

## Inhaltsverzeichnis

I.	Übung 1 Zahlenkomparator .....	2
a.)	Wahrheitstabelle.....	2
b.)	KV-Diagramm .....	2
c.)	Analyse .....	2
d.)	Aufbau der Schaltung.....	3
II.	Übung 2 8421 / 7-Segment Codeumsetzer .....	4
a.)	Wahrheitstabelle.....	4
b.)	KV-Diagramm .....	4
c.)	Analyse .....	5
d.)	Aufbau der Schaltung.....	6
III.	Übung 3 Volladdierer (1 Bit) .....	7
a.)	Wahrheitstabelle.....	7
b.)	KV-Diagramm (AND/OR) .....	7
c.)	Analyse (AND/OR) .....	7
d.)	KV-Diagramm (NAND) .....	8
e.)	Analyse (NAND).....	8
f.)	Aufbau der Schaltung (AND/OR).....	9
a.)	Aufbau der Schaltung (NAND).....	10

# I. Übung 1 Zahlenkomparator

Eine digitale Schaltung, mit der zwei Dualzahlen auf Gleich- oder Ungleichheit geprüft werden, wird als Zahlenkomparator bezeichnet. In der Übung wird eine Schaltung entwickelt, die in der Lage ist, zwei Dualzahlen P und Q mit jeweils 2 Bit zu vergleichen. Im Falle der Ungleichheit erfolgt die Ausgabe über ein Größer-Kleiner Signal, mittels LED.

## a.) Wahrheitstabelle

Eingänge				Ausgänge		
Zahl P		Zahl Q		P > Q	P = Q	P < Q
A	B	C	D	Y1	Y2	Y3
0	0	0	0	0	1	0
0	0	0	1	0	0	1
0	0	1	0	0	0	1
0	0	1	1	0	0	1
0	1	0	0	1	0	0
0	1	0	1	0	1	0
0	1	1	0	0	0	1
0	1	1	1	0	0	1
1	0	0	0	1	0	0
1	0	0	1	1	0	0
1	0	1	0	0	1	0
1	0	1	1	0	0	1
1	1	0	0	1	0	0
1	1	0	1	1	0	0
1	1	1	0	0	0	1
1	1	1	1	0	1	0

