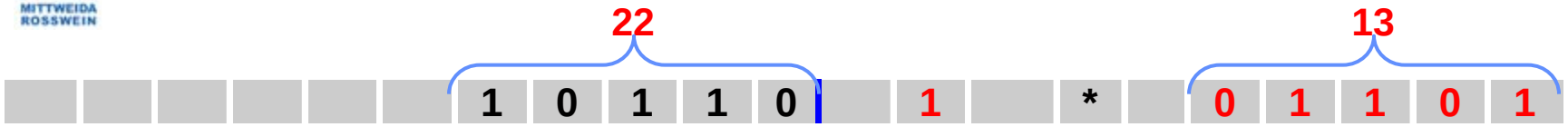
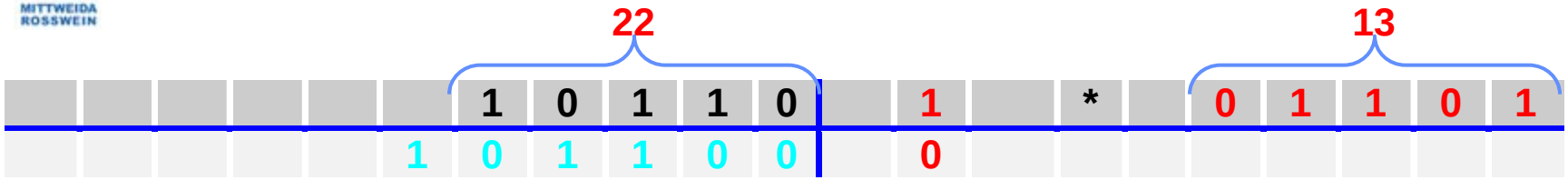
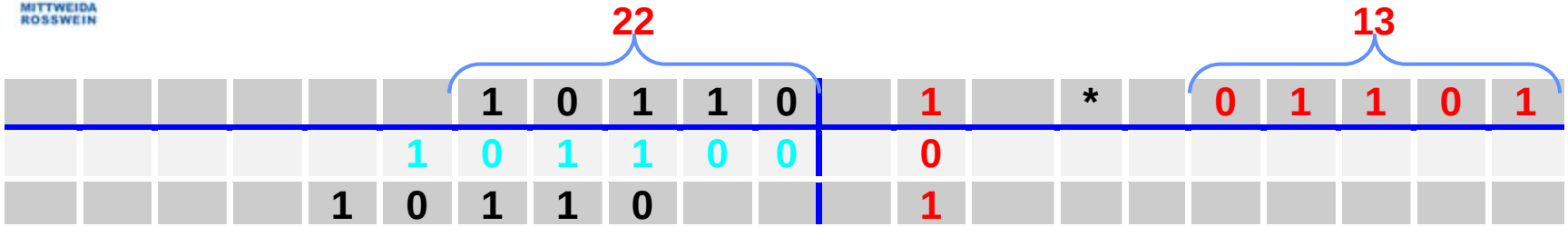
Multiplikation von 2 Dualzahlen (n = 5 Bit)



Multiplikation von 2 Dualzahlen (n = 5 Bit)



Multiplikation von 2 Dualzahlen (n = 5 Bit)



Multiplikation von 2 Dualzahlen (n = 5 Bit)

						1	0	1	1	0		1		*		0	1	1	0	1
					1	0	1	1	0	0		0								
				1	0	1	1	0				1								
			1	0	1	1	0					1								

22

13

Multiplikation von 2 Dualzahlen (n = 5 Bit)

						1	0	1	1	0		1		*		0	1	1	0	1
					1	0	1	1	0	0		0								
				1	0	1	1	0				1								
			1	0	1	1	0					1								
		1	0	1	1	0						0								

22

13

Multiplikation von 2 Dualzahlen (n = 5 Bit)

22 * 13

						1	0	1	1	0		1		*		0	1	1	0	1
					1	0	1	1	0	0		0								
				1	0	1	1	0				1								
			1	0	1	1	0					1								
		1	0	1	1	0						0								

Multiplikation von 2 Dualzahlen (n = 5 Bit)

22 13

						1	0	1	1	0		1		*		0	1	1	0	1
					1	0	1	1	0	0		0								
				1	0	1	1	0				1								
			1	0	1	1	0					1								
		1	0	1	1	0						0								
										0										

Multiplikation von 2 Dualzahlen (n = 5 Bit)

22 * 13

						1	0	1	1	0		1		*		0	1	1	0	1
					1	0	1	1	0	0		0								
				1	0	1	1	0				1								
			1	0	1	1	0					1								
		1	0	1	1	0						0								
									1	0										

Multiplikation von 2 Dualzahlen (n = 5 Bit)

22 13

						1	0	1	1	0		1		*		0	1	1	0	1
					1	0	1	1	0	0		0								
				1	0	1	1	0				1								
			1	0	1	1	0					1								
		1	0	1	1	0						0								
								1	1	0										

Multiplikation von 2 Dualzahlen (n = 5 Bit)

						1	0	1	1	0		1		*		0	1	1	0	1
					1	0	1	1	0	0		0								
				1	0	1	1	0				1								
			1	0	1	1	0					1								
		1	0	1	1	0						0								
						1	1	1	0											

22

13

Multiplikation von 2 Dualzahlen (n = 5 Bit)

22 * 13

						1	0	1	1	0		1		*		0	1	1	0	1
					1	0	1	1	0	0		0								
				1	0	1	1	0				1								
			1	0	1	1	0					1								
		1	0	1	1	0						0								
						1	1	1	1	0										

Multiplikation von 2 Dualzahlen (n = 5 Bit)

22 13

						1	0	1	1	0		1		*		0	1	1	0	1
					1	0	1	1	0	0		0								
				1	0	1	1	0				1								
			1	0	1	1	0					1								
		1	0	1	1	0						0								
					0	1	1	1	1	0										

Multiplikation von 2 Dualzahlen (n = 5 Bit)

22 13

						1	0	1	1	0		1		*		0	1	1	0	1
					1	0	1	1	0	0		0								
				1	0	1	1	0				1								
			1	0	1	1	0					1								
		1	0	1	1	0						0								
				0	0	1	1	1	1	0										

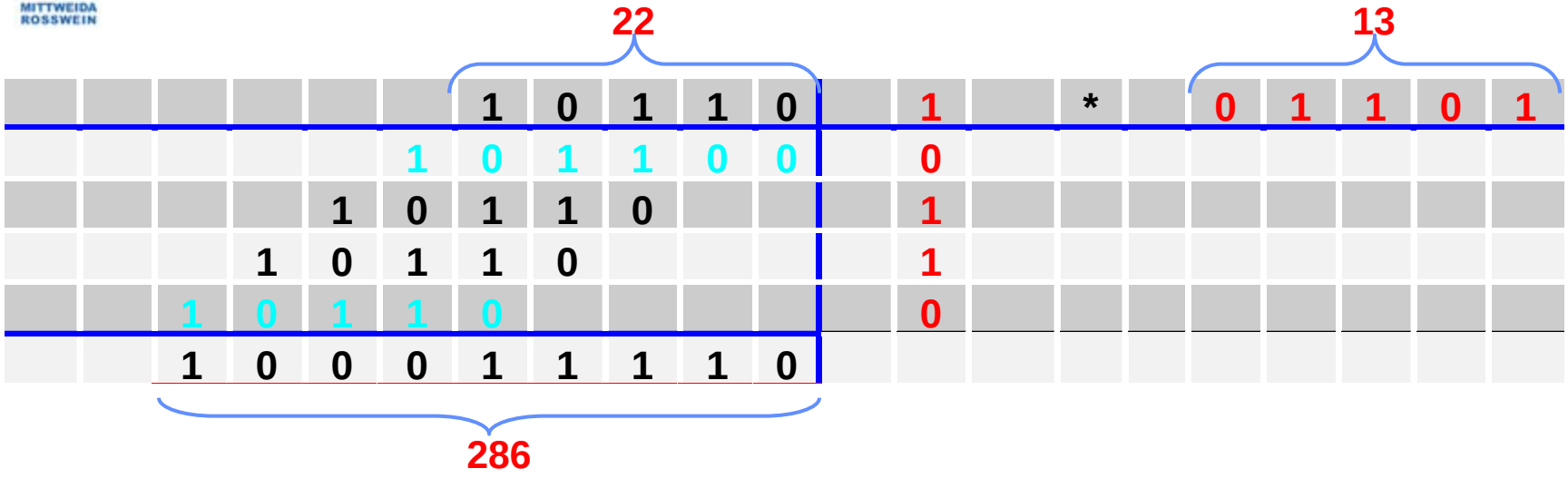
Multiplikation von 2 Dualzahlen (n = 5 Bit)

						1	0	1	1	0		1		*		0	1	1	0	1
					1	0	1	1	0	0		0								
				1	0	1	1	0				1								
			1	0	1	1	0					1								
		1	0	1	1	0						0								
			0	0	0	1	1	1	1	0										

22

13

Multiplikation von 2 Dualzahlen (n = 5 Bit)

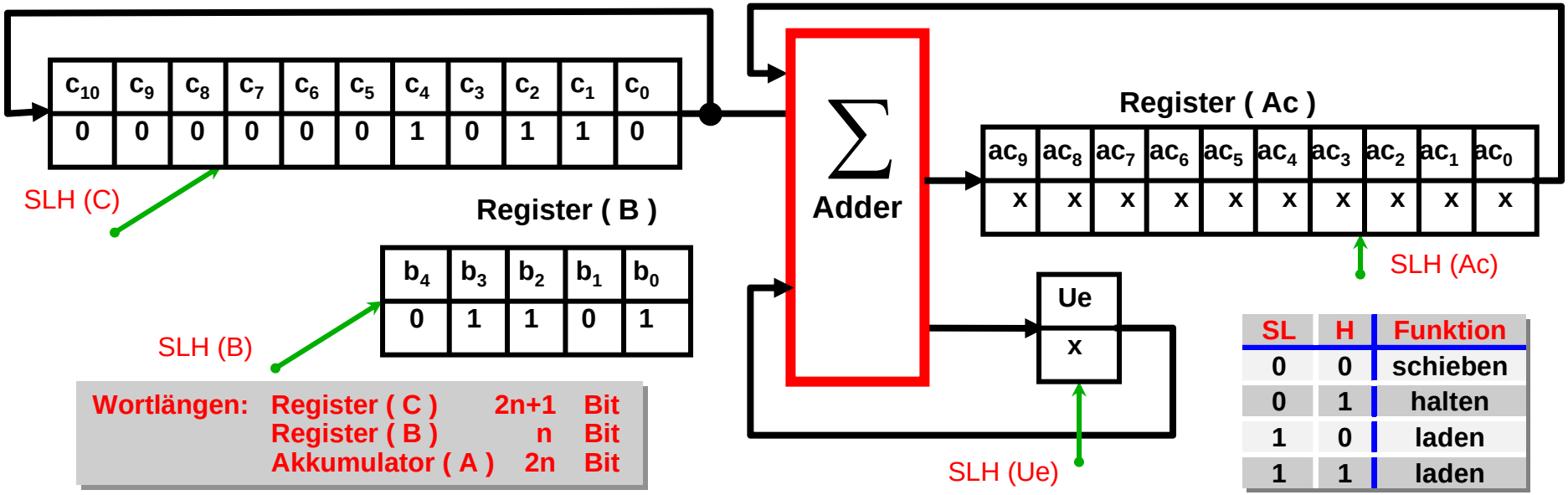


Multiplikation von 2 Dualzahlen (n = 5 Bit)

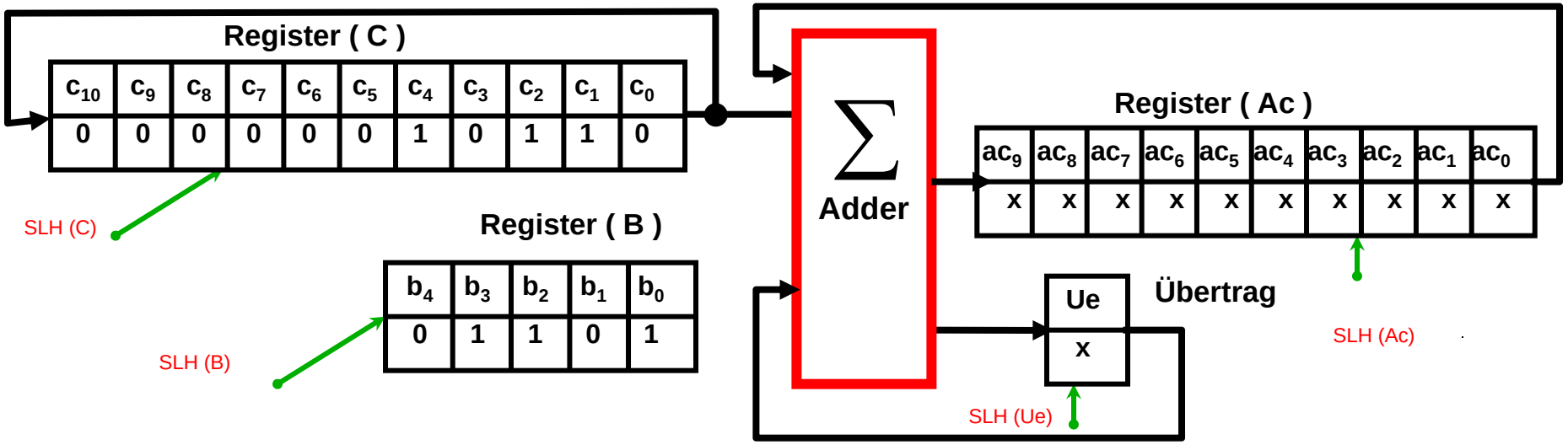
						1	0	1	1	0		1		*		0	1	1	0	1
					1	0	1	1	0	0		0								
				1	0	1	1	0				1								
			1	0	1	1	0					1								
		1	0	1	1	0						0								
		1	0	0	0	1	1	1	1	0										

22 13

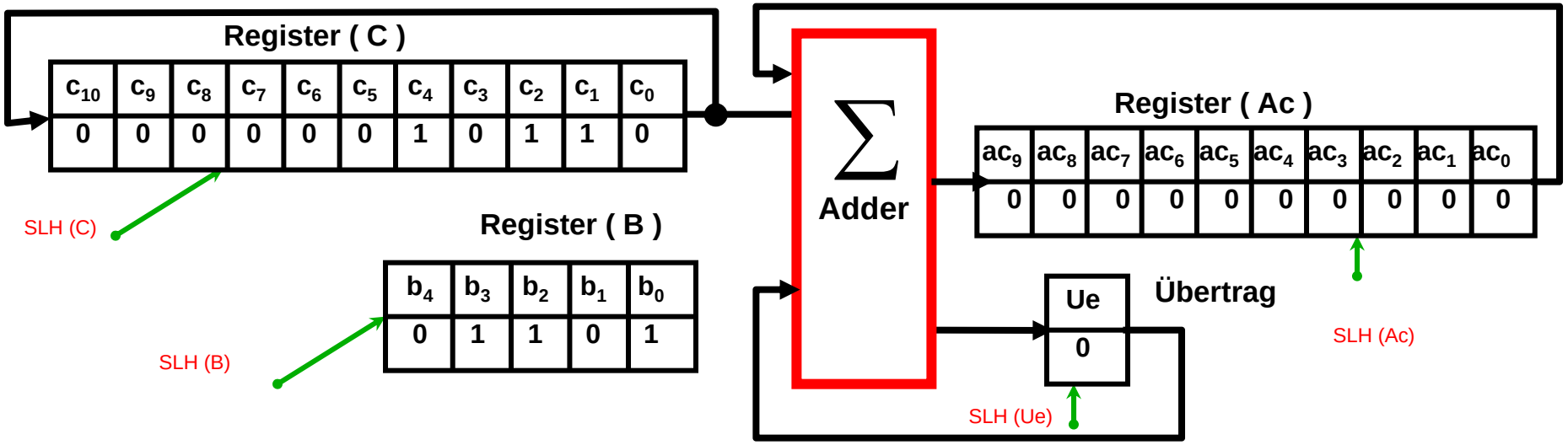
286



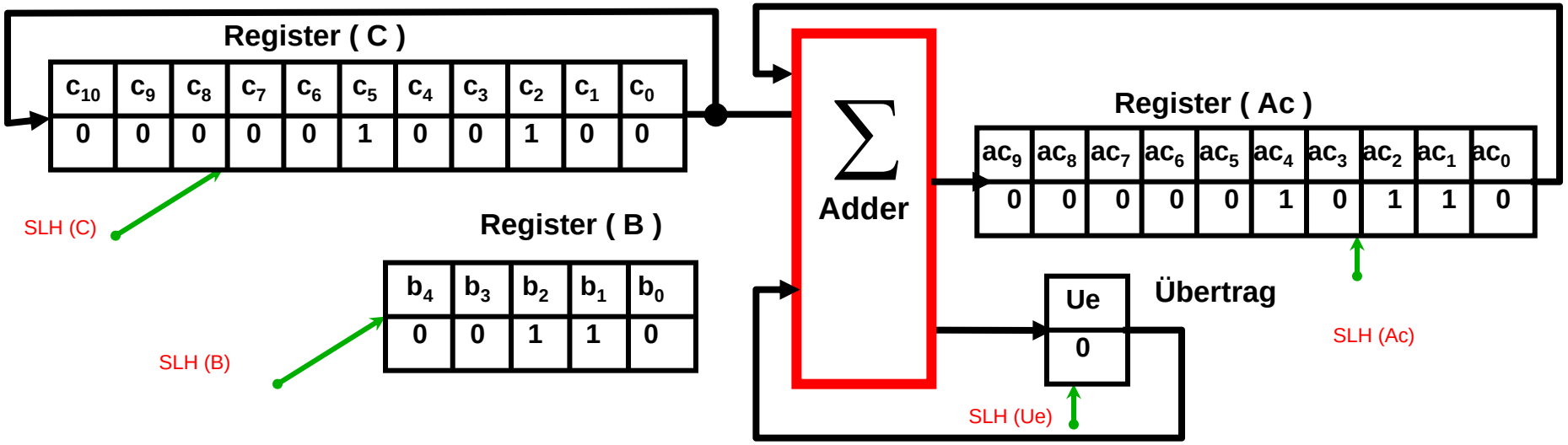
SL	H	Funktion
0	0	schieben
0	1	halten
1	0	laden
1	1	laden



