International Journal of
Molecular Sciences

Article

Towards a Bioelectronic Computer: A Theoretical Study of a Multi-Layer Biomolecular Computing System That Can Process Electronic Inputs

Katherine E. Dunn *,†, Martin A. Trefzer, Steven Johnson and Andy M. Tyrrell

Department of Electronic Engineering, University of York, Heslington, York YO10 5DD, UK; martin.trefzer@york.ac.uk (M.A.T.); steven.johnson@york.ac.uk (S.J.); andy.tyrrell@york.ac.uk (A.M.T.)

* Correspondence: k.dunn@ed.ac.uk; Tel.: +44-131-650-4845

† Current address: School of Engineering, Institute for Bioengineering, University of Edinburgh, The King's Buildings, Edinburgh EH9 3DW, UK.

Received: 27 July 2018; Accepted: 3 September 2018; Published: 4 September 2018

Abstract: DNA molecular machines have great potential for use in computing systems. Since Adleman originally introduced the concept of DNA computing through his use of DNA strands to solve a Hamiltonian path problem, a range of DNA-based computing elements have been developed, including logic gates, neural networks, finite state machines (FSMs) and non-deterministic universal Turing machines. DNA molecular machines can be controlled using electrical signals and the state of DNA nanodevices can be measured using electrochemical means. However, to the best of our knowledge there has as yet been no demonstration of a fully integrated biomolecular computing system that has multiple levels of information processing capacity, can accept electronic inputs and is capable of independent operation. Here we address the question of how such a system could work. We present simulation results showing that such an integrated hybrid system could convert electrical impulses into biomolecular signals, perform logical operations and take a decision, storing its history. We also illustrate theoretically how the system might be able to control an autonomous robot navigating through a maze. Our results suggest that a system of the proposed type is technically possible but for practical applications significant advances would be required to increase its speed.

Keywords: DNA nanotechnology; molecular computing; bioelectronics

1. Introduction

The field of DNA nanotechnology was founded by Seeman, who was the first to suggest that the self-assembly of DNA strands could be used for the fabrication of nanostructures [1]. In 2006 the subject was revolutionized by Rothemund's introduction of DNA 'origami', which involves the folding of a long single-stranded DNA 'scaffold' into a designed shape through hybridization with a large number of 'staple' strands [2]. Applications of DNA origami include nanolithography [3] and the investigation of fundamental biological questions [4]. In general, many applications of DNA nanotechnology are related to the possibility of using DNA to organize components in a programmable manner with exquisite spatial precision [5], utilizing chemical functionalization or modification of individual molecules as required. This is known as DNA nanoarchitectonics [6]. Other applications of DNA nanotechnology include dynamic machines such as walkers [7–9] and novel biomaterials [10], including hydrogels [11]. Increasingly, DNA nanotechnology complements or is complemented by developments in biotechnology and synthetic biology. For example, DNA origami can now be produced on large scales using biotechnological methods [12] and origami nanostructures have been used to construct analogues of viruses [13].

DNA-based systems can also be used for computation. Adleman demonstrated in 1994 that a system of DNA molecules could solve the travelling salesman problem [14] and since then great progress has been made in DNA computing. A wide variety of DNA logic gates have been designed, with operating mechanisms including strand displacement [15,16] and DNAzyme action [17]. It has been shown that 'see-saw' gates [18] can be used for thresholding and the construction of complex DNA computing systems that are capable of performing arithmetic functions [19] or demonstrating memory and inference [20]. Other DNA computing paradigms include chemical reaction networks [21], finite state machines [22,23] or algorithmic self-assembly of tiles [24], and it was recently demonstrated that DNA could be used to implement a non-deterministic universal Turing machine [25].

A DNA-based computer could be preferable to a silicon computer under circumstances in which biomolecular inputs need to be processed in vivo, due to the biocompatibility of DNA. Furthermore, the power dissipated by DNA logic gates can be approximately three orders of magnitude less than that of their silicon counterparts [26] and hence a DNA computer could offer a significant advantage over a silicon computer in a situation in which it is critical to keep power dissipation as low as possible. Both biocompatibility and low power dissipation would be important in a device such as a medical implant that autonomously performs calculations and makes decisions about the release of pharmacologically-active agents inside the body. The capabilities of DNA computing machines could be broadened further via their integration with electronic systems, which could underpin new approaches for input, processing and readout [26].

It has already been demonstrated that DNA molecular machines can be controlled using electrical signals, using electrochemical control of pH and pH-sensitive structures such as i-motifs or triplexes [27,28] or electronically triggered release of a signaling species from a surface [29]. In an alternative architecture, an enzymatic logic gate under certain conditions produces NADH (the reduced form of the cofactor nicotinamide adenine dinucleotide), which activates an electrochemical process that leads to release of DNA from an alginate matrix [30]. For these studies, electrical triggers were used to supply inputs, but electrochemical effects can also be used for signal readout. For instance, electrochemical DNA (E-DNA) sensors consist of DNA nanoswitches functionalized with a redox active moiety that can transport charge to the surface of an electrode when the structure is in one configuration but not when it is in an alternative state [31]. It is also possible to use the electrochemical signal from intercalated electroactive molecules (such as methylene blue) to confirm whether DNA is single-stranded or hydrogen-bonded in a duplex, quadruplex etc. [32].

In most cases, these electrochemical systems have a single layer of information processing capacity. In other words, they simply yield a particular output in the presence of a stimulus. While such devices are extremely useful for biosensing applications, more complex systems are required for computing. In this paper we present a proof-of-concept theoretical study of a new type of biomolecular computer. Our goal was to design and theoretically validate a system with multiple layers of information processing capacity, based primarily on well-characterized DNA machinery, with the inputs being provided electronically. Establishing that such a system is technically possible lays the foundations for further developments in this area, effectively providing a first step towards the construction of practical bioelectronic computers. Ideally, such computers would benefit from the advantages of low power massively parallel molecular machinery, while offering ease of integration with traditional electronics.

In our design, we use a diverse range of DNA elements, from pH-sensitive nanoswitches to catalytic cycles and logic gates. Our simulations focus on the kinetics of the DNA machines, and as each operating cycle of our DNA-based computer takes 10 h, our results suggest that further advances will be required to build such a computer that functions at a practical speed. The design of the system is shown in Figure 1a. The information flow in one cycle is shown in Figure 1b, and the full operation is shown schematically in Figure 1c. The initial inputs to the system could be the signals from four electronic sensors (or two sensors with dual-rail logic). One and only one of the inputs is true. The identity of the true input is used to set the delay between voltage pulses that are used to electrochemically increase the pH and then return it to its original value.

