BINARY COUNTING WITH CHEMICAL REACTIONS*

ALEKSANDRA KHARAM, HUA JIANG, MARC RIEDEL, and KESHAB PARHI

Electrical and Computer Engineering, University of Minnesota
Minneapolis, MN 55455
http://cctbio.ece.umn.edu
E-mail: {veden002, hua, mriedel, parhi}@umn.edu

This paper describes a scheme for implementing a binary counter with chemical reactions. The value of the counter is encoded by logical values of "0" and "1" that correspond to the absence and presence of specific molecular types, respectively. It is incremented when molecules of a trigger type are injected. Synchronization is achieved with reactions that produce a sustained three-phase oscillation. This oscillation plays a role analogous to a clock signal in digital electronics. Quantities are transferred between molecular types in different phases of the oscillation. Unlike all previous schemes for chemical computation, this scheme is dependent only on coarse rate categories for the reactions ("fast" and "slow"). Given such categories, the computation is exact and independent of the specific reaction rates. Although conceptual for the time being, the methodology has potential applications in domains of synthetic biology such as biochemical sensing and drug delivery. We are exploring DNA-based computation via strand displacement as a possible experimental chassis.

1. Introduction

In the nascent field of synthetic biology, researchers are striving to create biological systems with functionality not seen in nature. The field aims to apply engineering methods to biology in a deliberate way. Beyond engineering ends, such methods also provide a constructive means to validating new science. Understanding is achieved by constructing and testing simplified systems from the bottom up, teasing out and nailing down fundamental principles in the process.[1]

We bring a particular mindset to tackle the problem of synthesizing new biological functions. We tackle synthesis at a *conceptual* level, working with abstract molecular types. Working at this level, we implement *computational* constructs, that is to say, chemical reaction networks that compute specific outputs as a function of inputs. Then we map the conceptual designs onto specific chemical substrates.

We model the chemical dynamics in terms of mass-action kinetics:[2,3] reaction rates are proportional to (1) the quantities of the participating molecular types; and (2) reaction constants. We aim for robust constructs: systems that compute exact results independently of specific reaction constants. All of our designs are formulated in terms of two coarse rate categories (e.g., "fast" and "slow"). Given such categories, the computation is exact and independent of the specific reaction rates.

The analogy for this approach is the design flow for digital electronics, where different designs are systematically explored at a *technology-independent* level, in terms of Boolean functions. Once the best design is found, it is mapped to specific technology libraries in silicon.[4] An overarching goal of the digital paradigm is robustness: digital electronics delivers voltage values that correspond to zero and one reliably, in spite of fluctuations in the signals.

*This work is supported by an NSF EAGER Grant, #CCF-0946601, and by an NSF CAREER Award, #0845650.

In our prior and related work, we have described a variety of computational constructs for chemical reaction networks: logical operations such as copying, comparing and incrementing/decrementing;[5] programming constructs such as "for" and "while" loops;[6] arithmetic operations such as multiplication, exponentiation and logarithms;[5,6] and signal processing operations such as filtering.[7]

In this paper, we describe a scheme for implementing a binary counter with chemical reactions. The value of the counter is encoded by logical values of "0" and "1" that correspond to the absence and presence of specific molecular types, respectively. It is incremented by one every time molecules of a trigger type are injected. Synchronization is achieved with reactions that produce a sustained three-phase oscillation. This oscillation plays a role analogous to a clock signal in digital electronics. Quantities are transferred between molecular types in different phases of the oscillation.

This paper is organized as follows. In Section 2, we summarize the main principles and the basic algorithm for our implementation of the binary counter. In Section 3, we introduce some specific concepts that we use, namely the concepts of "prereactants" and "absence indicators." We also introduce the essential synchronization mechanism that we use, a three-phase oscillation that we call "red-green-blue" (RGB). Then we present the design of the molecular counter. In Section 4, we present simulation results obtained with an ordinary differential equations (ODE) solver. Finally, in Section 5, we discuss DNA strand-displacement reactions as a possible experimental chassis for our method.[8]

2. Counting in Binary

We first review some of the algorithmic principles of counting in binary. Then we present an intuitive description of our approach to implementing a molecular binary counter.

Z	Y	X
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1
⋮	⋮	⋮

Fig. 1. Sequence of values in a three-bit binary counter.

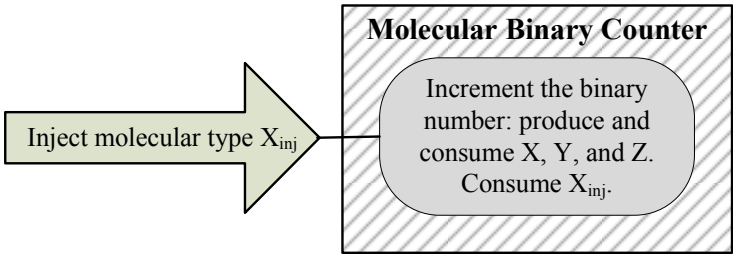

Fig. 2. Basic functionality of the molecular counter.

2.1. *General Principles*

Figure 1 lists the binary numbers that a 3-bit binary counter cycles through, starting at "000" and ending at "111."

