ACS SyntheticBiology

Research Article

pubs.acs.org/synthbio

Computational Design of Nucleic Acid Feedback Control Circuits

Boyan Yordanov,[†] Jongmin Kim,[‡] Rasmus L. Petersen,[†] Angelina Shudy,[§] Vishwesh V. Kulkarni,*[,§] and Andrew Phillips*[,†]

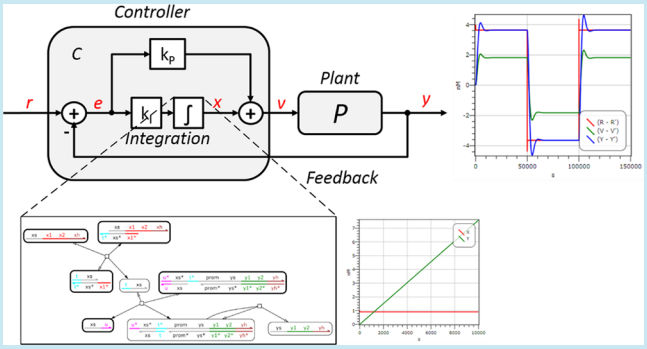
[†]Microsoft Research, Cambridge CB1 2FB, United Kingdom

[‡]Division of Biology and Biological Engineering, California Institute of Technology, Pasadena, California 91125, United States

[§]Department of Electrical Engineering, University of Minnesota, Minneapolis, Minnesota 55455, United States

Supporting Information

ABSTRACT: The design of synthetic circuits for controlling molecular-scale processes is an important goal of synthetic biology, with potential applications in future *in vitro* and *in vivo* biotechnology. In this paper, we present a computational approach for designing feedback control circuits constructed from nucleic acids. Our approach relies on an existing methodology for expressing signal processing and control circuits as biomolecular reactions. We first extend the methodology so that circuits can be expressed using just two classes of reactions: catalysis and annihilation. We then propose implementations of these reactions in three distinct classes of nucleic acid circuits, which rely on DNA strand displacement, DNA enzyme and RNA enzyme mechanisms, respectively. We use these implementations to design a Proportional Integral controller, capable of regulating the output of a system according to a given reference signal, and discuss the trade-offs between the different approaches. As a proof of principle, we implement our methodology as an extension to a DNA strand displacement software tool, thus allowing a broad range of nucleic acid circuits to be designed and analyzed within a common modeling framework.



KEYWORDS: *feedback control, molecular programming, nucleic acid circuits, DNA strand displacement, genelet, DNA toolbox*

The design of synthetic circuits for controlling processes at the molecular scale is an important goal of synthetic biology, with potential applications ranging from the metabolic production of biomaterials to the design of "smart" therapeutics capable of both diagnosis and treatment. So far, a number of synthetic devices have been designed and implemented *in vivo* using protein expression and gene regulation mechanisms, including logic gates,[1] memory elements,[2] oscillators,[3] filters,[4,5] and controllers of cellular differentiation processes.[6] However, the complexity of such circuits is often limited by the low availability of well-understood genetic components that can be reliably assembled, together with the lack of modular design and implementation strategies, as well as interactions with the host organism that are hard to predict.

Complementary to the design of *in vivo* synthetic circuits, cell-free synthetic biology has emerged as a powerful biotechnological framework, enabled by the capacity of biochemical processes to function outside of cells under appropriate conditions. Such approaches are already at the core of a number of experimental techniques, including Polymerase Chain Reaction and Gibson assembly, and are currently being developed for the biosynthesis of chemicals and materials.[7] At the same time, cell-free synthetic biology provides a platform for the study of natural systems, as in the use of cell-free models of genetic regulation.[8] In both cases, a cell-free approach offers the benefits of greater control, flexibility, and relaxed design constraints compared to *in vivo* implementations. However, as system complexity increases, novel strategies are needed to better control these systems.[9] Alongside improved experimental techniques, computational design and analysis of synthetic circuits for the regulation of biochemical processes could help address these challenges.

Recently, the direct use of nucleic acids for performing computation has emerged as a promising approach for the engineering of biological circuits.[10] In such systems, the sequences of nucleic acid components dictate the interactions between them, through well-understood mechanisms such as Watson−Crick base-pairing. This enables precise programming of molecular interactions by the choice of appropriate sequences. Such programmability, together with recent advances in synthesis methods and the ability to interface with molecular components, make nucleic acids a highly promising substrate for the implementation of biochemical circuits.

A number of approaches exist for the design of nucleic acid circuits. For example, the programmed self-assembly of RNA

Special Issue: IWBDA 2013

Received: October 22, 2013
Published: July 25, 2014

structures has been used to catalyze hydrogen biosynthesis within cells,[11] while DNA structures have been embodied with rudimentary logic to enable the conditional release of a molecular payload if two antigens, indicative of specific cancers, are present simultaneously on the surface of target cells.[12] More generally, however, synthetic circuits for the regulation of biochemical processes require computations that include more complex dynamics.[13] Recently, a number of promising approaches capable of implementing such dynamics have been proposed, based on DNA strand displacement,[14] DNA enzyme,[15] and RNA enzyme[16] strategies. Such approaches have been used for example to implement distributed consensus via feedback control mechanisms,[17] predator−prey dynamics,[18] and transcriptional oscillators,[19] respectively. Overall, the alternative approaches for implementing nucleic acid circuits offer different advantages, and there is a need to rigorously compare these approaches, for instance by applying them to the design of a common system. Such comparisons may help to identify design strategies for hybrid systems that combine the strengths of these different approaches, offering a promising direction toward the construction of novel systems of increased complexity.

Linear control and signal processing systems, such as low-pass, high-pass, and band-pass filters, represent an important class of systems that have found numerous applications in engineering fields including aerospace, telecommunication, automotive, storage, and power systems.[20,21] Crucially, even when a system is highly nonlinear and subject to unpredictable perturbations, it can be robustly regulated using a linear controller, provided that the performance demands, such as insensitivity to measurement noise and rejection of load disturbances, are not too high and that the process has almost monotone step response.[22] Biochemical implementation of such controllers could provide a solution to problems involving the optimization or regulation of a vast array of cellular processes, including cell metabolism and biomass production. In this paper, we focus on the implementation of Proportional/Integral (PI) controllers (see Figure 1 for an example), where the plant output v is computed as a weighted sum of the error signal e and the error signal accumulated (i.e., integrated) over time. The inclusion of the integral term renders such controllers dynamic, accelerating the tracking process and, in many cases, effectively eliminating the steady state error. Notably, simpler systems including on−off[23] or purely proportional feedback controllers[24] cannot always meet such performance requirements.

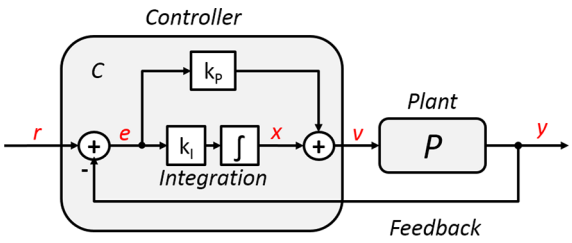


Figure 1. Feedback control system composed of a physical plant P and a controller C. The error signal e is computed as the difference between the reference signal r and the plant output y. The controller automatically computes and adjusts the plant input v to minimize the error and track the reference signal, according to the tuning parameters K_p and K_i.

Recently, a strategy for implementing linear systems with chemical reaction networks was proposed by Oishi and Klavins,[25] based on two key abstractions. First, a pair of chemical species (*e.g.*, X^+ and X^-) was used to represent the complementary positive and negative component of a real-valued signal, with a fast annihilation reaction (*e.g.*, $X^+ + X^- \rightarrow \varnothing$) ensuring a minimal signal representation. In the following, we use $X^\pm$ to jointly refer to the complementary signals X^+ and X^- from the representation of X. In addition, we use the notation $X^\pm \xrightarrow{k} X^\pm + Y^\mp$ as a shorthand for the two reactions $X^+ \xrightarrow{k} X^+ + Y^-$ and $X^- \xrightarrow{k} X^- + Y^+$ (note that in this example the signs of $Y^\mp$ are reversed). Second, three idealized chemical reactions—catalysis, degradation, and annihilation—were shown to be sufficient to implement the mathematical operations of integration, summation, and gain.[25] This provided a structured approach for the chemical implementation of finite-dimensional linear systems, which can be expressed in terms of these basic blocks.[20,21] A DNA strand displacement implementation of a controller was also proposed,[25] based on a scheme for converting an arbitrary chemical reaction network to a DNA strand displacement system.[26]

Motivated by potential applications in cell-free biotechnology, and as a step toward understanding the mechanisms that could enable future circuits within cells, in this paper, we explore a number of approaches for the implementation of dynamic computation using nucleic acids. In particular, we adapt existing DNA strand displacement, DNA enzyme, and RNA enzyme strategies to implement arbitrary linear systems expressed as chemical reaction networks.[25] To facilitate the design, study, and rigorous comparison of different implementations within a unified computational environment, we extend the Visual DSD tool,[27] which previously supported only the DNA strand displacement strategy, and use it to compare novel implementations of a common computational system—a chemical controller—realized using each of the three strategies. Our methods enable the design of more complex controllers than studied previously.[24] Besides the immediate importance of such systems to *in vitro* biotechnology, such comparisons may help identify improved hybrid designs that combine different implementation approaches, as potential inspiration for future *in vivo* systems.

RESULTS

DNA Strand Displacement Models. *DNA Strand Displacement* (DSD)[14] is an implementation strategy based on the hybridization of DNA strands with partial or full complementarity, resulting in the displacement of one or more prehybridized strands. Recent theoretical work[26] proposed a method for implementing an arbitrary Chemical Reaction Network (CRN) as a DSD system, where each chemical species was implemented as a single strand of DNA consisting of four distinct domains. This so-called *four domain* method was subsequently used to propose a DSD implementation of linear input output systems expressed as CRNs.[25] Recent examples of large-scale systems that have been implemented experimentally using the strand displacement approach include catalytic[28] and digital logic circuits,[29] neural networks,[30] and an analogue consensus algorithm.[17]

To enable more precise computational modeling of DSD systems, we extended the Visual DSD tool with the notion of a *degree of complementarity*[26] (see Supporting Information). This made it possible to model interactions between domains that are not exactly complementary but instead contain one or more mismatched bases. The extension is based on an earlier proposal,[31] which we generalized to be robust to rotation

dx.doi.org/10.1021/sb400169s | ACS Synth. Biol. 2014, 3, 600−616

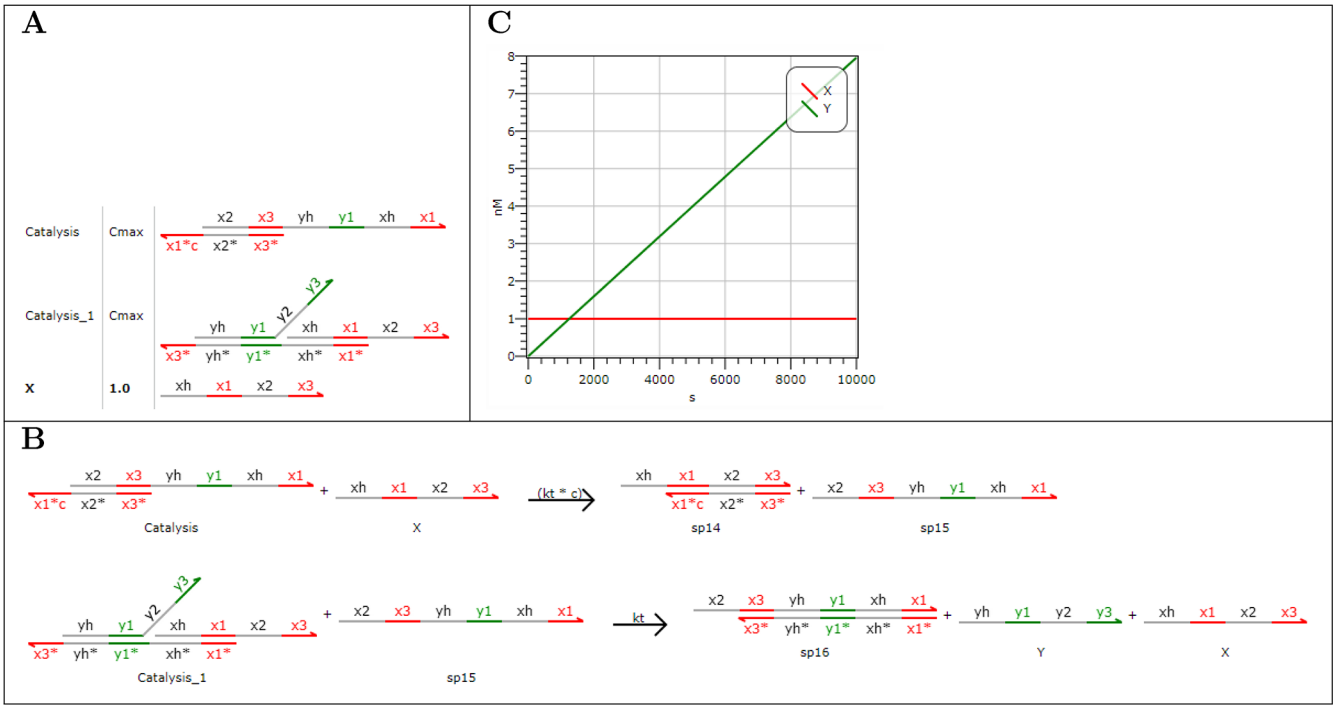


Figure 2. Visual DSD implementation of a catalytic 4-domain DNA strand displacement circuit. The circuit design and associated kinetic parameters were originally proposed by Oishi and Klavins.[25] (A) Initial concentrations (nM) of strand X and complexes Catalysis and Catalysis$_1$, with C_{max} = 1000 nM. (B) Strand displacement reactions generated automatically from the initial conditions by Visual DSD, with toehold binding rate $kt = 10^{-3}$ nM^{-1} s^{-1}. The binding rate of toehold x_1 is modulated by the degree of complementarity $c = 8 \times 10^{-4}$ resulting in an effective binding rate of 8×10^{-7}nM^{-1} s^{-1}. (C) Corresponding simulation results.