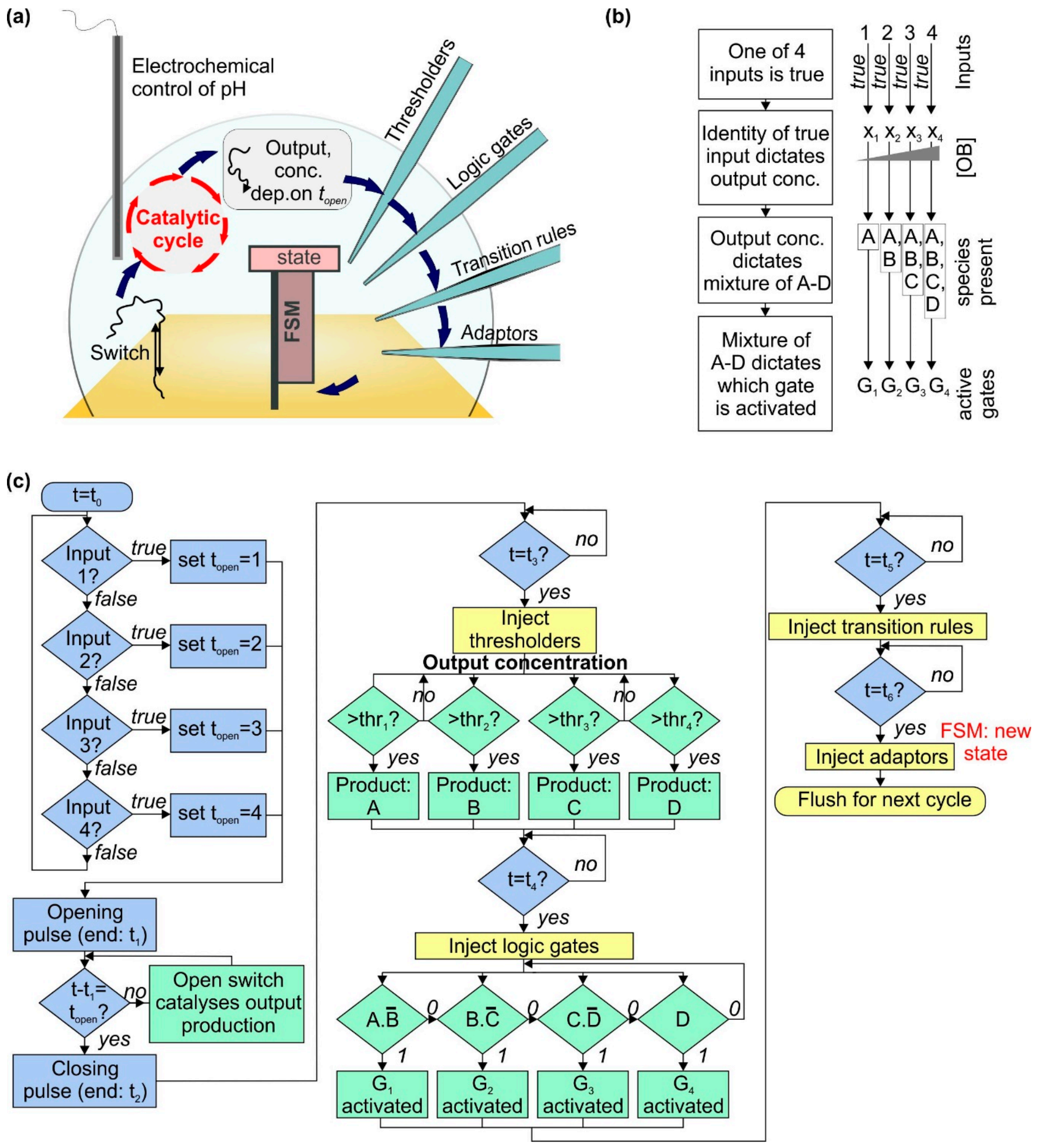

(a)
Electrochemical control of pH
Output, conc. dep.on t_{open}
Catalytic cycle
state
FSM
Switch
Thresholders
Logic gates
Transition rules
Adaptors
(b)
One of 4 inputs is true
Identity of true input dictates output conc.
Output conc. dictates mixture of A-D
Mixture of A-D dictates which gate is activated
1 2 3 4
true true true true
Inputs
X_1 X_2 X_3 X_4
[OB]
A | A, B | A, B, C | A, B, C, D
species present
G_1 G_2 G_3 G_4
active gates
(c)
$t=t_0$
Input 1? true set t_{open}=1 false
Input 2? true set t_{open}=2 false
Input 3? true set t_{open}=3 false
Input 4? true set t_{open}=4 false
Opening pulse (end: t_1)
$t-t_1=t_{open}$? no yes
Open switch catalyses output production
Closing pulse (end: t_2)
$t=t_3$? no yes
Inject thresholders
Output concentration
$>thr_1$? $>thr_2$? $>thr_3$? $>thr_4$? no yes
Product: A Product: B Product: C Product: D
$t=t_4$? no yes
Inject logic gates
$A.\bar{B}$ $B.\bar{C}$ $C.\bar{D}$ D 0 1
G_1 activated G_2 activated G_3 activated G_4 activated
$t=t_5$? no yes
Inject transition rules
$t=t_6$? no yes
FSM: new state
Inject adaptors
Flush for next cycle

Figure 1. Design for a bioelectronic processor that takes electronic inputs and makes decisions autonomously using biomolecular machines. (**a**) Schematic diagram of the system, described fully in the text. The switch is a pH-sensitive DNA nanomachine that changes conformation when the pH is altered electrochemically. Opening the switch reveals a sequence that acts as a catalyst for a cycle converting fuel and substrate molecules into waste product, signal and output. The concentration of the output is assessed using a thresholding mechanism, and the result is then assessed using logic gates. Activated logic gates act as inputs for the finite state machine (FSM), and transition rules contain instructions that specify the next state for the machine based on the previous state and the input; (**b**) flowchart showing how information is encoded in different molecular species within the system; (**c**) flowchart illustrating the mechanisms of system operation. The boxes are color-coded by type of process as follows: green—biomolecular, blue—electronic, yellow—microfluidic.

In our system, the voltage pulses cause conformational changes in pH-sensitive DNA nanoswitches, which are assumed to be based on a triplex [33] motif and could either float freely in solution or be immobilized on a surface. It has already been shown by Idili et al. (Ricci research group) that the pH at which triplex switches undergo the triplex-duplex transition can be tuned by the choice of sequence employed [34], which means that our design supports a range of operating conditions. Importantly, Amodio et al. (Ricci research group) have demonstrated that DNA triplex motifs can be used to enable control of strand displacement reactions with pH [35]. They also showed that clamp-like DNA triplexes could have a melting temperature as high as 82.3 °C. In their experiments on OH^- activated DNA strand displacement, the strand to be displaced was involved in a triplex and the toehold was located at the end of the construct. Their results revealed that the triplex binding was strong enough to block branch migration and prevent displacement even in this case, where the toehold was accessible (not hidden). In subsequent work Idili et al. (Ricci research group) demonstrated that a toehold could be effectively sequestered in a loop locked by a triplex [36], in an arrangement similar to that used here. In that paper, this mechanism allowed the use of pH to regulate a hybridization chain reaction.

For our architecture, as shown in Figure 1a, open DNA switches catalyze the operation of a cycle [37] that uses toehold-mediated strand displacement [38,39] reactions to convert DNA 'substrate' and 'fuel' molecules into 'output' molecules. Assuming there is only one species of switch present, and one set of catalytic molecules, it is not possible to assign one output to each input. Instead, the identity of the true input determines the quantity produced of the output, through the delay between opening and closing voltage pulses. The longer this delay, the longer the switch is open, and the more times the catalytic cycle occurs, leading to a higher concentration of output.

The concentration of output then determines what mixture of species A–D is left after the thresholder molecules have been injected and allowed to react. The combination of thresholder outputs A–D is assessed using injected logic gates G1–4. A–D react with the gates, activating or deactivating them according to the gate design. The parameters are set such that one gate is active at much higher concentration than the others. Activation of a gate enables it to act as the input for an ensemble of finite state machines immobilized on a surface. Injected transition rule molecules then determine the final state of the machine [22,23], based on the initial state and the input provided by the logic gates. Throughout this process, timing of events (i.e., the 'clock' of the system) is provided by the underlying electronics, which control the start time and duration of the initial voltage pulse, in addition to the timing of microfluidic injection of thresholders, logic gates, transition rules and adaptors. The latter molecules prepare the finite state machines for the next round of the loop. After each cycle the system must be flushed using microfluidic pumps as shown in Figure 1c. This will prevent accumulation of byproducts over many cycles. It might be necessary to stir the solution to ensure it remains well-mixed during each cycle.

2. Results

2.1. A Catalytic Cycle with an Electrochemically Controlled pH-Sensitive Nanoswitch as the Catalyst

The application of a voltage to a sufficiently-concentrated aqueous salt solution leads to electrolysis of water, with pH decreasing at the anode and increasing at the cathode. This effect is reversible, and can be used to modify the pH around an electrode by several units if desired. Other electrochemical reactions can be used in a similar way and it has already been shown that it is possible to control the conformation of pH-sensitive DNA structures electrochemically [27,28]. Depending on the pH of the solution, the nanoswitch is either open or closed. When the switch is closed, a short sequence is sequestered in the loop. Until the switch is opened, the rate of interaction between the hidden sequence and its complements is very low. The principle of concealing information in this way is extremely well-established [35,36,40].

When the switch is opened, the loop sequence can act as a toehold for downstream strand displacement reactions. However, for many applications it is undesirable to permanently inactivate switches by converting them into waste products, so it is necessary to employ a catalytic cycle in which the nanoswitch catalyst is regenerated. Various approaches could be employed, but the archetypal example of a catalytic cycle is that demonstrated by Zhang et al. [37]. This cycle has been characterized thoroughly and reliable models exist for the kinetics of all reactions in solution. Consequently, this is the cycle we choose for our proposed system, and we replace the original linear catalyst strand with the pH-sensitive nanoswitch (Figure 2a). We assume that the nanoswitch is based on a design of Idili et al. [34], and undergoes a transition at pH 7.5. We use the solution-phase kinetic parameters.

