Asynchronous Discrete-Time Signal Processing with Molecular Reactions

Sayed Ahmad Salehi, Marc D. Riedel, Keshab K. Parhi
Dept. of Electrical and Computer Engineering
University of Minnesota
200 Union Street SE
Minneapolis, MN 55455 USA

***Abstract*-This paper presents a new methodology to synthesize molecular reactions for discrete-time signal processing (DSP) computations that produce time-varying quantities of molecules as a function of time-varying input quantities. DSP structures include delay elements which need to be synchronized by a clock signal. This paper demonstrates an approach to synthesize molecular reactions to implement DSP operations without requiring a clock signal. In the proposed approach, each delay and output variables are mapped to two types of molecules. The scheduling of the reactions is controlled by absence indicators, i.e., signals transfer according to the absence of other signals. All computations are scheduled in four phases. The input signal and values stored in all delay elements are consumed for computations in the first phase. Results of computations are stored in the first types of molecule corresponding to the delay elements and output variables. During the second phase, the value of the first molecular type is transferred to the second molecular type for the output variable. In the third phase, the values of first types of molecule are transferred to the second types of molecule associated with each delay element. The output molecules are collected in the fourth phase. The method is illustrated by synthesizing a simple FIR filter, an IIR filter, and an 8-point real-valued fast Fourier transform (FFT). The synthesized systems consist of bimolecular reactions and are translated to DNA-strand displacement reactions. The methodology is validated through mass-action simulations of DNA kinetics. The proposed approach may play a potential role in applications such as drug delivery and in monitoring spectral content of proteins.**

I. Introduction

Like a voltage value in electronic circuits, a chemical concentration in a molecular system determines a signal value. The last few decades have seen significant progress in the implementation of DSP algorithms electronically. Such algorithms are also applicable to the field of molecular computing. Applications include spectral analysis of protein concentrations for targeted drug delivery [1]. The same signal processing algorithms that estimate channels in wired and wireless systems can be used to estimate models for cancer growth [2]. These estimations can be carried out either in the time domain or in the frequency domain. In the field of metabolic engineering, applications of molecular DSP include activating or inhibiting pathways based upon the fluctuation of protein concentration [3].

Prior work on molecular DSP has demonstrated methods for implementing algorithms using *synchronous* and *RGB* schemes [4,5]. In both schemes the arithmetic operations such as addition and multiplication are implemented by the same set of reactions. The difference between these schemes lies in the implementation of delay elements. Synchronous scheme has a global two-phase clock and each delay element in this scheme consists of two molecular types, D_i and D'_i [5]. In the first phase of the clock, the computation results are stored in D_i and in the second phase D_i transfers to D'_i. The *RGB* scheme does not require a global clock; however, to prevent conflicts in signal transfers, each delay element is implemented by three molecular types: so-called Red (R_i), Green (G_i), and Blue (B_i) [4]. An asynchronous scheme that doesn't require clock has been presented in [6]. In this design methodology, after the signal of a current node transfers to the next connected node(s), the signal of the previous connected node transfers to the current node. This method is applied to all the nodes starting with the output node tracing backwards until the input node. The next step involves detecting signal transfer conflicts and resolving them by adding some intermediate nodes. Our proposed systematic method, however, provides a conflict-free scheduling for the system.

We present a new asynchronous scheme in which each delay element is composed of two molecular types. We illustrate our methodology with the design of an FIR filter, a first-order IIR filter, and FFT computation of real-valued signals (RFFT).

In recent years, significant progress in molecular computing has demonstrated the feasibility of implementing molecular reactions using DNA-strand displacement cascades [7]-[10]. The translation to DNA-strands is feasible as long as the reactions are unimolecular or bimolecular [9]. Thus, we restrict all the reactions to bimolecular reactions so that they can be translated to DNA-strand displacement reactions.

This paper is organized as follows. Section 2 provides a brief background on molecular systems, molecular computing and absence indicators. The proposed methodology is described in Section 3. In Section 4 simulation results are presented. A comparison to prior work for RFFT implementation is presented in Section 5. Finally Section 6 concludes the paper.