## b.) KV-Diagramm

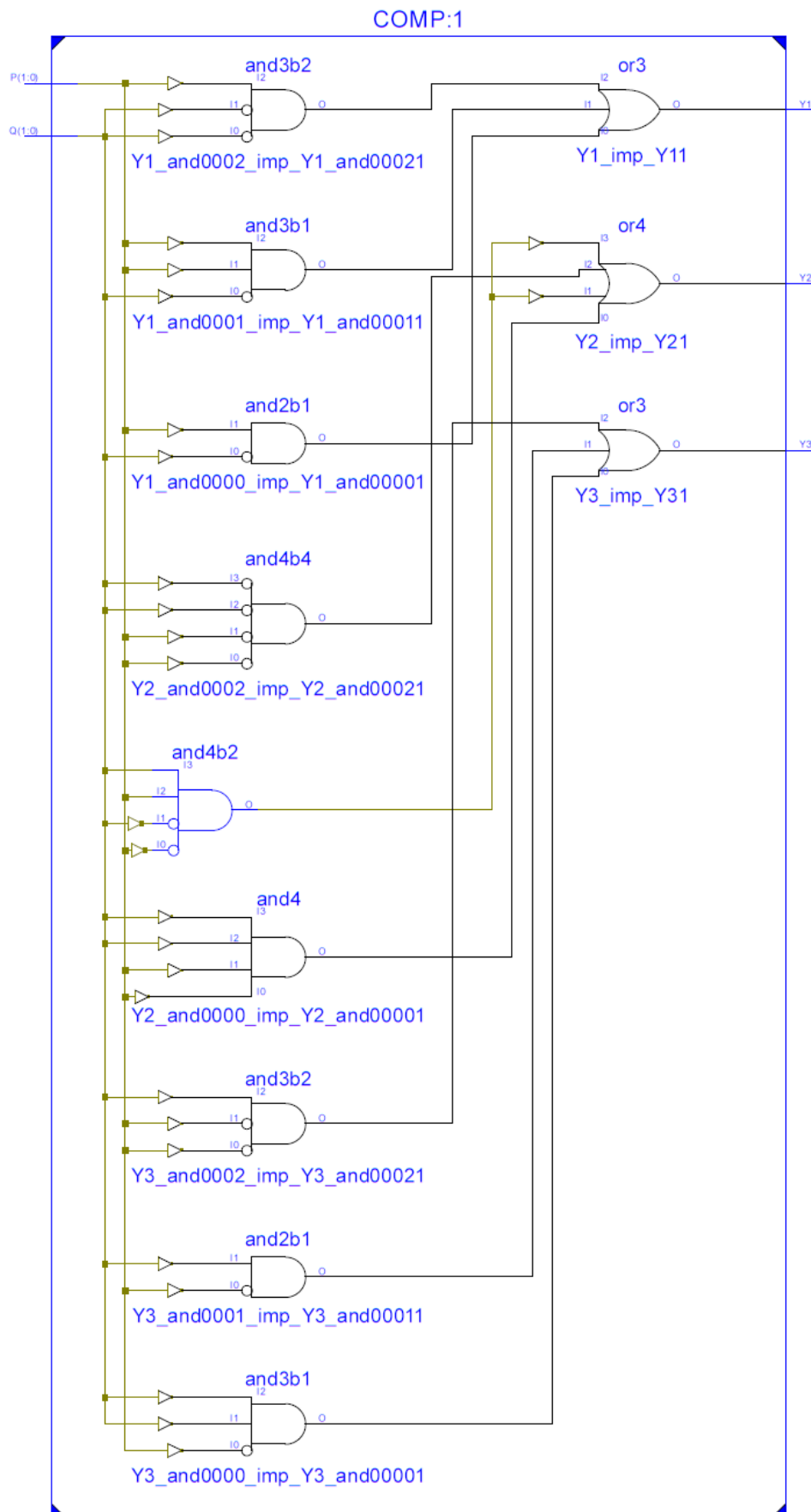
Y <sub>1</sub>	Y <sub>2</sub>	Y <sub>3</sub>
(I <sub>3</sub> =C; I <sub>2</sub> =D; I <sub>1</sub> =B; I <sub>0</sub> =A)	(I <sub>3</sub> =D; I <sub>2</sub> =B; I <sub>1</sub> =C; I <sub>0</sub> =A)	(I <sub>3</sub> =A; I <sub>2</sub> =B; I <sub>1</sub> =D; I <sub>0</sub> =C)
 ROT = 0 / GRÜN = 1	 ROT = 0 / GRÜN = 1	 ROT = 0 / GRÜN = 1
$(A \wedge \bar{C}) \vee (A \wedge B \wedge \bar{C}) \vee (B \wedge \bar{C} \wedge \bar{D})$ $(I_0 \wedge \bar{I}_3) \vee (I_0 \wedge I_1 \wedge \bar{I}_2) \vee (I_1 \wedge \bar{I}_2 \wedge \bar{I}_3)$	$(A \wedge B \wedge C \wedge D) \vee (A \wedge \bar{B} \wedge C \wedge \bar{D}) \vee (\bar{A} \wedge \bar{B} \wedge \bar{C} \wedge \bar{D}) \vee (\bar{A} \wedge B \wedge \bar{C} \wedge D)$ $(I_0 \wedge I_1 \wedge I_2 \wedge I_3) \vee (I_0 \wedge \bar{I}_2 \wedge I_1 \wedge \bar{I}_3) \vee (\bar{I}_0 \wedge \bar{I}_1 \wedge \bar{I}_2 \wedge \bar{I}_3) \vee (\bar{I}_0 \wedge I_2 \wedge \bar{I}_1 \wedge I_3)$	$(\bar{A} \wedge C) \vee (\bar{B} \wedge C \wedge D) \vee (\bar{A} \wedge \bar{B} \wedge D)$ $(\bar{I}_3 \wedge I_0) \vee (\bar{I}_2 \wedge I_0 \wedge I_1) \vee (\bar{I}_2 \wedge \bar{I}_3 \wedge I_1)$

## c.) Analyse

Die Minimierung und anschließende Simulation der Schaltung führte zu dem gewünschten Ergebnis. Es kann jeweils nur ein Ausgangssignal aktiv (HIGH) sein.

