Background

In January of 2003 we taught a month long intensive course on the design and specification of synthetic genetic systems. 16 students worked in four teams to design genetically encoded oscillators. The student's designs were inspired by Michael Elowitz's original repressilator [Elowitz & Leibler, 2000]. During the 2003 course students helped to invent and implement ideas about 'standard biological parts.'

In January of 2004 we taught a second course that challenged students to design and specify the DNA sequence encoding programmed patterns in clonal populations of bacteria. The student's designs made use of the past lessons in the area of pattern formation and the research ideas being developed in Amorphous Computing [http://www.swiss.ai.mit.edu/projects/amorphous/]. During the 2004 course students made use of a nascent abstraction hierarchy that supports the engineering of many-component integrated genetic systems [below].

Web pages describing the 2003 and 2004 projects are online [http://rosalind.csail.mit.edu/projects/index.cgi].

From June through November of 2004 students at five schools (Boston University, Caltech, MIT, Princeton University, and The University of Texas at Austin) worked to design and build genetically encoded finite state machines [Figure 1]. The student's work was sponsored by a one-time grant from NSF in support of a 'Synthetic Biology Competition.' The Texas team's project is described on the next page:

Figure 1. Participants of the 2004 Synthetic Biology Competition Jamboree, held November 6-7 2004 at the AAAS Norton's Woods facility in Cambridge, MA. Undergraduate and graduate students from Boston University (with Harvard Medical), Caltech, MIT, Princeton (with University Maryland, Baltimore County and Duke), and UT Austin (with UCSF) spent five months working together in teams to design and build genetically encoded finite state machines. Photo: Rettberg.

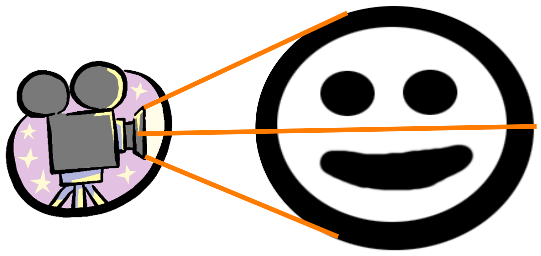

Figure 2a. The 2004 UT Austin team chose to design and build a biofilm that could perform distributed edge detection on a light-encoded image. In theory, each cell in a lawn of bacteria would update its state in response to light input and, depending on the states of neighboring cells, decide whether or not to change color.

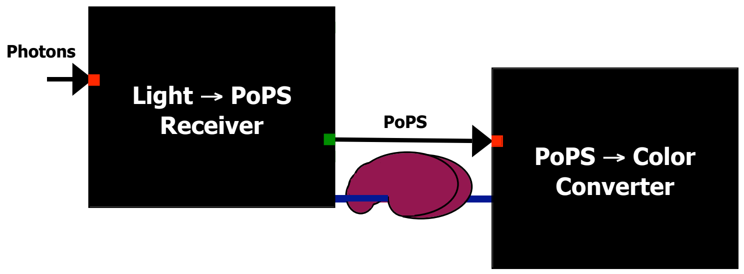

Figure 2b. An abstract, device-level depiction for part of the Texas system. A light-detection device (left) is connected to a colorimetric reporting device (right). Photons produce PoPS, a common carrier for gene expression [below], and PoPS produce color.

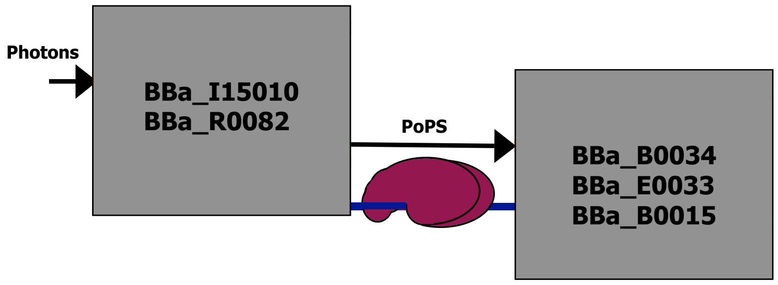

Figure 2c. The [light→PoPS] device was built using a new light-receiver part, BBa_I15010, and BBa_R0082, a pre-existing BioBrick. BBa_I15010, the gift of Anselm Levskaya (Voigt Lab, UCSF), is an engineered two component signaling protein based on *cph1* from *Synechocystis*. The [PoPS→color] device was built using three existing BioBricks.

