14
Development of Bacteria-Based Cellular Computing Circuits for Sensing and Control in Biological Systems

Michaela A. TerAvest, Zhongjian Li, and Largus T. Angenent

14.1
Introduction

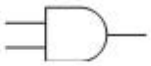
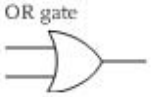
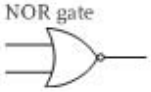
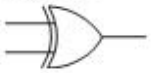
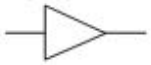
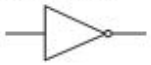
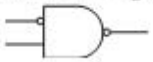
The term "*biocomputing*" encompasses many different types of unconventional computing, including DNA, RNA, and enzyme computing and others. One of the newest subareas of biocomputing is cellular computing, or the use of whole, live cells for computation. Cellular computing is currently in development in both eukaryotic and bacterial systems. Here we will discuss the current state and potential of cellular computing in bacterial systems. We will focus on the current implementations of bacterial cellular computing, which we will refer to as "*cellular computing circuits*" (i.e., bacterial strains engineered to perform basic computing functions, such as logic operations). Cellular computing circuits have been created to perform several basic computing functions through the use of engineered genetic circuits (i.e., the engineered gene sequence that confers computational functionality on the cell).

While cellular computing circuits can be noisier and more difficult to control than enzymatic or DNA-based computing systems, they also have several advantages. Whole, live cells are more resilient to environmental damage than pure enzymes or DNA because of their ability to maintain intracellular conditions and repair themselves. Cellular computing circuits also have the advantage that they reproduce on their own, and a bacterial culture is much less expensive to grow than enzyme purification or DNA synthesis. Another benefit of cellular computing circuits is that they are capable of self-optimizing if given the proper selective pressure – something that DNA and enzymes cannot do because they do not reproduce or mutate. This has been demonstrated by directed evolution of cellular computing circuits, which can turn a nonfunctional genetic circuit into a functional one using bacterial mutation and reproduction.

Directed evolution has been used successfully to optimize a genetic circuit in a study by Yokobayashi *et al.* [1]. Previously, Weiss [2] constructed a nonfunctional A IMPLY B gate (Table 14.1) using a modified lac repression system in *Escherichia coli*. In the initial study, the cellular computing was optimized by rational design using computer models to predict favorable mutations and testing each one in the laboratory. Yokobayashi *et al.* [1] used a simpler approach to functionalize

Biomolecular Information Processing: From Logic Systems to Smart Sensors and Actuators,
First Edition. Edited by Evgeny Katz.

Table 14.1 Cellular computing circuit logic gate operations.

Component	Function	References
AND gate 	True when A and B are true	[3–7]
NAND gate 	True except when A and B are true	[4]
OR gate 	True when A, B, or both are true	[4]
NOR gate 	True when neither A nor B is true	[4]
XOR gate 	True when A or B but not both are true	[4]
A gate 	True when A is true (filter)	[4]
NOT A gate 	True when A is not true (inverter)	[4]
A IMPLY B gate 	True when A is true and B is false	[4]
A NIMPLY B gate 	True except when A is true and B is false	[4]
TRUE gate 	Always true	[4]
EQUAL gate 	True when A and B are equal	[4]

the original nonfunctional strain. Instead of using computer models, the authors simply introduced random mutations into the strain and screened for mutants with the desired logic output. Several such mutants were discovered, and sequencing revealed more than one pathway to functionalize the genetic circuit. This is clearly a powerful advantage of cellular computing circuits and will likely become an important part of designing these systems in the future.