symmetry and implemented in Visual DSD. A domain x^* with degree of complementarity c is written as x^*c, where c is a real number between zero and one, with one denoting perfect complementarity. In an experimental setting, changing the degree of complementarity can significantly alter the binding rate of two complementary domains, with experimentally measured rates ranging from roughly 10^{-7} nM^{-1} s^{-1} to 10^{-3} nM^{-1} s^{-1} for domains between two and six nucleotides in length.[32] Since the exact relationship between DNA sequence and hybridization kinetics is not well-understood, we chose to model the degree of complementarity as a rate modifier. If c = 1 then hybridization between x^*c and x proceeds at the usual rate given by k_x, which is a property of the domain x that can be specified by the modeler or fit to experimental data.[17] If c < 1 then hybridization proceeds at a lower rate $k_x \times c$.

We used this extended Visual DSD tool to model a four-domain DSD catalytic circuit (Figure 2) corresponding to the reaction $X \rightarrow X + Y$. The signal X is implemented as a single strand $\langle x_h\ x_1\ x_2\ x_3 \rangle$ consisting of four distinct domains, where domains x_1 and x_3 are assumed to be short domains called *toeholds*, which bind reversibly to their complement. The reaction is implemented by two complexes, Catalysis and Catalysis$_1$, which consume the input X and produce the outputs X and Y (Figure 2A). The Visual DSD tool automatically generates the set of possible reactions that can be produced from these initial complexes (Figure 2B). The signal X binds to the Catalysis complex on toehold x_1, resulting in the displacement of the intermediate strand sp_{15}. The intermediate strand then binds to the Catalysis$_1$ complex, resulting in the displacement of the signals X and Y. Since x_1^* has degree of complementarity c, the reaction proceeds at rate $kt \times c$, where kt is the rate associated with all toeholds in the system. Crucially, the degree of complementarity allows the signal X to take part in other reactions on the same toehold but at different rates. This feature is essential for implementing different interaction rates involving the same signal. We simulated the behavior of the system over time, to demonstrate its catalytic activity (Figure 2C).

Following a similar approach, we modeled a complete four-domain PI controller design[25] (Supporting Information). The initial 48 species of the PI controller and 8 reactions of the associated production plant were programmed in a modular way, and all 40 reactions of the PI controller were generated automatically by Visual DSD, together with the additional 58 species produced by these reactions. The automatically generated reactions were shown to be consistent with the manually specified reactions of the original model.[25]

DNA Enzyme Models. An alternative strategy for implementing dynamic behavior in nucleic acids relies on a combination of DNA strands and DNA enzymes, including DNA polymerase, exonuclease, and nickase. In particular, the *PEN DNA toolbox* (also known as the *DNA toolbox*)[33] makes use of these enzymes together with elementary modules for activation, autocatalysis, and inhibition. These modules can be arbitrarily connected in circuits to encode a broad range of dynamic behaviors, including switchable memories[33] and oscillators.[15,18] In contrast to DNA strand displacement, the DNA toolbox does not seek to implement arbitrary chemical reactions directly but instead to support systems consisting of activation and inhibition mechanisms, inspired by naturally occurring gene regulation networks.

To enable precise computational modeling of DNA enzyme systems, we extended the Visual DSD tool with the notion of *user-defined reactions* (see Supporting Information). This allowed reactions of the form $X_1 + \cdots + X_n \xrightarrow{r} X_{n+1} + \cdots + X_m$ to be specified

dx.doi.org/10.1021/sb400169s | ACS Synth. Biol. 2014, 3, 600−616

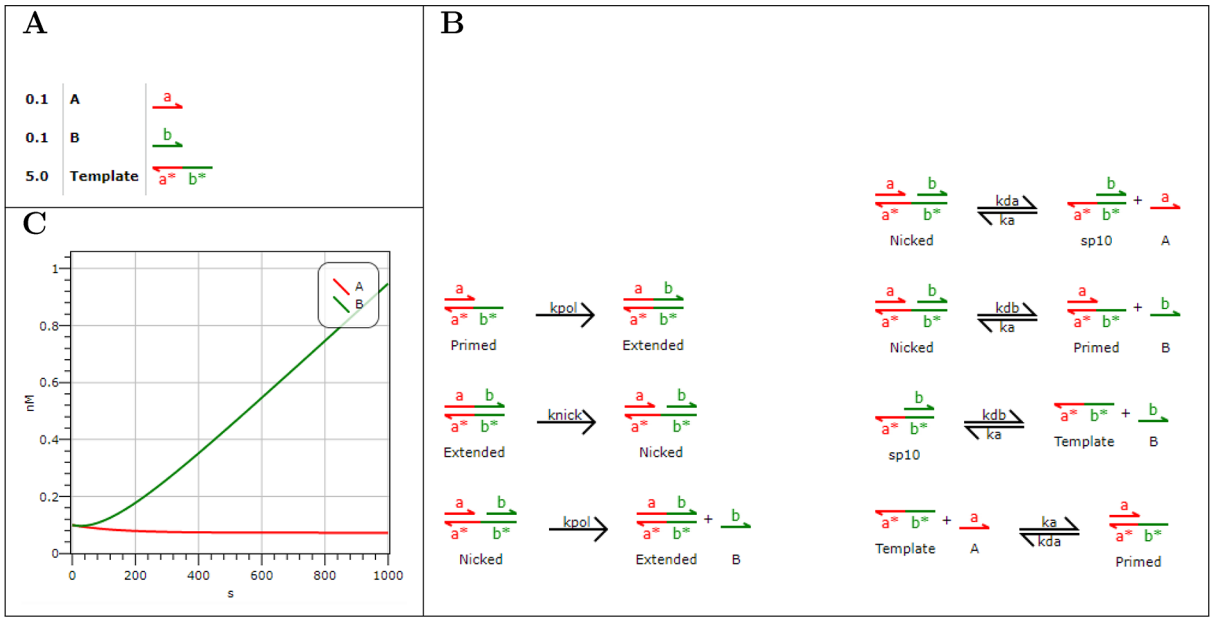


Figure 3. Visual DSD implementation of a catalytic DNA toolbox circuit. The circuit design and associated kinetic parameters were originally proposed by Montagne et al.[15] (A) Initial concentrations (nM) of strands *A*, *B*, and *Template*. (B) Enzymatic reactions modeled explicitly in Visual DSD (left column), with rates $kpol = 0.2833\ s^{-1}$ and $knick = 0.05\ s^{-1}$. Remaining reactions generated automatically from the initial conditions by Visual DSD (right column), with rates $ka = 4.3333 \times 10^{-4}\ nM^{-1}\ s^{-1}$, $kda = 0.0383\ s^{-1}$, and $kdb = 0.0135\ s^{-1}$ used as binding and unbinding rates for domains *a* and *b*. (C) Corresponding simulation results.

directly, where $X_1,...,X_m$ are arbitrary DSD species. During compilation, the user-defined reactions are combined with the automatically generated strand displacement reactions, enabling arbitrary enzymatic reactions to be modeled explicitly in Visual DSD. This also enabled the creation and editing of purely CRN models, which were used for direct comparison with more complex nucleic acid designs.

We used the extended Visual DSD tool to model a catalytic DNA toolbox circuit (Figure 3) corresponding to the reaction $A \rightarrow A + B$. We implemented the signal *A* as a single strand $\langle a \rangle$, and the catalytic reaction as a *Template* strand, whose sequence is the reverse complement of $\langle a\ b \rangle$. The enzymatic reactions involving nickase and polymerase were programmed explicitly (Figure 3B left column), and the remaining binding and unbinding reactions were generated automatically. After binding of the catalyst strand $\langle a \rangle$, DNA polymerase extends the 3′ end of the bound strand with an additional *b* domain. The resulting strand $\langle a\ b \rangle$ is then nicked by a nicking enzyme to produce two adjacent strands, $\langle a \rangle$ and $\langle b \rangle$, bound to the template. By extending the catalyst, the DNA polymerase displaces the initially bound product $\langle b \rangle$, which can also unbind spontaneously at the operating temperature of the experiment. For enzymatic reactions we assumed that all enzymes were in excess with approximately constant concentrations, such that the rate constants are first order, as in previously published models.[15] More complex functional forms such as Michaelis–Menten enzyme kinetics are also directly supported by Visual DSD. We simulated the behavior of the system over time, to demonstrate its catalytic activity (Figure 3C).

Following a similar approach, we modeled a complete DNA toolbox oscillator[15] (see Supporting Information). The initial 17 species and 15 enzymatic reactions were programmed in a modular way, and the remaining 12 reactions were generated automatically by Visual DSD, together with the additional 2 species produced by these reactions. The full system was simulated by Visual DSD and its behavior was shown to be consistent with that of the original model.[15]

RNA Enzyme Models. A third strategy for implementing dynamic behavior in nucleic acids relies on the use of RNA enzymes such as RNA polymerase and ribonuclease. In particular, *genelets*[16,34] make use of these enzymes, together with double-stranded DNA templates consisting of a promoter sequence followed by a transcript sequence. Crucially, the first few nucleotides of the promoter are single stranded and need to be completed by an activator strand in order for transcription to be initiated by RNA polymerase. Systems constructed experimentally using the genelet approach include bistable switches,[34,35] oscillators,[16] timing circuits,[36] and fold-change detectors.[37]

To enable precise computational modeling of RNA enzyme systems, we extended Visual DSD with the notion of a *composite domain* (see Supporting Information), which is a defined sequence of adjacent toehold domains. Composite domains can be defined by the programmer and associated with corresponding binding rates, which are used to automatically generate binding reactions on complementary composite domains.

We used the extended Visual DSD tool to model a genelet with a negative feedback loop (Figure 4). The genelet is composed of the double-stranded template T11, which contains the promoter sequence of RNA polymerase from bacteriophage T7, represented as $\langle t\ prom \rangle$. The region $\langle t \rangle$ of the promoter sequence is single-stranded, which results in significantly reduced transcription. To activate the genelet, the DNA activator strand A2 binds to the template and completes the promoter duplex, resulting in increased transcription. The RNA transcript rI2 can inhibit transcription of the genelet by displacing the activator A2 via toehold-mediated strand displacement. The transcript can also bind directly to A2 in solution. Degradation of RNA signal species is implemented using the enzyme ribonuclease (RNase) H, which targets RNA signals bound to single stranded DNA. As

