FRIEDRICH-SCHILLER-UNIVERSITÄT JENA
Fakultät für Mathematik und Informatik
Molekulare Algorithmen

Chemisches Digitalcomputermodell eines endlichen Automaten, der eine Ampelsteuerung realisiert

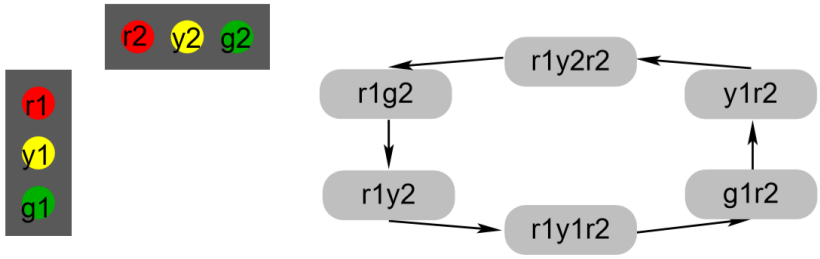
eingereicht von Roula Peekhaus und Mirko Johlke

Jena, den 12. Juli 2020

Inhaltsverzeichnis

1. Aufgabenstellung

- Der gesuchte endliche Automat M durchläuft zyklisch sechs Zustände, die jeweils die Ausgabe einer Ampelsteuerung symbolisieren:



- Der Automat arbeitet getaktet. Er schaltet mit jedem Takt in den nächsten Zustand und erzeugt die entsprechenden Ausgaben.
- Implementieren Sie den Automaten M als chemisches Digitalcomputermodell.
- Das Taktsignal wird durch einen chemischen Oszillator erzeugt, dem ggf. ein binärer Signalseparator nachgeschaltet wird.
- Pro Zustand werden drei Bit z_0, z_1, z_2 benötigt.
- Jedes Ausgabesignal `r1, y1, g1, r2, y2, g2` entspricht einer Spezies, deren Konzentration mit jedem Takt auf logisch `1` („leuchtet“) bzw. logisch `0` („leuchtet nicht“) gesetzt wird.
- Ermitteln der booleschen Schaltfunktionen für `z0, z1, z2` sowie für `r1, y1, g1, r2, y2, g2` und ggf. vereinfachen mittels Karnaugh-Plan oder einer alternativen Methode.
- Simulieren des Netzwerkverhaltens über mehrere Ampelzyklen.

2. Digitaler Schaltplan

Zu Beginn ist der erste Schritt, jedem Zustand des deterministischen endlichen Automaten M ein dreistelliges Bitmuster zuzuordnen. Die Bezeichnungen der Bits mit z_0, z_1 und z_2 könnte suggerieren, dass es sich um drei verschiedene Zustände handelt. Deswegen werden für eine eindeutigere Beschreibung abweichend von der Aufgabenstellung die Bits nicht z_0, z_1 und z_2 sondern b_1, b_2 und b_3 genannt. Da alle sechs Zustände von M zyklisch durchlaufen werden, sei zum einen angemerkt, dass in der Nummerierung der Bits keine Wertigkeit enthalten ist, weswegen die Indizes $1-3$ genauso vertretbar sind wie $0-2$.

Zum anderen ist es sinnvoll, wenn sich pro Zustandsänderung so wenig Bits wie möglich ändern - im Idealfall nur ein einziges Bit. Dadurch ist das System stabiler gegen das Eintreten undefinierter und damit ungewollter Zustände, die zum Beispiel durch verzögertes Umschalten zweier Bits auftreten können. Aus diesem Grund wurde für die Zuordnung der Zustände ein 3-Bit Gray-Code verwendet, durch den es möglich ist, den Idealfall zu erreichen, sodass nur ein Bit pro Zustandsänderung verändert wird. Ein 3-Bit Gray Code hat $2^3 = 8$ verschiedene Bitmuster, aber davon werden nur sechs benötigt. Das heißt, zwei Bitmuster werden keinem Zustand von M zugeordnet und sind damit verbotene Zustände, in denen sich der Automat nicht befinden darf.

	Zustand	**Bitmuster**
1	`r1g2`	`000`
2	`r1y2`	`001`
3	`r1y1r2`	`011`
4	`g1r2`	`010`
5	`y1r2`	`110`
6	`r1y2r2`	`100`

Tabelle 1: Zuordnung der Zustände des DEA zu einem Bitmuster nach Graycode.

In Tabelle 1 wird den Zuständen von M der 3-Bit Gray-Code so zugeordnet, dass auch beim Übergang vom letzten (6) zum ersten (1) Zustand nur ein Bit umgeschaltet wird. Des Weiteren ist aus dieser Tabelle abzuleiten, dass die verbotenen Zustände `101` und `111` sind.

Nachdem diese Zuordnung festgelegt ist, kann die Überführungstabelle (vgl. Tabelle 2) von einem Zustand in den Folgezustand angeführt werden. Dazu werden drei neue Bits b'_1, b'_2 und b'_3 eingeführt. Diese repräsentieren die logischen Werte, die die Bit b_i im nächsten Zustand annehmen sollen.

#	b_1	b_2	b_3	b'_1	b'_2	b'_3
1	0	0	0	0	0	1
2	0	0	1	0	1	1
3	0	1	1	0	1	0
4	0	1	0	1	1	0
5	1	1	0	1	0	0
6	1	0	0	0	0	0

Tabelle 2: Überführungstabelle.

Durch diese Zuordnung wird definiert, in welcher Reihenfolge die Bitmuster ($b_1b_2b_3 \rightarrow b'_1b'_2b'_3$) durchgeschaltet werden. Diese Reihenfolge entspricht der durch den Automaten M in der Aufgabenstellung vorgegebenen Zustandsfolge. Nun wird aus Tabelle 1 und Tabelle 2 der digitale Schaltplan erstellt, aus dem dann das chemische Digitalcomputermodell abgeleitet werden kann.

2.1. Schaltplan der Zustandbits

Dazu kann für jedes Bit b'_1, b'_2 und b'_3 jeweils eine disjunktive Normalform (kurz: DNF = über Disjunktionen verknüpfte Konjunktionsterme) aus logischen Verknüpfungen der Bits b_1, b_2 und b_3 erstellt werden. Die Vorgehensweise ist, dass für jede logische `1` eines b'_i aus Tabelle 2 die b_i über Konjunktionen so verknüpft werden, dass der resultierende Konjunktionsterm nur dann wahr ist, wenn sich der Automat M in dem Zustand befindet, in dem das b'_i die logische `1` hat. Zum Beispiel für Zustand #4 (`010`) ist b'_1 logisch `1` und der Konjunktionsterm aus b_1, b_2 und b_3, der in Zustand #4 wahr ist, ist $\neg b_1 b_2 \neg b_3$.

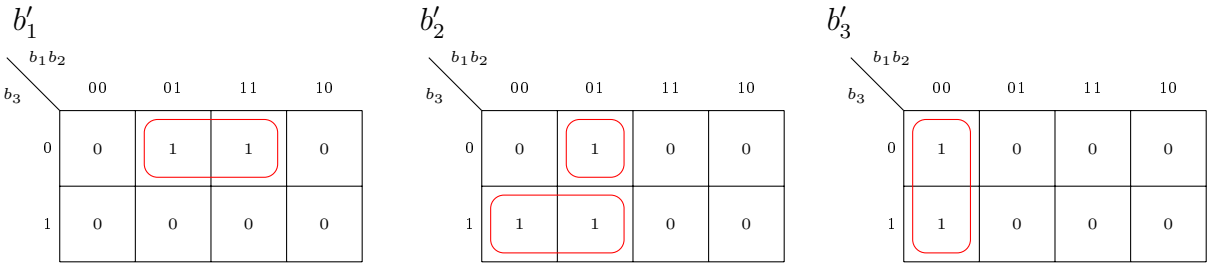
Alle Konjunktionsterme für ein b'_i werden am Ende über Disjunktionen verknüpft. Daraus ergeben sich für b'_1, b'_2 und b'_3 folgende disjunktive Normalformen:

$$
\begin{array}{lllllll}
b'_1: & \neg b_1 b_2 \neg b_3 & \vee & b_1 b_2 \neg b_3 & & \\
b'_2: & \neg b_1 \neg b_2 b_3 & \vee & \neg b_1 b_2 b_3 & \vee & \neg b_1 b_2 \neg b_3 \\
b'_3: & \neg b_1 \neg b_2 \neg b_3 & \vee & \neg b_1 \neg b_2 b_3 & &
\end{array}
$$