Recombination can also be used for self-optimization of cellular computing circuits, and this has been demonstrated by the success of a cellular computing circuit created by Baumgardner *et al.* [8]. This strain contained the genes required for fluorescent protein synthesis, but these genes were mixed up so that they could not function until specific recombination events had occurred. These recombination events were induced by flagging the mixed up DNA region with recombination sites. On growing this strain, the authors found colonies that had successfully produced fluorescence as the output signal by recombination during growth. This system was conceptualized as a Hamiltonian-path solver (i.e., it solves implementations of the traveling salesman problem) and the authors suggested uses for cellular computing circuits in solving problems similar to those for silicon-based computers. However, owing to exponentially increasing processing speeds in silicon computing, it is not likely that biocomputing will ever be competitive in processing speed or versatility. For this reason, we will focus on existing cellular computing circuits and their possible application in biosensing and in controlling biological systems.

Even for simpler computing functions, such as biosensing, a detailed understanding and manipulation of biological systems is necessary. Whole cells are much more complex than enzyme or DNA sequences and in engineering new functions there are often unintended consequences. This is why researchers have mainly used the relatively simple system of *E. coli* with fluorescent protein in cellular computing circuits. Introducing exogenous pathways that do not normally occur in *E. coli* greatly decreases the chance that the genetic circuit will interfere with normal cellular processes, or vice versa. In addition, fluorescent protein is an easily detectable output that does not interfere with cellular processes. However, our work has demonstrated that working within this narrow system is not necessary for creation of functional cellular computing circuits. We will use our recent work as a case study to explain how native bacterial regulation and function can be used in cellular computing circuits when genetic engineering is combined with rational system design to reduce complexity.

14.2 Cellular Computing Circuits

14.2.1 Genetic Toolbox

14.2.1.1 Engineered Gene Regulation

Genetic modification of bacteria is an important tool in the development of cellular computing circuits. A common modification approach has been the use

of inducible promoters (i.e., promoters that can be turned on and off by a specific chemical or environmental signal) to control expression of an output behavior. A wide variety of these promoters exist in nature and may respond to environmental signals, such as pH, temperature, sugar, pollutants, and light [9]. Extracting useful inducible promoters from wild-type bacteria and recombining them in novel ways has resulted in many regulatory networks for controlling cell behaviors. The different types of genetic circuits that have been created, thus far, have been comprehensively reviewed by Voigt [9]. The resulting engineered genetic circuits can control different functions in the bacterium, including the production of chemicals or behaviors, such as cell death. However, cellular computing circuits have not reached application yet, and those developed have mostly used fluorescent protein for the output signal as a proof of concept. Fluorescent protein has, indeed, been a useful output signal because it is simple to detect and does not interfere with normal cellular processes.

There are a few exceptions to this, and a striking example is a set of *E. coli* strains designed by Anderson *et al.* [10] to detect and invade cancer cells. These strains detect one input (arabinose, homoserine lactone (HSL), or oxygen) and link that output with invasin expression, which allows the cells to invade cancer cell lines. Another example is a population control strain created by You *et al.* [11]. This strain could detect population density and induce production of a killing protein to reduce population when the density became too high. These designs are clearly application-driven, but have limited utility because they are capable of detecting only one input, and are not yet optimized for real-world applications.

Increased complexity and functionality can be gained by combining different promoters in creative ways (i.e., to control not only cell functions, but also each other). This opens the possibility for complex Boolean logic function to simultaneously process multiple inputs. For example, an *E. coli* strain that could count three inputs was created by combining three inducible promoters to control expression of fluorescent protein [12]. This strain counts pulses of three different inducers (arabinose, tetracycline, and IPTG) and responds by producing fluorescent protein only after receiving these three signals in the correct order. The three promoters were combined using three single invertase memory modules in series. This allowed fluorescent protein production only after DNA inversion steps, which were controlled by the inducible promoters, and, because of their arrangement in series, the signals work only in the correct order. Combination of the inducible promoters with the single invertase memory modules demonstrates that combining different types of regulation can be useful for creation of complex genetic circuits. Using the same approach, the authors also created a strain that responded to three pulses of a single inducer (arabinose). This is in interesting implementation because it could be used to monitor pulses of a single substance, but the caveat is that the pulse length must be strictly controlled to avoid fluorescent protein expression after one long pulse. These counters were a successful proof of concept for counting function in bacterial cells; however, this system is subject to the stochastic nature of biology and the different behavior of each individual bacterial cell. Discrepancies between model predictions and the counters' performance show

that there are still issues in obtaining a strain that performs as expected, and in unison. Bacterial communication (e.g., quorum sensing) has, thus far, been useful in partially overcoming this obstacle.

