# Kapitel 2

## Kombinatorische Schaltungen

Definition nach DIN 44300/93
Ein Schaltnetz oder kombinatorischer Funktionsblock ist eine Funktionseinheit zum Verarbeiten von Schaltvariablen, deren Wert am Ausgang zu beliebiegem Zeitpunkt nur vom Eingangswert zum gleichen Zeitpunkt abhängt.

## Codierer und Decodierer

Ein Code ist (im Kontext der digitalen Informationsverarbeitung) ein Bitstring zur eindeutigen Bezeichnung eines Objektes. Eine Codierung ist dann die Erzeugung und Zuordnung von Objekten zu Codes. Formal ist eine Codierung eine Abbildung

C: M $\rightarrow$ $\{0,1\}^n$, wobei n die Länge eines Codes ist und M die Menge der Objekte.

Es gibt nun vielfältige Möglichkeiten, die Funktionsvorschrift festzulegen. Im Allgemeinen möchte man die Länge der Codes so kurz wie möglich halten. Ausserdem sollten alle Codes gleich lang sein.

Sei | M | die Kardinalität der Menge der zu codierenden Objekte. Alle Codes sollen dieselbe Länge haben.

Wie lang müssen die Codes sein?

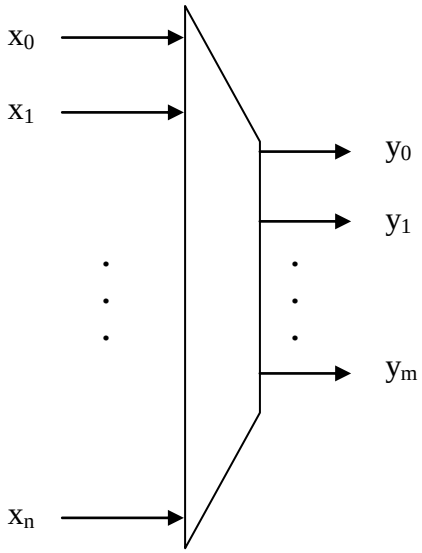
Wir betrachten im Folgenden nur noch Codierungen, die Codes gleicher Länge erzeugen. Nach der Erzeugung von Codes betrachten wir nun die Umkehrung, d.h. die Entschlüsselung von gegebenen Codes. Dabei soll für eine vorgegebene Menge von Codes jedem Code ein Objekt zugeordnet werden. Keine zwei Codes dürfen dasselbe Objekt bezeichnen. Formal ist eine Decodierung eine Abbildung:

D: $\{0,1\}^n$ $\rightarrow$ M, wobei n die Länge der Codes ist.

Unter der Vorraussetzung, dass alle Elemente aus $\{0,1\}^n$ gültige Codes sind:
Wir gross muss | M | sein?

## Codierer

In der Digitaltechnik werden Codierer durch Trapeze dargestellt:

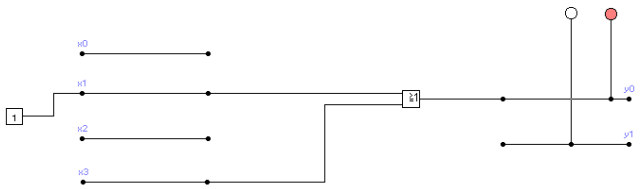



Die $x_i$ sind die Eingänge, die $y_j$ die Ausgänge. Jede Eingangsleitung steht für ein Objekt, zu dem ein Code = Bitstring $y_m \ldots y_0$ erzeugt werden soll. Es wird vorausgesetzt, dass immer nur genau eine Eingangsleitung aktiv (= 1) ist und alle anderen 0. Ein Code soll, als Betragszahl $y_m \ldots y_0$ interpretiert, den Index der aktiven Eingangsleitung darstellen.

