ACS SyntheticBiology

Research Article

pubs.acs.org/synthbio

An Implementation-Focused Bio/Algorithmic Workflow for Synthetic Biology

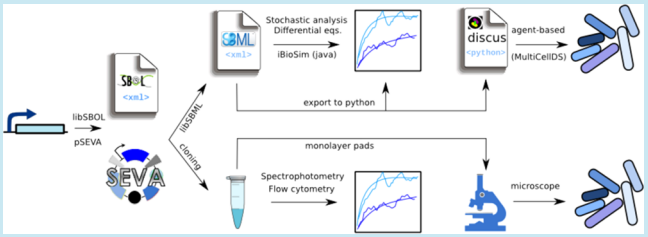
Angel Goñi-Moreno,[†] Marta Carcajona,[†] Juhyun Kim,[†] Esteban Martínez-García,[†] Martyn Amos,[‡] and Víctor de Lorenzo*[,†]

[†]Systems Biology Program, Centro Nacional de Biotecnología, Cantoblanco, 28049 Madrid, Spain

[‡]Informatics Research Centre, Manchester Metropolitan University, Manchester M1 5GD, United Kingdom

Supporting Information

ABSTRACT: As synthetic biology moves away from trial and error and embraces more formal processes, workflows have emerged that cover the roadmap from conceptualization of a genetic device to its construction and measurement. This latter aspect (*i.e.*, characterization and measurement of synthetic genetic constructs) has received relatively little attention to date, but it is crucial for their outcome. An end-to-end use case for engineering a simple synthetic device is presented, which is supported by information standards and computational methods and focuses on such characterization/measurement. This workflow captures the main stages of genetic device design and description and offers standardized tools for both population-based measurement and single-cell analysis. To this end, three separate aspects are addressed. First, the specific *vector features* are discussed. Although device/circuit design has been successfully automated, important structural information is usually overlooked, as in the case of plasmid vectors. The use of the Standard European Vector Architecture (SEVA) is advocated for selecting the optimal carrier of a design and its thorough description in order to unequivocally correlate digital definitions and molecular devices. A digital version of this plasmid format was developed with the Synthetic Biology Open Language (SBOL) along with a software tool that allows users to embed genetic parts in vector cargoes. This enables annotation of a mathematical model of the device's kinetic reactions formatted with the Systems Biology Markup Language (SBML). From that point onward, the experimental results and their *in silico* counterparts proceed alongside, with constant feedback to preserve consistency between them. A second aspect involves a framework for the *calibration* of fluorescence-based measurements. One of the most challenging endeavors in standardization, metrology, is tackled by reinterpreting the experimental output in light of simulation results, allowing us to turn arbitrary fluorescence units into relative measurements. Finally, integration of single-cell methods into a framework for *multicellular* simulation and measurement is addressed, allowing standardized inspection of the interplay between the carrier chassis and the culture conditions.



INTRODUCTION

Synthetic biology is concerned with the rational design and construction of biological information-processing devices.[1] The rigorous application of *engineering principles and processes* is fundamental to the success of this endeavor.[2−4] Significant attention is now being paid to the development of standardized *workflows*[5,6] that describe sequences of biological and algorithmic processes required to obtain a desired outcome. Such workflows specify a *tool chain* for synthetic biology. The anticipated benefits of using them include *modularity* (allowing individual processes to be implemented in several different ways), *robustness*, and *scalability*.

One of the overarching challenges for the field is the end-to-end *automation* of biodesign,[7,8] a process that includes two main stages:[6] [i] automatic selection and/or construction of biological components and their assembly into a network that, in principle, performs information processing according to a high-level specification and [ii] fine-tuning of the system components and/or architecture to obtain the desired performance. The first part of this process concerns the detailed *specification* of the components to be used[9,10] or fabricated,[11−13] the attendant data representation and storage issues,[14] and the correct arrangement of components into a *device/circuit* that can implement a given (logical) function. A wealth of so-called *bio-CAD* tools now exist for this latter task,[15,16] *e.g.*, SBROME,[17,18] TinkerCell,[19] SynBioSS,[20] and CELLO.[21] In terms of *fine-tuning* (the second stage), recent developments use postassembly modification of constructs based on observed network behavior[6] or the evolution of cell models,[22] facilitating an iterative *homing-in* approach toward genetic designs.

The work presented in this article focuses on the latter stages of the device/circuit engineering process (that is, the implementation stages that follow the *initial* development of a given design). The specific issues addressed with the

Special Issue: Synthetic Biology in Europe

Received: January 28, 2016
Published: July 25, 2016

DOI: 10.1021/acssynbio.6b00029
ACS Synth. Biol. 2016, 5, 1127−1135

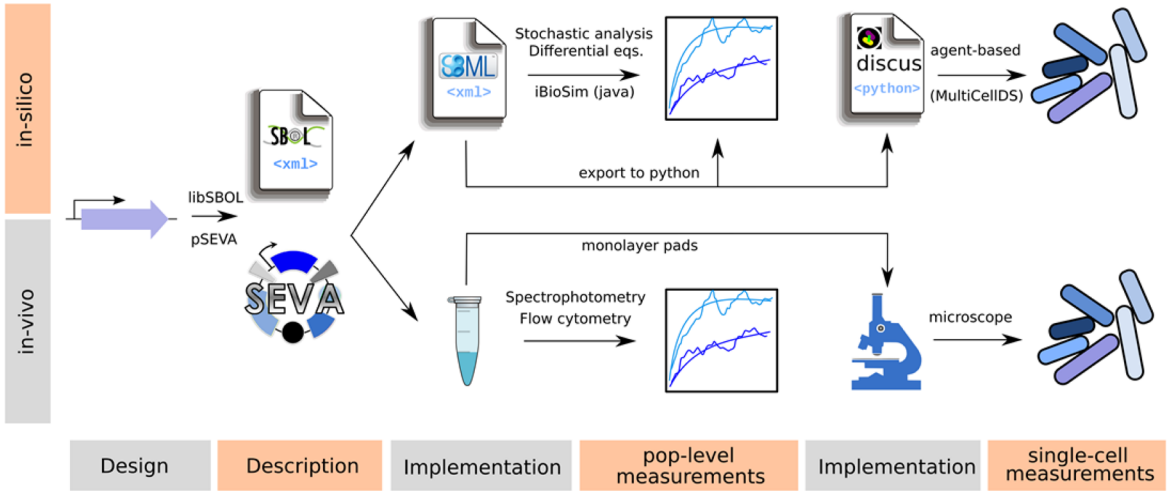


Figure 1. Workflow for an end-to-end synthetic biology use case. The description that follows (and modifies) the design of an idea is the starting point for the consequent experimental and computational methods. The device and its carrier vector are described using the Standard European Vector Architecture (SEVA) format for the *in vivo* workflow and the Synthetic Biology Open Language (SBOL) standard for the parallel *in silico* process. A first implementation round is then performed via synthesis and cloning methods in the wet lab and via Systems Biology Markup Language (SBML) for the modeling. The resulting material is then used for different measurements. First, laboratory equipment is used for population-based experiments (spectrophotometry and flow cytometry) to compare the output against simulation software (iBioSim and *ad hoc* Python code). Another implementation round prepares the samples for single-cell measurements. On the computational side, the SBML model is exported to a Python script that is ready to be used with the Discrete Simulation of Conjugation Using Springs (DiSCUS) software for cell movement. On the other side, the cells are grown on an agarose pad for two-dimensional populations that allow matching results.