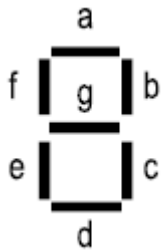


## d.) Aufbau der Schaltung

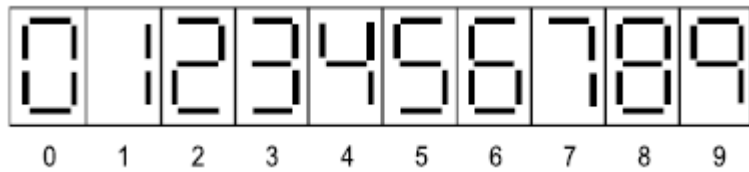


## II. Übung 2 8421 / 7-Segment Codeumsetzer

Der BCD-Code wird in großem Umfang angewendet. Häufig werden BCD-kodierte Informationen über 7-Segment-Anzeigeeinheiten ausgegeben. Die Kathode (x) der 7-Segment-Anzeige ist dabei auf LOW zu ziehen.



Segment Bezeichnungen



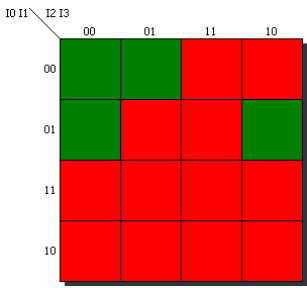
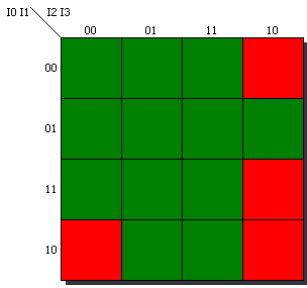
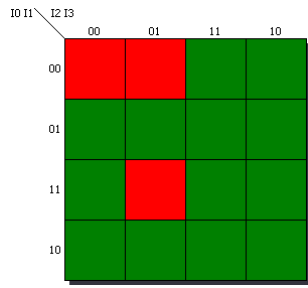
Numerische Bezeichnungen

### a.) Wahrheitstabelle

Eingänge				Ziffer	Ausgänge						
8 D	4 C	2 B	1 A		a	b	c	d	e	f	g
0	0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	1	0	1	1	0	0	0	0
0	0	1	0	2	1	1	0	1	1	0	1
0	0	1	1	3	1	1	1	1	0	0	1
0	1	0	0	4	0	1	1	0	0	1	1
0	1	0	1	5	1	0	1	1	0	1	1
0	1	1	0	6	1	0	1	1	1	1	1
0	1	1	1	7	1	1	1	0	0	0	0
1	0	0	0	8	1	1	1	1	1	1	1
1	0	0	1	9	1	1	1	0	0	1	1
1	0	1	0	A	x	x	x	x	x	x	x
1	0	1	1	B	x	x	x	x	x	x	x
1	1	0	0	C	x	x	x	x	x	x	x
1	1	0	1	D	x	x	x	x	x	x	x
1	1	1	0	E	x	x	x	x	x	x	x
1	1	1	1	F	x	x	x	x	x	x	x

