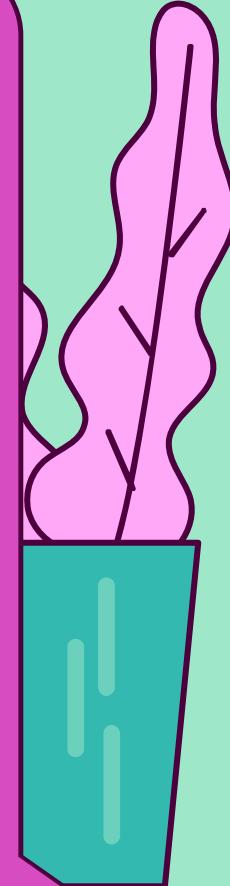
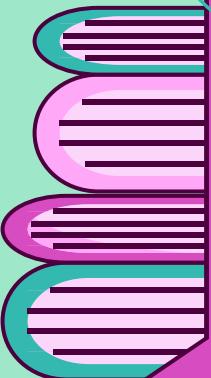
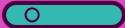


Herzlich willkommen
zum *sechsten* ERA
Tutorium des
Semesters!





Einstelliger Operator:

Negation

Umkehrung des Wahrheitswertes

- Symbol: \neg oder Querstrich über der Aussage
- Beispiel: $\neg a$, \bar{b}

Und-Verknüpfung

- Gesamtaussage ist wahr, wenn beide Teilaussagen wahr sind
- Symbol: \wedge oder *
- Beispiel: $a \wedge b$, $a * b$

Oder-Verknüpfung

- Gesamtaussage ist dann wahr, wenn mindestens eine der beiden Teilaussagen wahr ist
- Symbol: \vee oder +
- Beispiel: $a \vee b$, $a + b$

Äquivalenz-Verknüpfung

- Gesamtaussage ist dann wahr, wenn beide Aussagen den gleichen Wahrheitswert haben
- Symbol: \equiv oder \Leftrightarrow
- Beispiel: $a \equiv b$, $a \Leftrightarrow b$

Antivalenz

- Gesamtaussage ist wahr, wenn genau eine Teilaussage wahr ist
- Symbol: \neq
- Beispiel: $a \neq b$

Negiertes Exklusives Oder (XNOR)

- Gesamtaussage ist dann wahr, wenn beide Aussagen den gleichen Wahrheitswert haben
- Symbol: \equiv oder \Leftrightarrow
- Beispiel: $a \equiv b$, $a \Leftrightarrow b$

a	b	$a \text{ XNOR } b$
0	0	1
0	1	0
1	0	0
1	1	1

Exklusives Oder (XOR)

- Gesamtaussage ist wahr, wenn genau eine Teilaussage wahr ist
- Symbol: \oplus
- Beispiel: $a \oplus b$

■ Implikation

- Gesamtaussage ist nur dann falsch, wenn erste Teilaussage wahr und zweite Teilaussage falsch ist
- Entspricht in etwa der umgangssprachlichen Wenn-dann Formulierung
(ist aber exakter, z.B. wenn die erste Teilaussage falsch ist)
- Symbol: \Rightarrow
- Beispiel: $a \Rightarrow b$ ($:= \neg a \vee b$)
 a = „ich bestehe die Prüfung“
 b = „ich bin glücklich“

Kann ich glücklich sein ohne die Prüfung zu bestehen? ✓
Kann ich die Prüfung bestehen ohne glücklich zu sein? ⚡

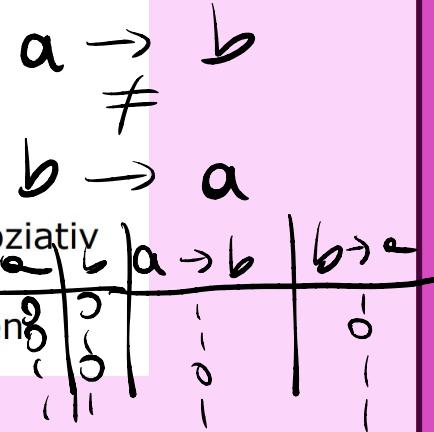
a	b	$a \Rightarrow b$
0	0	1
0	1	1
1	0	0
1	1	1