Auf diese Art braucht man zur Bestimmung des logischen Wertes vom Folgezustand eines Bits 31 logische Gatter (`Not, And, Or`). Bei diesem Anwendungsfall ist die Anzahl der Logikgatter zwar noch überschaubar, allerdings lässt sie sich über einen Karnaughplan für jedes b'_i reduzieren. Aus einem solchen Plan lässt sich ableiten, wie die Bits b_1, b_2 und b_3 logisch verknüpft werden können, um mit einer minimalen Anzahl logischer Gatter auszukommen. Von den zwei möglichen Methoden *Minterm* und *Maxterm* wird die Minterm-Methode verwendet, da die Tabelleneinträge überwiegend aus Nullen bestehen. Bei dieser Methode erhält man als Ergebnis ebenfalls eine DNF. Die Vorgehensweise ist, dass in unserem Anwendungsfall für jedes b'_i ein Karnaughplan erstellt wird. Bei drei Bits wird eine $2x4$ Tabelle erstellt, deren zwei Zeilen b_3 mit logisch `0` und `1` repräsentieren und die vier Spalten $b_1 \wedge b_2$ mit Belegungen von b_1 und b_2 in der Reihenfolge `00, 01, 11, 10` repräsentieren. Die Einträge des Karnaughplans von beispielsweise b'_1 entsprechen den logischen Werten, die b'_1 für die Belegung von b_{1-3} der jeweiligen Zeile und Spalte annimmt. Nachdem diese Tabelle ausgefüllt ist, werden benachbarte Einsen zu Gruppen zusammengefasst. Diese Gruppierung darf über den Rand hinaus gegenüber in der Tabelle fortgesetzt werden, aber darf nur eine Größe haben, die eine Zweierpotenz (1, 2, 4, 8, etc.) ist.

So erhalten wir nach dem Aufstellen der Karnaughpläne (vgl. Abb. 1) für b'_1, b'_2 und b'_3 jeweils eine DNF, die eine Minimalanzahl logischer Gatter beschreiben, die benötigt werden, um aus den Bits b_1, b_2 und b_3 die logischen Werte der Folgezustände zu bestimmen.

Aus diesen disjunktiven Normalformen lässt sich nun ein Schaltplan zum taktgesteuerten Durchschalten der sechs Zustände der Ampelsteuerung aufstellen. Zusätzlich zu den nun 12 statt vorher 31 logischen Gattern, die sich aus den Karnaughplänen ergeben, wird für jedes Bit ein FlipFlop zum Speichern des neuen Zustandes benötigt sowie ein



DNF: $b_2\neg b_3$ DNF: $\neg b_1 b_2 \neg b_3 \vee \neg b_1 b_3$ DNF: $\neg b_1 \neg b_2$

Abbildung 1: Karnaugh-Plan und DNF für b'_1, b'_2 und b'_3.

Taktgeber. Im Schaltplan (vgl. Abb. 2) wird bereits ein Problem adressiert, das bei der Simulation des chemischen Digitalcomputermodells aufgetreten ist und behoben wurde:

Während der Taktgeber auf logisch `1` ist, ändern sich die Outputs der FlipFlops (die b'_i werden übernommen und damit zu den neuen b_i) und daraufhin fälschlicherweise auch die Outputs der Logikgatter, die die Bits b'_1, b'_2 und b'_3 symbolisieren. Die Outputs dieser Logikgatter fließen wiederum in die FlipFlops ein und überschreiben damit den Wert, der eigentlich gespeichert werden sollte. So passiert es, dass in einem Takt ein undefiniertes Verhalten auftritt. Zum Beispiel wird ein Takt übersprungen oder eine Spezieskonzentration wird $\sim 0,5$ und springt undefiniert zu einem Bit. Um dieses Problem zu lösen, wird den Outputs der b'_i jeweils ein Gate nachgeschaltet, das öffnet, wenn der Takt logisch `0` ist. Somit kann der Input der FlipFlops, während der Takt logisch `1` ist, nicht geändert und der neue Zustand sicher gespeichert werden.

Eine weitere Maßnahme, um den Automaten gegen Fehler robuster zu machen ist, dass Überführungsfunktionen für die verbotenen Zustände in Abb. 1 vorgesehen werden. Befindet sich der Automat in einen der beiden Zustände `101` oder `111`, so werden alle Bits im nächsten Zustand auf `000` gesetzt. Auch für die Ampellampen werden später im Abschnitt '2.2 Schaltplan der Ampelbits' die verbotenen Zustände berücksichtigt, indem die Lampen nicht leuchten werden, wenn sich der Automat in einem dieser Zustände befindet.

2.2. Schaltplan der Ampelbits

Die Zustände der Ampelsteuerung können nun durchgeschaltet werden und was noch fehlt, damit am Ende die Lampen der Ampel leuchten, ist die Zuordnung vom Zustandsbitmuster zu den Ampellampen. Die Lampen werden durch sechs neue Bits r_1, y_1, g_1, r_2, y_2 und g_2 repräsentiert, für die logisch `0` 'leuchtet nicht' und logisch `1` 'leuchtet' bedeutet.

Auch für jedes dieser sechs neuen Bits muss ein kleiner Schaltplan erstellt und in den großen Schaltplan (vgl. Abb. 2) eingefügt werden, damit die Lampen im richtigen Zustand der Ampelsteuerung eingeschaltet bzw. ausgeschaltet werden. Dazu wird für die vier Bits r_1, y_1, r_2, y_2 aus Tabelle 3 ein Karnaughplan erstellt (vgl. Abb. 3), um aus dem Ergebnis die logischen Gatter für die Lampenbits abzuleiten. Für die Bits g_1 und g_2 ist das Aufstellen eines Karnaughplans nicht notwendig, da g_1 nur im Zustand `010` und g_2 nur bei `000` logisch `1` sind und somit durch einen Karnaughplan nicht vereinfacht werden können. Für die grünen Lampen ergibt sich die Schaltfunktion aus Tabelle 3 wie folgt: $g_1 : \neg b_1 b_2 \neg b_3$, sowie $g_2 : \neg b_1 \neg b_2 \neg b_3$.