(1) *Every time the binary count is incremented, the least significant (i.e., right-most) bit is flipped.* For instance, in the sequence $00\underline{0} \rightarrow 00\underline{1} \rightarrow 01\underline{0} \rightarrow 01\underline{1} \rightarrow 10\underline{0} \rightarrow 10\underline{1}$, note that the least significant bit (underlined) alternates: $0 \rightarrow 1 \rightarrow 0 \rightarrow 1 \rightarrow 0 \rightarrow 1$.

(2) *Every time the binary count is incremented, exactly one bit changes from "0" to "1". (However, several bits may change from "1" to "0.")*
For instance, in the sequence $00\underline{0} \rightarrow 0\underline{0}1 \rightarrow 01\underline{0} \rightarrow \underline{0}11 \rightarrow 10\underline{0} \rightarrow 1\underline{0}1$, the bits that change from "0" to "1" are underlined. Note that there is exactly one such bit each time. (As will be discussed in Section 3, this principle is important for synchronizing our molecular counter.)

(3) *When the binary count is incremented, a given bit changes from "0" to "1" only if all bits of lesser significance (i.e., all bits to the right of it) are "1."*
For instance, in the sequence $000 \rightarrow 00\underline{1} \rightarrow 01\underline{0}$, the second bit changes from "0" to "1" when the first bit is "1." In the sequence $0\underline{11} \rightarrow 1\underline{00} \rightarrow 101$ the third bit changes from "0" to "1" when the first and second bits are "1."

2.2. *Towards a Molecular Binary Counter*

Throughout this paper, the exposition will be in terms of a three-bit binary counter. The ideas can readily be generalized to an n-bit counter. We encode the binary values of "0" and "1" by the presence or absence of specific molecular types, respectively. For the binary sequence in Figure 1, we use the types X, Y and Z. (We will call these "bit types.") For instance, if types X and Z are present, while type Y is absent, the corresponding binary number is "101".

Figure 2 shows the basic functionality of our molecular counter. Every time we want to increment it, we inject some amount of a "trigger" type X_{inj}. The system consumes X_{inj} and increments the binary value specified by the quantities of X, Y and Z. Once all the molecules of X_{inj} have been consumed, the counter can be incremented again.

Tables 2 and 3 specify the set of chemical reactions for our three-bit counter. In order to elucidate the final design, we will provide a succession of design refinements:

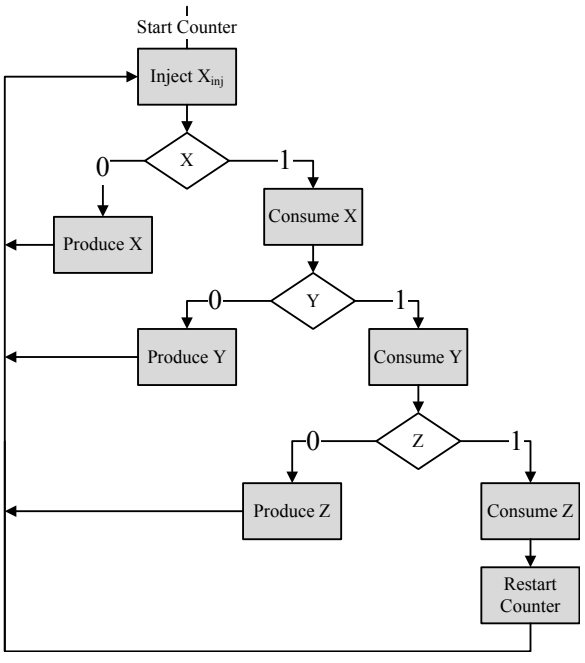

Fig. 3. Basic algorithm for the molecular counter.

(1) We start with a simple intuitive set of reactions, ignoring issues such as synchronization (Section 2.3).

(2) We introduce two specific concepts that we use to implement the counter: the concept of "pre-reactants" and that of "absence indicators" (Section 3.1).
(3) We introduce our synchronization mechanism: a three-phase chemical oscillation that we call "red-green-blue" (RGB) (Section 3.2).
(4) Finally, we provide the full design of the counter, consisting of 24 chemical reactions (Sections 3.3).

2.3. *Intuitive Model*

A molecular counter cannot directly set bits to "0" or to "1"; rather the functionality must be achieved by reactions that produce and consume the molecular types corresponding to these bits. Call the three bits of the counter, the *high*, *middle* and *low* bits, encoded by the presence/absence of types Z, Y and X, respectively. The low bit is set to "1" by producing molecules of X whenever the type X_{inj} is injected into the system:

$$X_{\text{inj}} \rightarrow X. \tag{1}$$

The middle bit is set to "1" by producing molecules of Y whenever the type X is present:

$$X \rightarrow Y. \tag{2}$$

The high bit is set to "1" by producing molecules of Z whenever both types X and Y present:

$$X + Y \rightarrow Z. \tag{3}$$

Note that, in each of these reactions, the system consume molecules of X, Y and Z, reseting the corresponding bits to "0." When molecules of all three types X, Y and Z are present, the corresponding binary number is "111". The counter is reset:

$$X + Y + Z \rightarrow \varnothing. \tag{4}$$

(The symbol $\varnothing$ as a product indicates "nothing", meaning that the type degrades into products that are no longer tracked or used.)

