BioSystems 109 (2012) 57–71

Contents lists available at SciVerse ScienceDirect

BioSystems

journal homepage: www.elsevier.com/locate/biosystems

Computational simulation of a gene regulatory network implementing an extendable synchronous single-input delay flip-flop

Imad Hoteit[a], Nawwaf Kharma[a,*], Luc Varin[b]

[a] ECE Department, Concordia University, Montreal, QC, Canada
[b] Biology Department, Concordia University, Montreal, QC, Canada

ARTICLE INFO

Article history:
Received 17 October 2011
Accepted 5 January 2012

Keywords:
Escherichia coli
Logic gates
Synthetic Biology
Delay flip flop
Finite state machine
Cellular automata

ABSTRACT

We present a detailed and extendable design of the first *synchronous single-input* delay flip-flop implemented as a gene regulatory network in *Escherichia coli* (*E. coli*). The device, which we call the *BioD*, has one data input (transacting RNA), one clock input (far-red light) and an output that reports the state of the device using green fluorescent protein (GFP). The proposed design builds on Gardner's toggle switch, to provide a more sophisticated device that can be synchronized with other devices within the same cell, and which requires only one data input. We provide a mathematical model of the system and simulation results. The results show that the device behaves in line with desired functionality. Further, we discuss the constraints of the design, which pertain to ranges of parameter values. The *BioD* is extended *via* the addition of an update function and input and output interfaces. The result is the *BioFSM*, which constitutes a synchronous and modular finite state machine, which uses an update function to change its state, stored in the *BioD*. The *BioFSM* uses its input and output interfaces for inter-cellular communications. This opens the door to the design of a circular cellular automata (the *BioCell*), which is envisioned as a number of communicating *E. coli* colonies, each made of clones of one *BioFSM*.

1. Introduction

Most of the complex processes that take place in a cell are governed by gene expression, which is regulated at several levels along the pathway leading from DNA to protein. Gene expression may be regulated during transcription and post-transcriptionally, including during protein translation and *via* post-translational modification of proteins. Notably, much of the control of gene expression is done either by regulatory proteins or by RNAs, which is themselves the products of genes. Hence, the interactions between DNA, RNAs, proteins, and other molecules, form natural gene regulatory networks (or GRNs) of varied complexity.

While studying these networks and their components provides invaluable information, it is essential to: (a) thoroughly investigate these components in different environments, while performing different functions, and (b) integrate this knowledge to build new *synthetic* gene regulatory networks and other devices. The discipline of Synthetic Biology aims at systematically designing, building, combining and testing new biological functions and systems that do not occur in nature. Indeed, individual parts such as promoters and protein coding sequences can be assembled into GRNs that perform desired functionalities, such as computing machines.

The synthesis of computing machines *via* the manipulation of DNA within or without living organisms, started in 1994 when Adleman executed an experimental procedure that used DNA, *in vitro*, to solve an instance of the directed Hamiltonian path problem (Adleman, 1994). In contrast, *in vivo* cell-based or cellular computing started in 1998 with the modification of the genome of the prokaryote *Escherichia coli*, to realize 1- and 2-input *combinatorial* Boolean logic gates (e.g. NOT, AND and IMPLIES) (Knight and Sussman, 1998; Weiss et al., 1998); a similar feat was achieved with eukaryotic cells by Kramer et al. (2004). Along another dimension, time-dependant or sequential Boolean logic devices have also been implemented in living cells, starting with a 2-input toggle switch by Gardner et al. (2000), and a synthetic oscillator by Elowitz and Leibler (2000). In fact, in one decade this field has grown to generate many elementary devices (Drubin et al., 2007; Boyle and Silver, 2009; Tigges et al., 2009; Haynes and Silver, 2009), including band-pass filters (Basu et al., 2005) and counters (Friedland et al., 2009). More complicated devices such as engineered multi-cellular pattern generators (Basu et al., 2004, 2005), single cell biosensors (Levskaya et al., 2005; Tecon et al., 2006), tumor-targeting bacteria (Anderson et al., 2006), cell-based computers (Cox et al., 2007;

Abbreviations: GRN, gene regulatory network; FR, far-red; RNA, ribonucleic acid; DNA, deoxyribonucleic acid; RNAP, RNA polymerase.

* Corresponding author at: Address: 1455 de Maisonneuve Blvd. West, Montreal, QC, Canada H3G 1M8. Tel.: +1 514 848 2424x3117; fax: +1 514 848 2802.
E-mail addresses: i_hoteit@ece.concordia.ca (I. Hoteit), kharma@ece.concordia.ca (N. Kharma), varinl@alcor.concordia.ca (L. Varin).

0303-2647/$ – see front matter © 2012 Elsevier Ireland Ltd. All rights reserved.
doi:10.1016/j.biosystems.2012.01.004

Balagadde et al., 2008), and biological memory devices (Chang et al., 2010) have also been synthesized or proposed.

In the particular case of switching devices, there has been a fair number of switches built or theorized, which involve (a) DNA modification (e.g. using invertases), (b) regulation of the process of transcription, (c) post-transcriptional regulation (involving various RNA molecules), as well as (d) post-translational regulation (by changing the state of expressed proteins).

The first example of the use of invertases is Ham et al. (2006), which places the promoter of a gene between two specific elements targeted by the `FimE` flipase. The flipase inverts the inversion region between these two elements (including them). This completely disables transcription from that promoter, rendering the associated gene silent. This is a unidirectional operation and it does not require qualification by a clock. In 2008, Ham et al. (2008) expanded their initial concept by using both the `hin` and `fimE` inversion mechanisms. This allowed them to use the relative positions of the elements marking the inversion regions to propose three- and five-state machines, which rely completely on the two flipases to change state. It is worth noting that this method of defining state is heritable as changes to the DNA are permanent and hence, inherited by the offspring.