14.2.1.2 Quorum Sensing

One of the most widespread forms of bacterial communication is quorum sensing. Quorum-sensing communication can be used for synchronization of one population or communication among populations in cellular computing circuits, and both these uses have already been demonstrated. Synchronization of a population increases the quality of the output signals by causing all the individual cells to act in unison, resulting in a higher signal maximum and lower signal background. Communication among different populations can be used to link cellular computing circuits together to create more complex functions [3, 4]. Quorum-sensing systems are applicable in synchronization and communication because these pathways control extracellular concentrations of bacterial communication chemicals, which can be detected by all cells in close vicinity. HSLs are used for quorum-sensing communication in gram-negative organisms and some of these HSLs are species specific while others are recognized among different bacterial species. The Lux system of *Vibrio fischeri* is the model gram-negative quorum-sensing pathway, and it controls luciferase expression. Lux-type pathways are found in many other bacterial species and consist of a LuxI (autoinducer) homolog and a LuxR (receptor) homolog [13]. LuxI produces an autoinducer HSL (in a positive feedback loop), which is transported freely across the cell membrane, resulting in equal HSL concentrations inside and outside the cell. When the concentration of the HSL in the medium (and therefore in the cell) is above a certain threshold, the HSL binds to LuxR and the LuxR–HSL complex induces expression of genes that are under control of this system, including *LuxI*. All cells that are exposed to the same solution will experience the same HSL concentration, and therefore, they will have synchronous expression of any genes under quorum-sensing control. Using different combinations of these systems (Lux-type systems have been found in at least 25 different bacteria [13]) will facilitate synchronization and communication for improvement of cellular computing circuits.

14.2.2 Implementations

14.2.2.1 Oscillators

Danino *et al.* [14] engineered an *E. coli* strain to act as synchronized oscillators with a fluorescence output signal using the concept of synchronization through quorum sensing. The genetic circuit was constructed using the Lux pathway, an HSL-degrading protein, and a fluorescent protein. These genes were arranged so that HSL and fluorescent protein production were directly related to HSL concentration. This would have created a simple positive feedback loop without the addition of the HSL-degrading protein, which is induced by HSL after a short delay. This delay caused HSL and the fluorescent protein to accumulate before

the HSL-degrading protein was expressed, resulting in fluorescence oscillations. The cells were grown in microfluidic devices with small trapping chambers where groups of cells could be monitored. In smaller trapping chambers, the group oscillated as a whole, while diffusion limitations in larger chambers caused waves of fluorescence to propagate through the group (see the original publication for videos). To change the frequency and amplitude of the oscillations, the authors had only to alter the flow rate in the microfluidic device. This oscillation function could someday be applicable in biomedical or industrial implementations because it could potentially be adapted for periodic drug or chemical delivery.

An earlier implementation of a cellular oscillator was performed by Elowitz and Leibler [15], and comparing this work to the oscillator created by Danino *et al.* [14] underscores the importance of quorum sensing to proper function. In this study, oscillation was based on regulation within each cell and the response was not averaged by the quorum-sensing system. This resulted in high cell-to-cell variability, making it very difficult to read the results of the oscillator from a bacterial population. The oscillation could only be measured by tracking the fluorescence of single cells microscopically. While this cellular computing circuit could potentially find use in some single-cell research applications, the quorum-sensing-regulated oscillator by Danino *et al.* is clearly a more functional oscillator on a population scale and is more likely to see further development in biotechnology applications.