workflow discussed below include formalization of the device description regarding the sequences of the parts of the system to be constructed and the effect of plasmid vectors on performance. An early technical standard for the description of biological parts was the BioBrick,[23,24] which is appropriate for the assembly of DNA segments. However, a key consideration (which is virtually neglected by earlier standards) is the variety of *plasmid vectors* that are available for the deployment of biological devices. In reality, the choice of plasmid vector can dramatically affect the performance of a given device; plasmid features such as replication origin, selection markers, and expression system need to be carefully selected.[25] Finally, the correlation of experimental observations versus simulation results is addressed. As computational tools to aid biodesign become more commonplace, more uniform types of circuits are reported in the literature. However, once they are built, the process of *measuring* the behavior of the designed system (in order to assess its fidelity to the desired output) may still vary substantially. This is the case because few existing workflows consider measurement and teams are free to choose their own tools for this mission. Mathematical and computational modeling have become fundamental tools in synthetic biology, but they are effective only when combined with useful *in vivo* observations of synthetic systems. In this context, the workflow reported below describes a methodology for easily mapping simulation results onto laboratory measurements.

RESULTS AND DISCUSSION

Figure 1 shows the different stages of the workflow discussed in this article. By the adoption of a combined experimental *in vitro/in silico* approach, the two perspectives become tightly coupled at key points. The various stages are ordered along time from left to right, and they begin once a device design is established. It should be noted that issues of design were not considered and that instead the workflow focuses on implementation and measurement. The first stage in the workflow is *description*, in which the design of the desired construct is captured by some representation(s). This then feeds into the *implementation* stage, in which the construct is built (or modeled). Once the device has been implemented, *population-level measurement* is carried out in order to obtain aggregate performance metrics. This then feeds into a *second implementation* phase, which facilitates closer (single-cell) observations. These workflow stages are detailed as follows.

Description. In order to obtain reliable and robust performance of the device, it is essential to have control over the vector and be able to compare its performance with the same plasmid in multiple scenarios. In order to achieve this for the *in vivo* component, the Standard European Vector Architecture (SEVA)[26] was adopted. This is an active[27−30] standard for the physical assembly of plasmid vectors and their nomenclature as well as an online database of functional sequences and constructs available to the community. Along with the SEVA depiction of the *plasmid* there is a digital representation of the *device/circuit* for the *in silico* component of the workflow, for which the Synthetic Biology Open Language (SBOL)[31] is used. This provides a *standard exchange format* for synthetic biology designs (between research groups and between different toolkits).

Implementation. In the first *in vitro* implementation phase, the vector is assembled using standard molecular biology procedures. This results in the synthesis of DNA segments, which are then inserted into the carrier plasmid. In parallel with this process (*i.e.*, during the *in silico* implementation phase), a standardized digital description using SBOL is constructed, with one SBOL document per genetic construct (Supplementary File S2). These documents are then combined using a Java-based tool (Supplementary Tool S1), resulting in a single SBOL file containing the sequences of interest in the correct cargo position according to the restriction enzyme sequences. This tool identifies those SBOL components that are common across components (*i.e.*, the restriction sites) and replaces all of

DOI: 10.1021/acssynbio.6b00029
ACS Synth. Biol. 2016, 5, 1127−1135

the information that exists in the cargo section from restriction site to restriction site with the functional cassette of interest. After this step, both the plasmid containing the device and its representation are fully standardized. It should be noted that the cassette to be inserted into the cargo section may be assembled (using any *wet* technique) or synthesized; the restriction sites of the SEVA vectors can be used in this implementation stage or in possible postmeasurement debugging.

Measurements. The use of mathematical modeling and computational analysis has become a fundamental part of synthetic biology because of the information they provide concerning the mechanical behavior of the systems. However, this potential can be used effectively only when it is combined with direct *in vivo* measurements.[32] This is helped by ongoing advances in metrology (see, *e.g.*, the TASBE tools at https://synbiotools.bbn.com). Recently, attempts have been made to standardize, *e.g.*, relative promoter units (RPU)[33] as a measuring standard for promoter activity based on a comparison against a reference promoter. On a more abstract level, the polymerase operations per second (PoPs) measure[9] is abstracted as the signal carrier in transcriptional devices. However, none of these methods is free of controversy.[15]

In order to simulate the model constructed in the implementation phase, the iBioSim[34] tool was used. Conveniently, iBioSim exports reactions to a single Systems Biology Markup Language (SBML)[35] file (Supplementary File S3), which is a computational standard for the representation of biochemical networks. Importantly, this allows the SBML *biochemical model* of the device to be linked with the SBOL description of its DNA components using the methodology described by Roehner and Myers[36] (Supplementary File S4). In turn, this connects (via SBOL) with the SEVA description of the vector, giving seamless integration of information across different standards that are used for different levels of description. An additional application (Supplementary Tool S2, based on libSBML[37]) was developed to convert a given SBML file into Python-coded scripts (Supplementary File S5), which are used for deterministic and stochastic simulations. Importantly, the details of the SBML model (*i.e.*, rates) correspond not only to the device itself but also to its carrier vector. This significantly reduces output variability: by the inclusion of details of the vector in the model characterization (via SEVA/SBML), the possibility that the carrier plasmid might later change as a result of decisions made at the implementation phase is considered. Any such change will in turn inevitably (although sometimes subtly) affect the observable behavior of the model when implemented. Including details of the vector thus allows fluctuations due to variable plasmid selection to be considered.

The inclusion of an extra step within the workflow for *multicellular* analysis also helps to reduce the variability caused by both the chassis and the culture conditions, as they add their own effects to the construct and its carrier. If the device has to be used under different scenarios, the cellular behavior should be quantified. There are behaviors in the example provided that cannot be measured with the cytometer (*i.e.*, noise inheritance or cell movement) and require time-lapse microscopy in order to be quantified. The parameters corresponding to these behaviors are therefore fitted according to single-cell measurements. Again, this information adds value to a potential specification sheet that accompanies the *in vivo* system.

Spectrophotometry is used to measure the fluorescence signal of the entire cell population; dividing this by the optical density at 600 nm (OD_{600}) over time yields the average fluorescence value per cell in the culture. Experimental values are used to fit kinetic rate parameters in the mathematical models so they produce similar profiles. Importantly, in the graphs that follow, the *y* axis refers to *arbitrary units* of fluorescence in experimental observations and the *number of molecules* (*e.g.*, mCherry proteins) in the simulated observations. Matching the latter with the former gives an important reference point concerning measurements that allows interpretation of subsequent results.

Stochastic analyses are then done in order to characterize noise in the system using the well-established Gillespie algorithm.[38] On the experimental front, data on noise is obtained using flow cytometry, which allows the user to check the fluorescence intensity value for (in principle) every single cell in the bacterial culture. Although the ready-to-use graphs produced by the cytometer (Figure S1) are used as standard in most laboratories, *raw values* before they are processed for presentation (normally in a *black box* fashion, which is opaque to the user) are preferred. There are three main reasons for using raw cytometry data: [i] Cytometers *count* cells using variable intervals of fluorescence at high values of a logarithmic scale that is not always constant and depends on a specific machine setup. This processing therefore introduces variability that is hidden from the user. [ii] Cell-specific values are needed in order to make direct comparisons with simulated cells within our framework. (3) Raw data values are more amenable to importing and processing by various tool-chain components. In contrast, automated extraction of specific values from graphs produced by cytometers introduces unnecessary complications and the possibility of misreading data.