dx.doi.org/10.1021/sb400169s | *ACS Synth. Biol.* 2014, 3, 600–616

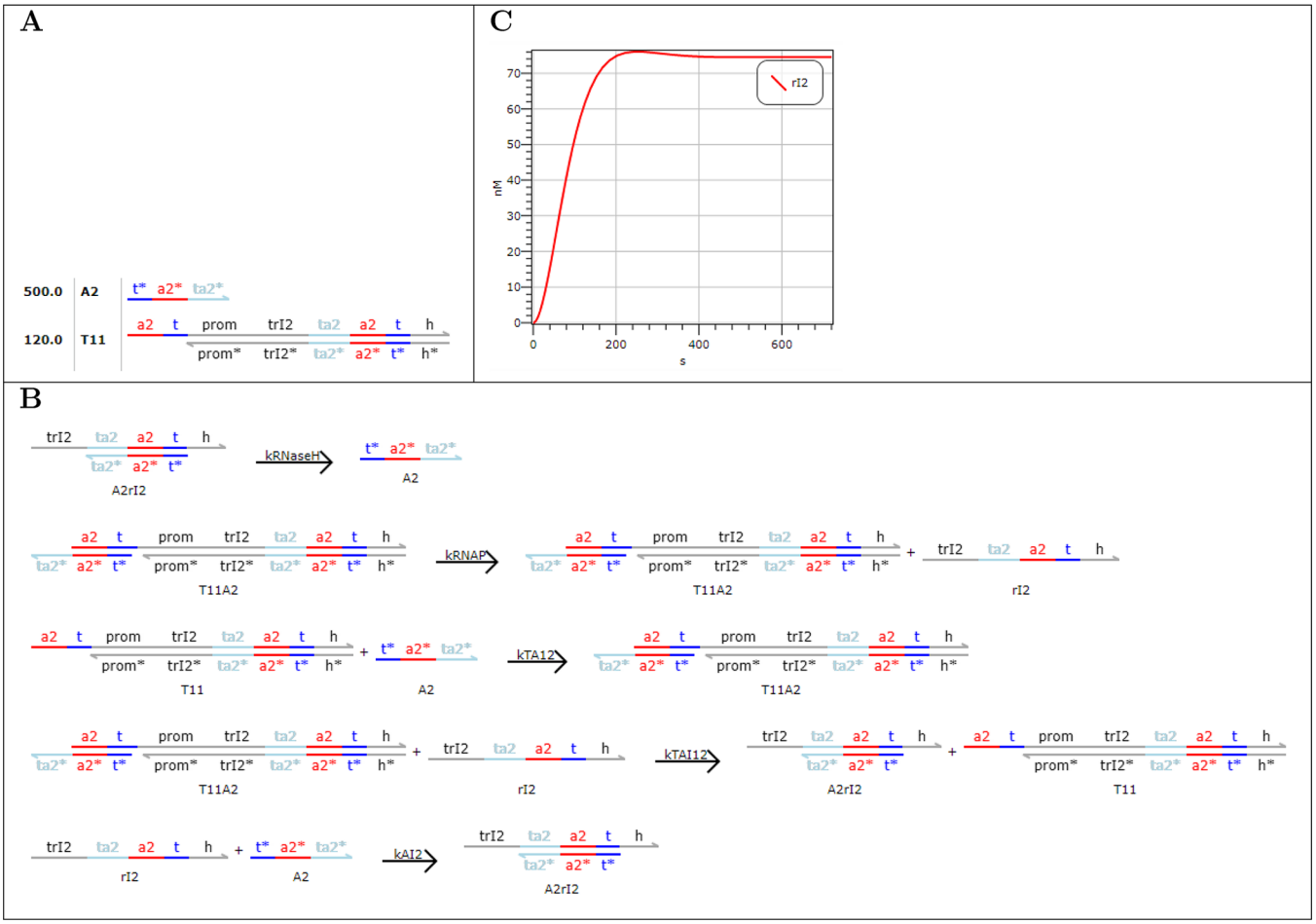


Figure 4. Visual DSD implementation of a genelet circuit with a negative feedback loop. The circuit design and associated kinetic parameters were originally proposed by Kim and Winfree,[16] except that here the output of the genelet directly inhibits its own production. (A) Initial concentrations (nM) of strand *A* and genelet *T*11. (B) The first two enzymatic reactions were modeled explicitly in Visual DSD, with rates $kRNAP = 0.0323\ s^{-1}$ and $kRNaseH = 0.0196\ s^{-1}$. The remaining reactions were generated automatically from the initial conditions by Visual DSD, with rates $kTA12 = 1.4 \times 10^{-5}$ $nM^{-1}\ s^{-1}$, $kTAI12 = 1.4 \times 10^{-4}\ nM^{-1}\ s^{-1}$ and $kAI2 = 3.1 \times 10^{-5}\ nM^{-1}\ s^{-1}$, used as binding rates for composite domain (*a*2;*t*), domain *ta*2 and composite domain (*ta*2;*a*2;*t*), respectively. (C) Corresponding simulation results.

with the DNA toolbox designs, we assumed that polymerase and ribonuclease enzymes are in excess at approximately constant concentrations. We approximated enzyme activity by first-order rate constants, as in previously published models.[37]

Following a similar approach, we modeled a complete genelet oscillator[16] (see Supporting Information). The initial 12 species and 6 enzymatic reactions were programmed in a modular way, and the remaining 8 reactions were generated automatically by Visual DSD. The full system was simulated by Visual DSD, and its behavior was shown to be consistent with that of the original model.[16]

Chemical Reaction Network Design. We used the Visual DSD extensions described previously to model a Proportional Integral (PI) controller as a set of idealized chemical reactions, following the method proposed by Oishi and Klavins.[25] We first modeled the elementary components of degradation, annihilation, and catalysis (Figure 5A−C) and then used these to model the more complex components of integration, gain, and summation (Figure 5D−F). The components were modeled and simulated under varying conditions (Figure 5A−F, Supporting Information), and the simulations were shown to be consistent with previously published results.[25] We then used these components to model the full PI controller connected to an elementary production plant (Figure 6A). When coding the models in Visual DSD, we used *X* and *X*′ instead of X^+ and X^-, respectively, since the (+) and (−) symbols are reserved for arithmetic operations. The output $V^{\pm}$ of the controller was used as an input to the plant, and the output $Y^{\pm}$ of the plant was used as an input to the controller, together with the reference signal $R^{\pm}$. We simulated this closed loop system for a given reference signal, and perturbed the system by increasing and then decreasing the load on the plant (Figure 6B). The controller automatically compensated for the changes in load by adjusting the plant input, such that the plant output continued to match the reference signal. We also simulated the closed loop system by varying the reference signal over time (Figure 6B). The controller automatically adjusted the plant input to ensure that the plant output continued to match the reference.

When considering alternative nucleic acid implementations of the idealized PI controller, one obstacle we encountered was the difficulty in implementing signal-specific degradation. In order for the controller to function correctly, only a subset of the available signals should be degraded (Figure 6A); however, certain DNA enzyme implementations degraded all signals uniformly. To address this issue, we investigated whether the scheme of idealized chemical reactions could be further

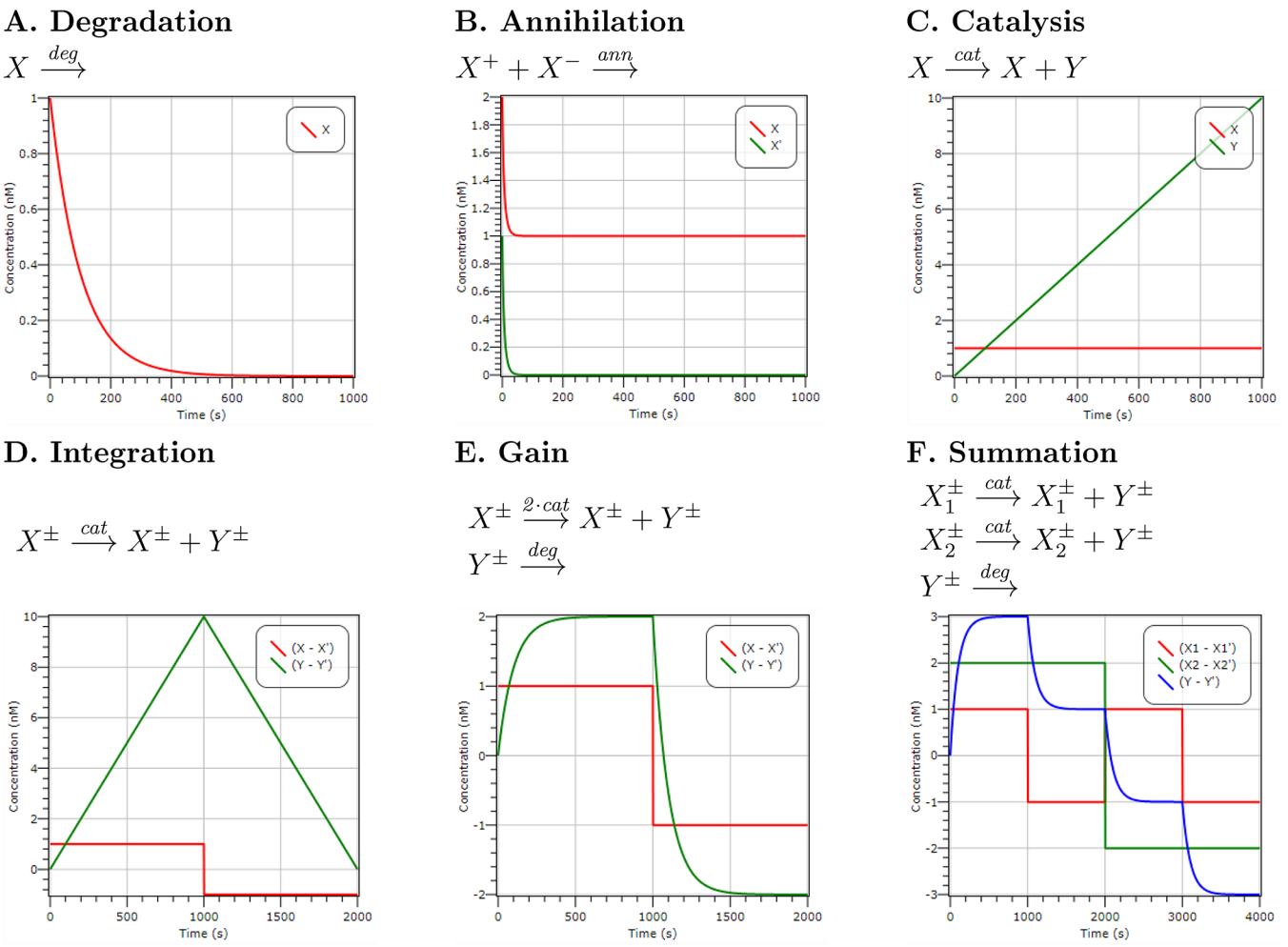


Figure 5. Chemical reaction models and simulations of basic components. For each pair of complementary signals X_i^+, X_i^- we assume the presence of an annihilation reaction $X_i^+ + X_i^- \xrightarrow{ann} \varnothing$, which is omitted for conciseness. Simulations were run for deg = cat = 0.01 s^{-1} and ann = 0.1 nM^{-1} s^{-1}.

generalized in order to reduce the constraints on the set of implementations that could be considered. The original scheme required three elementary reactions: degradation, catalysis, and annihilation. We hypothesized that, under certain conditions discussed below, this scheme could be further generalized by replacing each degradation reaction $X^\pm \xrightarrow{deg} \varnothing$ with a catalytic reaction $X^\pm \xrightarrow{deg} X^\pm + X^\mp$, together with an annihilation reaction $X^\pm + X^\mp \xrightarrow{ann} \varnothing$. This resulted in only two elementary reaction types, catalysis and annihilation. We term this generalization *catalytic degradation*. In order for catalytic degradation to be a valid approximation of degradation, we require the rate of change of X^+, written $\dot{X}^+$, to be equal in both schemes, such that $\dot{X}^+ = -ann\,[X^+][X^-] = -deg\,[X^+]$. We show that this equality holds under the Quasi Steady State (QSS) assumption $\dot{X}^- = 0$, since $[X^-] = deg/ann$. Thus, catalytic degradation of X^+ is a valid approximation provided X^- is in quasi steady state, which holds when $[X^+] \gg [X^-]$. Similarly, the symmetric approximation for catalytic degradation of X^- holds when $[X^-] \gg [X^+]$. Therefore, catalytic degradation is a valid approximation when X^+ is substantial and X^- is small or vice versa, which can be achieved through a fast annihilation process between the two species. To test this strategy, we used our generalized scheme to model the PI controller and plant (Figure 7), where each degradation reaction was substituted by catalytic degradation. For fast annihilation reactions, catalytic degradation was almost indistinguishable from standard degradation. However, when the annihilation rate was reduced such that ann ≈ deg, the approximation began to break down (Figure 7A). Despite this breakdown, the overall behavior of the PI controller was preserved (Figure 7B–C), although the individual signals were not reduced to a minimal representation (see Supporting Information). Since the original three-reaction scheme requires annihilation to be significantly faster than degradation, catalytic degradation remains a valid approximation under the same assumptions as the original scheme.

DNA Strand Displacement Implementation. We implemented the idealized PI controller as a DNA strand displacement system. Inspired by recent experimental results,[17] we based our implementation on a *two domain* DNA strand displacement scheme,[38] in which each chemical species is represented as a single strand of DNA consisting of only two domains. We proposed a number of refinements to the original scheme, to improve scalability and to reduce the number of strands needed, as summarized below.