14.2.2.2 Switches

An important part of computing function is switchable elements (i.e., circuits with bimodal stability). Gardner *et al.* [16] have produced cellular computing circuits that act as toggle switches. The authors created *E. coli* strains containing a simple genetic circuit that switches between high and low fluorescent protein expression based on inputs of IPTG and high temperature using inducible promoters. The switch functioned properly and could be reset. Switching from low to high condition took 6 h, while switching from high to low took only 35 min. This was a result of the different properties of the IPTG- and temperature-sensitive promoters. Although this switch is conceptualized as a memory component, the authors also suggest applications in gene therapy.

14.2.2.3 AND Logic Gates

Although there may be exciting applications for single input cellular computing circuits, an important research goal in this area is the development of strains that simultaneously process two or more inputs. Multi-input cellular computing circuits will be important for detection of several biomarkers simultaneously, which is often required to make a conclusive decision (e.g., disease state, security threat, and environmental contamination). A basic example of a multi-input strain is cellular AND logic gates. The AND Boolean function delivers a true output signal only when both input signals are true (Table 14.1).

AND logic gates in bacteria have previously been demonstrated. For example, Ramalingam *et al.* [6] constructed plasmid-encoded AND logic gates in *E. coli* with IPTG and tetracycline as the inputs and fluorescent protein as the output. The

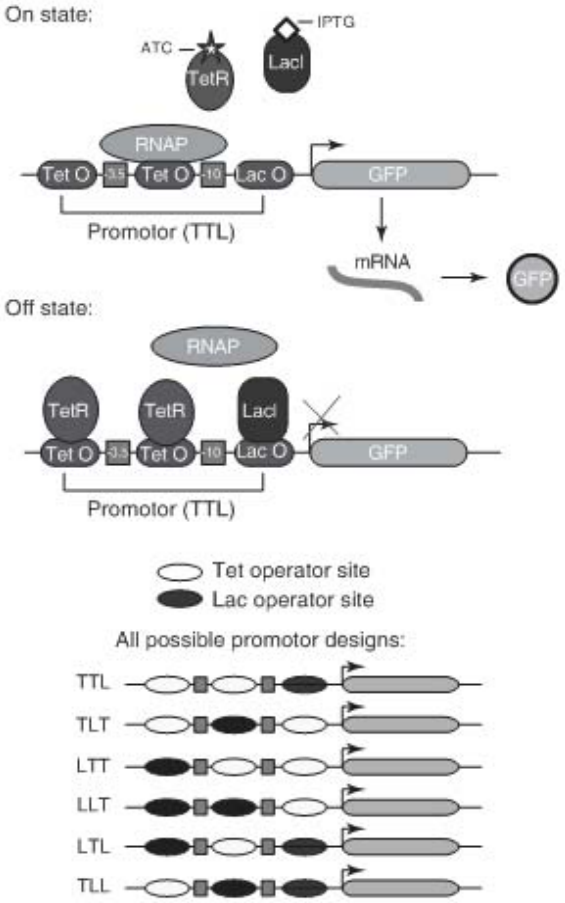


Figure 14.1 Genetic circuit and AND logic gate mechanism. (From [6] with permission.)

authors used six different combinations of two inducible promoters to construct their genetic circuits (Figure 14.1).

Comparison of the performance of the different genetic circuits showed that fine details of the genetic engineering approach have an impact on the characteristics of the strain as a whole. While none of the logic gates was 100% accurate, some logic gates performed better than others, indicating that alteration of the circuit arrangement can improve performance of cellular computing circuits.

14.2.2.4 Edge Detector

For more complex signal processing in one strain, it is sometimes necessary to incorporate more than one logic function. Tabor *et al.* [3] constructed an edge detection program in *E. coli* and this required the combination of three different logic gates. To create this program, the authors combined an AND logic gate with two NOT logic gates (Figure 14.2).