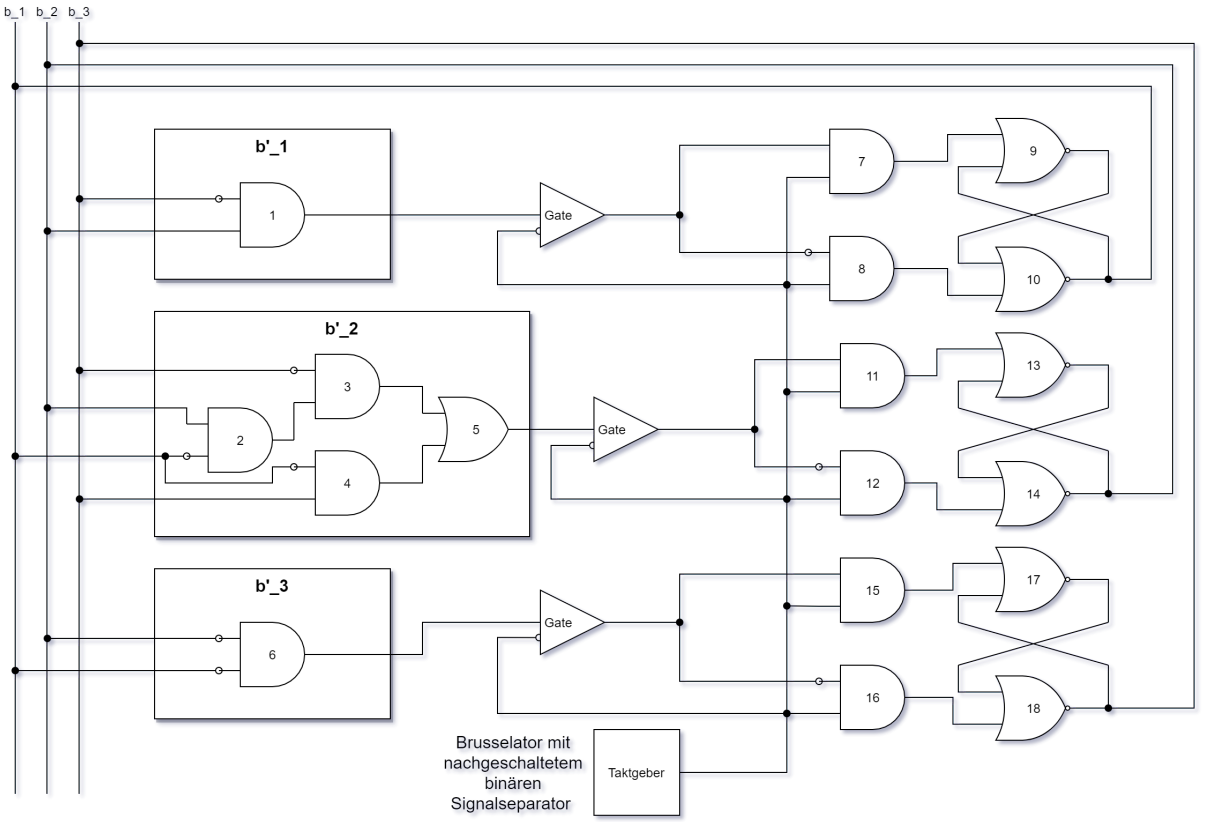


Abbildung 2: Schaltplan zum taktgesteuerten Durchschalten der Zustände des DEA M. Für eine eindeutige Identifizierung, Nachverfolgung und Wiederverwendung werden die logischen Gatter durchnummeriert.

Der Schaltplan aus Abb. 2 wird damit um Abb. 4 zum vollständigen Schaltplan für das chemische Digitalcomputermodell ergänzt (vgl. Abb. 11). Wie oben erwähnt ist an der Nummerierung der logischen Gatter zu sehen, dass einige wiederverwendet werden. Dies ist möglich, wenn alle Inputs zweier Gatter gleich sind. Im chemischen Digitalcomputermodell kann der Output von den mehrfach auftretenden logischen Gattern (zum Beispiel von Abb. 4 $\texttt{AND}_4$) für die folgenden Gatter verwendet werden, ohne dass es ein zweites Mal existieren muss.

#	Zustand	b_1	b_2	b_3	r_1	y_1	g_1	r_2	y_2	g_2
1	`r1g2`	0	0	0	1	0	0	0	0	1
2	`r1y2`	0	0	1	1	0	0	0	1	0
3	`r1y1r2`	0	1	1	1	1	0	1	0	0
4	`g1r2`	0	1	0	0	0	1	1	0	0
5	`y1r2`	1	1	0	0	1	0	1	0	0
6	`r1y2r2`	1	0	0	1	0	0	1	1	0

Tabelle 3: Zuordnung des Bitmusters zu den Ampellampenbits.

DNF für g_1 : $\neg b_1 b_2 \neg b_3$
DNF für g_2 : $\neg b_1 \neg b_2 \neg b_3$

r_1

b_3 \ b_1b_2	00	01	11	10
0	1	0	0	1
1	1	1	0	0

DNF: $\neg b_2 \neg b_3 \vee \neg b_1 b_3$

y_1

b_3 \ b_1b_2	00	01	11	10
0	0	0	1	0
1	0	1	0	0

DNF: $b_1 b_2 \neg b_3 \vee \neg b_1 b_2 b_3$

r_2

b_3 \ b_1b_2	00	01	11	10
0	0	1	1	1
1	0	1	0	0

DNF: $\neg b_1 b_2 \vee b_1 \neg b_3$

y_2

b_3 \ b_1b_2	00	01	11	10
0	0	0	0	1
1	1	0	0	0

DNF: $b_1 \neg b_2 \neg b_3 \vee \neg b_1 \neg b_2 b_3$

Abbildung 3: DNF für g_1 und g_2 und Karnaugh-Plan und DNF für r_1, y_1, r_2, y_2.

3. Chemisches Digitalcomputermodell

In diesem Kapitel wird der oben beschriebene Schaltplan in ein chemisches Digitalcomputermodell umgesetzt. Dieses Modell nutzt ausschließlich Massenwirkungskinetik. Das bedeutet, dass sich der durch eine Reaktion ausgelöste Fluss proportional zur Anzahl der vorhandenen zugehörigen Substratmoleküle verhält. Sättigungen werden hierbei nicht berücksichtigt. Als Einheit der Stoffkonzentrationen wird mmol/ml verwendet. Da die Einheiten für die Funktion des Systems irrelevant sind, werden die Konzentrationen im Folgenden allerdings ohne Einheit angegeben.

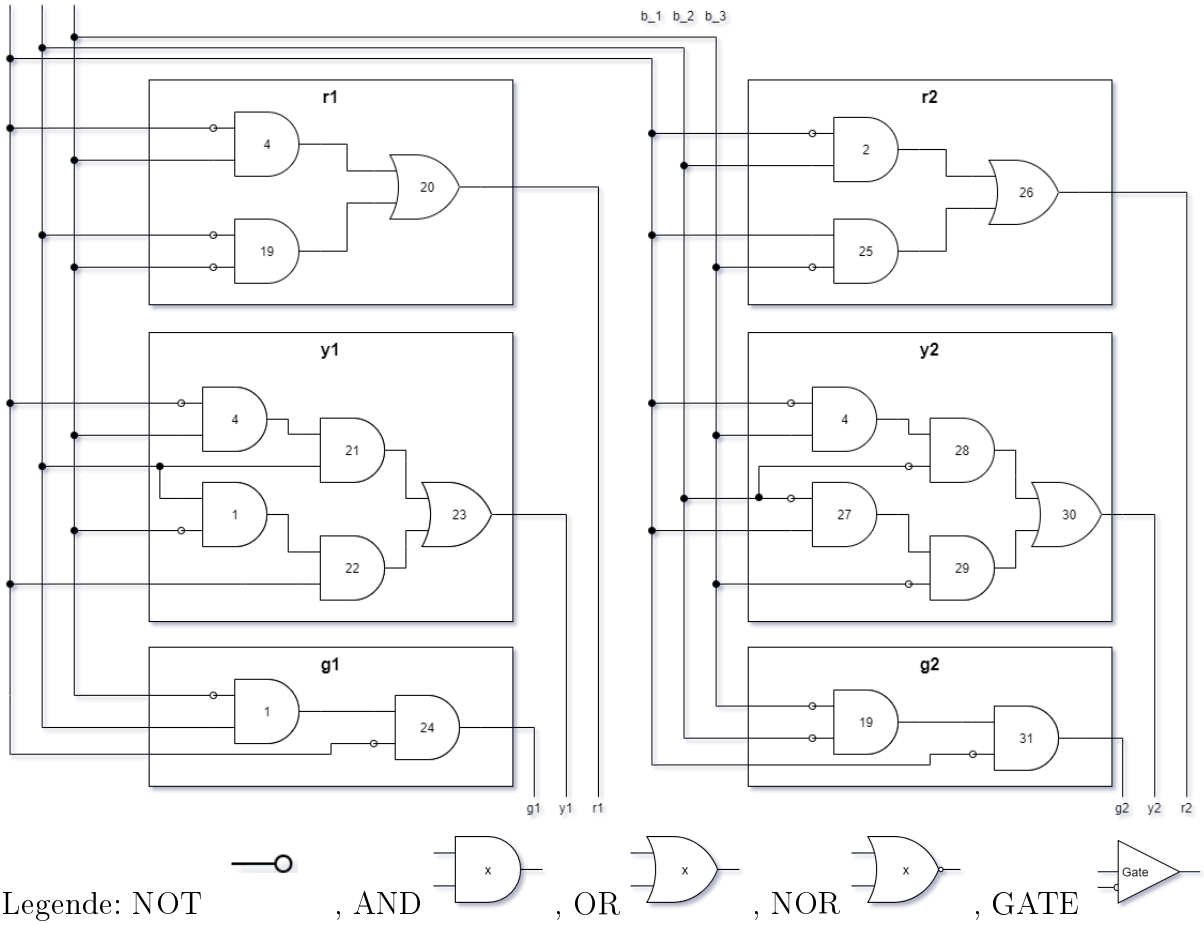


Abbildung 4: Fortsetzung von Schaltplan aus Abb. 2.