A simulated cytometry graph is obtained by running the Python version of the reactions (see Materials and Methods). This offers two potential benefits. On one hand, it gives a computational method (via an SBML model) for discarding invalid values from the raw cytometry information (see the later case study for an example). On the other hand, by overlap of the experimental and simulated plots, the arbitrary units (a.u.) of the cytometer and those from the spectrophotometer can be correlated. This procedure therefore seems to help in unifying machine-based measurements in the laboratory, as shown in the case study below.

Implementation. The behavior of the device under study is inevitably affected by the specific attributes of the host cell. A thorough characterization of a construct should therefore include information about the performance of the *chassis*[39] (which, in the case shown, is *Escherichia coli* CC118[40]). Rather than simply providing *added value*, this information is of vital importance in the case of multicellular applications,[41,42] which are becoming increasingly important as cell-to-cell communications are increasingly well-understood and customized.[43,44]

In order to study the behavior of devices *in vivo*, the DiSCUS[45] package previously developed to study bacterial growth was adopted as an agent-based simulation tool. Importantly, this platform considers *physical forces* between rod-shaped bacteria and is applicable to a wide range of organisms. DiSCUS uses the previously generated Python scripts for the intracellular genetic network that is implemented in the cells of interest. The SBML model is therefore embedded into the cellular objects of the agent-based simulator. It should be noted that there is a standard currently under development

DOI: 10.1021/acssynbio.6b00029
ACS Synth. Biol. 2016, 5, 1127−1135

called the MultiCellular Data Standard (MultiCellDS; http://multicellds.org/) that aims at sharing multicellular experimental, simulation, and clinical data. Hopefully, when released, it will facilitate partaking of configuration parameters for a specific chassis performance. In the case under examination (see below), a two-dimensional culture was prepared on an agarose pad,[46] and the cells were allowed to grow on a monolayer in order to facilitate visualization in the microscope.

Single-Cell Measurements. The movement and growth of the simulated cells were first calibrated according to experimental observations. For this, the successive positions of a specific cell until division were monitored and the displacements of its offspring during their lifetime(s) followed. These results were matched against the equivalent information obtained from the simulations and adjusted to DiSCUS parameters for fitting the experiments. In short (see Materials and Methods for more details), this information yields the most relevant features to prioritize in DiSCUS in order to reproduce the movement of the cells *in vivo* (see the example below).

Spatial Measurements. After characterization of the dynamics of the chassis that hosts the construct, its performance in a spatial scenario was measured. For this, we quantified the fluorescence intensity of the device *in vivo* and obtained a pixel-based image analysis of the specific color (red in the example) captured by the microscope. The ensuing analysis translates the scale bar into values proportional to those used in the mathematical model (see Materials and Methods for details of this conversion). As a consequence, a simulation run with the system's equations inside DiSCUS bodies could be directly compared with experiments with respect to device function.

Case Study. A combined *in vivo/in silico* study involving a simple construct was picked as an informative example of the proposed workflow. The starting point was an *always-on* gene expression device, *i.e.*, a genetic module enabling constitutive transcription and translation of a reporter gene (mCherry) (Figure 2A). Although this setup involves just a few components, it is also instrumental to highlight the main focus of this work, which lies in the *measurement* of such devices. That is, the complexity of the device to be constructed is less critical than the management of its *output*. Since fluorescence measurements are taken in fundamentally in the same way regardless of the size or complexity of a synthetic device, the question at stake is how such metrics might be *standardized* and related back to *in silico* studies in a useful and meaningful way.

As shown in Figure 2A, the two subcomponents of the device were [i] the *PEM7* constitutive promoter, and [ii] the red fluorescence reporter gene *mCherry* (see Materials and Methods and Supplementary File S1 for details). Once the initial design was in place, the system was taken to the description stage, where *PEM7*–*mCherry* was digitally formalized and physically built. The pSEVA231 vector (Figure 2B) was selected to implement the design. This plasmid contains a kanamycin marker, an origin of replication pBBR1, and the default cargo segment. As the cargo segment is a sequence of restriction sites, specific locations have to be selected for insertion of the desired parts. As shown in Figure 2A, the promoter component was flanked by restriction sites PacI and AvrII, while HindIII and SpeI were chosen for the reporter gene, thereby leaving a number of *empty* sites in between for possible future use.

Once the description phase was complete, the system was taken to the implementation stage. Figure 3A highlights the

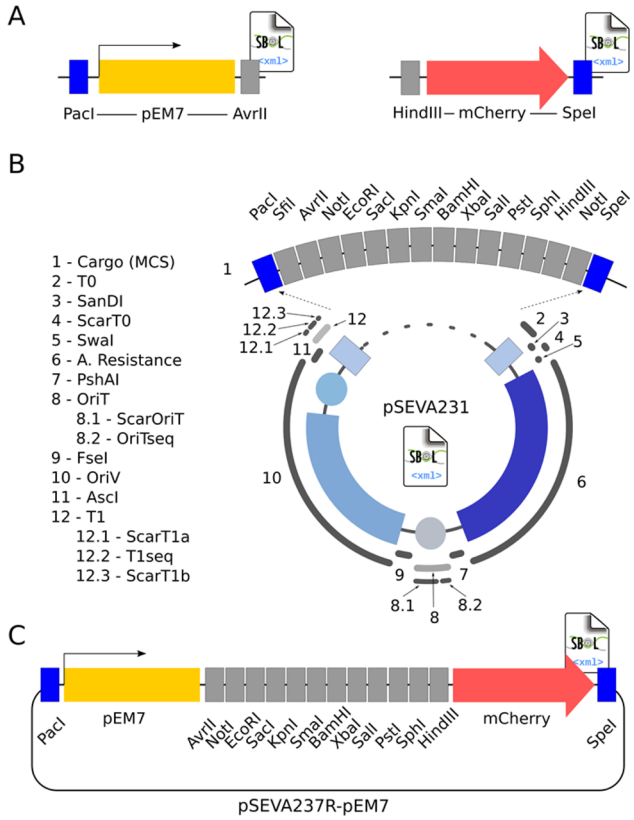


Figure 2. SBOL description of device and SEVA components. (A) Design modification where the components are flanked by the selected restriction sites that specify their situation inside the SEVA vector. The constitutive promoter *PEM7* is surrounded by PacI and AvrII sites, whereas the reporter mCherry is flanked by HindIII and SpeI. An SBOL document per component is created. (B) The plasmid selected to harbor the device is SEVA no. 231 (kanamycin resistance, pBBR1 origin of replication, default cargo). All of the vector features are recorded in a single SBOL document, including the cargo (multiple cloning site) component for further assembly of device-forming parts. (C) The *in vivo* and *in silico* protocols for building the final construct have the same basics, *i.e.*, introducing parts sequentially in the carrier vector. A software tool (Supplementary Tool S1) allows this to be done with SBOL documents.

kinetic rates involved and the ordinary differential equations (ODEs) that govern the continuous functioning of the *always-on* device. After cloning, Figure 4A shows the results for average fluorescence value per cell in the culture along with deterministic simulation runs (based on the ODEs) for both the SBML model (implemented using iBioSim) and its corresponding Python script. The stage was thus set to move to the population measurement phase.