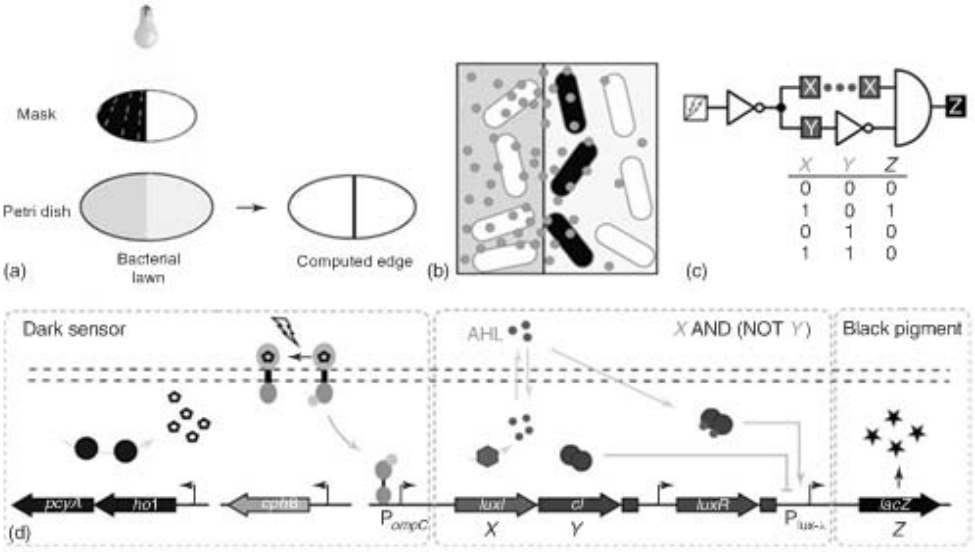


Figure 14.2 Genetic circuit and mechanism of genetically encoded edge detector showing overall input and output (a), diffusion of signal chemicals (b), logical function (c), and overview of the genetic circuit (d). (From [4] with permission.)

The three logic gates used carefully chosen inputs and outputs to create a complete genetic circuit with the desired output characteristics. This circuit caused transformed *E. coli* cells to produce HSL only in the dark, and pigment only in the presence of HSL and light. Therefore, in populations of this strain HSL and light will only coincide at light/dark edges, and pigment will be produced only at these edges. Black and white edge detection is a very specific function, but this approach has already been extended to engineer a cellular computing circuit that can also distinguish between red and green light [17].

14.2.2.5 Complex Logic Functions with Multiple Strains

When a function becomes very complex, it may become difficult to encode the entire processing pathway into a genetic circuit small enough and simple enough to be introduced into one strain. In this case, different parts of the genetic circuit can be introduced into different strains, provided that the strains can communicate and interact with each other. This concept has been demonstrated by Tamsir *et al.* [4], who engineered different genetic circuits into eight unique *E. coli* strains. Each individual strain carried only one simple logical function, but more complex processing was achieved by combining up to four of the individual strains through the Las and Rhl quorum-sensing pathways, which are other important quorum-sensing systems besides the Lux type [13]. Combinations were achieved by placing colonies of different strains near each other on agar plates so that HSLs could diffuse between them. Through these combinations, the authors were able to construct more complex logic operations, such as XOR, EQUAL, NAND, and A IMPLY B, some of which have not yet been demonstrated in single-strain cellular computing circuits.

The library of inputs contained two different HSL molecules (3-oxododecanoyl homoserine lactone (3-oxo-C12-HSL) and *N*-butyryl homoserine lactone (C4-HSL)), tetracycline, and arabinose, while the output library contained the same HSL molecules and fluorescent protein. This system can be conceptualized as a biocomputing platform with a library of interchangeable parts, although the specificity of connections between the strains will be an obstacle in future design [4].

An alternative use for cellular computing circuits that communicate and interact with each other (rather than creation of complex logic function) is control of environmental and industrial microbial communities. For example, cellular computing circuits could be introduced to industrial fermenters or environmental remediation sites to monitor and control activity of other bacterial populations. Brenner *et al.* [18] created a first step in functional bacterial communication by designing two *E. coli* strains that could sense each other and produce fluorescent protein as a signal. Each strain required one of two HSL molecules (3-oxo-C12-HSL and C4-HSL) to be produced by the other strain to generate the fluorescent protein. This is the first demonstration of a cellular computing circuit that uses other bacteria as an input signal. This study is an important first step toward cellular computing circuits that can sense, decide, and act, based on the cues of a microbial community.