Kommutativität

- $a \wedge b \equiv b \wedge a$
- Und, Oder, Äquivalenz und Antivalenz sind kommutativ
- Implikation ist nicht kommutativ

Assoziativität

- $(a \wedge b) \wedge c \equiv a \wedge (b \wedge c)$
- Und, Oder, Äquivalenz und Antivalenz sind assoziativ
- Implikation ist nicht assoziativ
- Wichtig bei der Verknüpfung mehrerer Aussagen

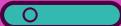


Tautologie

- Aussage, die immer wahr ist
- Beispiele:
 - $a \vee \neg a$
 - $(a \Rightarrow b) \vee (b \Rightarrow a)$

Kontradiktion

- Aussage, die immer falsch ist
- Beispiel: $a \wedge \neg a$



$$\begin{aligned}(1) \quad & a + b = b + a \\& a * b = b * a\end{aligned}$$

Kommutativität

$$\begin{aligned}(2) \quad & 0 + a = a \\& 1 * a = a\end{aligned}$$

Existenz von „0“ und „1“ (neutrale Elemente)

$$\begin{aligned}(3) \quad & (a + b) * c = (a * c) + (b * c) \\& (a * b) + c = (a + c) * (b + c)\end{aligned}$$

Distributivität

$$\begin{aligned}(4) \quad & a + a^{-1} = 1 \\& a * a^{-1} = 0 \\& \quad \quad \quad \text{!}\end{aligned}$$

**Zu jedem Element $a \in B$ existiert
ein komplementäres Element $a^{-1} \in B$**

Assoziativitt

- $(a + b) + c = a + (b + c)$
- $(a * b) * c = a * (b * c)$

Idempotenz

- $a + a = a$
- $a * a = a$

Absorption

- $(a + b) * a = a$
- $(a * b) + a = a$

De Morgansche Regeln

- $(a^{-1} * b^{-1}) = (a + b)^{-1}$
- $(a^{-1} + b^{-1}) = (a * b)^{-1}$

Spezielle Boolesche Algebra

- Zwei Elemente: $B = \{0, 1\}$
- $+$ = Oder (\vee)
- $*$ = Und (\wedge)
- a^{-1} = Negation ($\neg a$)

$$\square a \vee b = \neg(\neg a \wedge \neg b)$$

Variante (1) negieren

$$\square a \wedge b = \neg(\neg a \vee \neg b)$$

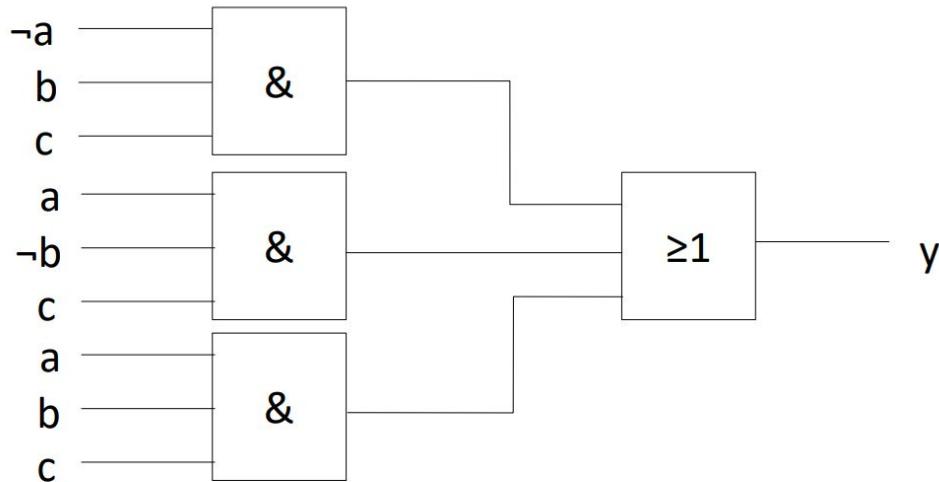
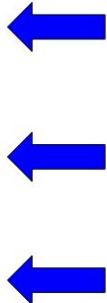
Variante (2) negieren