Figure 4B shows the fluctuations in molecular levels of the reactions of Figure 3A when the Gillespie algorithm was run on the SBML model (iBioSim) and its corresponding Python file. As expected, the observed variability was the same in both, as the kinetic rates remained unchanged (*i.e.*, the same as in the ODEs). The mean value was precisely situated on the steady-state value of the deterministic simulation. Raw data from the cytometer are plotted in Figure 4C, where the bimodal curve indicates that approximately half of the cells displayed strong fluorescence while the rest expressed none (or very little). The latter group corresponded to invalid values and could be discarded, as indicated by the control data (the same strain without the plasmid) and the already processed graph (Figure

DOI: 10.1021/acssynbio.6b00029
ACS Synth. Biol. 2016, 5, 1127–1135

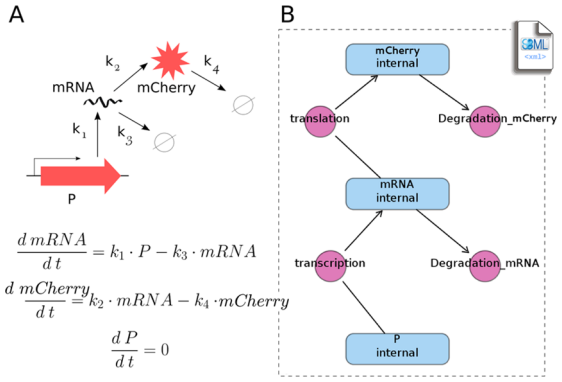


Figure 3. Mathematical modeling and its SBML format (A) Kinetic reactions (top) and the system's ordinary differential equations (ODEs) (bottom). The device's behavior can be effectively simulated with just four kinetic constants: the constitutive promoter P facilitates reporter transcription with rate k_1, and the resulting mRNA is translated with rate k_2, leading to the formation of red fluorescent protein (RFP); both elements are degradated at rates k_3 and k_4, respectively. The ODEs governing continuous dynamics are shown. (B) Scheme of the SBML model produced with the software iBioSim, a computer-aided design (CAD) package for systems biology. In the screen shot, blue elements represent substrates and red circles hide reaction rates. After the parameters are set, iBioSim allows the user to export the model to an XML file formatted following the SBML standard.

S1). Moreover, further microscopy tests showed strong fluorescence in all of the cells with a relatively narrow noise interval, confirming the correct elimination of that nonexpressing cell group. As described in the workflow description, this gives a computational standard way of discarding invalid values from raw cytometry information. Moreover, it yields a method for correlating outputs from different pieces of laboratory equipment. We illustrate this in the graph of Figure 4C, showing that arbitrary units of the cytometer could be correlated with those from the spectrophotometer: 1 a.u. in the former ≈1.2 a.u. in the latter (see Materials and Methods for details).

After population-level measurements were performed, the workflow proceeded to single-cell measurements. Figure 5A shows the results of experiments to track cell movements. Figure 5B shows the positions of a bacterium (from Figure 5A) until division and then the displacements of its daughter cells during their lifetimes. Figure 5C shows the most relevant features that need to be added to DiSCUS in order to reproduce the movement of the bacteria, starting from a very simple growth algorithm (which returns unrealistic patterns; Figure 5C.1). Ultimately, the qualities to be considered include [i] *cell size variations* (due to conditional growth), [ii] changes in *transversal angles* after division, [ii] *randomized directions of movement*, and [iv] slight *attraction between cells* (in order to avoid the appearance of holes within the colony).

Figure 5D compares the synchrony of growth within the experimental and simulated cells, yielding suggestions as to how to uncouple growth events. These graphs show the length of each cell in the population over *time* (in laboratory experiments) or *iterations* (in the simulation). Starting with just two cells (the same setup as in Figure 5A) that grow and divide at the same time, it becomes apparent that the lengths of the cells are no longer synchronized after the second division (eight cells in total).

The spatial scenario is considered next. Figure 6A shows the results of measuring the fluorescence intensity of the *PEM7–mCherry* device when placed in the *E. coli* CC118 strain along with a pixel-based image analysis of the red color captured by the microscope. As indicated above, the scale bar of the analysis was translated into values proportional to those of the mathematical model of Figure 4B. A simulation run in DiSCUS using the system's equations (Figure 6B, left) can be directly compared against experiments. For instance, it is possible to verify that daughter cells share output levels as they directly *copy* their mother's device at a given time (Figure 6B) and that cells with slower growth tend to display a stronger light signal (due to the accumulation of fluorescent proteins).

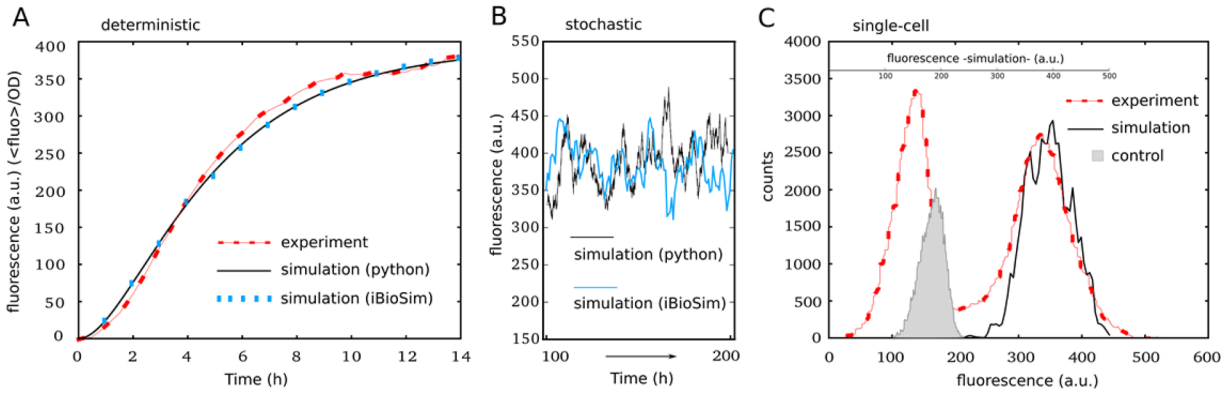


Figure 4. Population-based measurements in experimental and simulation setups. (A) Deterministic functioning of the device in terms of fluorescence intensity over time (during 14 h), averaging the value of the whole population. The red line corresponds to experimental results, while the blue and black lines show simulation runs of the model's differential equations with iBioSim and Python code, respectively. Experimental values were used to fit rate numbers in mathematical models so they produce similar continuous lines. (B) Stochastic behavior of the system according to simulations. The blue line shows the results of running the Gillespie algorithm with iBioSim, whereas the black line shows the Python script behavior. As expected (same algorithm with equal parameters), the fluctuations are alike. (C) Fluorescence intensity values for each cell in the population measure variability and expression noise. Experimental raw data extracted by flow cytometry (without processing by the cytometer; see the text for details) correspond to the red line. The black line results from counting expression values in the simulation with the Python script, while the gray area represents the control (plasmid-free cells) measured experimentally. It should be noted that scales are different for the simulation and experimental lines, standing for variability within arbitrary units (a.u.).

DOI: 10.1021/acssynbio.6b00029
ACS Synth. Biol. 2016, 5, 1127–1135

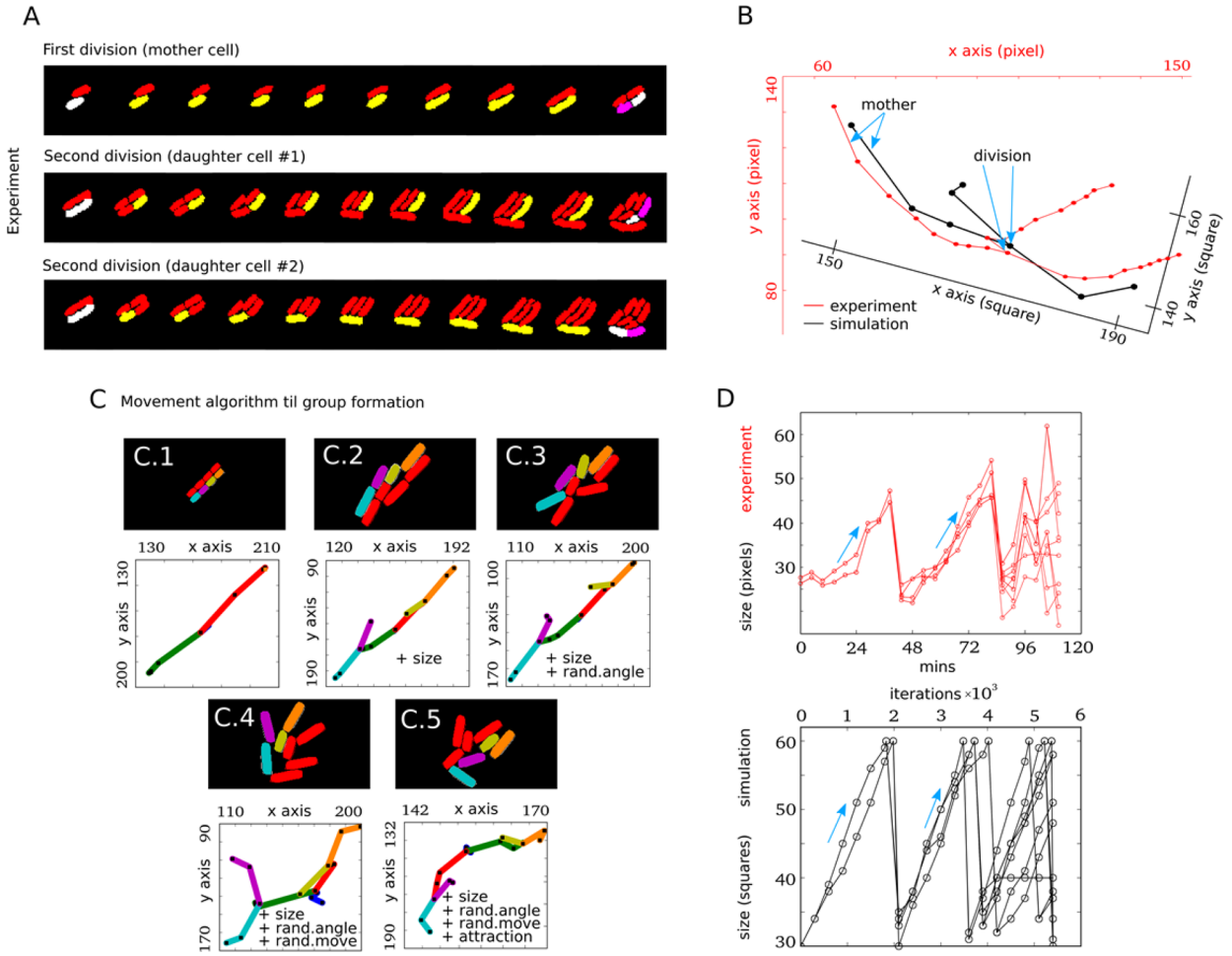


Figure 5. Characterization of chassis mechanics. (A) Tracking cell lineages in an experimental setup. Starting from the division of a single cell (top) the movements of its daughters were followed (middle and bottom) in order to define their movement behavior until the next division. (B) Position coordinates are recorded during the experiment (red line) and simulation (black line) to fit parameters by comparison of the two outputs. Cell traces are overlapped for visualization purposes, and the axis is rotated accordingly to show the dimensions. (C) Parameter estimation for cell movement. Different features are included sequentially in order to get the final moving procedure for *in silico* simulations. Starting from inaccurate movement (C.1), we added size variability due to pressure (C.2), random angles after division (C.3), irregular motion changes (C.4), and slight cell attraction to simulate viscous bodies (C.5). All of the simulations started from a single cell, and one lineage is colored to monitor coordinate positions. (D) Synchrony of cell growth. The length of each cell (*y* axis) is monitored over time (*x* axis) in both scenarios (experiment, top; simulation, bottom). The initial cells grow at the same time until the division point is reached, whereas the third generation of cells grow asynchronously.

CONCLUSION

The development of standardized workflows that allow for robust and reproducible genetic constructs is one of the contemporary challenges of synthetic biology. The framework presented above contributes to this endeavor by setting both computational and experimental approaches to build and measure synthetic devices using a simple synthetic device as an example. This approach is different from and complementary to other efforts that focus on specific steps, *e.g.*, automated circuit design,[5,16,18] mathematical modeling,[20] single-cell analysis,[46] metrology,[33] data representation,[31] and postconstruction modification.[6] In contrast, the workflow presented in this article can make use of several of them by concentrating on output measurements, thereby enriching design-oriented efforts. While the literature records other workflow propositions,[15] they tend to focus on enumeration rather than application of techniques. Instead, the tools presented here allow linking of standards, *e.g.*, merging of SBOL documents for SEVA description and the scripts to translate SBML into Python. The system will hopefully provide a useful starting point for newcomers to the field as well as (more generally) a standard workflow for robust programming of biological systems.

MATERIALS AND METHODS

Strains and Plasmids. The *E. coli* strain used in this work was CC118.[40] The vector plasmid for the expression device was pSEVA231 (kanamycin resistance, pBBR1 origin of replication, and default SEVA cargo) selected from the database http://seva.cnb.csic.es/. The *PEM7* promoter was cloned in pSEVA231 as a PacI/AvrII fragment and the mCherry reporter gene as a HindIII/SpeI DNA segment. The resulting plasmid was named pSEVA237R-*PEM7* (available in the SEVA database). An important aspect is that sequences of interest encoding the expression device were edited to remove any restriction site incompatible with the SEVA standard.

SBOL−SEVA Description. The SEVA format is highly structured in unambiguous functional segments, as shown in Figure 2B. The SEVA vector 231 was described using SBOL 2.0,[47,48] (Supplementary File S2). The previous description of

DOI: 10.1021/acssynbio.6b00029
ACS Synth. Biol. 2016, 5, 1127−1135

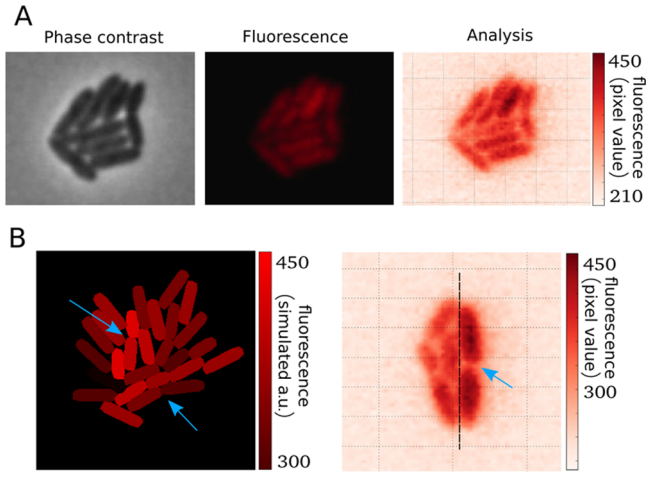


Figure 6. Spatial progress of the genetic device. (A) Phase contrast image of the population (left), fluorescent picture (middle), and computational analysis (right). In the last of these, the color scheme (right bar) represents the value of the red channel of every pixel from 0 to 255. However, it is transformed into a [0, 450] scale in order to allow comparisons with previous fluorescence measurements. (B) On the left, a simulation of a colony starting from a single cell is shown. The upper-left arrow highlights cells with a lower growth rate and RFP accumulation, while the bottom-right arrow points at a recently divided cell where the two daughters share similar RFP concentrations. On the right, expression noise inheritance is indicated with an arrow. Furthermore, RFP accumulation caused by slow growth can be observed by the black line separation; a single cell started from each side.

this vector using the GenBank format[49] was improved by adding missing features (*e.g.*, assembly scars) and by establishing structural and functional links. Two more SBOL documents were produced, one for each component of the device. Ultimately, a Java-based application was developed that can be fed the carrier plasmid and the cassettes that need to be inserted and provide the composite vector as output. The application searches in the carrier file for those restriction sites present in the cassettes (iteratively) and replaces the sequence in between. The resulting SBOL document has all of the location parameters (*i.e.*, *bioStart*) updated.

Mathematical Modeling and SBML-to-Python Conversion. In the model of Figure 3A, we show the promoter−reporter pair (18 copies, as estimated by previous observations for pBBR1 origin of replication[50]) along with the mRNA (*mRNA*) and the red fluorescent protein (*rfp*), both at zero molecules at the beginning of the simulation. In regard to the kinetic rates, k_1 is the transcription rate (27/18 h^{-1} from each plasmid), k_2 represents the translation rate (2.5 h^{-1}), k_3 is the degradation rate of the mRNA (0.65 h^{-1}), and k_4 is the protein degradation rate (0.265 h^{-1}). It should be noted that for such a small network, parameter assignment is nontrivial because of the number of constraints. The effort to assign numbers to rates[51] is thus of vital importance at this stage. The software iBioSim (http://www.async.ece.utah.edu/iBioSim/) was then used to write the model in SBML format and run the simulations with the hierarchical Runge−Kutta method for ODE solution along with the Gillespie algorithm for stochastic behavior. The model was exported in a *flat* (iBioSim option) XML file and converted into Python scripts with the tool provided (Supplementary Tool S2). Flow cytometry data were obtained from the FCS files without processing, and the simulated graph was obtained by [i] sampling a stochastic run in time (forcing equal time intervals), [ii] counting intensity values over a long enough period (∼600 h), and [iii] reinterpreting the *x*-axis values (originally, time) as individual cells in order to represent an intensity distribution comparable to the experimental plots. By making the simulations match experiments (Figure 4A), it appeared that ∼400 simulated molecules (s.m.) corresponded to ∼400 arbitrary units in the spectrophotometer (a.u.s.). As the computational measurements (s.m.) in the stochastic simulation are exactly the same, they were correlated with the fluorescent units of the cytometer (a.u.c.). As Figure 4C shows, ∼400 s.m. = ∼ 330 a.u.c.; therefore 1 a.u.s. = 400/330 a.u.c. We assumed that the sources of fluorescent signal were the same, as the cells remained unaltered.

Two-Dimensional *in Vivo* Setup. Samples for the microscope were prepared with agarose pads on a glass slide with an attached frame (1.7 cm × 2.8 cm, Life Technologies) following the method described by de Jong et al.[52] To this end, 500 μL of LB medium including 2% melted agarose was added into the middle of the frame and assembled with another glass slide. After 30 min at room temperature, one of the glass slides was carefully removed, maintaining an intact agarose pad. Then the pad was cut out to 5 mm width within the frame using a razor blade, and two strips of the pad were used to support growth of the bacterial cells. For this, strain carrying pSEVA237R-*PEM7* was precultured overnight in LB medium at 37 °C, diluted 100-fold in the same medium, and grown to exponential phase (OD_{600} = 0.2). Then 2.5 μL aliquots of the samples were spotted onto the agarose pad and assembled with cover glasses (24 mm × 50 mm) for further analysis. Wide-field fluorescence microscopy was used to observe the samples (Leica DMI6000B, Leica Microsystems) with a digital CCD camera (Orca-R2, Hamamatsu). Cell growth was monitored for 75 min under the microscope at 37 °C, and images were captured every 3 min with a 40.0×/0.75 NA dry objective or a 63.0×/1.3 NA glycerol immersion objective (depending on the experiment) with a bandpass filter for mCherry (BP 560/40 and EM 645/75) using the LAS AF version 2.6.0 software (Leica Microsystems). Images were analyzed with the MATLAB-based code Schnitzcells[53] in order to track both the positions of the cells and their length while growing.

Two-Dimensional *in Silico* Setup. DiSCUS (http://code.google.com/p/discus/) is an agent-based software for bacterial growth that uses Pymunk (http://pymunk.readthedocs.org/en/latest/), a two-dimensional physics library, to resolve collisions among cells. In the most basic test of Figure 5C.1, each cell is a body of a 16 × 30 square lattice that grows lengthwise until division, when the cell is cut in half. Pressure-based growth was simulated by counting the cells that push a body of interest (threshold at four cells) and slowing down the growth events (without stopping them). Random angle variations were introduced after division, whereby the daughter cells *copy* the angle of the mother and add a value in the interval (−25°, 25°). Furthermore, angle variations were included at the normal growth events, although to a smaller extent (maximum variation of 5°). The fact that the cells grow *in vivo* to form a circular group without holes was simulated using a slight gravity-like value that pushed the cells toward the middle of the population. This force can be eliminated when the population is about 20 cells big, at which point the circular shape is conserved without any other attraction.

DOI: 10.1021/acssynbio.6b00029
ACS Synth. Biol. 2016, 5, 1127−1135

In regard to the pixel intensity in the analysis of Figure 6A, the maximum value was set to be at the same level as the highest peak of the stochastic simulation of Figure 4B or the cytometry data of Figure 4C (~470 a.u.). Therefore, we calculated the percentage rate (470 × 100 divided by the maximum pixel value) to convert the intensity of every pixel into the scale shown by the experiments. Again, we assumed that the source of light was the same (*E. coli* CC118) and that variances were due to different machine measurements.