3.1. Umsetzung der booleschen Gatter

Zuerst werden jedem der Inputs bzw. Outputs der logischen Gatter jeweils zwei Spezies zugeordnet, deren Konzentrationen sich immer zwischen 0 und 1 bewegen. Die Namen dieser Spezies bestehen jeweils aus einer eindeutigen Bezeichnung, die sich auf den Schaltplan bezieht, und dem Suffix `_T` oder `_F`. Beispielsweise wird das Bit `b1` durch die beiden Spezies `b1_T` und `b1_F` repräsentiert. Die Konzentration der Spezies `b1_T` auf 1 oder 0 gerundet entspricht dabei dem momentanen logischen Wert von `b1`. Gegenläufig dazu ergibt sich die Konzentration von `b1_F` als $1 - [\texttt{b1\_T}]$. Um beim Zustand `b1b2b3` $=$ `000` zu starten, werden folglich die Initialkonzentrationen `[bi_T]` $= 0$ und `[bi_F]` $= 1$ festgelegt. Für die sechs verschiedenen, den Ampellampen zugeordneten Spezies gilt Ähnliches: befindet sich die Konzentration der zugehörigen Spezies zu Beginn eines Taktes nahe bei 1, ist die Lampe im folgenden Takt eingeschaltet, bei einer Konzentration von unter $0,5$ ausgeschaltet.

Alle Spezies der logischen Gatter werden so initialisiert, dass sie dem logischen Wert entsprechen, den sie auch im laufenden Modell im Zustand `b1b2b3 = 000` und `Taktgeber = 0` annehmen würden.

Nun wird jedes logische AND-Gatter des Schaltplans mithilfe von vier Reaktionen umgesetzt, von denen jede eine Zeile der zugehörigen logischen Übergangstabelle repräsentiert, wie man in Tabelle 4 sehen kann.

in_1	in_2	$in_1 \wedge in_2$	**Chemische Reaktion**
0	0	0	$\texttt{AND}_i\texttt{\_FF2F:}\ and_i\_T + in_1\_F + in_2\_F \rightarrow and_i\_F + in_1\_F + in_2\_F$
0	1	0	$\texttt{AND}_i\texttt{\_FT2F:}\ and_i\_T + in_1\_F + in_2\_T \rightarrow and_i\_F + in_1\_F + in_2\_T$
1	0	0	$\texttt{AND}_i\texttt{\_TF2F:}\ and_i\_T + in_1\_T + in_2\_F \rightarrow and_i\_F + in_1\_T + in_2\_F$
1	1	1	$\texttt{AND}_i\texttt{\_TT2T:}\ and_i\_F + in_1\_T + in_2\_T \rightarrow and_i\_T + in_1\_T + in_2\_T$

Tabelle 4: chemische Umsetzung eines logischen AND-Gatters.

Dort steht i für die Nummer des Gatters, das repräsentiert wird. Das Suffix des Reaktionsnamens (z.B. FF2F, lies: '`false false to false`') zeigt die logischen Werte der beiden verarbeiteten Inputs und des resultierenden Outputs. Die Spezies and_i_T und and_i_F repräsentieren den logischen Output des AND-Gatters auf die gleiche Weise wie die Spezies `bi_T` und `bi_F` die logischen Werte der Bits wiedergeben. Die Spezies in_1_T, in_1_F, in_2_T und in_2_F stehen stellvertretend für die Spezies, welche die logischen Werte der beiden Inputs des Gatters repräsentieren. Wichtig ist, dabei zu beachten, dass bei einem negierten Input die sich gegenläufig verhaltende Spezies im Vergleich zum Titel der Reaktion verwendet werden muss. Dies lässt sich am Beispiel vom Gatter AND1 veranschaulichen (vgl. Tabelle 5), dessen zweiter Input $\neg b_3$ ist.

b_2	$\neg b_3$	$b_2 \wedge \neg b_3$	**Chemische Reaktion**
0	0	0	$\texttt{AND}_1\texttt{\_FF2F:}\ and_1\_T + b_2\_F + b_3\_T \rightarrow and_1\_F + b_2\_F + b_3\_T$
0	1	0	$\texttt{AND}_1\texttt{\_FT2F:}\ and_1\_T + b_2\_F + b_3\_F \rightarrow and_1\_F + b_2\_F + b_3\_F$
1	0	0	$\texttt{AND}_1\texttt{\_TF2F:}\ and_1\_T + b_2\_T + b_3\_T \rightarrow and_1\_F + b_2\_T + b_3\_T$
1	1	1	$\texttt{AND}_1\texttt{\_TT2T:}\ and_1\_F + b_2\_F + b_3\_F \rightarrow and_1\_T + b_2\_T + b_3\_F$

Tabelle 5: chemische Umsetzung des logischen AND1-Gatters (vgl. Abb. 11).

Zusätzlich zu den vier Reaktionen, die für die Repräsentation der logischen Funktion eines Gatters notwendig sind, werden noch je AND-, OR- oder NOR-Gatter zwei autokatalytische Reaktionen eingefügt, die im Falle, dass der Output von 0 auf 1 oder von 1 auf 0 umgeschaltet werden muss, die Konzentrationsveränderungen beschleunigen. Diese lauten i.A.:

$$\texttt{AND}_i\texttt{/OR}_i\texttt{/NOR}_i\texttt{\_preserve\_F:}\quad and_i\_T + 2 * and_i\_F \rightarrow and_i\_F$$
$$\texttt{AND}_i\texttt{/OR}_i\texttt{/NOR}_i\texttt{\_preserve\_T:}\quad and_i\_F + 2 * and_i\_T \rightarrow and_i\_T$$

Analog zu den Reaktionen für das logische AND werden in Tabelle 6 Reaktionen für das logische ODER eingefügt.

in_1	in_2	$in_1 \vee in_2$	**Chemische Reaktion**
0	0	0	$\texttt{OR}_i\texttt{\_FF2F:}\ or_i\_T + in_1\_F + in_2\_F \rightarrow or_i\_F + in_1\_F + in_2\_F$
0	1	0	$\texttt{OR}_i\texttt{\_FT2T:}\ or_i\_F + in_1\_F + in_2\_T \rightarrow or_i\_T + in_1\_F + in_2\_T$
1	0	0	$\texttt{OR}_i\texttt{\_TF2T:}\ or_i\_F + in_1\_T + in_2\_F \rightarrow or_i\_T + in_1\_T + in_2\_F$
1	1	1	$\texttt{OR}_i\texttt{\_TT2T:}\ or_i\_F + in_1\_T + in_2\_T \rightarrow or_i\_T + in_1\_T + in_2\_T$

Tabelle 6: chemische Umsetzung eines logischen OR-Gatters.

Die entsprechenden Reaktionen für die NOR-Gatter erhält man, indem man bei den Substraten und Produkten der OR-Reaktionen die Spezies $\texttt{or}_i\texttt{\_F}$ durch $\texttt{nor}_i\texttt{\_T}$ und $\texttt{or}_i\texttt{\_T}$

durch $\texttt{nor}_i\texttt{\_F}$ ersetzt. Ein weiteres Bauteil im Schaltplan sind die drei Gates, die nur dann eine Veränderung der Konzentration der bi'-Spezies zulassen sollen, wenn der Taktgeber gerade logisch 0 ist, also die nachgeschalteten Gatter des FlipFlops inaktiv sind, weil sie den Taktgeber als Input eines ANDs besitzen. Natürlich wird im chemischen Modell auch der Taktgeber durch zwei Spezies, clk_T und clk_F, repräsentiert. Die Konzentrationen dieser Spezies pendeln kontinuierlich gegenläufig zwischen 0 und 1. Eine Konzentrationsveränderung der $\texttt{GateB}_i\texttt{'}$-Spezies soll nur dann zugelassen werden, wenn $[\texttt{clk\_T}] = 0$ und $[\texttt{clk\_F}] = 1$. Dazu werden die Ergebnisse der Berechnungen der $\texttt{b}_i\texttt{'}$, die in Form der Konzentrationen von $\texttt{and}_1\texttt{\_T}$ und $\texttt{and}_1\texttt{\_F}$ für $\texttt{b}_1\texttt{'}$, $\texttt{or}_5\texttt{\_T}$ und $\texttt{or}_5\texttt{\_F}$ für $\texttt{b}_2\texttt{'}$ und $\texttt{and}_6\texttt{\_F}$ und $\texttt{and}_6\texttt{\_T}$ für $\texttt{b}_3\texttt{'}$ vorliegen, nur dann in Konzentrationen der Spezies $\texttt{gateB}_i\texttt{'\_T}$ und $\texttt{gateB}_i\texttt{'\_F}$ umgesetzt, wenn die Spezies clk_F vorhanden ist. Das lässt sich durch folgende Reaktionen realisieren:

$\texttt{GateB}_i\texttt{'\_FF2F}$: $gateB'_i\_T + clk\_F + outB'_i\_F \rightarrow gateB'_i\_F + clk\_F + outB'_i\_F$
$\texttt{GateB}_i\texttt{'\_FT2T}$: $gateB'_i\_F + clk\_F + outB'_i\_T \rightarrow gateB'_i\_T + clk\_F + outB'_i\_T$

wobei $outB'_i$ stellvertretend für die Ergebnisse der vorgeschalteten logischen Gatter steht, die $\texttt{b}_i\texttt{'}$ repräsentieren.