### b.) KV-Diagramm

A	B	C	D
$(I_3=D; I_2=C; I_1=A; I_0=B)$  <b>ROT = 0 / GRÜN = 1</b>	$(I_3=D; I_2=C; I_1=B; I_0=A)$  <b>ROT = 0 / GRÜN = 1</b>	$(I_3=A; I_2=D; I_1=B; I_0=C)$  <b>ROT = 0 / GRÜN = 1</b>	$(I_3=D; I_2=A; I_1=C; I_0=B)$  <b>ROT = 0 / GRÜN = 1</b>
$(A \vee B \vee \bar{C} \vee D) \wedge$ $(\bar{A} \vee B \vee C \vee D)$ $(I_0 \vee (I_1 \wedge I_2) \vee (I_3 \vee (\bar{I}_1 \wedge \bar{I}_2)))$	$(\bar{A} \vee B \vee \bar{C} \vee D) \wedge$ $(A \vee \bar{B} \vee \bar{C} \vee D)$ $(\bar{I}_0 \wedge I_1) \vee (I_0 \wedge I_1) \vee I_3 \vee \bar{I}_2$	$A \vee \bar{B} \vee C \vee D$ $I_0 \vee I_2 \vee I_3 \vee \bar{I}_1$	$(A \vee B \vee \bar{C} \vee D) \wedge$ $(\bar{A} \vee B \vee C) \wedge$ $(\bar{A} \vee \bar{B} \vee \bar{C} \vee D)$ $(I_0 \wedge \bar{I}_2) \vee (I_0 \wedge \bar{I}_1) \vee$ $(\bar{I}_0 \wedge I_1 \wedge I_2) \vee (\bar{I}_1 \wedge \bar{I}_2)$

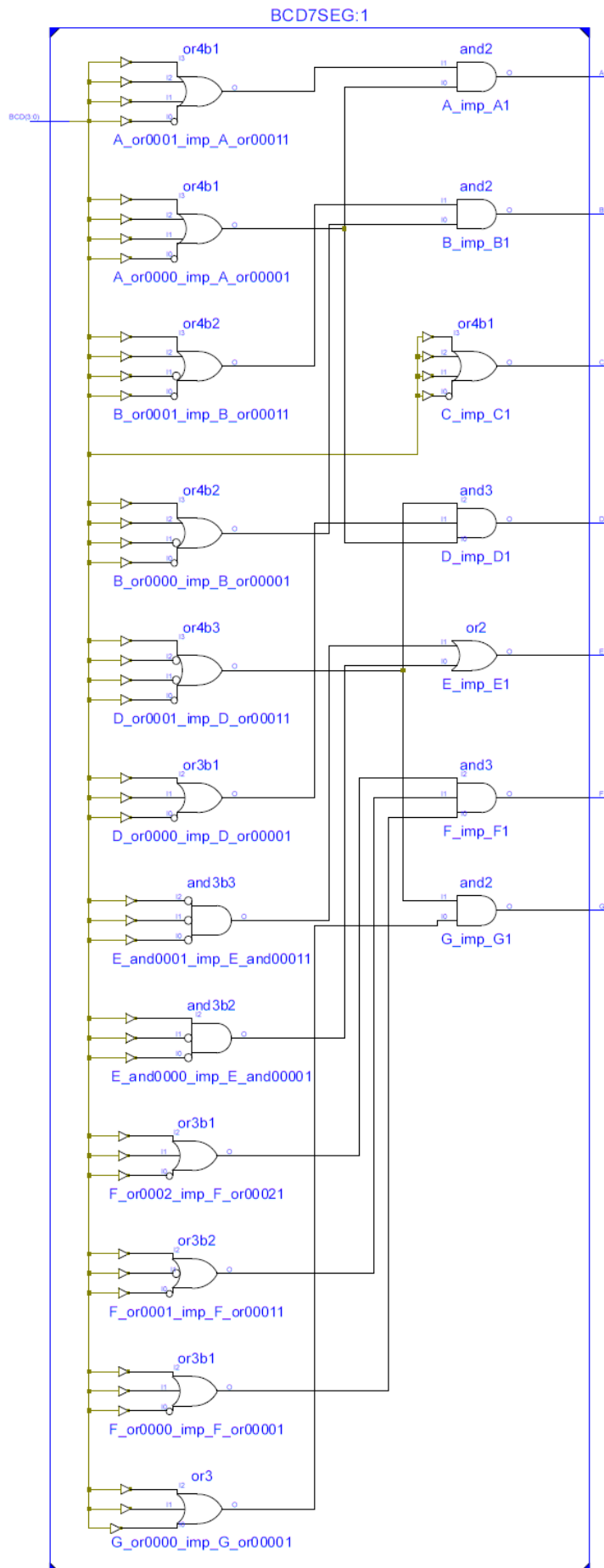
E	F	G
$(I_3=D; I_2=C; I_1=B; I_0=A)$	$(I_3=D; I_2=A; I_1=C; I_0=B)$	$(I_3=A; I_2=D; I_1=B; I_0=C)$
		