■ ASSOCIATED CONTENT

Supporting Information

The Supporting Information is available free of charge on the ACS Publications website at DOI: 10.1021/acssynbio.6b00029.

Sequences of the *PEM7* promoter and mCherry gene (ZIP)
SBOL files for the pSEVA231 plasmid, *PEM7* promoter, and mCherry reporter (ZIP)
Software tool to merge SBOL files and insert cassettes into a vector (ZIP)
SBML model (ZIP)
Annotated SBML file (ZIP)
Software tool to convert an SBML model into a Python script (ZIP)
Python scripts (ZIP)
Cytometry results (ZIP)

■ AUTHOR INFORMATION

Corresponding Author

*E-mail: vdlorenzo@cnb.csic.es.

Notes

The authors declare no competing financial interest.

■ ACKNOWLEDGMENTS

This work was supported by the CAMBIOS Project of the Spanish Ministry of Economy and Competitiveness; the EVOPROG, ARISYS, and EMPOWERPUTIDA Contracts from the European Union; and the PROMT Project of the Madrid Regional Government to V.d.L.

■ REFERENCES

(1) Church, G. M., Elowitz, M. B., Smolke, C. D., Voigt, C. A., and Weiss, R. (2014) Realizing the potential of Synthetic Biology. *Nat. Rev. Mol. Cell Biol. 15*, 289−294.
(2) Andrianantoandro, E., Basu, S., Karig, D. K., and Weiss, R. (2006) Synthetic biology: new engineering rules for an emerging discipline. *Mol. Syst. Biol. 2*, 2006.0028.
(3) Heinemann, M., and Panke, S. (2006) Synthetic biology - putting engineering into biology. *Bioinformatics 22*, 2790−9.
(4) Kitney, R., and Freemont, P. (2012) Synthetic biology - the state of play. *FEBS Lett. 586*, 2029−2036.
(5) Beal, J., Weiss, R., Densmore, D., Adler, A., Appleton, E., Babb, J., Bhatia, S., Davidsohn, N., Haddock, T., Loyall, J., Schantz, R., Vasilev, V., and Yaman, F. (2012) An end-to-end workflow for engineering of biological networks from high-level specifications. *ACS Synth. Biol. 1*, 317−331.
(6) Litcofsky, K. D., Afeyan, R. B., Krom, R. J., Khalil, A. S., and Collins, J. J. (2012) Iterative plug-and-play methodology for constructing and modifying synthetic gene networks. *Nat. Methods 9*, 1077−1080.
(7) Brophy, J. A., and Voigt, C. A. (2014) Principles of genetic circuit design. *Nat. Methods 11*, 508−520.
(8) Densmore, D. M., and Bhatia, S. (2014) Bio-design automation: software + biology + robots. *Trends Biotechnol. 32*, 111−113.
(9) Baker, D., Church, G., Collins, J., Endy, D., Jacobson, J., Keasling, J., Modrich, P., Smolke, C., and Weiss, R. (2006) Engineering life: building a fab for biology. *Sci. Am. 294*, 44−51.
(10) Varadarajan, P. A., and Del Vecchio, D. (2009) Design and characterization of a three-terminal transcriptional device through polymerase per second. *NanoBioscience, IEEE Trans. 8*, 281−289.
(11) Nielsen, A. A., Segall-Shapiro, T. H., and Voigt, C. A. (2013) Advances in genetic circuit design: novel biochemistries, deep part mining, and precision gene expression. *Curr. Opin. Chem. Biol. 17*, 878−892.
(12) Khalil, A. S., Lu, T. K., Bashor, C. J., Ramirez, C. L., Pyenson, N. C., Joung, J. K., and Collins, J. J. (2012) A synthetic biology framework for programming eukaryotic transcription functions. *Cell 150*, 647−658.
(13) Villalobos, A., Ness, J. E., Gustafsson, C., Minshull, J., and Govindarajan, S. (2006) Gene Designer: a synthetic biology tool for constructing artificial DNA segments. *BMC Bioinf. 7*, 285.
(14) Canton, B., Labno, A., and Endy, D. (2008) Refinement and standardization of synthetic biological parts and devices. *Nat. Biotechnol. 26*, 787−793.
(15) MacDonald, J. T., Barnes, C., Kitney, R. I., Freemont, P. S., and Stan, G.-B. V. (2011) Computational design approaches and tools for synthetic biology. *Integr. Biol. 3*, 97−108.
(16) Marchisio, M. A., and Stelling, J. (2009) Computational design tools for synthetic biology. *Curr. Opin. Biotechnol. 20*, 479−485.
(17) Huynh, L., Tsoukalas, A., Köppe, M., and Tagkopoulos, I. (2013) SBROME: a scalable optimization and module matching framework for automated biosystems design. *ACS Synth. Biol. 2*, 263−273.
(18) Huynh, L., and Tagkopoulos, I. (2014) Optimal part and module selection for synthetic gene circuit design automation. *ACS Synth. Biol. 3*, 556−564.
(19) Chandran, D., Bergmann, F. T., and Sauro, H. M. (2009) TinkerCell: modular CAD tool for synthetic biology. *J. Biol. Eng. 3*, 19.
(20) Hill, A. D., Tomshine, J. R., Weeding, E. M., Sotiropoulos, V., and Kaznessis, Y. N. (2008) SynBioSS: the synthetic biology modeling suite. *Bioinformatics 24*, 2551−2553.
(21) 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* (6281), aac7341.
(22) Cao, H., Romero-Campero, F. J., Heeb, S., Cámara, M., and Krasnogor, N. (2010) Evolving cell models for systems and synthetic biology. *Syst. Synth. Biol. 4*, 55−84.
(23) Knight, T. (2003) *Idempotent vector design for standard assembly of biobricks*, DTIC document.
(24) Shetty, R. P., Endy, D., and Knight, T. F., Jr. (2008) Engineering BioBrick vectors from BioBrick parts. *J. Biol. Eng. 2*, 5.
(25) Preston, A. (2003) Choosing a cloning vector, in *E. coli Plasmid Vectors* (Casali, N., and Preston, A., Eds.), pp 19−26, Humana Press, Totowa, NJ.
(26) Martínez-García, E., Aparicio, T., Goñi-Moreno, A., Fraile, S., and de Lorenzo, V. (2015) SEVA 2.0: an update of the Standard European Vector Architecture for de-/re-construction of bacterial functionalities. *Nucleic Acids Res. 43*, D1183−D1189.
(27) Calero, P., Jensen, S. I., and Nielsen, A. T. (2016) Broad host range ProUSER vectors enable fast characterization of inducible promoters and optimization of *p*-coumaric acid production in *Pseudomonas putida* KT2440. *ACS Synth. Biol. 5*, 741−753.
(28) Wright, O., Delmans, M., Stan, G.-B., and Ellis, T. (2014) GeneGuard: a modular plasmid system designed for biosafety. *ACS Synth. Biol. 4* (3), 307−316.
(29) Kim, S. H., Cavaleiro, A. M., Rennig, M., and Nørholm, M. H. H. (2016) SEVA linkers: a versatile and automatable DNA backbone exchange standard for synthetic biology. *ACS Synth. Biol.*, DOI: 10.1021/acssynbio.5b00257.
(30) Zobel, S., Benedetti, I., Eisenbach, L., de Lorenzo, V., Wierckx, N., and Blank, L. M. (2015) Tn7-Based Device for Calibrated