This work was supported in part by the US National Science Foundation under grants CCF–14234707 and CCF-1117168.

II. BACKGROUND

A. Molecular Reaction System

A molecular system consists of a set of chemical reactions, where reactants combine to form products. For example (1) shows a molecular reaction, where *x* and *y* are reactants, z is a product, and *k* is the rate constant. Consider the reaction:

$$x + y \xrightarrow{k} 3z. \tag{1}$$

In this reaction, one molecule of *x* combines with one molecule of *y* to produce three molecules of *z*. If [.] denotes concentration, the dynamic behavior of input and output concentrations is modeled by (2) [11,12].

$$-\frac{d[x]}{dt} = -\frac{d[y]}{dt} = 3\frac{d[z]}{dt} = k[x][y]. \tag{2}$$

The rate of reaction is proportional to the concentration of inputs and the rate constant. Unfortunately *k* can vary due to factors such as cell volume and temperature. To have a robust construct we use the idea of coarse rate categories ('fast' and 'slow' rates), proposed in [5]. Given such categories, the computations are independent of exact values of *k*. Here only fast rate constants, k_f, must be fast relative to slow ones, k_s. Therefore, we assume when a fast reaction and a slow one can be fired simultaneously; the fast reaction is fired to completion and doesn't allow the slow reaction to be fired.

B. Molecular Computing

Basic arithmetic operations such as addition and multiplication can be implemented by molecular reactions.

Addition can be implemented by choosing several reactions with the same product. For example

$$y = x_1 + x_2$$

is implemented by the following reactions:

$$\begin{aligned} x_1 &\xrightarrow{k_f} y \\ x_2 &\xrightarrow{k_f} y. \end{aligned} \tag{3}$$

When both reactions are completed, the concentration of y will be the sum of the former concentration of x_1 and the former concentration of x_2.

Multiplication in the form of

$$y = \frac{c_2}{c_1}x$$

is implemented by reaction

$$c_1\, x \xrightarrow{k_f} c_2\, y, \tag{4}$$

where c_1 and c_2 are constants. Every time this reaction fires, c_1 molecules of x transfer to c_2 molecules of y. Once the reaction has fired to completion the requisite operation of multiplication is complete. Our system incurs the limitation of having only bimolecular reactions, that is to say, reactions with exactly two reactant molecules. Therefore, we can only implement

$$y = \frac{c_2}{2}x$$

by each reaction. Multiplications with other coefficients can be approximated first and then implemented by cascading these reactions. For example

$$y = \frac{\sqrt{2}}{2}x \approx \left(\frac{1}{2} + \frac{1}{8} + \frac{1}{16}\right)x \tag{5}$$

is implemented by the following reactions

$$\begin{aligned} 2x &\xrightarrow{k_f} x_2 + y \\ 2x_2 &\xrightarrow{k_f} x_4 \\ 2x_4 &\xrightarrow{k_f} x_8 + y \\ 2x_8 &\xrightarrow{k_f} y. \end{aligned} \tag{6}$$

C. Absence Indicators

Consider that we have a group of molecular types $s_1, s_2, \dots, s_n$. The "absence indicator" for this group is molecular type a_s such that when molecules of any type in this group are present, concentration of a_s is nearly zero; otherwise, when concentrations of s_i $(i = 1, \dots, n)$ are all zero, concentration of a_s is nonzero. This is satisfied by the reactions below [4]:

$$\begin{aligned} src &\xrightarrow{k_s} a_s \\ s_1 + a_s &\xrightarrow{k_f} s_1 \\ s_2 + a_s &\xrightarrow{k_f} s_2 \\ &\vdots \\ s_n + a_s &\xrightarrow{k_f} s_n \end{aligned} \tag{7}$$

Reactant src in the first reaction of (7) denotes a large and replenishable source so that its concentration doesn't change significantly. The first reaction slowly but constantly generates molecules of a_s. However, in the following reactions, $s_1, s_2, \dots, s_n$ quickly consume a_s, but keep their own concentrations. Therefore, molecules of a_s accumulate only when all types in $s_1, s_2, \dots, s_n$ are absent. In the proposed asynchronous methodology, absence indication is used to control signal transfers, as discussed in the next section.