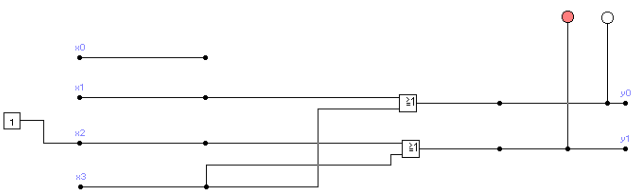
Ein Beispiel: n=15, m=3, $x_{12} = 1$, dann soll der Codierer $y_3 \ldots y_0 = 1100$ liefern.

Überlegungen zur Konstruktion am Beispiel eines 4 zu 2 Codierers: mögliche Eingaben: 0001, 0010, 0100, 1000; Ausgaben: 00, 01, 10, 11

Ist der Index des (aktiven) Eingangs ungerade, muss y0 auf 1 gesetzt werden:



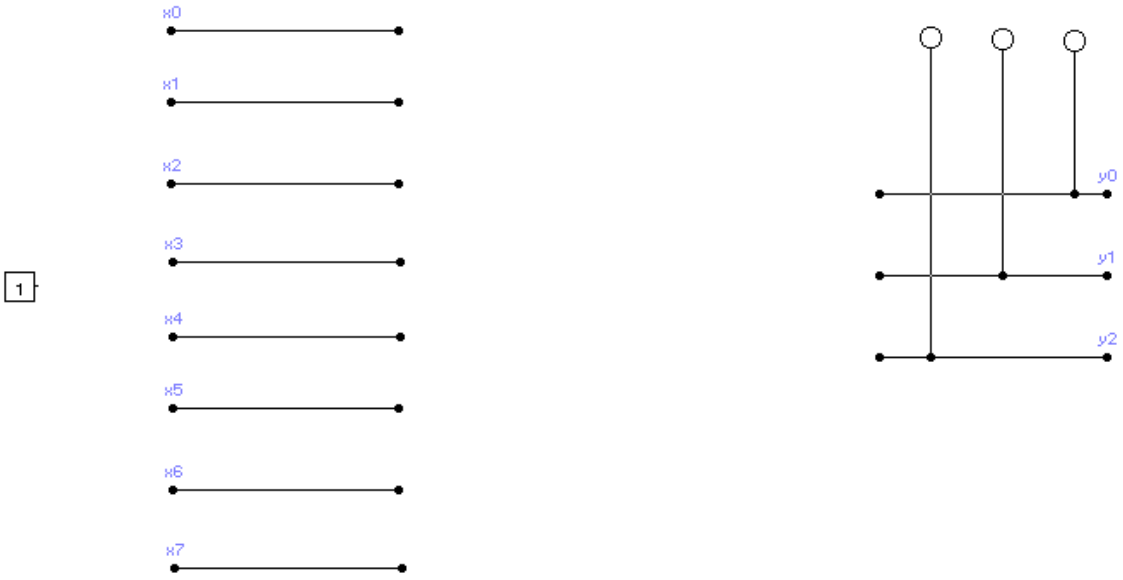
Ist der Index des aktiven Eingangs ≥ 2, muss y1 auf 1 gesetzt werden:



## Versuch 200 8-zu-3-Codierer

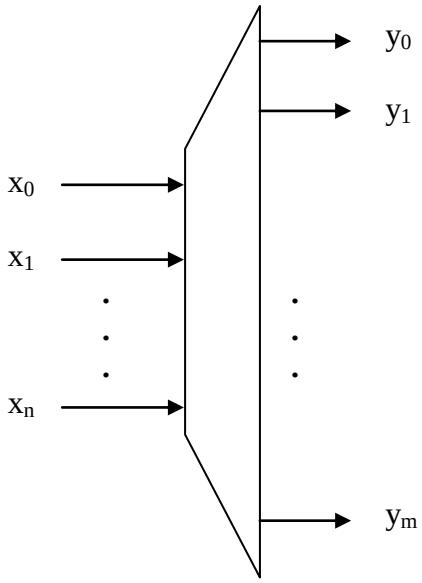
Bauen Sie in der Datei v200 einen 8-zu-3-Codierer, der folgende Codierungsvorschrift umsetzt.