Annihilation. Each formal reaction $X_1,...,X_N \rightarrow r\,Y_1,...,Y_M$ was encoded in two parts,[38] a Join circuit that collects the reactants $X_1,..,X_N$, and a Fork circuit that produces the products $Y_1,...,Y_M$. In the case of the annihilation reaction $X + X' \xrightarrow{ann} \varnothing$ there are no products, so the reaction is encoded as a Join circuit with two

dx.doi.org/10.1021/sb400169s | ACS Synth. Biol. 2014, 3, 600–616

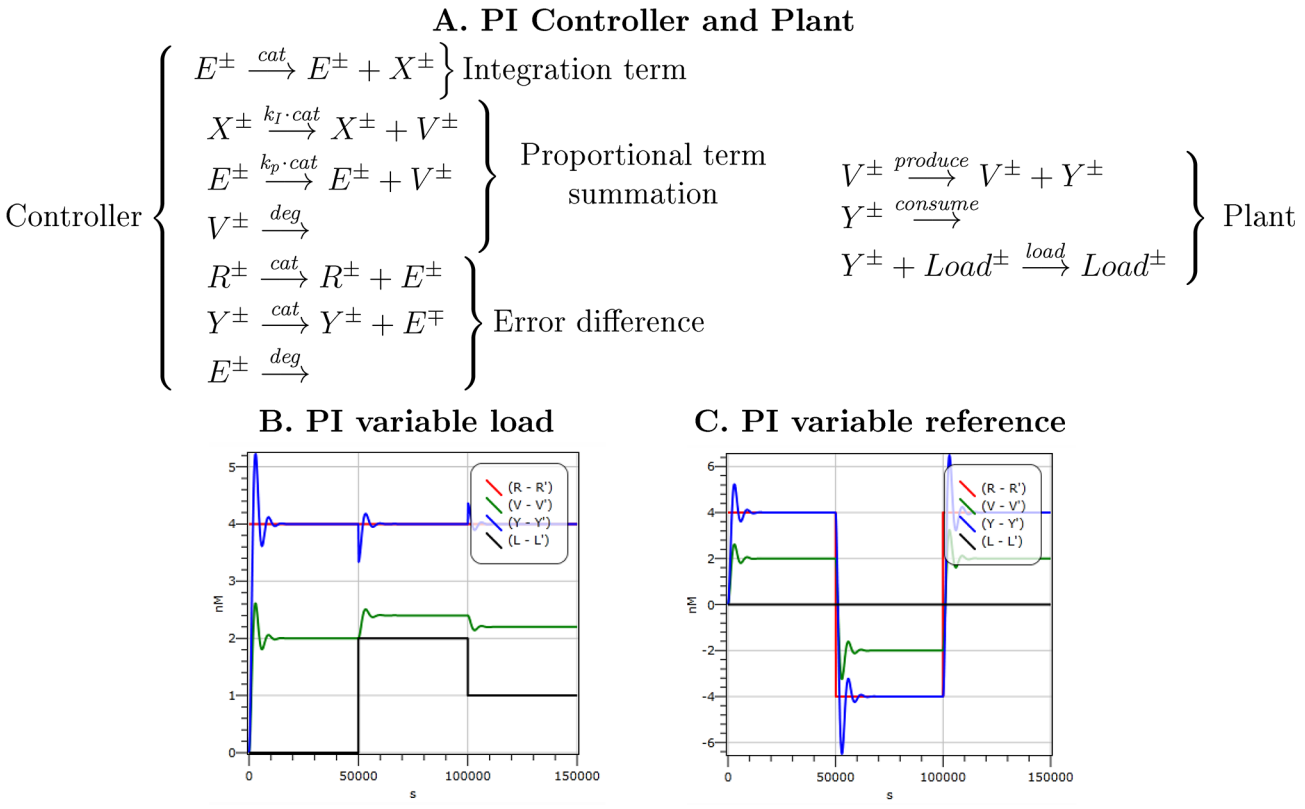


Figure 6. Chemical reaction model and simulation of a Proportional Integral controller, connected to a production plant. For each pair of complementary signals $X^{\pm}$, $Y^{\pm}$, $E^{\pm}$, $V^{\pm}$, and $R^{\pm}$ an annihilation reaction is also present, for example $X^{+} + X^{-} \xrightarrow{ann} \varnothing$ for signals $X^{\pm}$, but is omitted for conciseness. Simulations were run for controller tuning parameters $k_I = k_p = 1.0$ and reaction rates deg = cat = 0.0008 s^{-1}, ann = 0.01 nM^{-1} s^{-1}, produce = 0.2 s^{-1}, consume = 0.1 s^{-1}, and load = 0.01 $nM^{-1}s^{-1}$. Plots show absolute values of reference R, plant input V, plant output Y, and load L.

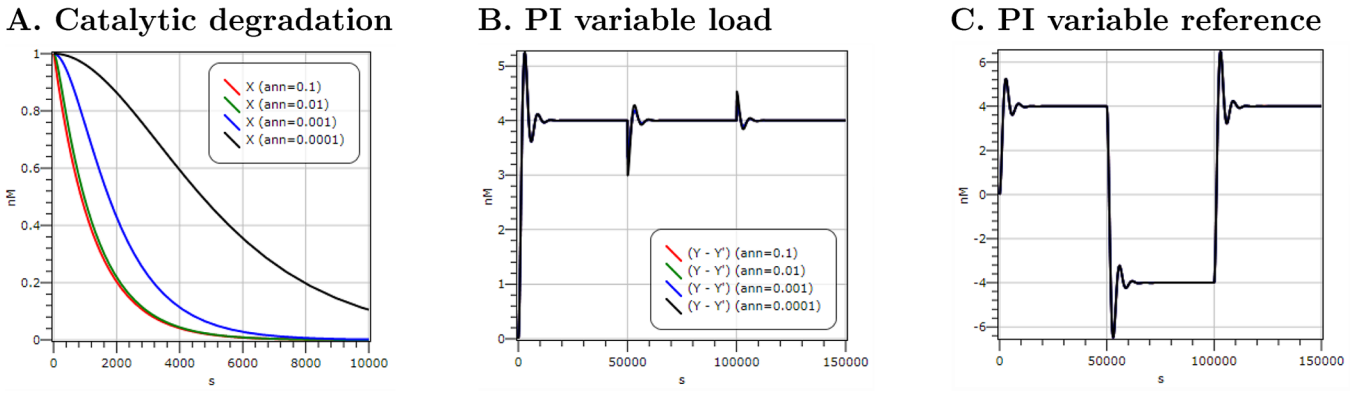


Figure 7. Simulation results for a simplified *catalytic degradation* scheme in which each degradation reaction $X^{\pm} \xrightarrow{deg} \varnothing$ is replaced by a catalytic reaction $X^{\pm} \xrightarrow{deg} X^{\pm} + X^{\mp}$ together with an annihilation reaction $X^{\pm} + X^{\mp} \xrightarrow{ann} \varnothing$. In each case, standard degradation was compared with catalytic degradation for deg = 0.0008 s^{-1} and ann $\in$ {0.1,0.01,0.001,0.0001}; nM^{-1} s^{-1}. (A) For fast annihilation reactions, catalytic degradation accurately approximates standard degradation, with ann = 0.1 indistinguishable from standard degradation (not shown). However, the approximation breaks down as we approach ann $\approx$ deg. (B–C) Nevertheless, the correct behavior of the PI controller is still achieved using the catalytic degradation approximation.

inputs and no output (Figure 8A). The first strand displacement reaction takes a signal strand $\langle t\, x \rangle$ representing the species X and produces a reverse strand $\langle x\, t \rangle$ as an intermediate. Crucially, the reaction is reversible, so that if the complementary species X' is not present, the reverse reaction ensures that species X is not permanently consumed and is free to engage in other reactions. This enables all reactions to proceed with correct stoichiometry. The second strand displacement reaction takes a signal strand $\langle t\, x' \rangle$ representing the complementary species X' and produces inert waste. All reactions proceed at the same rate kt, which denotes the rate of strand displacement. The next two strand displacement reactions in Figure 8A implement the same annihilation reaction but with the order of species reversed, in that X' binds first, followed by X. This duplication ensures that the annihilation proceeds entirely symmetrically. This symmetry is needed because, although the first reaction is reversible, it still results in a fraction of species being consumed temporarily. Having the symmetric reaction ensures that equal fractions of

dx.doi.org/10.1021/sb400169s | *ACS Synth. Biol.* 2014, 3, 600–616

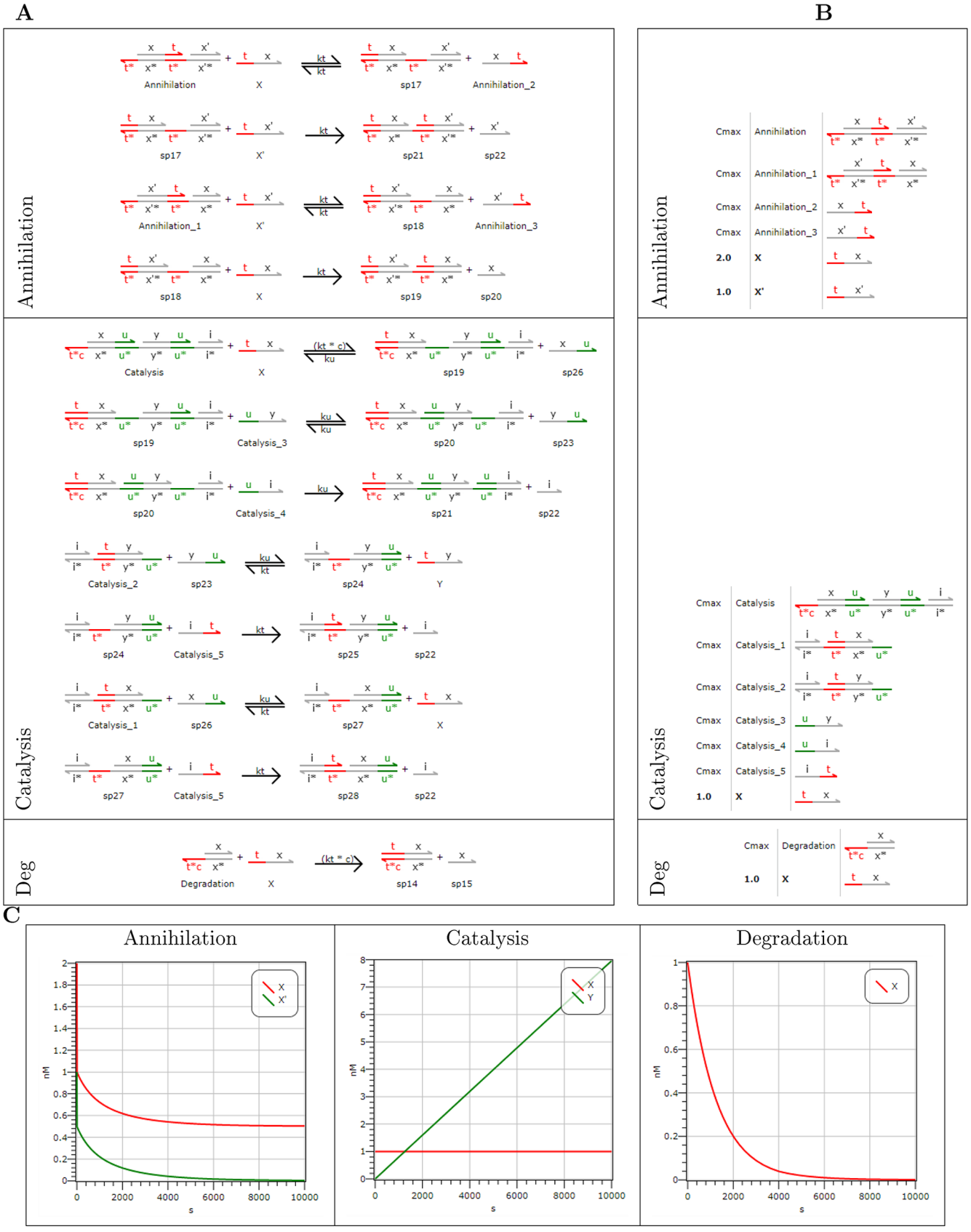


Figure 8. Two-domain strand displacement implementation of annihilation, catalysis, and degradation reactions. (A) The strand-displacement reactions implementing each ideal chemical reaction are generated automatically. (B) Initial concentrations of species used in nM, where C_{max} = 1000 nM. (C) Simulation results for each implementation. Rate constants $ku = kt = 0.001$ nM^{-1} s^{-1} and constant $c = 0.0008$ were used for all simulations.