III. PROPOSED METHODOLOGY

We present a new approach for designing and implementing discrete-time signal processing algorithms with molecular reactions. In the new framework, each delay element of the structure is assigned two molecular types, D_i and D'_i. Transferring signals among delay elements is implemented by transferring concentrations between molecular types assigned to delay elements. The entire computation is completed in four phases. Signal transfers in each phase are triggered by the absence indicators of the other phases. In the proposed scheme, two types of transfer are not allowed. These restrictions are illustrated in Fig. 1. First, all outgoing edges of a node must be scheduled in the same phase. Fig. 1(a) illustrates a violation of this constraint. Second, if outgoing edges of a node are scheduled at phase "i", none of its incoming edges can be scheduled at phase "$i + 1$". Figure 1(b) illustrates a violation of this constraint. A synthesis approach for mapping any DSP algorithm to molecular reactions is described below:

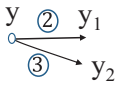
1- Draw the data flow graph (DFG) according to the block diagram of the DSP algorithm. Replace the output node y by nodes y and y', and each delay element D_k by a pair of nodes D_k and D'_k.

2- Assign phase 1 to the outgoing edges of the input node and the outgoing edges of each D'_k node.

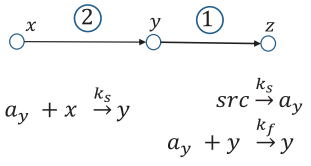
3- Assign phase 2 to the fan out edge of output node (y).

4- All edges between D_k and D'_k are scheduled to phase 3.

5- The outgoing edge of y' is scheduled to phase 4.



(a)



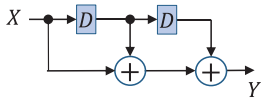
(b)

Figure 1. Two types of signal transfer not allowed in our molecular scheme: (a) Outgoing edges scheduled in different times (b) Incoming edge with assigned phase $\boldsymbol{i+1}$ for a node with outgoing edge to assigned to phase $\boldsymbol{i}$.

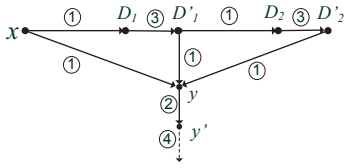
6- The molecular reactions for absence indicators, computations, and signal transfers are synthesized according to the assigned scheduling phases.

The proposed *4-phase* method is now illustrated by three DSP operations: an FIR filter, a first-order IIR filter and a FFT computation for real-valued signals.

a. FIR filter: Figure 2(a) shows a three-tap FIR filter. For simplicity, all tap coefficients are assumed to be 1. The flow



(a)



(b)

Figure 2. A three-tap FIR filter: (a) Block diagram, (b) Data flow graph and scheduling based on the proposed method.

graph in Figure 2(b) illustrates the phase assignments. The molecular reactions for the flow graph are described by (8).

$$src \xrightarrow{k_s} a_1 + a_2 + a_3 + a_4$$

Phase 1:
$$a_1 + x \xrightarrow{k_f} x$$
$$a_1 + D'_1 \xrightarrow{k_f} D'_1$$
$$a_1 + D'_2 \xrightarrow{k_f} D'_2$$

Phase 2:
$$a_2 + y \xrightarrow{k_f} y \qquad (8)$$

Phase 3:
$$a_3 + D_1 \xrightarrow{k_f} D_1$$
$$a_3 + D_2 \xrightarrow{k_f} D_2$$

Phase 4:
$$a_4 + y' \xrightarrow{k_f} y'$$

Then reactions (9) provide the signal transfers associated with related absence indicators. Signal transfers of each phase are enabled by the absence indicator of the previous phase. Note that these are all slow reactions.