ROT = 0 / GRÜN = 1	ROT = 0 / GRÜN = 1	ROT = 0 / GRÜN = 1
$(\bar{A} \wedge B \wedge \bar{D}) \vee (\bar{A} \wedge \bar{B} \wedge \bar{C})$	$(\bar{A} \vee C \vee D) \wedge$ $(\bar{A} \vee \bar{B} \vee D) \wedge$ $(\bar{B} \vee C \vee D)$	$(B \vee C \vee D) \wedge (\bar{A} \vee \bar{B} \vee \bar{C} \vee D)$
$(\bar{I}_0 \wedge I_1 \wedge \bar{I}_3) \vee (\bar{I}_0 \wedge \bar{I}_1 \wedge \bar{I}_2)$	$(I_1 \wedge \bar{I}_3) \vee (\bar{I}_0 \wedge I_1) \vee$ $I_3 \vee (I_0 \wedge \bar{I}_2)$	$(\bar{I}_0 \wedge I_1) \vee I_2 \vee (I_0 \wedge \bar{I}_1) \vee$ $(I_0 \wedge \bar{I}_3)$

### c.) Analyse

Der Analyse ist zu entnehmen das die KV-Minimierung für die gewünschten Zustände funktioniert.

bcd[3:0]	1001	0000	0001	0010	0011	0100	0101	0110	0111	1000
a	1									
b	1									
c	1									
d	0									
e	0									
f	1									
g	1									

## d.) Aufbau der Schaltung



### III. Übung 3 Volladdierer (1 Bit)

Der Volladdierer ist eines der grundlegenden Elemente in der ALU (Arithmetic Logic Unit) eines Mikroprozessors. Ein Nachteil des Volladdierer besteht darin, dass der Übertrag jeder einzelnen Addierer Stufe bis zum Ende der Berechnung weitergereicht werden muss. Daher werden bei Steigender Bit Zahl auch proportional mehrere Taktzyklen benötigt. Dies kann durch Paralleladdierer (Carry-Lookahead-Addierer) mit Übertragsvorausberechnung und anderen Funktionen verbessert werden.

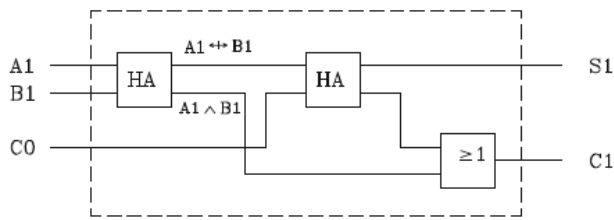


Abb.: 1: Volladdierer

Der Volladdierer besteht aus zwei Halbaddierern die wiederum in verschiedenen Technologien AND, OR, XOR oder NAND bestehen können. In der Übung ist der Addierer vorerst zu berechnen und anschließend in NAND Technologie umzusetzen.

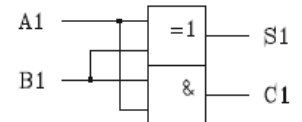


Abb.: 2: Halbaddierer

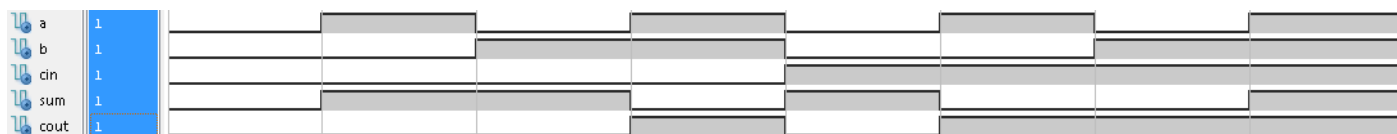
#### a.) Wahrheitstabelle

Eingänge			Ausgänge	
C <sub>in</sub>	B	A	SUM Σ	C <sub>out</sub>
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