dx.doi.org/10.1021/sb400169s | *ACS Synth. Biol.* 2014, 3, 600−616

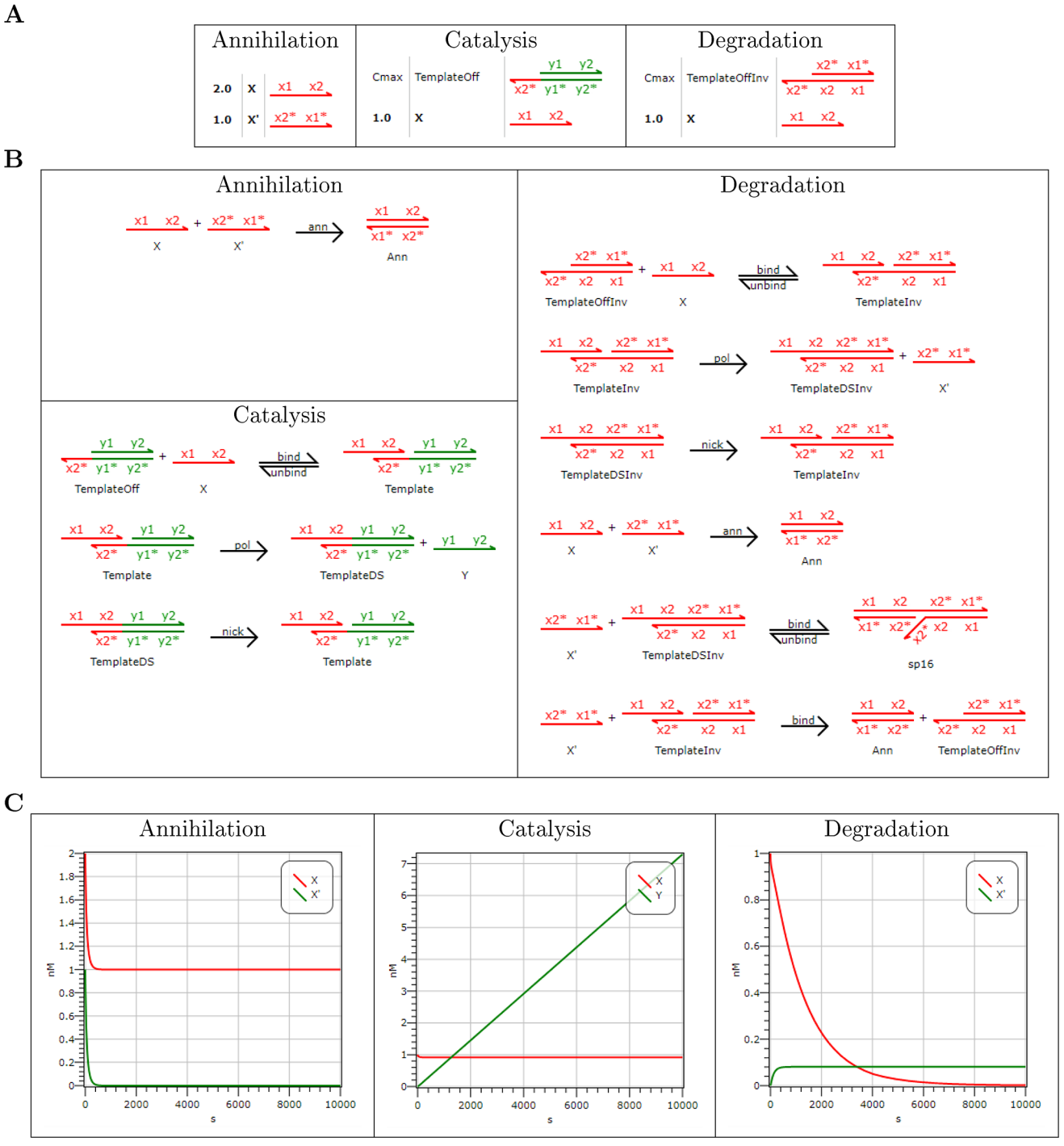


Figure 9. DNA enzyme implementations of the high-level reactions for annihilation, catalysis, and degradation. (A) Initial concentrations and names for each species, with concentrations expressed in nM. (B) Low-level reactions for each implementation. We assume that polymerase and nicking enzymes are in excess with approximately constant concentrations,[15] such that rate constants are first order with *pol* = *nick* = 1 min^{-1}. (C) Simulation results for each implementation. Rate constants ann = 0.01 nM^{-1} s^{-1}, bind = 5.4×10^{-6} nM^{-1} s^{-1}, unbind = 0.1126 s^{-1}, and initial conditions C_{max} = 1000 nM were used for all simulations.

species *X* and *X*′ are bound simultaneously. The initial conditions of the reaction indicate equal initial concentrations of gates and reverse strands, given by C_{max} = 1000 nM, which act as the chemical "fuel" driving the computation. The concentration of gates is high to ensure that annihilation is as fast as possible, and the concentration of reverse strands is also high to ensure that only a fraction of the species (in this case 50%) remains bound at any given time. The fraction can be reduced by increasing the concentration of reverse strands; however, this also has the effect of slowing down the annihilation reaction. To compensate for the fast initial equilibration, the initial value of each input signal involving an annihilation reaction was therefore increased by a factor of 2.

Catalysis. For the catalysis reaction $X \xrightarrow{cat} X + Y$ there are two products, so a Fork circuit is needed. Here, we have optimized the Fork circuit by allowing each product to be produced by a separate fork gate, in this case, one for *X* and one for *Y*. In the

dx.doi.org/10.1021/sb400169s | *ACS Synth. Biol.* 2014, 3, 600−616

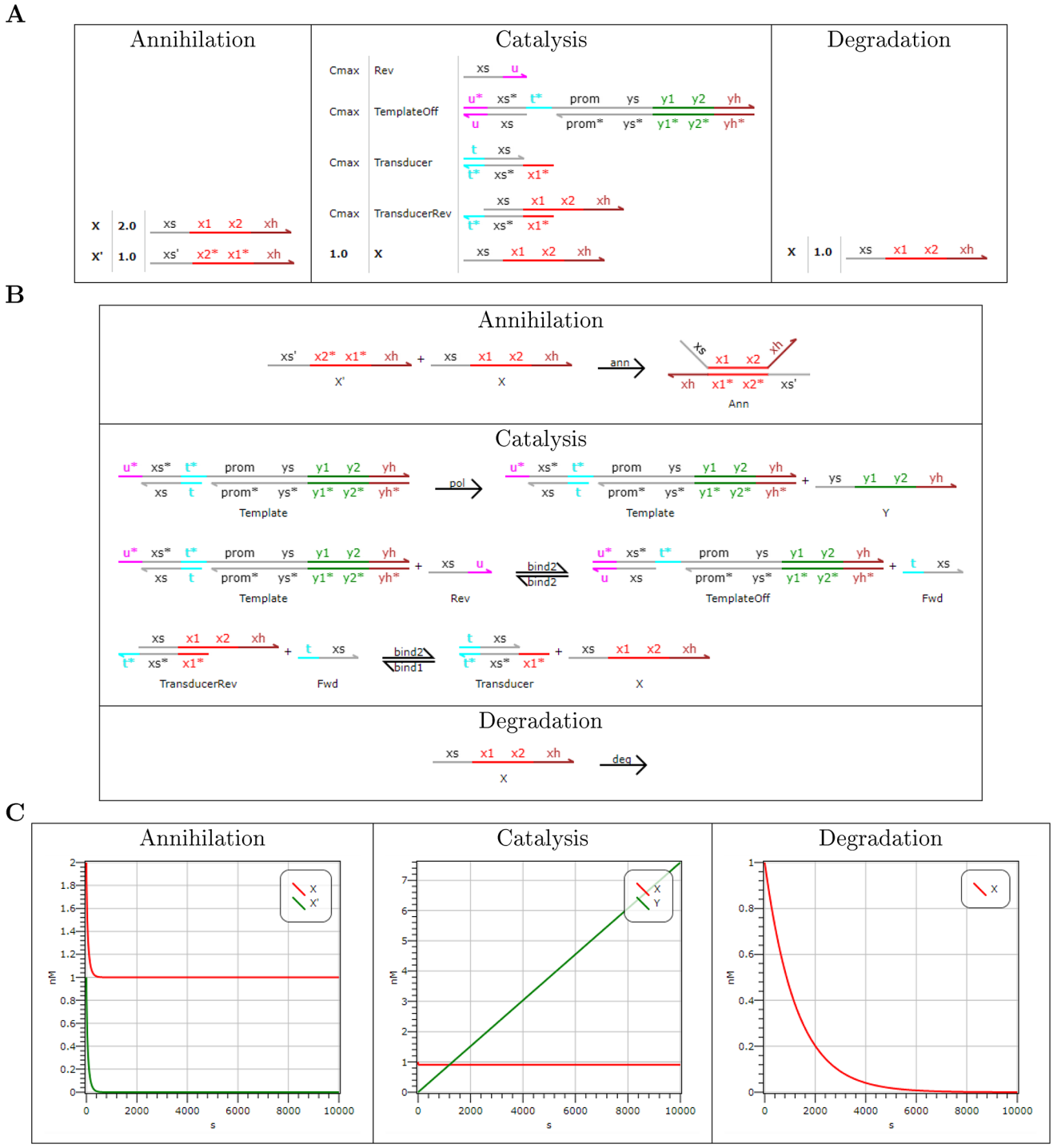


Figure 10. RNA Enzyme implementations of the high-level reactions for annihilation, catalysis, and degradation. (A) Initial concentrations and names for each species, with concentrations expressed in nM. (B) Low-level reactions for each implementation. We assume that polymerase enzymes are in excess with approximately constant concentrations,[16] such that rate constants are first order with pol = 1 min^{-1}. Similarly, we assume that degradation is first order but that the concentration of enzyme is adjusted for a rate constant of deg = 0.0008 s^{-1}.[37] (C) Simulation results for each implementation. Rate constants ann = 0.01 nM^{-1} s^{-1}, $bind_1$ = 0.001 nM^{-1} s^{-1}, $bind_2$ = 0.00005 nM^{-1} s^{-1}, unbind = 0.1126 s^{-1}, and initial conditions C_{max} = 1000 nM were used for all simulations.

original two domain scheme[38] each combination of products was produced sequentially using a single fork gate, resulting in a single gate to produce *X* and *Y* sequentially. However, in the case of the PI controller, no two reactions have the same combination of products (Figure 6A), which means that a separate fork gate would be needed for each reaction. By having a separate gate for each individual product, it is possible to share gates between reactions that share even a single product. For instance, in the PI controller circuit (Figure 6A), without considering the plant reactions, species $X^{\pm}$ and $V^{\pm}$ are the products of two reactions, while species $E^{\pm}$ is the product of four reactions. To achieve this optimization, an additional toehold *u* is used to trigger the release of a product. The first strand displacement reaction in Figure 8A consumes species $\langle t\ x \rangle$ and produces a transducer strand $\langle x\ u \rangle$.

dx.doi.org/10.1021/sb400169s | *ACS Synth. Biol.* 2014, 3, 600−616

The second reaction consumes a helper strand $\langle u\ y\rangle$ and produces a transducer strand $\langle y\ u\rangle$, and the third reaction consumes a helper strand $\langle u\ i\rangle$ to seal the Join gate. This acts as a garbage-collection step, by covering up the exposed u toehold to prevent spurious interactions and render the used Join gate effectively inert. The remaining strand displacement reactions in Figure 8A involve the transducers $\langle x\ u\rangle$ and $\langle y\ u\rangle$ binding to their respective Fork gates to produce the species $\langle t\ x\rangle$ and $\langle t\ y\rangle$, respectively. The used Fork gates are then sealed by strands $\langle i\ t\rangle$ to prevent any subsequent rebinding of the product. Crucially, once the catalyst binds to the Join gate, all of the subsequent steps happen very quickly, at rate $kt = ku$. We modify the degree of complementarity c of the t^* toehold so that the first strand displacement step occurs more slowly at rate $kt \times c$. Such mismatches can give rise to significantly lower rates of strand displacement, with a range of over 4 orders of magnitude.[32] Another point to note is that the catalyst X is consumed and a new catalyst X produced, rather than the original catalyst simply unbinding, which has a potential advantage in that *retroactivity*—a term referring to the influence of downstream processes on upstream ones[36,39]—can be significantly reduced between catalytic circuits. In particular, once the initial catalyst has become bound, a new catalyst can be quickly produced at the same time as the product, without waiting for the original catalyst to unbind. This allows the rate of catalysis to remain linear, provided there is an excess of catalytic gates.

Degradation. For the degradation reaction $X \xrightarrow{\text{deg}} \varnothing$ again there are no products, so the reaction is encoded as a Join circuit with only a single input and no output. Since the rate of degradation is meant to be significantly slower than annihilation, we multiply the strand displacement rate by a scaling factor c, representing the degree of complementarity of the toehold domains.

DNA Enzyme Implementation. We implemented the idealized PI controller as a DNA enzyme system. We based our implementation on the DNA Toolbox approach,[16] but introduced a number of important modifications, as summarized below.

Annihilation. An important consideration was the need to implement fast and irreversible annihilation reactions. Since the DNA Toolbox does not enable bimolecular reactions to be implemented directly, a custom implementation of these reactions was required. We anticipated that a generalized implementation in terms of the elementary DNA Toolbox operations of catalysis, inhibition, and degradation would result in significant overhead, and therefore, we developed a custom implementation that was specific to annihilation reactions. We chose perhaps one of the simplest possible approaches, by implementing complementary signals directly as complementary sequences (Figure 9). This implementation required the length of signals to be considerably greater than in the original DNA Toolbox approach,[15] in order to ensure irreversible binding of complementary signals at the operating temperature of the experiment.