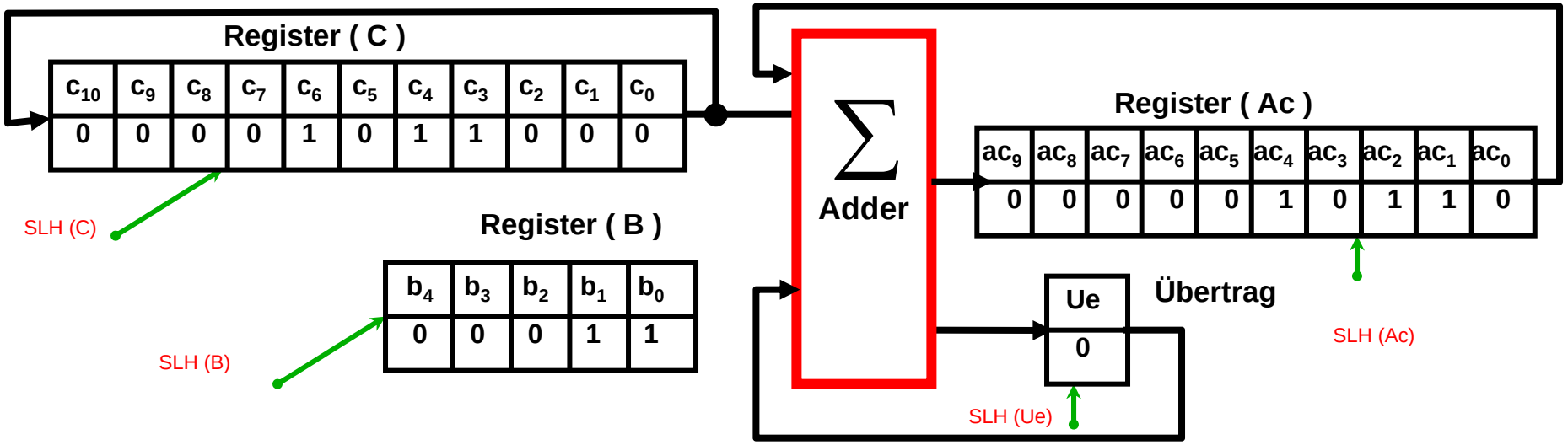
Befehl	SLH (C) (schieben)	<C>	SLH (B) (schieben)	<B>	SLH (Ac) (schieben)	<Ac>	Bemerkung
0	0	00000010110	0	01101	0	xxxxxxxxxxx	Laden der Register



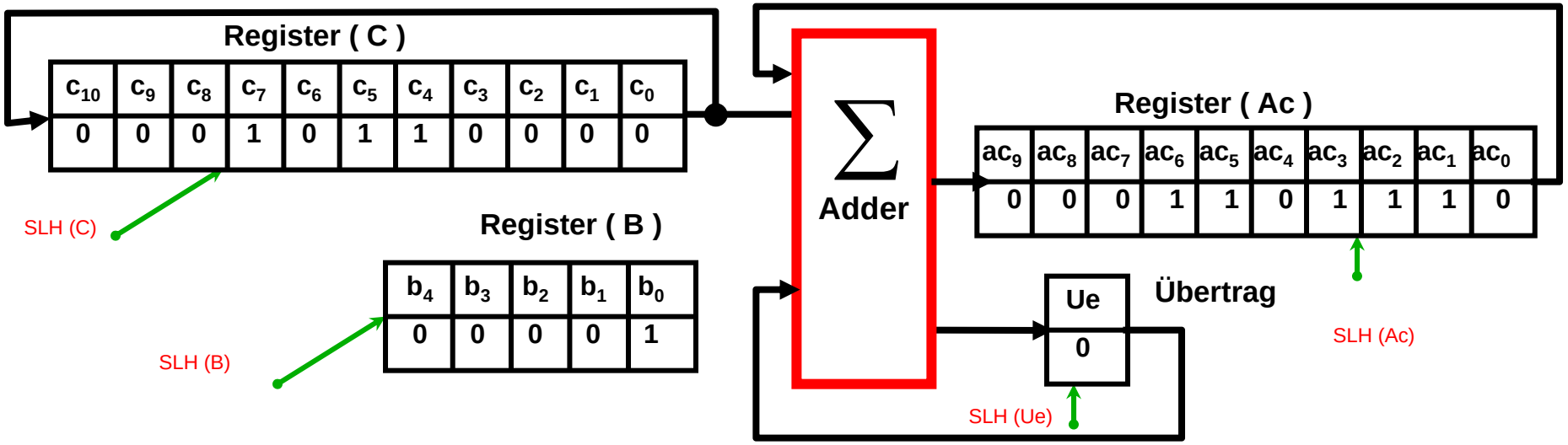
Befehl	SLH (C) (schieben)	<C>	SLH (B) (schieben)	<B>	SLH (Ac) (schieben)	<Ac>	Bemerkung
0	0	00000010110	0	01101	0	xxxxxxxxxxx	Laden der Register
1	0	00000010110	0	01101	0	0000000000	Lösche AC



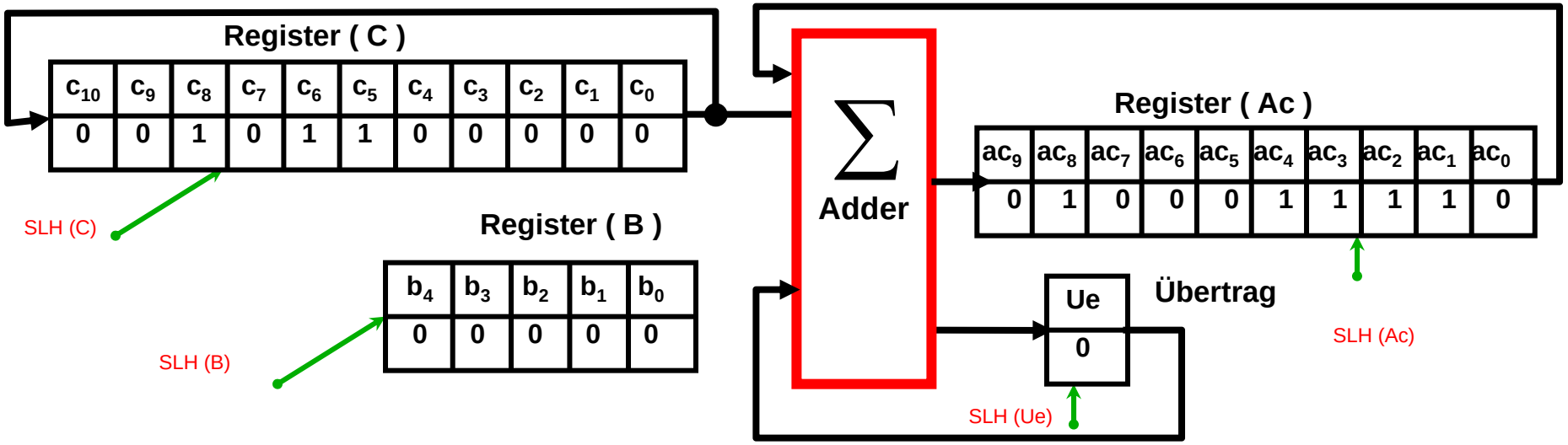
Befehl	SLH (C) (schieben)	<C>	SLH (B) (schieben)	<B>	SLH (Ac) (schieben)	<Ac>	Bemerkung
0	0	00000010110	0	01101	0	xxxxxxxxxx	Laden der Register
1	0	00000010110	0	01101	0	0000000000	Lösche AC
2	10	00000101100	1	00110	10	0000010110	1. Addition



Befehl	SLH (C) (schieben)	<C>	SLH (B) (schieben)	<B>	SLH (Ac) (schieben)	<Ac>	Bemerkung
0	0	00000010110	0	01101	0	xxxxxxxxxx	Laden der Register
1	0	00000010110	0	01101	0	0000000000	Lösche AC
2	10	00000101100	1	00110	10	0000010110	1. Addition
3	10	00001011000	1	00011	0	0000010110	

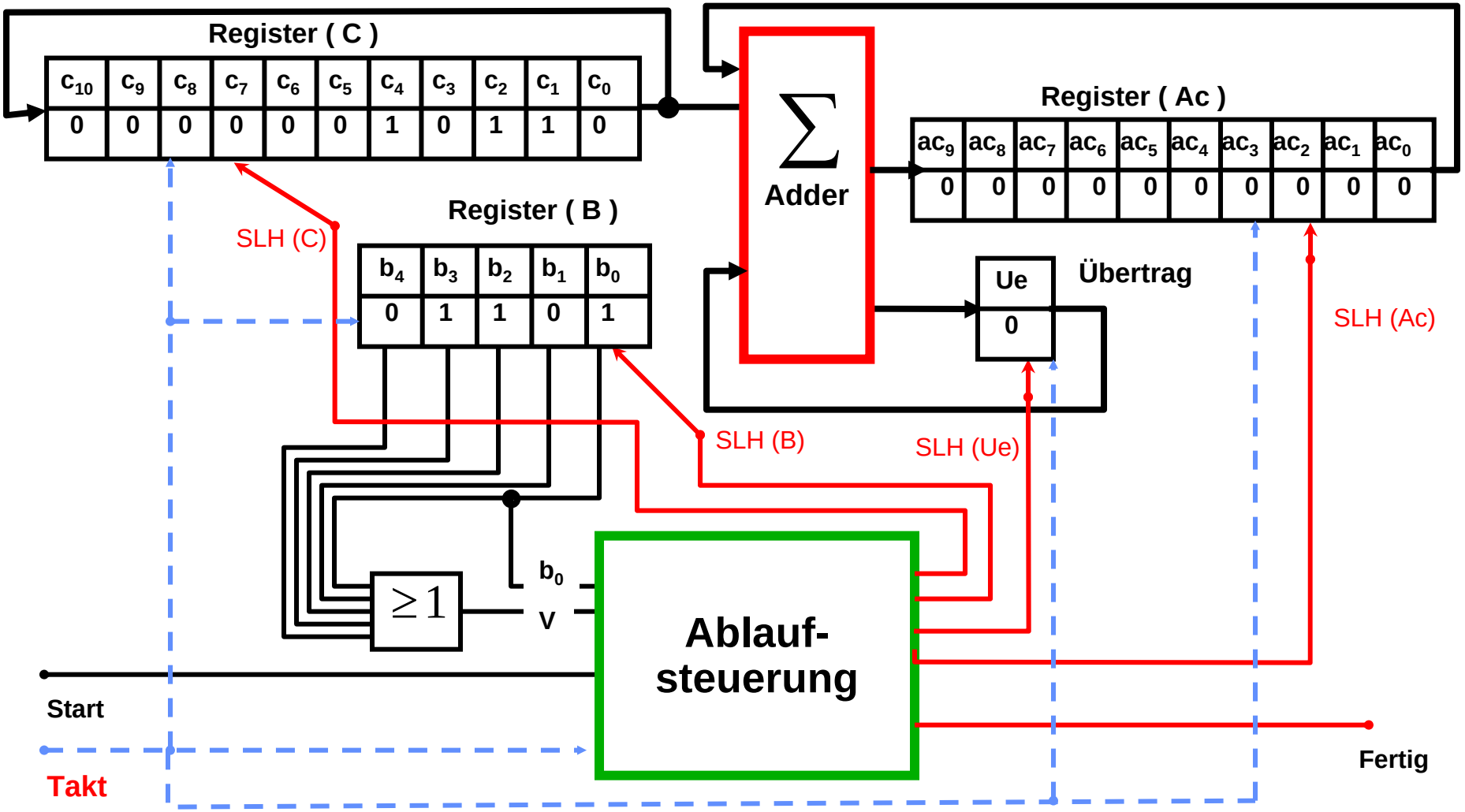


Befehl	SLH (C) (schieben)	<C>	SLH (B) (schieben)	<B>	SLH (Ac) (schieben)	<Ac>	Bemerkung
0	0	00000010110	0	01101	0	xxxxxxxxxx	Laden der Register
1	0	00000010110	0	01101	0	0000000000	Lösche AC
2	10	00000101100	1	00110	10	0000010110	1. Addition
3	10	00001011000	1	00011	0	0000010110	
4	10	00010110000	1	00001	10	0001101110	2. Addition

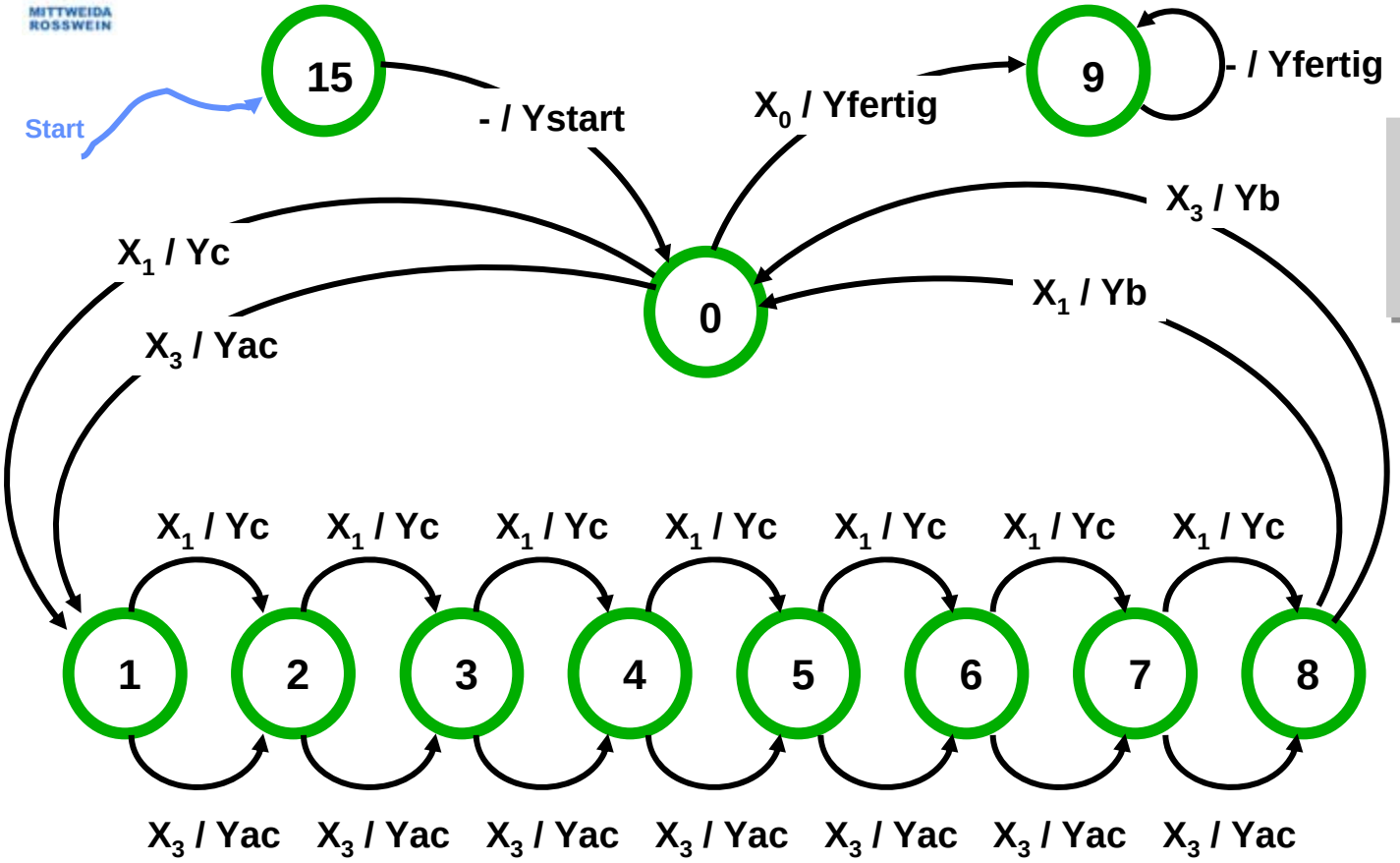


Befehl	SLH (C) (schieben)	<C>	SLH (B) (schieben)	<B>	SLH (Ac) (schieben)	<Ac>	Bemerkung
0	0	00000010110	0	01101	0	xxxxxxxxxx	Laden der Register
1	0	00000010110	0	01101	0	0000000000	Lösche AC
2	10	00000101100	1	00110	10	0000010110	1. Addition
3	10	00001011000	1	00011	0	0000010110	
4	10	00010110000	1	00001	10	0001101110	2. Addition
5	10	00101100000	1	00000	10	0100011110	3. Addition
							Fertig

Ansteuerung der Register



Zustandsgraph für 4 Bit- Multiplizierer



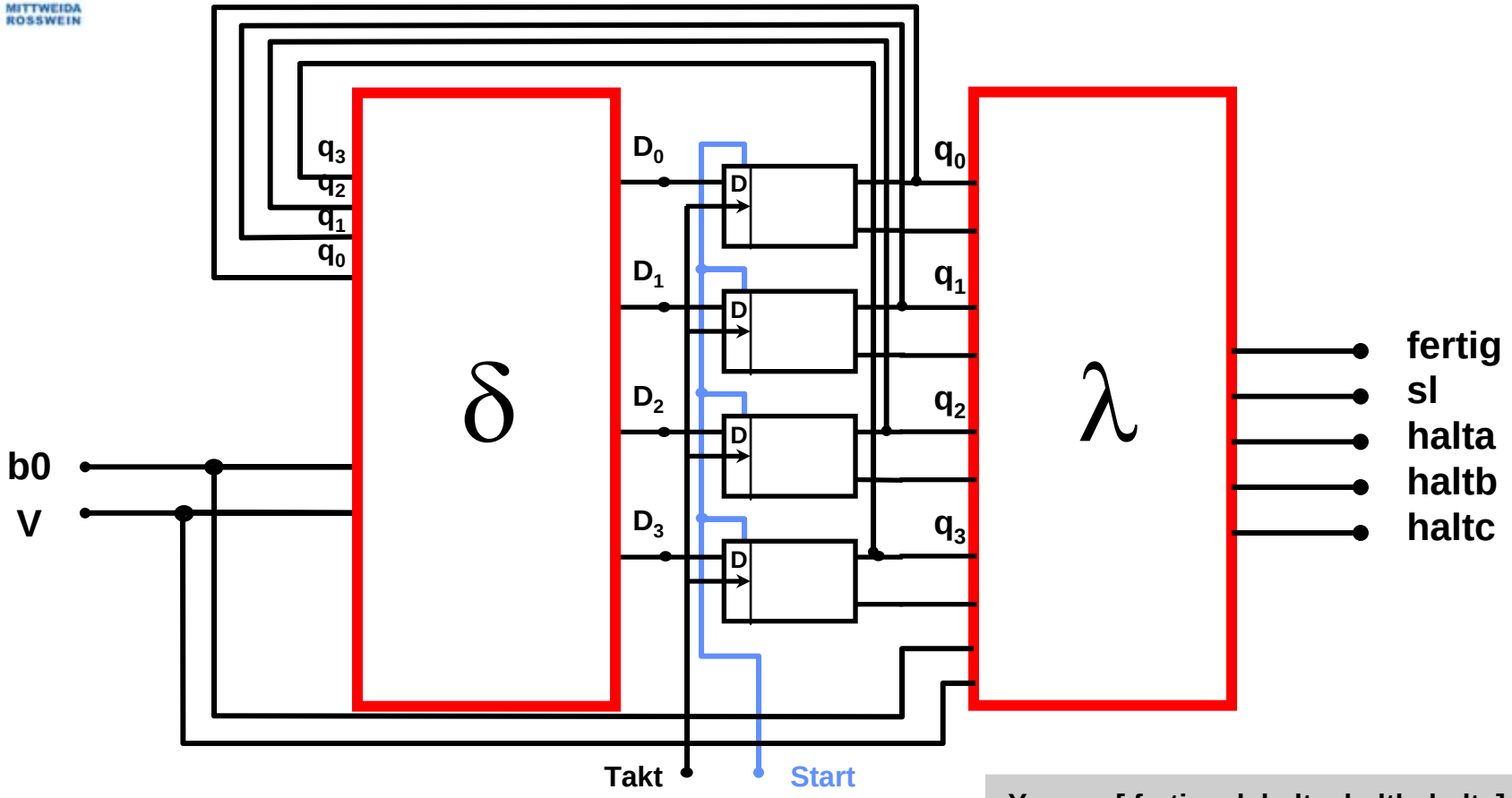
Start durch statisches Setzen aller FF's (Z_{15})