freie Symbole	Schaltsymbole nach DIN 40 700 Teil 14		amerikanische Symbole	logische Darstellung
	seit 1976	bis 1976		
UND				$x_1 \wedge \dots \wedge x_n$
ODER				$x_1 \vee \dots \vee x_n$
Anti-valenz				$x_1 \neq \dots \neq x_n$
NAND				$\overline{x_1 \wedge x_2 \wedge \dots \wedge x_n}$
NOR				$\overline{x_1 \vee x_2 \vee \dots \vee x_n}$
Negation				\bar{x}_1

Synthese (simpler)

- Realisierung beliebiger Wahrheitstabellen durch Grundgatter möglich
- Vorgehen:
 - Für jede Zeile mit Ausgabewert 1:
Und-Gatter mit passender Eingangsbeschaltung
 - Oder-Verknüpfung aller Und-Gatter
- Funktioniert für alle Tabellen, aber
 - teuer und
 - nicht skalierbar

a	b	c	y
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1



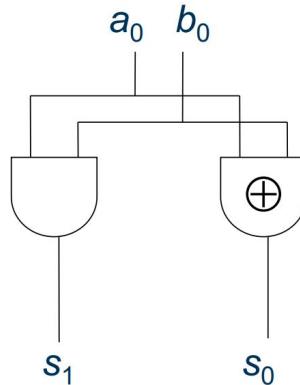
Der Halbaddierer dient zur Addition zweier 1-Bit-Zahlen *ohne* Eingangsübertrag.

Er berechnet die Funktion:

a_0	b_0	ha_1	ha_0
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Folglich:

$$ha_0 = a_0 \oplus b_0 \quad ha_1 = a_0 \wedge b_0$$



Kosten und Tiefe eines HA:

$$C(\text{HA}) = 2, \quad \text{depth}(\text{HA}) = 1$$

Der Volladdierer dient zur Addition zweier 1-Bit-Zahlen *mit* Eingangsübertrag.

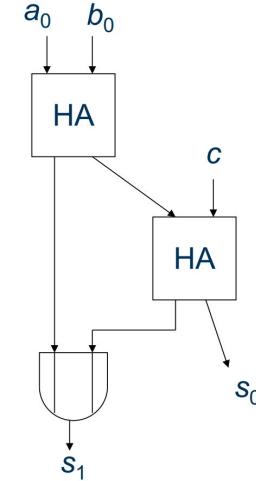
Er berechnet die Funktion:

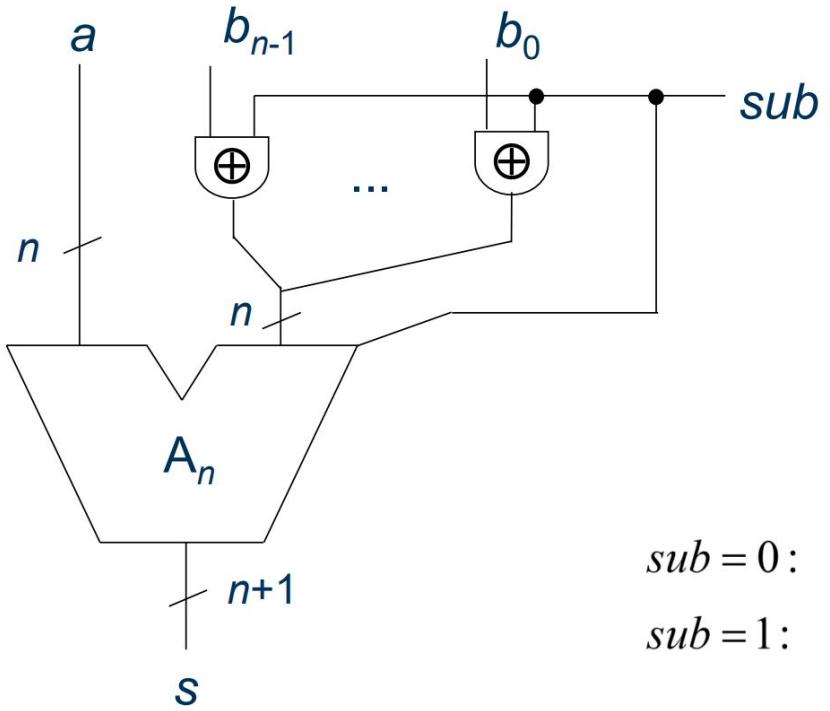
$$fa_0 = a_0 \oplus b_0 \oplus c = ha_0(c, ha_0(a_0, b_0))$$