A flowchart for the algorithm that we use is given in Figure 3. In the figure, decisions to produce and consume molecular types are made according to the presence and absence of types. (As we refine the design, we will have to implement these "decisions" through chemical reactions.) Let us assume the current binary number is set to "101". This number corresponds to the absence of Y and the presence of X and Z. Suppose that we inject the trigger type X_{inj}; we move to the first decision box. Since X is present, we do not produce more of it. We consume molecules of X and move to the next decision box. Here we check for the presence of type Y. Since Y is absent, we move to the left and produce molecules of Y. With the absence of X and the presence of Y and Z, the binary number has changed to "110". Next we return to "idle state," waiting for the next injection.

3. Synchronization

The challenge in setting up the molecular counter is that all the chemical reactions fire asynchronously. Each reaction starts producing its products as soon as its reactants are available. If these products participate as reactants in other reactions, then they immediately start getting consumed.

Accordingly, with Reactions 1–4, we will not get a binary counter, encoded by the presence and absence of X, Y and Z. Rather, we will get a jumble of all of these. In particular, note that with Reaction 2, Y is produced from X. As soon as molecules of Y are available, Reaction 3 starts consuming molecules of X and Y to produce molecules of Z. This contradicts the second principle described in Section 2.1: we should only change one bit from "0" to "1" in each increment operation. To mitigate against this issue, we introduce additional molecular types called "prereactants." We also introduce "absence indicator" types to coordinate the transfer between prereactants and reactants.

3.1. *Prereactants and Absence Indicators*

We use the following notation to describe these concepts. For each bit i of the counter,

(1) Q_i is a **bit type** corresponding to i^{th} bit. (For our three-bit molecular counter, we have $Q_1 = X$, $Q_2 = Y$ and $Q_3 = Z$.)
(2) a_{qi} is an **absence indicator** type for type Q_i. (For our three-bit molecular counter, we have $a_{q1} = a_x$, $a_{q2} = a_y$ and $a_{q3} = a_z$.)
(3) Q_{pi} is a **prereactant** type for Q_i. (For our three-bit molecular counter, we have $Q_{p1} = X_p$, $Q_{p2} = Y_p$ and $Q_{p3} = Z_p$.)
(4) We set $X_p = X_{\text{inj}}$: the trigger type is the first prereactant.

All the absence indicators a_{qi} are produced continuously at the slow rate:

$$\varnothing \xrightarrow{\text{slow}} a_{qi}; \tag{5}$$

Here the symbol $\varnothing$ as a reactant indicates that the reaction does not alter the quantity of the reactant types, perhaps because the quantity of these is large or replenishable. If Q_i is present, then its absence indicator a_{qi} is destroyed at the fast rate:

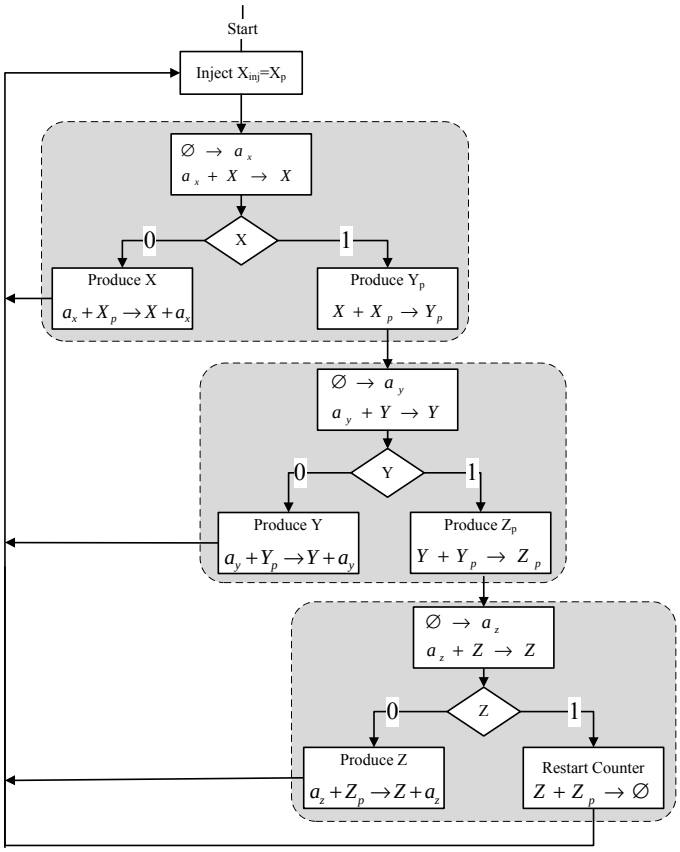

Fig. 4. Modified algorithm for the molecular counter, with prereactants and absence indicators.

$$a_{qi} + Q_i \xrightarrow{\text{fast}} Q_i. \tag{6}$$

However, if Q_i is absent, then a_{qi} persists, so its presence indicates the absence of Q_i, as required. If both a prereactant Q_{pi} and the absence indicator a_{qi} for the i-th bit are present, we produce type Q_i

at the fast rate:

$$a_{qi} + Q_{pi} \xrightarrow{\text{fast}} Q_i + a_{qi}. \tag{7}$$

Finally, the prereactant $Q_{p(i+1)}$ for the $(i+1)$-st bit is produced at the fast rate if both the prereactant Q_{pi} and the type Q_i for the i-th bit are present:

$$Q_i + Q_{pi} \xrightarrow{\text{fast}} Q_{p(i+1)}. \tag{8}$$

Table 1 lists the corresponding reactions for our three-bit counter in terms of the bit types X, Y and Z instead of generic Q_i's. Figure 4 shows a modified version of the flowchart in Figure 3, this time with prereactants and absence indicators.