Schaltbedingung: aktive Schaltflanke am **Takt**

X	[b0, V]
X0	0 0
X1	0 1
X2	1 0
X3	1 1

Z	[q3,q2,q1,q0]
Z0	0 0 0 0
Z1	0 0 0 1
:	:
Z15	1 1 1 1

Y	[fertig, sl, halta, haltb, haltc]					
Yfertig	1	0	1	1	1	„ Fertiganzeige
Ystart	0	1	1	1	1	„ alle Reg. laden
Yb	0	0	1	0	1	„ schieben Reg b
Yc	0	0	1	1	0	„ schieben Reg c
Yac	0	0	0	1	0	„ schieben Reg a+c



X	[b0, V]
X0	0 0
X1	0 1
X2	1 0
X3	1 1

Z	[q3,q2,q1,q0]
Z0	0 0 0 0
Z1	0 0 0 1
:	:
Z15	1 1 1 1

Y	[fertig, sl, halta, haltb, haltc]				
Yfertig	1	0	1	1	1
Ystart	0	1	1	1	1
Yb	0	0	1	0	1
Yc	0	0	1	1	0
Yac	0	0	0	1	0

Arbeitsblatt Schaltbelegungstabelle

U	b0	V	q3	q2	q1	q0	T3	T2	T1	T0	F	sl	ha	hb	hc
0	0	0	0	0	0	0									
9	0	0	1	0	0	1									
15	0	0	1	1	1	1									
16	0	1	0	0	0	0									
17	0	1	0	0	0	1									
18	0	1	0	0	1	0									
19	0	1	0	0	1	1									
20	0	1	0	1	0	0									
21	0	1	0	1	0	1									
22	0	1	0	1	1	0									
23	0	1	0	1	1	1	δ					λ			
24	0	1	1	0	0	0									
25	0	1	1	0	0	1									
31	0	1	1	1	1	1									
48	1	1	0	0	0	0									
49	1	1	0	0	0	1									
50	1	1	0	0	1	0									
51	1	1	0	0	1	1									
52	1	1	0	1	0	0									
53	1	1	0	1	0	1									
54	1	1	0	1	1	0									
55	1	1	0	1	1	1									
56	1	1	1	0	0	0									
57	1	1	1	0	0	1									
63	1	1	1	1	1	1									
	sonst														

T	q^{t+1}
0	q^{t}
1	$!\, q^{t}$

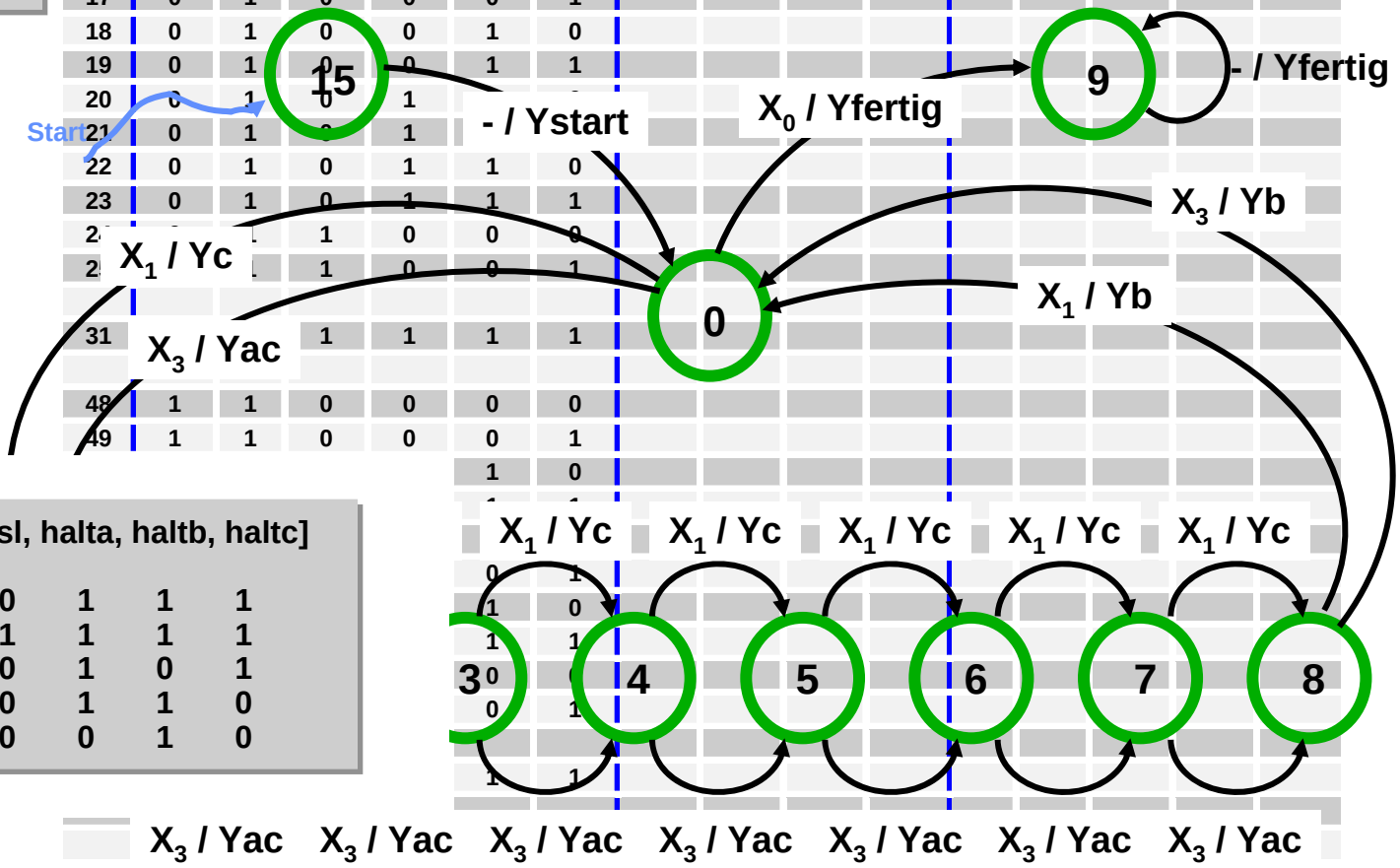
Arbeitsblatt Schaltbelegungstabelle

X	[b0, V]	
X0	0	0
X1	0	1
X2	1	0
X3	1	1

T	q^{t+1}
0	q^t
1	$!\,q^t$

Y	[fertig, sl, halta, haltb, haltc]				
Yfertig	1	0	1	1	1
Ystart	0	1	1	1	1
Yb	0	0	1	0	1
Yc	0	0	1	1	0
Yac	0	0	0	1	0

U	b0	V	q3	q2	q1	q0	T3	T2	T1	T0	F	sl	ha	hb	hc
0	0	0	0	0	0	0									
9	0	0	1	0	0	1									
15	0	0	1	1	1	1									
16	0	1	0	0	0	0									
17	0	1	0	0	0	1									
18	0	1	0	0	1	0									
19	0	1	0	0	1	1									
20	0	1	0	1	0	0									
21	0	1	0	1	0	1									
22	0	1	0	1	1	0									
23	0	1	0	1	1	1									
24	1	1	0	0	0	0									
25	1	1	0	0	0	1									
31	1	1	1	1	1	1									
48	1	1	0	0	0	0									
49	1	1	0	0	0	1									



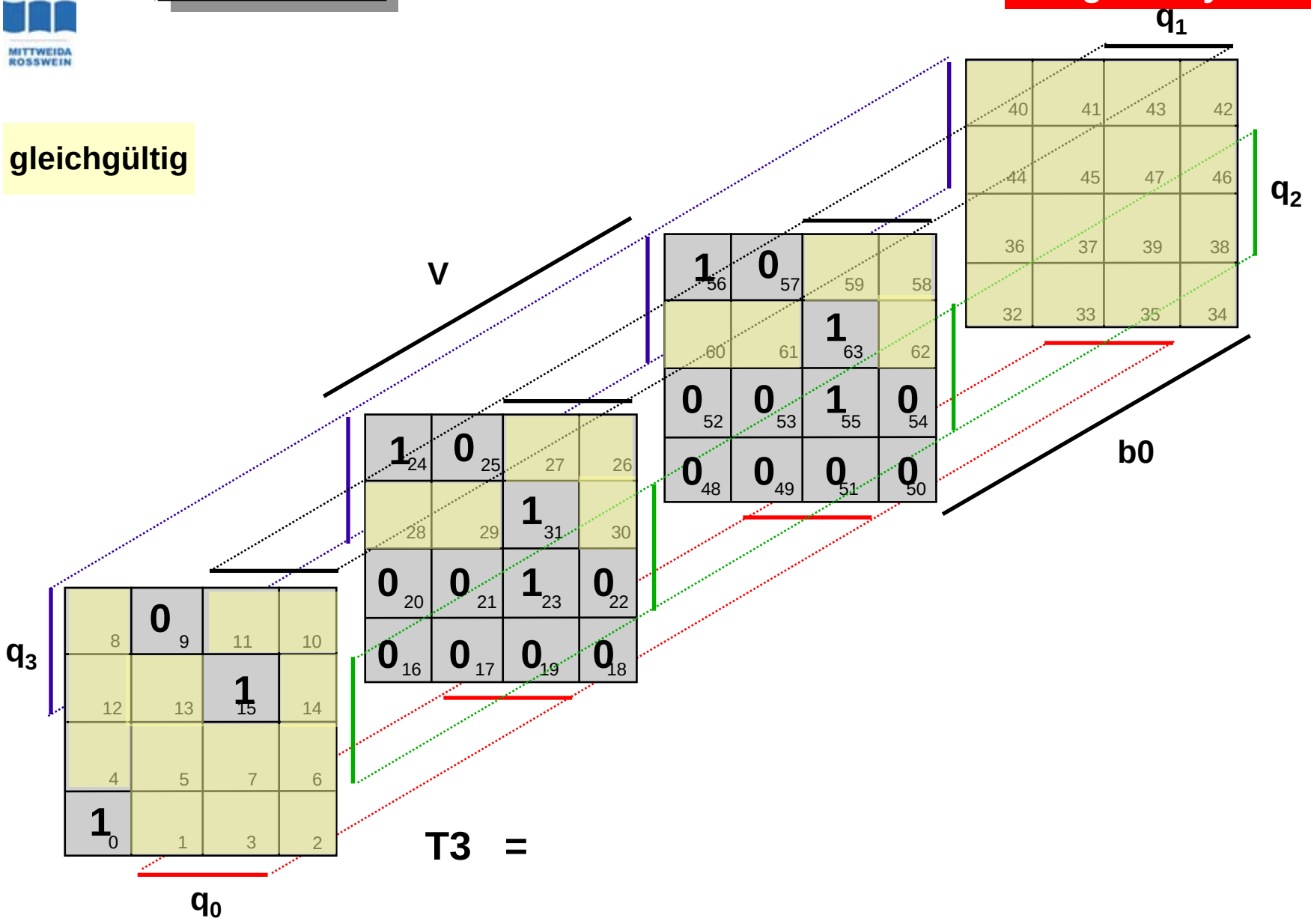
Schaltbelegungstabelle für 4 Bit- Multiplizierer

U	b0	V	q3	q2	q1	q0	T3	T2	T1	T0	F	sl	ha	hb	hc
0	0	0	0	0	0	0	1	0	0	1	1	0	1	1	1
9	0	0	1	0	0	1	0	0	0	0	1	0	1	1	1
15	0	0	1	1	1	1	1	1	1	1	0	1	1	1	1
16	0	1	0	0	0	0	0	0	0	1	0	0	1	1	0
17	0	1	0	0	0	1	0	0	1	1	0	0	1	1	0
18	0	1	0	0	1	0	0	0	0	1	0	0	1	1	0
19	0	1	0	0	1	1	0	1	1	1	0	0	1	1	0
20	0	1	0	1	0	0	0	0	0	1	0	0	1	1	0
21	0	1	0	1	0	1	0	0	1	1	0	0	1	1	0
22	0	1	0	1	1	0	0	0	0	1	0	0	1	1	0
23	0	1	0	1	1	1	1	1	1	1	0	0	1	1	0
24	0	1	1	0	0	0	1	0	0	0	0	0	1	0	1
25	0	1	1	0	0	1	0	0	0	0	0	1	1	1	1
31	0	1	1	1	1	1	1	1	1	1	0	1	1	1	1
48	1	1	0	0	0	0	0	0	0	1	0	0	0	1	0
49	1	1	0	0	0	1	0	0	1	1	0	0	0	1	0
50	1	1	0	0	1	0	0	0	0	1	0	0	0	1	0
51	1	1	0	0	1	1	0	1	1	1	0	0	0	1	0
52	1	1	0	1	0	0	0	0	0	1	0	0	0	1	0
53	1	1	0	1	0	1	0	0	1	1	0	0	0	1	0
54	1	1	0	1	1	0	0	0	0	1	0	0	0	1	0
55	1	1	0	1	1	1	1	1	1	1	0	0	0	1	0
56	1	1	1	0	0	0	1	0	0	0	0	0	1	0	1
57	1	1	1	0	0	1	0	0	0	0	1	0	1	1	1
63	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1
				sonst			-	-	-	-	-	-	-	-	-