a_0	b_0	c	fa_1	fa_0
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Kosten und Tiefe eines FA:

$$C(FA) = 5, \text{depth}(FA) = 3$$





$$b_i \oplus 0 = b_i$$

$$b_i \oplus 1 = \bar{b}_i$$

$$sub = 0 : [a] + [b] + 0$$

$$sub = 1 : [a] + [\bar{b}] + 1 = [a] - [b]$$

Überprüfe mittels einer vollständigen Wahrheitstabelle die (Un-)Gleichheit folgender Aussagen:

Hinweis: Beachte, dass wir zu Übungszwecken in den folgenden Teilaufgaben beide Notationen aus der Vorlesung verwenden, d.h., wahlweise $+$, \cdot oder \vee , \wedge .

a) $\overline{x+y} = \overline{x} \cdot \overline{y}$

x	y	$x+y$	$\overline{x+y}$	\overline{x}	\overline{y}	$\overline{x} \cdot \overline{y}$
0	0	0	1	1	1	1
0	1	1	0	1	0	0
1	0	1	0	0	1	0
1	1	1	0	0	0	0

Now your turn!

c) $f := \neg((\neg x \wedge y) \vee (\neg y \wedge z)) \wedge (\neg x \wedge z)$ ist eine Kontradiktion

x	y	z	$\neg x \wedge y$	$\neg y \wedge z$	$\neg x \wedge z$	$\neg c$	$\neg(\neg x \wedge z)$	$\neg c \wedge d$
0	0	0	0	0	0	1	0	0
0	0	1	0	1	1	0	1	0
0	1	0	1	0	1	0	0	0
0	1	1	1	0	1	0	1	0
1	0	0	0	0	0	1	0	0
1	0	1	0	0	0	1	0	0
1	1	0	0	0	1	0	0	0
1	1	1	0	0	0	1	0	0

Prüfe oder widerlege die folgenden Aussagen und verwende dazu die Regeln und Gesetze der Booleschen Algebra. Gib dazu bei jedem Beweisschritt die verwendete Regel bzw. das verwendete Gesetz an.

Hinweis: Beachte, dass wir zu Übungszwecken in den folgenden Teilaufgaben beide Notationen aus der Vorlesung verwenden, d.h., wahlweise $+$, \cdot oder \vee , \wedge .

a)
$$\text{LHS} = \overline{a \cdot b} + b = 1 \quad \text{rhs}$$

$$\begin{aligned} \text{LHS} &= \overline{a \cdot b} + b && \text{De Morgan} \\ &= \bar{a} + \bar{b} + b && \text{Komplementierung} \\ &= \bar{a} + 1 && \text{Extremalgesetze} \\ &= 1 && \text{LHS} = \text{rhs} \end{aligned}$$

Now your turn!

c) $f := \neg((\neg x \wedge y) \vee (\neg y \wedge z)) \wedge (\neg x \wedge z)$ ist eine Kontradiktion.

