Tutorat 5

Logikfunktionen, Formale Beschreibung von Schaltkreisen, Boolesche Algebra

Gruppe 9

Präsentator:
Jürgen Mattheis
(juergmatth@gmail.com)

Vorlesung von:
Prof. Dr. Scholl

Übungsgruppenbetreuung:
Tobias Seufert

11. Juni 2023

Universität Freiburg, Lehrstuhl für Rechnerarchitektur

Gliederung

Aufgabe 1

Aufgabe 1 I

Logikfunktionen

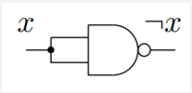
Lösung 1.1

x	y	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}	f_{15}	f_{16}
0	0	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
0	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	0	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
		$x \wedge \neg x$	x NOR y, $\neg x \wedge \neg y$	$\neg x \wedge y$	$\neg x$	$x \wedge \neg y$	$\neg y$	x XOR y, $(\neg x \wedge y) \vee (x \wedge \neg y)$	x NAND y, $\neg x \vee \neg y$	$x \wedge y$	x XNOR y, $(x \wedge y) \vee (\neg x \wedge \neg y)$	BUF(y)	$\neg x \vee y$	BUF(x)	$x \vee \neg y$	$x \vee y$	$x \vee \neg x$

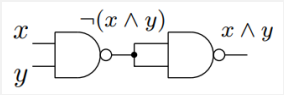
Aufgabe 1 II

Logikfunktionen

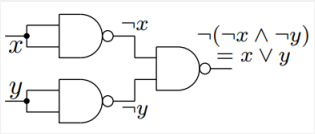
Lösung 1.2



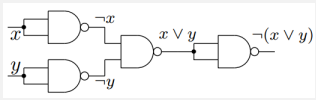
(a) NOT



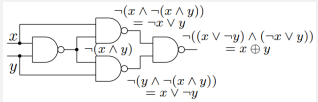
(b) AND



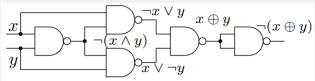
(c) OR



(d) NOR



(e) XOR



(f) XNOR

Aufgabe 1 III

Logikfunktionen

Anmerkungen

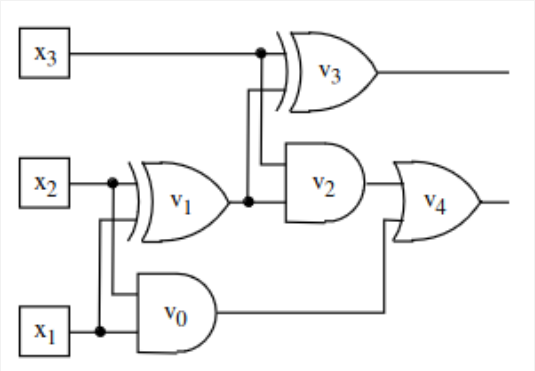
- Aufgaben a) und b) zusammen zeigen, dass man mit Hilfe des NAND_2 alle Funktionen $f : \mathbb{B}^2 \to \mathbb{B}$ realisieren kann.

Aufgabe 2

Aufgabe 2 I

Formale Beschreibung von Schaltkreisen

Lösung 2.1



Aufgabe 2 II

Formale Beschreibung von Schaltkreisen

Lösung 2.2

x_1	x_2	x_3	v_3	v_4
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

Aufgabe 3

Aufgabe 3 I

Boolesche Algebra

Lösung 3.1

x	y	$\neg x$	$\neg y$	$\neg x \vee \neg y$	$x \wedge y$	$\neg(x \wedge y)$	$\neg x \wedge \neg y$	$x \vee y$	$\neg(x \vee y)$
0	0	1	1	1	0	1	1	0	1
0	1	1	0	1	0	1	0	1	0
1	0	0	1	1	0	1	0	1	0
1	1	0	0	0	1	0	0	1	0

Appendix

Appendix I

Radix-Komplement

- Das Radix-Komplement (r's Komplement) von einer Zahl y mit $n-$Ziffern mit Radix b:

$$b^n - y$$

- Es gibt auch ein verringertes Radix-Komplement ((r-1)'s Komplement), und zwar:

$$(b^n - 1) - y$$

[2]

Appendix II

Radix-Komplement

- Komplementärverfahren (engl. Method of complements[1]): Subtraktion als Addition berechnen. Beispiele sind:

$$\begin{aligned} 622_{10} - 451_{10} \text{ ist } 622_{10} + (1000_{10} - 451_{10}) - 1000_{10} &= 622_{10} + (999_{10} - 451_{10} + 1) - 1000_{10} \\ &= 622_{10} + 549_{10} - 1000_{10} \\ &= 1171_{10} - 1000_{10} = 171_{10} \end{aligned}$$

$$\begin{aligned} 110_2 - 011_2 \text{ ist } 110_2 + (1000_2 - 011_2) - 1000_2 &= 110_2 + (999_2 - 011_2 + 1) - 1000_2 \\ &= 110_2 + 101_2 - 1000_2 \\ &= 1011_2 - 1000_2 = 11_2 \end{aligned}$$