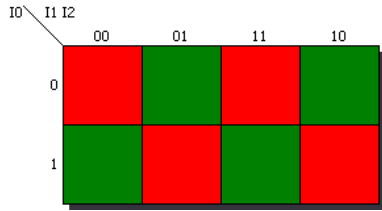
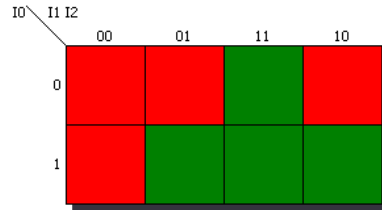
#### b.) KV-Diagramm (AND/OR)

SUM	C <sub>out</sub>
<p>(I<sub>2</sub>=C<sub>in</sub>; I<sub>1</sub>=B; I<sub>0</sub>=A)</p> <p>ROT = 0 / GRÜN = 1</p> <p> <math>(A \wedge B \wedge C_{in}) \vee (A \wedge \overline{B} \wedge \overline{C_{in}}) \vee</math>  <math>(\overline{A} \wedge \overline{B} \wedge C_{in}) \vee (\overline{A} \wedge B \wedge \overline{C_{in}})</math>  <math>(\overline{I_0} \wedge I_1 \wedge \overline{I_2}) \vee (\overline{I_0} \wedge \overline{I_1} \wedge I_2) \vee</math>  <math>(I_0 \wedge I_1 \wedge I_2) \vee (I_0 \wedge \overline{I_1} \wedge \overline{I_2})</math> </p>	<p>(I<sub>2</sub>=A; I<sub>1</sub>=C<sub>in</sub>; I<sub>0</sub>=B)</p> <p>ROT = 0 / GRÜN = 1</p> <p> <math>(A \wedge C_{in}) \vee (A \wedge B) \vee (B \wedge C_{in})</math>  <math>(I_0 \wedge I_2) \vee (I_1 \wedge I_2) \vee (I_0 \wedge I_1)</math> </p>

#### c.) Analyse (AND/OR)



## d.) KV-Diagramm (NAND)

SUM	Cout
<p><math>(I_2=C_{in}; I_1=B; I_0=A)</math></p>  <p>ROT = 0 / GRÜN = 1</p> $\overline{(A \wedge B \wedge C_{in})} \wedge \overline{(A \wedge \overline{B} \wedge \overline{C_{in}})} \wedge \overline{(\overline{A} \wedge \overline{B} \wedge C_{in})} \wedge \overline{(\overline{A} \wedge B \wedge \overline{C_{in}})}$ $(\overline{I_0} \wedge I_1 \wedge \overline{I_2}) \vee (\overline{I_0} \wedge \overline{I_1} \wedge I_2) \vee (I_0 \wedge I_1 \wedge I_2) \vee (I_0 \wedge \overline{I_1} \wedge \overline{I_2})$	<p><math>(I_2=A; I_1=C_{in}; I_0=B)</math></p>  <p>ROT = 0 / GRÜN = 1</p> $\overline{(A \wedge B \wedge C_{in})} \wedge \overline{(A \wedge B)} \wedge \overline{(B \wedge C_{in})}$ $(I_0 \wedge I_2) \vee (I_1 \wedge I_2) \vee (I_0 \wedge I_1)$

Es ist nicht möglich  $(A \text{ nand } B \text{ nand } C_{in})$  durchzuführen, somit wird eine Eingabe mit  $\text{not}(\text{not}(A \text{ and } B \text{ and } C_{in}))$  automatisch wieder zurückgekürzt, siehe Ergebnis im Kapitel KV-Diagramm (AND/OR).