De Morgan

$$= (\neg(\neg x \wedge y) \wedge \neg(\neg y \wedge z)) \wedge (\neg x \wedge z) \text{ De Morgan}$$

$$= ((x \vee \neg y) \wedge (y \vee \neg z)) \wedge (\neg x \wedge z) \text{ Ass. Komm.}$$

$$= ((x \vee \neg y) \wedge \neg x) \wedge ((y \vee \neg z) \wedge z) \text{ DS.}$$

$$= ((x \wedge \neg x) \vee (\neg y \wedge \neg x)) \wedge ((y \wedge z) \vee (\neg z \wedge z)) \text{ Komplement}$$

$$= (0 \vee (\neg y \wedge \neg x)) \wedge ((y \wedge z) \vee 0) \text{ Neutralität}$$

$$= (\neg y \wedge \neg x) \wedge (y \wedge z) \text{ Komm.}$$

$$= (\neg x \wedge \neg y) \wedge (y \wedge z) \text{ Ass.}$$

$$= \neg x \wedge \neg ((\neg y \wedge y) \wedge z) \text{ Komplement}$$

Erinnerung

$$\begin{aligned} &= \neg x \wedge \neg (0 \wedge z) \\ &= \neg x \wedge 0 \\ &= 0 \end{aligned}$$

d) Was ist der duale Ausdruck von $f = \neg((\neg x \wedge y) \vee (\neg y \wedge z)) \wedge (\neg x \wedge z)$. Was können wir über diesen dualen Ausdruck aussagen?

$$f^D = \neg((\neg x \vee y) \wedge (\neg y \vee z)) \vee (\neg x \vee z)$$

$$f^D = 0^D = 1 \rightarrow \text{Tautologie}$$

Der duale Ausdruck ergibt sich durch Vertauschung von \wedge und \vee sowie 0 und 1. Daher ist der duale Ausdruck von f :

$$f^D = \neg((\neg x \vee y) \wedge (\neg y \vee z)) \vee (\neg x \vee z)$$

In der vorigen Teilaufgabe haben wir gezeigt, dass f eine Kontradiktion ist, d.h. $f = 0$ eine wahre Aussage ist. Auf Grund der Dualität folgt unmittelbar, dass $f^D = 0^D = 1$ ebenfalls eine wahre Aussage ist, d.h. f^D ist eine Tautologie.

Hinweis: Beachte, dass $f^D = ((x \Rightarrow y) \wedge (y \Rightarrow z) \Rightarrow (x \Rightarrow z))$. Wir haben also auf etwas umständliche Weise die Transitivität der Implikation bewiesen.

Eine Menge $F \subset \mathcal{F}$ an Booleschen Funktionen heißt *funktional vollständig* wenn jede Boolesche Funktion sich als Komposition von Funktionen $f \in F$ sowie den Variablen x_1, \dots, x_n schreiben lässt. Man beachte, dass auch die Konstanten 0 und 1 als Funktionen in \mathcal{F} aufgefasst werden, nämlich diejenigen, die alle Variablenbelegungen konstant auf 0 oder 1 abbilden. Man kann zwar in der Praxis davon ausgehen, dass man beim Schaltungsentwurf immer Zugriff auf eine konstante 0- und 1-Leitung hat, in der Theorie der funktionalen Vollständigkeit bleibt dies jedoch unberücksichtigt. In der Vorlesung wurde bereits erwähnt, dass die Menge $\{\wedge, \vee, \neg, 0, 1\}$ funktional vollständig ist.

Zeige oder widerlege die funktionale Vollständigkeit der folgenden Mengen:

Hinweis: Du weißt bereits aus der Zentralübung, dass auch $\{\wedge, \neg\}$ funktional vollständig ist.

- {NOR}

$$\text{NOR}(a, a) = \neg(a \vee a) = \neg a$$

$$\text{NOR}(\text{NOR}(a, b), \text{NOR}(a, b))$$

$$= \neg(\neg(a \vee b) \vee \neg(a \vee b))$$

$$= \neg\neg(a \vee b) \wedge \neg\neg(a \vee b)$$

$$= (a \vee b) \wedge (a \vee b) = a \vee b$$

$\text{NOR}(\text{NOR}(a, a), \text{NOR}(b, b))$

$= \neg (\neg (a \vee a) \vee \neg (b \vee b))$

$= \neg \neg (a \vee a) \wedge \neg \neg (b \vee b)$

$= a \wedge b$

$$NOR(a, a) = \neg(a \vee a) = \neg a$$

$$NOR(NOR(a, b), NOR(a, b)) = \neg(\neg(a \vee b) \vee \neg(a \vee b)) = \neg\neg(a \vee b) \wedge \neg\neg(a \vee b) = (a \vee b) \wedge (a \vee b) = a \vee b$$

Mittels DeMorgan erhalten wir aus \vee und \neg auch \wedge und damit ist $\{NOR\}$ funktional vollständig.