Table 1. Reactions for the molecular counter, with prereactants and absence indicators.

#	Q_i	Z	Y	X
1	$\varnothing \xrightarrow{\text{slow}} a_{qi}$ $a_{qi} + Q_i \xrightarrow{\text{fast}} Q_i$	$\varnothing \xrightarrow{\text{slow}} a_z$ $a_z + Z \xrightarrow{\text{fast}} Z$	$\varnothing \xrightarrow{\text{slow}} a_y$ $a_y + Y \xrightarrow{\text{fast}} Y$	$\varnothing \xrightarrow{\text{slow}} a_x$ $a_x + X \xrightarrow{\text{fast}} X$
2	$a_{qi} + Q_{pi} \xrightarrow{\text{fast}} Q_i + a_{qi}$	$a_z + Z_p \xrightarrow{\text{fast}} Z + a_z$	$a_y + Y_p \xrightarrow{\text{fast}} Y + a_y$	$a_x + X_p \xrightarrow{\text{fast}} X + a_x$
3	$Q_i + Q_{pi} \xrightarrow{\text{fast}} Q_{p(i+1)}$	$Z + Z_p \xrightarrow{\text{fast}} \varnothing$	$Y + Y_p \xrightarrow{\text{fast}} Z_p$	$X + X_p \xrightarrow{\text{fast}} Y_p$

3.2. *Three-Phase Synchronization*

Including absence indicators and prereactants establishes an order for the transfers of molecular quantities in the counter, but we need a mechanism to ensure that each transfer completes before the next one begins. As indicated on the left-hand side of Figure 5, we must ensure that the accumulation or destruction of the absence indicator completes before the production of the bit type begins; in turn, we must ensure that the production of the bit type completes before the production of the next prereactant begins, and so on. Similarly, as indicated on the right-hand side of Figure 5, we must ensure that the bit types X, Y and Z are not produced simultaneously. We must turn the two "dials" shown in Figure 5 simultaneously. To do so, we introduced a synchronization mechanism based on sustained *chemical oscillation*.

Chemical oscillations, such as those produced by Belousov–Zhabotinsky (BZ) system, have been widely studied by the chemical engineering community.[9] For our purposes, we require an oscillator with a specific property: it must have three symmetric phases for synchronizing both of the "dials" in Figure 5. To this end, we have developed a scheme for chemical oscillation that we call "Red-Green-Blue" (RGB). A detailed analysis of the scheme is given in related work.[7] Here we give only a cursory description of it.

Like the BZ system, our scheme is a perfect oscillator, producing sustained oscillations for a wide range of reaction rates. The scheme is illustrated in Figure 6. Reactions are "color coded" – that is to say assigned to one of the three categories. Quantities are transferred between color categories based

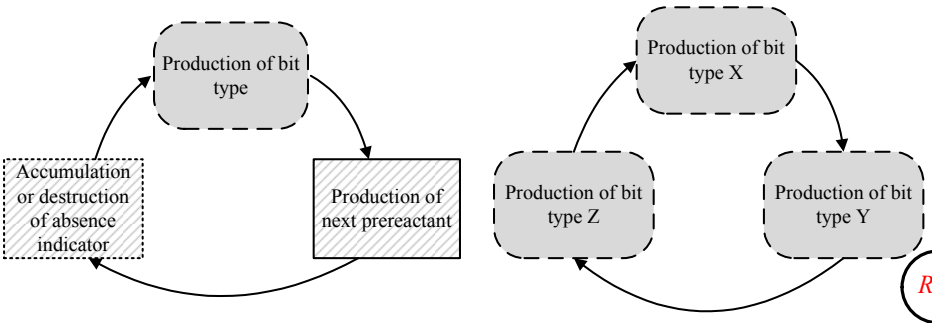

Fig. 5. Sequence of reactions for the molecular counter.

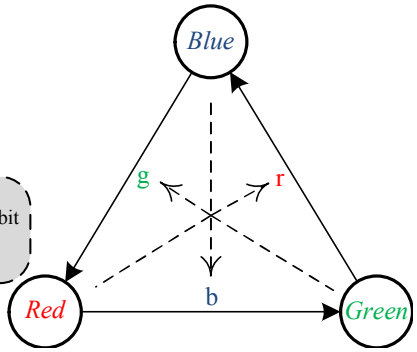

Fig. 6. The three-phase transfer scheme.