3.2. Taktgeber

Um einen kontinuierlichen Taktgeber zu erzeugen, wird ein Brusselator genutzt, der aus folgenden vier Reaktionen besteht (vgl. [Hin13, S. 44]):

BrusselatorTP2S: $br\_P + br\_T \rightarrow br\_S$
BrusselatorSTT2TTT: $br\_S + 2 * br\_T \rightarrow 3 * br\_T$,
BrusselatorQ2T: $br\_Q \rightarrow br\_T$
BrusselatorT2W: $br\_T \rightarrow br\_W$

Dabei oszillieren die Spezies S und T, während P,Q und W nur verbraucht bzw. produziert werden. Um eine gleichbleibende Oszillation zu erzeugen und zu verhindern, dass ein Substrat aufgebraucht wird oder das Reaktionssystem übersättigt wird, werden die Initialkonzentrationen von P,Q und W als unveränderlich festgelegt. Konkret lauten sie: $[P] = 3$, $[Q] = 1$, $[W] = 0$. Die oszillierenden Spezieskonzentrationen von br_S und br_T werden initial auf $0,5$ gesetzt. Da die Berechnung der logischen Ausdrücke etwas Zeit für die asymptotische Annäherung der Spezieskonzentrationen an die richtigen Ergebnisse braucht, wird die Ratenkonstante für die Reaktionen des Brusselators nur halb so groß gewählt wie die Ratenkonstanten der Reaktionen für die logischen Ausdrücke. Ein weiterer Grund für diese Notwendigkeit ist, dass die Reaktionsgeschwindigkeit proportional zur Substratkonzentration ist. Diese bewegt sich bei allen logischen Operationen nur zwischen 0 und 1, wohingegen sie beim Brusselator auch auf über 3 ansteigt. Im hier beschriebenen Modell betragen die genannten Ratenkonstanten 0,05 beim Brusselator und 0,1 bei den logischen Operationen. Sie lassen sich aber skalieren, um z.B. realistischere Zeiten einer Ampelschaltung umzusetzen. Für die weitere Verbesserung des Signals mithilfe eines binären Signalseparators wird als Katalysator auch eine Spezies benötigt, die das Inverse des Oszillationsverlaufes widerspiegelt. Die dazu notwendige Inversion der Konzentration der oszillierenden Spezies br_S wird durch eine chemische Subtraktion durchgeführt. Dafür wird mithilfe chemischer Reaktionen der momentane Konzentrationswert der Spe-

zies `br_S` vom Maximalwert $(4,72)$, den die Konzentration der Spezies `br_S` während des Oszillationsverlaufes annimmt, abgezogen. Die zugrundeliegenden Reaktionen sind:

`Signalinvertierer1:` $br\_S \rightarrow br\_S + sInv1$
`Signalinvertierer2:` $sInv2 \rightarrow sInv2 + sInv3$
`Signalinvertierer3:` $sInv3 + sInv1 \rightarrow$
`Signalinvertierer4:` $sInv3 \rightarrow$

Hierbei passt sich die Konzentration von `sInv1` laufend der Konzentration von `br_S` an und wird deshalb mit $0,5$ initialisiert. Die Konzentration der Spezies `sInv2` dagegen repräsentiert durchgängig mit $4,72$ den Wert des Minuenden. Zuletzt fungiert die Konzentration der Spezies `sInv3` als das invertierte Oszillationssignal, weshalb ihr Initialwert auf $4,72$ (Maximalwert von [`br_S`]) $-0,5$ (Initialwert von [`br_T`]) $= 4,22$ gesetzt wird. Da sich die Konzentration von `sInv3` durch diese Reaktionen asymptotisch der richtigen Differenz nähert und sich das Oszillationssignal währenddessen jedoch stetig verändert, entspricht der Konzentrationsverlauf von `sInv3` nicht genau einer zum Verlauf von [`br_S`] inversen Kurve. Um die Anpassung an das Oszillationssignal zu beschleunigen, werden die Ratenkonstanten der Reaktionen für die Signalinversion auf das 10-Fache der Ratenkonstanten für die Reaktionen des Brusselators gesetzt. In diesem Modell entspricht das einer Ratenkonstante von $0,5$. Somit ist der invertierte Wert für die weitere Signalseparation eine ausreichende Näherung.

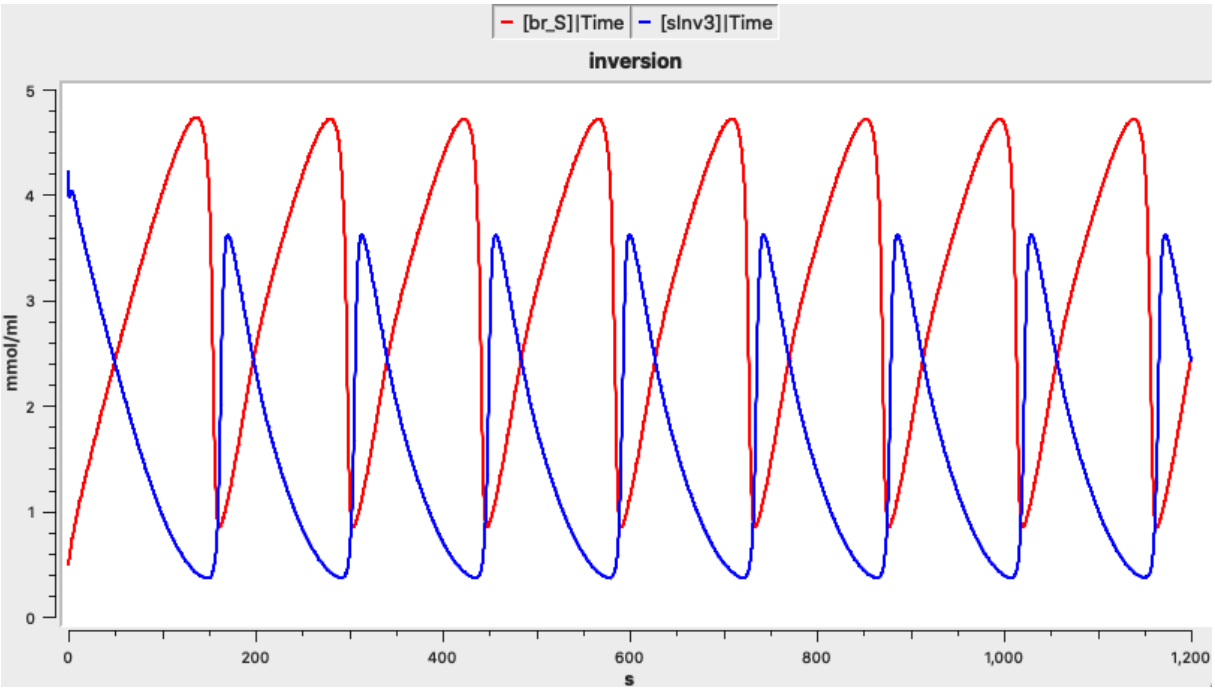


Abbildung 5: Oszillator- und invertierter Konzentrationsverlauf.
Das invertierte Signal (blau) verhält sich gegenläufig zum Konzentrationsverlauf der oszillierenden Spezies `br_S` (rot), auch wenn es nicht identische Werte annimmt.