Hinweis: In der Zentralübung haben wir gezeigt, dass $\{NAND\}$ funktional vollständig ist. Der Beweis war sehr ähnlich, was sich durch die Dualität von NAND und NOR begründet.

- $\{\oplus, 0, 1\}$

Mittels $\{\oplus, 0, 1\}$ kann eine Negation dargestellt werden: $a \oplus 1 = \neg a$. Allerdings ist $\{\oplus, 0, 1\}$ alleine nicht funktional vollständig wie folgender Beweis zeigt.

Sei $|f| = |\{a \mid f(a) = 1\}|$ die Anzahl der Belegungen bei der eine Boolese Funktion zu 1 evaluiert und seien $f, g : \{0, 1\}^2 \rightarrow \{0, 1\}$ zwei Boolese Funktionen über 2 Variablen. $f \oplus g$ ist weiterhin eine Funktion über 2 Variablen (speziell gilt $(f \oplus g)(a, b) = f(a, b) \oplus g(a, b)$). Nehmen wir an, dass $|f \oplus g|$ ungerade ist. Dann muss $|f \oplus g|$ entweder 1 oder 3 sein.

- Wenn $|f \oplus g| = 1$ gilt, müssen f und g bei drei Belegungen gleich und bei einer Belegung ungleich sein. Das geht direkt aus der Definition des \oplus hervor da $a \oplus b = 1$ genau dann wenn $a \neq b$. Dann folgt aber sofort, dass entweder $|f|$ oder $|g|$ ebenfalls ungerade ist. Bei den drei Belegungen bei denen f und g übereinstimmen, müssen f und g die selbe Anzahl an 1 Evaluationen haben, also auch die Parität der beiden Funktionen auf diesen drei Belegungen gleich sein. Da f und g bei der vierten Belegung ungleich sind, muss auch die Parität der beiden Funktionen unterschiedlich sein und somit ist entweder $|f|$ oder $|g|$ ungerade.
- Wenn $|f \oplus g| = 3$ gilt, müssen f und g bei einer Belegung gleich und bei drei Belegungen ungleich sein. Ein ähnliches Argument wie im ersten Fall zeigt, dass dies wiederum nur sein wenn entweder $|f|$ oder $|g|$ ebenfalls ungerade ist.

Damit gilt, dass $|f \oplus g|$ genau dann ungerade ist wenn genau entweder $|f|$ oder $|g|$ ungerade ist. Das heißt aber, dass wenn sowohl $|f|$ als auch $|g|$ gerade ist, $|f \oplus g|$ ebenfalls gerade sein muss.

Mit Anwendung nur eines \oplus auf 2 Variablen gibt es nur folgende Funktionen

$$\begin{array}{llll} f_1(a, b) = a \oplus b & f_2(a, b) = a \oplus a & f_3(a, b) = b \oplus b & f_4(a, b) = a \oplus 1 \\ f_5(a, b) = a \oplus 0 & f_6(a, b) = b \oplus 1 & f_7(a, b) = b \oplus 0. & \end{array}$$

Berechnung zeigt, dass $|f_i|$ für alle i gerade ist.

Insgesamt zeigt dies das rein durch \oplus nur Funktionen erzeugt werden die eine gerade Anzahl an 1 Evaluationen haben. Ein \wedge hat aber eine ungerade Anzahl an 1 Evaluationen. Das heißt $\{\oplus, 0, 1\}$ ist nicht funktional vollständig.