Appendix III

Radix-Komplement

Anmerkungen

- Das verringerte Radix-Komplement kann man leicht erhalten, indem man einfach die Ziffern einer Zahl mit den Ziffern, die man benötigt um $Radix - 1$ zu erhalten, ersetzt. Zum Beispiel, dass verringerte Radix-Komplement für die 2-Ziffern Dezimalzahl 56 ist 43. Man kann das Radix-Komplement einfach erhalten, indem man Eins zu dem verringerten Radix-Komplement addiert $43 + 1 = 44$
- Im Dezimalzahlensystem ist das Radix-Komplement auch als Zehnerkomplement (10'er Komplement) und das verringerte Radix-Komplement als Neunerkomplement (9'er Komplement) bekannt
- Im Binärsystem ist das Radix-Komplement als Zweierkomplement (2'er Komplement) und das verringerte Radix-Komplement als Einerkomplement (1'er Komplement) bekannt. Ein Einerkomplement kann man einfach durch das Umkehren von Bits einer Zahl erhalten. Zweierkomplemente werden in Computern für die Darstellung von negativen Ganzzahlen verwendet

Appendix I

Halbaddierer

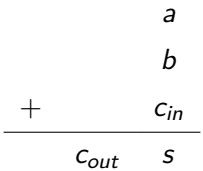
$$\begin{array}{rcl} & & a \\ + & & b \\ \hline & c_{out} & s \end{array}$$

a	b	c_{out}	s
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

- der HA kann 2 Inputs zusammenrechen, daher sind die möglichen Outputs 00, 01 und 10
- ein HA ist einfach nur die direkte Umsetzung von $a + b = c_{out} \cdot 2^1 + s \cdot 2^0$ nach der obigen Tabelle:

Appendix I

Volladdierer



c_{in}	a	b	c_{out}	s
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

- der FA kann 3 Inputs zusammenrechnen, daher sind die möglichen Outputs 00, 01 10 und 11

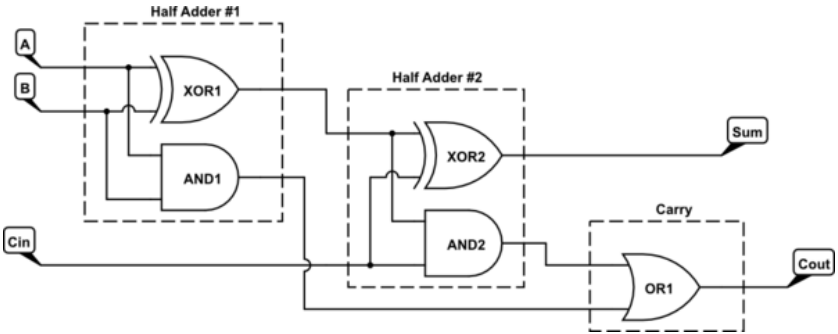
Appendix II

Volladdierer

- ein FA ist einfach nur die Umsetzung von $a + b + c_{in} = c_{out} \cdot 2^1 + s \cdot 2^0$, ein HA übernimmt $a + b$ und der andere HA übernimmt $s_{a+b} + c_{in}$, wobei es entweder beim ersten HA oder beim zweiten HA die Möglichkeit gibt, dass ein Übertrag entsteht
- ob die beiden Überträge am Ende ge$\oplus$rt oder ge$\vee$rt werden spielt keine Rolle, da es sowieso niemals vorkommen kann, dass beide HA einen Übertrag haben
  - beim ersten HA kommt es zu einem Übertrag, wenn a und b beide 1 sind
  - beim zweiten HA kommt es zu einem Übertrag, wenn s und c_{in} beide 1 sind
  - wenn der erste HA einen Übertrag hat, bedeutet es für den zweiten, dass $s = 0$ sein muss es somit beim zweiten zu keinem Übertrag kommen kann. Bei HA kann es nicht vorkommen, dass $c_{out} = 1$ und $s = 1$
  - wenn der zweite HA einen Übertrag hat, bedeutet es für den ersten, dass bei diesem $s = 1$ ist und dieser daher keinen Übertrag haben kann

Appendix III

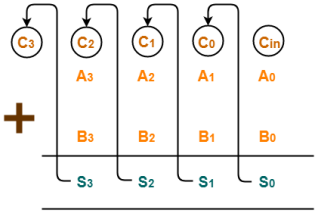
Volladdierer



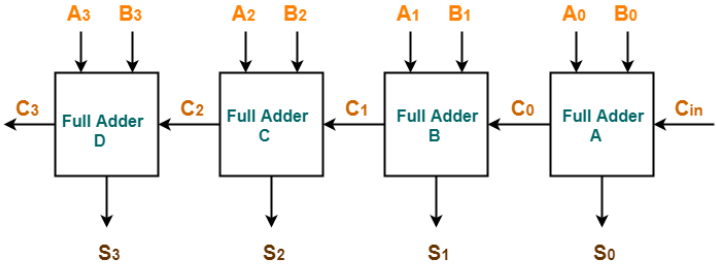
- es spielt bei FA bei der Verwendung keine Rolle, welches der 3 Inputs eigentlich für das Carry vorgesehen ist

Appendix I

4-Bit Addierer



Adding two 4-bit Numbers



4-bit Ripple Carry Adder

- FA FA FA HA, wobei man allerdings meistens FA FA FA FA herstellt und das carry vom letzten FA unberührt lässt aber theoretisch Addierer zu größeren Addierern zusammenfügen kann

Literatur

Online

[1] *Method of Complements*. Wikipedia. 29. März 2023. URL: https://en.wikipedia.org/w/index.php?title=Method_of_complements&oldid=1147184194 (besucht am 08. 06. 2023).

[2] *Online-Rechner: Numerische Komplemente*. URL: https://de.planetcalc.com/8574/ (besucht am 08. 06. 2023).