Digital Signal Processing with DNA

Marc D. Riedel, Electrical and Computer Engineering, University of Minnesota

Just as electronic systems implement computation in terms of voltage (*energy per unit charge*), molecular systems compute in terms of chemical concentrations (*molecules per unit volume*). In this paper, we discuss methods for implementing digital signal processing (DSP) operations with molecular reactions. From a high-level specification of the computation, we demonstrate how to synthesize generic chemical reaction networks that perform iterative operations such as filtering. We illustrate our design methodology with several types of common DSP operations including low-pass filtering and Fast Fourier transforms (FFTs).

Our chemical reaction networks compute robustly, assuming only that some reactions are faster than others. Indeed, at the chemical reaction level, we use only coarse rate categories ("fast" and "slow") for the kinetic constants. Given such categories, the computation is accurate regardless of the specific reaction rates. Rather than depending on rates, our designs are "self-timed": the next input value is not sampled until the current output value is cleared to zero. (Alternatively, we are also exploring synchronization via a *clock signal* that is generated through robust, sustained chemical oscillations.) All of our designs consist of either unimolecular or bimolecular reactions, i.e., reactions with one or two reactants, respectively.

We demonstrate how to map our chemical reaction networks to DNA-strand displacement reactions (as discussed by Soloveichik *et al.*, PNAS 2010). We validate our designs through ODE simulations of the mass-action kinetics of the DNA reactions.

Here we illustrate our methodology with the design and simulation of a finite impulse response (FIR) filter. This system computes a *moving average*: given a time-varying input signal X, the output Y is a smoother version of it (i.e., the system is a *low-pass filter*).

Mathematically, the system sets the output to be one-half the current input value plus one-half the previous value. Reactions – implement this filter. (In what follows, when we say a "signal" we mean the concentration of a chemical type.) In these reactions, the input signal X is transfered to the signals A and C (a *fanout* operation). Then A and C are reduced by half (*scalar multiplication* operations). Then A is transfered to the output signal Y and C is transfered to the signal R, the first part of a *delay*. The delay operation transfers the signal from R to B to G. Once the signal has moved through the delay operation, B is transfered to Y. (Since this signal is is combined with the signal produced from A, this is an *addition*.)

The transfers within the delay operation are accomplished by Reactions (). Conceptually, there are three stages: red, green and blue. Transfers between two color categories are enabled by the absence of 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. Reactions () implement this synchronization. These reactions continually generate the types r, g, and b. These types only persist in the absence of the corresponding signals. Reactions () introduce positive feedback: they speed up transfers between two color categories as molecules in one category are "pulled" to the next one. Note that input signal X is sampled in the green-to-blue phase. We assume that an external source supplies the input signal and an external sink consumes the output signal.

The figure shows the results of an ODE simulation of the DNA reactions for our moving-average filter. We use $C_{max} = 10^{-5} mol/L$; all the other parameters are similar to those used in Soloveichik *et al.*, PNAS 2010.

$$
\begin{aligned}
g + X &\xrightarrow{\text{slow}} A + C \\
2C &\xrightarrow{\text{fast}} R \\
2A &\xrightarrow{\text{fast}} Y \\
b + R &\xrightarrow{\text{slow}} G \\
r + G &\xrightarrow{\text{slow}} B \\
g + B &\xrightarrow{\text{slow}} Y
\end{aligned}
\qquad (1)
$$

$$
\begin{aligned}
\varnothing &\xrightarrow{\text{slow}} r \\
R + r &\xrightarrow{\text{fast}} R \\
Y + r &\xrightarrow{\text{fast}} Y \\
\varnothing &\xrightarrow{\text{slow}} g \\
G + g &\xrightarrow{\text{fast}} G \\
\varnothing &\xrightarrow{\text{slow}} b \\
B + b &\xrightarrow{\text{fast}} B \\
X + b &\xrightarrow{\text{fast}} X
\end{aligned}
\qquad (2)
$$

$$
\begin{aligned}
2R &\underset{\text{fast}}{\overset{\text{slow}}{\rightleftharpoons}} I_R \\
I_R + X &\xrightarrow{\text{fast}} A + C + 2 \\
2G &\underset{\text{fast}}{\overset{\text{slow}}{\rightleftharpoons}} I_G \\
I_G + R &\xrightarrow{\text{fast}} 3G \\
2B &\underset{\text{fast}}{\overset{\text{slow}}{\rightleftharpoons}} I_B \\
I_B + G &\xrightarrow{\text{fast}} 3B \\
2Y &\underset{\text{fast}}{\overset{\text{slow}}{\rightleftharpoons}} I_Y \\
I_Y + B &\xrightarrow{\text{fast}} 3Y
\end{aligned}
\qquad (3)
$$