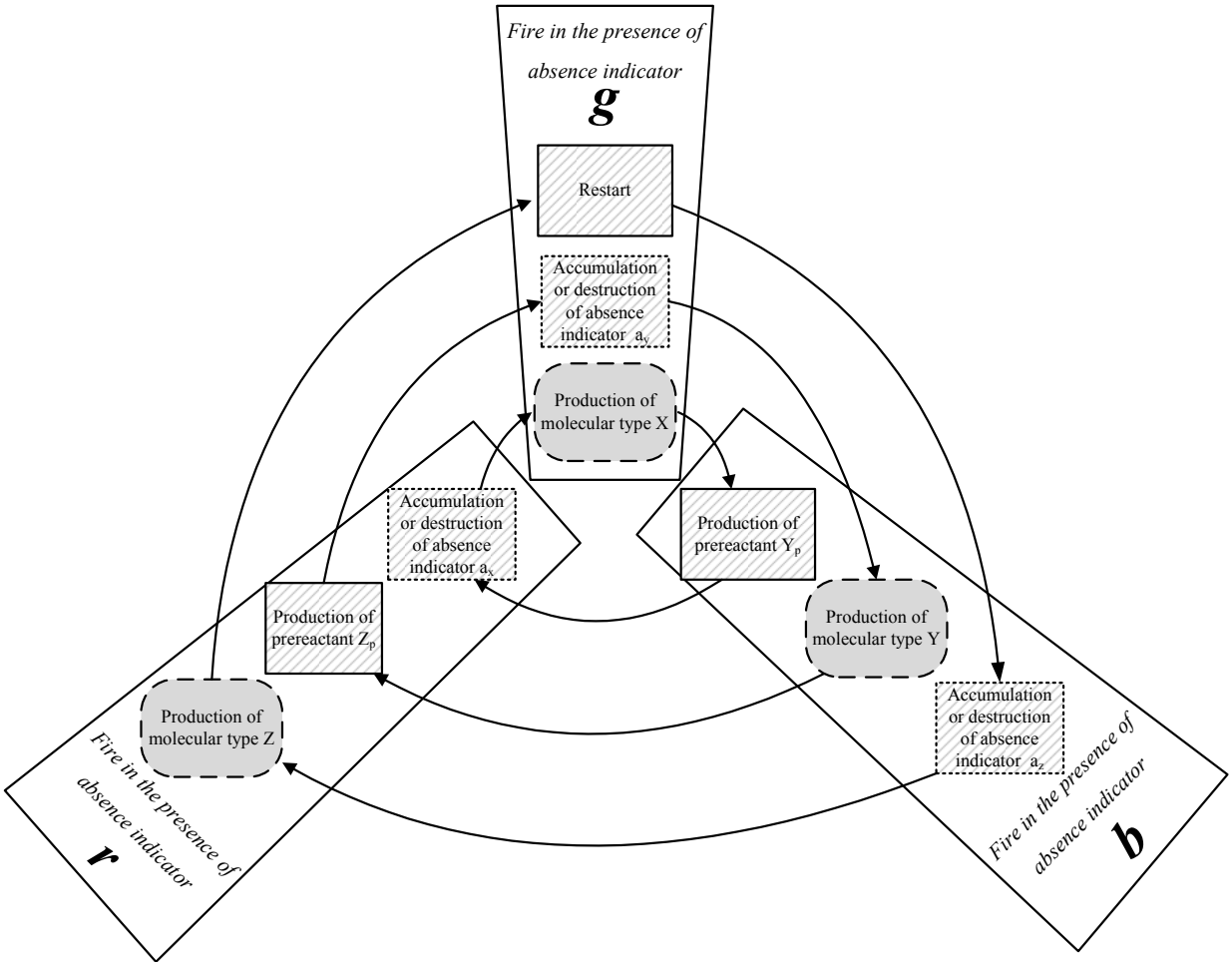

Fig. 7. Combined diagrams for synchronization of reactions.

on the absence of types in the third category: red goes to green in the absence of blue; green goes to blue in the absence of red; and blue goes to red in the absence of green. We introduce molecular types R, G and B. Computation cycles are implemented by transferring quantities among three types R, G and B, with following reactions:

$$\mathbf{b} + R \xrightarrow{\text{slow}} G + \mathbf{b} \quad (9) \qquad \mathbf{r} + G \xrightarrow{\text{slow}} B + \mathbf{r} \quad (10) \qquad \mathbf{g} + B \xrightarrow{\text{slow}} R + \mathbf{g} \quad (11)$$

We generate "absence indicators" types **r**, **g** and **b** corresponding to R, G and B:

$$\begin{aligned} \varnothing &\xrightarrow{\text{slow}} \mathbf{r} \\ R + \mathbf{r} &\xrightarrow{\text{fast}} R \end{aligned} \quad (12) \qquad \begin{aligned} \varnothing &\xrightarrow{\text{slow}} \mathbf{g} \\ G + \mathbf{g} &\xrightarrow{\text{fast}} G \end{aligned} \quad (13) \qquad \begin{aligned} \varnothing &\xrightarrow{\text{slow}} \mathbf{b} \\ B + \mathbf{b} &\xrightarrow{\text{fast}} B \end{aligned} \quad (14)$$

The absence indicators are continually generated. However, they only persist in the absence of the corresponding color-coded signals, since they are quickly consumed by signal molecules in their corresponding color categories. This feature assures that as long as any reaction in a given phase has not fired to completion, the succeeding phase cannot begin. We also include reactions that accelerate