Catalysis. An additional constraint was the need for signals to bind reversibly to templates, in order to prevent the permanent consumption of signals during catalysis. To achieve this, we let each signal X consist of two shorter sequences x_1x_2, such that the sequence x_2 binds reversibly to the template, while the full sequence x_1x_2 binds irreversibly to its complement (Figure 9). One effect of this increased signal length was that the product of catalysis no longer unbinds spontaneously from the template, but instead needs to be displaced by the polymerase. Since the specific polymerase used in the DNA Toolbox approach (e.g., Bst polymerase) can displace a bound strand without being significantly impeded,[15] we hypothesized that irreversible binding of the product to the template would not significantly affect system performance. As a result, all templates are assumed to be bound to their product strand at all times. We also assumed that the nicking recognition site on the product $y = y_1y_2$ overlaps both the y_1 and y_2 domains, and produces a nick between x_2 and y_1. Similarly, the nicking recognition site for $x = x_1x_2$ overlaps both the x_1 and x_2 domains.

Degradation. A final constraint was the need for sequence-specific degradation. Since the DNA Toolbox degrades all single strands indiscriminately by means of exonucleases, an alternative implementation of degradation was required. While it is possible to achieve sequence-specific degradation by chemical modification of DNA strands, such modifications are problematic in cases where the strands are being continually produced. We therefore implemented a degradation reaction in terms of the elementary reactions of catalysis and annihilation (Figure 9), based on our analysis in Figure 7 and the catalytic degradation strategy we proposed. Note that the nicking recognition site for $x^* = x_2^*x_1^*$ overlaps both the x_2^* and x_1^* domains, and produces a nick between x_2 and x_2^*.

RNA Enzyme Implementation. We implemented the idealized PI controller as an RNA enzyme system. We based our implementation on the genelet approach,[16] but introduced a number of important modifications, as summarized below.

Annihilation. Similar to our DNA enzyme implementation, we implemented annihilation of two signals by hybridization of complementary domains (Figure 10A). We encode a signal X using two domains, $\langle xs\ x\rangle$, with the complementary signal X' encoded as $\langle xs'\ x^*\rangle$, with xs and xs' as distinct sequences and $x = x_1x_2$. This allowed complementary signals to annihilate each other via direct hybridization, while also allowing them to take part in independent strand displacement reactions. Moreover, since signals are a product of transcription they consist of RNA strands rather than DNA strands.

Catalysis. Catalysis is implemented by means of genelet activation, which requires completion of a single-stranded promoter region (Figure 10B). Two main challenges needed to be overcome in order to achieve signal-specific catalysis. First, it was necessary to ensure that the catalyst binds reversibly to the genelet, so that it is not consumed. Second, the sequence of the catalyst needed to be a subsequence of the promoter region. This presented a conflict, since reversible binding required short sequences, while promoter completion required the presence of a specific promoter sequence. To reconcile these conflicting goals, we introduced a translator gate that took an RNA signal $\langle xs\ x\rangle$ as input and produced an intermediate DNA strand $\langle t\ xs\rangle$ as output, which contained the missing promoter sequence represented by the toehold t = *TATTA*. By using the same toehold t in all intermediate strands, a single promoter could be used for all of the genelets. The translator gate also ensured that the promoter was completed by a DNA strand rather than an RNA transcript, which circumvents potential issues related to additional bases that may result from nonspecific extension of RNA transcripts.[16] In order to ensure that only a fraction of the signal $\langle xs\ x\rangle$ was translated to intermediates, the toehold x_1^* on the translator gate was chosen to only complement a portion of the sequence $x = x_1x_2$, thus enabling toehold unbinding. Furthermore, an initial concentration of reverse translator gates was introduced, which take the intermediate strand $\langle t\ xs\rangle$ as input and produce the

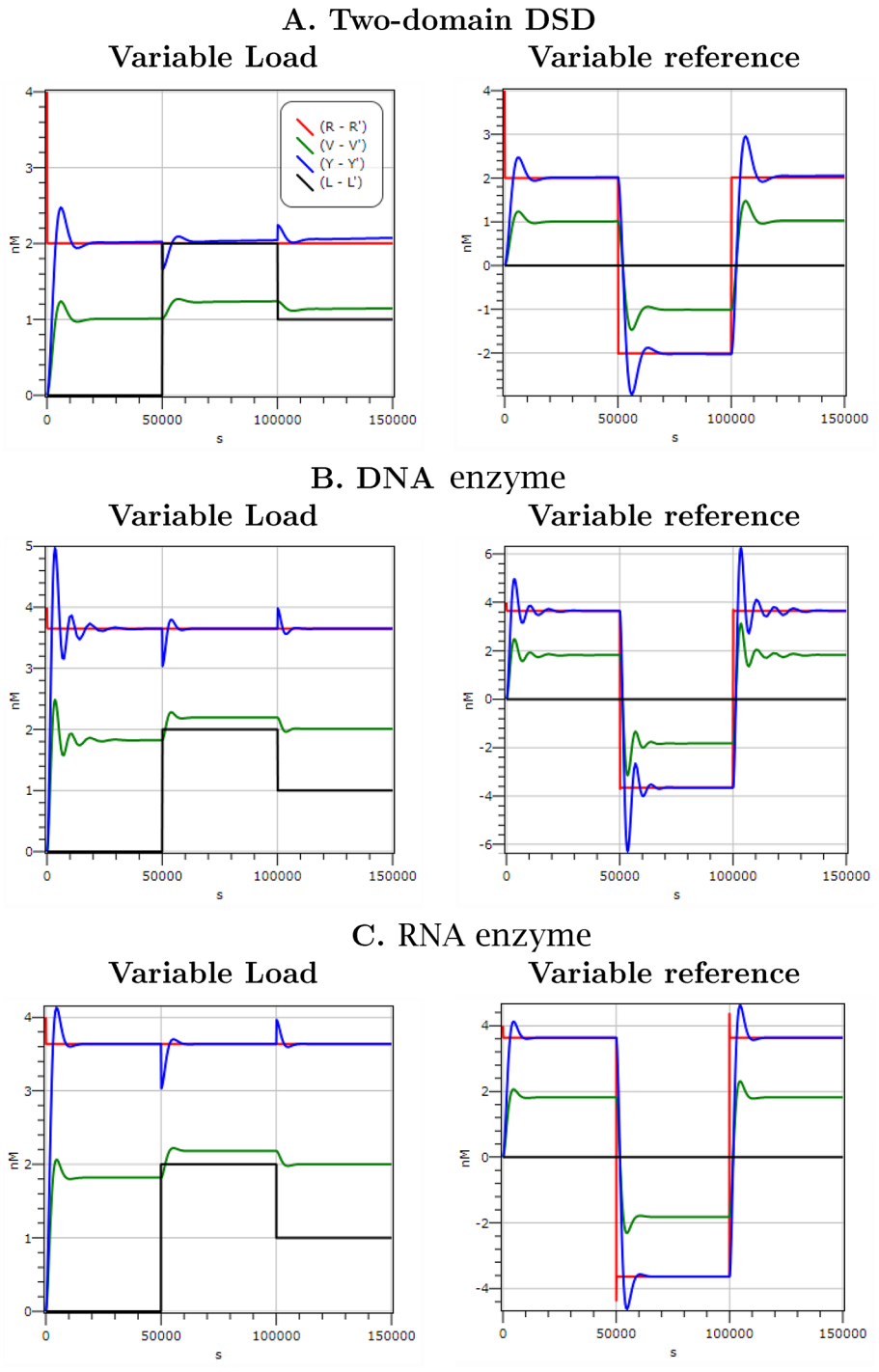


Figure 11. Comparison of PI controller designs. For all mechanisms, a plant implemented using ideal chemical reactions (as in Figure 6) was coupled to a PI controller implemented using (A) DNA strand displacement, (B) DNA enzyme, and (C) RNA enzyme approaches. The difference between the concentrations of positive and negative species are plotted for reference (red), controller output/plant input (green), plant output (blue), and load (black) signals. Simulation events were used to trigger the changes in the reference signal and load at predefined times.

signal $\langle xs\ x \rangle$. To compensate for the stronger binding of DNA-RNA hybrids relative to DNA−DNA molecules with the same sequence, we assume that the toehold lengths can be adjusted accordingly to give rise to a toehold exchange reaction with the appropriate kinetics. The second reaction in Figure 10B represents the reversible binding of the intermediate strand to the genelet, with toehold t completing the promoter region. The first reaction represents transcription of the product signal $\langle ys\ y \rangle$.

Degradation. For signal degradation, we chose to use the exonuclease RNase R, which is a 3′ to 5′ exonuclease.[37] Crucially, the activity of this enzyme is significantly reduced in the presence of secondary structure on the 3′ end, which limits the available single-stranded region for initiation of degradation.[40] The use of a translator gate to convert a fraction of an RNA signal to an intermediate DNA signal enabled us to exploit this property of RNase R to achieve sequence-specific degradation. For each signal of the form $\langle xs\ x \rangle$ that needed to be protected from

dx.doi.org/10.1021/sb400169s | *ACS Synth. Biol.* 2014, 3, 600−616

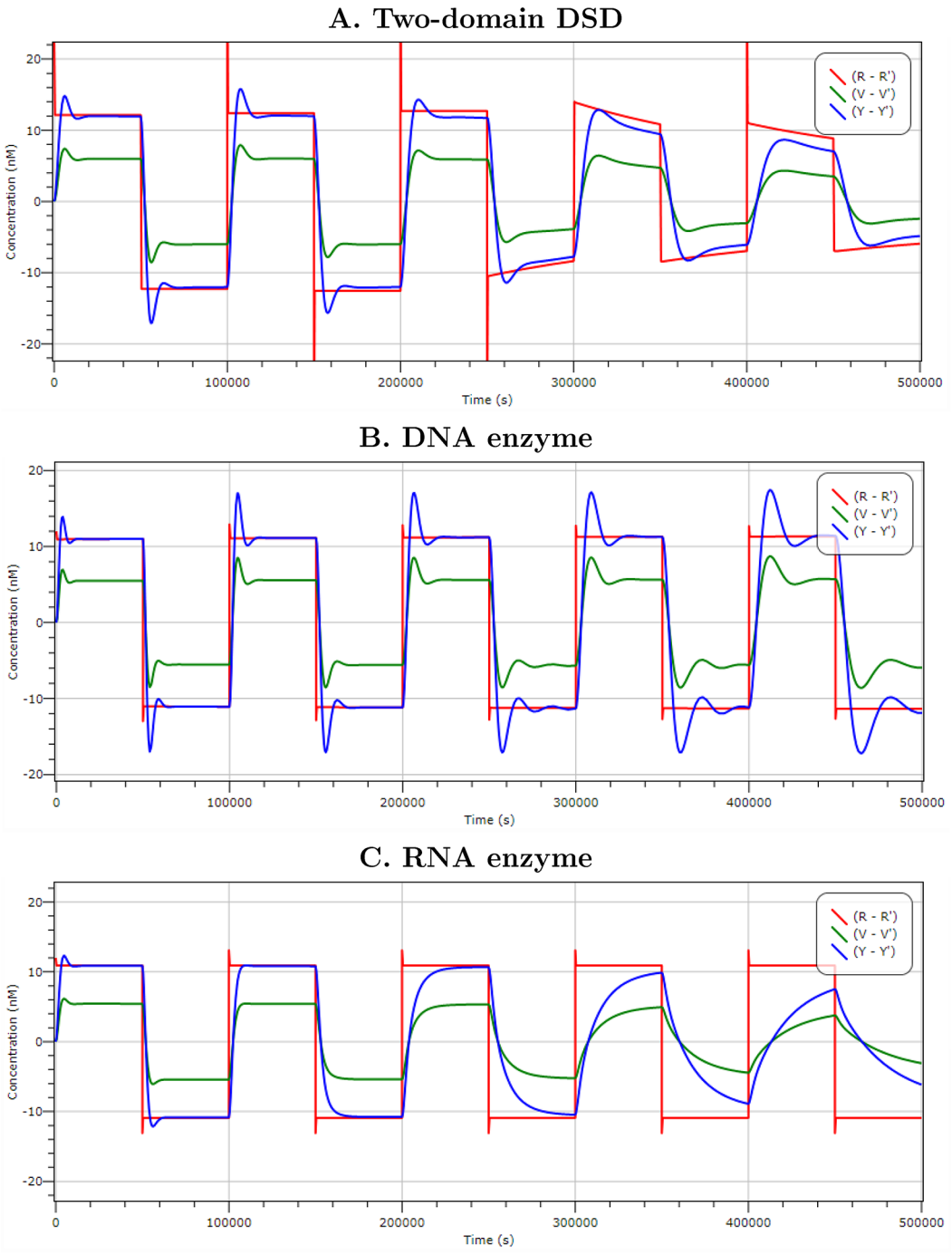


Figure 12. Long-term performance of PI controllers implemented using (A) DNA strand displacement, (B) DNA enzyme and (C) RNA enzyme approaches, when coupled to a plant implemented using ideal chemical reactions (as in Figure 6). An initial pool of 10.0 μM dNTPs (or NTPs) which are consumed through polymerase extension reactions were introduced in the DNA and RNA enzyme implementation designs to account for the consumption of resources. The difference between the concentrations of positive and negative species are plotted for the reference (red), controller output/plant input (green) and plant output (blue) signals. Simulation events were used to trigger the changes in the reference signal at predefined times. For the RNA enzyme design, the plant output drifts away from the reference signal. This takes longer to converge for the DNA enzyme design as resources are consumed, while for the DNA strand displacement design the reference signal could not be accurately modulated over the course of the experiment.