Um einen Taktgeber zu erhalten, dessen Konzentrationswerte möglichst schnell zwischen 0 und 1 hin- und herspringen, damit es keine langen Umschaltphasen gibt und das Signal gut erkannt werden kann, wird dem Brusselator ein binärer Signalseparator nach-

geschaltet (vgl. [Hin13, S. 59]). Dieser besteht aus vier Stufen mit je vier Reaktionen, die dazu dienen, den Konzentrationsanstieg und -abfall zu beschleunigen und die Oszillationskurve plateauförmiger zu gestalten. Für jede Stufe i werden zwei Spezies, `sSepi_T` und `sSepi_F` eingeführt, wobei die Konzentration von `Ssepi_T` mit der Konzentration der oszillierenden Spezies `br_S` korreliert und `sSepi_F` mit der inversen Konzentration [`sInv3`]. Dann besteht jede Stufe aus den vier Reaktionen:

`Signalseparatori_F2T:`	$sSep_i - 1\_T + sSep_i\_F$	$\rightarrow$	$sSep_i - 1\_T + sSep_i\_T$
`Signalseparatori_T2F:`	$sSep_i - 1\_F + sSep_i\_T$	$\rightarrow$	$sSep_i - 1\_F + sSep_i\_F$
`Signalseparatori_autokatF:`	$sSep_i\_T + 2 * sSep_i\_F$	$\rightarrow$	$3 * sSep_i\_F$
`Signalseparatori_autokatT:`	$sSep_i\_F + 2 * sSep_i\_T$	$\rightarrow$	$3 * sSep_i\_T$

Die ersten beiden Reaktionen nutzen das jeweils gegenläufige Oszillationssignal der vorherigen Stufe als Katalysator, um die steigende Flanke des stufeneigenen Verlaufs zu verschärfen. Die anderen beiden Reaktionen verstärken diesen Effekt, indem sie wie bei den logischen Gattern ein schnelleres Umschalten zwischen 0 und 1 ermöglichen. Für die Reaktion Signalseparator1 entspricht `sSepi-1_T` der oszillierenden Spezies `br_S` und `sSepi-1_F` der sich invers verhaltenden Spezies `sInv3`. Außerdem heißen die Spezies der vierten Stufe `clk_T` und `clk_F`, da sie zugleich der endgültige Taktgeber für alle weiteren Reaktionen sind. Weil auch hier eine schnelle Anpassung an das Oszillatorsignal notwendig ist, werden die Ratenkonstanten ebenfalls auf 0.5 und damit das 10-Fache der Ratenkonstanten beim Brusselator gesetzt.

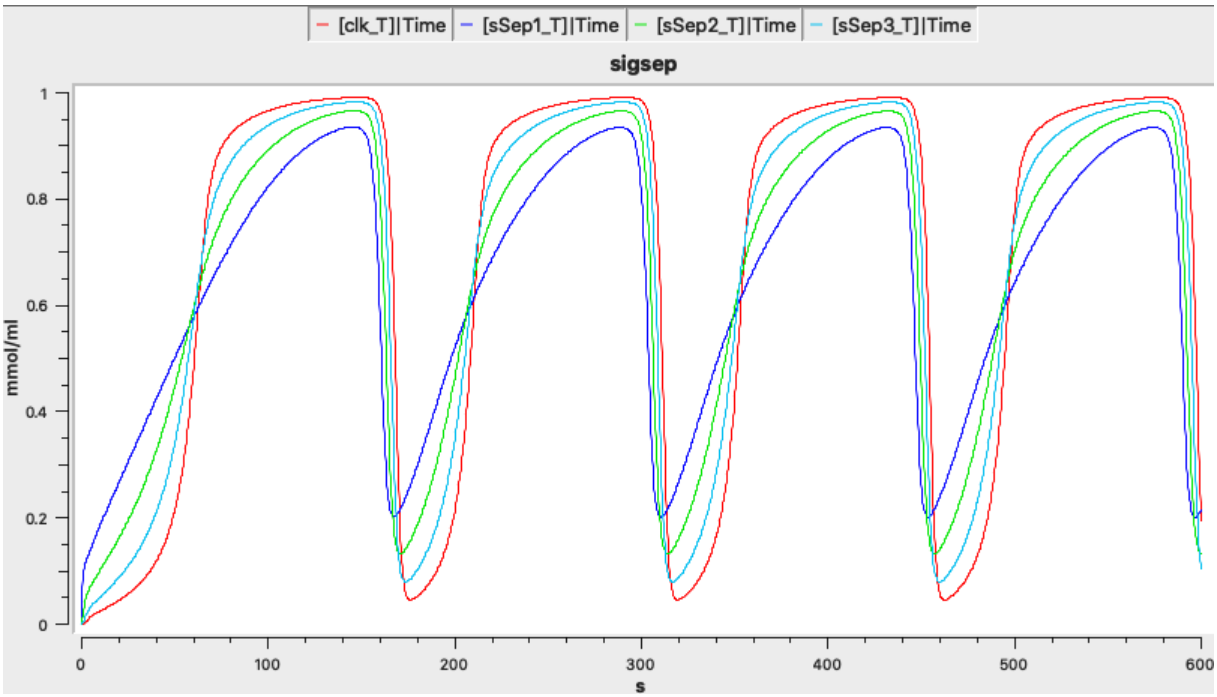


Abbildung 6: Mithilfe von Copasi modellierte Verläufe der Signalseparationskaskade. Es ist gut zu erkennen, dass der Anstieg nach der vierten Kaskadenstufe (rot) beschleunigt und die Konzentrationen verglichen mit den Kaskadenstufen 1-3 (dunkelblau, grün, hellblau) gegen 0 und 1 gedrückt werden.

4. Simulationsergebnisse und Interpretation

Den Konzentrationsverläufen von `r1_T`, `y1_T`, `g1_T`, `r2_T`, `y2_T`, `g2_T` (vgl. Abb. 7 und Abb. 10) kann man entnehmen, dass die Ampelschaltung funktioniert und genau die zu Beginn beschriebene Zustandsfolge durchlaufen wird. Auch die Konzentrationen von `b1_T`, `b2_T` und `b3_T` zeigen das erwünschte Verhalten (vgl. Abb. 8 und Abb. 9).

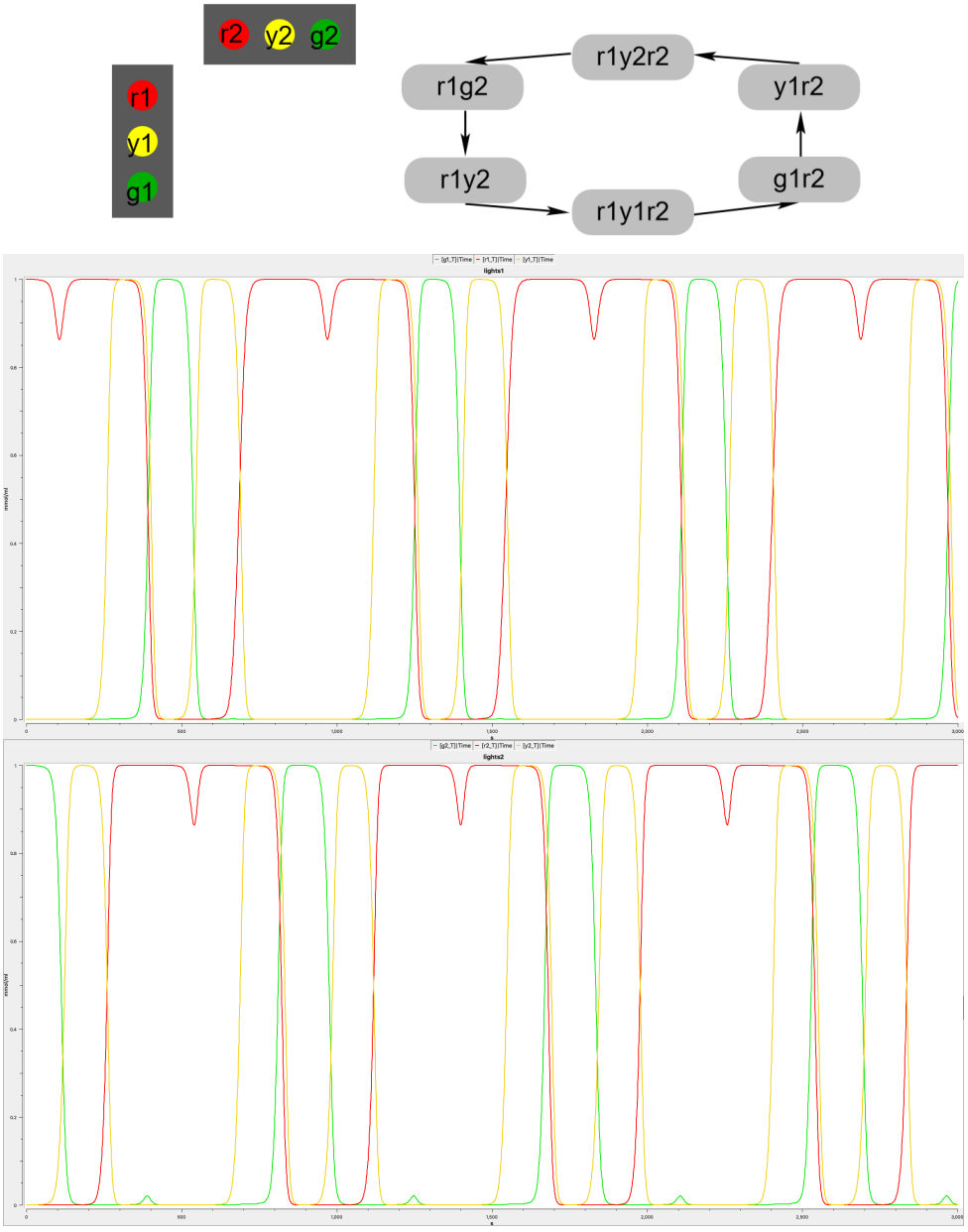
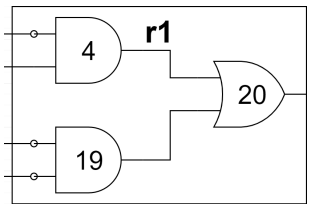