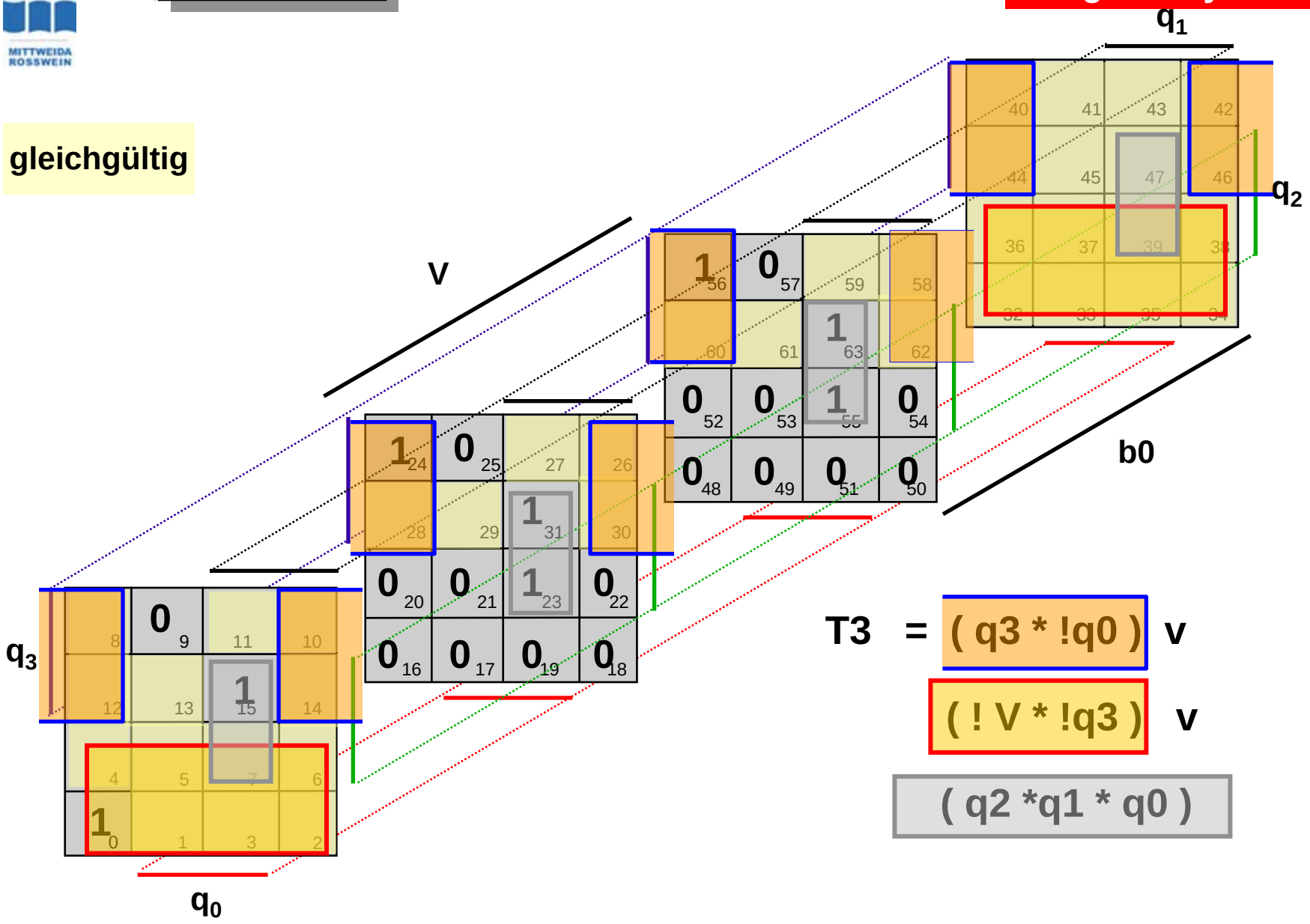
δ λ

Karnaughplan T3

gleichgültig



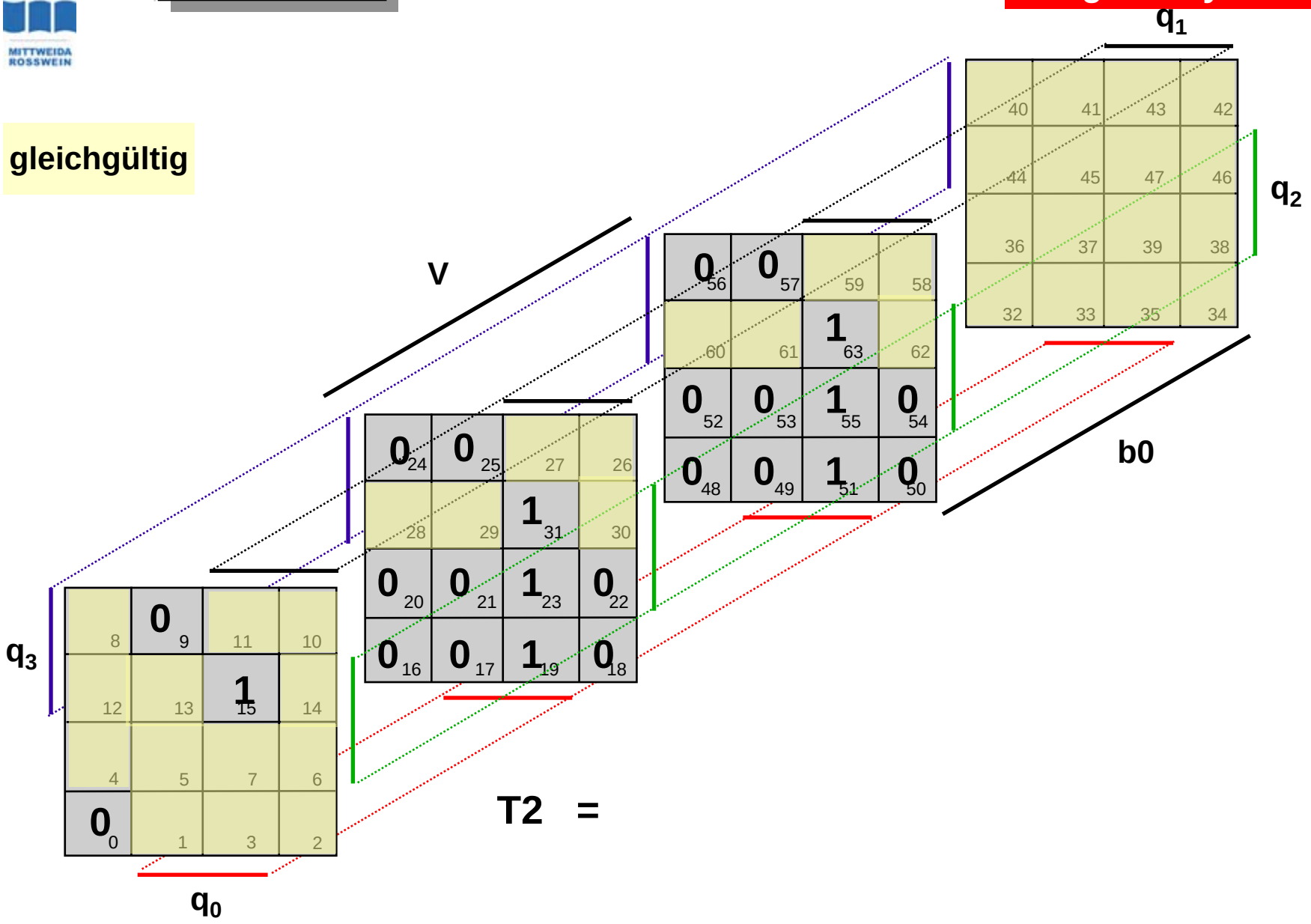
gleichgültig



$$T3 = (q3 * !q0)\ v\ (!V * !q3)\ v\ (q2 * q1 * q0)$$

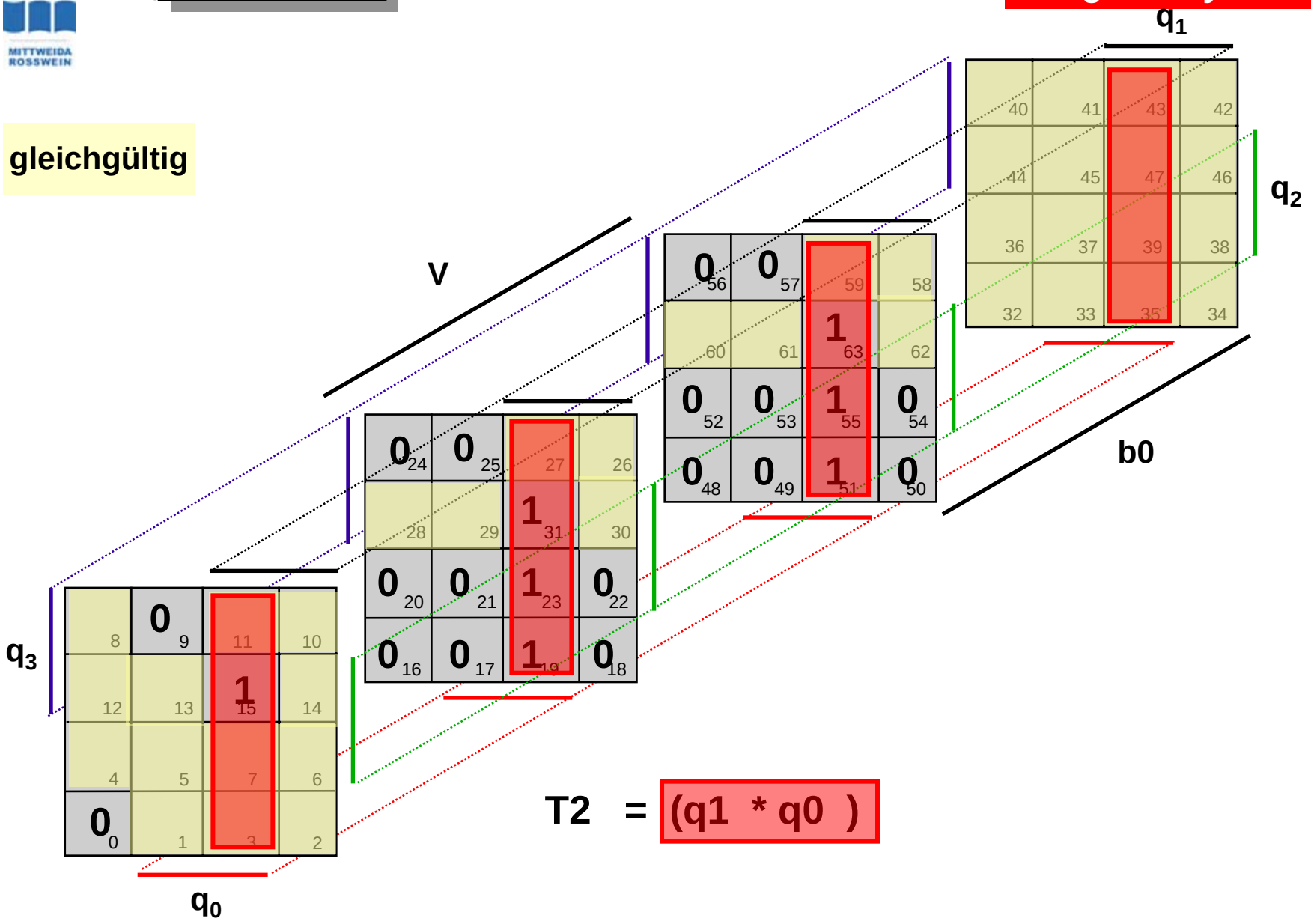
Karnaughplan T2

gleichgültig



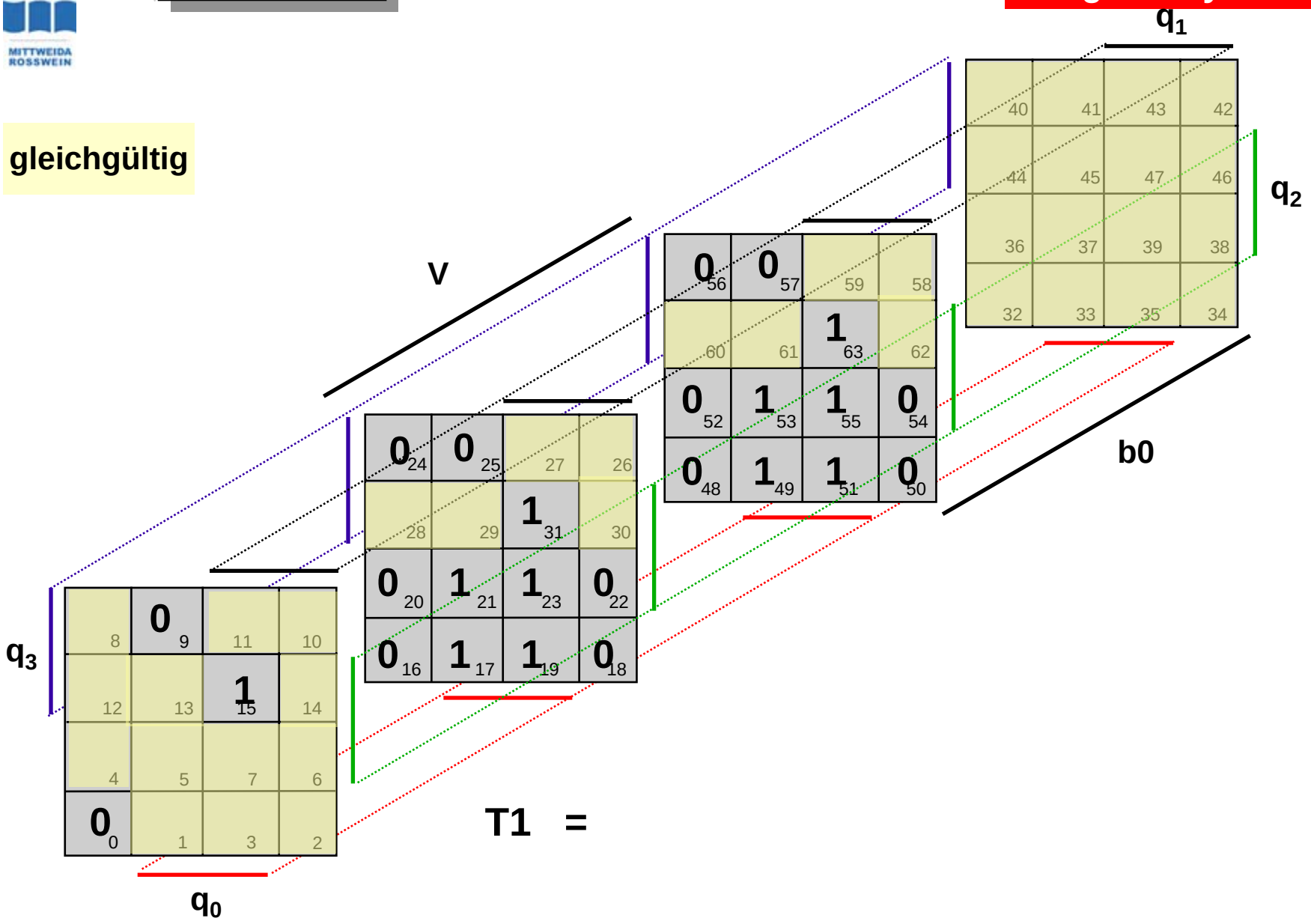
Karnaughplan T2

gleichgültig



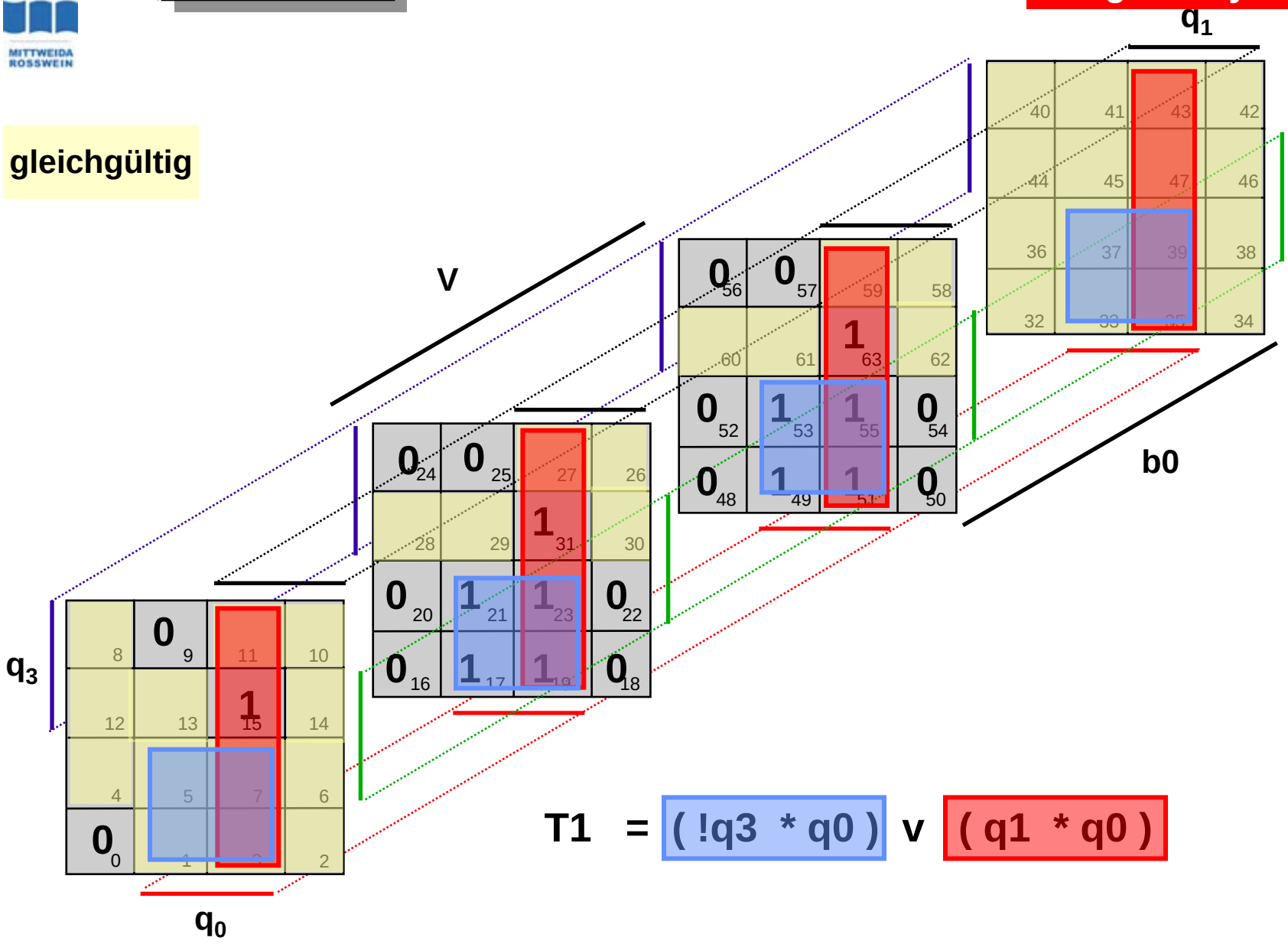
$$T2 = (q1 * q0)$$

gleichgültig



Karnaughplan T1

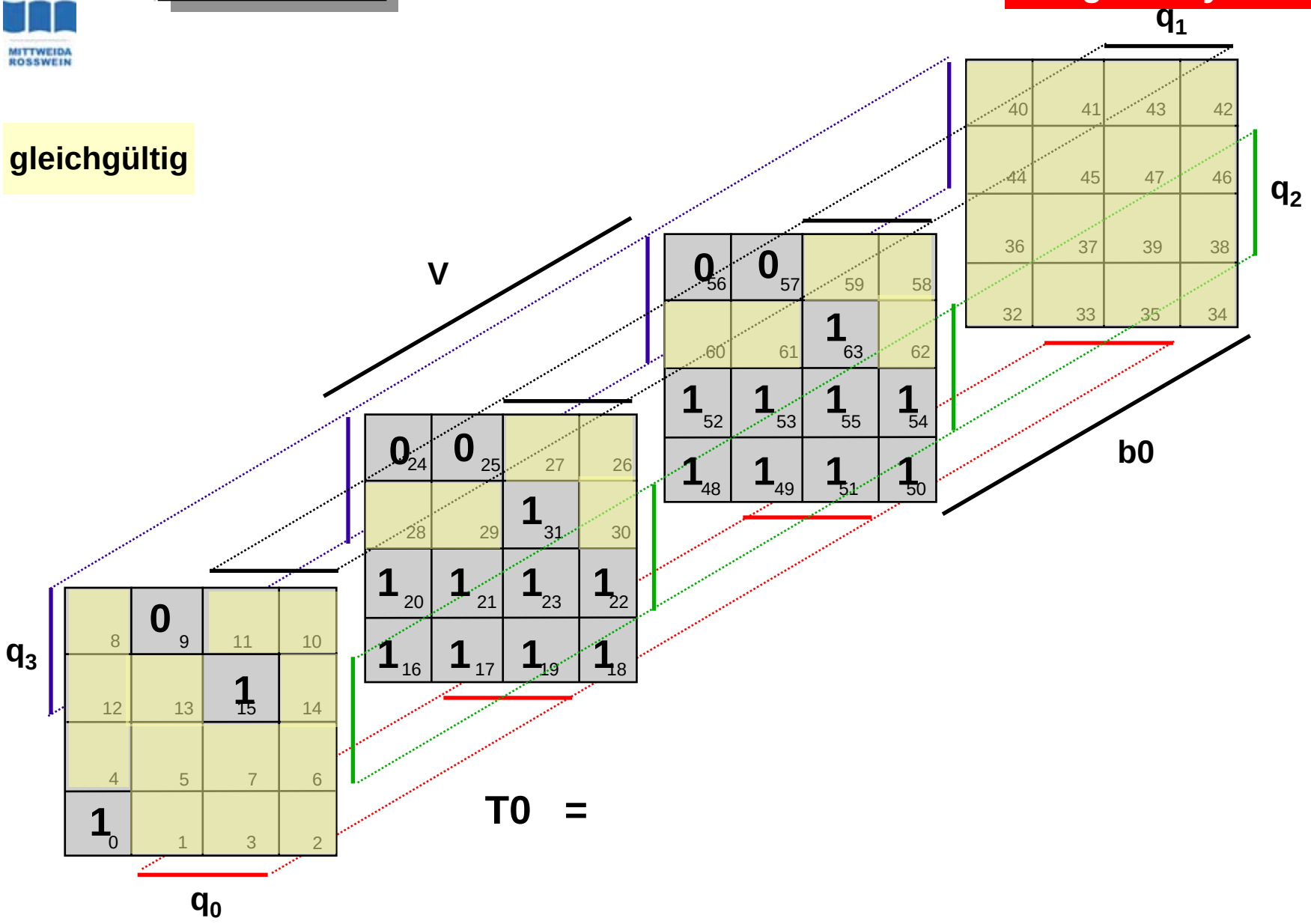
gleichgültig



$$T1 = (!q3 * q0) \; v \; (q1 * q0)$$

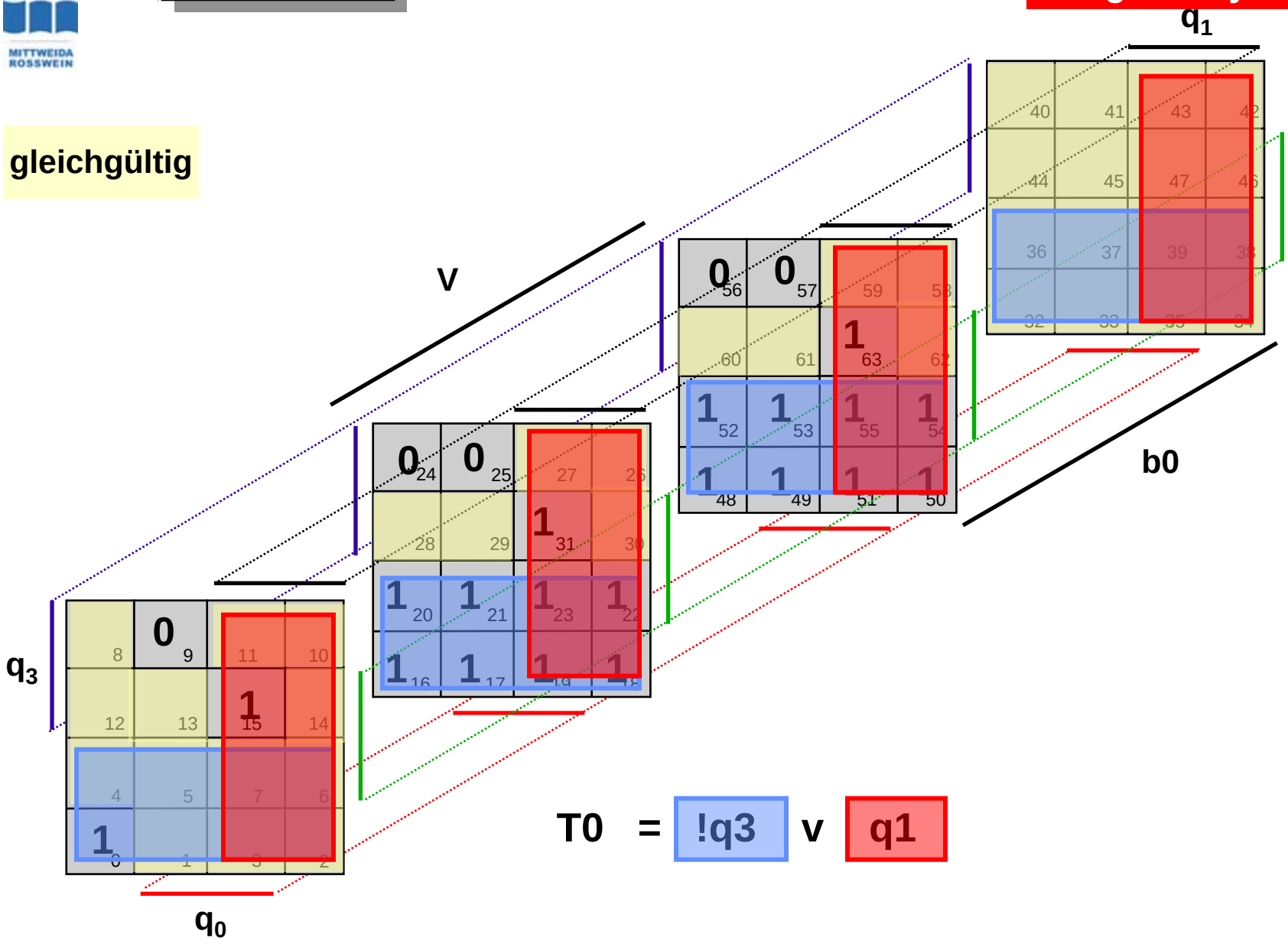
Karnaughplan T0

gleichgültig



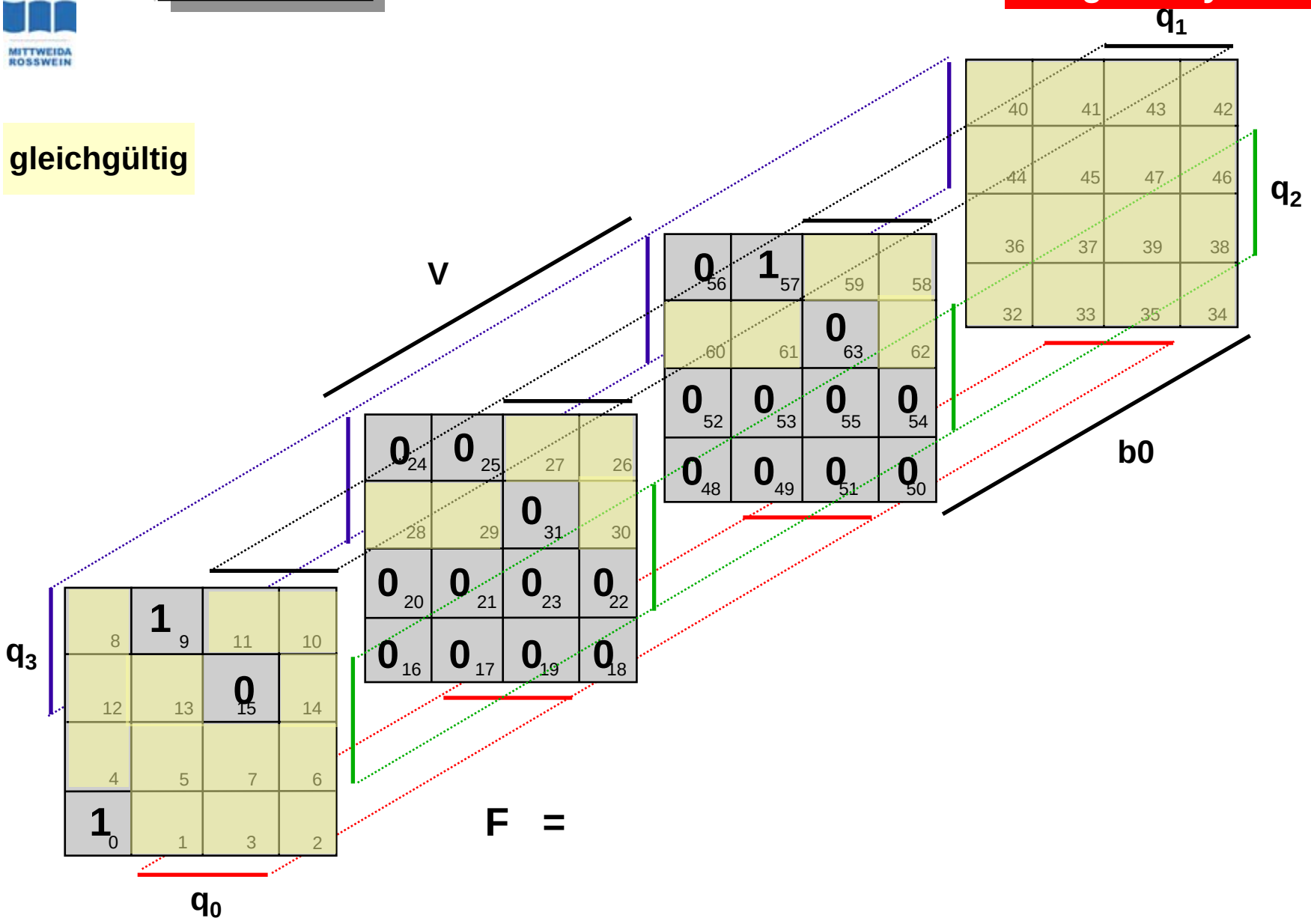
Karnaughplan T0

gleichgültig