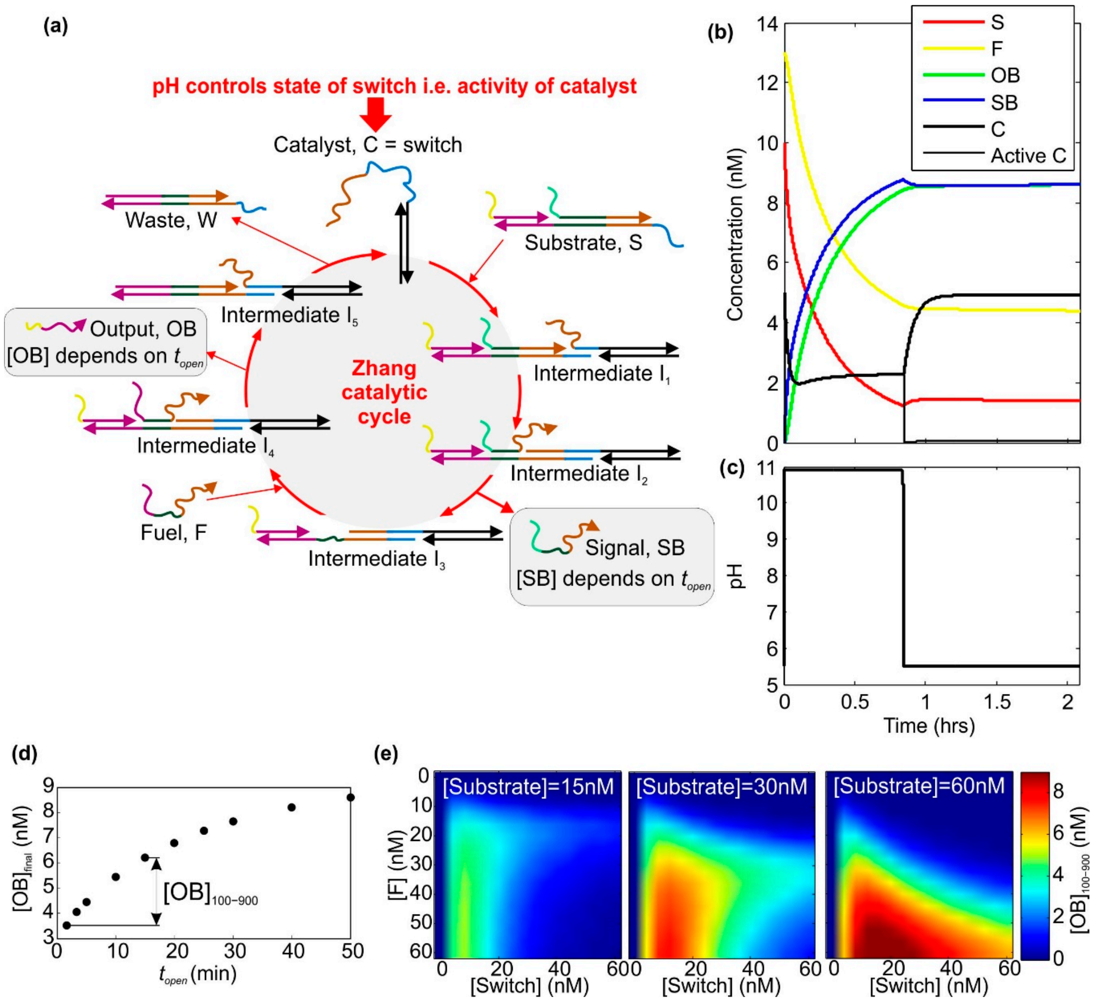


Figure 2. The Zhang catalytic cycle with a pH-sensitive nanoswitch as the catalyst. (**a**) Schematic diagram of the Zhang cycle, based on Figure 1b from Reference [37] (adapted with permission from the American Association for the Advancement of Science, AAAS), with the linear catalyst replaced by a nanoswitch based on the design of Idili et al. [34]; (**b**) simulated dynamics, computed using the reaction rates and information provided by Zhang et al. [37]. Here, the initial concentrations are: substrate—10 nM, fuel—13 nM, switch—5 nM. For the case illustrated, the separation of the voltage pulses is 3000 s, which means that the switch remains open for approximately 50 min. The voltage pulses lasted for 20 s, and had an amplitude of 4 V. The reaction volume was 100 µL; (**c**) the pH dynamics corresponding to the simulation in part (**b**); (**d**) the range of $[OB]_{final}$ values accessible for particular values of t_{open}, where t_{open} is assumed to be equal to the temporal separation of opening and closing voltage pulses. The concentrations and voltage pulse characteristics here are the same as for part (**b**,**c**); (**e**) illustration that the parameters are inter-dependent. $[OB]_{100\text{-}900}$ is defined as shown in part (**d**), and represents the difference in $[OB]_{final}$ values obtained for t_{open} = 100 s and t_{open} = 900 s. A large $[OB]_{100–900}$ value implies a steep curve in the plot shown in (**d**). Note: throughout this paper, the concentration of a species, 'X', is denoted using square brackets, i.e., [X].

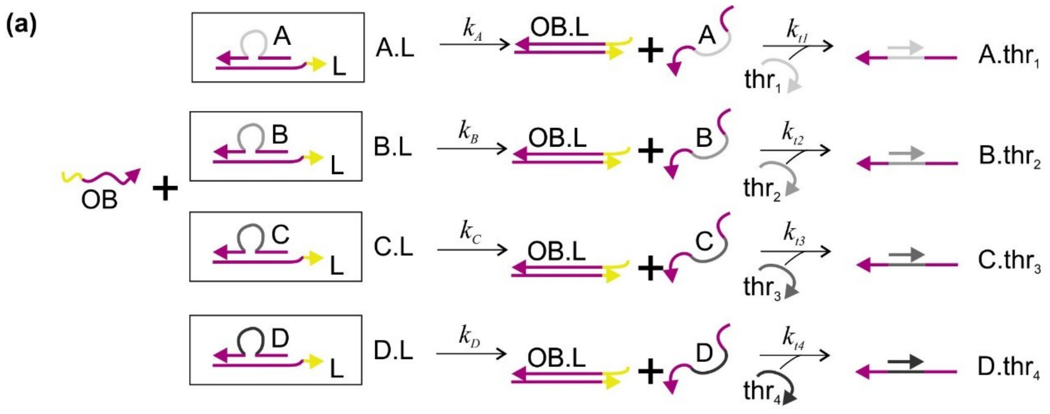
Effectively, this system implements a 'while' loop. While the pH is high, SB and OB are produced. The simulated dynamics of this system are shown in Figure 2b, and the corresponding pH is shown in Figure 2c. The simulation essentially involves the solution of a set of coupled differential equations, as described by Zhang et al., with the addition of an activity variable that defines whether the switch is active as a catalyst (i.e., open) or inactive (i.e., closed). This variable depends on the pH, which is defined by voltage pulses. It will be seen that after the switch has been opened the catalytic cycle can begin operating, with the result that fuel and substrate concentrations fall. The concentration of output OB and signal SB rise, and these curves are very similar. If there is plenty of fuel and all reactions run to completion, the reactions produce one OB for every SB. When the pH is reset, the activity of the catalyst falls to almost zero and the production of OB and SB slows or stops. The concentration of closed switch is restored to its original value.

The final concentration of the output, $[OB]_{final}$, is shown in Figure 2d as a function of the period for which the switch is open, for one set of parameters. As t_{open} increases, $[OB]_{final}$ increases, but at a decreasing rate. The range of $[OB]_{final}$ values obtained for a given range of opening times can be changed by altering the concentration of switch, fuel and substrate, as shown in Figure 2e. Here, the quantity $[OB]_{100–900}$ is the difference in the final [OB] values (125 min after the start of the initial voltage pulse) for t_{open} = 100 s and t_{open} = 900 s. The data reveal that for a given concentration of substrate and fuel, there is a particular concentration of switch catalyst for which $[OB]_{100–900}$ is maximal. Increasing the switch concentration beyond this value leads to a decrease in $[OB]_{100–900}$. For higher concentrations of substrate and fuel, the final concentration of OB is higher.

The simulation shows that the parameters are interlinked—if one is changed then others must also be changed to ensure optimal performance, where the optimum is defined according to the application. In this paper, we assume an operating range of OB concentrations of 1.5–5 nM, which can be achieved using the following parameters: switch catalyst concentration of 1.7 nM, substrate and fuel concentrations of 11 nM, t_{open} ranging from approximately 1 min to approximately 40 min, total reaction time 125 min.

2.2. Using a Thresholding Mechanism to Assess the Concentration of the Output from the Catalytic Cycle

The thresholding system is shown in Figure 3a, and resembles some of the thresholding mechanisms that have been reported previously [15,18]. Here, the complexes A.L, B.L, C.L and D.L are present in considerable excess. The output OB at concentration $[OB]_{final}$ can displace the species A–D from the strand L through toehold mediated strand displacement, releasing $[OB]_{final}/4$ of each of A–D. Each of the strands thr_1-thr_4 can then bind specifically to one of A–D, rendering it inactive. Thus, in order for one of the species A–D to be observed in solution, $[OB]_{final}$ must exceed $4\times$ the concentration of the corresponding thr. In our simulations, we set the concentrations of thr_1-thr_4 to be 0.25, 0.5, 0.75 and 1 nM respectively. In this situation, if we wish to produce free A–C, but no free D, we need $[OB]_{final}$ to lie below $4[thr_4]$ = 4 nM but above $4[thr_3]$ = 3 nM. The higher $[OB]_{final}$ lies within this range, the larger the signal from A–C. Figure 3b shows how the concentration of species A–D changes as a function of time for four values of $[OB]_{final}$, with the parameters given here.



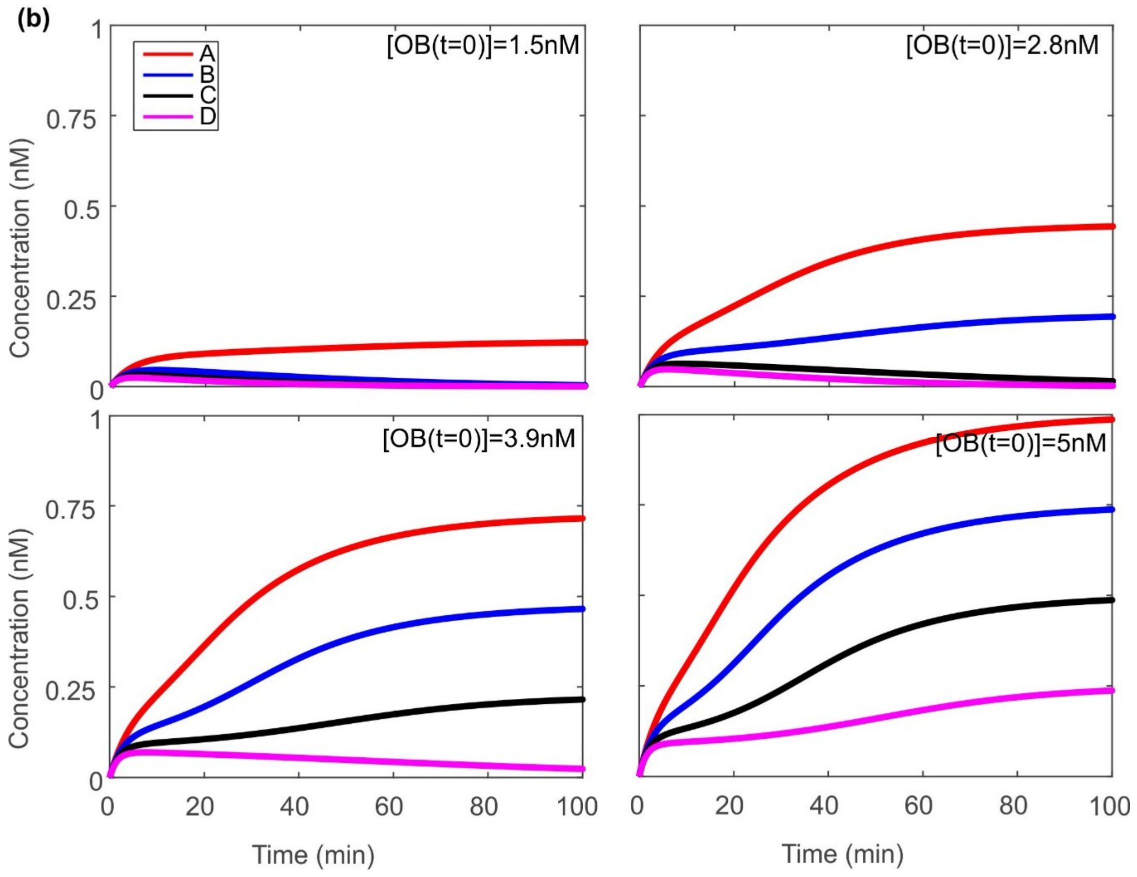


Figure 3. Using a thresholding mechanism to assess how much output has been produced by the Zhang cycle. (**a**) The thresholding mechanism. Zhang cycle output OB displaces A–D from the initial complexes. Single-stranded molecules thr_1–thr_4 bind to the central (grey) domain of A–D. If the concentration of a thr strand exceeds the concentration of its target A–D, the target species will be entirely consumed by thr. If the target species concentration is greater than that of its corresponding thr, some of the target will be present free and unbound; (**b**) dynamics of A–D as a function of time for four different concentrations of OB. Here, the initial concentrations of thr_1–thr_4 are 0.25, 0.5, 0.75, 1 nM respectively. The total concentrations of A–D are 20nM. Rate constants k_A–k_D are all equal to 1×10^4 $M^{-1} \cdot s^{-1}$, and k_{t1}–k_{t4} are equal to 1×10^7 $M^{-1} \cdot s^{-1}$. 2×10^5 time steps were used, with a spacing of 0.05 s.

2.3. Activation and Deactivation of Dumb-Bell Logic Gates

Within the system we have defined, there are four scenarios we need to consider: (i) A-only, (ii) A–B with no C and no D, (iii) A–C and no D, (iv) A–D all present. These situations can be distinguished by simple logic operations. The first may be identified by the case A AND NOT B = TRUE. For case ii, the characteristic condition is B AND NOT C = TRUE. The third case corresponds to C AND NOT D = TRUE, while the fourth case is simply D = TRUE. (Each condition is initially characterized by a four-variable statement, but these can be reduced to statements of two variables when it is noted that the four scenarios listed are the cases to be considered. For instance, in this system, it is not possible to have A and C with no B.)

To distinguish these four situations, we propose the use of dumb-bell hairpin logic gates—unimolecular constructs that are activated by one input and deactivated by the second (as shown in Figure 4a for inputs A and B). Static dumb-bell hairpins were used by Rothemund as a marker on the surface of DNA origami tiles [2]. The loop of each hairpin conceals three domains. The finite state machine binding domain is the same for all gates and allows the gate to bind to finite state machines as input. The gate identity domain is the sequence that specifies what the input is, and is unique for each gate. The lock domain may or may not be the same for all gates. On one side of the dumb-bell the sequences in the hairpin loop are the reverse complements of those on the other side. The double-stranded neck domains on each side of the dumb-bell are not complementary.

When the first species binds to the gate, it opens the stem on one side of the dumb-bell. When the second species binds, it opens the other stem. These hairpin loops are complementary and therefore hybridize. It is important to note that this dumb-bell hairpin could not be synthesized as a single strand, because in the absence of any constraints it would assemble incorrectly, with the lock, identity and FSM-binding domains bound to each other from the start. However, the dumb-bell could be prepared by straightforward ligation of two separate hairpin constructs.

Figure 4b shows how the species A–D interact with the gates. In this simulation, the initial concentrations of A–D are taken from the thresholder simulation, and all gates are assumed to be present at 0.1 nM. The rate at which species A and B displace the corresponding domains in gate 1 is 6×10^6 $M^{-1} \cdot s^{-1}$, as is the rate at which B and C displace their targets in gate 2. Species C and D bind to gate 3 at a rate of 2.5×10^6 $M^{-1} \cdot s^{-1}$, while D binds to gate 4 more slowly, with a rate constant of 2.5×10^5 $M^{-1} \cdot s^{-1}$. These rate constants are based on published figures for the rates of toehold mediated strand displacement [38,41] and hybridization [42], and can be tuned by modifying factors such as domain length, GC-content and complementarity (i.e., the presence of mismatches).

As the data show, the concentrations of A–D fall after introduction of the logic gates. The concentration of unreacted gates also falls, and the rate depends on the concentration of the inputs. The dynamics of the active gates depend on the combination of the inputs, as intended. In the example shown in Figure 2b, we wanted gate 3 to be activated and the other gates to remain unreacted or be inactivated. It will be seen that this is the result obtained, with very low levels of active gates 1, 2 and 4. Some gate 3 is inactivated but most remains active. The three other scenarios are shown in Figure 4c. In each case, only one gate remains strongly activated, and the concentration of other active gates remains low. This is exactly the desired behavior. In these simulations, the final concentration of the activated target gate ranges from 0.04 to 0.1 nM.

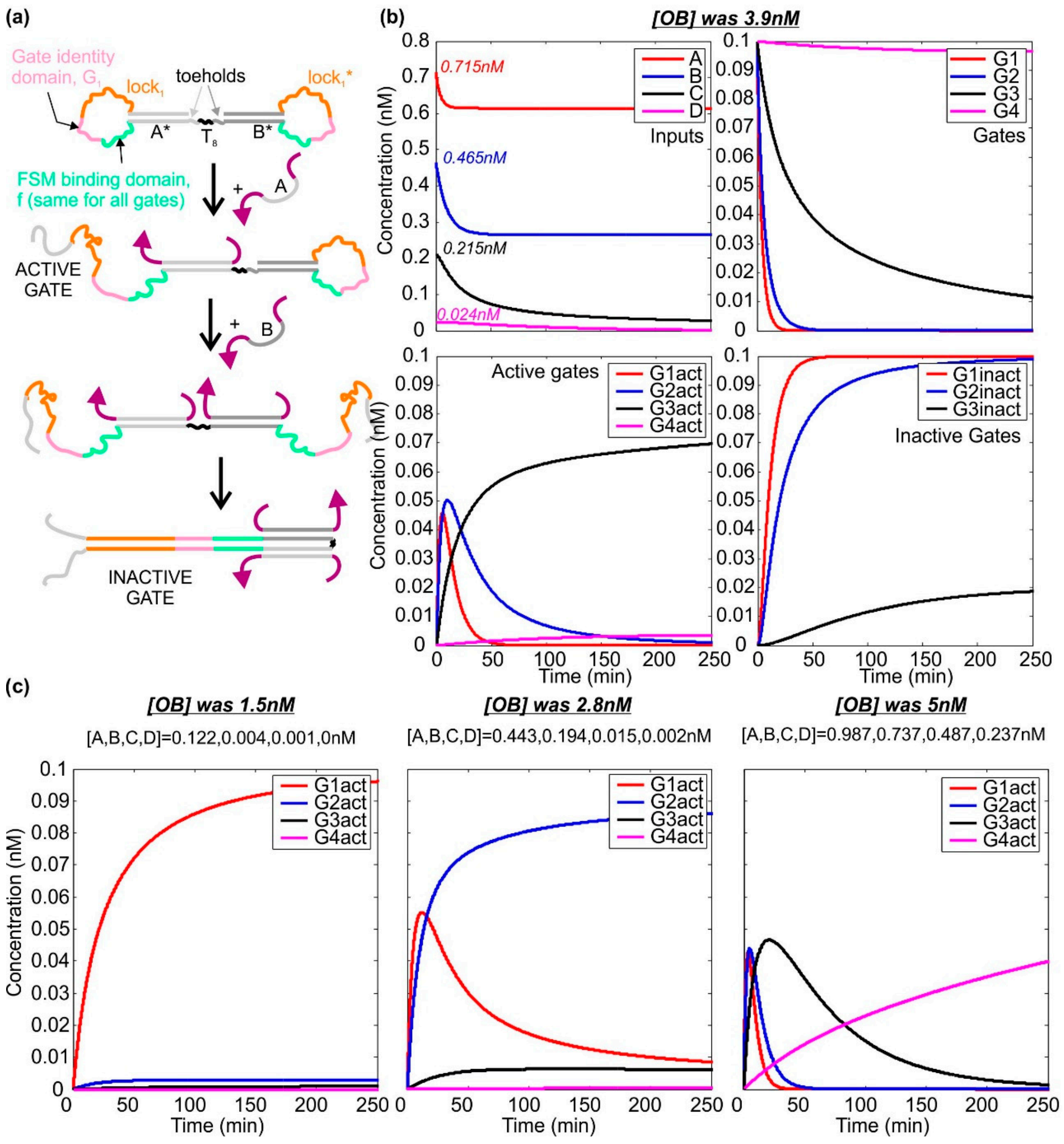


Figure 4. Dumb-bell logic gates—activation and deactivation. (**a**) Operating mechanism of the gates. Binding of species A activates the gate by revealing its identity and lock domains. Binding of species B reveals the reverse complement of both of these domains, leading to gate deactivation. This can occur through the mechanism of toehold-mediated strand displacement even if the FSM-binding domain has already bound to its target. Gates that respond to other pairs of species differ in the sequence of the domains; (**b**) simulated dynamics for the molecules involved in thresholding, for the case in which the concentration of OB was 3.9 nM, and the initial concentrations of A–D were 0.715, 0.465, 0.215 and 0.024 nM respectively. Plots show the behavior of the inputs, the unreacted gates, the active gates and the inactive gates, as labelled. Gate 1 is activated by A and deactivated by B. Gate 2 is activated by B and deactivated by C. Gate 3 is activated by C and deactivated by D. Gate 4 is activated by D and is not deactivated. Gates are present at 0.1 nM. The activation and deactivation rates for Gates 1 and 2 are all 6 $\times$ 10^6 $M^{-1}{\cdot}s^{-1}$, the maximum for toehold-mediated strand displacement. For Gate 3, the corresponding rates are 2.5 $\times$ 10^6 $M^{-1}{\cdot}s^{-1}$, and for Gate 4 the activation rate is 2.5 $\times$ 10^5 $M^{-1}{\cdot}s^{-1}$; (**c**) dynamics of the active gates for three other concentrations of [OB]. In each case, one particular logic gate is activated and the others either do not react or become inactivated.

2.4. Operating a Finite State Machine Using the Dumb-Bell Logic Gates as Input

The result of the steps described so far is the production of a single species—the active gate. For the next step, the transition rules are injected. These are molecules that dictate what the final state of the finite state machines should be, based on the initial state and the input provided by the active gate. One possible molecular implementation of this system is shown in Figure 5a. The design is based on that of Costa Santini et al. [23], but here the finite state machine is immobilized on the surface and there is no molecular clock. In our system activated dumb-bell hairpins can bind to the immobilized state machines, and the transition rule then binds to the domain formed by the conjunction of the initial state domain of the state machine and the identity domain of the hairpin gate. In principle, at one stage all gates are activated before being inactivated and all active gates can bind to state machines. However, inactivation of the gate will result in its removal from the state machine by strand displacement, and it is not possible for a state transition to occur in the intervening time because the transition rules are not supplied until the logic gates have reached equilibrium with the species A–D.

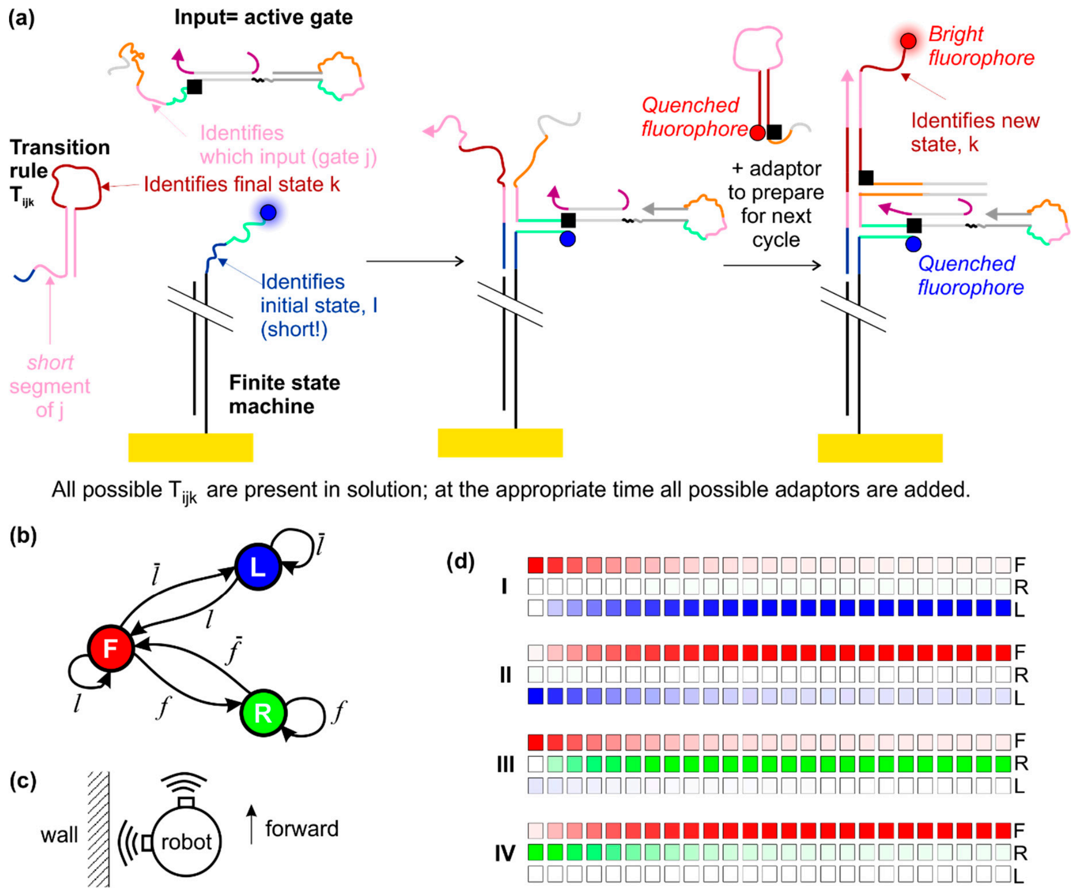

(a)
Input= active gate
Transition rule T_{ijk}
Identifies which input (gate j)
Identifies final state k
Identifies initial state, I (short!)
short segment of j
Finite state machine
Quenched fluorophore
+ adaptor to prepare for next cycle
Bright fluorophore
Identifies new state, k
Quenched fluorophore
All possible T_{ijk} are present in solution; at the appropriate time all possible adaptors are added.
(b)
L
F
R
$\bar{l}$
l
$\bar{f}$
f
(c)
wall
robot
forward
(d)
I
II
III
IV
F
R
L

Figure 5. Operating a finite state machine using the dumb-bell logic gates as inputs. (**a**) Molecular implementation of the system, based on the design of Costa Santini et al. [23], with modifications. A possible fluorescence readout scheme is shown; (**b**) schematic diagram of a finite state machine for a maze-crawling robot that follows the left wall. States are represented in circles and transitions between them are labelled with the condition that induces the transition; (**c**) schematic diagram of envisaged robot with two sensors; (**d**) results of simulations modelling four cycles of the system, illustrating how the state changes over time. The results take into account predicted concentrations at the end of every step in each cycle, and are presented in more detail in Table 3. Temporal separation of consecutive squares is 4 min.

As a specific example, we may consider the finite state machine that describes the operation of a maze-crawling robot that follows the left wall (Figure 5b). This robot may be in three states—moving forward (F), rotating to the right (R) or rotating to the left (L). These states may be represented by a three-element vector $s = (a,b,c)$. In a pure state, a, b and c are either 1 or 0, and only one of them is non-zero. (1,0,0) represents state F, (0,1,0) represents state L and (0,0,1) represents state R. There are two sensors, which register true in the proximity of a wall. One sensor is forward-facing and one faces left. The signal f corresponds to a true reading from the forward-facing sensor, and $\overline{f}$ is a false reading from the same sensor. l and $\overline{l}$ are the corresponding true/false readings for the sensor facing left. We may thus describe the sensors using inputs 1–4 as indicated in Table 1 for representative states and inputs.

Table 1. Idealized behaviour of the finite state machine for examples of specified initial conditions and inputs. The state vector contains three elements and is normalized such that it has a magnitude of 1. The input vector is also normalized, but contains four elements, to reflect the four possible input signals.

	Initial State	Initial State Vector	Signal	True Input	Input Vector	Final State	Final State Vector
I	F	(1,0,0)	f	1	(1,0,0,0)	R	(0,0,1)
II	R	(0,0,1)	$\overline{f}$	2	(0,1,0,0)	F	(1,0,0)
III	F	(1,0,0)	$\overline{l}$	3	(0,0,1,0)	L	(0,1,0)
IV	L	(0,1,0)	l	4	(0,0,0,1)	F	(1,0,0)

The input from the sensors may be represented as a four column vector, $g = (w,x,y,z)$. If the input is pure, all but one of the elements is zero and the other element is 1. This also reflects the ideal (normalized) result of the logic gate operation described above. Using index notation and summation convention, the final state $\Psi_r = s_p g_q T_{pqr}$, where T_{pqr} is the transition rule. The transition rules are defined in Table 2. T_{21r}, T_{22r}, T_{33r} and T_{44r} are meaningless and thus denoted by (0,0,0).

Table 2. The full set of transition rules for the finite state machine illustrated in Figure 5, using the mathematical formalism $\Psi_r = s_p g_q T_{pqr}$. There are three states and four possible input vectors, which means that 12 transition rules can be mathematically defined. However, four of these rules are irrelevant because the encoded condition could not be observed in reality (T_{21r}, T_{22r}, T_{33r}, T_{44r}).

s_p	g_q	T_{pqr}
(1,0,0) F	(1,0,0,0)	T_{11r} = (0,0,1)
	(0,1,0,0)	T_{12r} = (1,0,0)
	(0,0,1,0)	T_{13r} = (0,1,0)
	(0,0,0,1)	T_{14r} = (1,0,0)
(0,1,0) L	(1,0,0,0)	T_{21r} = (0,0,0)
	(0,1,0,0)	T_{22r} = (0,0,0)
	(0,0,1,0)	T_{23r} = (0,1,0)
	(0,0,0,1)	T_{24r} = (1,0,0)
(0,0,1) R	(1,0,0,0)	T_{31r} = (0,0,1)
	(0,1,0,0)	T_{32r} = (1,0,0)
	(0,0,1,0)	T_{33r} = (0,0,0)
	(0,0,0,1)	T_{44r} = (0,0,0)

Our simulations can be used to predict the behaviour of the finite state machines for a particular scenario, and the results are shown in Table 3. We assume that the system is initially homogeneous and all machines are in state (1,0,0); i.e., the robot is moving forwards. Next, we assume that the sensors detect a wall in front of the robot. This corresponds to input 1 being true. This is encoded as a specific separation of opening and closing voltage pulses (approximately 1 min) that gives rise to [OB] = 1.5 nM. The resulting concentrations of A–D and the active gates are given in the table, together with the computed final state. We consider three subsequent steps, each starting from the final configuration of the previous step. The robot rotates to the right, detects that there is no longer a wall in front of it, moves forward, then detects that there is no longer a wall on the left, rotates to the left, detects that it has found the left wall again and starts moving forward (bottom right entry in table). We assume that the speed of the robot and the range of its sensors are such that action is always taken before the robot hits the wall.

Table 3. Results of the simulations described in this paper, for the series of states and inputs described in Table 1. The value of [OB] is exact, as is the first initial state. Values of [A]–[D] and the concentration of the active gates [G_1^{act}]–[G_4^{act}] are given to 3 d.p. and the state vector elements are given to 4 d.p. Where the state vectors are represented as column vectors, this is for reasons of space rather than mathematical necessity.

	Initial State	**[OB] (nM)**	**[A] (nM)**	**[B] (nM)**	**[C] (nM)**	**[D] (nM)**	**[G_1^{act}] (pM)**	**[G_2^{act}] (pM)**	**[G_3^{act}] (pM)**	**[G_4^{act}] (pM)**	**Final State**
I	(1,0,0)	1.5	0.122	0.004	0.001	0	96.084	2.841	0.931	0	(0.0296, 0.0097, 0.9995)
II	(0.0296, 0.0097, 0.9995)	2.8	0.443	0.194	0.015	0.002	8.520	86.072	6.229	0.571	(0.9951, 0.0027, 0.0985)
III	(0.9951, 0.0027, 0.0985)	3.9	0.715	0.465	0.215	0.024	0	1.044	69.702	3.560	(0.0673, 0.9977, 0)
IV	(0.0673, 0.9977, 0)	5.0	0.987	0.737	0.487	0.237	0	0	1.377	39.846	(0.9994, 0.0345, 0)

Figure 5d represents the results visually, illustrating the dynamics using a time series of colored blocks in which the intensity of the color indicates the probability of being in the indicated state. Here, red denotes state F, green denotes state R, blue denotes state L. As an approximation, we assume that the rate for the change of state is 0.001 s^{-1}, based on extrapolation of our previously published experimental data [43] to the low concentration/surface density regime. The time step between blocks is 4 min. We observe that a very high percentage of the finite state machines (over 99%) enter the desired final state.

Although we have considered a specific scenario here, in principle the system we have designed would allow the robot to navigate autonomously through an arbitrary maze without human intervention. For each operating cycle, it would sense its environment using proximity sensors (for example based on infrared radiation) and use a look-up table to automatically encode the sensory input as a particular time delay between opening and closing pulses. Sequential injection of thresholders, logic gates, transition rules and adaptors would follow, where each species would be added automatically at a precise time based on an underlying electronic clock. If the external situation did not change for a long time, a number of operational cycles could pass without a change of state of the finite state machine.

2.5. Readout Schemes

For implementation a method of probing the finite state machines will be required. For example, fluorescence signals could be used to indicate the final state. In our framework each adaptor takes the form of a hairpin, which opens upon binding to the finite state machine. Each adaptor could be labelled with a specific fluorophore (e.g., from the Atto family) and an appropriate quencher (e.g., Iowa Black), as illustrated in Figure 5a. Opening of the adaptor hairpin would dramatically increase the separation of fluorophore and quencher, leading to a significant rise in the fluorescence signal. The use of multiple colors of fluorophore (one for each state) would allow the output to be read. Of course, it would be necessary to quench the bound fluorophore at the start of the subsequent cycle, to avoid an increase in the background signal. Hence, the input (gate) would also need to be functionalized with a quencher (Figure 5a). To measure the fluorescence, the robot would need an appropriate light source, filters and detectors. Alternatively an electrochemical strategy could be employed, based on an E-DNA reporter circuit making use of charge transfer between an electrode surface and a redox-active molecule such as methylene blue.

3. Discussion

We have presented a possible design for a hybrid bioelectronic computer that processes electrical signals using DNA molecular machines and makes decisions autonomously. Our system has not yet been validated experimentally, but we have performed extensive simulations of its behaviour. Our results suggest that it will operate as expected, and lay the foundations for the construction and testing of a real system. Our own experimental work in this area is ongoing, but we discuss some experimental considerations in Supplementary Discussion 1. We note that the finite state machines presented in Figure 5 contain in their structure a memory of their entire history. In principle an add-on module could be designed that would deconstruct the machines and read back the operations performed, allowing the system to repeat its actions in reverse.

In this work, we considered the possibility of using our system to control a maze-crawling robot. This would be a proof-of-concept demonstrator, and the principles could subsequently be expanded to other situations. Our approach is likely to be of particular utility in situations where both biomolecular and electronic stimuli are required. However, as presently envisaged, our system would take 10 h for each step of the finite state machine, due to the speed of the DNA reactions (Supplementary Discussion 2). This may be prohibitive for many applications, and it would be highly desirable to increase the reaction rate by three or four orders of magnitude—reducing a ten-hour process to a few seconds.

Slight increases in speed could potentially be achieved by optimization of the operating parameters [38,39,41], such as the concentration of various species and design of DNA sequences. However, increasing the processing speed to the desired levels would require a paradigm shift. For example, Qian and Winfree [18] have pointed out that the phenol emulsion reassociation technique [44] may be a viable route for increasing the speed of DNA computers, perhaps by up to four orders of magnitude. Organic solvents can also increase the speed of hybridization [45]. Alternatively, it may be possible to use methods developed to enhance the speed of biochemical assays involving DNA, such as isotachophoresis [46,47]. Furthermore, it has also been shown that recombination proteins may be able to significantly enhance the rate of hybridization [48,49], and such proteins may have the potential to speed up DNA computers.

For future development of hybrid bioelectronic computers that contain large numbers of interacting DNA strands it will also be necessary to optimize the sequences to minimize spurious interactions and leak reactions. It may be possible to do this by using evolutionary algorithms to identify the most appropriate sequences [50]. Alternatively, DNA circuits could be localized on a surface, such as a DNA origami tile, which would allow re-use of sequences, minimizing leak reactions [51]. This approach could also increase reaction speed in comparison with systems that have freely diffusing components.

It is possible that hybrid bioelectronic processors will scale better than DNA-only systems. In many DNA-based computing systems, the number of unique sequences required does not scale well with the complexity of the problem and the number of molecular species required is likely to be impractical for problems other than the simplest. For certain applications bioelectronic architectures may overcome this limitation because the electronic components could be used to reprogram the computer, and it would not be necessary to synthesize an entirely new set of strands.

In DNA computing, many systems are characterized by 'one-pot' reactions, where all reactants are present at the same time in the same volume. Our approach does not follow this paradigm. In our proposed system, multiple steps must occur within each information processing layer before the final product is passed to the next layer. If we attempted to employ a one-pot procedure, intermediates from one layer could react prematurely with components from the next layer. For instance, in a one-pot scenario the species 'A' produced by the top thresholding reaction in Figure 3a could either interact with the species thr, or the dumb-bell hairpin logic gate shown in Figure 4a. In this case, the thresholder circuit would not function correctly. In principle, the effects of cross-reactivity could be reduced by careful selection of domain lengths and sequences, but eliminating the requirement to use a one-pot procedure allows more flexible design of DNA molecules and opens up the design space for multi-layer information processing systems, enabling a more modular approach.

Our system is limited in terms of speed and takes 10 h to complete a cycle. One-pot approaches might be an order of magnitude faster, but one hour would still be too slow for many applications. Thus, a one pot procedure may have a speed advantage, perhaps by an order of magnitude, but is still too slow for many applications. This means that significant advances will be needed to speed up DNA computers for practical use even if they do employ a one-pot approach.

Furthermore, we note that fundamental information processing operations in biology frequently do not take place in a one-pot reaction. For example, we may consider gene expression in eukaryotes. In these organisms, transcription occurs in the nucleus and mRNA transcripts must exit through nuclear pores so that translation can occur at the ribosomes. This multi-layer processing mechanism allows eukaryotes to utilize more complex mechanisms for gene regulation than their prokaryotic counterparts. Taking inspiration from biology, we suggest that departure from the one-pot reaction paradigm will provide new opportunities for DNA computing, with advantages that outweigh the limitations.

If the speed of the molecular processes can be increased significantly and steps are taken to minimize leak reactions, DNA-based bioelectronic computers will have considerable potential. In principle, coupling electronic circuitry and biomolecular devices could enable the construction of low-power biocompatible thinking machines that process information in a highly parallel and potentially fault-tolerant manner. Here we have presented theoretical testing of an initial design for a basic hybrid system, illustrating how information could be transmitted seamlessly across a bioelectronic interface and processed dynamically for autonomous decision making. Future research will build on this study, and could lead to practical devices that could be used in real-world environments.

4. Materials and Methods

All simulations were performed using MATLAB R2013a (Mathworks, Glasgow, UK). Each phase of the cycle was modelled as a separate operation. All code is available in the Supplementary Information (Supplementary Methods).

4.1. Calculating the Effect of a Voltage Pulse on the pH-Sensitive Nanoswitch

We assume that the pH changes as a result of the electrolysis of water. We also assume that the current, I, is 0.4 mA (voltage of 4 V, resistance of 10 kΩ) and the volume of solution is 100 µL. We neglect diffusion time for OH^- and H^+ ions, effectively assuming that they are produced uniformly throughout the solution. The number of ions produced in a time interval dt is then $n = I\,\mathrm{d}t/(N_A e)$, where N_A is Avagadro's number and e is the electron charge. The concentration change is $\Delta c = n/v$, where v is the reaction volume. This allows the pH to be calculated. The initial pH is assumed to be 5.5.

The effective pK_a of the switch under consideration is taken to be 7.5. In accordance with the work of Idili et al. [21], this can be adjusted by changing the switch design. pK_a is defined as $-\log_{10}(K_\mathrm{a})$, where K_a = [A^-][H^+]/[AH]. Identifying HA as the protonated (closed) form of the switch and A^- as the deprotonated (open) form, and using the definition of pH, we obtain [Open]/[Closed] = $10^{\mathrm{pH}-\mathrm{p}K\mathrm{a}}$. Defining [Total] = [Open] + [Closed] yields the result that [Open] = [Total]/(1 + $10^{\mathrm{p}K\mathrm{a}-\mathrm{pH}}$). The quantity [Total] is conserved and defined at the start of the experiment.

4.2. Simulating the Behaviour of a Zhang Catalytic Cycle Where the Catalyst Is a pH-Sensitive Nanoswitch

The voltage is switched on at the start of the simulation, and left on for 20 s. After a delay, a voltage of equal and opposite amplitude is applied for the same length of time. Using the result from above, the activity of the catalyst is defined as [Open]/[Total]. This is computed for all time points in advance. The coupled differential equations provided by Zhang et al. [37] are then solved, assuming that only the active catalyst molecules participate in the reaction. With this approach, we assume that a nanoswitch can only reclose when its loop domain is not bound to another molecule. The time interval used is 0.05 s and the number of time steps is 150,000, corresponding to a total simulation time of 125 min. Substrate, fuel and total switch concentrations are varied.

4.3. Modelling the Thresholders

The system of thresholders is modelled as follows. To facilitate compact mathematical formulation, OB is represented as x and A-D are referred to as A_m, where A_1 corresponds to A while A_2 corresponds to B, etc. Y.Z represents the construct formed by hybridization of Y and Z, where Y and Z may be A_m, L_m, th_m.

$$x + A_m{\cdot}L_m \rightarrow x{\cdot}L_m + A_m \tag{1}$$

$$A_m + th_m \rightarrow A_m{\cdot}th_m \tag{2}$$

The rate of the first reaction is k_m and the rate of the second is k_{tm}. Here, $k_{tm} >> k_m$. The reactions are irreversible.

The rate of change of free A_m is given by:

$$d[A_m]/dt = k_m[x][A_m{\cdot}L_m] - k_{tm}[A_m][th_m] \tag{3}$$

However, we know that the total amount of A_m (free or bound to another molecule) is conserved:

$$[A_m{\cdot}L_m] + [A_m] + [A_m{\cdot}th_m] = A_{m,TOT} \tag{4}$$

Similarly:

$$[A_m{\cdot}th_m] + [th_m] = th_{m,TOT} \tag{5}$$

This means that:

$$[A_m{\cdot}L_m] = A_{m,TOT} - [A_m] - th_{m,TOT} + [th_m] \tag{6}$$

Hence:

$$d[A_m]/dt = k_m[x](A_{m,TOT} - [A_m] - th_{m,TOT} + [th_m]) - k_{tm}[A_m][th_m] \tag{7}$$

And

$$d[th_m]/dt = -k_{thm}\,[A_m][th_m] \tag{8}$$

The quantity of output is also conserved, as is the quantity of each L_m:

$$\begin{aligned}[x] &= x_{TOT} - \Sigma_m\, x_m{\cdot}L_m = x_{TOT} - \Sigma_m(L_{m,TOT} - [A_m{\cdot}L_m]) \\ &= x_{TOT} - \Sigma_m(L_{m,TOT} - A_{m,TOT} + [A_m] + th_{m,TOT} - [th_m])\end{aligned} \tag{9}$$

where x_{TOT} and $L_{m,TOT}$ are the conserved quantities.

Using the notation that $L = \Sigma_m\, L_{m,TOT}$, $T = \Sigma_m\, th_{m,TOT}$ and $A = \Sigma_m\, A_{m,TOT}$, we obtain:

$$x = x_{TOT} - L - T + A + \Sigma_m([th_m] - [A_m]) \tag{10}$$

In the simulation, Equations (7), (8) and (10) are solved for every time point to yield the dynamics of the species of interest. We assume that for all m, $k_m = 10^4$ M$^{-1}\cdot$s^{-1} and $k_{tm} = 10^7$ M$^{-1}\cdot$s^{-1}, in accordance with published data [42]. The total concentration of each A_m is taken to be 20 nM, and the total concentrations of th_{1-4} are 0.25 nM, 0.5 nM, 0.75 nM and 1 nM respectively. These parameters could be adjusted as appropriate.

4.4. Simulating the Dynamics of the Dumb-Bell Logic Gates

The following reactions occur:

$$A + G_1 \rightarrow G_1^{act} \qquad \text{rate: } k_{g1a} \tag{11}$$

$$B + G_1^{act} \rightarrow G_1^{inact} \qquad \text{rate: } k_{g1b} \tag{12}$$

$$B + G_2 \rightarrow G_2^{act} \qquad \text{rate: } k_{g2a} \tag{13}$$

$$C + G_2^{act} \rightarrow G_2^{inact} \qquad \text{rate: } k_{g2b} \tag{14}$$

$$C + G_3 \rightarrow G_3^{act} \qquad \text{rate: } k_{g3a} \tag{15}$$

$$D + G_3^{act} \rightarrow G_3^{inact} \qquad \text{rate: } k_{g3b} \tag{16}$$

$$D + G_4 \rightarrow G_4^{act} \qquad \text{rate: } k_{g4a} \tag{17}$$

Here, the superscript 'act' denotes an activated gate and 'inact' denotes an inactivated gate. These reactions give rise to the following set of coupled differential equations, which are solved numerically in MATLAB.

$$d[A]/dt = -k_{g1a}[G_1][A] \tag{18}$$

$$d[G_1^{act}]/dt = k_{g1a}[G_1][A] - k_{g1b}[G_1^{act}][B] \tag{19}$$

$$d[G_1^{inact}]/dt = k_{g1b}[G_1^{act}][B] \tag{20}$$

$$d[B]/dt = -k_{g2a}[G_2][B] - k_{g1b}[G_1^{act}][B] \tag{21}$$

$$d[G_2^{act}]/dt = k_{g2a}[G_2][B] - k_{g2b}[G_2^{act}][C] \tag{22}$$

$$d[G_2^{inact}]/dt = k_{g2b}[G_2^{act}][C] \tag{23}$$

$$d[C]/dt = -k_{g3a}[G_3][C] - k_{g2b}[G_2^{act}][C] \tag{24}$$

$$d[G_3^{act}]/dt = k_{g3a}[G_3][C] - k_{g3b}[G_3^{act}][D] \tag{25}$$

$$d[G_3^{inact}]/dt = k_{g3b}[G_3^{act}][D] \tag{26}$$

$$d[D]/dt = -k_{g4a}[G_4][D] - k_{g3b}[G_3^{act}][D] \tag{27}$$

$$d[G_4^{act}]/dt = k_{g4a}[G_4][D] \tag{28}$$

The rate constants are as follows:

$$k_{g1a,g1b,g2a,g2b} = 6 \times 10^6\ M^{-1}\cdot s^{-1}$$
$$k_{g3a,g3b} = 2.5 \times 10^6\ M^{-1}\cdot s^{-1}$$
$$k_{g4a} = 2.5 \times 10^5\ M^{-1}\cdot s^{-1}$$

4.5. Simulating the Behaviour of the Finite State Machine

Using index notation and summation convention, the final state of the machine is defined as $\Psi_r = s_p g_q T_{pqr}$, where s_p is the initial state, g_q is the input (the activated gate) and T_{pqr} is the transition rule. Here the state and input vectors are normalized such that $|\boldsymbol{s}\cdot\boldsymbol{s}| = 1$ and $|\boldsymbol{g}\cdot\boldsymbol{g}| = 1$.

Here it is not necessary to solve any differential equations numerically because the reaction dynamics can be found analytically. We compute the final state and infer the dynamics by assuming single exponential kinetics. The rate is taken to be 1×10^{-3} s^{-1}. This value was estimated based on the results we have obtained previously for hybridization of incoming strands to a surface-immobilized target, with extrapolation to a regime in which the solution concentration is low and the surface density is also low.

Supplementary Materials: Supplementary materials can be found at http://www.mdpi.com/1422-0067/19/9/2620/s1.

Author Contributions: Conceptualization, K.E.D.; Methodology, K.E.D.; Software, K.E.D.; Validation K.E.D.; Formal Analysis, K.E.D.; Investigation, K.E.D.; Resources, A.M.T.; Data Curation, K.E.D.; Writing-Original Draft Preparation, K.E.D.; Writing-Review and Editing, K.E.D.; M.A.T.; S.J.; A.M.T.; Visualization, K.E.D.; Supervision, K.E.D.; M.A.T.; S.J.; A.M.T.; Funding Acquisition, A.M.T.; M.A.T.; S.J.

Funding: The authors thank the UK Engineering and Physical Sciences Research Council (EPSRC) for funding under Platform Grant EP/K040820/1 (all authors) and Programme Grant EP/P030017/1 (S.J.).

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, and in the decision to publish the results.

References

1. Seeman, N.C. Nucleic acid junctions and lattices. *J. Theor. Biol.* **1982**, *99*, 237–247. [CrossRef]
2. Rothemund, P.W.K. Folding DNA to create nanoscale shapes and patterns. *Nature* **2006**, *440*, 297–302. [CrossRef] [PubMed]
3. Shen, B.; Link, V.; Tapio, K.; Pikker, S.; Lemma, T.; Gopinath, A.; Gothelf, K.V.; Kostiainen, M.A.; Toppari, J.J. Plasmonic structures through DNA-assisted lithography. *Sci. Adv.* **2018**, *4*, eaap8978. [CrossRef] [PubMed]
4. Ketterer, P.; Ananth, A.N.; Laman Trip, D.S.; Mishra, A.; Bertosin, A.; Ganji, M.; van der Torre, K.; Onck, P.; Dietz, H.; Dekker, C. DNA origami scaffold for studying intrinsically disordered proteins of the nuclear pore complex. *Nat. Commun.* **2018**, *9*, 902. [CrossRef] [PubMed]
5. Funke, J.J.; Dietz, H. Placing molecules with Bohr radius resolution using DNA origami. *Nat. Nanotechnol.* **2016**, *1*, 47–52. [CrossRef] [PubMed]
6. Komiyama, M.; Yoshimoto, K.; Sisido, M.; Ariga, K. Chemistry Can Make Strict and Fuzzy Controls for Bio-Systems: DNA Nanoarchitectonics and Cell-Macromolecular Nanoarchitectonics. *Bull. Chem. Soc. Jpn.* **2017**, *90*, 967–1004. [CrossRef]
7. Sherman, W.B.; Seeman, N.C. A precisely controlled DNA biped walking device. *Nano Lett.* **2004**, *4*, 1203–1207. [CrossRef]
8. Shin, J.S.; Pierce, N.A. A synthetic DNA walker for molecular transport. *J. Am. Chem. Soc.* **2004**, *126*, 10834–10835. [CrossRef]
9. Qu, X.; Zhu, D.; Yao, G.; Su, S.; Chao, J.; Liu, H.; Zuo, X.; Wang, L.; Shi, J.; Wang, L.; et al. An Exonuclease III-Powered, On-Particle Stochastic DNA Walker. *Angew. Chem. Int. Ed.* **2017**, *56*, 1855–1858. [CrossRef] [PubMed]
10. Pandian, G.N.; Sugiyama, H. Nature-Inspired Design of Smart Biomaterials Using the Chemical Biology of Nucleic Acids. *Bull. Chem. Soc. Jpn.* **2016**, *89*, 843–868. [CrossRef]
11. Kahn, J.S.; Hu, Y.; Willner, I. Stimuli-Responsive DNA-Based Hydrogels: From Basic Principles to Applications. *Acc. Chem. Res.* **2017**, *50*, 680–690. [CrossRef] [PubMed]
12. Praetorius, F.; Kick, B.; Behler, K.L.; Honemann, M.N.; Weuster-Botz, D.; Dietz, H. Biotechnological mass production of DNA origami. *Nature* **2017**, *552*, 84–87. [CrossRef] [PubMed]
13. Burns, J.R.; Lamarre, B.; Pyne, A.L.B.; Noble, J.E.; Ryadnov, M. DNA Origami Inside-Out Viruses. *ACS Synth. Biol.* **2018**, *7*, 767–773. [CrossRef] [PubMed]
14. Adleman, L.M. Molecular computation of solutions to combinatorial problems. *Science* **1994**, *266*, 1021–1024. [CrossRef] [PubMed]
15. Seelig, G.; Soloveichik, D.; Zhang, D.Y.; Winfree, E. Enzyme-free nucleic acid logic circuits. *Science* **2006**, *314*, 1585–1588. [CrossRef] [PubMed]
16. Frezza, B.M.; Cockroft, S.L.; Ghadiri, M.R. Modular multi-level circuits from immobilized DNA-based logic gates. *J. Am. Chem. Soc.* **2007**, *129*, 14875–14879. [CrossRef] [PubMed]
17. Stojanovic, M.N.; Mitchell, T.E.; Stefanovich, D. Deoxyribozyme-based logic gates. *J. Am. Chem. Soc.* **2002**, *124*, 3115–3123. [CrossRef]
18. Qian, L.; Winfree, E. A simple DNA gate motif for synthesizing large-scale circuits. *J. R. Soc. Interface* **2011**, *8*, 1281–1297. [CrossRef] [PubMed]
19. Qian, L.; Winfree, E. Scaling up digital circuit computation with DNA strand displacement cascades. *Science* **2011**, *332*, 1196–1201. [CrossRef] [PubMed]