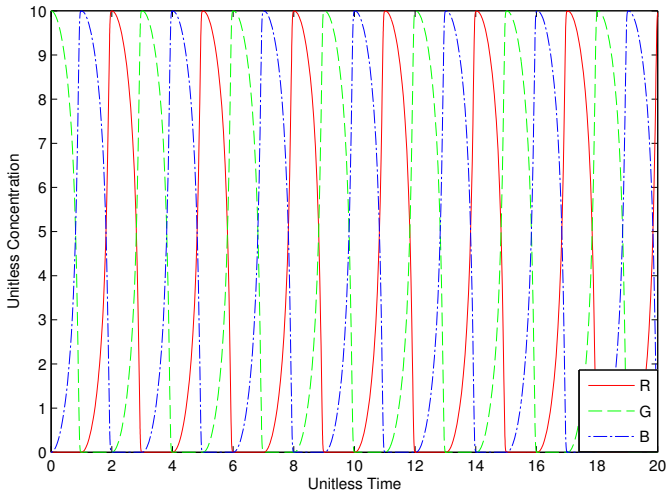

Fig. 8. Simulation results for RGB oscillation.

and isolate the transfers in each phase. For instance, in Reaction 15 two molecules of G combine with one molecule of R to produce three molecules of G. The transfer will occur at a higher rate. Simulation results illustrating the RGB oscillation are shown in Figure 8. In the next section, we incorporate this scheme to synchronize the molecular counter, using RGB in a way analogous to a clock signal in digital electronics.

$$R + 2G \xrightarrow{\text{slow}} 3G \quad (15) \qquad G + 2B \xrightarrow{\text{slow}} 3B \quad (16) \qquad B + 2R \xrightarrow{\text{slow}} 3R \quad (17)$$

3.3. *The Molecular Binary Counter with RGB scheme*

Figure 7 shows the assignment operations to phases of the computation. Absence indicators **r**, **g** and **b** are used to initiate reactions in each phase. In lieu of the generic transfer reactions 9– 11, we use transfer reactions that produce the absence indicators a_x, a_y and a_z for X, Y and Z, respectively:

$$\mathbf{r} + G \xrightarrow{\text{slow}} B + a_x + \mathbf{r} \quad (18) \qquad \mathbf{g} + B \xrightarrow{\text{slow}} R + a_y + \mathbf{g} \quad (19) \qquad \mathbf{b} + R \xrightarrow{\text{slow}} G + a_z + \mathbf{b} \quad (20)$$

This obviates the need for reactions of the form of Reaction 5 to generate a_x, a_y and a_z.

A set of reactions for the counter that incorporates the RGB transfer reactions is described in Figure 9. This is nearly the final design. However, we need a few more reactions to deal with accumulation of unused absence indicators. The transfer reactions 18–20 supply absence indicators a_x, a_y and a_z in every RGB cycle. The scheme cycles continuously, irrespective of injections of X_{inj}. Accordingly, unused absence indicators a_x, a_y and a_z will accumulate. To mitigate against this, we include "**clean-up**" reactions initiated in the presence of corresponding absence indicators **r**, **g** and **b**:

$$\mathbf{b} + a_x \xrightarrow{\text{fast}} \mathbf{b} \quad (21) \qquad \mathbf{r} + a_y \xrightarrow{\text{fast}} \mathbf{r} \quad (22) \qquad \mathbf{g} + a_z \xrightarrow{\text{fast}} \mathbf{g} \quad (23)$$

As shown in Figure 9 the corresponding clean-up reactions always complete before the production of a_x, a_y and a_z begins. For instance, the absence indicator a_z is pushed into the system by Reaction 20 whenever the absence indicator **b** is present. Therefore, the clean-up Reaction 23 for a_z fires in the preceding RGB phase that was initiated in the presence of absence indicator **g**.

Table 2 shows the final set of RGB reactions and Table 3 shows the final set of reactions for X, Y and Z. Together, these comprise our complete design of the molecular counter.

Table 2. Final version of RGB reactions for the molecular counter.

Production of r, g, b	Destruction of r, g, b	Transfer reactions	Speed-up reactions	Clean-up reactions
$\varnothing \xrightarrow{\text{slow}} \mathbf{r}$	$R + \mathbf{r} \xrightarrow{\text{fast}} R$	$\mathbf{b} + R \xrightarrow{\text{slow}} G + a_z + \mathbf{b}$	$R + 2G \xrightarrow{\text{slow}} 3G$	$\mathbf{b} + a_x \xrightarrow{\text{fast}} \mathbf{b}$
$\varnothing \xrightarrow{\text{slow}} \mathbf{g}$	$G + \mathbf{g} \xrightarrow{\text{fast}} G$	$\mathbf{r} + G \xrightarrow{\text{slow}} B + a_x + \mathbf{r}$	$G + 2B \xrightarrow{\text{slow}} 3B$	$\mathbf{r} + a_y \xrightarrow{\text{fast}} \mathbf{r}$
$\varnothing \xrightarrow{\text{slow}} \mathbf{b}$	$B + \mathbf{b} \xrightarrow{\text{fast}} B$	$\mathbf{g} + B \xrightarrow{\text{slow}} R + a_y + \mathbf{g}$	$B + 2R \xrightarrow{\text{slow}} 3R$	$\mathbf{g} + a_z \xrightarrow{\text{fast}} \mathbf{g}$

Table 3. Final version of reactions for molecular types X, Y and Z.