Heterologous Gene Expression in *Pseudomonas putida*. *ACS Synth. Biol. 4* (12), 1341−1351.
(31) Galdzicki, M., Clancy, K. P., Oberortner, E., Pocock, M., Quinn, J. Y., Rodriguez, C. A., Roehner, N., Wilson, M. L., Adam, L., Anderson, J. C., Bartley, B. A., Beal, J., Chandran, D., Chen, J., Densmore, D., Endy, D., Grünberg, R., Hallinan, J., Hillson, N. J., Johnson, J. D., Kuchinski, A., Lux, M., Misirli, G., Peccoud, J., Plahar, H. A., Sirin, E., Stan, G.-B., Villalobos, A., Wipat, A., Gennari, J. H., Myers, C. J., and Sauro, H. M. (2014) The Synthetic Biology Open Language (SBOL) provides a community standard for communicating designs in synthetic biology. *Nat. Biotechnol. 32*, 545−550.
(32) Kelwick, R., MacDonald, J. T., Webb, A. J., and Freemont, P. (2014) Developments in the tools and methodologies of synthetic biology. *Front. Bioeng. Biotechnol. 2*, 60.
(33) Kelly, J. R., Rubin, A. J., Davis, J. H., Ajo-Franklin, C. M., Cumbers, J., Czar, M. J., de Mora, K., Glieberman, A. L., Monie, D. D., and Endy, D. (2009) Measuring the activity of BioBrick promoters using an in vivo reference standard. *J. Biol. Eng. 3*, 4.
(34) Myers, C. J., Barker, N., Jones, K., Kuwahara, H., Madsen, C., and Nguyen, N.-P. D. (2009) iBioSim: a tool for the analysis and design of genetic circuits. *Bioinformatics 25*, 2848−2849.
(35) Hucka, M., Finney, A., Sauro, H. M., Bolouri, H., Doyle, J. C., Kitano, H., Arkin, A. P., Bornstein, B. J., Bray, D., Cornish-Bowden, A., Cuellar, A. A., Dronov, S., Gilles, E. D., Ginkel, M., Gor, V., Goryanin, I. I., Hedley, W. J., Hodgman, T. C., Hofmeyr, J.-H., Hunter, P. J., Juty, N. S., Kasberger, J. L., Kremling, A., Kummer, U., Le Novère, N., Loew, L. M., Lucio, D., Mendes, P., Minch, E., Mjolsness, E. D., Nakayama, Y., Nelson, M. R., Nielsen, P. F., Sakurada, T., Schaff, J. C., Shapiro, B. E., Shimizu, T. S., Spence, H. D., Stelling, J., Takahashi, K., Tomita, M., Wagner, J., and Wang, J. (2003) The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models. *Bioinformatics 19*, 524−531.
(36) Roehner, N., and Myers, C. J. (2014) A methodology to annotate systems biology markup language models with the synthetic biology open language. *ACS Synth. Biol. 3*, 57−66.
(37) Bornstein, B. J., Keating, S. M., Jouraku, A., and Hucka, M. (2008) LibSBML: an API library for SBML. *Bioinformatics 24*, 880−881.
(38) Gillespie, D. T. (1976) A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. *J. Comput. Phys. 22*, 403−434.
(39) Danchin, A. (2012) Scaling up synthetic biology: do not forget the chassis. *FEBS Lett. 586*, 2129−2137.
(40) Manoil, C., and Beckwith, J. (1985) TnphoA: a transposon probe for protein export signals. *Proc. Natl. Acad. Sci. U. S. A. 82* (23), 8129−8133.
(41) Amos, M. (2014) Population-based microbial computing: a third wave of synthetic biology? *Int. J. Gen. Syst. 43*, 770−782.
(42) Macía, J., Posas, F., and Solé, R. V. (2012) Distributed computation: the new wave of synthetic biology devices. *Trends Biotechnol. 30*, 342−9.
(43) Tamsir, A., Tabor, J. J., and Voigt, C. A. (2011) Robust multicellular computing using genetically encoded NOR gates and chemical 'wires'. *Nature 469*, 212−215.
(44) Goñi-Moreno, A., Amos, M., and de la Cruz, F. (2013) Multicellular computing using conjugation for wiring. *PLoS One 8*, e65986.
(45) Goni-Moreno, A., and Amos, M. (2015) DiSCUS: A simulation platform for conjugation computing, presented at Unconventional and Natural Computation (UCNC 2015), Auckland, New Zealand, Aug 31−Sept 4, 2015.
(46) Skinner, S. O., Sepúlveda, L. A., Xu, H., and Golding, I. (2013) Measuring mRNA copy number in individual *Escherichia coli* cells using single-molecule fluorescent in situ hybridization. *Nat. Protoc. 8*, 1100−1113.
(47) Bartley, B., Beal, J., Clancy, K., Misirli, G., Roehner, N., Oberortner, E., Pocock, M., Bissell, M., Madsen, C., Nguyen, T., Zhang, Z., Gennari, J. H., Myers, C., Wipat, A., and Sauro, H. (2015) Synthetic Biology Open Language (SBOL) Version 2.0.0. *J. Integr. Bioinf. 12*, 272.
(48) Roehner, N., Beal, J., Clancy, K., Bartley, B., Misirli, G., Grünberg, R., Oberortner, E., Pocock, M., Bissell, M., Madsen, C., Nguyen, T., Zhang, M., Zhang, Z., Zundel, Z., Densmore, D., Gennari, J. H., Wipat, A., Sauro, H. M., and Myers, C. J. (2016) Sharing Structure and Function in Biological Design with SBOL 2.0. *ACS Synth. Biol. 5*, 498−506.
(49) Benson, D. A., Karsch-Mizrachi, I., Lipman, D. J., Ostell, J., Rapp, B. A., and Wheeler, D. L. (2000) GenBank. *Nucleic Acids Res. 28*, 15−18.
(50) Lee, T. S., Krupa, R. A., Zhang, F., Hajimorad, M., Holtz, W. J., Prasad, N., Lee, S. K., and Keasling, J. D. (2011) BglBrick vectors and datasheets: a synthetic biology platform for gene expression. *J. Biol. Eng. 5*, 12.
(51) Ronen, M., Rosenberg, R., Shraiman, B. I., and Alon, U. (2002) Assigning numbers to the arrows: parameterizing a gene regulation network by using accurate expression kinetics. *Proc. Natl. Acad. Sci. U. S. A. 99*, 10555−10560.
(52) de Jong, I. G., Beilharz, K., Kuipers, O. P., and Veening, J.-W. (2011) Live cell imaging of *Bacillus subtilis* and *Streptococcus pneumoniae* using automated time-lapse microscopy. *J. Visualized Exp. 53*, 3145.
(53) Young, J. W., Locke, J. C., Altinok, A., Rosenfeld, N., Bacarian, T., Swain, P. S., Mjolsness, E., and Elowitz, M. B. (2012) Measuring single-cell gene expression dynamics in bacteria using fluorescence time-lapse microscopy. *Nat. Protoc. 7*, 80−88.

DOI: 10.1021/acssynbio.6b00029
ACS Synth. Biol. 2016, 5, 1127−1135