Phase 1:
$$x + a_4 \xrightarrow{k_s} D_1 + y$$
$$D'_1 + a_4 \xrightarrow{k_s} D_2 + y$$
$$D'_2 + a_4 \xrightarrow{k_s} y$$

Phase 2:
$$y + a_1 \xrightarrow{k_s} y' \qquad (9)$$

Phase 3:
$$D_1 + a_2 \xrightarrow{k_s} D'_1$$
$$D_2 + a_2 \xrightarrow{k_s} D'_2$$

According to reactions (8) and (9), molecules of x, D'_1, and D'_2 transfer in the first phase. After all molecules of x, D'_1, and D'_2 are transferred, phase 2 starts and y is transferred to y'. In phase 3, $D_1$and D_2 transfer, respectively, to D'_1 and D'_2 after all molecules of y transfer to y'. Concentration of $D'_1$and D'_2 are stored to be used for the computation of the next output. Thus, each pair of D_i and D'_i $(i = 1,2)$ functions as a delay element.

One should notice that the final output y' is collected whenever the absence indicator of the third phase, a_3, is nonzero, implying the third phase has been completed. While the new input is also injected at the same time, it is not used by the system until all molecules of y' are collected.

b. IIR Filter: As another simple DSP operation, we describe the 4-phase method for a simple first-order IIR filter. The block diagram of this filter is shown in Figure 3(a). The filter contains a multiplication by 0.5 inside a feedback loop. From steps 1 to 5 of the synthesis flow, the scheduled 4-phase flow graph for the filter is obtained as shown in Figure 3(b). The set of required absence indicator reactions are illustrated in (10).

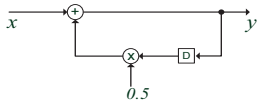
$$src \xrightarrow{k_s} a_1 + a_2 + a_3 + a_4$$

Phase 1:
$$a_1 + x \xrightarrow{k_f} x$$
$$a_1 + D'_1 \xrightarrow{k_f} D'_1$$

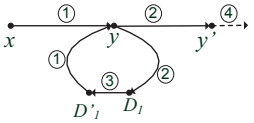
Phase 2:
$$a_2 + y \xrightarrow{k_f} y \qquad (10)$$

Phase 3:
$$a_3 + D_1 \xrightarrow{k_f} D_1$$

Phase 4:
$$a_4 + y' \xrightarrow{k_f} y'$$



(a)



(b)

Figure 3. An IIR filter: (a) Block diagram, (b) Data flow graph and scheduling for molecular implementation.

Signal transfers and computations are implemented by the reactions in (11).

$$\begin{aligned} &\textit{Phase 1:} && x + a_4 \xrightarrow{k_s} y \\ & && D'_1 + a_4 \xrightarrow{k_s} Ty \\ & && 2Ty \xrightarrow{k_f} y \\ &\textit{Phase 2:} && y + a_1 \xrightarrow{k_s} y' + D_1 \\ &\textit{Phase 3:} && D_1 + a_2 \xrightarrow{k_s} D'_1 \end{aligned} \quad (11)$$

Note that the third reaction in (11), related to the multiplication by 0.5, fires to completion much faster than the transfer reactions. Thus, each two Ty molecules are immediately combined to produce one y molecule. In other words, D'_1 is transferred to temporary molecules Ty and in the same phase, Ty is multiplied by 0.5 to produce y.

The presented method can be easily generalized for DSP algorithms with more than one input/output. The following example illustrates such an algorithm with four inputs and four outputs.

c. *Real-valued FFT (RFFT):* Discrete Fourier transform (DFT) computes the spectral contents of a signal at various frequencies. Fast Fourier transform (FFT) computes DFT using a fast approach when the number of required multiplications can be reduced from $O(N^2)$ to $O(Nlog_2N)$ [13]. We implement FFT, as a canonical algorithm in DSP, with molecular reactions. Molecular implementation of FFT can be used to monitor the frequency content of a protein over time in applications such as drug delivery or cell growth modeling. Like all of the physical signals the concentration of input molecules is a real-valued signal. Therefore, we consider implementation of an FFT system with real-valued inputs, called RFFT. Figure 4(a) shows the block diagram for a 4-parallel 8-point RFFT. 8 samples of the input signal, $x(n)$, arrive in two stages. In the first stage, $x(0)$ to $x(3)$ arrive while multiplexers choose their select input $s1$. In the second stage $x(4)$ to $x(7)$ arrive and multiplexers select input $s2$. All of the internal datapaths For an RFFT structure can be real-valued (not complex-valued) datapaths [14]. For more information about RFFT the reader is referred to [15].

The proposed synthesis method assigns scheduling of phases to the flow graph as shown in Figure 4(b). Multiplexers in Figure 4(a) are implemented as shown in Figure 5. As figure 5 shows when $s1$ is nonzero x transfers to z and when $s2$ is nonzero y transfers to z. So signal $s1$ and $s2$ are control signals for multiplexers. In the first stage $s1$ is nonzero while $s2$ is zero. At the end of each stage $s1$ and $s2$ toggle to be ready for the next stage. For this purpose, in the second phase $s1$ transfers to $s2'$ and $s2$ transfers to $s1'$ simultaneously. Then in the fourth phase, $s2'$ transfers to $s2$ and $s1'$ transfers to $s1$. The circular flow graph at the bottom of Figure 4(b) represents the toggling of $s1$ and $s2$. This flow graph is implemented by the reactions in (12).

$$\begin{aligned} s1 + a_1 &\xrightarrow{k_s} s2' \\ s2' + a_3 &\xrightarrow{k_s} s2 \\ s2 + a_1 &\xrightarrow{k_s} s1' \\ s1' + a_3 &\xrightarrow{k_s} s1 \end{aligned} \quad (12)$$

Similar to FIR and IIR filters, the reactions related to the computation, signal transfer, and absence indicators for each phase can be synthesized from the flow graph in Figure 4(b). As described in Section 2.2, multiplication by $\frac{\sqrt{2}}{2}$ is implemented using $(\frac{1}{2}+\frac{1}{8}+\frac{1}{16})$ approximation.

In general a signal value can be negative, while concentration of a molecular type can't be negative. Therefore, we use one molecular type for positive and another one for negative part of each signal. We perform computations and signal transfers for each part independently. Finally these two parts cancel out each other and the one with larger concentration determines the sign

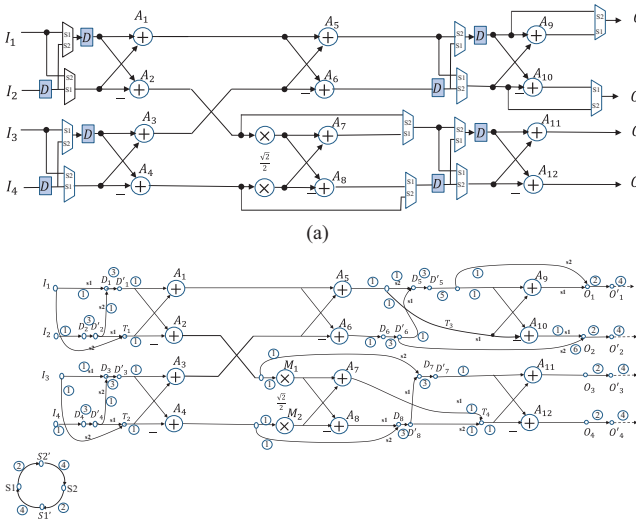


Figure 4. 4-parallel 8-point RFFT: (a)Block diagram, (b)Data flow graph and scheduling obtained by the proposed method.

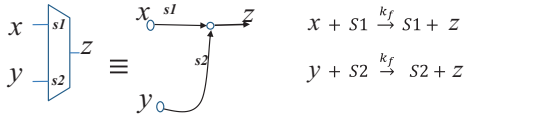
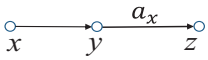


Figure 5. Implementation of multiplexer by molecular reactions.



$$a_x + y \xrightarrow{k_s} z$$
$$z + y \xrightarrow{k_s} 2z$$

Figure 6. Speeding up signal transfers by positive feedback.

and value of the signal [5]. For example x_p and x_n represent positive and negative part of signal x, and (13) describes positive-negative cancellation reaction by transferring equal concentrations of positive and negative parts to an external sink.

$$x_p + x_n \xrightarrow{k_f} sink \quad (13)$$

In order to improve the accuracy and speed of the implemented molecular systems, we add three sets of reactions to them. We call these reactions: *threshold*, *negative feedback*, and *positive feedback* reactions.