Accumulation or destruction of absence indicators	Production of molecules	Production of prereactant
$\mathbf{r} + a_x + X \xrightarrow{\text{fast}} X + \mathbf{r}$	$\mathbf{g} + a_x + X_p \xrightarrow{\text{fast}} X + a_x + \mathbf{g}$	$\mathbf{b} + X + X_p \xrightarrow{\text{fast}} Y_p + \mathbf{b}$
$\mathbf{g} + a_y + Y \xrightarrow{\text{fast}} Y + \mathbf{g}$	$\mathbf{b} + a_y + Y_p \xrightarrow{\text{fast}} Y + a_y + \mathbf{b}$	$\mathbf{r} + Y + Y_p \xrightarrow{\text{fast}} Z_p + \mathbf{r}$
$\mathbf{b} + a_z + Z \xrightarrow{\text{fast}} Z + \mathbf{b}$	$\mathbf{r} + a_z + Z_p \xrightarrow{\text{fast}} Z + a_z + \mathbf{r}$	$\mathbf{g} + Z + Z_p \xrightarrow{\text{fast}} \mathbf{g}$

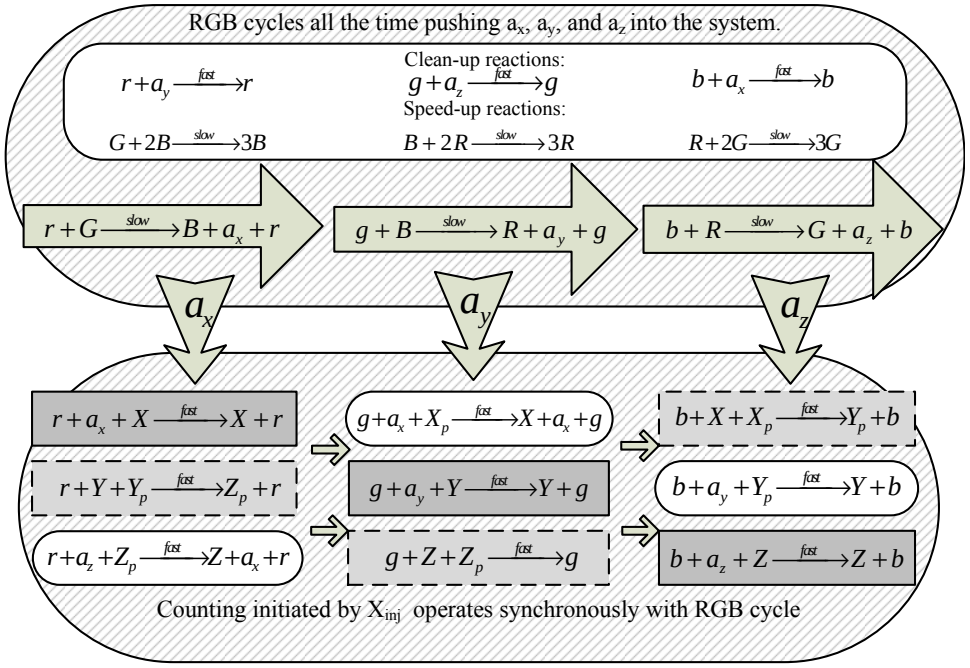

Fig. 9. Diagram for the molecular counter with RGB synchronization.

4. Simulation results

As we discuss in Section 5, we are targeting DNA strand displacement as a potential experimental chassis for our molecular counter.[8] Accordingly, the constituent chemical reactions must all be either uni- or bimolecular reactions. Thus, we split all trimoleclar reactions of the form

$$R_1 + R_2 + R_3 \xrightarrow{\mathrm{k}_1} R_4 + \cdots \tag{24}$$

in Table 2 and Table 3 into the sequence of bimolecular reactions

$$\begin{aligned} R_1 + R_2 &\underset{\mathrm{k}_2}{\overset{\mathrm{k}_1}{\rightleftharpoons}} I \\ I + R_3 &\xrightarrow{\mathrm{k}_2} R_4 + \cdots . \end{aligned} \tag{25}$$

The first step in this process is reversible: two molecules R_1 and R_2 can combine at a rate k_1, but in the absence of any molecules R_3, the combined form will dissociate back into molecules R_1 and R_2 at a rate k_2 which is greater than k_1. In the presence of R_3, the sequence of reactions will proceed, producing $R_4 + \cdots$. The overall rate of reactions is determined by the slowest reaction and therefore set by k_1.

With such transformations into uni- and bimolecular reactions, we simulate the chemical kinetics of our molecular counter with an ordinary differential equation (ODE) solver. We chose the parameters, the concentration values and reaction rates as follows. The concentrations are unitless; for an experimental setup, these would be scaled appropriately. The initial concentration of our trigger type X_p was set to 0.05. (Recall that we use X_p as the trigger type, so $X_{\text{inj}} = X_p$.) The initial concentration

of G was set to a value much greater than that of X_p, namely 10. The rates of all the "slow" reactions were set to unity. The rates of all the "fast" reactions were set five orders of magnitude higher.