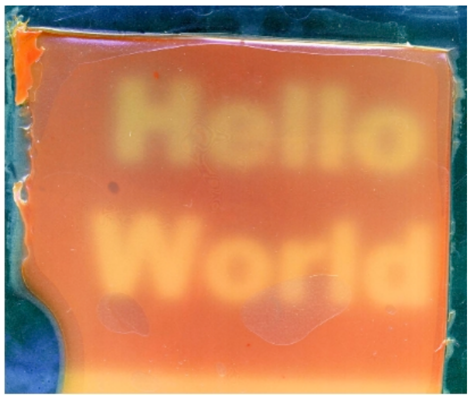

Figure 2d. Students at UT Austin used the image 'Hello World' to illuminate a ~10cm square lawn of bacteria containing their initial system (the first picture encoded on their biofilm film is shown here). A lawn of *E. coli* should be able to capture images at about gigapixel per square inch resolution. Photo: Jeff Tabor.

The following pages provide background material and perspectives relevant to the iGEM competition.

Research Context

If you are a scientist: One direct approach for testing models of biological systems is to design and construct physical instances of the systems in accordance with our models. Differences between expected and observed behavior highlight either relevant science that needs doing, or failures in our modeling and simulation technology that need fixing. If science needs labels, such work might be called 'reverse systems biology.'

If you are an engineer: Biology is a technology for processing chemicals, energy, information, and materials. However, our ability to engineer biology is now limited. Foundational technologies are needed to help enable the systematic engineering of biology. Past lessons from other engineering disciplines whose relevance to biological engineering is worth exploring include: (i) standardization of components and conditions, (ii) component and device abstraction, and (iii) decoupling of system design from system fabrication.

Educational Context

In teaching the 2003 and 2004 courses and competition, we have learned how scientists might begin to apply engineering principles to hide the otherwise overwhelming details and intricacy of function found in natural biological systems. We have also observed that engineers best learn about biology by pursuing their natural inclination to design and build. For example, if you want to motivate an engineering undergraduate to learn how proteins bind DNA to regulate gene expression, ask her to design a set of genetically encoded inverters!

Technology Context

The construction of student-designed biological systems is being driven, in part, by improvements in DNA synthesis, a technology that enables the information specifying biological function to be converted to physical genetic material. Bulk DNA synthesis capacity appears to have doubled every ~18 months for the last ten years; the commercial price of synthesis of long fragments of DNA (>500 base pairs) has decreased by a factor of ~2 each of the last three years [Carlson, 2003; Google sponsored links for "dna synthesis"]. More recently, downstream methods have been developed that allow for the rapid assembly of commercial DNA synthesis products into still-longer molecules [e.g., Stemmer et al., 1995; Smith et al. 2003; Kodumal et al. 2004; Tian et al., 2004]. DNA synthesis helps to enable the *decoupling* of system design from system fabrication.

We have also begun to make progress on *standardization* of components via a pilot Registry of Standard Biological Parts [http://parts.mit.edu/], and on *abstraction* via a provisional hierarchy that supports the engineering of many-component integrated genetic systems [next page]. Much more work is needed.

Abstraction

The purpose of an abstraction hierarchy is to hide information and manage complexity [Figure 3]. To be useful, individuals must be able to work independently at each level of the hierarchy. In biology, for example, parts-level researchers might need to know what sorts of parts that device-level researchers would like to use, how different types of parts actually work (e.g., atomic interactions between an amino acid and the major groove of DNA), and how to order a piece of DNA. But, parts-level researchers do *not* need to know anything about phosphoramidite chemistry, how short oligonucleotides are assembled into longer contiguous DNA fragments, or how a ring oscillator works, et cetera.

An abstraction hierarchy can be used with other technologies (e.g., 'standard part families') to quickly design and specify the DNA sequence encoding many integrated genetic systems. For example, a ring oscillator *system* can be built from three inverter *devices*. Each inverter *device* can in turn be built from four *parts*, and *each* part can be encoded by a pre-specified sequence of *DNA*. Other systems can be quickly specified as different combinations of devices, and so on.

One detail of the abstraction hierarchy depicted in Figure 3 that's worth elaborating is the definition of the input and output signal carrier for gene-expression based devices. A device is a combination of parts that performs some useful function. For example, one

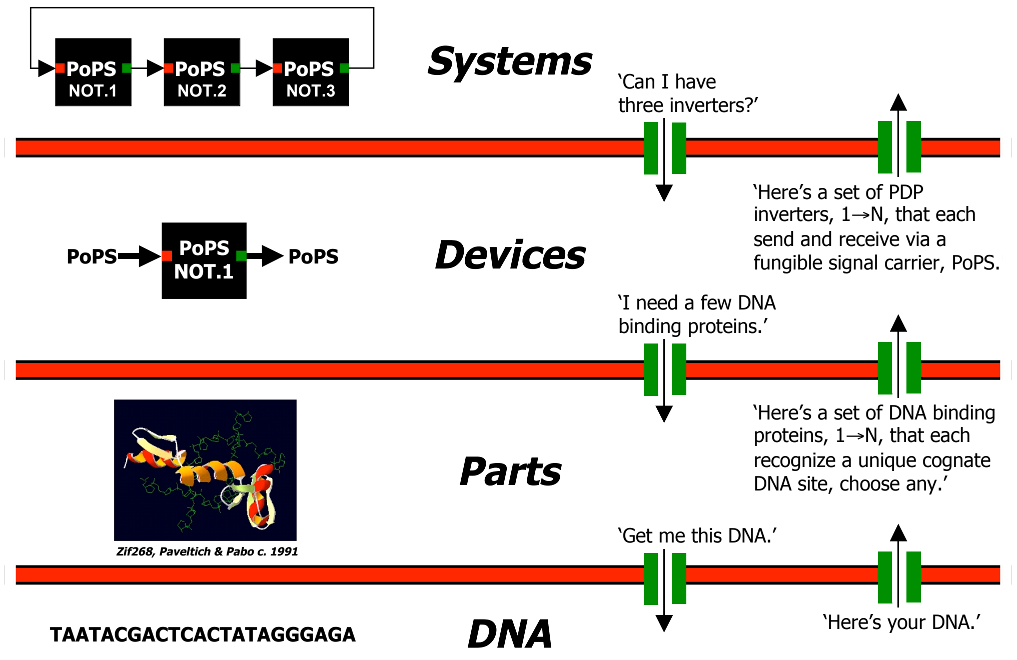
PoPS
NOT.1
PoPS
NOT.2
PoPS
NOT.3
Systems
'Can I have three inverters?'
'Here's a set of PDP inverters, 1→N, that each send and receive via a fungible signal carrier, PoPS.'
PoPS
PoPS
NOT.1
PoPS
Devices
'I need a few DNA binding proteins.'
'Here's a set of DNA binding proteins, 1→N, that each recognize a unique cognate DNA site, choose any.'
Parts
Zif268, Paveltich & Pabo c. 1991
'Get me this DNA.'
'Here's your DNA.'
TAATACGACTCACTATAGGGAGA
DNA

Figure 3. An abstraction hierarchy that supports the modeling, design, and construction of integrated genetic systems. Abstraction levels are listed (DNA, Parts, Devices, Systems). Abstraction barriers (red) block all exchange of information between levels. Interfaces (green) enable the limited and principled exchange of information between levels. Example exchanges are given in quotes.

type of device is an inverter. An inverter takes an input signal and produces the opposite output signal (e.g., HIGH input produces LOW output and vice versa; an inverter can function like a Boolean NOT). A genetic inverter can be assembled from four parts [Figure 4]. However, the signal carrier for an inverter depends on how we organize (i.e., 'black box') the parts that make up the device. For example, a classical model for an inverter receives as input the concentration of repressor A and, via gene expression, sends as output the concentration of repressor B [Figure 4, left]. In this model, the signal carriers are the proteins A and B, and the signal levels are the concentrations of proteins A and B. One immediate problem with such a device is that the input and output signal carriers are different. Such an inverter can only be connected to an upstream device that sends protein A and a downstream device that receives protein B. A collection of devices that each use device-specific molecules as signal carriers cannot be used in combination to make many systems.

We can solve the signal carrier problem by reorganizing the parts that make up our inverter [Figure 4, right]. Now, the input signal to the inverter is carried by polymerase per second, or PoPS. PoPS is the flow of RNA polymerase molecules along DNA (i.e., 'current' for gene expression). The PoPS level is set by the amount of RNA polymerase molecules that trundle past a specific position on DNA each second. Importantly, the output signal from the inverter is also carried by PoPS. The inversion function of the device is still executed by a repressor protein acting on its cognate site at the operator DNA, but all details specific to this interaction are internal to the device. As a result, PoPS-based devices can be used in combination.