Threshold reactions: As described in Section 2.3 when a type of molecule exists in the system, its absence indicator is nearly zero but not exactly zero. Although they are very slow a small nonzero value of absence indicator can fire the next phase reactions before completion of the current phase reactions. To avoid this, we initially inject a small concentration of so-called threshold molecules, T_x. The first reaction in (14) is a fast reaction. Thus, the absence indicator molecules, a_x, can't fire slow reactions before consuming T_x. In other words, the concentration of a_x must be more than the concentration of T_x, in order to fire signal transfer reactions. When x is present, the second reaction in (14) replenishes the threshold molecules T_x.

$$a_x + T_x \xrightarrow{k_f} sink$$
$$x + a_x \xrightarrow{k_f} x + T_x \qquad (14)$$

Negative feedback reactions: The absence indicator molecules, a_x, are produced constantly from the src. If x doesn't exist for a while, the concentration of a_x becomes larger and larger. Then when x is produced it takes more time to consume all molecules of a_x. The first reaction in (15) limits the increase of a_x concentration. The second reaction in (15) controls T_x in the same manner.

$$2a_x \xrightarrow{k_s} a_x + T_x$$
$$2T_x \xrightarrow{k_s} a_x + T_x \qquad (15)$$

Positive feedback reaction: As shown in Figure 6, in positive feedback reactions, destination of a signal transfer, z, is used to speed up the signal transfer y to z. In other words, when the first reaction in Figure 6 starts, second reaction speeds up its completion.

IV. Simulation Results

The molecular reactions are mapped to DNA-strand displacement reactions. Critical for mapping to DNA strands, all of our reactions are bimolecular reactions [9]. We simulated the kinetic of reactions in our designs exploring the mechanism and software tools for DNA-strand displacement developed by Winfree's group at Caltech.

For all DNA simulations for the presented designs we used the following parameters: The initial concentration of auxiliary complexes, $C_{max} = 10^{-5}M$, the maximum strand displacement rate constant, $q_{max} = 10^{6}M^{-1}s^{-1}$, $k_s = 5.56 \times 10^{4}M^{-1}s^{-1}$ and $k_f = 10 \times k_s$. The initial concentration for the source molecular type, src, is set to $0.2\ nM$.

The simulation results for the FIR and IIR filters are shown in Figure 7 and Figure 8, respectively. The input is a time-varying signal x with both high frequency and low frequency components. The output is a time-varying concentration y'. For the FIR filter, molecules of x are injected into the system and molecules y' are collected from the system every 20 hours. For IIR filter the injection/collection time is every 30 hours. The Figures show the theoretical outputs as well as simulated outputs. The simulated outputs track the theoretical outputs with some errors. The errors come mainly from the leakage among molecular types. Although it has one more delay element and more number of signal transfers, the average relative error for the FIR filter is less than the IIR filter. Generally speaking, IIR filters have higher errors than FIR filters since feedback in such filters leads to error accumulation. Therefore, we considered longer interval between output collections for IIR filter in order to improve its output accuracy.

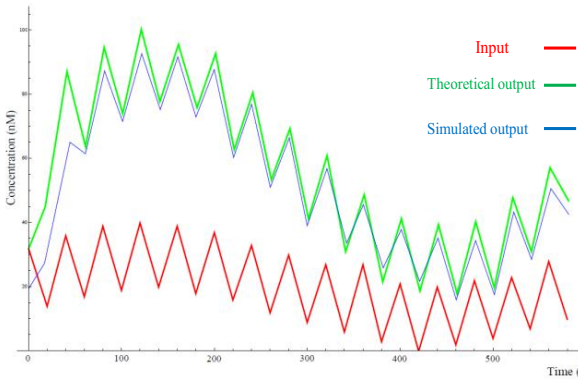


Figure 7. Simulation results for FIR filter.