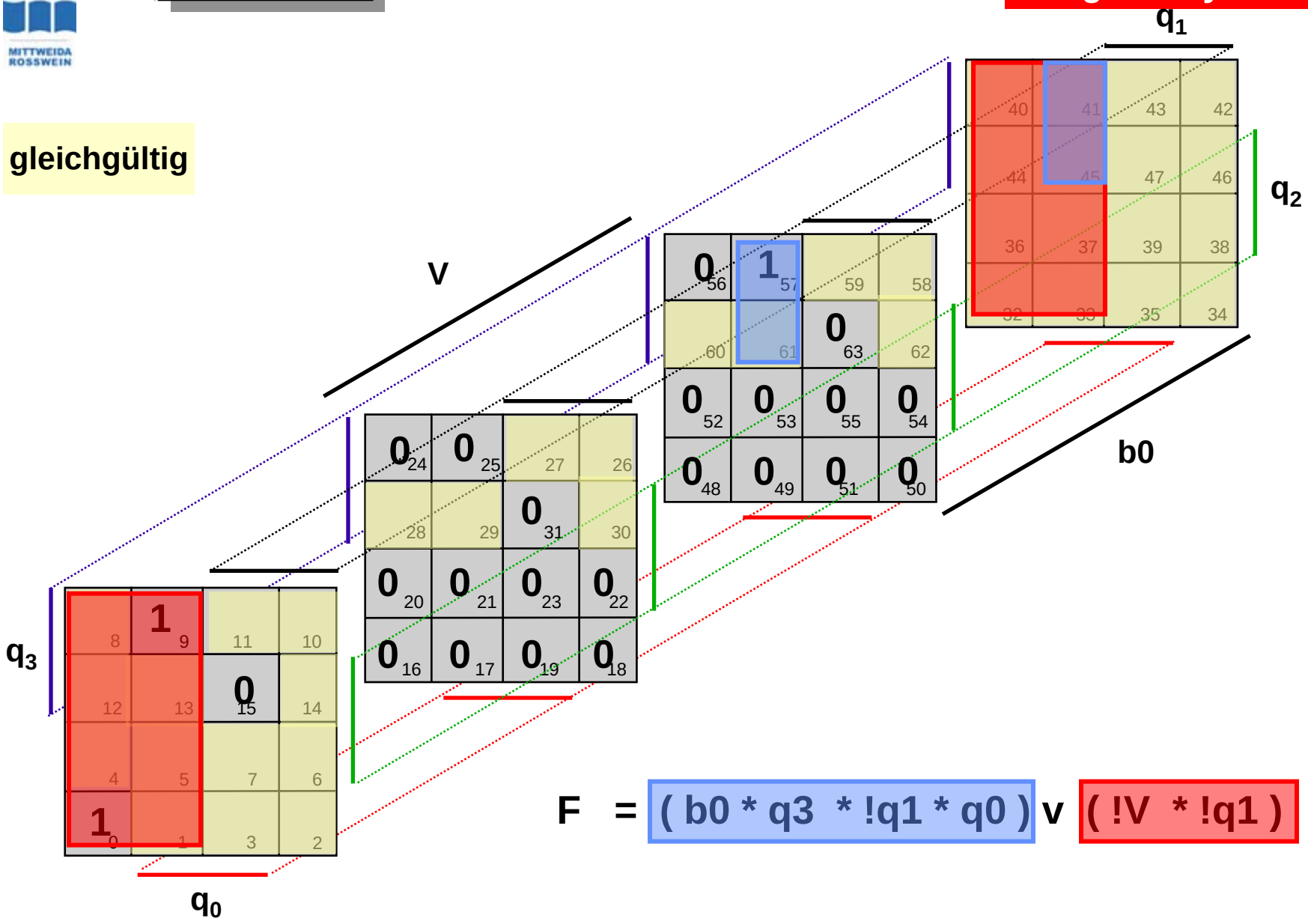


$$T0 = !q3 \; v \; q1$$

gleichgültig

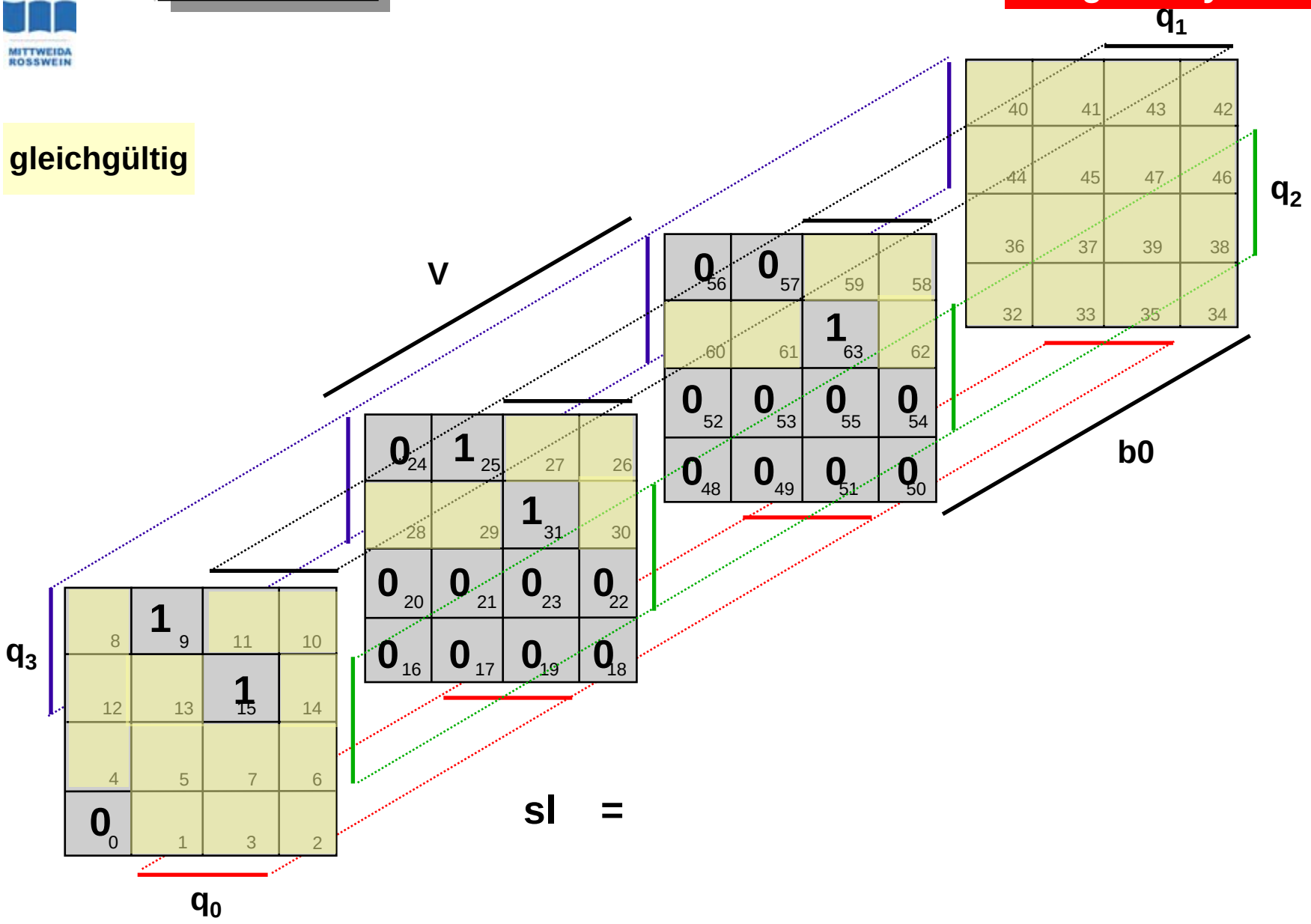


gleichgültig

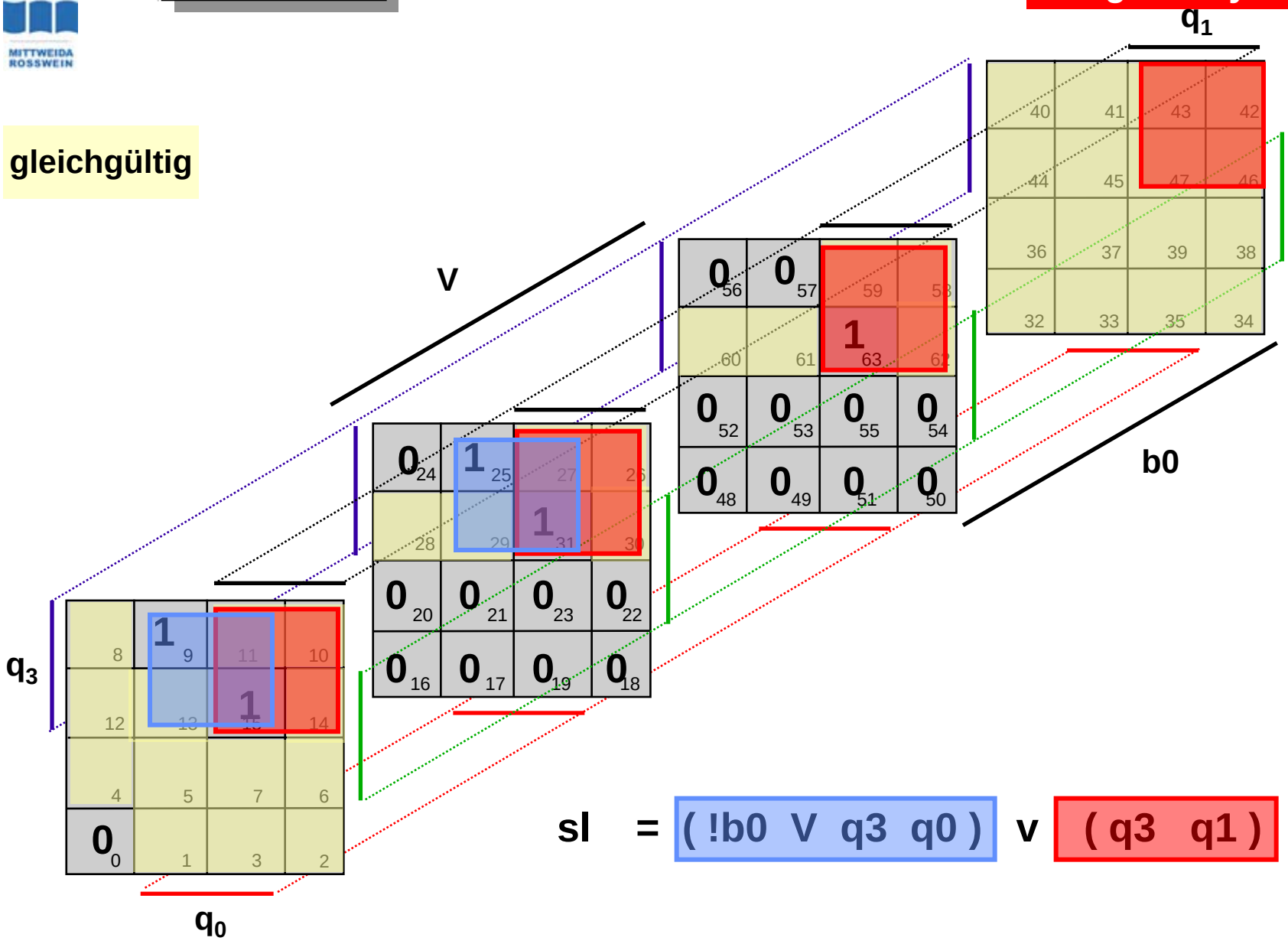


Karnaughplan sl

gleichgültig



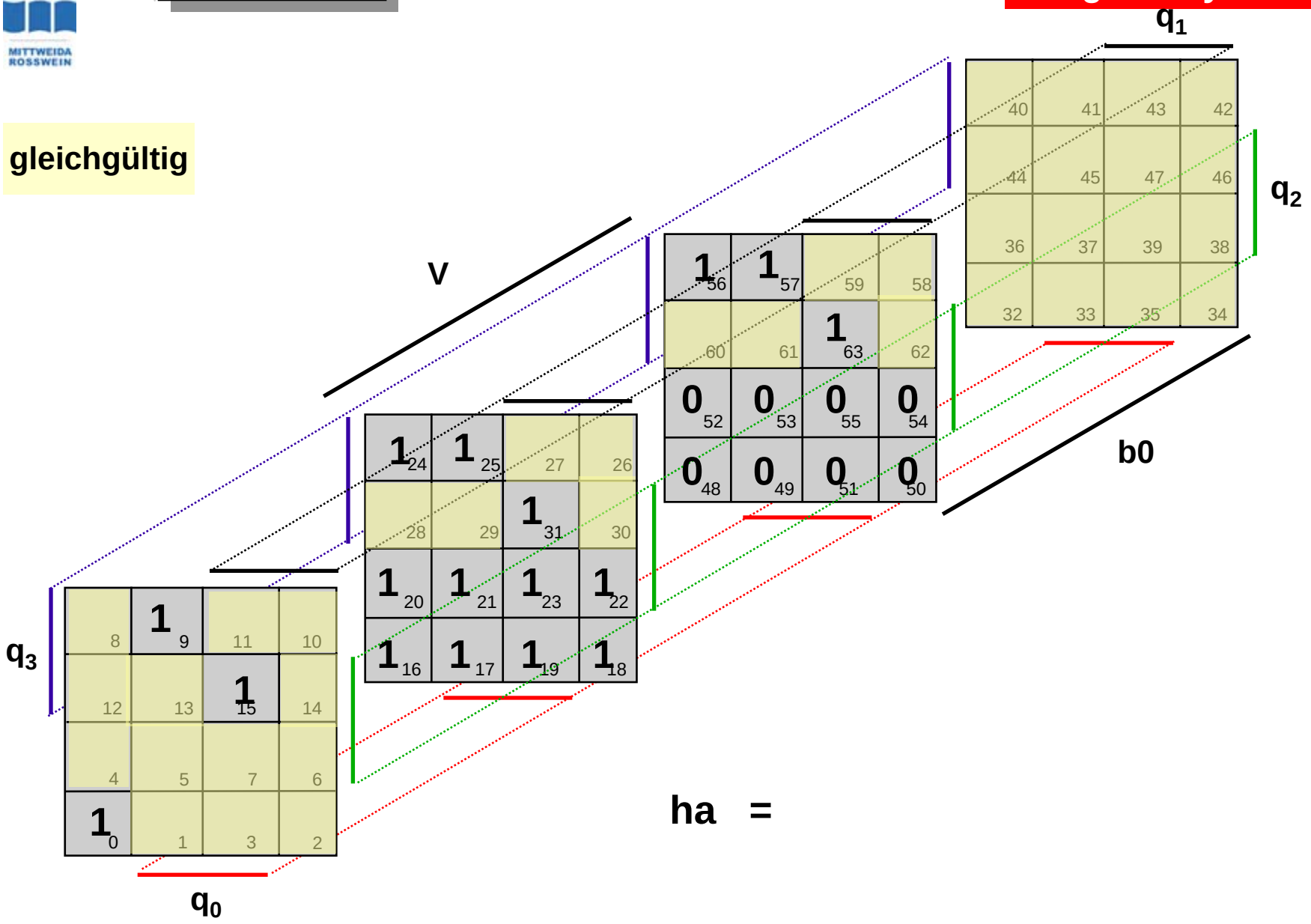
gleichgültig



$$sl = (\,!b0 \;\; V \;\; q3 \;\; q0\,) \;\; v \;\; (\,q3 \;\; q1\,)$$

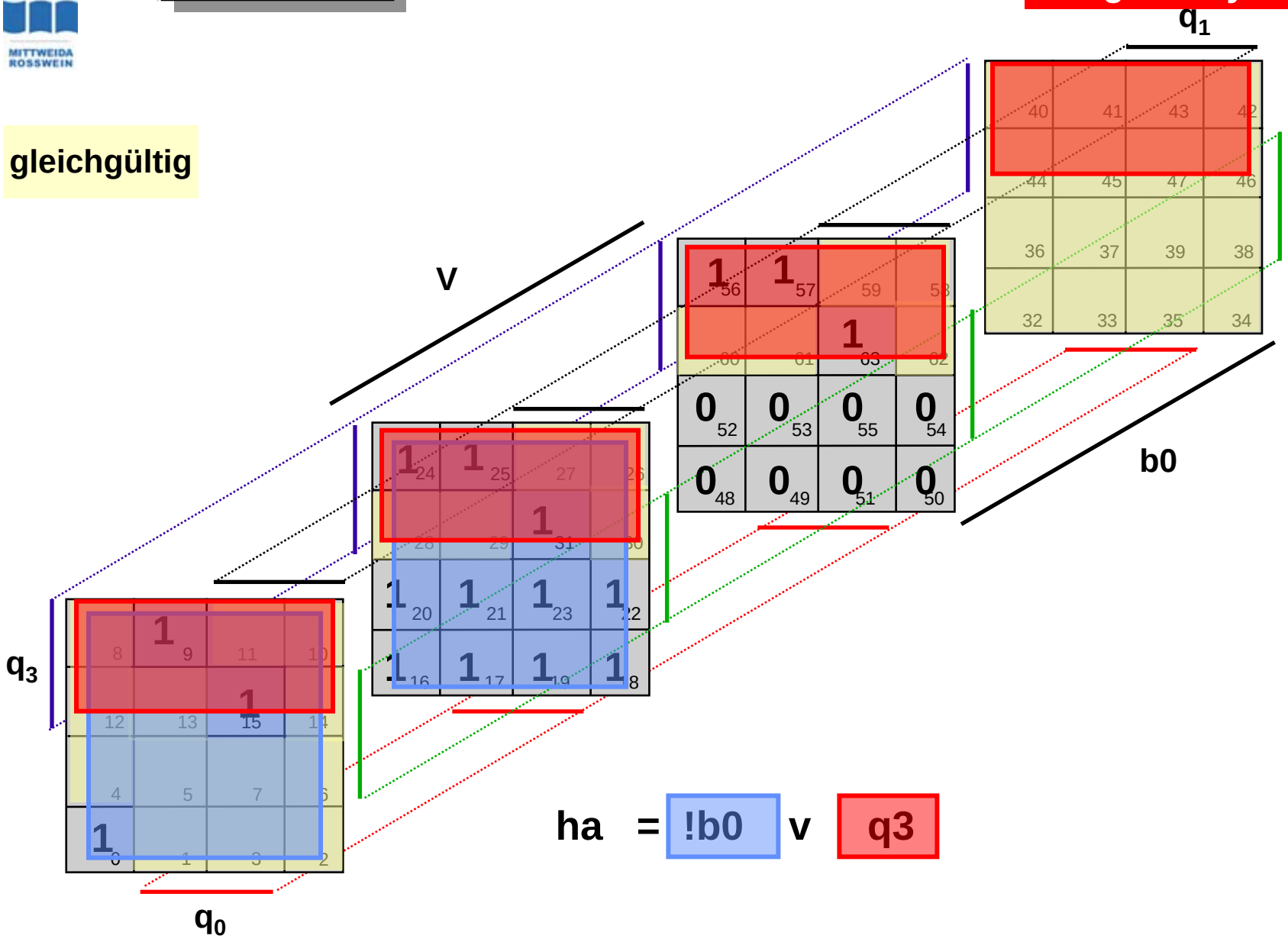
Karnaughplan ha

gleichgültig



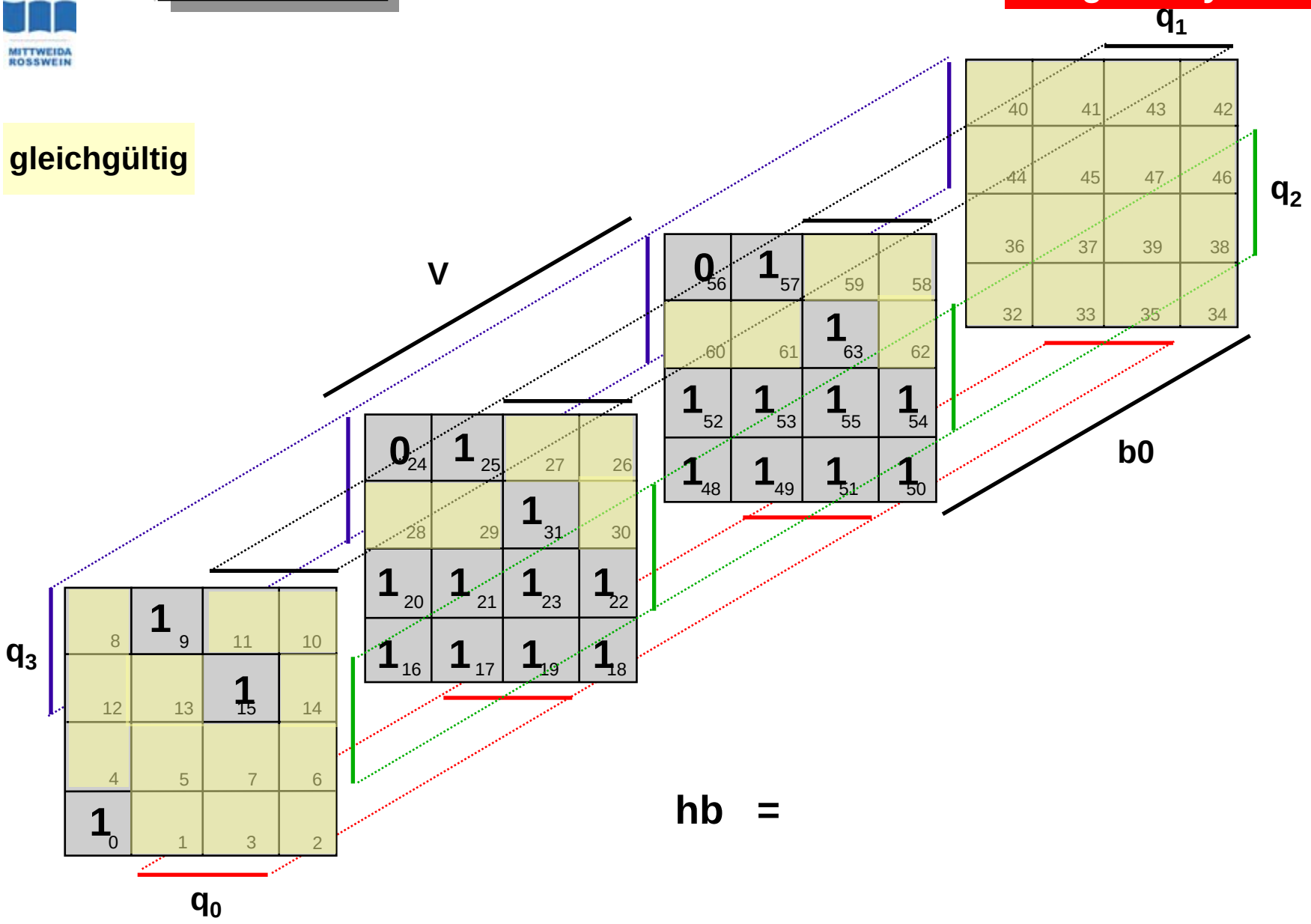
Karnaughplan ha

gleichgültig



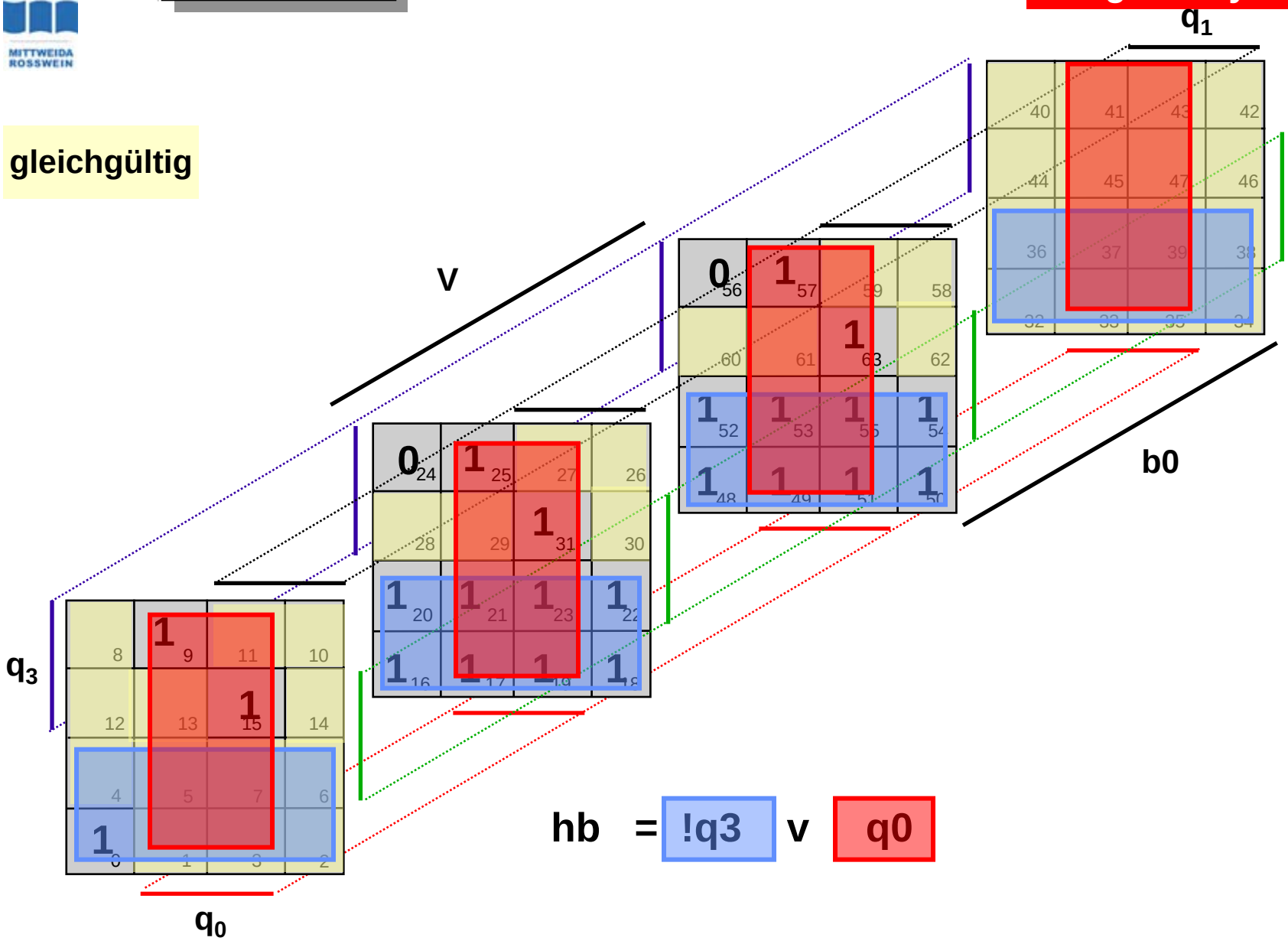
$$ha = !b0 \; v \; q3$$

gleichgültig

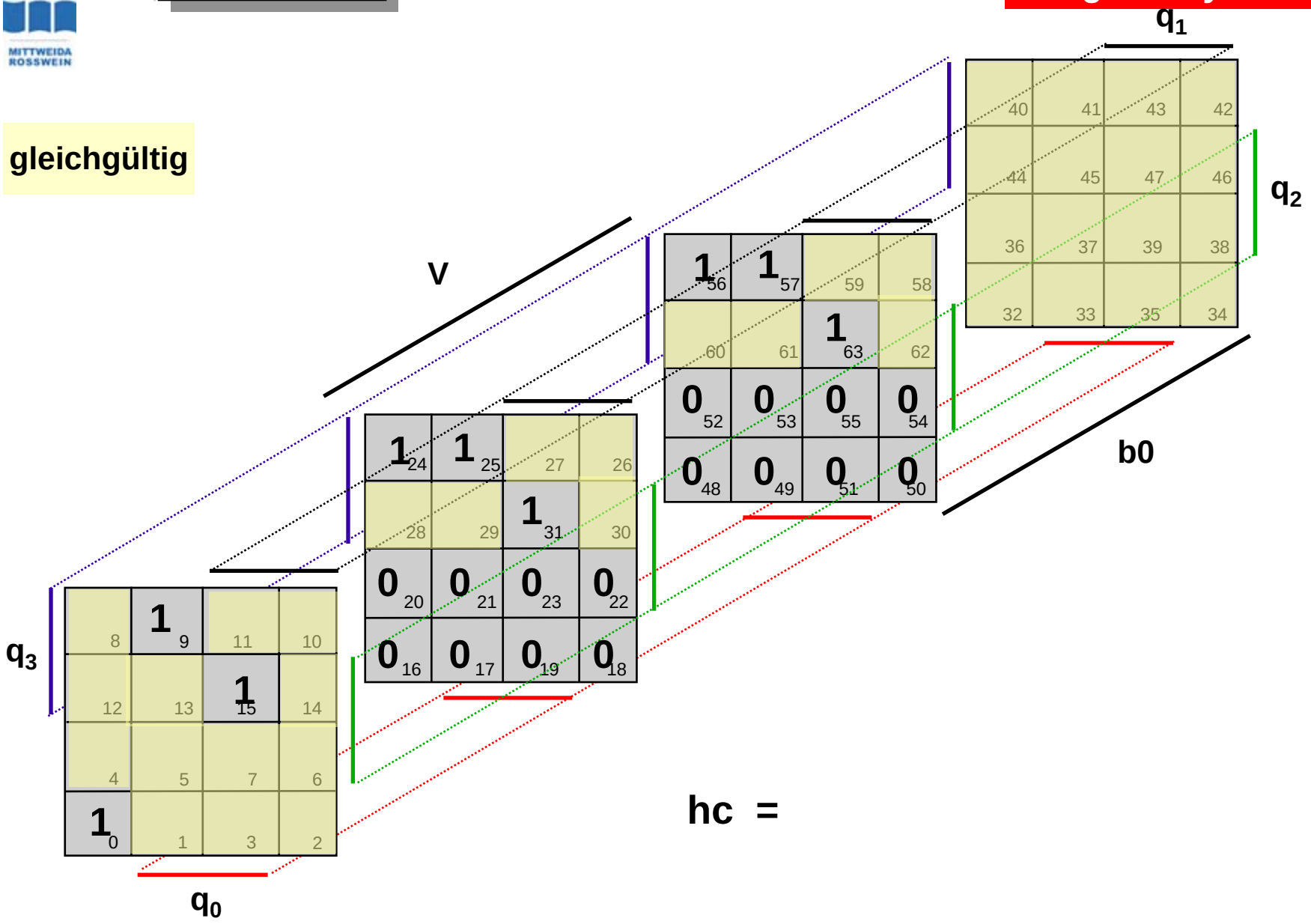


Karnaughplan hb

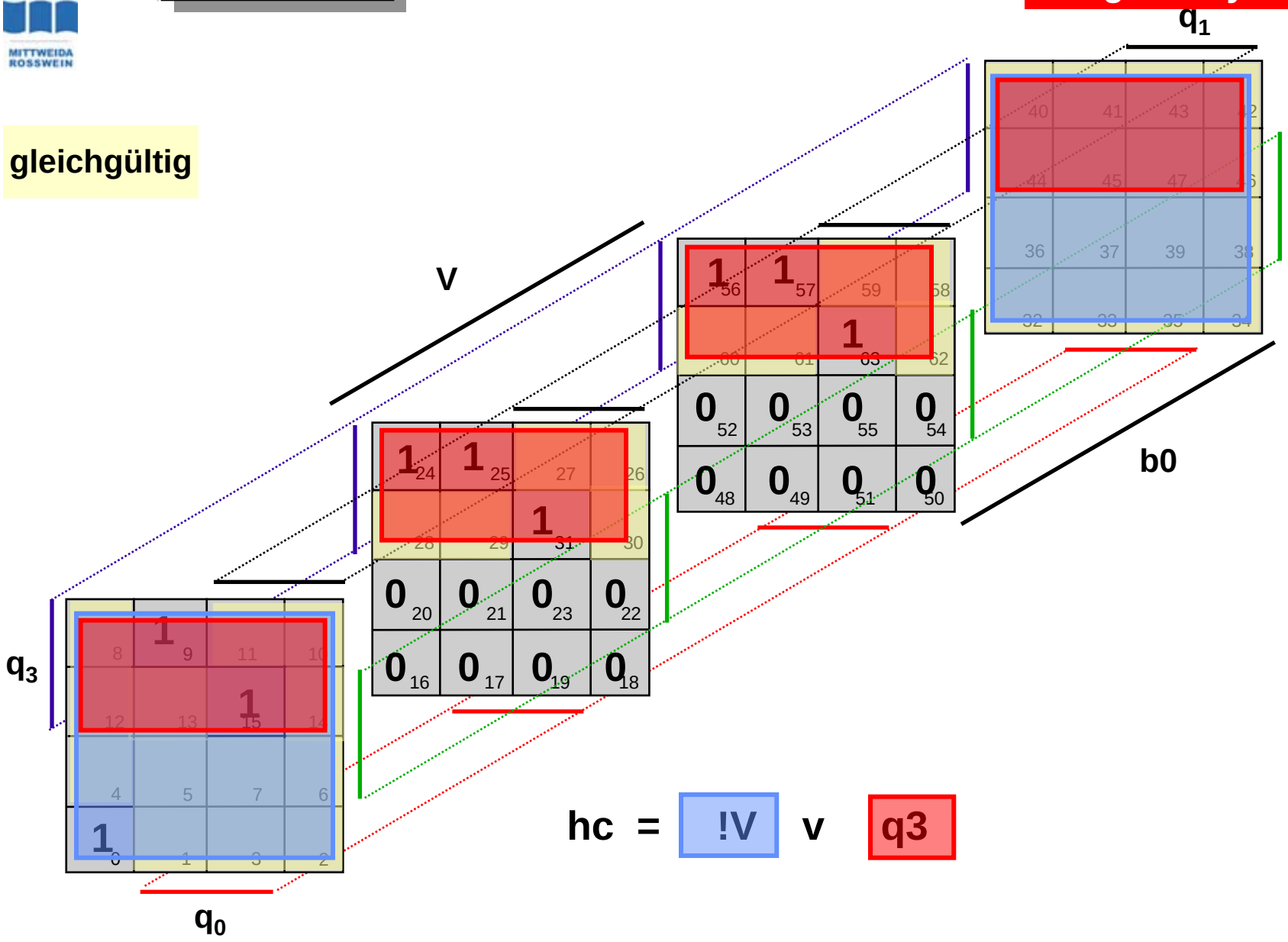