- $\{\neg, \leftrightarrow\}$



$$\leftrightarrow \equiv \neg(a \oplus b)$$

$$\equiv (a \oplus b) \oplus 1$$

Aus der vorigen Aufgabe geht sofort hervor, dass die Menge nicht funktional vollständig ist da $a \leftrightarrow b = \neg(a \oplus b) = (a \oplus b) \oplus 1$ und $\neg x = x \oplus 1$ gilt. Angenommen, $\{\neg, \leftrightarrow\}$ wäre funktional vollständig, dann würde daraus folgen, dass auch $\{\oplus, 1\}$ (und somit auch $\{\oplus, 0, 1\}$) funktional vollständig wäre. Dies widerspricht jedoch dem Ergebnis der vorigen Aufgabe. Daher ist $\{\neg, \leftrightarrow\}$ *nicht* funktional vollständig.

- {ITE, 0, 1}, wobei ITE: $\{0, 1\}^3 \rightarrow \{0, 1\}$ mit $\text{ITE}(x, y, z) := (x \wedge y) \vee (\neg x \wedge z)$.
if then else

$$\text{ITE}(x, 0, 1) = (x \wedge 0) \vee (\neg x \wedge 1) = 0 \vee \neg x = \neg x$$

$$\text{ITE}(x, y, 0) = (x \wedge y) \vee (\neg x \wedge 0) = (x \wedge y) \vee 0 = x \wedge y$$

Damit ist $\{\text{ITE}, 0, 1\}$ funktional vollständig.

Hinweis: Der ITE-Operator und die funktionale Vollständigkeit von $\{\text{ITE}, 0, 1\}$ wird uns bei der Synthese von binären Entscheidungsbäumen erneut begegnen.

Gegeben sind zwei 2-bit unsigned Integer $a = a_1a_0$ und $b = b_1b_0$. Es soll ein Schaltkreis entworfen werden der den Absolutwert der Differenz $y = y_1y_0 = |a - b|$ berechnet.

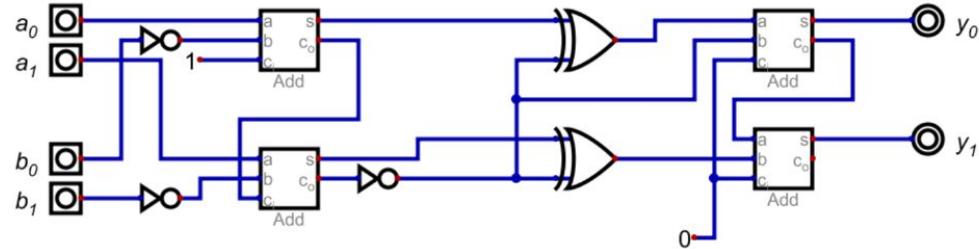
- Entwerfe den Schaltkreis unter Verwendung von Halb- und Volladdierern.
- Simuliere den Schaltkreis in Digital.



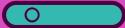
Wichtig für die HA :

Wie in der Vorlesung vorgestellt, kann ein Addierer als Subtrahierer verwendet werden indem man den Subtrahend invertiert und eine Konstante 1 als Carry-In im ersten Volladdierer verwendet. Dadurch kann das Zwischenergebnis $c = c_2c_1c_0 = a - b$ berechnet werden. *Vorsicht:* Um c zu berechnen haben wir a und b implizit als 3-Bit Zahlen im Zweierkomplement aufgefasst. wenn nur mit den 4 Eingabebits gerechnet wird und a und b nicht vorzeichenweiterweitert werden, muss darauf geachtet werden, dass das Vorzeichenbit c_2 richtig gesetzt wird. Deshalb wird das Carry-Out in der folgenden Lösung am Ende invertiert. Das Vorzeichen vom Zweierkomplement von b wäre ja 1.

Um den Absolutbetrag von c_1c_0 zu berechnen, muss das Zweierkomplement genau dann berechnet werden wenn $c_2 = 1$. Das Erreichen wir ebenfalls mit dem Zweierkomplement-Trick aus der Vorlesung.



- Betrachte wie y_0 von den Eingängen abhängt. Fällt dir eine einfache Schaltung ein die y_0 direkt berechnet?



$$y_0 = a_0 \oplus b_0$$