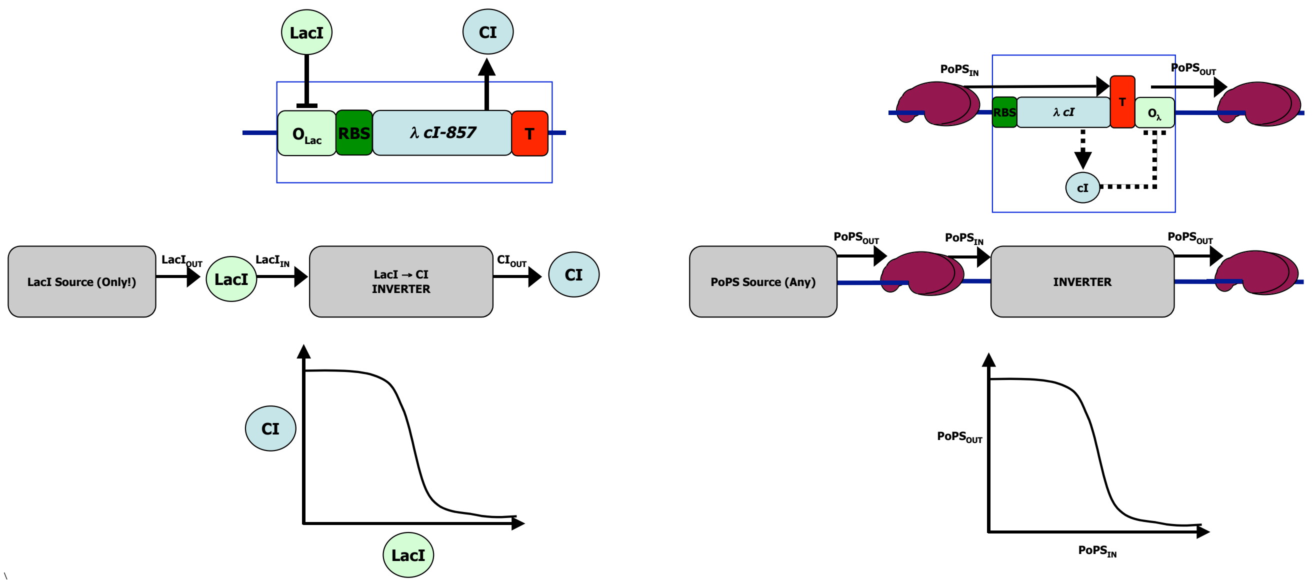

Figure 4. Two definitions for genetically encoded inverter devices. Left – A 'classical' genetically encoded inverter that takes as input the concentration of a repressor protein, LacI. In the presence of LacI, expression of the downstream cI gene is inhibited. In the absence of the LacI, expression of the downstream cI occurs via transcription initiating at O_{Lac}, producing a high CI output signal. Right – A PoPS-based inverter (see text above). When the input PoPS-level is high, CI is produced. CI acts at O_{lambda} to keep the output PoPS-level low. The molecule-specific details of a PoPS-based inverter are internal to the device and can be hidden; PoPS-based devices can thus be used in combination with (i.e., connected to) any other PoPS-based devices.

References

Carlson R. 2003. The pace and proliferation of biological technologies. *Biosecur Bioterror*. **1**:203.

Elowitz MB, Leibler S. A synthetic oscillatory network of transcriptional regulators. *Nature*. 2000 Jan 20;**403**(6767):335-8.

Kodumal SJ, Patel KG, Reid R, Menzella HG, Welch M, Santi DV. 2004. Total synthesis of long DNA sequences: Synthesis of a contiguous 32-kb polyketide synthase gene cluster. PNAS USA 101:15573.

Smith HO, Hutchison CA 3rd, Pfannkoch C, Venter JC. Generating a synthetic genome by whole genome assembly: phiX174 bacteriophage from synthetic oligonucleotides. *Proc Natl Acad Sci U S A*. 2003 Dec 23;**100**(26):15440-5.

Stemmer WP, Crameri A, Ha KD, Brennan TM, Heyneker HL. Single-step assembly of a gene and entire plasmid from large numbers of oligodeoxyribonucleotides. *Gene*. 1995 Oct 16;**164**(1):49-53.

Tian J, Gong H, Sheng N, Zhou X, Gulari E, Gao X, Church G. Accurate multiplex gene synthesis from programmable DNA microchips. *Nature*. 2004 Dec 23; **432**(7020): 1050-4.