gleichgültig



gleichgültig



gleichgültig



Zusammenfassung der Schaltfunktionen

$$T_3 = q_3\bar{q}_0 \vee \bar{V}\,\bar{q}_3 \vee q_2q_1q_0$$

$$T_2 = q_1q_0$$

$$T_1 = \bar{q}_3q_0 \vee q_1q_0$$

$$T_0 = \bar{q}_3 \vee q_1$$

$$F = b_0q_3\bar{q}_1q_0 \vee \bar{V}\,\bar{q}_1$$

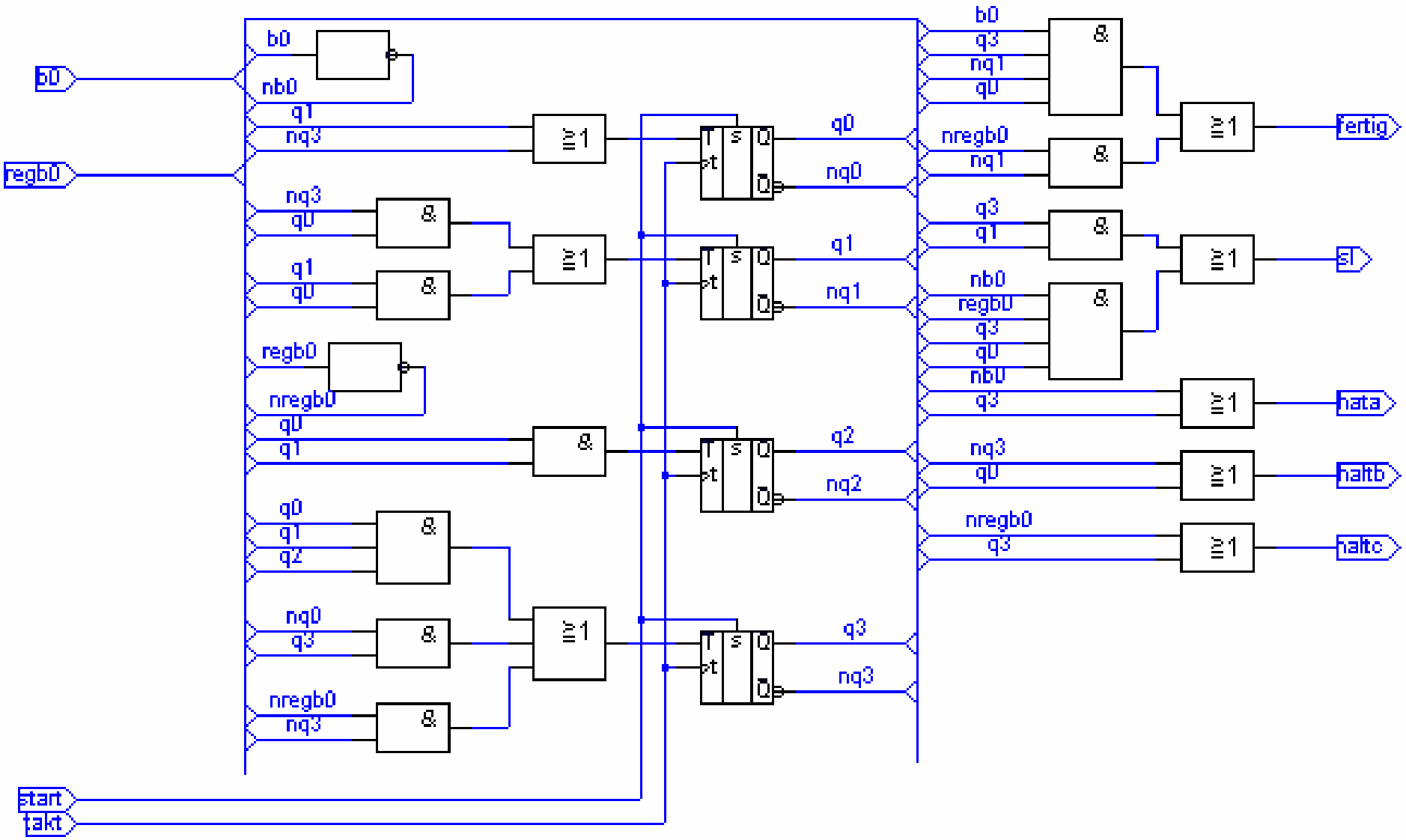
$$sl = \bar{b}_0q_3q_0 \vee q_3q_1$$

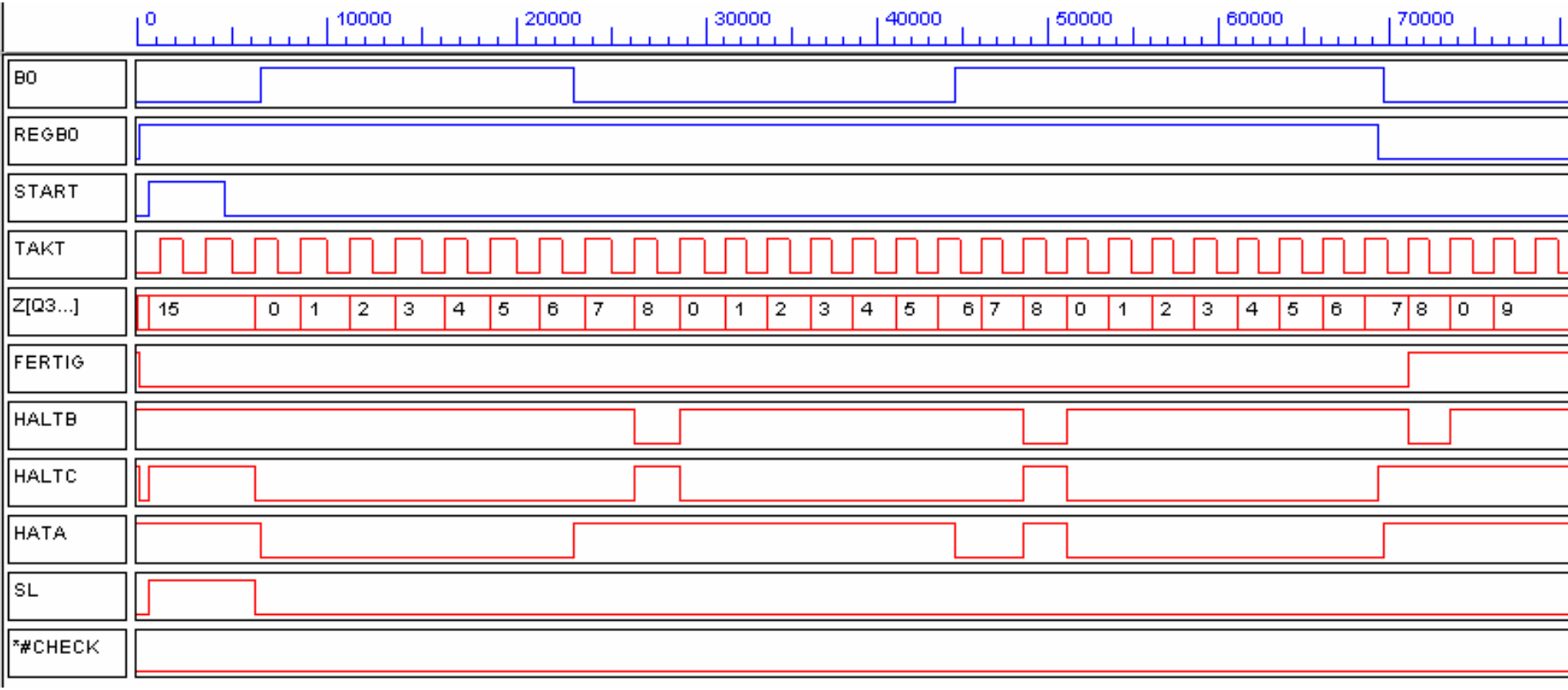
$$ha = \bar{b}_0 \vee q_3$$

$$hb = \bar{q}_3 \vee q_0$$

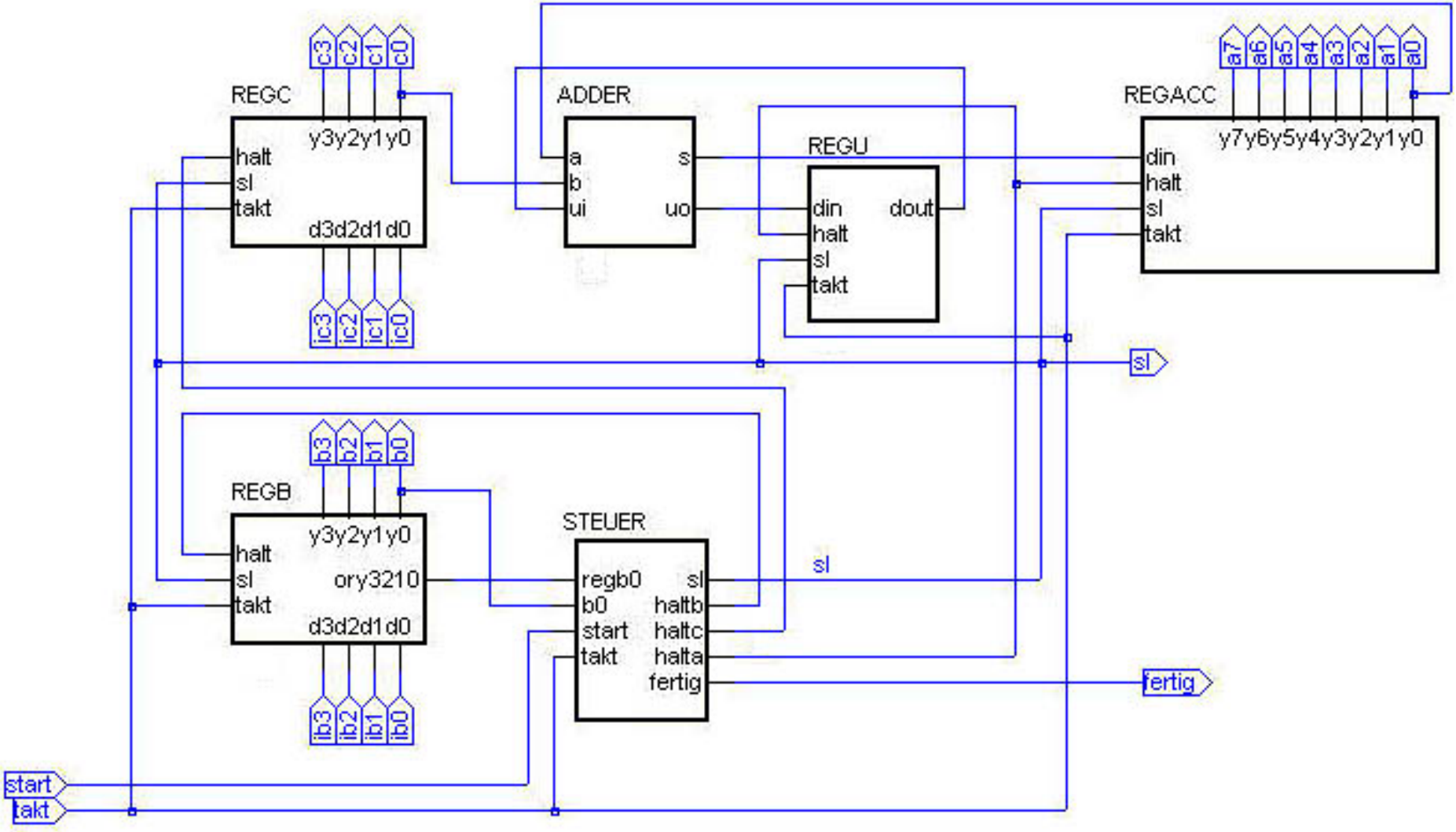
$$hc = \bar{V} \vee q_3$$

Logikplan der Steuerung für 4 Bit- Multiplizierer

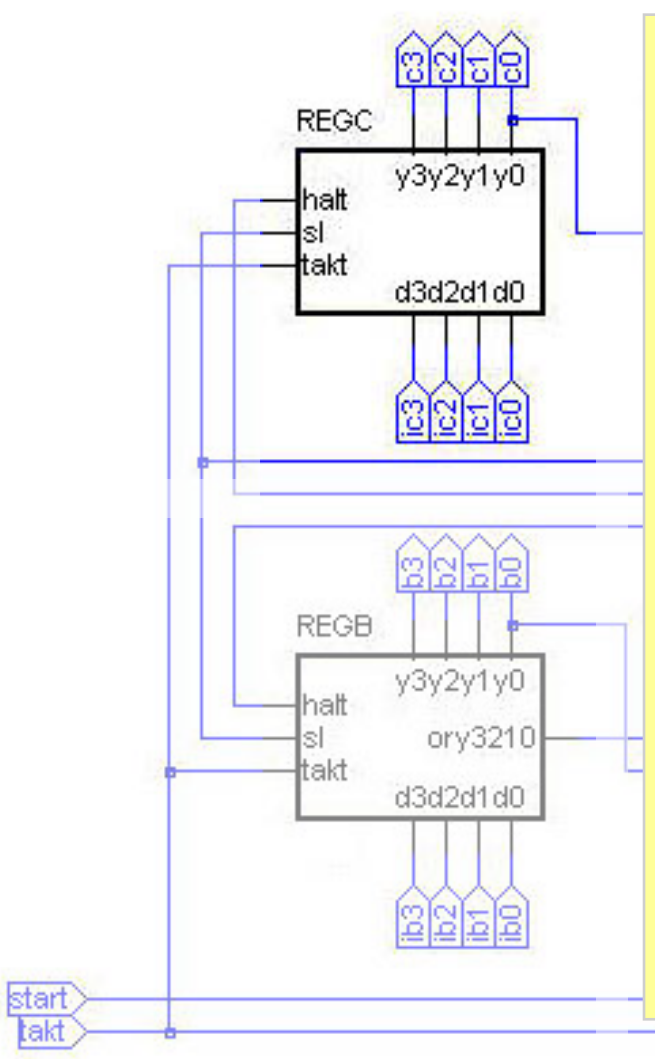




Design der Gesamtschaltung

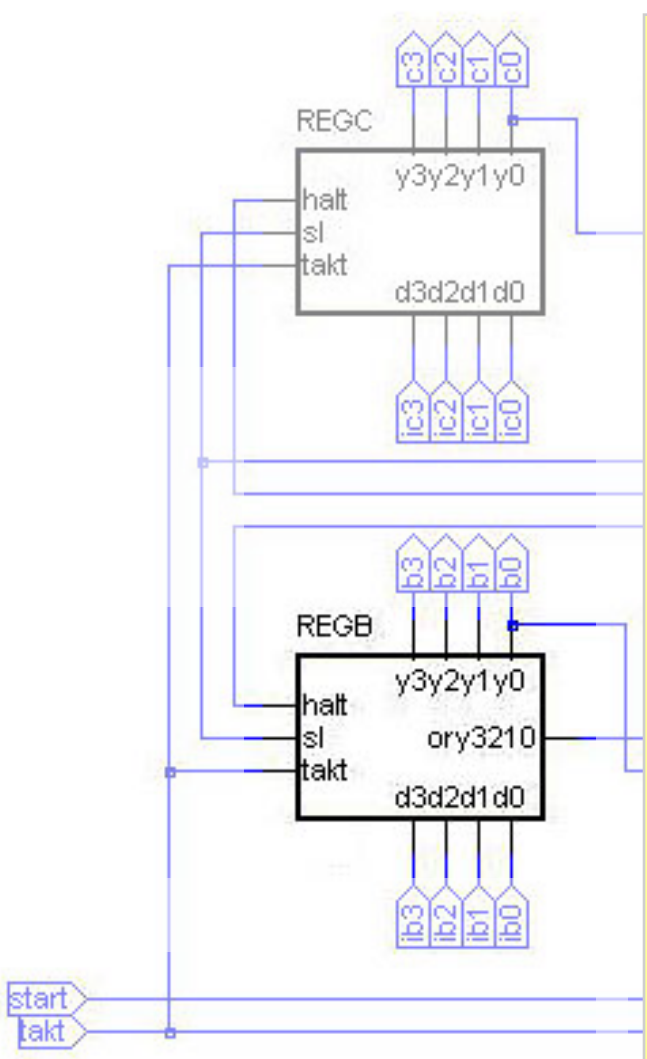


