

Kontemplation und Motivation

- > Bisher: zweistufige Realisierung von Schaltnetzen. Erst UND-Gatter dann ODER-Gatter, oder umgekehrt → kurze Signallaufzeiten
- > Minimierungsverfahren:
 - möglichst wenig Gattereingänge sowie
 - möglichst wenig UND- und ODER-Gatter
- > Unterschiedliche Bausteinformate (AND3, AND4, OR5) ergeben **Probleme** für die Automatisierung des Chip-Designs

Komplexere Bausteine

- > Verwenden von komplexeren Standardbausteinen zur Realisierung logischer Funktionen durch
 - Multiplexer/Demultiplexer/Decoder
 - Speicherbausteine sowie
 - programmierbare Logikbausteine PLA, FPLA und PAL

3.3.3 Spezielle Strukturen

- Anstelle aus logischen Gattern (UND, ODER, NICHT, NAND, NOR, ... usw.) lassen sich Schaltnetze auch mit komplexeren Standardbausteinen realisieren.
- **Einfachere und oft flexiblere Realisierung**
- > Minimierung bedeutet hierbei dann nicht die Reduktion von Gattern.
- Es muss die Anzahl und Größe komplexer Bausteine reduziert werden.

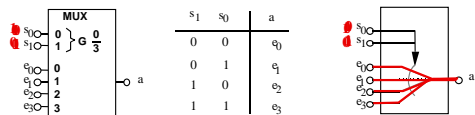
Multiplexer

Ein **Multiplexer** (Abk.: **MUX**) ist ein Baustein mit mehreren Eingängen und einem Ausgang, wobei über **n** Steuerleitungen einer der **2ⁿ** Eingänge auf den Ausgang geschaltet wird.

Multiplexer werden nach ihrer Größe als **2ⁿ-1-Multiplexer** (alternativ als **1-aus-2ⁿ-Multiplexer**) klassifiziert.

Multiplexer

Schaltbild und logisches Verhalten eines 4:1-Multiplexers



Steuerung der Signalpfade



Multiplexer

Ein Multiplexer kann nicht nur zur Steuerung von Datenflüssen sondern auch zur Realisierung logischer Funktionen verwendet werden.

Man kann mit einem 2ⁿ-1-Multiplexer eine logische Funktion mit n+1 Variablen implementieren.

Hierzu wird die sog. **Implementierungstabelle** verwendet.

Implementierungstabelle

- > Die Tabelle besteht aus:
 - **2ⁿ** Spalten für die möglichen Belegungen der **n** Steuereingänge (x_0, x_1, \dots, x_{n-1})
 - **2** Zeilen für die negierte und nicht negierte (**n+1**)-te Variable x_n
- > In die Tabelle werden die Funktionswerte in Abhängigkeit von den Variablen eingetragen.
- > Anschließend betrachtet man jede Spalte für sich und ordnet ihr eine einstellige Funktion $g \in \{0, 1, \bar{x}_n, x_n\}$ zu, mit der dann der Eingang belegt wird, der zu der entsprechenden Steuervariablen-Kombination gehört.

Beispiel

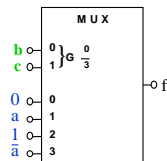
Realisierung einer Funktion mit Multiplexer: $n=3$
 $f = \bar{a}c \vee \bar{b}c \vee a b \bar{c}$ $2^3=1-MUX$

$f = \bar{c} \bar{b} \cdot 0 \vee \bar{c} \bar{b} \cdot a \vee c \bar{b} \cdot 1 \vee c b \bar{a}$

Implementierungstabelle bei Wahl von b und c als Steuereingänge:

c b	00	01	10	11
a=0	0	0	1	1
a=1	0	1	1	0
g =	0	a	1	\bar{a}

Realisierung



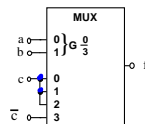
Beispiel 2

Realisierung einer Funktion mit Multiplexer:
 $f = \bar{a}c \vee \bar{b}c \vee a b \bar{c}$

Implementierungstabelle bei Wahl von a und b als Steuereingänge:

b a	00	01	10	11
c=0	0	0	0	1
c=1	1	1	1	0
g =	c	c	c	\bar{c}

Realisierung:



Shannonscher Entwicklungssatz

Man kann auch Teilfunktionen mit einem Multiplexer realisieren.

Hierzu verwendet man den Shannonschen Entwicklungssatz.

Wdh.: Shannonentwicklung nach c:

$$f(a,b,c) = [c \wedge f(a,b,1)] \vee [\bar{c} \wedge f(a,b,0)]$$

Vorgehensweise:

Die Funktion wird nach allen Steuervariablen des Multiplexers entwickelt, die verbleibenden Restfunktionen sind an die entsprechenden Eingänge des Multiplexers zu führen.

Beispiel

Die Funktion

$$f = \bar{a}c \vee \bar{b}c \vee a b \bar{c}$$

soll mit einem 2:1-Multiplexer und Gattern entworfen werden. Steuervariable des Multiplexers sei c

Entwicklung nach c:

$$f = c \cdot (\bar{a} \vee \bar{b}) \vee \bar{c} \cdot a b$$

$$f(c=1) = \bar{a} \vee \bar{b} = \bar{a} \bar{b}$$

$$f(c=0) = a b$$

Realisierung:

