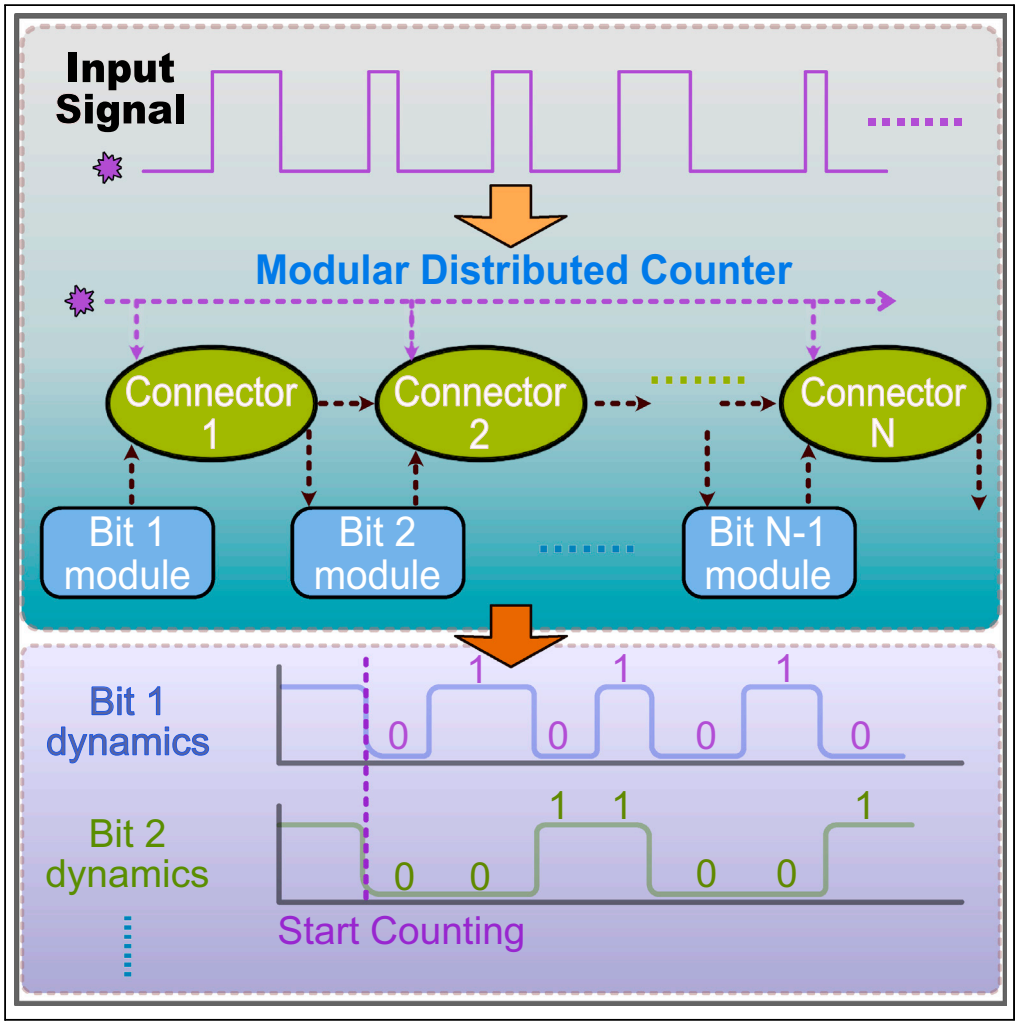
iScience

CellPress
OPEN ACCESS

Article

A synthetic distributed genetic multi-bit counter



Tianchi Chen, M. Ali Al-Radhawi, Christopher A. Voigt, Eduardo D. Sontag

e.sontag@northeastern.edu

Highlights

A single-bit counter is designed for a repressor-based genetic circuit

A scalable multi-bit counter is enabled by distributing the design across cells

A computational optimization framework is proposed to guide the design

Chen et al., iScience 24, 103526
December 17, 2021
https://doi.org/10.1016/j.isci.2021.103526

iScience

CellPress
OPEN ACCESS

Article

A synthetic distributed genetic multi-bit counter

Tianchi Chen,[1,5] M. Ali Al-Radhawi,[2,5] Christopher A. Voigt,[4] and Eduardo D. Sontag[1,2,3,6,*]

SUMMARY

A design for genetically encoded counters is proposed via repressor-based circuits. An *N*-bit counter reads sequences of input pulses and displays the total number of pulses, modulo 2^N. The design is based on distributed computation with specialized cell types allocated to specific tasks. This allows scalability and bypasses constraints on the maximal number of circuit genes per cell due to toxicity or failures due to resource limitations. The design starts with a single-bit counter. The *N*-bit counter is then obtained by interconnecting (using diffusible chemicals) a set of *N* single-bit counters and connector modules. An optimization framework is used to determine appropriate gate parameters and to compute bounds on admissible pulse widths and relaxation (inter-pulse) times, as well as to guide the construction of novel gates. This work can be viewed as a step toward obtaining circuits that are capable of finite automaton computation in analogy to digital central processing units.

INTRODUCTION

We introduce a new design of counters: an *N*-bit counter which reads sequences of input pulses and keeps a tally of how many pulses have been seen until now, modulo 2^N; for example, when $N = 1$ the counter simply computes the parity (odd or even) of the number of pulses seen so far.

The study of combinations of circuits and memory in living cells was initiated by Subsoontorn and Endy (2012), Kim et al. (2019), Siuti et al. (2013), and Purcell and Lu (2014). A key step toward building artificial genetic memory was the design of the synthetic toggle switch (Gardner et al., 2000). Realizing the counting capability requires more elaborate logical operations. An early synthetic genetic counter that can count up to three induction events was based on transcriptional and recombinase-based cascades (Friedland et al., 2009). Also based on recombinases and information stored in DNA structure are the counters proposed by Subsoontorn and Endy (2012) and, more recently, by Zhao et al. (2019). A pulse detecting circuit that responds only at the falling edge of a pulse was proposed by Noman et al. (2016).

In this work, we present an *in silico* scalable and distributed counter design based on standard gene repressor networks. The design that we propose can be theoretically used to count modulo 2^N for an arbitrary fixed integer *N*. However, practically implementing such an approach, even for very small *N*, faces a hurdle due to the practical impossibility of placing a large number of gates in single cells. It is known that large circuits can place stress on cells and overload them, causing failure of the design (Borkowski et al., 2016). This has led to the theoretical "retroactivity" and to the construction of feedback mechanism to avoid their effects (Del Vecchio et al., 2008, 2018). To avoid this issue, we base our design on distributed computation, with specialized cell types allocated to specific tasks, such as the computation of carry bits. This is an approach that we have also proposed for designing large classes of Boolean functions (Al-Radhawi et al., 2020). The communication between the different types of cells can be in principle implemented by diffusible small molecules such as quorum-sensing molecules. Distributed computation allows one to bypass constraints on the maximal possible number of circuit genes per cell, thus avoiding toxicity and making the design more scalable. (We assume relatively fast diffusion in a well-mixed environment.) To respect current experimental constraints (Gander et al., 2017; Nielsen et al., 2016), the maximum number of logical gates per cell is set to be seven. Our design is based upon a systematic and scalable optimization framework which aims to pick gate parameters so as to provide a degree of robustness, and which is especially suited to distributed implementations.

Outline of approach

Our design process will follow the paradigm adopted by Nielsen et al. (2016), which is well suited to the design of transcriptional circuits. Promoters and their associated genes are traditionally specified as pairs or larger

[1]Department of Bioengineering, Northeastern University, Boston, MA 02115, USA

[2]Department of Electrical and Computer Engineering, Northeastern University, Boston, MA 02115, USA

[3]Laboratory of Systems Pharmacology, Program in Therapeutic Science, Harvard Medical School, Boston, MA 02115, USA

[4]Department of Biological Engineering, Massachusetts Institute of Technology, Cambridge, MA 02139, USA

[5]These authors contributed equally

[6]Lead contact

*Correspondence: e.sontag@northeastern.edu

https://doi.org/10.1016/j.isci.2021.103526

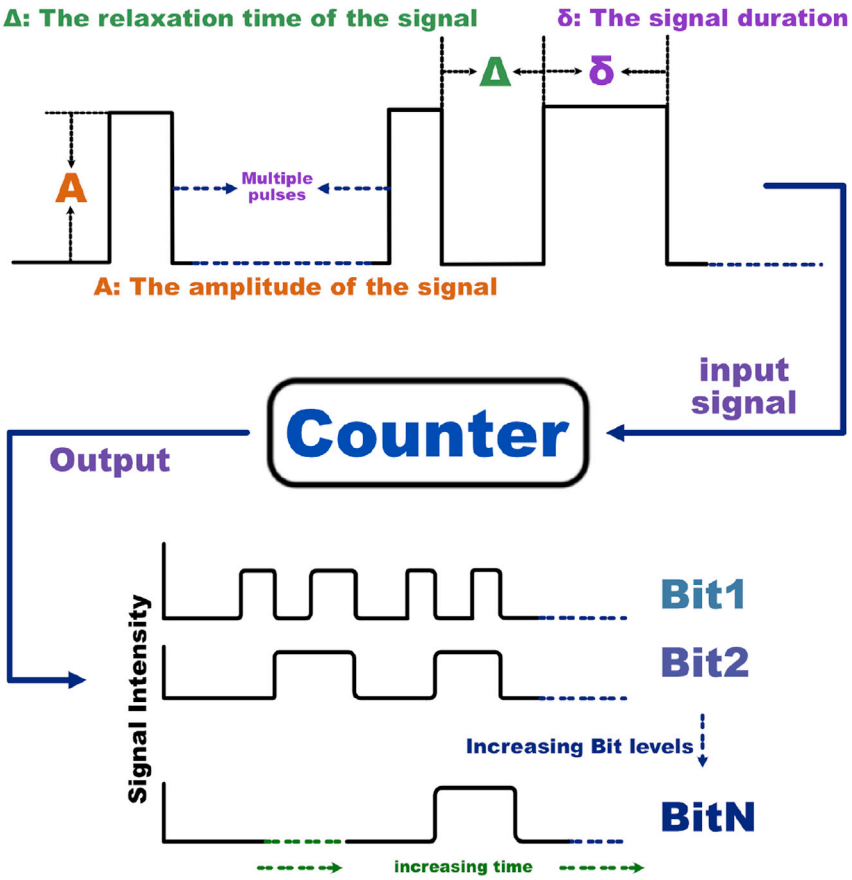


Figure 1. A schematic view of a multi-bit counter

A general *N*-bit counter takes an input pulse train as an input, and outputs a modulo-2^N count of the total number of pulses, represented by the binary state of each bit (right). The pulse trains will be assumed to respect constraints on pulse width δ, spacing (relaxation time that permits the system to approach steady state until the next pulse.) between signals Δ, and pulse amplitudes *A*.

groups in which a single or tandem promoter lies in front of a coding region. In this approach, a biological gate consists of a gene coding region, together with the promoter region of a target gene that is regulated by the promoter of another gene. Mathematically, the inputs and the outputs of each gate are promoter levels rather than repressor or protein levels.

Viewing gates in such a manner makes it easier to obtain expressions for a multi-input gate response, each quantified by its relative promoter strength. A more complicated cellular network can be made up of interconnections of such gates, each using promoter strengths as inputs and a promoter strength as its output.

Using the proposed paradigm, we design first a single-bit counter, i.e. a parity checker. When exposed to a sequence of *M* pulses of an external input (which might be a chemical inducer or a physical signal such as light at a specific frequency), the network will produce a binary output to indicate if the number *M* of pulses seen so far is even or odd. The single-bit counter uses an SR latch for implementing the memory function (Andrews et al., 2018). Our design is asynchronous, meaning that it will detect pulse signals that are not equally spaced, though with a minimal separation time. The pulses themselves will be subject to specifications of minimal and maximal width. The single-bit counter serves as the key component of an *N*-bit counter, in which a count modulo 2^N of the observed number of pulses is stored and displayed. The *N*-bit counter, as illustrated in Figure 1, can be constructed using *N* single-bit counters together with additional gates that implement the "carry" operations.

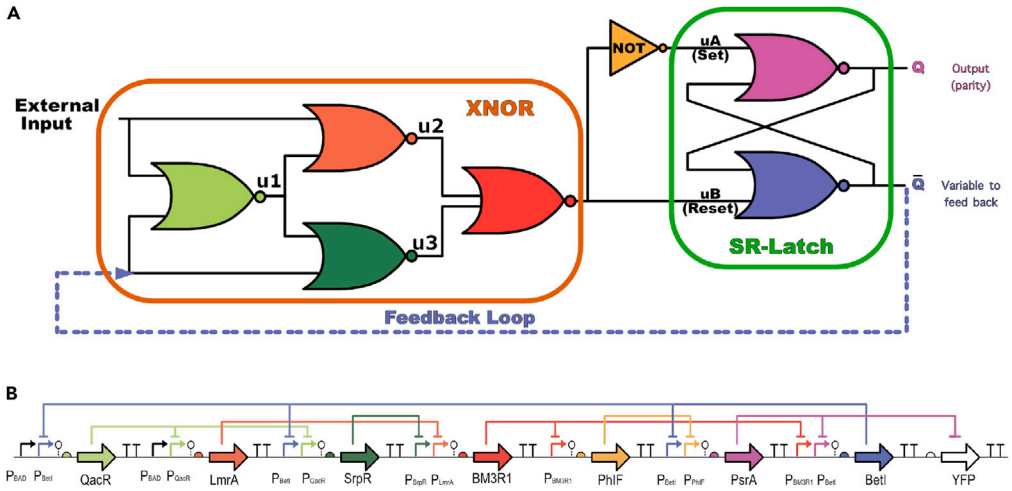


Figure 2. The single-bit counter design

(A) Circuit design diagram for the single-bit counter. Shown is a seven NOR-gate implementation of the parity checker. This parity checker is a single-bit counter that contains two components. An XNOR component is shown inside the orange box, and an SR-latch component is shown in the green box. The design has a feedback loop from $\overline{Q}$ to the XNOR gate. The other input of the XNOR gate is the external input signal. The output of the single-bit counter is Q, which is the parity of the external input.

(B) A genetic network implementation of a single-bit counter which consists of insulated gates. The diagram shows the promoter-gene-repressor network of one possible plasmid implementation in an *E. coli* system. Repressor colors shown in this genetic construct are matched with the gate colors shown in (A). All the terminators are shown as a black "TT".

RESULTS

The single bit counter circuit

Digital design construction

Using digital circuit design methods (Hardy and Steeb, 2001; Jaeger and Blalock, 1997; Bostock, 1988), we propose a design for the single-bit counter as shown in Figure 2A.

There are seven NOR gates in the design (the NOT gate is a NOR gate with the input repeated). This design consists of two components: an XNOR gate and an SR latch. One of the XNOR gate inputs is the external input to the counter, and in our application it will consist of a sequence of pulses. The other input of the XNOR gate is fed back from one of the SR latch outputs. The SR latch module functions as a memory switch and the XNOR gate is a comparator whose output is one if and only if the inputs are identical.

The design in Figure 2 functions as a binary counter in the following sense. Let us suppose that we start with the circuit at a steady state in which Q is low, and $\overline{Q}$ is high, which we think of as the "zero" or even parity state. This state should be stable. When an external signal changes from low to high, the output steady-state of the circuit will switch from $(Q, \overline{Q})$ = (low,high) to $(Q, \overline{Q})$ = (high,low), which is interpreted as "one" or "odd parity". If the input signal goes back to zero soon enough, the circuit's output will stay at this steady state, until the next input signal triggers a switch back to even parity, and so forth.

As an illustration, we show an example of how we would implement a single-bit counter in *Escherichia coli*. By writing the binary state transition table for the logical circuit of the single-bit counter shown in Figure 2A, we have designed an *in silico* genetic network using Cello Nielsen et al. (2016); Andrews et al. (2018) as an initial step. Then, we have used an optimization framework proposed in this paper to improve the aforementioned design by maximizing the range of admissible pulse widths. The final circuit is depicted in Figure 2B.

Circuit behavior

Modeling framework

In order to simulate the behavior of the genetic circuit quantitatively, we write an ordinary differential equation (ODE) model at the translational level. We use a standard Hill function to characterize the input-output

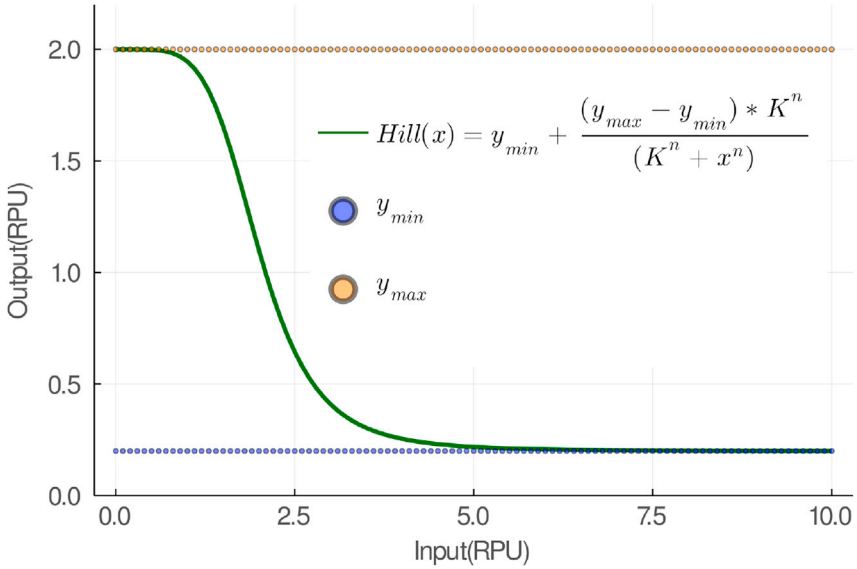


Figure 3. The response function of an arbitrary gate written as a shifted Hill function

The parameters y_{min} and y_{max} set the lower and upper bounds the response function output. The parameter K sets the width of the response range for the input. The parameter n controls the steepness of the response function.

response function of the gate Nielsen et al. (2016). For each gate, we use the following general ODE to model its dynamics:

$$\frac{dm_{gate}}{dt} = \text{Hill}(p, x) - \gamma m_{gate}, \qquad \text{(Equation 1)}$$

where m_{gate}, p, x, γ are the concentration of the mRNA, the gate parameters, the inputs, and the mRNA decay rate, respectively. An archetypical Hill function and its parameters are shown in Figure 3.

The overall ODE model of the circuit is given by interconnecting the individuals gates via their inputs and outputs as is given in the STAR methods section.

We checked through simulations that our design in Figure 2B works as a parity checker, see Figure 5. We study the stability of the single-bit counter and the circuit's dynamical response to various external inputs signals in the following subsections.

Circuit stability

The first property to be checked is that the system should reliably store the memory of the last state (even or odd) whenever the input signal stays at zero. This means that the autonomous (unexcited) system needs to be bistable, meaning that all solutions will generically converge to one of two states: high output or low output.

In the single-bit counter circuit shown in Figures 2B and 2A system of the ODEs with seven state variables describing a seven-repressor circuit is used for modeling the dynamics of the circuit evolution. For the output gates of the circuit, before understanding how its dynamics are affected by an external input signal, one needs to check first whether the outputs of the circuit produce binary values. In order to numerically check the bistability of the circuit, we randomly initialize all the seven state variables (repressor gate concentrations) and simulate the trajectory of a 7-dimensional differential equation until it settles on a steady state. The system's outputs are the states of the SR latch: $\mathbf{Q}$, $\overline{\mathbf{Q}}$. Figure 4 shows the projected trajectories of $(\mathbf{Q}, \overline{\mathbf{Q}})$ where all states asymptotically converge to either a high $\mathbf{Q}$ (PsrA) or a high $\overline{\mathbf{Q}}$ (BetI) value at steady state (but not both).

Counting capability

A functioning single-bit counter must switch outputs if the input stays high for a sufficiently long duration of time. The duration of the input signal pulse δ and the relaxation time Δ (when the signal is off) are both

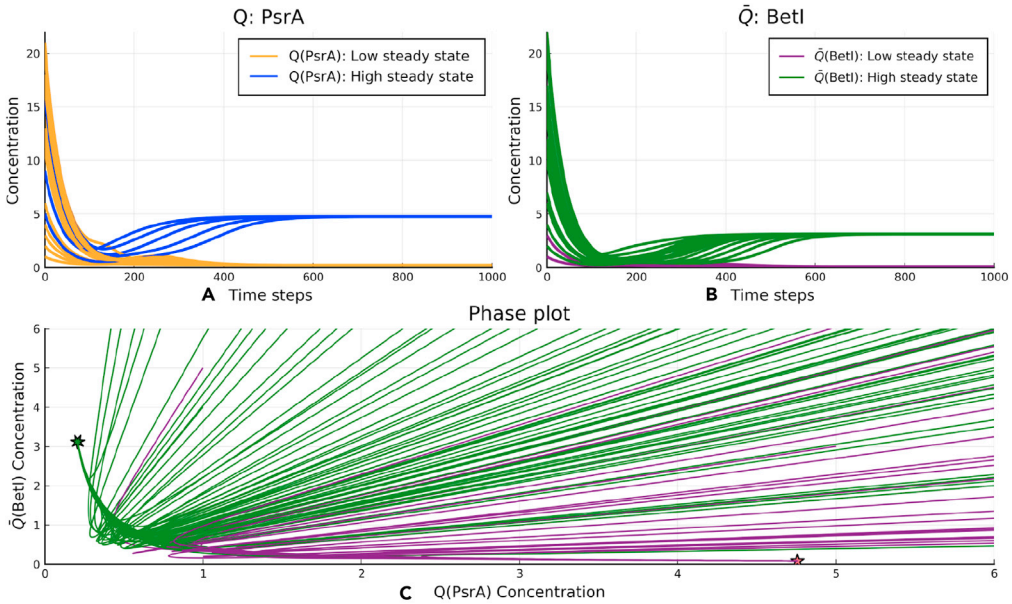


Figure 4. The bistability verification of the single-bit counter

(A and B) With 50 randomized initial conditions, the SR latch states ($Q, \overline{Q}$), which determine the circuit's output, asymptotically settle into one of the two steady states. The higher value steady state (for both Q or $\overline{Q}$) represents the binary "1" state, and the lower value steady state represents the binary "0" state.

(C) In the phase plot with random initial conditions, trajectories converge to one of the two stable steady states, represented here by two solid stars. Note that the intersecting trajectories represent projections of the original disjoint 7-dimensional trajectories. The two steady-state attractors are shown with stars in the ($Q, \overline{Q}$) phase plane.

critical factors that affect the correct functioning of the parity checker (Figure 1). It is necessary for the pulses to be long enough to allow the SR-latch to flip, yet not too long, since a long input will flip the latch back to its original state. Similarly, the interval between pulses should be large enough to allow the protein concentrations to settle into their steady states.

A key part of the design concerns the characterization of the types of pulses that the counter can reliably count. This means finding the ranges of the two parameters δ, Δ (Figure 1), which are the pulse duration and the relaxation time, respectively. To that end, we have performed simulations to estimate the empirical duration of the constant input signal that guarantees that when the input pulse turns from high to low, the system stays in the newly switched steady state. Using an optimization method, we found that when the pulse duration is in the range of 300–550 min, the switching behavior is persistent and stable. We will revisit the issue of parameter selection later in the article.

Figure 5A shows an example illustrating how our circuit responds to an input which is a square wave signal with a fixed duration and randomly varying relaxation times. When the circuit is at a steady state and an input signal is applied, the system output switches state. When the input signal is at a low value, the circuit stays in the new switched state until a new pulse arrives to switch the output to the previous steady state.

Constant inputs lead to oscillations. In our proposed design, a high external input continuously tries to flip the output as per the state transition diagram (STAR methods, Table 1). Hence, we expect the output to show a stable oscillatory behavior when the external constant input stays high. Such a design is reminiscent of the standard oscillator architecture consisting of a negative-loop with a delay. We indeed confirmed the oscillatory behavior of the circuit by simulations in Figure 5B for the gates listed in Table 2.

Furthermore, we have extensively explored the oscillatory behavior of the circuit dynamics for a constant input signal. The kinetic constant K, the cooperativity index n, and the maximum value y_{max} of the Hill function are the three most important parameters for the circuit dynamics. A 3D bifurcation plot in the phase

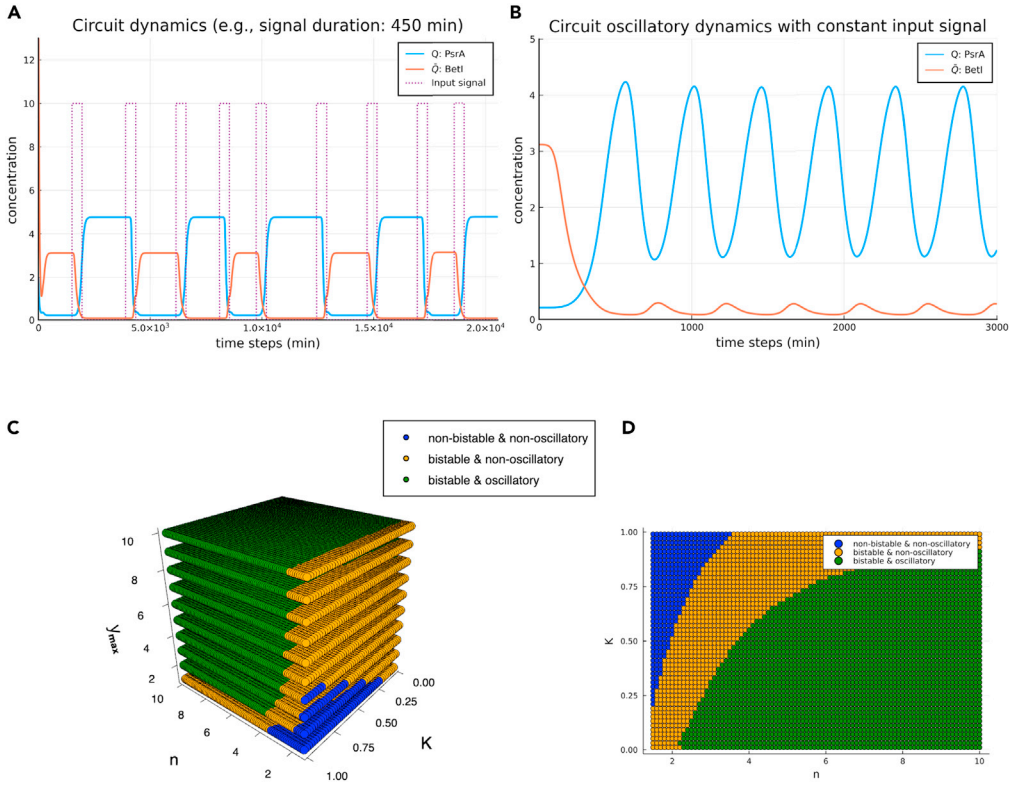


Figure 5. The single-bit counter dynamics and parameter space

(A) The behavior of the single-bit counter under a pulse train input. The circuit's output (represented by the gate **Q**) keeps track of the input's parity correctly, as does the negated output $\overline{\mathbf{Q}}$. In this particular example, the input pulse has a duration of 450 min but with random relaxation times between adjacent pulses.

(B) A demonstration of a stable oscillatory trajectory displayed by the counter when subjected to an external constant input. With the input's amplitude $A = 10$, the circuit's outputs **Q** and $\overline{\mathbf{Q}}$ oscillate with 180° phase difference.

(C) A 3D bifurcation plot in the (K, n, y_{max}) space. Each point is marked by whether the dynamics is bistable, oscillatory, both, or neither.

(D) A layer of the 3D bifurcation plot in (C) with $y_{max} = 1$.

space spanned by K, n, and y_{max} values is demonstrated in the Figure 6C. Our simulations show that the phase space has three distinct regions, which are "no oscillation and no bistablity", "no oscillation and bistablity", and "oscillation and bistablity".

Multi-bit counter

In the previous section, we have proposed a genetic circuit design for a single-bit counter. This design functions as a universal parity checker that can be re-used in a multi-bit counter design.

Design principle

We design the multi-bit counter system in a highly modular fashion. Using a single-bit counter as the building block module, we construct a multi-bit counter by adopting the following pattern: **1-bit counter – Connector – 1-bit counter**. Hence, a single-bit counter will be implemented repetitively as a module, which simplifies the design process. We will also design connectors to interface single-bit counters for intercellular communication. The overall design can be scaled up, in principle, to any arbitrary number of bits by interconnecting single-bit modules and connectors modules. Such a scaled-up design can be achieved as long as enough orthogonal diffusible small molecules such as quorum-sensing molecules are available.

Our design of the multi-bit counter enjoys a degree of robustness, which we will quantify using the trade-off between the mean value of the pulse duration time (δ-mean) and the standard deviation of the set of all

Table 1. State transition table of the open-loop circuit

X	Y	Q	$\overline{Q}$
0	0	1	0
0	1	0	1
0	0	1	0
0	1	0	1
1	1	1	0
1	0	0	1
1	1	1	0
0	1	0	1
0	0	1	0
1	0	0	1

possible pulse duration times (δ-std). In order to avoid a computationally infeasible combinatorial search among different gate parameters, we first mathematically study the idealized case in which every gate has the same characteristics (Hill coefficient, y_{min}, y_{max}, K, n). Later in the paper, we will discuss designing the counter with gates that have different response functions.

Modular design construction

Following the general principle outlined above, we propose the following scheme for constructing a multi-bit counter system. Any design of the multi-bit counter system consists of the following three essential modules. Different modules talk with each other via diffusible small molecules.

a) **Module 1: The parity checker.** This module is the building block for constructing a multi-bit counter system, and it is responsible for storing and updating the value of the corresponding bit. For example, if one wants to build a 3-bit counter, we will use three copies of the first module, and each one will be used for one of the bits.

b) **Module 2: Connector type 1.** This specific module uses one diffusible small molecule connecting the first parity checker and the second parity checker only. As shown in Figure 6, the inputs are the external signal and the first parity checker's output. The output of the connector will be communicated to the parity checker and the connector at the next stage by diffusible small molecules.

c) **Module 3: Connector type 2.** This module serves as the universal connector that communicates between the remaining parity checkers beyond the second parity checker.

With the proposed three modules, Figure 6 shows a schematic view of the design of a generic N-bit counter.

Given the proposed design scheme, the next step is to find the Boolean representation of the two types of connectors. Figure 7 shows how to find the corresponding Boolean function for connectors using a truth table. We illustrate this process by finding the connector type 1 Boolean function as an example. The figure lists all four stages of the possible states of a 2-bit counter and the state transition table associated with each stage. For example at stage 2 in Figure 7, the counter is at its internal state B_2B_1 = "0–1". When the input signal changes from "0" to "1", the 2-bit counter should switch from "0–1" to "1 - 0". The connector takes the external signal and the first bit as inputs, and its output will be "1" when its inputs are "1-1". Similarly, we can repeat the same process for the rest of the counting situations to derive the desired output for all possible inputs. As for the simplest case, we can directly see that the Boolean function of the connector type 1 is just an "AND" gate. Similarly, the Boolean function that represents the connector type 2 can also be determined from the requirement of the binary representation of the state transition table at the higher connector levels beyond the second-bit level.

With the connector type 1 functioning as an "AND" gate, one is able to implement such a connector with two NOT gates and one NOR gate as shown in Figure 8A.

Table 2. Gate number - biological gate mapping

Gate number	Biological gate
1	QacR
2	LmrA
3	SrpR
4	BM3R1
5	PhlF
6	PsrA
7	BetI

Schematics of the 2-bit counter

In Figure 9A, we show the schematic design for a 2-bit counter that is constructed by combining two parity checkers and a connector type 1. Figure 9B demonstrates the counting capability of a design for pulses that fall within the proper ranges.

The persistent comb-like structure in the first row indicates that the carry bit information of the first-bit has been successfully transmitted to the second-bit. To examine if the chosen parameter set can give rise to a working 2-bit counter, we first randomize the initial conditions for all state variables in the 2-bit counter dynamical equations and wait for the counter to equilibrate to one of its steady states. Then we give the system an external input signal, which is a train of short pulses. To capture the nature of the input signal's noisy environment, we add noise to the pulse widths to simulate asynchronous counting scenarios, which would mimic more realistic biological situations. Specifically, we allow the Δ s to be randomly sampled from the range of $[\Delta_0, 2\Delta_0]$, with a given relaxation time Δ_0.

A 2-bit counter will follow the pattern of "0 - 0", "0–1", "1 - 0", "1 - 1", and then return to "0 - 0", which corresponds to counting input pulses from 0 to 4. As illustrated in Figure 9B, we show a full cycle trajectory of a 2-bit counter. Each pulse that presents as an input to the counter will increase the internal state of the counter.

Initialization scheme

To determine the initialization scheme for a multi-bit counter, we consider two issues. The first one is whether the counter has two distinct steady state solutions for the output Q. As demonstrated in Figures 5C and 5D, the circuit is bistable for a wide range of parameters, and all the trajectories (with randomly-chosen initial conditions) converge to a steady state. The second issue is when should counting start and what constitutes a zero. This is a major issue when designing dynamical circuits. The problem is that it may be difficult to set a system, and our counter in particular, to a desired initial state. We have solved this problem as follows. The system should be preconditioned by giving it a sequence of training pulses. The initial time is then set as the first time that the last bit switches. Extensive simulations with random initial conditions show that this always results in the correct initialization after a few (no more than 5 for most of the time, for a three bit counter discussed below) pulses. An alternative approach is to define "zero" as the initial state of the counter, and the actual events counted can be computed as the difference between the state of the counter at the current and the initial state. Yet another approach would be to include a "reset button" in the design, at the cost of adding a constitutive promoter that is controlled by an additional inducer which, when applied, sets the initial state.

Schematics of the 3-bit counter

We take the previous design a step further and showcase a 3-bit counter built by implementing an additional connector type 2 and a third single-bit counter on top of the 2-bit counter system.

We introduce the connector type 2 as a universal connector module for connecting single-bit counters beyond the second single-bit counter in a general multi-bit setup. Figure 8B shows a schematic view of the connector type 2 using NOT and NOR gates. Similar to the connector type 1 module, we can identify the Boolean function that the connector type 2 represents. In a 3-bit counter system, the corresponding transient state table for the Boolean function is similar to the one depicted in Figure 7, but is much larger

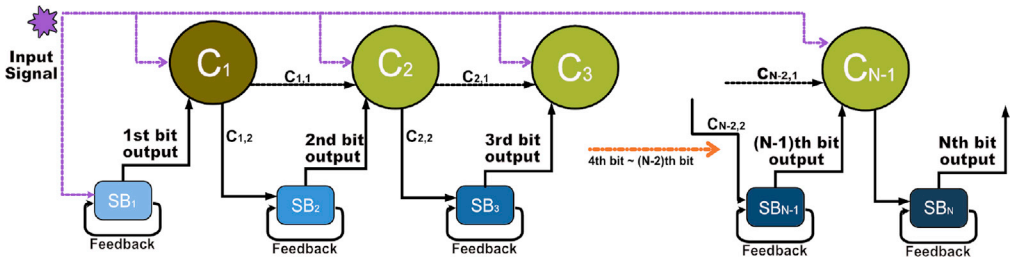


Figure 6. Schematic design of a generic modular multi-bit counter

The counter employs three types of modules, each implemented in a different cell type. One connector of type 1 and $N-2$ connectors of type 2 (colored in dark and light green respectively) are shown in the first row and N single-bit counters (colored for different bit levels gradient blue) are in the second row. The external input signal is colored in dashed purple lines, and it acts on all connectors simultaneously. Black arrows denote communication via diffusible small molecules. Each connector outputs two signals $C_{i,1}$, $C_{i,2}$ at ith bit level. The $C_{i,1}$ is the AND operation result between the previous bit's output and the previous connector's output. The $C_{i,2}$ is the AND operation result between the $C_{i,1}$ and the external signal.

as it consists of 8 stages. The result is that a connector type 2 consists of a cascade of two AND gates. Each AND gate is realized by 2 NOT gates and 1 NOR gate. The first AND gate takes two inputs, which are the outputs of the two preceding parity checkers. The second AND gate takes the output from the first AND gate and the external signal as inputs and produces the connector type 2's output for the next bit level by using a diffusible small molecule.

Similar to the 2-bit counter, we show a schematic design and the dynamics for each module of the 3-bit counter in Figure 10. An example of a cycle that successfully counts the number of pulses from 0 to 8 is shown in Figure 10B. As with the two-bit counter, the initial time is set by the first time that the last bit switches, after a preconditioning set of pulses. Therefore, the initial counter state "0-0-0" is defined as the state during the times between the dashed lines labeled "start" and "P1" (see Figure 10B).

For a multi-bit counter system with more than three bits, one can easily generalize the counter system and build upon the 3-bit counter by adding additional modules of connector type 2 and single-bit counters as we have demonstrated in Figure 6.

Scalability of the counter. So far, we have designed a single bit counter, a 2-bit counter, and a 3-bit counter. The design principle that we proposed can be applied to the design of a distributed N-bit counter. As a demonstration, we showcase the dynamics of an 8-bit counter in Figure S5.

For a large N-bit counter, the variation in the timing of the input signal that reaches to each module (a connector or a parity checker) can be an important factor when building a feasible counter experimentally. Depending on whether the actual cellular colonies are grown on agar or in a mixed liquid culture, the tolerance of the variation for the input signal for each bit level may depend on the spatial distribution of each module's cellular colony, but also on the viscosity of the liquid, the temperature, and the Hill coefficients of the gates.

Design selection graphs

In the previous sections, we described a generic way of constructing a multi-bit counter. We have simulated wide ranges of parameter values and produced databases for different multi-bit counters (2-bit and 3-bit counters), see Figure 11. In the generated database, many parameter sets can give rise to a working counter. However, in synthetic biology, experimentalists often have a limited selection of genetic gates, and these gates usually respond to a restrictive dynamic range of input. For simplicity, we first assume all the activation functions to be identical. Then, we sample gates from the generated multi-bit counters' database to produce generically working counters in which each gate has its own activation function.

From an experimental point of view, it is desirable to pick genetic components so as to obtain a robust counter. Toward this goal, we propose a method for optimizing the selection process when limited genetic components are available. Such a process will allow designing the counters based on availability of parameters. The parameters that are essential for constructing a feasible multi-bit counter are the shifted Hill

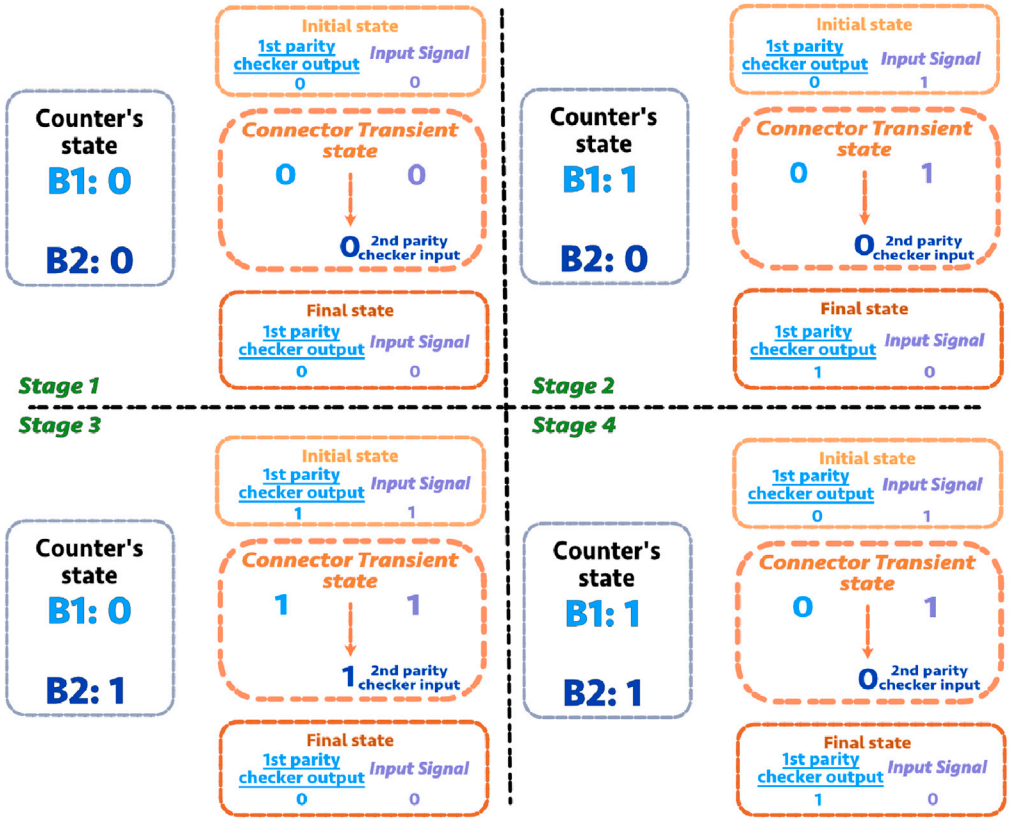


Figure 7. Connector type 1 state transition table

Four stages of the 2-bit counter are shown. The state of 2-bit counter can be represented by the expression of $B1 * 2^0 + B2 * 2^1$. **B1** represents the first-bit binary state, and **B2** represents the second-bit binary state. **B1** and **B2** are either "1" or "0". On the right side of each quadrant in the orange outline, the connector transient state table shows how the first parity checker's output and the external signal input change the second parity checker's input. Each stage (each quadrant in the figure) contains four pieces of information: 1) The binary representation of the steady state for each bit (the first-bit is denoted by **B1** and the second-bit is denoted by **B2**). 2) The initial state of the first-bit output and the input signal. 3) The connector transient state (the middle orange dashed circle). 4) The final state of the first-bit output and the input signal.

coefficients y_{min}, y_{max}, K, n, the signal amplitude A, and the pulse width δ. A general response function (shifted Hill function) has been illustrated in Figure 3.

We make design selection graphs that can help with experimental design in two ways. First, given a physical genetic gate with a kinetic parameter (K) and a cooperativity index (n), our formalism will estimate an optimal operating range for the input signal and how robust the pulse width will be against perturbations. Second, if we are instead given desired durations of external signals, we can find a range of parameters of a genetic gate that results in a functioning counter. If such a gate is not yet available, this will suggest how to design a new synthetic gate according to the desired Hill parameter sets.

For each parameter set of the Hill function, we sample many possible input signals with different amplitudes and durations. We then calculate the mean values and the standard deviations of the subset of the simulated pulse widths (δ) that gives rise to a working counter as shown in Figure 11 A. Figure 11B shows currently available synthetic gates in the *E. coli* system, and each gate is associated with a unique (n, K) pair. For instance, given the current knowledge of the available gates listed in Figure 11B, one can select an optimal gate parameter set given constraints that $n<3$, $K<0.5$ that has the highest robustness. By "robustness," we mean: given a mean value of a pulse width δ, the larger standard deviation around this mean value, the more robust the counter would be because the larger value of δ standard deviation corresponds to more available counters in the (n, K) space. Given a particular set of parameters for the gates, one can also obtain the optimal pulse width for assessing the counter's robustness.

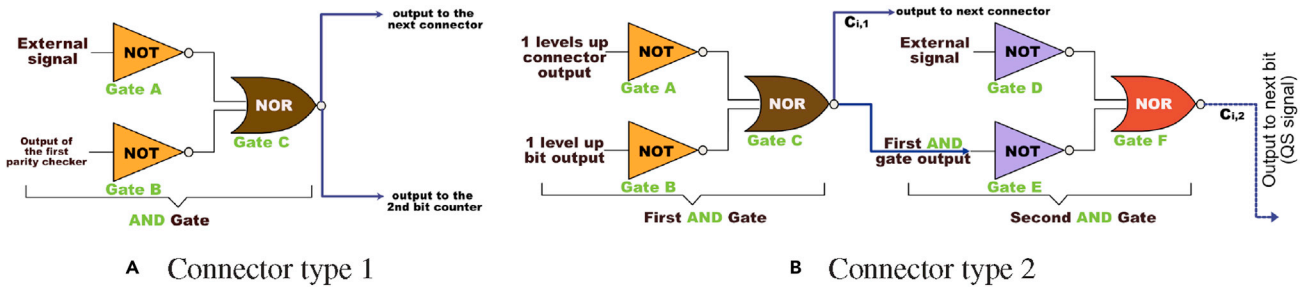


Figure 8. The connectors' design

(A) Connector type 1. This module will be used to interface the first and the second parity checker. The depicted design implements an AND gate using a 2-input NOR gate.

(B) The connector type 2 module design. The Boolean expression for this module in the multi-bit counter design is represented by a cascade of two AND gates. The first AND gate integrates the signals from the outputs of the preceding two parity checkers. The output of the first AND gate will be one of the inputs of the second AND gate. The other input for the second AND gate is the external signal. The connector 2 output will use a diffusible small molecule for inter-cellular communication.

Once given available genetic gate characteristics, represented by the (n, K) pairs, one can refer to Figure 11 to find the mean external pulse width and the mean level of y_{max} that is required for a design. To quantify how robust a given genetic gate is the standard deviation of δ and y_{max} can provide useful information about the tolerable variations in the duration of the external signal and in the maximum value of the Hill activation function.

In order to guide gate design, we present the data alternatively in Figure 12 for the 1-bit, the 2-bit counter, and the 3-bit counter. It can be noticed that the mean of the pulse width in each counter changes wildly in the (n, K) plane. On the other hand, when comparing the overall shift in the whole (n, K) space, the signal mean distributions are not dramatically different between the 1-bit and 2-bit counters. For those synthetic gates with low cooperativity indices, the counter will operate with signals with lower values for the mean of δ. As for the trend of the standard deviation of δ, we can see from the second column that robustness can be gained by increasing the cooperativity index n.

To understand the trade-off between the mean pulse width (δ mean) and gates' robustness and to understand how such a trade-off evolves as we scale up to a multi-bit counter situation, we show a comparison graph for the 1-bit and 2-bit counters in Figure 13 as an illustration. In this graph, we plot δ-mean vs. δ-std for the 1-bit, the 2-bit counter, and the 3-bit counter, and show the trend of how they shift in the (n, K) plane.

To see how our design selection graphs shed light on experimental synthetic gate design, we compare the available experimental gate parameters with the above analysis. To gain insight for the two commonly used systems, Figure 14 shows the transcriptional gates available for both the *E. coli* and yeast systems in the simulated (n, K) space as shown in Figure 12.

Figure 14 shows that most synthetic gates built for the *E. coli* system display, on average, less robustness than the synthetic gates built for the yeast system. The most robust gate of all experimental gates seems to be the yeast gate with $n = 4.6$ and $K = 0.35$. The plots also show that the currently-available physical gates are not in the most robust region of the space, which leaves lots of room for improvement in synthetic gate design.

From the perspective of designing synthetic gates, the above comparison between the experimentally used genetic gates and the more robust simulated gates can provide guidance when designing new synthetic transcriptional gates.

Different activation functions

The above study covered the ideal case in which the counter system can be built out of gates with identical activation functions. However, typically one must use gates with different parameters. Next, we show a generalization of the aforementioned design with the 1-bit counter database as an example, which allows for implementing gates with different activation functions.

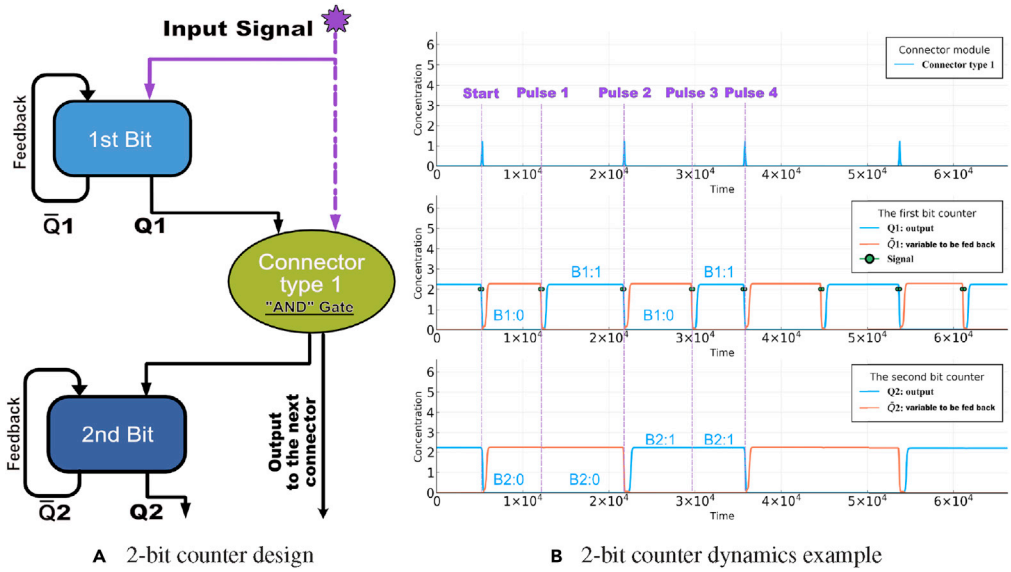


Figure 9. The 2-bit counter design

(A) Schematic design of the 2-bit counter. The signal Q 1 is the output while $\overline{Q}$ 1 is the feedback signal for the first parity checker. The connector type 1 takes Q 1 and the external input signal as inputs. The output of the connector module can be a diffusible small molecule which serves as the input for the second parity checker.

(B) We show an example trajectory of each module aligned with the schematic design in (A). The counter starting time is determined by the first time that the last bit switches state, as explained in the text. Due to the plotted long relaxation time (relative to input pulse width) between adjacent pulses, each green dot in the graph, if zoomed in, represents one input pulse as shown in Figure 5. The representation of the 2-bit counter "B2-B1" is used to indicate the output state "Q 2-Q 1" for the counter. The parameters for this working 2-bit counter are: the scaling factor $\xi = 0.025$, the degradation rate $\gamma = 0.025$, the equilibrium constant $K = 0.011$, the cooperativity index $n = 1.5$, the duration of external signal $\delta = 300$, the amplitude of the external signal $A = 20$, the shifted hill coefficient maximum value $y_{max} = 3$, the shifted hill coefficient minimum value $y_{min} = 0.002$, and the initial and between signals relaxation times $\Delta_0 = \Delta = 5000$.

In Figure 15, we show an example of how one could easily generate various instances of a 1-bit counter in which each gate is associated with a different activation function, specified by (n,K). Given a pulse width δ, we search our generated databases for seven gates that are *simultaneously feasible*. This means that the feasible ranges of pulse widths of the seven selected gates have a maximal intersection. We can see that as long as the seven selected gates have an overlap in their δ distributions, a feasible 1-bit counter can be realized using the selected parameters. For example, we found that, in general, if one only focuses on the range of δ from the 25% quantile to 75% quantile for each gate's δ distribution, it usually can give rise to a feasible 1-bit counter.

The set of feasible parameter sets can be collected in a database as shown in Figure 16 for example. Regarding the uniqueness of the optimal solution for a general N-bit counter, we quantify the optimality of an N-bit counter by δ_{cv} which is the percentage of the coverage of the input pulse width within the counter's maximum allowed δ range. For example, Tables S1 and S2 list δ_{cv} for various circuits. Also, we have generated the corresponding boxplot in Figure S6 for a better visualization. Computationally, the optimal solution for a 1-bit counter is uniquely associated with a particular (n, K) pair. However, for multi-bit counters, there can exist multiple circuits with the same δ_{cv}. This provides experimentalists with more options to accommodate constraints that are not captured by our modeling framework.

In order to achieve a feasible counter which operates according to a specific parameter set, one could perform a conditional sampling to generate a synthetic counter from the database. Using such a strategy for generating feasible 1-bit counters shows much greater flexibility in choosing a different desired range of activation functions. This affords more options to fine-tune the gates' parameters and avoid construction

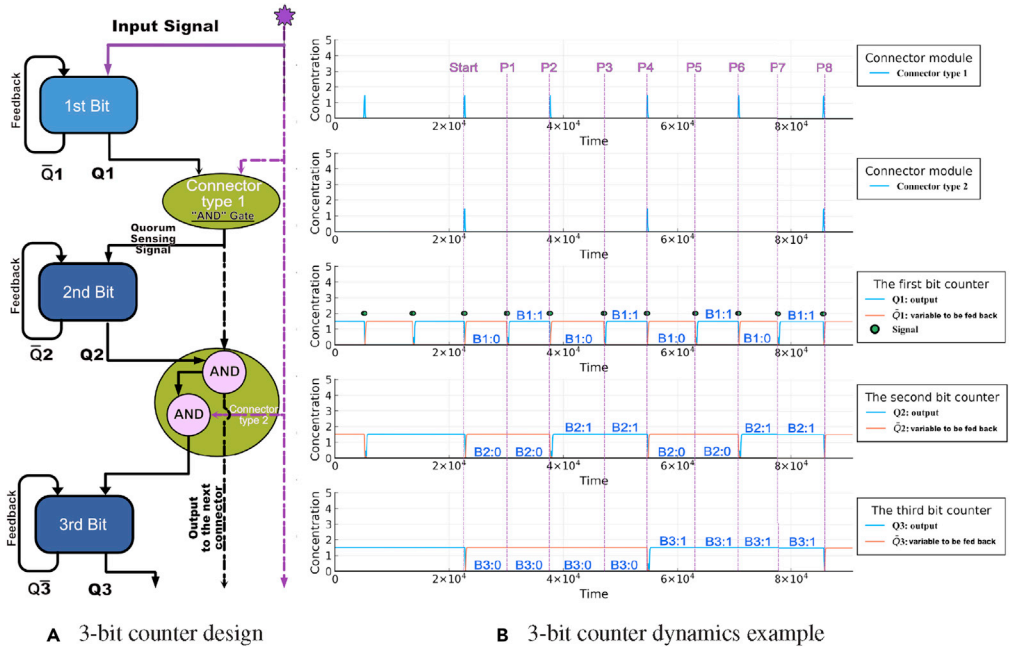


Figure 10. The 3-bit counter design schematics

The inputs are shown in the middle panel, for ease of reference.

(A) The diagram shows a modular design of the 3-bit counter following the pattern of "counter-connector-counter". The only difference between the 3-bit counter and the 2-bit counter is that the second carry bit module uses the connector type 2. **Q** 1, **Q** 2, and **Q** 3 are the outputs of each bit level, and $\overline{\mathbf{Q}}$ 1, $\overline{\mathbf{Q}}$ 2, and $\overline{\mathbf{Q}}$ 3 are the variables to be fed back.

(B) We show an example trajectory for each module. The starting time of the 3-bit counter is set by by the first time that the last bit switches state, as explained in the text. The steady state binary representation for each parity checker **B1**, **B2**, and **B3** are marked on top of the gates' trajectories. The parameters for this 3-bit counter example are: the scaling factor ξ = 0.025, the degradation rate γ = 0.025, the equilibrium constant K = 0.081, the cooperativity index n = 2.81, the duration of external signal δ = 270, the amplitude of the external signal A = 20, the shifted hill coefficient maximum value y_{max} = 1.5, the shifted hill coefficient minimum value y_{min} = 0.002, and the initial and between signals relaxation times $\Delta_0 = \Delta$ = 20,000.

constraints. Moreover, a machine learning classifier could be used to produce feasible 1-bit counter designs with an expected probability associated with each gate.

DISCUSSION

Synthetic biology is an emergent interdisciplinary field of research whose ultimate aim is to design novel biological systems, or to redesign existing ones, with the purpose of achieving new functionalities in medical, energy, environmental, chemical threat detection, and other applications (Cheng and Lu, 2012; Andrianantoandro et al., 2006; Cameron et al., 2014), as well as contributing to the understanding of natural processes. In engineered synthetic circuits, basic cellular components such as genes, mRNAs, proteins, and metabolites are "rewired" in order to construct bio-molecular circuits that realize desired computations and capabilities (Sprinzak and Elowitz, 2005). The resulting devices sense data, such as the presence or absence of particular surface features on a cell being interrogated, or of a chemical in the environment. They perform logical operations on this data and, based on the results of these operations, decide upon appropriate responses – such as the secretion of a toxic chemical in order to kill a cell that has been identified as being part of a tumor or as being infected by a pathogen, or the expression of a fluorescent signal to indicate an undesired trait. Specific application examples include the regulation of cell fate decisions (Sedlmayer et al., 2018; Vogel et al., 2019), building cellular memory (Inniss and Silver, 2013), engineering CAR-T cells for adaptive immunotherapy (Cho et al., 2018), designing epigenetic reading and writing systems (Park et al., 2019), and developing numerous other specialized circuits (Callura et al., 2012; Archer et al., 2012; Elowitz and Leibler, 2000; Yang et al., 2014; Gupta et al., 2013; Hwang et al., 2014; Saeidi et al., 2011; Kohanski and Collins, 2008).

A

		δ		A		Y_max	
		mean	std	mean	std	mean	std
K	n						
0.011	1.5	291.470588	10.014695	20.0	0.0	2.364706	0.429336
	1.6	296.849315	13.781702	20.0	0.0	2.087671	0.584747
	1.7	307.012987	14.873584	20.0	0.0	2.067532	0.608588
	1.8	317.588235	15.991288	20.0	0.0	2.056471	0.604831
	1.9	326.542553	15.692382	20.0	0.0	1.996809	0.579665
...	...	...	...	...	...	...	...
0.091	2.6	286.054217	22.024850	20.0	0.0	2.305422	0.480688
	2.7	290.468750	23.669608	20.0	0.0	2.263021	0.496097
	2.8	294.236111	25.159857	20.0	0.0	2.222222	0.514774
	2.9	296.431624	26.069024	20.0	0.0	2.180769	0.528921
	3.0	297.802419	26.467253	20.0	0.0	2.137500	0.545950

B

	repressor	RBS	Y_min	Y_max	K	n
0	AmeR	F1	0.200	3.8	0.09	1.4
1	AmtR	A1	0.060	3.8	0.07	1.6
2	BetI	E1	0.070	3.8	0.41	2.4
3	BM3R1	B1	0.004	0.5	0.04	3.4
4	BM3R1	B2	0.005	0.5	0.15	2.9
5	BM3R1	B3	0.010	0.8	0.26	3.4
6	HlyIIR	H1	0.070	2.5	0.19	2.6
7	IcaRA	I1	0.080	2.2	0.10	1.4
8	LitR	l1	0.070	4.3	0.05	1.7
9	LmrA	N1	0.200	2.2	0.18	2.1

	repressor	RBS	Y_min	Y_max	K	n
10	PhIF	P1	0.010	3.9	0.03	4.0
11	PhIF	P2	0.020	4.1	0.13	3.9
12	PhIF	P3	0.020	6.8	0.23	4.2
13	PsrA	R1	0.200	5.9	0.19	1.8
14	QacR	Q1	0.010	2.4	0.05	2.7
15	QacR	Q2	0.030	2.8	0.21	2.4
16	SrpR	S1	0.003	1.3	0.01	2.9
17	SrpR	S2	0.003	2.1	0.04	2.6
18	SrpR	S3	0.004	2.1	0.06	2.8
19	SrpR	S4	0.007	2.1	0.10	2.8

Figure 11. Libraries of simulated parameters and experimental gates

(A) The optimization results for the 2-bit counter. For each pair of (n,K), the table hierarchically displays the values of the mean and the standard deviation of the pulse widths δs, and the y_{max} value of the Hill coefficient for a given external pulse train with amplitude $A = 20$. Due to the simulation time constraint, we only simulated discrete values of K and n with increment by 0.01, and 0.1 respectively. The " ..." line represents the part of the database which is not displayed here. The complete table can be found in the github link https://github.com/danielchen26/circuit-design.

(B) A library of parameters table for a list of synthetic transcriptional gates in *E. coli* (Nielsen et al., 2016).

One of the basic information-processing circuits is the sequential counter, which keeps track of the number of events of a certain type that have been detected. In digital computers, precision is limited and counts are always made modulo an integer; we take the same view here. We see the *N*-bit counter as a key component of future genetic computing devices, and analogous ideas should be useful in building more complicated feedback components. Potential applications of such a circuit would include programmable cell death after detecting a certain number of events (Khalil and Collins, 2010). The resetting capability of the counter enables it to function periodically, such as directing the expression of specific enzymes in an ordered and periodic manner. It can also produce an output to match a natural cycle, for example, based on sensed hormones and other circadian signals.

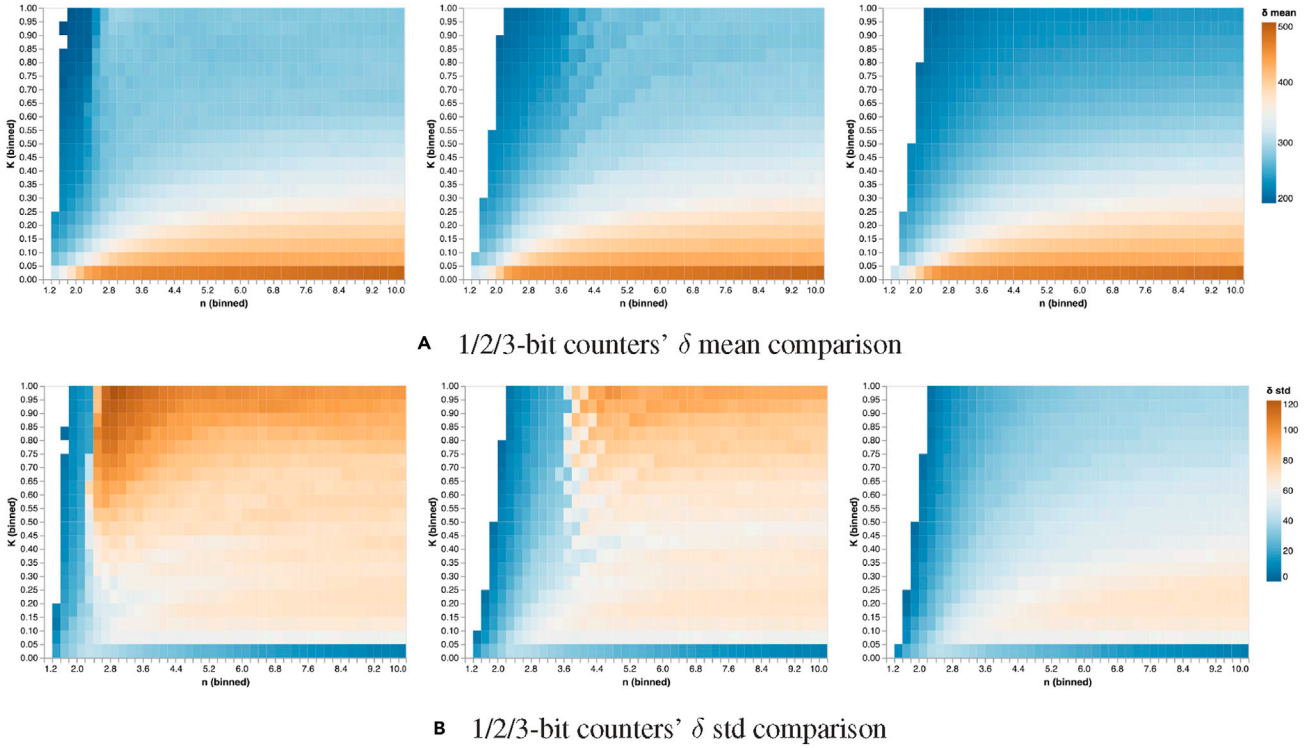


Figure 12. The mean and the standard deviation of the pulse width (δ) for the 1-bit counter, the 2-bit counter, and the 3-bit counter for various (n, K) pairs

We explore the (n, K) space with the cooperativity index n ranging from 1 to 10, and the kinetic constant K ranging from 0 to 1. The pulse width's mean (δ-mean) for each (n, K) pair indicates the mean of the feasible range of pulse widths for a specific (n, K) pair. The pulse width's standard deviation (δ-std) quantifies the robustness of a given counter against perturbations. Right, middle, and left plots show, respectively, the mean (top) and standard deviation (bottom) for 1-bit, 2-bit and 3-bit counters ordered from left to right. The plots show that the 1-bit counter can operate with smaller values of n. The region of robustness for the 2-bit and the 3-bit counters shifts rightward in the (n, K) plan compared to the 1-bit counter.

Additionally, parity checkers or, more generally, counting modulo an integer, are used in error-correcting codes, and can be expected to play an important role in synthetic digital inter-cellular communication devices.

Our work can be viewed in the broader context of modeling and mathematical and control-theoretic analysis in synthetic biology seen as an engineering discipline (Del Vecchio et al., 2018). These analysis tools help guide bottom-up approaches to synthesis based on combining individual genetic components (Bloom et al., 2014), modules (Hasty et al., 2002), and programmed feedback (Bloom et al., 2014, 2015; Stapleton et al., 2012). Novel engineering approaches and designs have greatly enhanced the ability to harness gene regulatory circuits for various applications (Bacchus et al., 2013; Endo et al., 2013; Weber and Fussenegger, 2009; Tigges et al., 2009), implementing Boolean logic functions (Miyamoto et al., 2013; Purcell and Lu, 2014; Rinaudo et al., 2007; Daniel et al., 2013; Farzadfard and Lu, 2014; Roquet and Lu, 2014; Bonnet et al., 2013), and targeting specific disease states (Aubel and Fussenegger, 2010).

In this work, we proposed a generic framework for building a distributed synthetic multi-bit counter. We are ultimately motivated by the long-term goal of designing synthetic genetic circuits that are "universal" computing devices; such circuits should be capable of finite-automaton computation, in analogy to central processing units in digital computers or, more abstractly, the "head" of a Turing machine. We started by designing the building block module, the single-bit counter (a parity checker). We then scaled up the single-bit counter design to a multi-bit counter, distributing the computation using connector modules communicating between the individual parity checkers via diffusible small molecules. We have showcased a feasible single-bit counter design by implementing a Cello-optimized transcriptional circuit in the *E. coli* system *in silico*. Next, we have systematically explored the parameter space of genetic circuits with a shared shifted Hill activation function across all gates. The results can then be used to populate a database for designing more general counters. Our generic computational framework is universal, and the connector

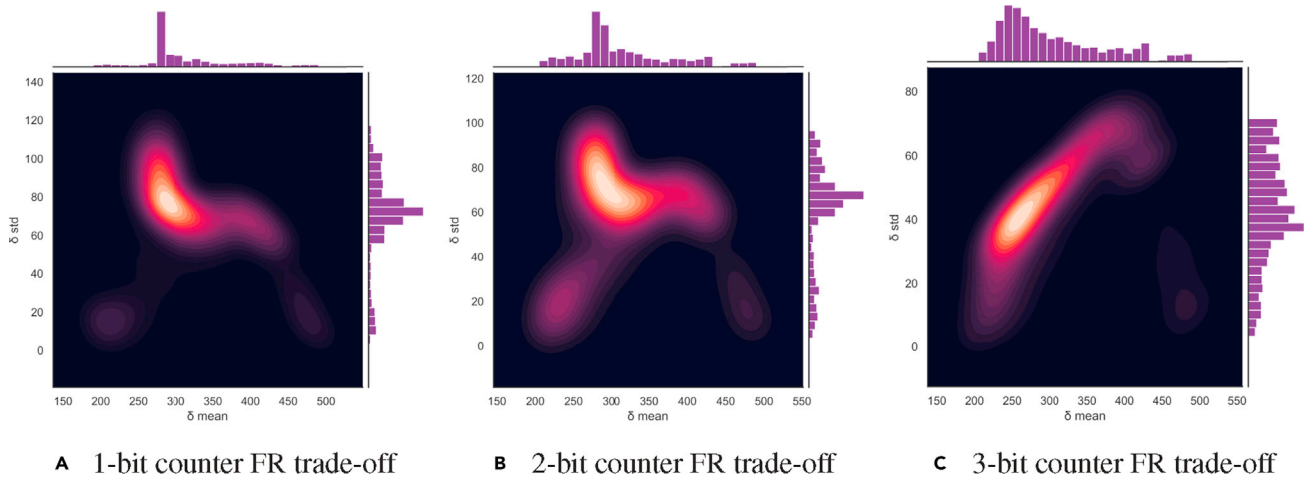


Figure 13. Feasibility vs. robustness (FR) trade-off comparison between the 1-bit counter, the 2-bit counter, and the 3-bit counter

For all simulated synthetic gates, we plot a 2D density graph with contours for δ mean vs. δ-std. The plots help identify the preferred region for designing new synthetic gates. The contour lines represent different levels of probability density for data points. The darker, the higher probability it represents. Right, middle, and left plots show, respectively, the contours for 1-bit, 2-bit and 3-bit counters. For the single-bit counter, the best region for designing a synthetic gate is around δ-mean = 300 and δ-std = 75. For the 2-bit counter, the best region for designing a synthetic gate is δ-mean from around 260 to 350, and δ-std can range from 60 to 90. For the 3-bit counter, the best region for designing a synthetic gate is around 240 to 320, and δ-std can range from 20 to 70.

modules are flexible enough to be used with different cell-cell communication signaling systems (Fredriksson et al., 2003). For example, the communication signals can be realized not only by quorum-sensing molecules but also, potentially, by small peptides, and the EGFR system (Wieduwilt and Moasser, 2008). Our multi-bit counter framework has quantifiable robustness against variation in the width of the input pulses, and it can naturally handle asynchronous counting situations. Our computational design framework can shed light on how experimentalists can build more robust distributed multi-bit counters by tuning the characteristic curves (described by Hill functions) of the gates to better match the range of expected pulse widths. This computational framework can significantly reduce the number of trial-and-error experiments by using the design selection graphs provided in Figure 14 as a showcase example.

Our work, while theoretical, lays out a clear path to the development of scalable counter. The experimental realization is feasible, as shown by the obtained designs and specific DNA sequences. On the other hand, the size of the required plasmids makes the actual experiments to be just beyond the edge of current technology. New tools (at Voigt's lab and others) are being developed which will enable an implementation in the future.

Over the last decade, synthetic biology has played an increasing role in the understanding and controlling of complex cellular systems. However, numerous challenges remain. In order to connect different gates, one needs to deal with the problem of matching different dynamic ranges of multiple circuit outputs (Brophy and Voigt, 2014), and synthetic circuits place a burden on host cells. Traditional genetic design was very labor-intensive, but this process has been greatly streamlined with the advent of genetic circuit design automation pipelines, including Cello (Yaman et al., 2012; Bhatia et al., 2017; Nielsen et al., 2016). It is now possible to obtain a promoter wiring diagram of a synthetic plasmid by supplying a Boolean function specified by an input-output truth table to Cello. Cello then attempts to minimize host burden while at the same time automatizing the search for optimal designs by employing simulated annealing and other optimization algorithms so as to minimize a "circuit score" with respect with a biological constraint library.

However, combinational logic is not sufficient. The intricate feedback wiring networks in living systems are evidence that cellular systems can dynamically perform complex jobs via memory-like units containing sequential logic circuits, and show how desired cell states emerge with cellular sensing. For example, living cells that differentiate into multi-cellular structures are usually controlled by sequential logic instead of purely combinational logic. One of the most compact examples for understanding cellular sequential logic is the "switch"-like memory system. By the turn of the millennium, Gardner et al. (2000) constructed a

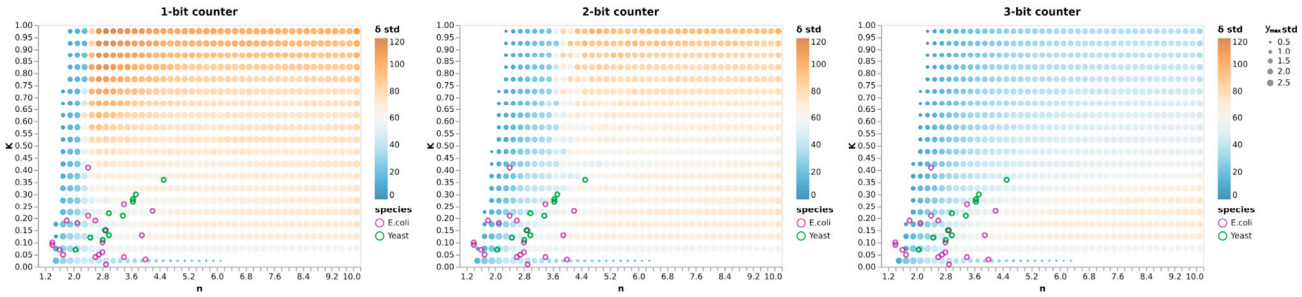


Figure 14. Experimental transcriptional gates in the simulated (n, K) space

We plot experimentally built synthetic gates for the *E.coli* and yeast system on top of the δ-std plane for both 1-bit counter, 2-bit counter, and 3-bit counter as an example. The purple circles represent the experimental values of the synthetic gates for the *E. coli* system, and the green circles are for the yeast system. The δ-std is colored with the solid dot, and the size of the dot represents the standard deviation of the y_{max} value. We have sampled n from 1 to 10 with a 0.2 step size, and K from 0 to 1 with a 0.05 step size in the simulations.

genetic toggle switch containing promoters that drive mutually inhibitory transcriptional repressors' expression. This work has laid the foundation for building memory-based circuits, and it serves as a crucial framework for building the SR-latch used to realize the single-bit counter in our generic design. Later on, Friedland et al. (2009) studied two versions of genetic counter in *E. coli* that can count up to three induction events only. In comparison, our 2-bit counter design has the advantage of counting an arbitrary number of pulses modulo 4.

Several synthetic circuits with memory have been proposed in the literature (Inniss and Silver, 2013). For example, the work by Hillenbrand et al. (2013) has studied a JK-latch. Instead of using the universal logic gate "NOR" everywhere, the authors proposed the JK-latch with a design combining "AND" gates and "NOR" gates. Such a design is not readily compatible with CRISPRi-based automation toolboxes (such as Cello), making it harder to implement and scale-up. In 2014, a genetic sequential logic circuit was explored with a clock pulse generator by Chuang and Lin (2014), thus relying upon periodicity of inputs. In contrast, our design naturally handles asynchronous counting scenarios. A counter design for a pulse-detecting circuit that only responds to the falling edge of the input pulse was proposed by Noman et al. (2016), detecting the completion of an event; in contrast, our design detects pulses with durations within a specific interval.

With the growing importance of engineering synthetic genetic circuits, we are currently facing scalability bottlenecks caused mainly by the inability of host cells to endure "large" foreign circuits. Due to the limitations of traditional recombinase-based state machines (Chiu and Jiang, 2017; Yehl and Lu, 2017), the CRISPRi system is being explored more actively to encode cellular-based memory (Yehl and Lu, 2017; Andrews et al., 2018). Our counter framework can be readily used with CRISPRi for implementing more complex circuits in host cells and can potentially realize additional functions per cell (Jusiak et al., 2016). Although some designs for multicellular recombinase logic have been proposed lately (Guiziou et al., 2018), a generic design framework with integrated feedback loops that performs asynchronous distributed counting has not been proposed so far.

CRISPRi holds the potential of being the right candidate for realizing more complex functions and offering more gates in a single cell. However, it critically relies on high expression levels of Cas9 which is toxic to the host cells (Zhang and Voigt, 2018). Hence, adopting new principles and new synthetic architectures like distributed computation have also been widely discussed (Xiang et al., 2018; Karkaria et al., 2020; Al- Radhawi et al., 2020). Similarly, our approach for designing a multi-bit counter system tries to solve scalability issues via distributed computation using cell-cell communication. Until now, a generic scalable paradigm for implementing distributed computation with regulatory feedback for multi-cellular counting tasks had not been proposed. As a simple example for realizing distributed computation with memory in the sequential logic realm, our synthetic genetic counter can be widely adopted in various biological counting scenarios. The computational framework can be easily translated to be used in other biological contexts. The advantage of such a counting framework is not restricted to being implemented as an event detection machine. It can be viewed as a thresholding machine for selecting sequential binding events in biology, such as in kinetic-proof reading systems (Hopfield, 1974). Viewing the counter as a probing system can

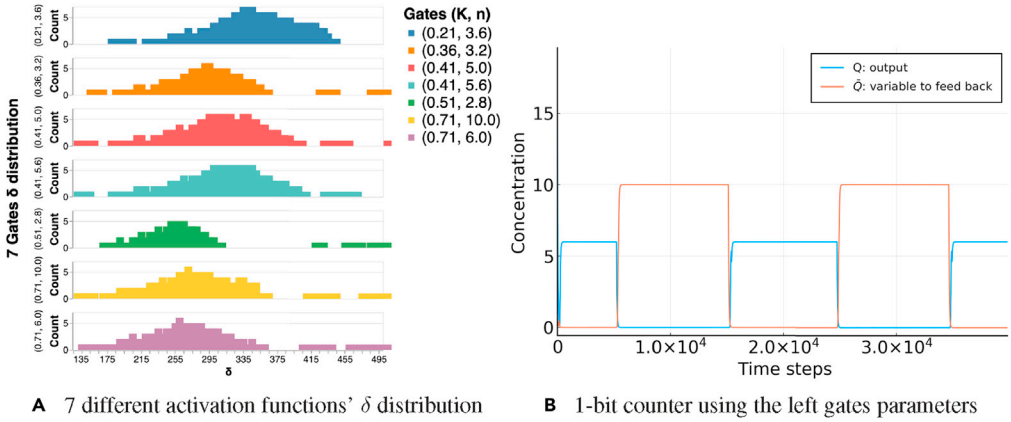


Figure 15. An example of a 1-bit counter with unique activation functions for each gate

(A) Given an external pulse width δ, seven gates are selected conditioned on having the target pulse width within their distribution. Here we show the δ distributions associated with each gate that contains different (n, K).

(B) With the selected seven gates' parameters from (A), we show an example trajectory of a 1-bit counter.

help us understand cellular mechanisms more deeply. It has been suggested that sequential logic can overcome the information bottlenecks inherent in complex networks (Letsou and Cai, 2016). The combinatorial binding of transcription factors at promoters likely contributes to cell-type-specific gene expression in a static view. However, sequential logic may reveal a dynamic picture of the underlying reaction landscape.

As for future directions, we are intrigued by the idea of applying similar methods in mammalian cells, and specifically for immunotherapy. For example, one can think of engineering CAR-T cell systems for distributed immune system computation (using cytokines as signals such as in the work by Cho et al. (2018)), with the integrated ability to sense an external cancerous environment with tunable thresholding. Another example stems from developmental biology, where the distributed multi-bit counter can be considered as a guiding system for triggering various biochemical events at different developmental stages. Each bit level can be implemented to produce a specific set of transcription factors or essential proteins to trigger the desired cellular signaling events.

Furthermore, in another direction, epigenetic modifications play crucial roles in cell fate networks and the different developmental stages of a cell. A long-term goal in epigenetics is to realize systems that sense and count certain cellular events (Prochazka et al., 2017; Sheth and Wang, 2018; Bashor and Collins, 2018). A synthetic epigenetic system can be implemented with the ability to over-express TET to affect DNA methylation and other transcriptional factors to further affect RNA modifications, DNA methylation, and histone modifications. Promisingly, a very recent discovery (Hino et al., 2020) of an ERK-mediated mechano-chemical feedback system that can generate complicated multi-cellular patterns has shed light on how cellular waves can contribute to the long-range correlation between different distant modules. Such long-range communication mechanisms will complement the diffusive nature of cell-cell communication signaling. They can be built on top of our counter system to achieve additional complex cellular functions and be used as an additional approach for realizing parallel cellular computation.

Limitations of the study

From an analytical point of view, our analysis is based on the assumption that cells grow in a homogeneous medium; therefore, time delays due to diffusion and other spatial effects are assumed to be negligible. Growth of colonies on plates will have to be modeled with partial differential equations and the behavior of the circuit may depend on the geometry of the setting.

From an implementation viewpoint, scaling up the counter will require a larger number of orthogonal diffusible molecules than available at present. Current technology provides six such diffusible molecules (four types of quorum sensing signals based on AHL plus two additional ones as proposed by Du et al. (2020), thus limiting the realizable counters to four bits.

	K	n	δ	A	Y_max
0	0.01	1.4	295.0	20.0	4.0
1	0.01	1.4	305.0	20.0	5.0
2	0.01	1.4	310.0	20.0	6.0
3	0.01	1.4	315.0	20.0	7.0
4	0.01	1.4	320.0	20.0	8.0
...	...	...	...	...	...
94651	0.96	10.0	300.0	20.0	10.0
94652	0.96	10.0	305.0	20.0	9.0
94653	0.96	10.0	305.0	20.0	10.0
94654	0.96	10.0	310.0	20.0	10.0
94655	0.96	10.0	315.0	20.0	10.0

Figure 16. A snapshot of the database of the 3-bit counter

Each row in the data table corresponds to a feasible parameter set that gives rise to a working 3-bit counter. In this table, we only show what the resulting database looks like, and the sampled parameter K ranges from 0.011 to 0.091, the parameter n ranges from 1.5 to 3.0, the parameter δ ranges from 275 to 350, the parameter y_{max} ranges from 1 to 3, and we fixed the parameter A to 20. The full table can be found in https://github.com/danielchen26/circuit-design

STAR★METHODS

Detailed methods are provided in the online version of this paper and include the following:

- KEY RESOURCES TABLE
- RESOURCE AVAILABILITY
  - Lead contact
  - Data and code availability
- METHODS DETAILS
  - Biological gate design & assignment
  - Multi-bit counter
  - Parameter optimization

SUPPLEMENTAL INFORMATION

Supplemental information can be found online at https://doi.org/10.1016/j.isci.2021.103526.

ACKNOWLEDGMENT

We thank Shuyi Zhang for initial discussions.

This research was supported in part by Grants SRC SB-2837-B and NSF 1849588, for the project "Very large-scale genetic circuit design automation".

AUTHOR CONTRIBUTIONS

Conceptualization: T.C, M.A, C.V, and E.S; Methodology and Investigation: T.C, M.A, and E.S; Formal Analysis: T.C, M.A; Visualization: T.C; Data Curation: T.C; Writing – Original Draft, T.C; Writing – Review & Editing: T.C, M.A, C.V, and E.S. Project Administration: E.S; Funding Acquisition: E.S.

DECLARATIONS OF INTERESTS

The authors declare no competing or financial interests.

Received: May 13, 2021
Revised: September 30, 2021
Accepted: November 23, 2021
Published: December 17, 2021

REFERENCES

Al-Radhawi, M.A., Tran, A., Ernst, E., Chen, T., Voigt, C., and Sontag, E. (2020). Distributed implementation of Boolean functions by transcriptional synthetic circuits. ACS Synth. Biol. *9*, 2172–2187.

Andrews, L.B., Nielsen, A.A.K., and Voigt, C.A. (2018). Cellular checkpoint control using programmable sequential logic. Science *361*. https://science.sciencemag.org/content/361/6408/eaap8987.

Andrianantoandro, E., Basu, S., Karig, D.K., and Weiss, R. (2006). Synthetic biology: new engineering rules for an emerging discipline. Mol. Syst. Biol. *2*. https://doi.org/10.1038/msb4100073.

Archer, E.J., Robinson, A.B., and Süel, G.M. (2012). Engineered E. coli that detect and respond to gut inflammation through nitric oxide sensing. ACS Synth. Biol. *1*, 451–457.

Aubel, D., and Fussenegger, M. (2010). Mammalian synthetic biology–from tools to therapies. Bioessays *32*, 332–345.

Bacchus, W., Aubel, D., and Fussenegger, M. (2013). Biomedically relevant circuit-design strategies in mammalian synthetic biology. Mol. Syst. Biol. *9*, 691.

Bashor, C.J., and Collins, J.J. (2018). Understanding biological regulation through synthetic biology. Annu. Rev. Biophys. *47*, 399–423.

Bhatia, S.P., Smanski, M.J., Voigt, C.A., and Densmore, D.M. (2017). Genetic design via combinatorial constraint specification. ACS Synth. Biol. *6*, 2130–2135.

Bloom, R.J., Winkler, S.M., and Smolke, C.D. (2014). A quantitative framework for the forward design of synthetic miRNA circuits. Nat. Methods *11*, 1147.

Bloom, R.J., Winkler, S.M., and Smolke, C.D. (2015). Synthetic feedback control using an RNAi-based gene-regulatory device. J. Biol. Eng. *9*, 5.

Bonnet, J., Yin, P., Ortiz, M.E., Subsoontorn, P., and Endy, D. (2013). Amplifying genetic logic gates. Science *340*, 599–603.

Borkowski, O., Ceroni, F., Stan, G.-B., and Ellis, T. (2016). Overloaded and stressed: whole-cell considerations for bacterial synthetic biology. Curr. Opin. Microbiol. *33*, 123–130.

Bostock, G. (1988). Programmable Logic Devices: Technology and Applications (McGraw-Hill Companies).

Brophy, J.A., and Voigt, C.A. (2014). Principles of genetic circuit design. Nat. Methods *11*, 508.

Callura, J.M., Cantor, C.R., and Collins, J.J. (2012). Genetic switchboard for synthetic biology applications. Proc. Natl. Acad. Sci. U S A *109*, 5850–5855.

Cameron, D.E., Bashor, C.J., and Collins, J.J. (2014). A brief history of synthetic biology. Nat. Rev. Microbiol. *12*, 381–390.

Cheng, A.A., and Lu, T.K. (2012). Synthetic biology: an emerging engineering discipline. Annu. Rev. Biomed. Eng. *14*, 155–178.

Chiu, T.-Y., and Jiang, J.-H.R. (2017). Logic synthesis of recombinase-based genetic circuits. Sci. Rep. *7*, 1–13.

Cho, J.H., Collins, J.J., and Wong, W.W. (2018). Universal chimeric antigen receptors for multiplexed and logical control of T cell responses. Cell *173*, 1426–1438.

Chuang, C.-H., and Lin, C.-L. (2014). Synthesizing genetic sequential logic circuit with clock pulse generator. BMC Syst. Biol. *8*, 63.

Daniel, R., Rubens, J.R., Sarpeshkar, R., and Lu, T.K. (2013). Synthetic analog computation in living cells. Nature *497*, 619–623.

Del Vecchio, D., Ninfa, A.J., and Sontag, E.D. (2008). Modular cell biology: retroactivity and insulation. Mol. Syst. Biol. *4*, 161.

Del Vecchio, D., Qian, Y., Murray, R.M., and Sontag, E.D. (2018). Future systems and control research in synthetic biology. Annu. Rev. Control *45*, 5–17.

Du, P., Zhao, H., Zhang, H., Wang, R., Huang, J., Tian, Y., Luo, X., Luo, X., Wang, M., Xiang, Y., et al. (2020). De novo design of an intercellular signaling toolbox for multi-channel cell–cell communication and biological computation. Nat. Commun. *11*, 1–11.

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

Endo, K., Hayashi, K., Inoue, T., and Saito, H. (2013). A versatile cis-acting inverter module for synthetic translational switches. Nat. Commun. *4*, 1–9.

Farzadfard, F., and Lu, T.K. (2014). Genomically encoded analog memory with precise in vivo DNA writing in living cell populations. Science *346*, 1256272.

Fredriksson, R., Lagerström, M.C., Lundin, L.-G., and Schiöth, H.B. (2003). The G-protein-coupled receptors in the human genome form five main families. phylogenetic analysis, paralogon groups, and fingerprints. Mol. Pharmacol. *63*, 1256–1272.

Friedland, A.E., Lu, T.K., Wang, X., Shi, D., Church, G., and Collins, J.J. (2009). Synthetic gene networks that count. Science *324*, 1199–1202.

Gander, M.W., Vrana, J.D., Voje, W.E., Carothers, J.M., and Klavins, E. (2017). Digital logic circuits in yeast with CRISPR-dCas9 NOR gates. Nat. Commun. *8*, 1–11.

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

Guiziou, S., Ulliana, F., Moreau, V., Leclere, M., and Bonnet, J. (2018). An automated design framework for multicellular recombinase logic. ACS Synth. Biol. *7*, 1406–1412.

Gupta, S., Bram, E.E., and Weiss, R. (2013). Genetically programmable pathogen sense and destroy. ACS Synth. Biol. *2*, 715–723.

Hardy, Y., and Steeb, W.-H. (2001). Logic gates. In Classical and Quantum Computing (Springer), pp. 79–90.

Hasty, J., Dolnik, M., Rottschäfer, V., and Collins, J.J. (2002). Synthetic gene network for entraining and amplifying cellular oscillations. Phys. Rev. Lett. *88*, 148101.

Hillenbrand, P., Fritz, G., and Gerland, U. (2013). Biological signal processing with a genetic toggle switch. PLoS One *8*, e68345.

Hino, N., Rossetti, L., Marín-Llauradó, A., Aoki, K., Trepat, X., Matsuda, M., and Hirashima, T. (2020). ERK-mediated mechanochemical waves direct collective cell polarization. Dev. Cell *53*, 646–660.e8.

Hopfield, J.J. (1974). Kinetic proofreading: a new mechanism for reducing errors in biosynthetic processes requiring high specificity. Proc. Natl. Acad. Sci. U S A *71*, 4135–4139.

Hwang, I.Y., Tan, M.H., Koh, E., Ho, C.L., Poh, C.L., and Chang, M.W. (2014). Reprogramming microbes to be pathogen-seeking killers. ACS Synth. Biol. *3*, 228–237.

Inniss, M.C., and Silver, P.A. (2013). Building synthetic memory. Curr. Biol. *23*, R812–R816.

Jaeger, R.C., and Blalock, T.N. (1997). Microelectronic Circuit Design (McGraw-Hill).

Jusiak, B., Cleto, S., Perez-Piñera, P., and Lu, T.K. (2016). Engineering synthetic gene circuits in living cells with CRISPR technology. Trends Biotechnol. *34*, 535–547.

Karkaria, B.D., Treloar, N.J., Barnes, C.P., and Fedorec, A.J. (2020). From microbial communities to distributed computing systems. Front. Bioeng. Biotechnol. *8*, 834.

Khalil, A.S., and Collins, J.J. (2010). Synthetic biology: applications come of age. Nat. Rev. Genet. *11*, 367–379.

Kim, H., Bojar, D., and Fussenegger, M. (2019). A crispr/cas9-based central processing unit to program complex logic computation in human cells. Proc. Natl. Acad. Sci. U S A *116*, 7214–7219. https://www.pnas.org/content/116/15/7214.

Kohanski, M.A., and Collins, J.J. (2008). Rewiring bacteria, two components at a time. Cell *133*, 947–948.

Letsou, W., and Cai, L. (2016). Noncommutative biology: sequential regulation of complex networks. PLoS Comput. Biol. *12*, e1005089.

Miyamoto, T., Razavi, S., DeRose, R., and Inoue, T. (2013). Synthesizing biomolecule-based Boolean logic gates. ACS Synth. Biol. *2*, 72–82.

Moon, T.S., Lou, C., Tamsir, A., Stanton, B.C., and Voigt, C.A. (2012). Genetic programs constructed from layered logic gates in single cells. Nature *491*, 249–253.

Nielsen, A.A.K., Der, B.S., Shin, J., Vaidyanathan, P., Paralanov, V., Strychalski, E.A., Ross, D., Densmore, D., and Voigt, C.A. (2016). Genetic circuit design automation. Science *352*. https://science.sciencemag.org/content/352/6281/aac7341.

Noman, N., Inniss, M., Iba, H., and Way, J.C. (2016). Pulse detecting genetic circuit–a new design approach. PLoS One *11*, e0167162.

Park, M., Patel, N., Keung, A.J., and Khalil, A.S. (2019). Engineering epigenetic regulation using synthetic read-write modules. Cell *176*, 227–238.

Prochazka, L., Benenson, Y., and Zandstra, P.W. (2017). Synthetic gene circuits and cellular decision-making in human pluripotent stem cells. Curr. Opin. Syst. Biol. *5*, 93–103.

Purcell, O., and Lu, T.K. (2014). Synthetic analog and digital circuits for cellular computation and memory. Curr. Opin. Biotechnol. *29*, 146–155.

Rackauckas, C., and Nie, Q. (2017). Differential equations.jl: a performant and feature-rich ecosystem for solving differential equations in Julia. J. Open Res. Softw. *5*, 15.

Rinaudo, K., Bleris, L., Maddamsetti, R., Subramanian, S., Weiss, R., and Benenson, Y. (2007). A universal RNAi-based logic evaluator that operates in mammalian cells. Nat. Biotechnol. *25*, 795–801.

Roquet, N., and Lu, T.K. (2014). Digital and analog gene circuits for biotechnology. Biotechnol. J. *9*, 597–608.

Saeidi, N., Wong, C.K., Lo, T.-M., Nguyen, H.X., Ling, H., Leong, S.S.J., Poh, C.L., and Chang, M.W. (2011). Engineering microbes to sense and eradicate Pseudomonas aeruginosa, a human pathogen. Mol. Syst. Biol. *7*, 521.

Sedlmayer, F., Aubel, D., and Fussenegger, M. (2018). Synthetic gene circuits for the detection, elimination and prevention of disease. Nat. Biomed. Eng. *2*, 399–415.

Sheth, R.U., and Wang, H.H. (2018). DNA-based memory devices for recording cellular events. Nat. Rev. Genet. *19*, 718–732.

Siuti, P., Yazbek, J., and Lu, T.K. (2013). Synthetic circuits integrating logic and memory in living cells. Nat. Biotechnol. *31*, 448.

Sprinzak, D., and Elowitz, M.B. (2005). Reconstruction of genetic circuits. Nature *438*, 443–448.

Stapleton, J.A., Endo, K., Fujita, Y., Hayashi, K., Takinoue, M., Saito, H., and Inoue, T. (2012). Feedback control of protein expression in mammalian cells by tunable synthetic translational inhibition. ACS Synth. Biol. *1*, 83–88.

Subsoontorn, P., and Endy, D. (2012). Design and analysis of genetically encoded counters. Proced. Comput. Sci. *11*, 43–54, Proceedings of the 3rd International Conference on Computational Systems-Biology and Bioinformatics. https://www.sciencedirect.com/science/article/pii/S1877050912005893.

Tigges, M., Marquez-Lago, T.T., Stelling, J., and Fussenegger, M. (2009). A tunable synthetic mammalian oscillator. Nature *457*, 309–312.

Vogel, A.M., Persson, K.M., Seamons, T.R., and Deans, T.L. (2019). Synthetic biology for improving cell fate decisions and tissue engineering outcomes. Emerging Top. Life Sci. *3*, 631–643.

Weber, W., and Fussenegger, M. (2009). Engineering of synthetic mammalian gene networks. Chem. Biol. *16*, 287–297.

Wieduwilt, M.J., and Moasser, M.M. (2008). The epidermal growth factor receptor family: biology driving targeted therapeutics. Cell Mol. Life Sci. *65*, 1566–1584.

Xiang, Y., Dalchau, N., and Wang, B. (2018). Scaling up genetic circuit design for cellular computing: advances and prospects. Nat. Comput. *17*, 833–853.

Yaman, F., Bhatia, S., Adler, A., Densmore, D., and Beal, J. (2012). Automated selection of synthetic biology parts for genetic regulatory networks. ACS Synth. Biol. *1*, 332–344.

Yang, L., Nielsen, A.A., Fernandez-Rodriguez, J., McClune, C.J., Laub, M.T., Lu, T.K., and Voigt, C.A. (2014). Permanent genetic memory with > 1-byte capacity. Nat. Methods *11*, 1261–1266.

Yehl, K., and Lu, T. (2017). Scaling computation and memory in living cells. Curr. Opin. Biomed. Eng. *4*, 143–151.

Zhang, S., and Voigt, C.A. (2018). Engineered dcas9 with reduced toxicity in bacteria: implications for genetic circuit design. Nucleic Acids Res. *46*, 11115–11125.

Zhao, J., Pokhilko, A., Ebenhöh, O., Rosser, S.J., and Colloms, S.D. (2019). A single-input binary counting module based on serine integrase site-specific recombination. Nucleic Acids Res. *47*, 4896–4909. https://doi.org/10.1093/nar/gkz245.

STAR★METHODS

KEY RESOURCES TABLE

REAGENT or RESOURCE	SOURCE	IDENTIFIER
Software and algorithms		
Julia Computing Software	Julia Core Development Team	https://doi.org/10.1137/141000671

RESOURCE AVAILABILITY

Lead contact

Further information should be directed to Eduardo Sontag (Email: e.sontag@northeastern.edu).

Data and code availability

The programming code that was used to analyze the raw data that supports the findings of this study is available in Github https://github.com/danielchen26/circuit-design.

METHODS DETAILS

This work has proposed a new distributed circuit construction framework for a counter. We first show how we design the building block unit, which is a single-bit counter. Next, we describe our multi-bit counter system model that utilizes the same single-bit counter construction repetitively for scaling up the system. This section will first show the single-bit counter promoter design obtained by using available automated genetic design software (Nielsen et al., 2016). We will explain how we come up with the gate assignment and perform the consequent mathematical modeling. All our simulations are performed using Julia (version 1.5.1) (Rackauckas and Nie, 2017).

Biological gate design & assignment

Biological gate design. To show a working example for the single-bit counter in *E. coli*, we employ the genetic automation pipeline Cello 2.0 to map the logic gates to biological gates. The feedback in our system and the fact that inputs will be pulses introduce a major difficulty to an off-the-shelf application of Cello 2.0. This is because the current version of Cello handles combinational circuits only. Nevertheless, a recent extension of Cello was employed to design sequential logic circuits (Andrews et al., 2018), and we use the latter version here.

In order to assign gates to our parity checkers, we consider a slight variation in which we temporarily ignore the feedback part. Thus, we first specify a state transition table (Table 1) for the open-loop circuit without feedback, and consider all possible inputs to the XNOR gate. The Cello 2.0 software takes this binary transition state table, together with the circuit topology specified by a Verilog file, and generates a biological gate assignment. Table 1 presents all possible transition states for the 2 inputs (X, Y) and 2 outputs ($\mathbf{Q}$, $\overline{\mathbf{Q}}$) of the open-loop version.

Cello 2.0 outputs a gate assignment that is optimized with respect to the cross-talk between different biological promoters, and aims at producing the best dynamic range of the overall circuit. However, Cello 2.0's optimization algorithm does not maximize the range of δ (pulse duration), which is the relevant objective function in our design. Hence, we have performed an additional optimization step by replacing one gate at a time with another gate from the *E.coli* gate library. We tested all such one-gate-replacement candidate circuits and compared them to the Cello-produced circuit. The optimized circuit is depicted in Figure S10B which shows the biological gate diagram of the plasmid based on the following repressors: QacR, LmrA, SrpR, BM3R1, PhlF, PsrA, BetI. This diagram follows Cello conventions and has produced the best δ range among all candidates. The list of all gates is given in Table 2. In the supplemental information, we show other feasible circuits.

The diagram in Figure S10B is based on the topology that was specified by our open-loop circuit Verilog file. However, our desired design contains a feedback path from the BetI gate to the QacR gate. How does one biologically "connect" this output signal back to the circuit's input to obtain the closed-loop version of

the circuit? The "gene-promoter" paradigm is perfectly suited to achieve such a goal: The answer is to copy the promoter in front of the PsrA gene (which is the target of the BetI gene as part of the SR-latch) to the promoter region of the QacR gene (which is a tandem promoter also containing the target of the input signal, pBAD). Thus, the QacR gate will receive a signal from the BetI gate without changing the characteristic response curve for the gate. Changes in the promoter region of a gate do not change the response curve of that particular gate because the parameters that define the response curve only relate to what is inside the "gate box", namely, the gene coding region and the promoter region of a target gene that the protein expressed by the previous gene can bind to repress.

Mathematical model for the single-bit counter. As the repressors have been assigned to the logical gates, we introduce a differential equation model and use it to analyze the effects of closing the feedback loop and applying the input pulses.

Using the gate assignments from Cello 2.0, we are now able to write down our differential equation model for the single-bit counter. In this case, pBAD is considered as the external input, and the BetI gate feeds back to the QacR gate. The circuit output will be the yellow fluorescent protein (YFP) produced by the gate PsrA.

Our model is given by the following system of seven differential equations:

$$\frac{dm_{\text{QacR}}}{dt} = \xi\cdot\left(\text{QacR}_{\min} + \frac{\text{QacR}_K^{\text{QacR}_n}\cdot(\text{QacR}_{\max}-\text{QacR}_{\min})}{\text{QacR}_K^{\text{QacR}_n} + \left(x_{\text{BetI}}+p\right)^{\text{QacR}_n}}\right) - \gamma\cdot m_{\text{QacR}} \quad \text{(Equation 2)}$$

$$\frac{dm_{\text{LmrA}}}{dt} = \xi\cdot\left(\text{LmrA}_{\min} + \frac{\text{LmrA}_K^{\text{LmrA}_n}\cdot(\text{LmrA}_{\max}-\text{LmrA}_{\min})}{\text{LmrA}_K^{\text{LmrA}_n} + \left(x_{\text{QacR}}+p\right)^{\text{LmrA}_n}}\right) - \gamma\cdot m_{\text{LmrA}} \quad \text{(Equation 3)}$$

$$\frac{dm_{\text{SrpR}}}{dt} = \xi\cdot\left(\text{SrpR}_{\min} + \frac{\text{SrpR}_K^{\text{SrpR}_n}\cdot(\text{SrpR}_{\max}-\text{SrpR}_{\min})}{\text{SrpR}_K^{\text{SrpR}_n} + \left(x_{\text{QacR}}+x_{\text{BetI}}\right)^{\text{SrpR}_n}}\right) - \gamma\cdot m_{\text{SrpR}} \quad \text{(Equation 4)}$$

$$\frac{dm_{\text{BM3R1}}}{dt} = \xi\cdot\left(\text{BM3R1}_{\min} + \frac{\text{BM3R1}_K^{\text{BM3R1}_n}\cdot(\text{BM3R1}_{\max}-\text{BM3R1}_{\min})}{\text{BM3R1}_K^{\text{BM3R1}_n} + \left(x_{\text{LmrA}}+x_{\text{SrpR}}\right)^{\text{BM3R1}_n}}\right) - \gamma\cdot m_{\text{BM3R1}} \quad \text{(Equation 5)}$$

$$\frac{dm_{\text{PhlF}}}{dt} = \xi\cdot\left(\text{PhlF}_{\min} + \frac{\text{PhlF}_K^{\text{PhlF}_n}\cdot(\text{PhlF}_{\max}-\text{PhlF}_{\min})}{\text{PhlF}_K^{\text{PhlF}_n} + x_{\text{BM3R1}}^{\text{PhlF}_n}}\right) - \gamma\cdot m_{\text{PhlF}} \quad \text{(Equation 6)}$$

$$\frac{dm_{\text{PsrA}}}{dt} = \xi\cdot\left(\text{PsrA}_{\min} + \frac{\text{PsrA}_K^{\text{PsrA}_n}\cdot(\text{PsrA}_{\max}-\text{PsrA}_{\min})}{\text{PsrA}_K^{\text{PsrA}_n} + \left(x_{\text{PhlF}}+x_{\text{BetI}}\right)^{\text{PsrA}_n}}\right) - \gamma\cdot m_{\text{PsrA}} \quad \text{(Equation 7)}$$

$$\frac{dm_{\text{BetI}}}{dt} = \xi\cdot\left(\text{BetI}_{\min} + \frac{\text{BetI}_K^{\text{BetI}_n}\cdot(\text{BetI}_{\max}-\text{BetI}_{\min})}{\text{BetI}_K^{\text{BetI}_n} + \left(x_{\text{BM3R1}}+x_{\text{PsrA}}\right)^{\text{BetI}_n}}\right) - \gamma\cdot m_{\text{BetI}} \quad \text{(Equation 8)}$$

This model, and its parameters, are derived from the Cello 2.0 specifications as well as the paper by Andrews et al. (2018). The differential equation model has seven state variables $m_{gate\ name}$, which represent the mRNA concentrations associated with the respective gates. (Protein concentrations are assumed to be proportional to these.) Within the parentheses, we model each gate with a Hill equation in terms of each gate's relative promoter strength, and then convert units to mRNA concentrations by the scaling factor ξ. For each differential equation, we also include a degradation term for mRNA. For more details on this procedure, and the parameters used, see the work of Nielsen et al. (2016).

The assumption made in such models is that mRNA production rate is proportional to the relative promoter units (RPUs) for each promoter's activity. The $m_{gate\ name}$ is the concentration of mRNA produced by the output promoter. The constant ξ is a scaling factor that is used to convert relative promoter strength to mRNA concentrations for gate variables, and γ is the degradation rate of the mRNA of each gene. The promoter activities (7 gates) are reported in relative promoter units (RPUs). The input for each gate $x_{\text{gate name}} = \xi\gamma^{-1}m_{\text{gate name}}$, noting that $\xi\gamma^{-1}=1$ for our choice of parameters. Both ξ and γ are taken as constants here. They were arbitrarily set to $\gamma = 0.025\text{min}^{-1}$ and $\xi = 0.025$ [mRNA]/min − RPU in order to produce time trajectories consistent with bio-molecular timescales (Andrews et al., 2018).

Multi-bit counter

In the single-bit counter analysis, we have used the gates' assignment from the *E. coli* system. Each biological gate is unique and has different activation Hill functions. Next, we show how we model the multi-bit counter in which all gates share a single activation function. This means that the parameter sets for each gate are the same across seven differential equations as shown above. The shared version differential equations are indicated in Equation 9.

$$\frac{dm_{gate_i}}{dt} = \xi \cdot \left(dn_{shared} + \frac{\left(y_{\max,shared} - y_{\min,shared}\right) \cdot K_{shared}^{n_{shared}}}{K_{shared}^{n_{shared}} + x_{gate}^{n_{shared}}} \right) - \gamma \cdot m_{gate_i} \quad \text{(Equation 9)}$$

Since the multi-bit counter system is modular, for a counting system that counts to 2^N, we will need $N-1$ connectors, with a connector type 1 placed between the first two bits, and the rest of the connectors are all type 2. The dynamics of these two types of connectors are shown below.

Mathematical model for the connector type 1. The connector type 1 (AND gate) module in Figure 8 can be modeled as follows. There are three gates in this module. Each gate's dynamics is modeled by a shifted Hill activation function and a degradation function in general. The gate A and gate B in Figure 8 are two NOT gates with a single input, which is modeled by Equations 10 and 11. The gate C is a NOR gate that takes the output from gate A and gate B as inputs, and the dynamical equation for gate C is given by Equation 12. The overall system can be written as follows:

$$\frac{dm_A}{dt} = \xi \cdot \left(dn + \frac{(y_{max} - y_{min}) \cdot K^n}{K^n + (p)^n} \right) - \gamma \cdot m_A \quad \text{(Equation 10)}$$

$$\frac{dm_B}{dt} = \xi \cdot \left(dn + \frac{(y_{max} - y_{min}) \cdot K^n}{K^n + (x_{Bit1\ output})^n} \right) - \gamma \cdot m_B \quad \text{(Equation 11)}$$

$$\frac{dm_C}{dt} = \xi \cdot \left(dn + \frac{(y_{max} - y_{min}) \cdot K^n}{K^n + (x_A + x_B)^n} \right) - \gamma \cdot m_C \quad \text{(Equation 12)}$$

In this set of equations, the scaling factor (ξ) and the degradation rate (γ) are constants, which are the ones in the single-bit counter. **The shifted Hill function is specified by the parameters:** y_{min}, y_{max}, K, n. The meaning of the variables has been illustrated in Figure 3.

Mathematical model for the connector type 2. With the diagram of the connector type 2 shown in Figure 8B. We use the following dynamical equations to model the dynamics of the 6 gates in the connector type 2 circuit module. In general, take the n-bit counter as an example, the connector between nth and the $(n-1)$th 1-bit counters follow the following ODE:

$$\frac{dm_A}{dt} = \xi \cdot \left(dn + \frac{(y_{max} - y_{min}) \cdot K^n}{K^n + (x_{Connector_{N-1}\ output})^n} \right) - \gamma \cdot m_A \quad \text{(Equation 13)}$$

$$\frac{dm_B}{dt} = \xi \cdot \left(dn + \frac{(y_{max} - y_{min}) \cdot K^n}{K^n + (x_{Bit_{N-1}\ output})^n} \right) - \gamma \cdot m_B \quad \text{(Equation 14)}$$

$$\frac{dm_C}{dt} = \xi \cdot \left(dn + \frac{(y_{max} - y_{min}) \cdot K^n}{K^n + (x_A + x_B)^n} \right) - \gamma \cdot m_C \quad \text{(Equation 15)}$$

$$\frac{dm_D}{dt} = \xi \cdot \left(dn + \frac{(y_{max} - y_{min}) \cdot K^n}{K^n + (External\ input)^n} \right) - \gamma \cdot m_D \quad \text{(Equation 16)}$$

$$\frac{dm_E}{dt} = \xi \cdot \left(dn + \frac{(y_{max} - y_{min}) \cdot K^n}{K^n + (x_C)^n} \right) - \gamma \cdot m_E \quad \text{(Equation 17)}$$

$$\frac{dm_F}{dt} = \xi \cdot \left(dn + \frac{(y_{max} - y_{min}) \cdot K^n}{K^n + (x_D + x_E)^n} \right) - \gamma \cdot m_F \quad \text{(Equation 18)}$$

The parameters ξ and γ are constants and are the same as the ones described in connector type 1 dynamical equations. The shifted Hill coefficients are y_{min}, y_{max}, K, n.

So far, we have written all ODEs that describe the underlying dynamical systems for each of the modules. Therefore, one can build a full mathematical model for an arbitrary multi-bit counter just by merely assembling together the corresponding modules. Additionally, one should note that because we have assumed

that repressor-promoter binding/unbinding dynamics occur at a fast time scale, we can assume that the promoter concentration (strain density) is at a quasi-equilibrium state. Next, we construct the counter's parameter searching algorithm.

Parameter optimization

We optimize the parameters by a computational search. The goal is to find the valid range of the input pulse width and find the minimum duration for the relaxation time (defined as the time difference between the previous signal off time and the next input signal). Our simulation results show that within the valid range of the input duration and relaxation times, the designed circuits behave as desired.

The key to successfully constructing a feasible counter circuit is to find valid ranges of the parameters that the counting dynamics allows. Specifically, when an input signal is given to the counter, the SR-latch should exhibit a switching behavior. The appropriate duration of an input pulse is the one that makes the state of the SR-latch switch to the other steady state as shown in, Table 1.

All gates in the multi-bit counter system share a single input-output characteristic for the gate activation function. In reality, the activation functions for different transcriptional logic NOR gates can be different. Experimentalists have built a library of transcriptional gates with various dynamical ranges for different genetic gates. However, in our multi-bit counter design, we constrain all genetic gates to share the same activation function to reduce computational complexity and avoid combinatorial explosion. Feasible parameter ranges for the activation function will be explored for a multi-bit counter system.

The parameters essential for constructing a feasible multi-bit counter system are the shifted Hill coefficients y_{min}, y_{max}, K, n, the signal amplitude A, and the pulse width δ. The shifted Hill coefficients define the counter system's intrinsic dynamics. The external signal amplitude and the pulse width can be controlled exogenously. From the engineering and control perspective, given a system dynamics with a fixed set of parameters for the shifted Hill coefficients, researchers will want to find the feasible range for the counter's pulse width and amplitude. On the other hand, if researchers have an interest in a particular range of pulse widths, our framework will provide estimates of the optimal parameters.

In what follows, we include a detailed description of what the essential parameters are and why a feasible counter system depends on them:

- **External pulse width** δ. It specifies the duration of the external pulse. The feasible ranges of δ's mean and standard deviation values are important indicators that represent the counter's desired operating regime and the robustness against perturbations respectively.
- **External signal amplitude** A. It specifies the external signal strength. Knowing the feasible range for these parameters can be beneficial for experiments when dealing with both the quantity and cost of using chemicals or biological inducers.
- **The maximum value y_{max} of Hill function.** To control it from an engineering perspective, one can manipulate the gates' dynamical range to be reactive to certain input pulse widths. It also relates to the value of the circuit steady-state outputs. Controlling "up" can be useful when considering the counter system as a thresholding machine.
- **Equilibrium constant** K**.** It controls the cut-off point of the activation function.
- **cooperativity index** n**.** It controls the steepness of the activation function. In general, the synthetic gates with smaller "n" values turn out to be easier to be synthesized (Moon et al., 2012).

In addition, if the characteristics of the gates are given, we need to determine the parameter ranges that are most robust to external perturbations. Specifically, for each group of Hill coefficient pairs (n, K), we need to determine the corresponding ranges of δ, A and y_{max} around their means.

To determine if a counter is "good" or "bad", we use a cost function to select the parameters. The general optimization procedure is stated as below:

- **Alignment of the carry bits:** For the 2-bit counter which is a special case of a general *N*-bit ($N>3$) , the carry bit dynamics alone is an indicator of whether the connector type 1 can pass the information to the next level. Taking the 3-bit counter as an example, we consider that counting will start at the time when the two carry bits are first aligned, which coincides with the first time that the last bit switches, and thus sets the initial time reference point of the 3-bit counter dynamics.
- **Building a local switch cost:** In response to an external input signal, a local switch cost is needed to determine whether to switch the steady states at each bit level. For example, for the first bit level in the multi-bit counter system, one is expected to have the output switch from a low state to a high state (or reversed direction) upon detecting an input pulse. At the second bit level, the output is expected to switch once every time the first-bit output switches twice. For the third bit level, the output should switch once every time the first-bit output switches four times.
- **Periodic activation criterion:** For a multi-bit counter, especially when *N* is large, we require the first-bit to switch upon every input pulse stably. For example, a full cycle for a 3-bit counter requires at least eight stable switches for the first bit, four stable switches for the second bit, and one stable switch for the third bit. We have illustrated the switching pattern of the 3-bit counter in Figure 10B.
- **Global cost function:** In order to have a functional counter with a pulse that has the right duration, we need to combine all the constraints mentioned above for constructing a global cost function. In addition to the local cost criterion, we need to compute two additional quantities that can impose the conditions on combining all local switching costs at different bit levels altogether. These two quantities are:

First,

$$\mathbf{Q}_{cp}^{i}=\sum_{j=1:2^{i}:M} Binary\left(\mathbf{Q}_{j}+\mathbf{Q}_{j+1}\right), \overline{\mathbf{Q}}_{cp}^{i}=\sum_{j=1:2^{i}:M} Binary\left(\overline{\mathbf{Q}}_{j}+\overline{\mathbf{Q}}_{j+1}\right)$$

where *i* indicates the *i*th bit level for the multi-bit counter, *j* represents the index of the *j*th pulse and *M* represents the total number of input pulses seen by the counter. The notation $j=1:2^{i}:M$ means to sum *j* from 1 to *M* every 2^{i}. The "Binary" operator in the formula will compare the $\mathbf{Q}$ or $\overline{\mathbf{Q}}$ with $up/2$. if $\mathbf{Q}$ or $\overline{\mathbf{Q}}>up/2$, the operator will return 1, otherwise it will return 0. Take the 3-bit counter as an example, for a full counting cycle in which the first bit takes 8 pulses, the $\mathbf{Q}_{cp}^{1}$ and the $\overline{\mathbf{Q}}_{cp}^{1}$ should be 4 for the first bit level, $\mathbf{Q}_{cp}^{2}$ and the $\overline{\mathbf{Q}}_{cp}^{2}$ should be 2 for the second bit level and $\mathbf{Q}_{cp}^{3}$ and the $\overline{\mathbf{Q}}_{cp}^{3}$ should be 1 for the third bit level.

Second,

$$\mathbf{Q}_{mp}^{i}=\prod_{j=1:2^{i}:M} Binary\left(\mathbf{Q}_{j}+\mathbf{Q}_{j+1}\right), \overline{\mathbf{Q}}_{mp}^{i}=\prod_{j=1:2^{i}:M} Binary\left(\overline{\mathbf{Q}}_{j}+\overline{\mathbf{Q}}_{j+1}\right)$$

where *i* indicates the *i*th bit level for the multi-bit counters, *j* represents the index of the *j*th pulse and *M* represents the total number of input pulses that are inputs to the counter. The "Binary" operator has been defined above. If a set of parameter gives a feasible counter, then $\mathbf{Q}_{mp}^{i}$ and $\overline{\mathbf{Q}}_{mp}^{i}$ should all be equal to 0.

Therefore, when we impose the global condition for a feasible counter, we merge all constraints mentioned above with the "AND" operation for determining the feasible parameter set of a multi-bit counter. We define the overall cost function as follows:

$$\mathrm{Cost}_{\mathrm{global}}=\bigwedge_{i=1}^{N}\left(\mathbf{Q}_{cp}^{i}\wedge\mathbf{Q}_{mp}^{i}\wedge\overline{\mathbf{Q}}_{cp}^{i}\wedge\overline{\mathbf{Q}}_{mp}^{i}\right).$$

If all conditions are True, then $\mathrm{Cost}_{\mathrm{global}}=1$, otherwise $\mathrm{Cost}_{\mathrm{global}}=0$. With the cost function defined above, one can set the *N*-bit counter criterion with *M* pulses seen by the system. The above optimization will lead us to find the feasible parameter sets.

We summarize the procedure in the following algorithm. The pseudo-code algorithm shows how we optimize the system and find the feasible parameter ranges below:

Algorithm 1: Optimization of choosing feasible parameters for a generic multi-bit counter

Result: Global cost = 1 gives the feasible parameter set for a counter

start = a set of aligned times for all the carry bits' peaks;

if start is not Ø(empty set) **then**

  thred = a threshold value for detecting the carry bit peaks;

  for i = 1:N **do**

    compute local cost for the gate;

  end

  compute global cost for the gate;

  if all conditions in global cost are True, **then**

    Global cost = 1

  else

    Global cost = 0

  end

else

  Global cost = 0

end

With the general algorithm stated above, we can generate a database of feasible parameters for any multi-bit counter system. We showcase a snapshot of how a 3-bit counter database looks like in Figure 16. The parameter ranges are chosen according to the available experimental genetic gates database shown in Figure 11. Surprisingly, we found that *A* behaves mostly like a "free" variable that only needs to pass a low threshold value. In other words, *A* has a wide feasible range. This gives us the advantage of using a relatively small number of inducers for making the counter to work, and it also indicates that the counter can be less sensitive to the input signal concentration as long as it passes a small threshold.