Design der Gesamtschaltung Register (C)



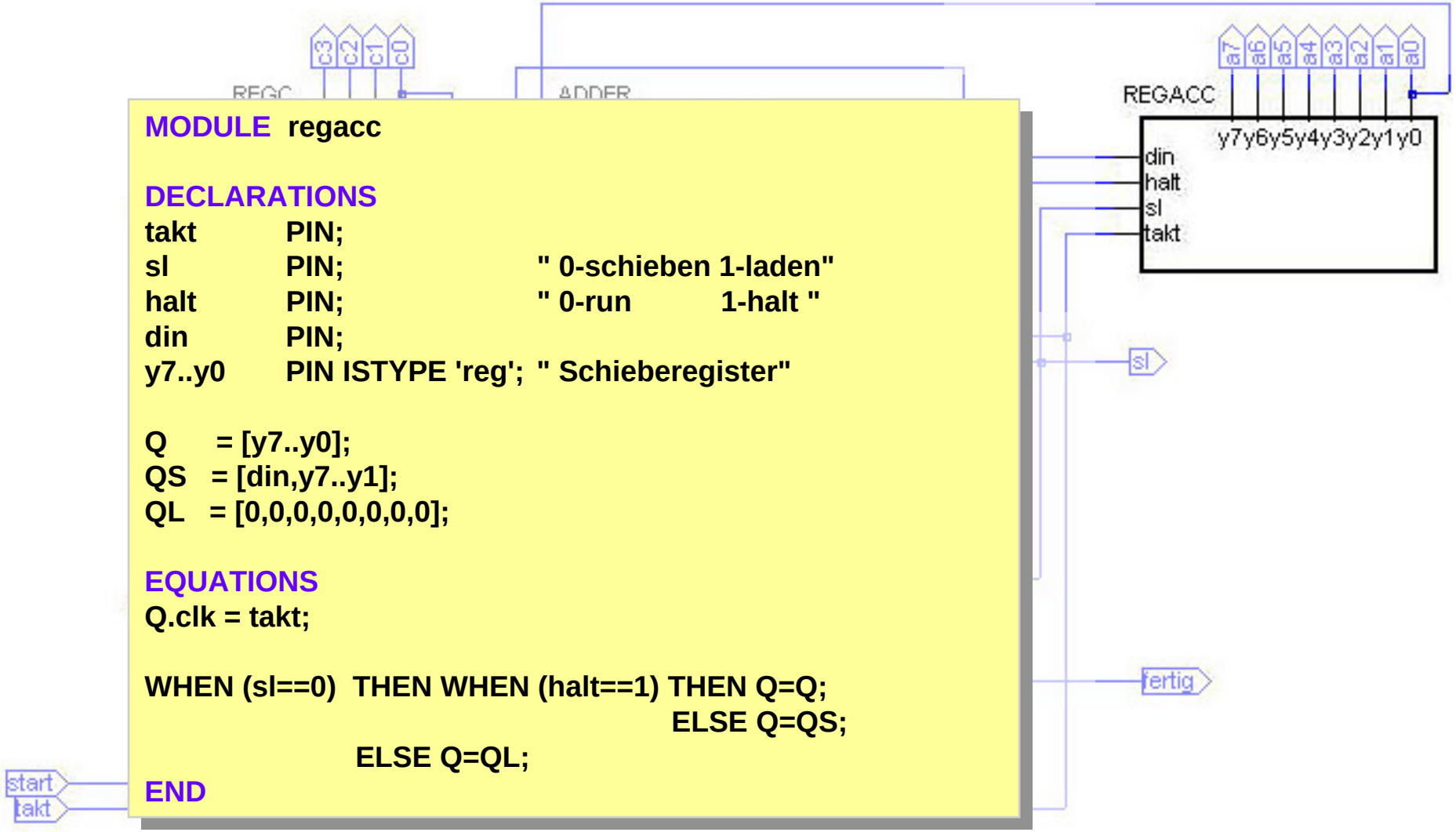

```
MODULE  regc

DECLARATIONS
takt      PIN;
sl        PIN;                  " 0-schieben 1-laden"
halt      PIN;                  " 0-run        1-halt"
d3..d0    PIN;                  " Dateneingaenge "
y3..y0    PIN ISTYPE 'reg';     " Schieberegisterteil sichtbar"
y8..y4    NODE ISTYPE 'reg';    " Schieberegisterteil vergraben"

Q         = [y8..y0];
QS        = [y0,y8..y1];
QL        = [0,0,0,0,0,d3..d0];

EQUATIONS
Q.clk     = takt;

  WHEN (sl==0)  THEN WHEN (halt==1) THEN Q=Q;
                                    ELSE Q=QS;
                ELSE Q=QL;

END
```