Figure 10 shows the change in concentration X, Y and Z for 20 injections. We observed a stable behavior of the molecular binary counter for 40 injections. The data from the simulation for the first

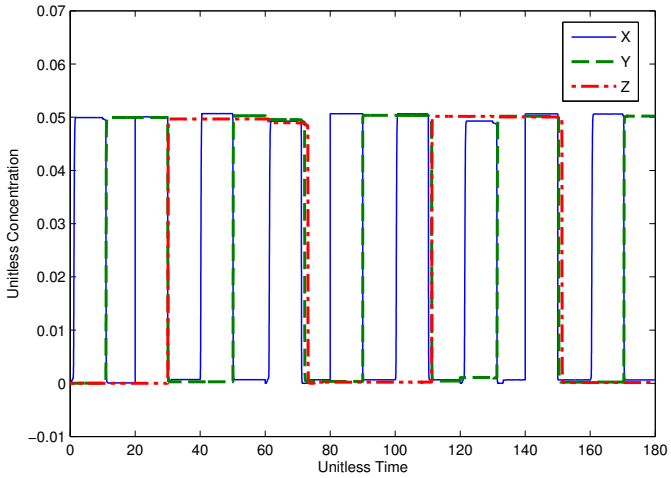

Fig. 10. Simulation results for molecular types X, Y and Z for 20 injections.

8 injections is shown in Table 4. We see exactly the behavior that we expect for a binary counter.

- The threshold for logical "1" for the bit types X, Y and Z can be set at 97% of the injected concentration of X_p.
- The threshold for logical "0" for the bit types X, Y and Z can be set at 3% of the injected concentration of X_p.

Table 4. Data from the ODE simulation of the molecular counter for 8 successive increment operations.

# Injection	Binary number	Concentration Z	Concentration Y	Concentration X
0	000	0.0000	0.0000	0.0000
1	00**1**	0.0000	0.0000	**0.0499**
2	0**1**0	0.0000	**0.0500**	0.0001
3	0**11**	0.0000	**0.0500**	**0.0501**
4	**1**00	**0.0497**	0.0003	0.0007
5	**101**	**0.0497**	0.0003	**0.0507**
6	**11**0	**0.0497**	**0.0503**	0.0007
7	**111**	**0.0490**	**0.0496**	**0.0493**
8	000	0.0002	0.0004	0.0007

5. Discussion

We have demonstrated the design of a molecular counter that is robust and accurate. Given only rate categories of "slow" and "fast, our counter computes exact binary values. It does not matter how fast any "fast" reaction is relative to another, or how slow any "slow" reaction is relative to another – only that "fast" reactions are fast relative to "slow" reactions. Throughout the paper, the exposition was in terms of a three-bit counter. In future work, we will generalize the construction to n bits.

Our contribution is to tackle the problem of synthesizing computation at a conceptual level, working not with actual molecular types but rather with abstract types. In future work, we will demonstrate our binary counter through *in vitro* experiments with DNA. It has been shown that DNA strand displacement reactions can emulate the chemical kinetics of nearly any chemical reaction network. Indeed, in recent work, researchers at Caltech have developed a compiler that translates abstract chemical reactions of the sort that we design into specific DNA reactions.[8]

Recent work has demonstrated both the scale of computation that is possible with DNA-based computing,[10] as well as exciting applications.[11] We comment that our design of a molecular counter could be applied for the task of counting cell divisions. This task is important for the analysis of aging and, perhaps, for the detection of cancer, where cell divisions run rampant. Also, our design might find applications in biochemical sensing and drug delivery.

References

1. D. Endy, "Foundations for engineering biology," *Nature*, vol. 438, pp. 449–453, 2005.
2. F. Horn and R. Jackson, "General mass action kinetics," *Archive for Rational Mechanics and Analysis*, vol. 47, pp. 81–116, 1972.
3. P. Érdi and J. Tóth, *Mathematical Models of Chemical Reactions: Theory and Applications of Deterministic and Stochastic Models*. Manchester University Press, 1989.
4. L. Lavagno, G. Martin, and L. Scheffer, *Electronic Design Automation for Integrated Circuits Handbook*. CRC Press, 2006.
5. P. Senum and M. D. Riedel, "Rate-independent biochemical computational modules," in *Proceedings of the Pacific Symposium on Biocomputing*, 2011.
6. A. Shea, B. Fett, M. D. Riedel, and K. Parhi, "Writing and compiling code into biochemistry," in *Proceedings of the Pacific Symposium on Biocomputing*, 2010, pp. 456–464.
7. H. Jiang, A. P. Kharam, M. D. Riedel, and K. K. Parhi, "A synthesis flow for digital signal processing with biomolecular reactions," in *IEEE International Conference on Computer-Aided Design*, 2010.
8. D. Soloveichik, G. Seelig, and E. Winfree, "DNA as a universal substrate for chemical kinetics," *Proceedings of the National Academy of Sciences*, vol. 107, no. 12, pp. 5393–5398, 2010.
9. I. R. Epstein and J. A. Pojman, *An Introduction to Nonlinear Chemical Dynamics: Oscillations, Waves, Patterns, and Chaos*. Oxford Univ Press, 1998.
10. L. Qian and E. Winfree, "A simple DNA gate motif for synthesizing large-scale circuits," in *DNA Computing*, 2009, pp. 70–89.
11. S. Venkataramana, R. M. Dirks, C. T. Ueda, and N. A. Pierce, "Selective cell death mediated by small conditional RNAs," *Proceedings of the National Academy of Sciences*, 2010 (in press).