A similar example is a pulse-generating cellular computing circuit designed by Basu *et al.* [19]. Rather than two co-dependent strains, this system consisted of a sender and a receiver strain. The sender strain produced a constant output of an HSL signal, while the receiver strain responded to the HSL by producing a pulse of fluorescent protein. The specifics of the genetic circuit used also caused the receiver strain to be sensitive to the rate of signal increase. Because of this property and the separation of this function into two strains, the authors were able to use patterning of the two strains on solid medium to generate specific patterns of output fluorescence. Essentially, the receiver strain is capable of detecting the sender strain within a certain proximity, suggesting that this cellular computing circuit could be used to detect distance-sensitive variables in biotechnological applications. Indeed, the authors later published another study in which the receiver cells were engineered to detect specific concentration ranges of HSL, and the two strains were used to create complex fluorescence patterns [20].

14.2.3
Transition to *In Silico* Rational Design

A common element of most current cellular computing circuits is that the basic structure of the genetic circuit has been designed manually. This is a time-consuming endeavor, requiring both great depth and great breadth of knowledge of biochemical regulatory mechanisms. However, this process can be aided by computational design of genetic circuits. Rational redesign of existing circuits is also an important part of this process, whereby algorithms design modifications of existing genetic machinery to make it more favorable for implementation in cellular computing circuits [21]. Preliminary studies also show that computational models can do more than just model the behavior of a defined circuit, but also

design some genetic circuits, if given the right building blocks. Batt *et al.* [22] have shown that genetic circuits can be improved by characterizing and compiling genetic circuits, which can be combined and tested *in silico*.

A recent study by Purcell *et al.* [23] demonstrates *in silico* design of genetic circuits for cellular computing functions. The authors used mathematical computer modeling to design a genetic circuit with multiple functions: frequency multiplication, oscillation, and switching. The different functionalities could be used with different input concentrations and temporal profiles. The frequency multiplier is a particularly interesting function because it has not yet been reported in a cellular computing circuit. Unfortunately, the authors did not base the computer model on existing genetic components, but rather idealized components. Thus, implementation of this abstract genetic circuit will not be simple, since real genetic components will not behave like the idealized components used in the model. Future efforts toward *in silico* design will need to incorporate not only design parameters of the desired system but also properties of available genetic circuit components.

In silico design opens the door toward much faster strain design, testing, and creation. A pipeline for future creation of cellular computing circuits through computer-aided design is already under way, which will allow faster tailoring of cellular computing circuits toward real-world applications. In addition, increased complexity will likely be achieved through use of sophisticated design algorithms. Computer-aided design will be necessary for development of this field because of the extreme complexity of living cells. In contrast, enzyme computing has already reached a higher level of complexity without *in silico* design owing to the simplicity and specificity of single enzymes.

14.2.4 Transition from Enzyme Computing to Bacteria-Based Biocomputing

Although enzyme computing has already reached greater complexity than bacterial computing, there are several reasons to develop bacterial computing. To overcome the disadvantages of enzyme-based logic gates (e.g., sensitivity to environmental factors and inability to reproduce), transition from enzyme-based Boolean logic gates to bacteria-based logic gates is desirable. Although enzyme-based Boolean logic gates are more selective and specific, bacteria-based logic gates are self-renewing and more robust. These advantages make bacteria-based logic gates more suitable for certain applications, especially when the computation speed is not critical. In addition, the ability to produce more complex output signals than enzyme-based Boolean logic gates allows bacteria-based logic gates to perform more complex tasks than enzymatic cascades. Therefore, we translated an enzyme-based Boolean logic gate with a biofuel cell (a type of bioelectrochemical system (BES)) that is able to produce a direct electric current digital signal into a bacteria-based logic gate by replacing the electrochemically active enzyme with an electrochemically active bacterium [24]. Since direct electric current is an easily measurable and codified digital signal, the translation steps for converting analog signal to digital signal

(e.g., fluorescence measurement) can be omitted. This will simplify the biocomputing process, decrease error propagation, and simplify the bacteria–machine communication interface.