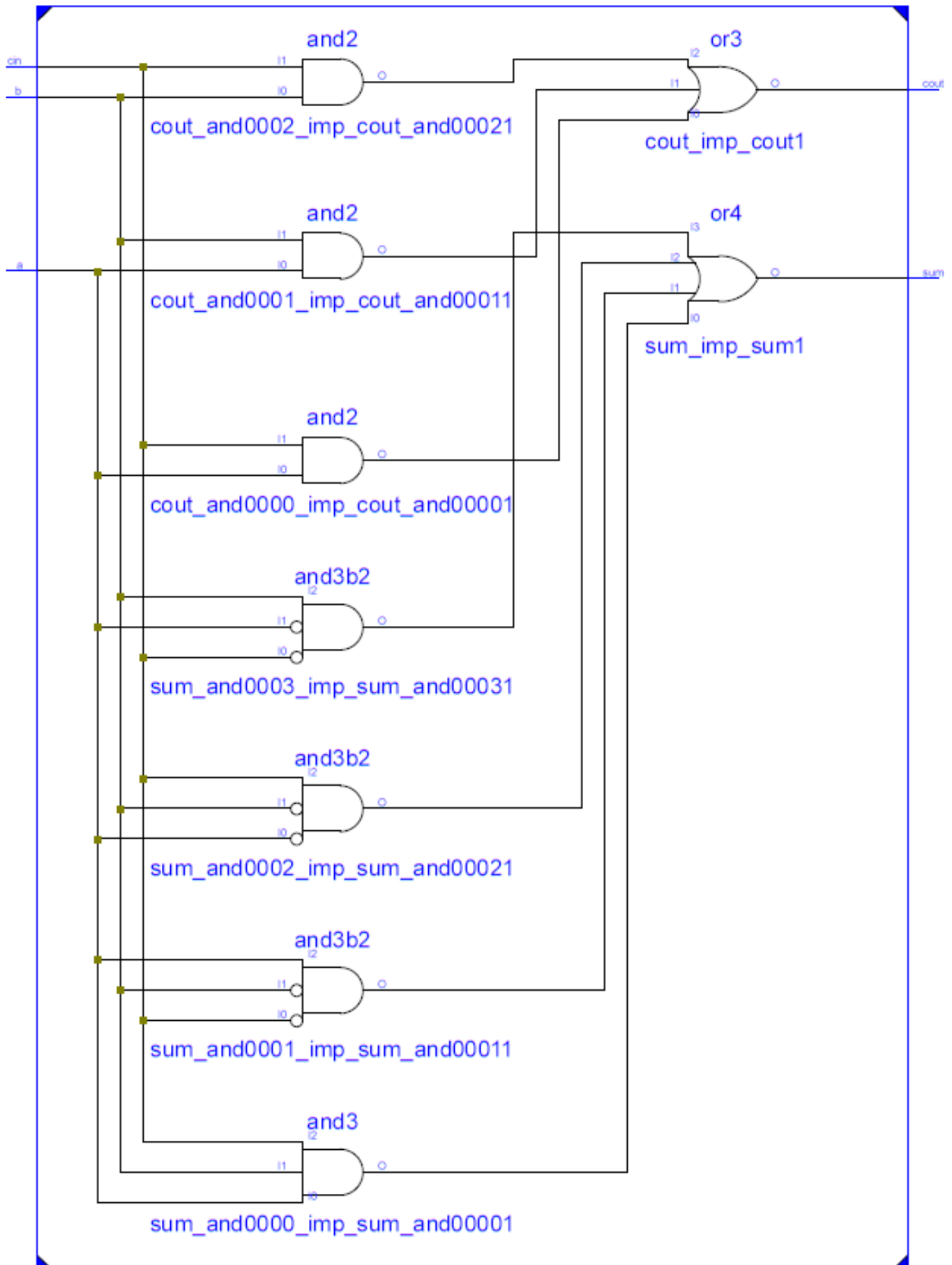
## e.) Analyse (NAND)

a	1								
b	1								
cin	1								
sum	1								
cout	1								



## f.) Aufbau der Schaltung (AND/OR)

addSUM:1



## a.) Aufbau der Schaltung (NAND)