degradation, the sequence *x* could be extended with a tail domain *xh* that folds into a secondary structure at the 3′ end, resulting in a strand $\langle xs\ x\ xh \rangle$.[37]

Comparison of Implementation Strategies. We studied three designs of PI controllers connected to a production plant as in Figure 1 (Figure 11, Supporting Information). Each design was an implementation of the ideal chemical reaction system from Figure 6, constructed using the annihilation, catalysis, and degradation components of the DNA strand displacement, DNA enzyme, and RNA enzyme approaches we illustrated in Figures 8−10. We perturbed each system as in Figure 6, by changing the load on the plant and by varying the reference signal over time. In all cases, the controllers automatically adjusted the plant input to ensure that the plant output continued to match the reference. As

dx.doi.org/10.1021/sb400169s | *ACS Synth. Biol.* 2014, 3, 600−616

Table 1. Comparison between DNA Strand Displacement, DNA Enzyme, and RNA Enzyme Design Strategies for the Implementation of a PI Controller Circuit, Expressed As a System of Chemical Reactions

	Advantages	Challenges
DNA strand displacement	*DNA signals can consist of multiple domains and can be relatively long, thus reducing interference between signals. *Toehold-mediated strand displacement can be used to limit interference between components, allowing systems to scale to large numbers of molecular species and reactions. *Catalytic reactions consume the catalyst and quickly produce both catalyst and product, limiting retroactivity between components. *No additional enzymes are required: the entire system can be constructed from DNA.	*Energy is obtained by converting active strands and gates to inert waste, meaning that strands and gates need to be continually supplied for computation to be sustained over long periods. This is potentially challenging to do in a cellular context. *The lack of enzymes for degradation or production of strands means that all computations need to be implemented as strand displacement reactions. This increases the number of components that are needed. *Annihilation rates are limited by the rates of toehold mediated strand displacement, which are generally slower than rates of DNA hybridization.
DNA enzymes	*Most reactions are implemented by enzymes which operate with very high efficiency. *The use of multiple categories of enzymes means that complex systems can be designed with only a small number of strands. *The dynamic behavior of systems is relatively well understood	*Additional enzymes need to be provided for the system to function. *Signals are represented as single strands, which need to be relatively short so that they can bind reversibly to a template. As the number of species increases, this could potentially result in cross-talk between species. *Nicking enzymes place additional sequence constraints on signals. *Bimolecular interactions are not directly expressible and need to be encoded in terms of catalysis and inhibition. Alternatively, custom solutions can be tailored to a particular system design, such as mechanisms for strand annihilation. *Enzymes such as polymerase and exonuclease are not sequence specific, which constrains system design.
RNA enzymes	*The use of transcriptional machinery and enzymes means that computations can be designed with relatively small numbers of strands. *The use of RNA enables more flexibility in the design, such as using enzymes that enable sequence-specific degradation of RNA. *DNA signals can consist of multiple domains and can be relatively long, resulting in reduced signal interference. *Hybrid designs involving strand displacement, transcriptional regulation and enzymatic processing can be combined. This gives a high degree of design flexibility.	*Additional transcriptional machinery needs to be provided, which complicates the experimental setup. *The additional complexity means that potential sources of interference are less well-understood. *Bimolecular interactions are not directly expressible and custom mechanisms for strand annihilation are required.

discussed previously, for the DNA Strand displacement implementation, the reference signal was scaled by a factor of 2 in order to attenuate the 50% sequestering effect of the annihilation reaction. We also tested the performance of the controller over extended periods of time and for more extreme variations in reference signal (Figure 12). While the consumption of chemical fuel in the form of DNA strands was captured in our implementation of a DNA strand displacement PI controller, initially, the consumption of resources such as dNTPs (or NTPs) was not explicitly modeled in the implementations using DNA and RNA enzymes. In order to compare the behavior of these systems during long running computations, we introduced an initial pool of dNTPs (or NTPs), which were consumed through polymerase extension reactions in the DNA and RNA enzyme implementations (Figure 12). Under such conditions, we observed a drift away from the reference signal as time increased in the RNA enzyme implementation and a slower settling time in the DNA enzyme implementation, due to the consumption of the finite dNTP resources. For the DNA strand displacement implementation, even the reference signal could not be modulated indefinitely due to the consumption of DNA fuel strands, particularly for annihilation reactions between complementary signals.

Several advantages and challenges for constructing molecular circuits using the DNA strand displacement, DNA enzyme, and RNA enzyme approaches were observed by comparing the corresponding PI controller implementations (see Table 1 for a summary). Our comparison is based on the particular case studies discussed in this paper; however, a number of the identified properties could apply more generally to circuits constructed using the different implementation approaches. In addition, some of the outlined disadvantages could potentially be circumvented using new design strategies not considered here.

In terms of implementation complexity, DNA strand displacement circuits can be constructed entirely from DNA molecules, while circuits designed with the other approaches require additional enzymes. However, by introducing additional biochemical machinery, complex computational systems become realizable with only a relatively small number of distinct DNA strands. Toehold-mediated strand displacement, together with

dx.doi.org/10.1021/sb400169s | ACS Synth. Biol. 2014, 3, 600−616

flexible designs of signal sequences, can be used to manage component interference and allow the implementation of large DNA strand displacement circuits.[29] In contrast, additional sequence constraints are required when nickase recognition sites or partial promoter sequences are considered as part of the DNA and RNA enzyme approaches. Furthermore, certain enzymatic reactions used as part of these approaches, including the polymerase extension of primed templates and the nuclease degradation of signals, are not sequence specific and additional strategies are required to protect specific signals from participating in such reactions.

Certain retroactivity effects[39] were observed in the PI controllers implemented using the two enzymatic mechanisms, for example resulting in the reference signal offset in Figure 11B–C, and became more pronounced if faster circuit dynamics were implemented through increased template concentrations for catalysis reactions (see Figure 9B and Figure 10B). In contrast, the DNA strand displacement implementation did not suffer from similar retroactivity issues, due to the fast production of both catalyst and output signal strands as part of catalysis reactions (see Figure 8A). Finally, the difficulty of implementing bimolecular reactions, which were not directly expressible in the original DNA Toolbox and genelets approaches, prompted the development of the strategies we proposed, for example to achieve fast annihilation of signals. This allowed the implementation of the circuits we considered but resulted in the production of nonreactive (waste) DNA and RNA species, which were not degraded or reused in subsequent computations. The production of such waste is common in DNA strand displacement circuits, but our proposed modification to the DNA toolbox and genelets approaches also introduced it as a feature of DNA and RNA enzyme circuits. As a result, additional considerations are required to manage the buildup of waste in order to achieve accurate longer running computations.

DISCUSSION

The results presented in this paper are representative of an increased focus in the use of abstraction for the design of molecular systems. In building a design theory for molecular programming, chemical reaction networks are a natural intermediate representation, in the middle of the "hourglass"[41] between higher-level design requirements and lower-level implementations. Many different high-level languages and formalisms have been compiled to idealized chemical reactions, which are in turn implementable using a variety of low level, physically realizable molecular mechanisms, such as the strand displacement or enzymatic implementations discussed here. In some cases, an equivalence between idealized and low-level reaction systems can be proven formally.[42−44]

The biochemical implementation of complex signal processing and control circuits requires the development of scalable, modular design strategies and accompanying computational tools. A number of tools exist for modeling the molecular conformations of nucleic acid strands, including NUPACK,[45] OxDNA[46] and Multistrand,[47] which can inform the design of nucleic acid sequences for use in computational circuits. A number of software tools also exist for design at the circuit level, in particular for circuits expressed in terms of DNA strand displacement, DNA Toolbox and genelet mechanisms. Design tools include Visual DSD,[27,31,48] for modeling a broad class of strand displacement systems, together with the DNA Toolbox software.[49] Custom solutions have also been implemented, including a compiler for translating an arbitrary feed forward digital logic circuit to a strand displacement system.[29] Models have also been developed using general-purpose modeling environments such as Mathematica or Matlab, for example to model oscillator systems constructed from genelets.[16]

To design and model the circuits discussed in this paper, we chose to extend the DSD programming language,[31,48] which previously supported only DNA strand displacement devices. The DSD compiler[27] automatically generates the set of all possible reactions between chemical species based on the rules of the extended DSD language, which simplifies the design process and identifies potential interferences that are often hard to find manually for complicated devices. In addition, several Visual DSD abstractions,[48] such as the representation of DNA sequences as abstract domains and the use of parametrized modules, extend naturally beyond DNA strand displacement systems. The extensions to the computational methods we presented in this paper allowed us to apply these abstractions, together with the visualization, simulation and analysis methods available within Visual DSD, to more general nucleic acid circuits including ones built using the DNA Toolbox[15] and genelet approaches[16] (see Supporting Information for details), as well as the modified enzymatic strategies we proposed in this paper. We used this novel functionality to concisely encode models of a PI controller designed using the DNA strand displacement, DNA enzyme and RNA enzyme approaches we discussed in this paper and compare them within a common simulation and modeling framework.

Our results indicate that the three molecular programming strategies offer different advantages. The two enzymatic mechanisms suffered from retroactivity effects,[39] where the load from a downstream component influenced the upstream signal as a result of the binding between signal and template species. In the PI controller system this effect manifested as a decrease in the reference signal R from its intended value and became more pronounced if faster circuit dynamics were implemented through increased template concentrations. In other applications, similar retroactivity has been counteracted through the introduction of additional "insulation" circuit components,[36] leading to increased construction complexity. In contrast, the strand displacement implementation strategy for catalysis did not suffer from similar retroactivity issues, due to the fast production of both catalyst and output signal strand simultaneously.

Besides the effects of retroactivity, the ability of circuits to sustain computation over extended time periods was compared, using the PI controller as a case study. Our results indicate that, within the range of feasible strand and gate concentrations, the performance of the strand displacement system degrades over time (Figure 12A). This is not surprising, since computation in such enzyme-free implementations is driven by the consumption of DNA fuel, which is eventually depleted. When increasing the availability of fuel is no longer experimentally feasible, additional buffering mechanisms can be implemented to allow longer computations,[48] however this strategy leads to increased circuit complexity. A similar deterioration of the performance in controllers implemented using DNA or RNA enzymes was also observed, once the consumption of finite resources (in this case, the NTPs and dNTPs involved in polymerase extension reactions) was captured explicitly in the models (Figure 12B–C). Even so, since such resources were shared across all enzymatic reactions, their continuous addition could be more easily managed in experimental implementations, compared to

the signal-specific DNA fuel strands necessary for the strand displacement approach.

While the DNA and RNA enzyme approaches were comparable with respect to retroactivity they offer different advantages for experimental implementations. For example, the RNA enzyme approach allowed specific RNA species to be targeted for degradation by enzymes, while others were protected by secondary structure. In contrast, a direct DNA enzyme implementation of degradation based on exonuclease (as in the DNA Toolbox approach) affected all single-stranded DNA species. Some length dependence effects on exonuclease rates have been observed previously, for example where 10 base long DNA strands were degraded much slower (by a factor of 3) than 15 base long ones.[15] Thus, the length of DNA signal could be used as a tuning strategy with this approach, at the cost of additional constraints on the sequence designs. Instead, we overcome such limitations by proposing a strategy for the approximation of degradation reaction through catalytic degradation (a catalysis reaction together with an annihilation reaction between complementary signals). While this approximation required fast annihilation, spatial effects could potentially increase reaction rates, for instance, when a negative species is produced through catalysis in close proximity to the positive species, it may facilitate faster subsequent annihilation.