20. Qian, L.; Winfree, E.; Bruck, J. Neural network computation with DNA strand displacement cascades. *Nature* **2011**, *475*, 368–372. [CrossRef] [PubMed]
21. Soloveichik, D.; Seelig, G.; Winfree, E. DNA as a universal substrate for chemical kinetics. *Proc. Natl. Acad. Sci. USA* **2010**, *107*, 5393–5398. [CrossRef] [PubMed]
22. Benenson, Y.; Paz-Elizur, T.; Adar, R.; Keinan, E.; Livneh, Z.; Shapiro, E. Programmable and autonomous computing machine made of biomolecules. *Nature* **2001**, *414*, 430–434. [CrossRef] [PubMed]
23. Costa Santini, C.; Bath, J.; Tyrrell, A.M.; Turberfield, A.J. A clocked finite state machine built from DNA. *Chem. Commun.* **2013**, *49*, 237–239. [CrossRef] [PubMed]
24. Rothemund, P.W.K.; Papadakis, N.; Winfree, E. Algorithmic self-assembly of DNA Sierpinski triangles. *PLoS Biol.* **2013**, *2*, e424. [CrossRef] [PubMed]
25. Currin, A.; Korovin, K.; Ababi, M.; Roper, K.; Kell, D.B.; Day, P.J.; King, R.D. Computing exponentially faster: Implementing a non-deterministic universal Turing machine using DNA. *J. R. Soc. Interface* **2017**, *14*, 20160990. [CrossRef] [PubMed]
26. Dunn, K.E.; Trefzer, M.A.; Johnson, S.; Tyrrell, A.M. Assessing the potential of surface-immobilized molecular logic machines for integration with solid state technology. *Biosystems* **2016**, *146*, 3–9. [CrossRef]
27. Yang, Y.; Liu, G.; Liu, H.; Li, D.; Fan, C.; Liu, D. An electrochemically actuated reversible DNA switch. *Nano Lett.* **2010**, *10*, 1393–1397. [CrossRef] [PubMed]
28. Minero, G.A.S.; Wagler, P.F.; Oughli, A.A.; McCaskill, J.S. Electronic pH switching of DNA triplex reactions. *RSC Adv.* **2015**, *5*, 27313–27325. [CrossRef]
29. Ranallo, S.; Amodio, A.; Idili, A.; Porchetta, A.; Ricci, F. Electronic control of DNA-based nanoswitches and nanodevices. *Chem. Sci.* **2016**, *7*, 66–71. [CrossRef] [PubMed]
30. Guz, N.; Fedotova, T.A.; Fratto, B.E.; Schlesinger, O.; Alfonta, L.; Kolpashchikov, D.M.; Katz, E. Bioelectronic interface connecting reversible logic gates based on enzyme and DNA reactions. *ChemPhysChem* **2016**, *17*, 2247–2455. [CrossRef] [PubMed]
31. Fan, C.; Plaxco, K.W.; Heeger, A.J. Electrochemical interrogation of conformational changes as a reagentless method for the sequence-specific detection of DNA. *Proc. Natl. Acad. Sci. USA* **2003**, *100*, 9134–9137. [CrossRef] [PubMed]
32. Ge, L.; Wang, W.; Sun, X.; Hou, T.; Li, F. Versatile and programmable DNA logic gates on universal and label-free homogeneous electrochemical platform. *Anal. Chem.* **2016**, *88*, 9691–9698. [CrossRef] [PubMed]
33. Frank-Kamenetskii, M.D.; Mirkin, S.M. Triplex DNA structures. *Annu. Rev. Biochem.* **1996**, *392*, 65–95. [CrossRef] [PubMed]
34. Idili, A.; Vallée-Bélisle, A.; Ricci, F. Programmable pH-triggered DNA nanoswitches. *J. Am. Chem. Soc.* **2014**, *136*, 5836–5839. [CrossRef] [PubMed]
35. Amodio, A.; Zhao, B.; Porchetta, A.; Idili, A.; Castronovo, M.; Fan, C.; Ricci, F. Rational design of pH-controlled DNA strand displacement. *J. Am. Chem. Soc.* **2014**, *136*, 16469–16472. [CrossRef] [PubMed]
36. Idili, A.; Porchetta, A.; Amodio, A.; Vallée-Bélisle, A.; Ricci, F. Controlling hybridization chain reactions with pH. *Nano Lett.* **2015**, *15*, 5539–5544. [CrossRef] [PubMed]
37. Zhang, D.Y.; Turberfield, A.J.; Yurke, B.; Winfree, E. Engineering entropy-driven reactions and networks catalyzed by DNA. *Science* **2007**, *318*, 1121–1125. [CrossRef] [PubMed]
38. Zhang, D.Y.; Winfree, E. Control of DNA strand displacement kinetics using toehold exchange. *J. Am. Chem. Soc.* **2009**, *131*, 17303–17314. [CrossRef] [PubMed]
39. Srinivas, N.; Ouldridge, T.E.; Šulc, P.; Schaeffer, J.M.; Yurke, B.; Louis, A.A.; Doye, J.P.K.; Winfree, E. On the biophysics and kinetics of toehold-mediated DNA strand displacement. *Nucleic Acids Res.* **2013**, *41*, 10641–10658. [CrossRef] [PubMed]
40. Green, S.J.; Lubrich, D.; Turberfield, A.J. DNA Hairpins: Fuel for autonomous DNA devices. *Biophys. J.* **2006**, *91*, 2966–2975. [CrossRef] [PubMed]
41. Machinek, R.R.F.; Ouldridge, T.E.; Haley, N.E.C.; Bath, J.; Turberfield, A.J. Programmable energy landscapes for kinetic control of DNA strand displacement. *Nat. Commun.* **2014**, *5*, 5324. [CrossRef] [PubMed]
42. Morrison, L.E.; Stols, L.M. Sensitive fluorescence-based thermodynamic and kinetic measurement of DNA hybridization in solution. *Biochemistry* **1993**, *32*, 3095–3104. [CrossRef] [PubMed]
43. Dunn, K.E.; Trefzer, M.A.; Johnson, S.; Tyrrell, A.M. Investigating the dynamics of surface-immobilized DNA nanomachines. *Sci. Rep.* **2016**, *6*, 29581. [CrossRef] [PubMed]

44. Kohne, D.E.; Levison, S.A.; Byers, M.J. Room temperature method for increasing the rate of DNA reassociation by many thousandfold: The Phenol Emulsion Reassociation Technique. *Biochemistry* **1977**, *16*, 5329–5341. [CrossRef] [PubMed]
45. Dave, N.; Liu, J. Fast molecular beacon hybridization in organic solvents with increased target specificity. *J. Phys. Chem. B* **2010**, *114*, 15694–15699. [CrossRef] [PubMed]
46. Bercovici, M.; Han, C.M.; Liao, J.C.; Santiago, J.G. Rapid hybridization of nucleic acids using isotachophoresis. *Proc. Natl. Acad. Sci. USA* **2012**, *109*, 11127–11132. [CrossRef] [PubMed]
47. Han, C.M.; Katilus, E.; Santiago, J.G. Increasing hybridization rate and sensitivity of DNA microarrays using isotachophoresis. *Lab Chip* **2014**, *14*, 2958. [CrossRef] [PubMed]
48. Schoen, I.; Krammer, H.; Braun, D. Hybridization kinetics is different inside cells. *Proc. Natl. Acad. Sci. USA* **2009**, *106*, 21649–21654. [CrossRef] [PubMed]
49. Mortensen, U.H.; Bendixen, C.; Sunjevaric, I.; Rothstein, R. DNA strand annealing is promoted by the yeast Rad52 protein. *Proc. Natl. Acad. Sci. USA* **1996**, *93*, 10729–10734. [CrossRef] [PubMed]
50. Goodman, R.P. NANEV: A program employing evolutionary methods for the design of nucleic acid nanostructures. *Biotechniques* **2005**, *38*, 548–550. [CrossRef] [PubMed]
51. Chatterjee, G.; Dalchau, N.; Muscat, R.A.; Phillips, A.; Seelig, G. A spatially localized architecture for fast and modular DNA computing. *Nat. Nanotechnol.* **2017**, *12*, 920–927. [CrossRef] [PubMed]