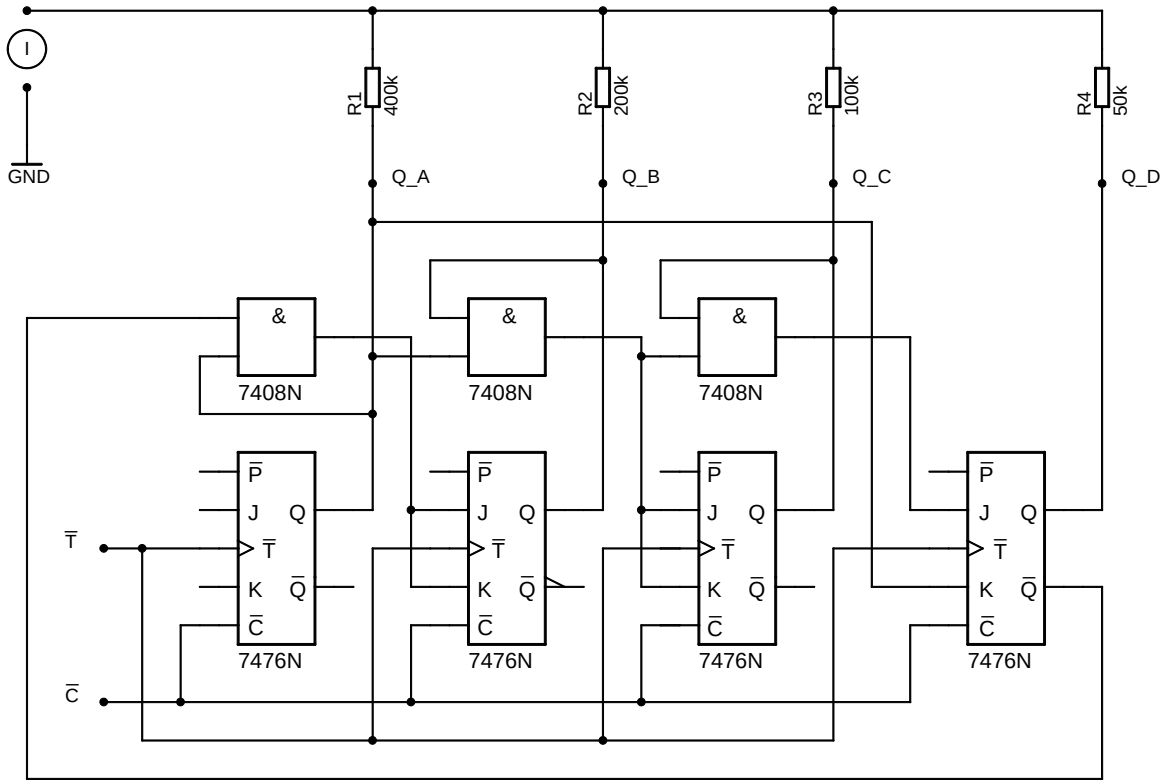
Die digitale Anzeige der in Aufgabe 6.2 und 6.4 aufgebauten Dezimalzähler soll nun mit Hilfe eines Drehspulmessinstruments analog angezeigt werden. Das Messgerät kann Ströme von $0\,\mu\mathrm{A}$ bis $100\,\mu\mathrm{A}$ mit einer Nadel auf einer Skala darstellen. Daher liegt es nahe eine Schaltung aufzubauen, die einem Dezimalzahlschritt von 1 einen Strom von $10\,\mu\mathrm{A}$ zuordnet und an das Drehspulmessinstrument weitergibt. Eine 1 an Q_A soll daher einen Strom von $10\,\mu\mathrm{A}$ erzeugen, eine 1 an Q_B den doppelten, also $20\,\mu\mathrm{A}$, usw. Erreicht wird dies durch verschiedene Widerstände an den Ausgängen der Stufen. Es wird mit einer Eingangsspannung von $U = 4\,\mathrm{V}$ gerechnet:

$$R_1 = \frac{U}{I} = \frac{4\,\mathrm{V}}{10\,\mu\mathrm{A}} = 400\,\mathrm{k\Omega}$$

Und entsprechend für die anderen Stufen:

Widerstand	Strom	Bit	Binärzahl	Dezimalzahl
$R_1 = 400\,\mathrm{k\Omega}$	$10\,\mu\mathrm{A}$	niedrigstes	0001	1
$R_2 = 200\,\mathrm{k\Omega}$	$20\,\mu\mathrm{A}$	zweitniedrigstes	0010	2
$R_3 = 100\,\mathrm{k\Omega}$	$40\,\mu\mathrm{A}$	drittniedrigstes	0100	4
$R_4 = 50\,\mathrm{k\Omega}$	$80\,\mu\mathrm{A}$	viertniedrigstes	1000	8

Die anderen Enden der Widerstände werden parallel an den Pluspol des Messinstruments angeschlossen. Die Schaltung sieht dann wie folgt aus:



Es kann zu Ungenauigkeiten bei der analogen Anzeige kommen, da der angenommene Strom nicht genau 4 V beträgt.

Quellenangabe

Sämtliche Schaltskizzen entstammen der Vorbereitungsmappe.