Abbildung 7: Mit Copasi simulierte Konzentrationsverläufe der Spezies für die Ampellampen `r1, y1, g1, r2, y2, g2`. Die Farben der Kurven entsprechen den Farben der Lampen.

Auffällig ist lediglich, dass die Konzentrationen der Spezies für beide roten Lampen zwei Takte nach ihrer Aktivierung kurzzeitig auf etwa $0,86$ fallen. Das lässt sich durch die boolesche Architektur erklären:



Im Falle von `r1` kommt es nach dem ersten Takt dazu, dass `AND19` (gemäß Schaltplan) von 1 auf 0 schaltet, da `b3` auf 1 geschaltet und somit die Bedingung $\neg b2 \wedge \neg b3$ nicht mehr erfüllt ist. Zugleich schaltet aber `AND04` von 0 auf 1, da nun $\neg b1 \wedge b3$ gilt. Folglich fällt die Konzentration der Inputspezies `and19_T` des `OR20` ab, während die Konzentration der anderen Inputspezies, `and04_T`, dieses Gatters steigt, aber noch nicht ganz 1 erreicht hat. Demzufolge existieren für einen kurzen Moment einige Elemente der Spezies `and04_F` und `and19_F` gleichzeitig und führen dazu, dass die Reaktion `OR20_FF2F` in geringem Maß stattfinden kann. Da ein Wert von $0,86$ aber noch immer gut als zu 1 gehörig interpretiert werden kann und sich die Konzentration am Ende des Taktgebersignals wieder bei 1 befindet, ist das Modell durch dieses Rauschen in seiner Funktionalität nicht beeinträchtigt. Um den Effekt zu verringern, könnte man höhere Ratenkonstanten für die Reaktionen des `AND19` und `AND04` festlegen, oder eine Kopplung dieser beiden Gatter implementieren, sodass `AND19` erst schaltet, wenn `AND04` schon umgeschaltet hat. Analoges gilt für `r2`.

Eine weitere Auffälligkeit lässt sich in einem kleinen Anstieg von [g2_T] auf 0.02 beobachten. Dieser entsteht durch eine Anomalie von [and19_T], das scheinbar auf eine minimale Schwankung von [b2_T] reagiert. Für die Funktionalität des Modells ist diese Auffälligkeit aber nicht weiter beachtenswert.

A. Literaturverzeichnis

Literatur

Hinze, Dr.-Ing. Thomas. *Computer der Natur*. Deutsch. 31. März 2013.