Our implementation of annihilation reactions through modifications of the original DNA Toolbox and genelet approaches was motivated by the challenges of realizing bimolecular reactions within those frameworks. While these modifications led to correct circuit behavior, they also resulted in the production of nonreactive "waste"—DNA and RNA species which are not degraded or reused in subsequent computations. The production of such waste is a common feature for circuits implemented using the DNA strand displacement approach, and has also been encountered in genelet and, to a lesser extent, DNA Toolbox designs. However, waste buildup might affect longer running computations and this problem is worsened in the proposed annihilation strategies for the DNA and RNA enzyme approaches, especially when the catalytic degradation strategy was used. Thus, for experimental implementations such waste products must be managed at low enough concentrations to minimize crosstalk.

Several future additions to the methodology and case studies reported in this paper are possible. For example, the Visual DSD extensions we proposed allowed us to express rich dynamical laws beyond mass action kinetics, for example to capture Michaelis−Menten enzyme kinetics or additional competition effects. Our current PI controller models involved first order reaction approximations, consistent with previous work in the field,[15,37] but detailed models capturing effects such as enzyme concentrations, processivity and competition[50] could enable the study of such systems in more general contexts. In this paper, we only considered the performance of the proposed controllers when coupled to a simple plant system implemented using ideal chemical reactions−the applicability of such circuits to realistic *in vivo* or *in vitro* biochemical processes will be investigated in the future as a strategy toward designing practical molecular control circuits. Furthermore, since the approaches we investigated are applicable to the engineering of a broad range of control and signal processing systems, our methodology could be used to design a range of systems for regulating biochemical processes, beyond the PI controllers we studied here. The design procedures and tools developed as part of this work are a step toward making the physical realization of such complex computational circuits more feasible in the near future. Planned future extensions to Visual DSD would also accommodate programmed, sequence-specific recognition of nicking enzymes as well as the automatic generation of all nonsequence specific reactions such as polymerase extension and nuclease degradation. In our experience, such automatic compilation has proven very useful for identifying all possible reactions, including ones that could be easily missed for complex circuits during the manual construction of models.

Finally, an important consideration for future work is the ability to implement DNA strand displacement, DNA enzyme and RNA enzyme control circuits in a cellular context. Chemical alternatives to DNA such as XNA[51] have been shown to be unaffected by the cellular degradation machinery, but have not yet been applied to the construction of strand displacement circuits. Furthermore, since some of the proposed strategies rely on enzymes that are typically present in a cellular context, more work is needed to prevent interference from other enzymes that may also be present.

■ ASSOCIATED CONTENT

Supporting Information

This material is available free of charge via the Internet at http://pubs.acs.org.

■ AUTHOR INFORMATION

Corresponding Authors

*Email: vkulkarn@umn.edu.

*Email: andrew.phillips@microsoft.com.

Notes

The authors declare no competing financial interest.

■ ACKNOWLEDGMENTS

V.V.K. and A.S. were supported, in part, by the National Science Foundation (NSF CAREER Award 0845650, NSF CCF 0946601, NSF CCF 1117168) and AFOSR. J.K. was supported by NSF Award 0832824 (The Molecular Programming Project). The authors thank Yannick Rondelez for his helpful suggestions and the anonymous reviewers for their feedback.

■ REFERENCES

(1) Tamsir, A., Tabor, J. J., and Voigt, C. A. (2011) Robust multicellular computing using genetically encoded NOR gates and chemical 'wires'. *Nature 469*, 212−215.

(2) Gardner, T. S., Cantor, C. R., and Collins, J. J. (2000) Construction of a genetic toggle switch in *Escherichia coli*. *Nature 403*, 339−342.

(3) Elowitz, M., and Leibler, S. (2000) A synthetic oscillatory network of transcriptional regulators. *Nature 403*, 335−338.

(4) Basu, S., Gerchman, Y., Collins, C. H., Arnold, F. H., and Weiss, R. (2005) A synthetic multicellular system for programmed pattern formation. *Nature 434*, 1130−1134.

(5) Sohka, T., Heins, R. A., Phelan, R. M., Greisler, J. M., Townsend, C. A., and Ostermeier, M. (2009) An externally tunable bacterial band-pass filter. *Proc. Natl. Acad. Sci. U.S.A. 106*, 10135−10140.

(6) Galloway, K. E., Franco, E., and Smolke, C. D. (2013) Dynamically reshaping signaling networks to program cell fate via genetic controllers. *Science 341*, 1235005.

(7) Hodgman, C. E., and Jewett, M. C. (2012) Cell-free synthetic biology: Thinking outside the cell. *Metabol. Eng. 14*, 261−9.

(8) Genot, A. J., Fujii, T., and Rondelez, Y. (2013) *In vitro* regulatory models for systems biology. *Biotechnol. Adv. 31*, 789−96.

(9) Billerbeck, S., Härle, J., and Panke, S. (2013) The good of two worlds: Increasing complexity in cell-free systems. *Curr. Opin. Biotechnol. 24*, 1037−43.

(10) Padirac, A., Fujii, T., and Rondelez, Y. (2013) Nucleic acids for the rational design of reaction circuits. *Curr. Opin. Biotechnol. 24*, 575−80.
(11) Delebecque, C. J., Lindner, A. B., Silver, P. A., and Aldaye, F. A. (2011) Organization of intracellular reactions with rationally designed RNA assemblies. *Science 333*, 470−474.
(12) Douglas, S. M., Bachelet, I., and Church, G. M. (2012) A logic-gated nanorobot for targeted transport of molecular payloads. *Science 335*, 831−834.
(13) Daniel, R., Rubens, J. R., Sarpeshkar, R., and Lu, T. K. (2013) Synthetic analog computation in living cells. *Nature 497*, 619−23.
(14) Zhang, D. Y., and Seelig, G. (2011) Dynamic DNA nanotechnology using strand-displacement reactions. *Nat. Chem. 3*, 103−13.
(15) Montagne, K., Plasson, R., Sakai, Y., Fujii, T., and Rondelez, Y. (2011) Programming an *in vitro* DNA oscillator using a molecular networking strategy. *Mol. Syst. Biol. 7*, 466.
(16) Kim, J., and Winfree, E. (2011) Synthetic *in vitro* transcriptional oscillators. *Mol. Syst. Biol. 7*, 465.
(17) Chen, Y.-J., Dalchau, N., Srinivas, N., Phillips, A., Cardelli, L., Soloveichik, D., and Seelig, G. (2013) Programmable chemical controllers made from DNA. *Nat. Nanotechnol. 8*, 755−762.
(18) Fujii, T., and Rondelez, Y. (2013) Predator−prey molecular ecosystems. *ACS Nano 7*, 27−34.
(19) Weitz, M., Kim, J., Kapsner, K., Winfree, E., Franco, E., and Simmel, F. C. (2014) Diversity in the dynamical behavior of a compartmentalized programmable biochemical oscillator. *Nat. Chem. 6*, 295−302.
(20) Franklin, G., Powell, F., and Emami-Naeini, A. (2010) *Feedback Control of Dynamic Systems*, 6th ed.; Pearson Education, Upper Saddle River, NJ.
(21) Kailath, T. (1980) *Linear System Theory*; Prentice Hall, Englewood, NJ.
(22) Åström, K. J., and Hägglund, T. (2005) *PID Controller: Theory, Design, and Tuning; ISA—The Instrumentation*; Systems, and Automation Society, Research Triangle Park.
(23) Xie, Z., Wroblewska, L., Prochazka, L., Weiss, R., and Benenson, Y. (2011) Multi-input RNAi-based logic circuit for identification of specific cancer cells. *Science 333*, 1307−1311.
(24) Franco, E., Giordano, G., Forsberg, P.-O., and Murray, R. M. (2014) Negative autoregulation matches production and demand in synthetic transcriptional networks. *ACS Synth. Biol.*, DOI: 10.1021/sb400157z.
(25) Oishi, K., and Klavins, E. (2011) Biomolecular implementation of linear I/O systems. *IET Syst. Biol. 5*, 252−260.
(26) Soloveichik, D., Seelig, G., and Winfree, E. (2010) DNA as a universal substrate for chemical kinetics. *Proc. Natl. Acad. Sci. U.S.A. 107*, 5393−5398.
(27) Lakin, M. R., Youssef, S., Polo, F., Emmott, S., and Phillips, A. (2011) Visual DSD: A design and analysis tool for DNA strand displacement systems. *Bioinformatics 27*, 3211−3.
(28) Zhang, D., Turberfield, A. J., Yurke, B., and Winfree, E. (2007) Engineering entropy-driven reactions and networks catalyzed by DNA. *Science 318*, 1121−1125.
(29) Qian, L., and Winfree, E. (2011) Scaling up digital circuit computation with DNA strand displacement cascades. *Science 332*, 1196−201.
(30) Qian, L., Winfree, E., and Bruck, J. (2011) Neural network computation with DNA strand displacement cascades. *Nature 475*, 368−72.
(31) Phillips, A., and Cardelli, L. (2009) A programming language for composable DNA circuits. *J. R. Soc., Interface 6* (Suppl 4), S419−36.
(32) Zhang, D. Y., and Winfree, E. (2009) Control of DNA strand displacement kinetics using toehold exchange. *J. Am. Chem. Soc. 131*, 17303−17314.
(33) Padirac, A., Fujii, T., and Rondelez, Y. (2012) Bottom-up construction of *in vitro* switchable memories. *Proc. Natl. Acad. Sci. U.S.A. 109*, E3212−3220.
(34) Kim, J., White, K. S., and Winfree, E. (2006) Construction of an *in vitro* bistable circuit from synthetic transcriptional switches. *Mol. Syst. Biol. 2*, 68.
(35) Subsoontorn, P., Kim, J., and Winfree, E. (2012) Ensemble Bayesian analysis of bistability in a synthetic transcriptional switch. *ACS Synth. Biol. 1*, 299−316.
(36) Franco, E., Friedrichs, E., Kim, J., Jungmann, R., Murray, R., Winfree, E., and Simmel, F. C. (2011) Timing molecular motion and production with a synthetic transcriptional clock. *Proc. Natl. Acad. Sci. U.S.A. 108*, E784−E793.
(37) Kim, J., Khetarpal, I., Sen, S., and Murray, R. (2014) Synthetic circuit for exact adaptation and fold-change detection. *Nucleic Acids Res.*, DOI: 10.1093/nar/gku233.
(38) Cardelli, L. (2013) Two-domain DNA strand displacement. *Math. Struct. Computer Sci. 23*, 247−271.
(39) Del Vecchio, D., Ninfa, A. J., and Sontag, E. D. (2008) Modular cell biology: Retroactivity and insulation. *Mol. Syst. Biol. 4*, 161.
(40) Vincent, H. A., and Deutscher, M. P. (2006) Substrate recognition and catalysis by the exoribonuclease RNase R. *J. Biol. Chem. 281*, 29769−29775.
(41) Doyle, J., and Csete, M. (2007) Rules of engagement. *Nature 446*, 860.
(42) Dong, Q. A bisimulation approach to verification of molecular implementations of formal chemical reaction networks. M.Sc. thesis, Stony Brook University, Stony Brook, NY, 2012.
(43) Shin, S. W. Compiling and verifying DNA-based chemical reaction network implementations. Master's thesis, California Institute of Technology, Pasadena, CA, 2012.
(44) Lakin, M. R., Phillips, A., and Stefanovic, D. (2013) Modular verification of DNA strand displacement networks via serializability analysis. *DNA*, 133−146.
(45) Zadeh, J. N., Steenberg, C. D., Bois, J. S., Wolfe, B. R., Pierce, M. B., Khan, A. R., Dirks, R. M., and Pierce, N. A. (2011) NUPACK: Analysis and design of nucleic acid systems. *J. Comput. Chem. 32*, 170−173.
(46) Srinivas, N., Ouldridge, T. E., Sulc, P., Schaeffer, J. M., Yurke, B., Louis, A. A., Doye, J. P. K., and Winfree, E. (2013) On the biophysics and kinetics of toehold-mediated DNA strand displacement. *Nucleic Acids Res.*, DOI: 10.1093/nar/gkt801.
(47) Schaeffer, J. Stochastic simulation of the kinetics of multiple interacting nucleic acid strands. Master's thesis, California Institute of Technology, Pasadena, CA, 2012.
(48) Lakin, M. R., Youssef, S., Cardelli, L., and Phillips, A. (2012) Abstractions for DNA circuit design. *J. R. Soc. Interface 9*, 470−86.
(49) Aubert, N., Mosca, C., Fujii, T., Hagiya, M., and Rondelez, Y. (2014) Computer-assisted design for scaling up systems based on DNA reaction networks. *J. R. Soc. Interface 11*, 20131167.
(50) Rondelez, Y. (2012) Competition for catalytic resources alters biological network dynamics. *Phys. Rev. Lett. 108*, 018102.
(51) Pinheiro, V. B., Taylor, A. I., Cozens, C., Abramov, M., Renders, M., Zhang, S., Chaput, J. C., Wengel, J., Peak-Chew, S.-Y., McLaughlin, S. H., Herdewijn, P., and Holliger, P. (2012) Synthetic genetic polymers capable of heredity and evolution. *Science 336*, 341−344.