Design der Gesamtschaltung Register (B)



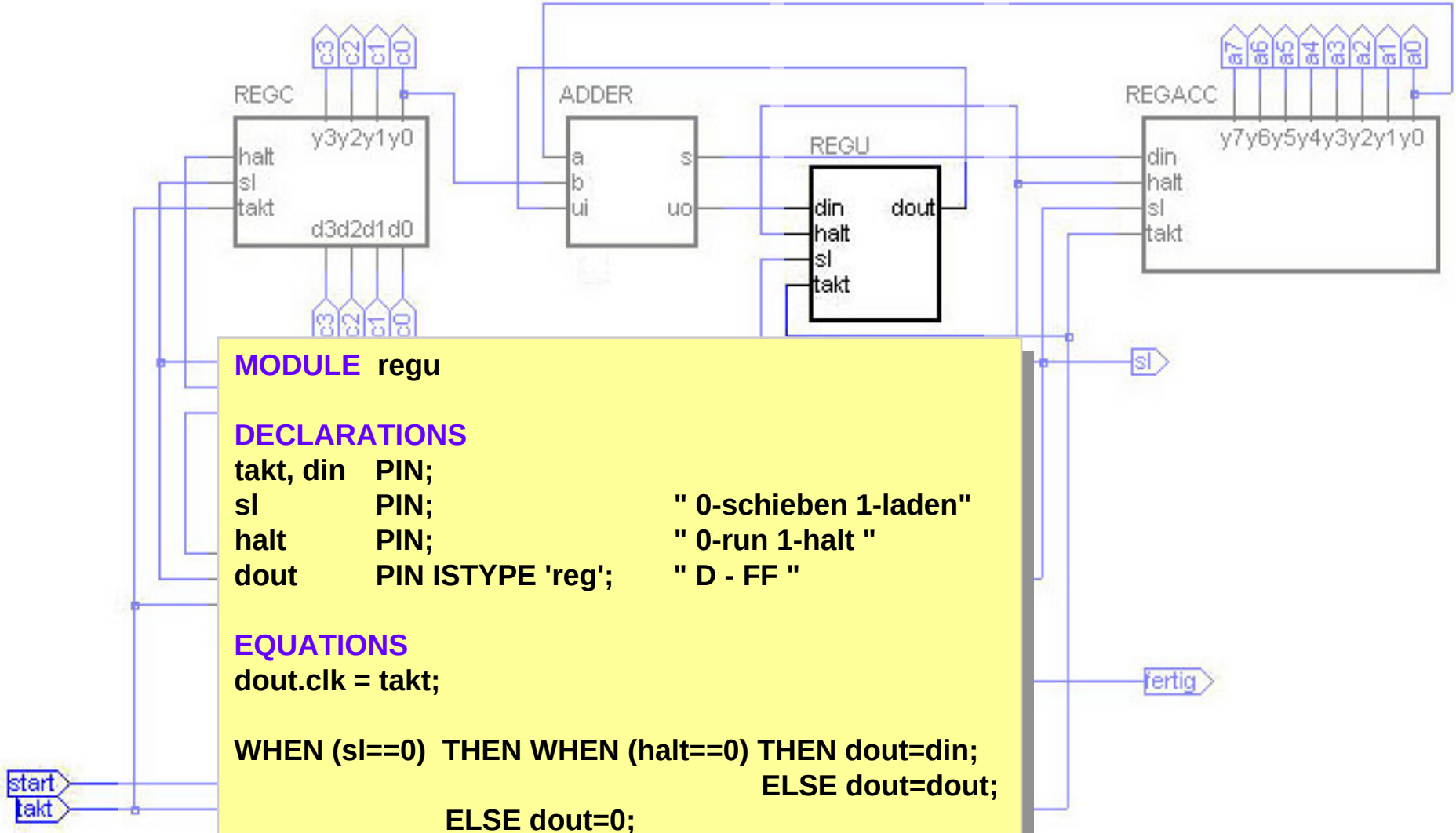

```
MODULE  regb

DECLARATIONS
takt        PIN;
sl          PIN;                " 0-schieben 1-laden"
halt        PIN;                " 0-run        1-halt"
ory3210     PIN;                " 0 wenn regb==0
d3..d0      PIN;                " Dateneingaenge "
y3..y0      PIN ISTYPE 'reg';   " Schieberegister "

Q       = [y3..y0];
QS      = [0,y3..y1];
QL      = [d3..d0];

EQUATIONS
Q.clk       = takt;

ory3210     = y3 # y2 # y1 # y0;

WHEN (sl==0)  THEN WHEN (halt==1) THEN Q=Q;
                                  ELSE Q=QS;
              ELSE Q=QL;
END
```

Design der Gesamtschaltung Register (AC)



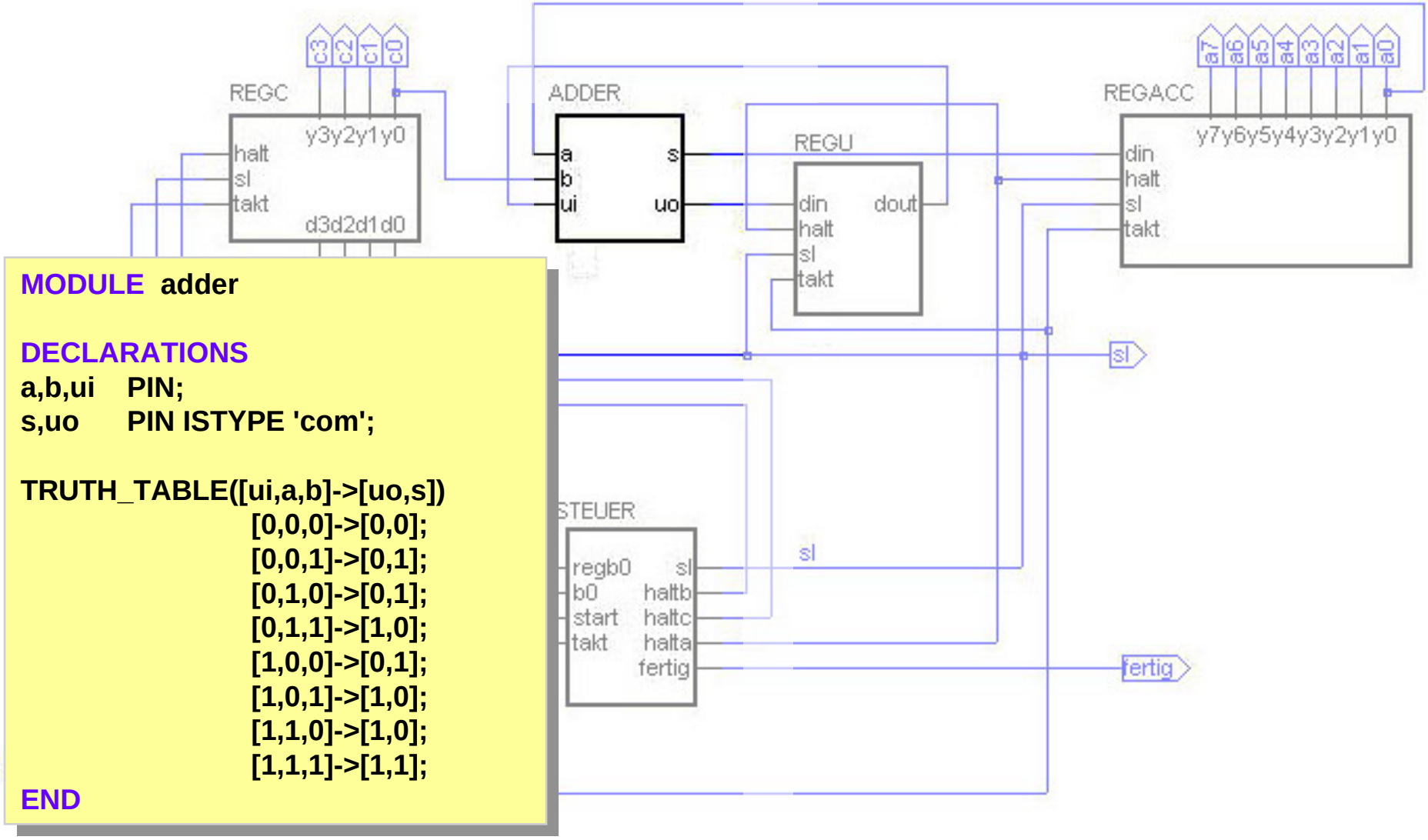

```
MODULE  regacc

DECLARATIONS
takt        PIN;
sl          PIN;                " 0-schieben 1-laden"
halt        PIN;                " 0-run         1-halt "
din         PIN;
y7..y0      PIN ISTYPE 'reg';   " Schieberegister"

Q     = [y7..y0];
QS    = [din,y7..y1];
QL    = [0,0,0,0,0,0,0,0];

EQUATIONS
Q.clk = takt;

WHEN (sl==0)  THEN WHEN (halt==1) THEN Q=Q;
                                  ELSE Q=QS;
              ELSE Q=QL;
END
```

Design der Gesamtschaltung Register (U)



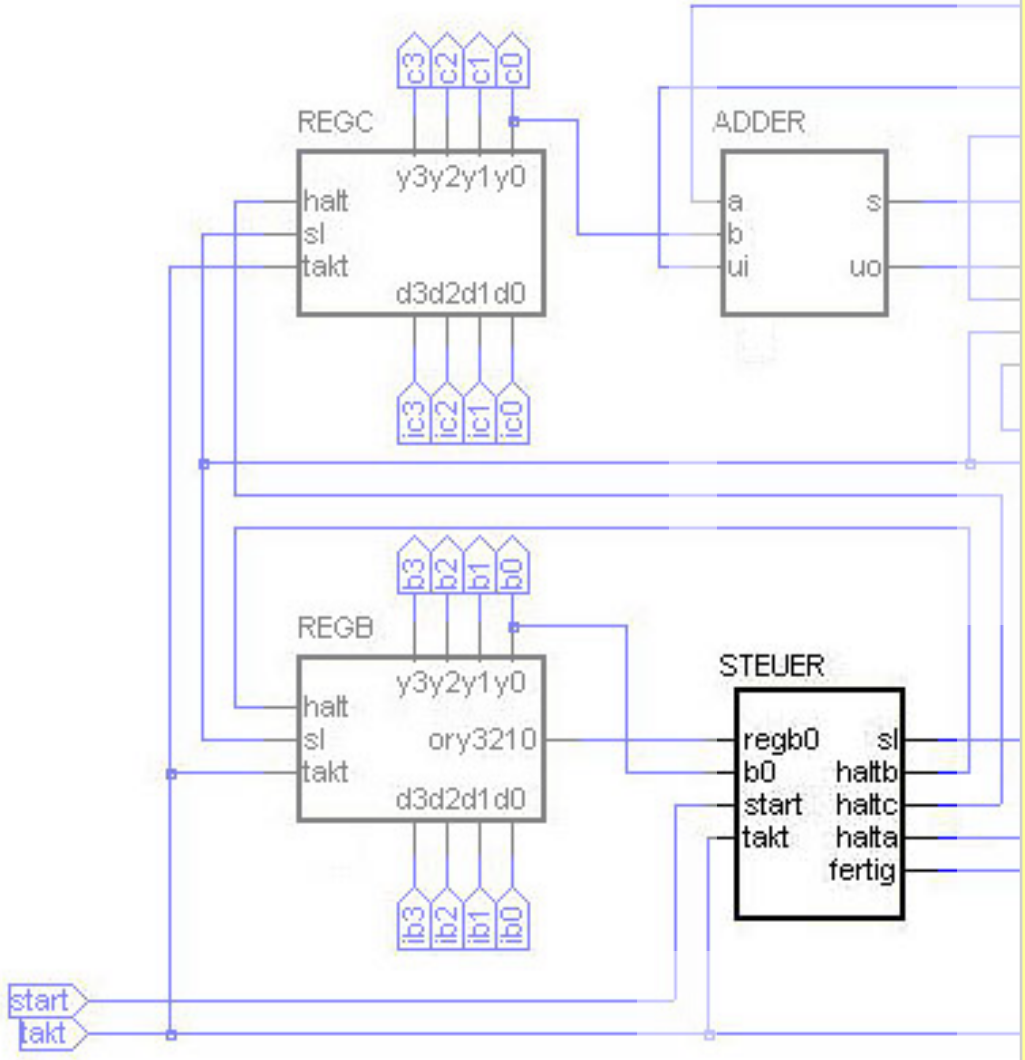

```
MODULE  regu

DECLARATIONS
takt, din   PIN;
sl          PIN;                    " 0-schieben 1-laden"
halt        PIN;                    " 0-run 1-halt "
dout        PIN ISTYPE 'reg';       " D - FF "

EQUATIONS
dout.clk = takt;

WHEN (sl==0)  THEN WHEN (halt==0) THEN dout=din;
                                  ELSE dout=dout;
              ELSE dout=0;
END
```

Design der Gesamtschaltung Adder




```
MODULE  adder

DECLARATIONS
a,b,ui    PIN;
s,uo      PIN ISTYPE 'com';

TRUTH_TABLE([ui,a,b]->[uo,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];

END
```

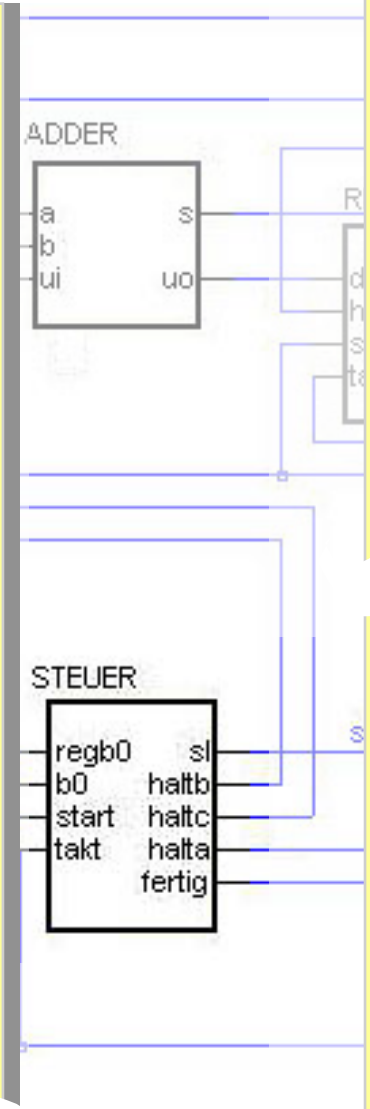
Gesamtschaltung; Steuerung als Boolesche Gleichungen




```
MODULE  steuer

DECLARATIONS
takt        PIN;
regb0       PIN;
b0          PIN;
start       PIN;
sl          PIN ISTYPE 'com';
halta       PIN ISTYPE 'com';
haltb       PIN ISTYPE 'com';
haltc       PIN ISTYPE 'com';
fertig      PIN ISTYPE 'com';
q3..q0  NODE ISTYPE 'reg';

Z      = [q3,q2,q1,q0];

EQUATIONS
Z.clk   = takt;
Z.ap    = start;
q0.t    = !q3  # q1;
q1.t    = !q3 &  q0          # q1 & q0;
q2.t    =  q1 b0 & q3 & !q1 & q0   # !regb0 & !q1;
sl      = !b0 &  q0 ;
q3.t    =  q3 & !q0  # !regb0 & !q3   # q2 & q1 & q0;
fertig  =  & regb0 & q3 & q0 # q3 & q1;
halta   = !b0   # q3;
haltb   = !q3   # q0;
haltc   = !regb0  # q3;
END
```

Gesamtschaltung; Steuerung als Zustandsübergangsgleichungen

```
MODULE  steuer
DECLARATIONS
Takt, start     PIN;
Regb0, b0       PIN;
sl              PIN ISTYPE 'com';
halta           PIN ISTYPE 'com';
haltb           PIN ISTYPE 'com';
haltc           PIN ISTYPE 'com';
fertig          PIN ISTYPE 'com';
q3..q0  NODE ISTYPE 'reg';

" ---Eingangsbuchstaben--- "
X               = [b0,regb0];
X0   MACRO {(X == [0,0])};
X1   MACRO {(X == [0,1])};
X3   MACRO {(X == [1,1])};

" ---Zustandsbuchstaben--- “
Z       = [q3,q2,q1,q0];
Z0      = [0,0,0,0];
..
Z15   = [1,1,1,1];

" ---Ausgangsbuchstaben--- "
Y    = [fertig,sl,halta,haltb,haltc];
Ystart      MACRO {Y=[0,1,1,1,1];};
Yfertig     MACRO {Y=[1,0,1,1,1];};
Yb          MACRO {Y=[0,0,1,0,1];};
Yc          MACRO {Y=[0,0,1,1,0];};
Yac         MACRO {Y=[0,0,0,1,0];};
```




```
EQUATIONS
Z.clk  = takt;
Z.ap   = start;

STATE_DIAGRAM Z

STATE Z0:
        IF X3    THEN Z1    WITH Yac
        IF X1    THEN Z1    WITH Yc
        IF X0    THEN Z9    WITH Yfertig
STATE Z1:
        IF X3    THEN Z2    WITH Yac
        IF X1    THEN Z2    WITH Yc

STATE Z8:
        IF X3    THEN Z0    WITH Yb
        IF X1    THEN Z0    WITH Yb

STATE Z9:
        IF X3    THEN Z9    WITH Yfertig
        IF X1    THEN Z9    WITH Yfertig
        IF X0    THEN Z9    WITH Yfertig
STATE Z15:
        IF X3    THEN Z0    WITH Ystart
        IF X1    THEN Z0    WITH Ystart
        IF X0    THEN Z0    WITH Ystart
END
```

Simulation am Beispiel 7 * 5 = 35