B. Anhang

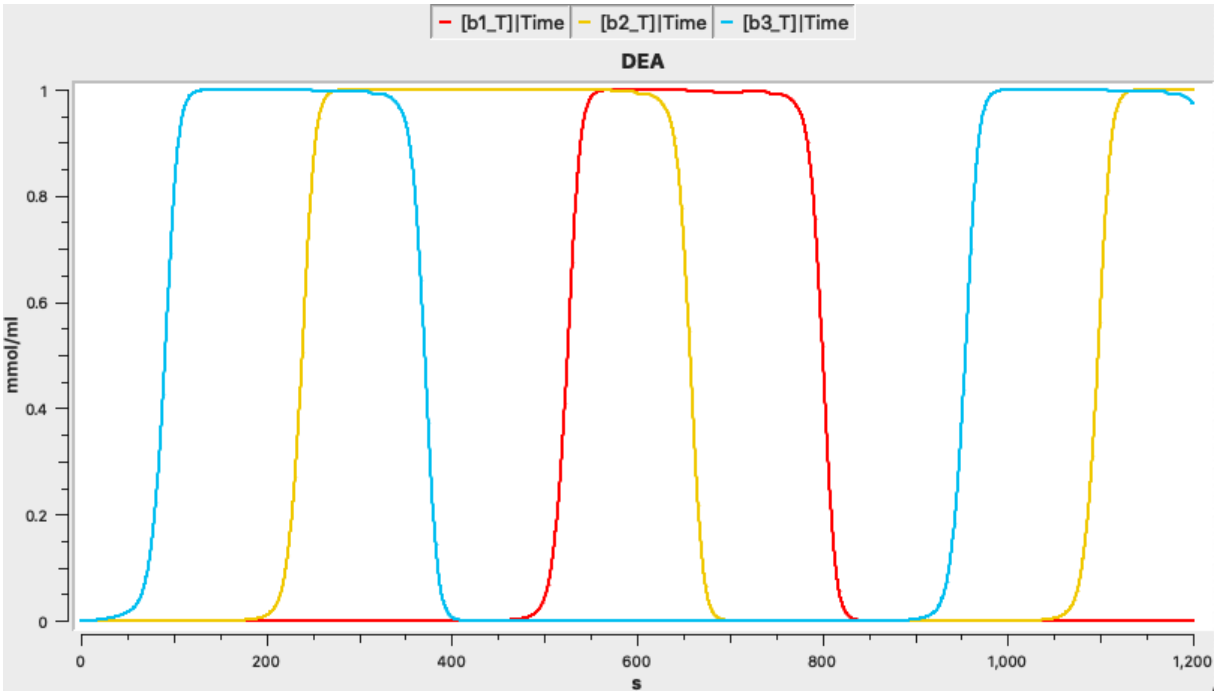


Abbildung 8: Konzentrationsverlauf der Bits `b1, b2, b3` ohne Taktgeber

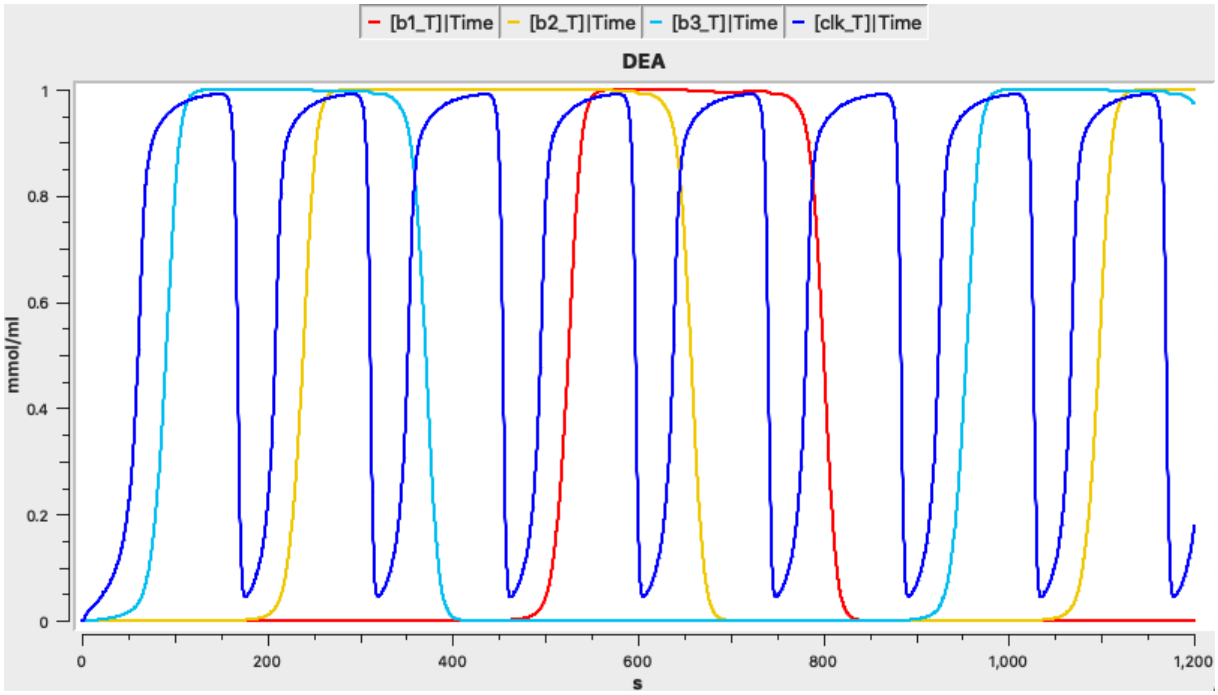


Abbildung 9: Konzentrationsverlauf der Bits `b1`, `b2`, `b3` mit Taktgeber

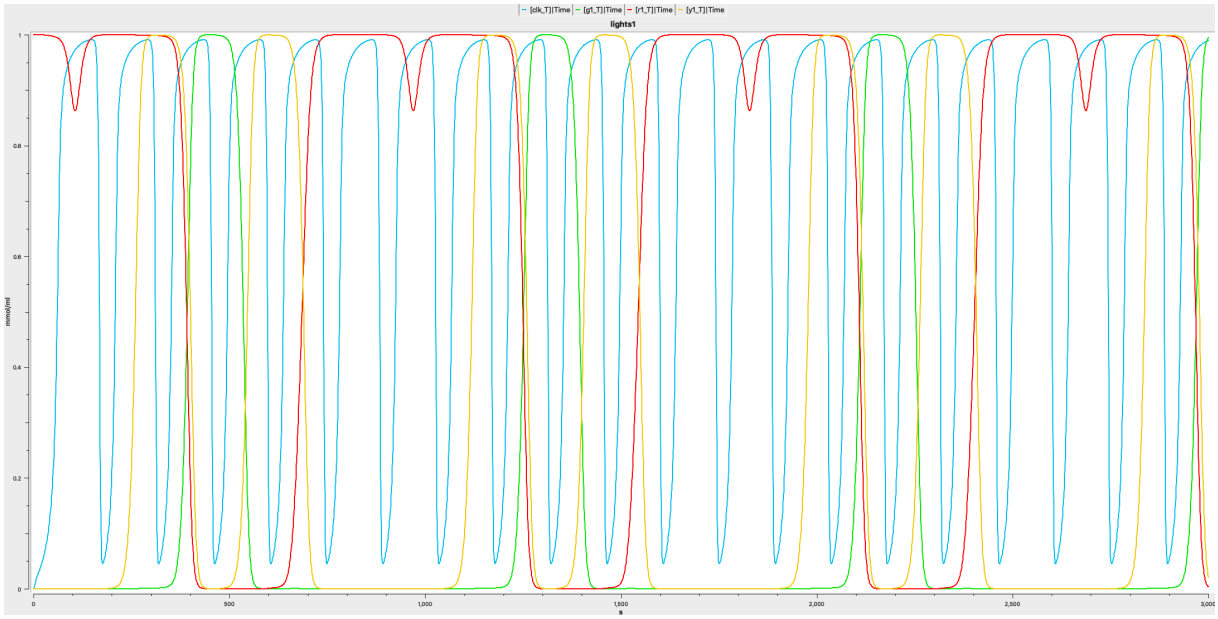
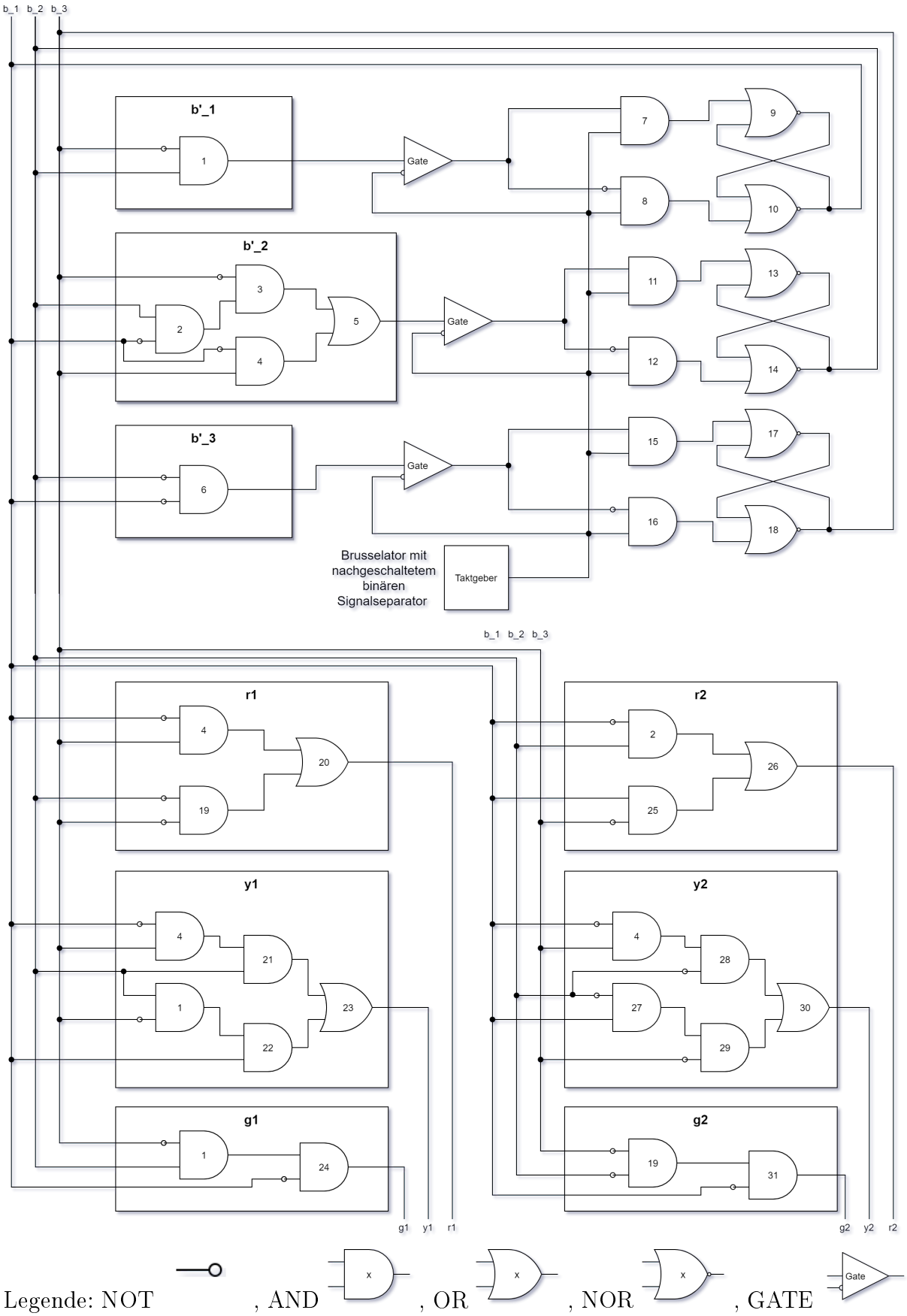


Abbildung 10: Ampellampen `r1`, `y1`, `g1` mit Taktgeber



Legende: NOT , AND , OR , NOR , GATE

Abbildung 11: Schaltplan des chemischen Digitalcomputermodells