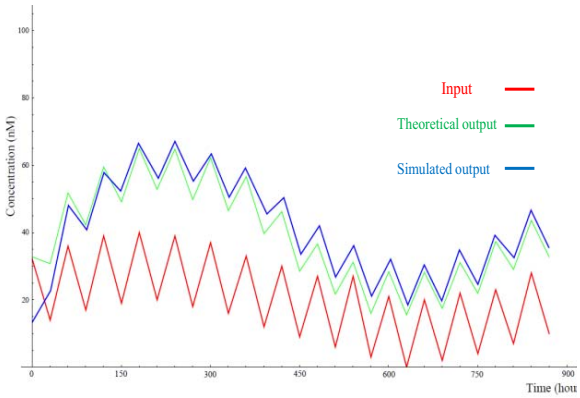


Figure 8. Simulation results for IIR filter.

For an 8-point 4-parallel RFFT implementation, the simulation results are illustrated in Figure 9. Concentrations for the inputs in the first and second stages and their corresponding theoretical outputs are tabulated in Table 1. The injection for inputs and collection for outputs are scheduled once every 250 hours.

Table 1. Input and output order for the 8-point RFFT

stage	I1	I2	I3	I4	O1	O2	O3	O4
first	1	1.5	1	0	7	1	0	0
second	1	1.5	1	0	0	3	0	0

Table 2 summarizes the simulation results of the three operations, namely, the FIR filter, the IIR filter, and the RFFT transform in the proposed framework. The errors in this table are computed as the difference between the output value obtained by simulation, o_s, and the theoretical output, o_t. Table 2 shows that as the complexity of operation is increased, or equivalently the number of reactions in the system is increased, the calculation time and the output error increase.

Table 2. Simulation results for FIR, IIR, and RFFT molecular systems in 4-phase scheme

system	no. of reactants	no. of reactions	Sample period (hrs)	% error $(\left\|\frac{o_s - o_t}{o_t}\right\| \times 100$
FIR	29	34	20	7.65
IIR	24	30	30	12.69
RFFT	184	202	750	23.6

V. COMPARISON

To evaluate the performance of our presented method, we compare the RFFT implementation with prior work in Table 3. As the table shows our 4-phase implementation is the fastest one; however, its accuracy is degraded. It is noticeable that even if we allow longer calculation time for 4-phase RFFT, it doesn't improve its output accuracy significantly. The 4-phase and synchronous schemes have less number of reactions and reactants compared to the RGB scheme. However, the number of reactants is not a limiting factor because DNA strands can easily generate a vast number of reactant types.

Table 3. Comparison for RFFT molecular systems.

system	No. of reactants	No. of reactions	Sample period (hrs)	% Error ($\left\|\frac{o_s - o_t}{o_t}\right\| \times 100$)
Synchronous	119	202	2400	7.8
RGB	213	225	850	22
Proposed Asynchronous	184	202	750	23.6

VI. CONCLUSION

This paper has presented an asynchronous 4-phase method for implementing discrete-time signal processing algorithms with molecular reactions. The proposed synthesis flow guarantees a conflict-free scheduling for any arbitrary DFG related to a DSP operation including computations and delay elements. Although the *in-vitro* simulation results using DNA strands validate the functionality of the method, it is essential to improve the speed and robustness of the method. Future work will be directed towards synthesis of signal processing functions using DNA with one to two orders of magnitude faster sampling rates.