Jede Eingangsleitung steht für ein Objekt, zu dem ein Code = Bitstring $y_2y_1y_0$ erzeugt werden soll. Es wird vorausgesetzt, dass immer nur genau eine Eingangsleitung aktiv (= 1) ist und alle anderen 0. Ein Code soll, als Betragszahl $y_2y_1y_0$ interpretiert, den Index der aktiven Eingangsleitung darstellen.



## Decodierer

Decodierer werden ebenfalls durch Trapeze dargestellt:

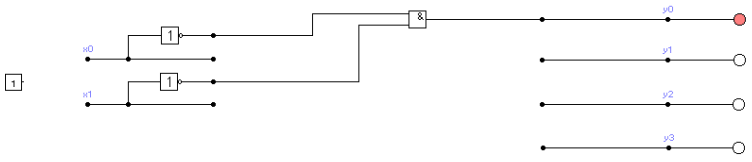



Die $x_i$ sind die Eingänge, die $y_j$ die Ausgänge. Die Belegung aller Eingänge durch Nullen und Einsen stellt einen Code dar. Der Decodierer aktiviert (= 1 setzen) diejenige Ausgangsleitung $y_i$, deren Index dem Bitstring $x_n \ldots x_0$, interpetiert als Betragszahl, enspricht. Alle anderen Ausgänge werden auf Null gesetzt.

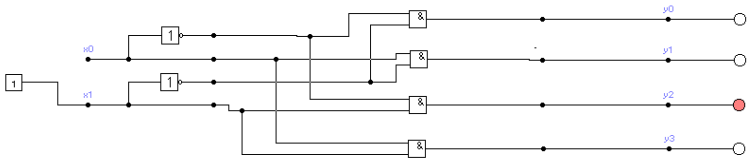
Ein Beispiel: n=3, m=15, $x_3x_2x_1x_0 = 1000$, dann soll der Decodierer $y_8 = 1$ und alle anderen $y_i = 0$ setzen.

Überlegungen zur Konstruktion am Beispiel eines 2 zu 4 Decodierers: mögliche Eingaben: 00, 01, 10, 11. Ausgaben: 0001, 0010, 0100, 1000.

y0 soll nur genau dann 1 sein, wenn x0 und x1 beide 0 sind.



Analoge Überlegungen für die anderen drei Ausgänge.



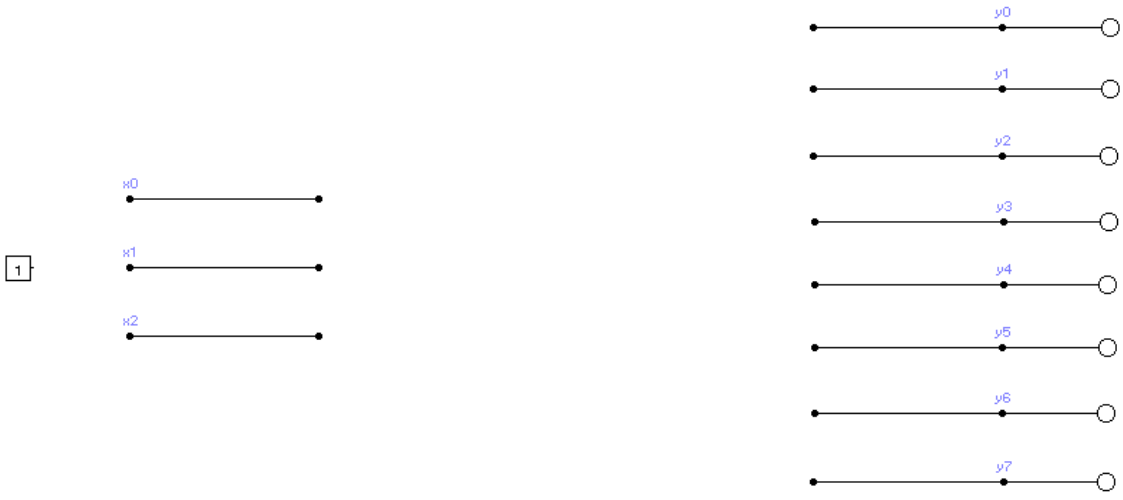
## Versuch 205 3-zu-8-Decodierer

Bauen Sie in der Datei v205 einen 3-zu-8-Decodierer mit folgender Funktion:

Die $x_i$ sind die Eingänge, die $y_j$ die Ausgänge. Die Belegung aller Eingänge durch Nullen und Einsen stellt einen Code dar. Der Decodierer aktiviert (= 1 setzen) diejenige Ausgangsleitung $y_i$, deren Index dem Bitstring $x_2x_1x_0$, interpetiert als Betragszahl, enspricht. Alle anderen Ausgänge werden auf Null gesetzt.

Nebenbedingung: Sie dürfen höchstens elf Gatter verwenden.

Anmerkung: durch Doppelklick auf ein Gatter kann die Anzahl der Eingänge verändert werden.



## Prioritätenencoder

Encoder ist nur ein anderes Wort (aus dem Engl. abgeleitet) für Codierer.

Bei $2^n$-zu-n-Codierern wird gefordert, dass genau ein Eingang auf 1 und alle anderen auf 0 gesetzt werden. Diese Bedingung ist oft nicht einzuhalten bzw. auch gar nicht erwünscht. Beim Prioritätenencoder wird diese Bedingung fallengelassen.

Im Gegenzug muss die Semantik dieses Bausteins erweitert werden. Ist wie beim bisherigen Codierer nur genau eine Eingangsleitung 1, liefert auch der Prioritätenencoder einen Code, der, als Betragszahl interpretiert, dem Index der aktiven Eingangsleitung entspricht. Sind aber zwei oder mehr Eingangsleitungen 1, wertet der Baustein eine vorher festgelegte Priorität unter den Eingangsleitungen aus.

Beispiel für eine Prioritätenfestlegung:
$x_i$ hat höhere Priorität als $x_j$, falls $i < j$.

Der Encoder liefert dann den Code für den Index des Eingangs mit der höchsten Priorität.

Für obiges Beispiel bedeutet dies umgangssprachlich, es wird der Eingang bestimmt, der mit 1 belegt ist und für den gilt, dass es keinen mit 1 belegten Eingang mit niedrigerem Index gibt, oder noch kürzer:

$\text{Min}\ (\ \{i \mid x_i = 1\}\ )$

Für den so bestimmten Index wird die Betragszahl als Code ausgegeben.

Wie sieht der mathematische Ausdruck aus, wenn die Priorität wie folgt festgelegt ist:

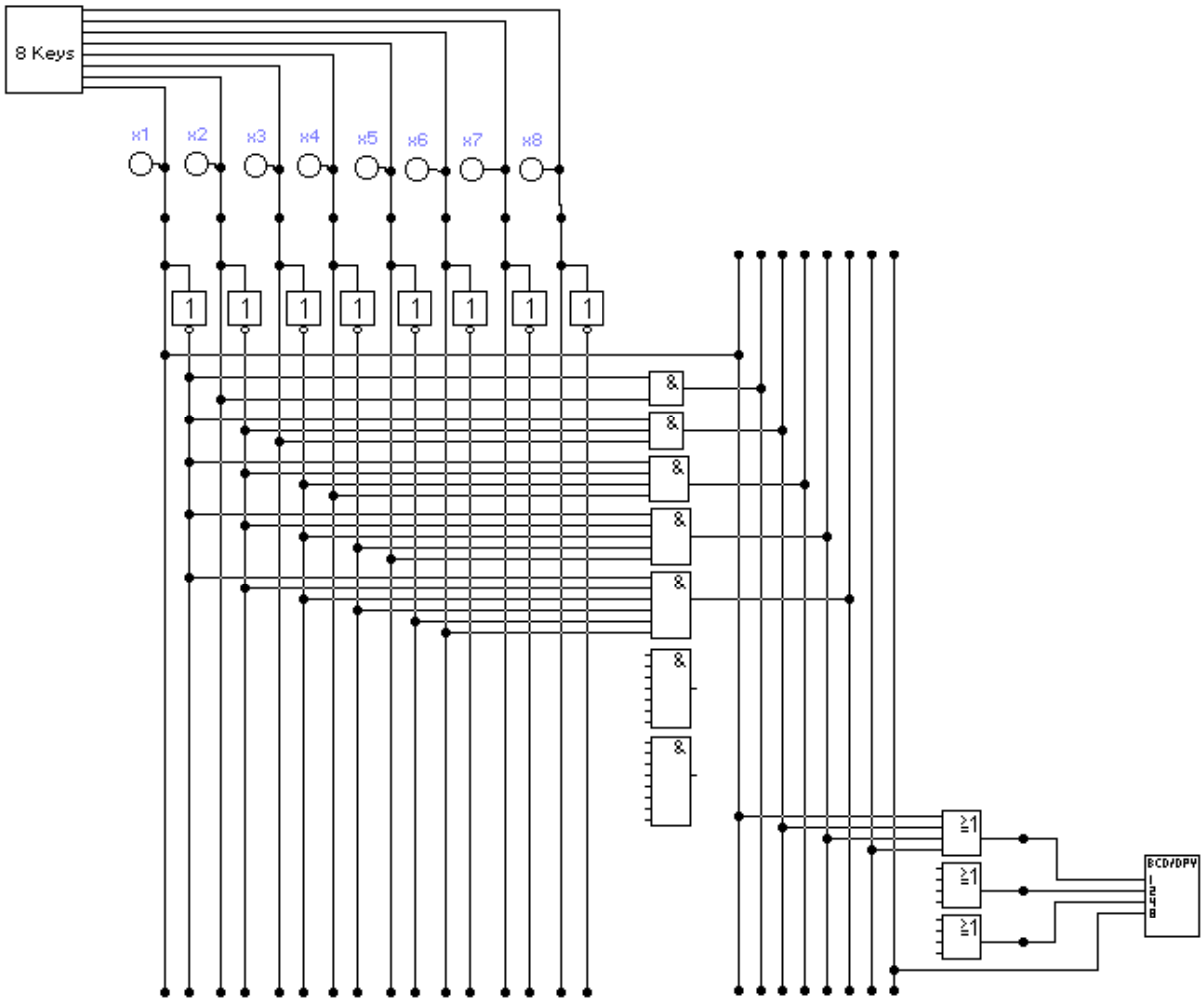
$x_i$ hat höhere Priorität als $x_j$, falls $i > j$.

## Versuch 210 Prioritätenencoder

Vervollständigen Sie in der Datei v210 die Schaltung eines Prioritätenencoders mit acht Eingängen und der Prioritätenfestlegung "Min ( $\{i \mid x_i = 1\}$ ) hat die höchste Priorität".

Anmerkung: Die Ausgabe wurde um ein Bit erweitert und die Eingänge von 1 bis 8 indiziert, um für den Fall, dass kein Eingang mit 1 belegt ist, eine 0 ausgeben zu können.

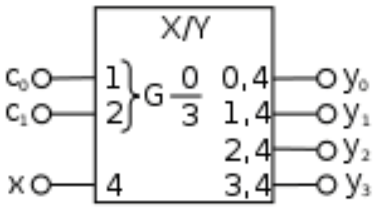
Testen Sie ihre Schaltung. Die Eingänge können mit Hilfe der Tasten 1 bis 8 auf 0 oder 1 gesetzt werden.