The most prominent example of a toggle switch that is transcriptionally controlled is that of Gardner et al. (2000). However, this toggle switch requires two inputs and operates asynchronously (is not controlled by a clock input). Elowitz and Leibler (2000) synthesized a three gene oscillator (plus an additional gene for reporting), dubbed *repressilator*. The product of each of the three genes represses the next gene in a loop, with the last gene repressing the first one. The repressilator is not a bi-stable switch but rather a self-maintaining oscillator that proceeds from one state to the next, autonomously and without the need for any clock input. Kobayashi et al. (2004) utilized slightly modified versions of Gardner's toggle switch as memory modules of larger networks that sensed specific events (e.g. DNA damage) and generated particular responses (e.g. biofilm formation). In this case, the toggle switch is, *by default*, in one specific state, which flips in response to the sensed event. It does not have two inputs, but it does not have two stable states either. And, as is the case with Gardner's switch, it operates asynchronously. Stricker et al. (2008) synthesized a two gene oscillating network, where one gene is responsible for the activation of both genes, and the other gene is responsible for repressing both genes. This network improves on the repressilator in terms of speed, durability of the oscillation and the ability to externally tune its oscillations. Nevertheless, this network is not a switch that can be used as a memory module, such as Gardner's toggle. Lou et al. (2010) propose a single-input toggle switch, made of a Gardner-like two-gene memory module and a single-gene NOR gate module. The memory module is, by default, in a particular stable state. Upon the introduction of a UV input, several proteins degrade, which causes the memory module, with help from the NOR module to switch to a new state and maintain it. This is, in fact, a single-input switch, but it lacks a clock input.

One very significant work of RNA-based switching behavior is that of Bayer and Smolke (2005). They present devices that are regulated post-transcriptionally using RNA *riboswitches*. A riboswitch is an RNA molecule containing two domains: (i) a ligand-binding aptamer domain and (ii) an antisense regulator domain. The latter is used to block the ribosome binding site (RBS) and prevent translation, while the former binds a ligand that triggers a conformational change in the riboswitch, resulting in either the covering or uncovering of the anti-sense regulator domain. Riboswitches have the advantage that they can be designed and/or evolved to respond to many ligands including proteins and RNA molecules. Riboswitches have been synthesized to respond to one or more inputs (ligands). Although current riboswitches change state uni-directionally, it is possible to imagine riboswitches that respond to inducible small protein ligands. So far, riboswitches act asynchronously.

Finally, a good example of how switches can be regulated at the protein level is the work of Dueber et al. (2003), which modified the natural N-WASP allosteric switch to synthesize 1- and 2-input synthetic protein switches. In the 2-input switch, the hybrid protein was engineered to have two A-terminal auto-inhibitory domains that correspond to the output domain and a C-terminal domain on the protein. The way in which the protein responded to the two input ligands (PDZ and Cdc42) relied on the relative positioning of the four domains. They used this to synthesize various switches, whose state (active or not) depended on combinatorial functions of the two inputs. All of their devices are asynchronous and unidirectional.

Despite the many works on genetic switches (also called flip-flops), all published synthesized and proposed designs work *asynchronously*, usually utilizing *more* than one external logical input. A notable exception is (Lou et al., 2010) which is a single-input switch, albeit still asynchronous. Lack of synchronization-ability entails that the operation of a flip-flop cannot be synchronized with the operation of other parts of a larger system, using a single global clock. Also, a true delay flip-flop has but one logical input. Though the use of a single input complicates design, it does simplify use and allow for easier expansion of function. We call the proposed GRN embodying a synchronous single-input delay flip-flop the *BioD*. It is, in summary, a novel GRN that changes states in response to single logical input, and only on the rising edge of a clock signal. Its specification and detailed design, modeling and simulation results follow.

In parallel to advances in GRN design, mathematical modeling and simulation tools have been developed to help make approximate predictions of the behavior of GRNs before significant resources are allotted to their synthesis. These include, but are not limited to, deterministic (Hindmarsh et al., 2005) and stochastic simulation algorithms (Gillespie, 1977), metabolic control analysis (MCA) (Olivier et al., 2005), structural analysis (Olivier et al., 2005) and flux-balance analysis (FBA) (Orth et al., 2010). Deterministic simulation models include differential equations, Boolean networks, logical networks and rule-based formalisms (de Jong, 2002). Stochastic models include P systems (Romero-Campero et al., 2009), Bayesian networks and master equations (de Jong, 2002). An interesting comparison was offered by Twycross et al. (2010) of the benefits of each of the deterministic and stochastic models and presented as a case study using an auxin-transport example as a common base of comparison. MCA quantifies how variables, such as fluxes and species concentrations, depend on network parameters. Structural analysis is mostly used for genome-scale models to determine reduced stoichiometric matrices. FBA is used for optimizing the growth rate of a modeled organism, while falling within the constraints of its internal metabolites.

There exists a long list of software packages and libraries capable of implementing one or more of the above mentioned simulation methods. A very important clustering of these tools can be found under the SBML.org umbrella. The Systems Biology Markup Language (SBML) was developed by a small team of researchers who identified the need to enable interoperability between the vast arrays of simulation software that became available (Hucka et al., 2003). Although we wrote our own software to simulate our networks, there exist hundreds of very powerful software packages in the SBML repository. Suffice it to say, the scope of this paper does not cover the plethora of tools out there, so we instead highlight a good qualitative modeling tool like the Genetic Network Analyzer (GNA) (Batt et al., 2012), a more complete and quantitative collection of tools like the Systems Biology Workbench (SBW) (Hucka et al., 2002), and a good multi-cell simulation tool, the Infobiotics Workbench (Blakes et al., 2011).