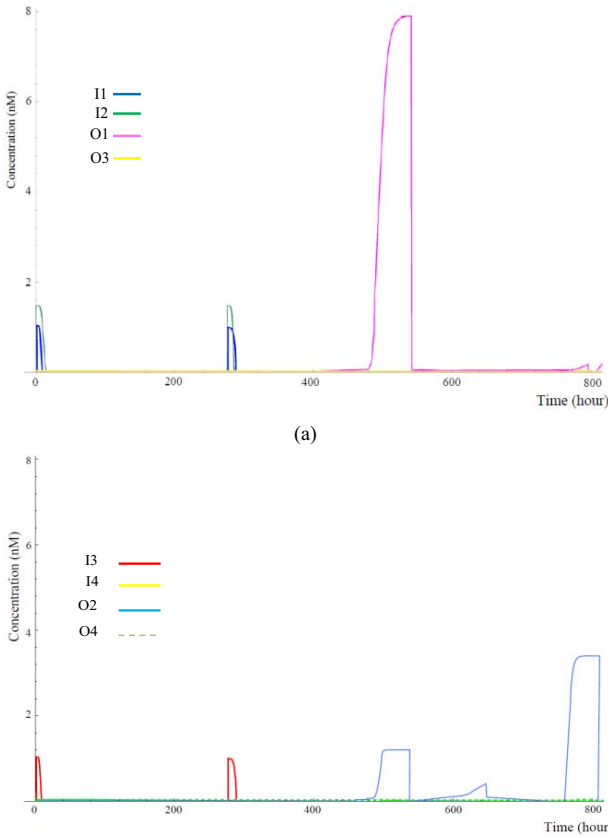


Figure 9. Simulation results for 8-point RFFT.

REFERENCES

[1] M. Samoilov, A. Arkin, and J. Ross, "Signal processing by simple chemical systems," *Journal of Physical Chemistry A*, pp. 10205-10221, 2002.

[2] J. C. Anderson, E. J. Clarke, A. P. Arkin, and C. A. Voigt, "Environmentally controlled invasion of cancer cells by engineered bacteria," *Journal of Molecular Biology*, pp. 619–627, 2006.

[3] J. D. Keasling, "Manufacturing Molecules through Metabolic Engineering," *Science*, pp. 1355-1358, 2010.

[4] H. Jiang, S. A. Salehi, M. D. Riedel, and K. K. Parhi, "Discrete-Time Signal Processing with DNA," *American Chemical Society (ACS) Synthetic Biology*, pp. 245-254, 2013.

[5] H. Jiang, M. D. Riedel, and K. K. Parhi, "Synchronous Sequential Computation with Molecular Reactions," *Proc. of ACM/IEEE Design Automation Conference (DAC2011)*, pp. 836-841, 2011.

[6] H. Jiang, M. D. Reidel, and K. K. Parhi, "Asynchronous Computations with Molecular Reactions," *Proc. of Asilomar Conf. on Sig., Sys. and Comp.*, pp. 493-497, Nov. 2011.

[7] L. Qian, D. Soloveichik, and E. Winfree, "Efficient turing-universal computation with DNA polymers," *Lecture Notes in Computer Science (DNA16),* 6518: 123-140, 2011.

[8] G. Seelig, D. Soloveichik, D. Y. Zhang, and E. Winfree, "Enzyme-free nucleic acid logic circuits," *Science*, vol. 314, pp. 1585–1588, 2006.

[9] D. Soloveichik, G. Seelig, and E. Winfree, "DNA as a universal substrate for chemical kinetics," *Proc. of the National Academy of Sciences*, pp. 5393–5398, 2010.

[10] B. Yurke, et al., "A DNA-fuelled molecular machine made of DNA," *Nature*, 406:605–608, 2000.

[11] P. Érdi and J. Tóth, *Mathematical Models of Chemical Reactions: Theory and Applications of Deterministic and Stochastic Models,* Manchester University Press, 1989.

[12] F. Horn and R. Jackson, "General mass action kinetics," *Archive for Rational Mechanics and Analysis*, 47:81–116, 1972.

[13] A. V. Oppenheim, R. W. Schafer, and J. R. Buck, *Discrete-Time Singal Processing*, 2nd ed. Englewood Cliffs, NJ: Prentice Hall, 1998.

[14] S. A. Salehi, R. Amirfattahi, and K. K. Parhi, "Pipelined Architectures for Real-Valued FFT and Hermitian-Symmetric IFFT with Real Datapaths," *IEEE Trans. Cir. Sys. II:Transactions Briefs,* 60(8), pp. 507-511, 2013.

[15] M. Garrido, K. K. Parhi, and J. Grajal, "A pipelined FFT architecture for real-valued signals," *IEEE Trans. Cir. Sys. I: Reg. Papers*, 56(12), pp. 2634 – 2643, 2009.