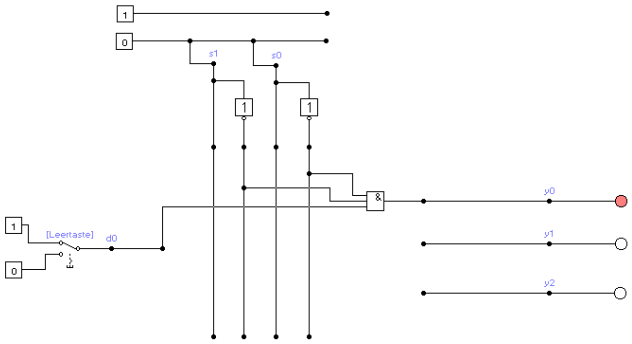

Wie genau und wodurch wurde die Prioritätensteuerung umgesetzt?

## Demultiplexer

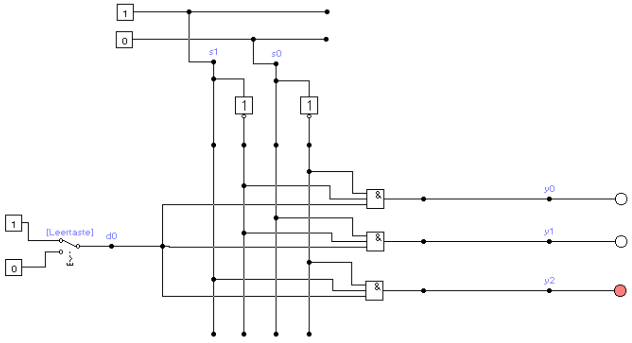
In einem Rechner benötigt man zwischen Komponenten Leitungen, um Daten auszutauschen. Da ein voll vermaschtes Netz zwischen allen Kompenenten sehr aufwendig und häufig auch nicht erforderlich ist, z.B. wenn nicht alle Datenwege gleichzeitig benötigt werden, braucht man je nach Transferrichtung “Verteiler“ oder “Sammler“. Ein Demultiplexer ist ein Verteiler, der ein Bündel von Leitungen auf zwei oder mehr andere Bündel von Leitungen durchschaltet. Ein Beispiel:




Das Eingangsbündel besteht hier nur aus der Leitung x. Es gibt vier Ausgangsbündel $y_0$ bis $y_3$, die jeweils auch nur aus einer Leitung bestehen. Die Steuereingänge $c_0$ und $c_1$ bestimmen, auf welches y die Leitung x durchgeschaltet wird.
Betrachten wir nun den einfachen Fall einer einzigen Leitung, die alternativ auf drei andere Leitungen durchgeschaltet werden soll.
Neben diesen Ein- und Ausgängen benötigt der Demultiplexer noch Steuereingänge, um zu “wissen“, auf welchen Ausgang der Eingang geschaltet werden soll. Ähnlich wie beim Decodierer legen wir fest, dass die Belegung der Steuereingänge, als Binärzahl interpretiert, dem Index des auszuwählenden Ausgangs entspricht.
Um einen aus drei Ausgängen auszuwählen, brauchen wir zwei Bits. Die Codierung (11) wird nicht benötigt, da es keinen Ausgang mit dem Index 3 gibt (wir beginnen die Indizierung wie üblich mit 0).
Überlegungen zur Konstruktion: Nur wenn $s_1s_0 = 00$, soll $d_0$ auf $y_0$ durchgeschaltet werden:



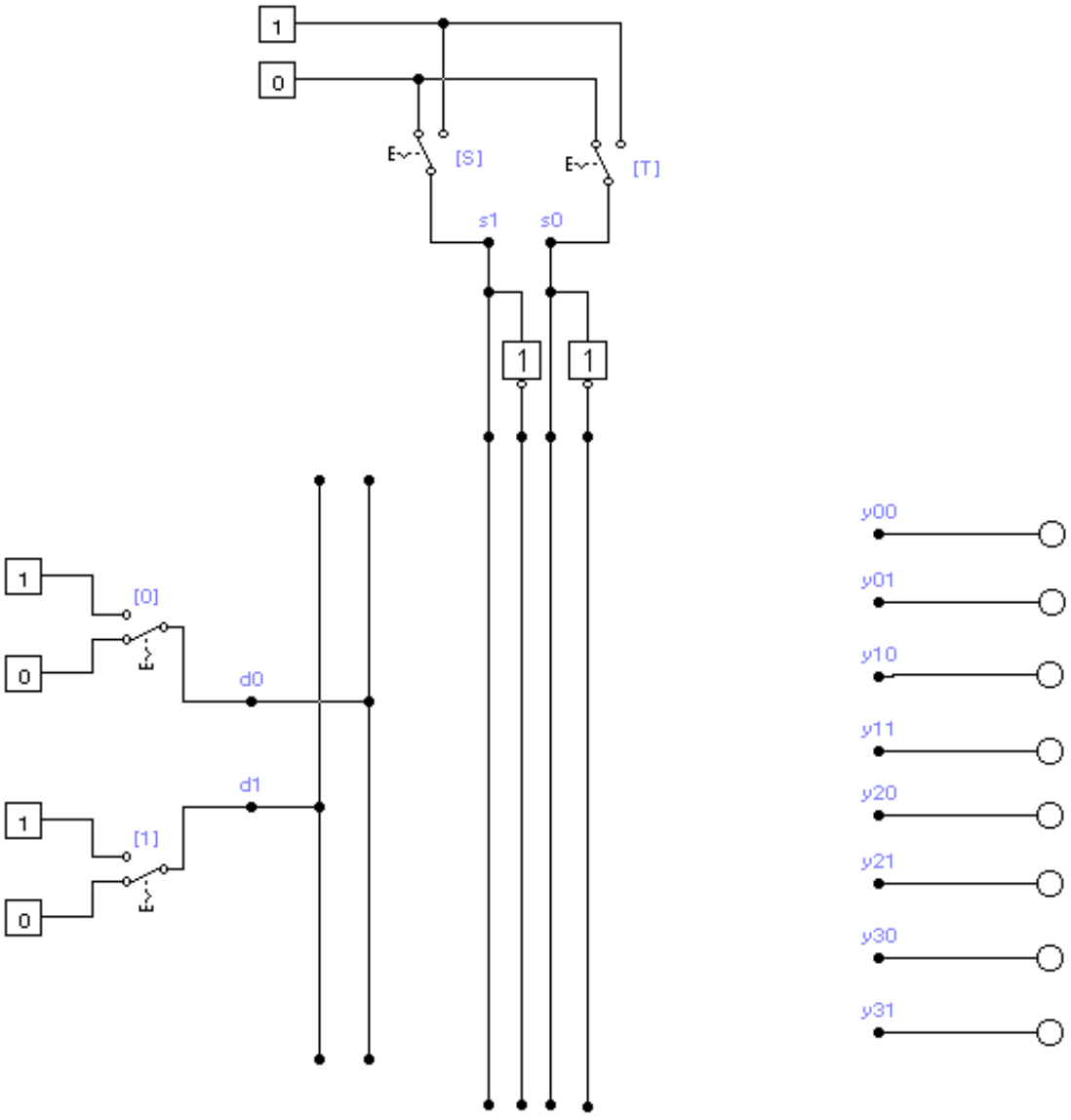

Analog werden die beiden anderen Ausgänge beschaltet:



## Versuch 215 2-zu-4-Demultiplexer

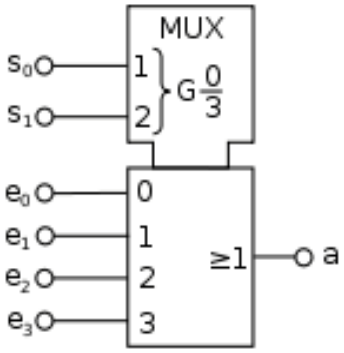
Bauen Sie in der Datei v215 einen 2-zu-4-Demultiplexer. Sei i die durch $s_1s_0$ codierte Betragszahl. Dann soll gelten:

$d_j$ soll auf $y_{ij}$ durchgeschaltet werden für j = 0..1.



## Versuch 220 Multiplexer

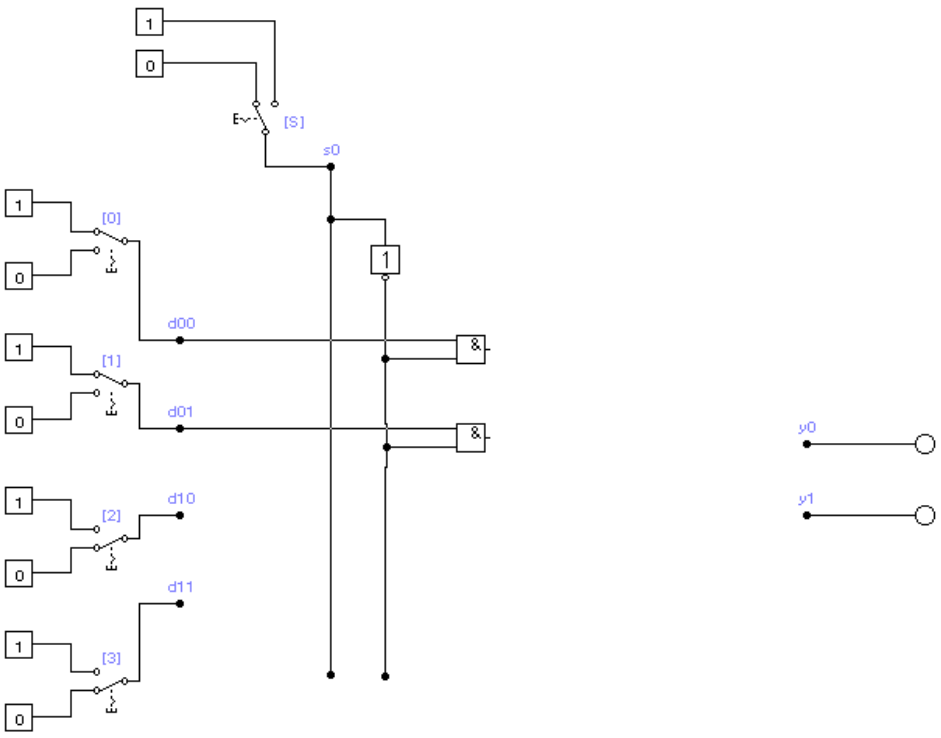
Multiplexer sind das Gegenstück zum Demultiplexer. Sie „sammeln“ Bündel von Leitungen und schalten ein Bündel auf das Ausgangsbündel durch. Anwendungen finden sich bspw. in einem Rechenwerk, in dem eine ALU ihre Operatoren aus mehreren verschiedenen Quellen beziehen kann. Auch hier wieder ein Bespiel:




Einer der Eingänge $e_i$ soll mit Hilfe der Steuereingänge $s_0$ und $s_1$ auf den Ausgang a durchgeschaltet werden.

Betrachten wir nun einen Multiplexer mit zwei Eingangsbündeln mit je zwei Leitungen.

Überlegungen zur Konstruktion: nur bei $s_0 = 0$ soll $d_{00}$ auf $y_0$ und $d_{01}$ auf $y_1$ durchgeschaltet werden.




Ergänzen Sie obige Schaltung in Datei v220 zunächst für die Eingänge $d_{10}$ und $d_{11}$. Was müssen Sie beachten wenn Sie nun die Ausgänge der UND-Gattern auf die Ausgänge $y_0$ und $y_1$ schalten wollen? Dürfen Sie Augänge von Gattern miteinander verbinden?