Our system was built with a three-electrode potentiostatically controlled BES with *Pseudomonas aeruginosa* PA 14 as the core part of the logic gate. In our previous work, quorum sensing had proven to be an effective mechanism for controlling electric current production in BES [25]. Here, we used a *P. aeruginosa* PA14 Δ*lasI*/Δ*rhlI* double mutant with both quorum-sensing chemicals (3-oxo-C12-HSL and C4-HSL) as input signals for our system because we used a mutant strain without the self-secretion autoinducers LasI and RhlI, but with functional LasR and RhlR receptors [5]. This mutant cannot produce the signal but can still detect it and trigger the quorum-sensing cascade to induce phenazine production. Phenazines are electron mediators for electron transfer between *P. aeruginosa* PA14 Δ*lasI* and electrodes [25–27], resulting in an electric current signal. Thus, the phenazine production could be upregulated only by adding these two quorum-sensing signals. Indeed, our result showed that the electric current increased dramatically only when both input signals were present in the system. By setting a proper threshold, the logic gate gave an output signal 1 only when both input signals were true (presence of the input signal is defined as true and absence of the input signal is defined as false), which makes the logic gate an AND logic gate. This result illustrates that, although replacing the enzyme with bacteria increases the system complexity, highly specific input signals can reduce the uncertainty in signal processing and deliver expected and clear output signals.

The AND logic gate was built in two BES configurations: microbial fuel cell (MFC) and microbial three-electrode cell (M3C). In MFC mode, a natural potential difference is utilized for electric current production and no external power supply is required, which makes this AND logic gate a self-powered biosensor [5, 28]. To obtain a clear output signal, at least 115 h were needed to allow the output signal definition to increase to the desired level. The very slow processing speed makes this biosensor suitable only for the areas where the computing speed is not critical. Another problem that should be noted is that in MFC mode, the output signal can be affected by both anode and cathode performance. Thus, both the anode and cathode noise propagate through the entire signal processing and can cause a decrease in the output signal definition (Figure 14.3).

We circumvented these problems and attained a more robust output signal by using a M3C in which the working electrode potential is controlled at a stable level by a potentiostat, resulting in a better defined electrochemical environment than MFC. As expected, the output signal definition was improved in the M3C mode (Figure 14.4).

The direct digital output signal is one of the major improvements in this system compared to other bacterial logic gates with analogous output signals (e.g., fluorescent protein, optical density). These analogous signals require additional equipment to detect, and thus might decrease the signal definition. The electrochemically active bacteria are core parts to create these logic gates. Using the native regulatory systems of electrochemically active bacteria is a good strategy for

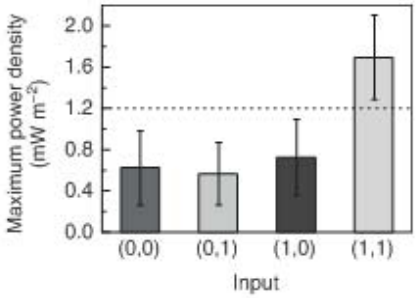


Figure 14.3 Bar diagram showing maximum MFC power density for all input combinations. (From [5] with permission.)

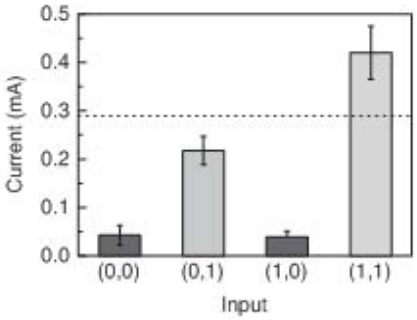


Figure 14.4 Bar diagram showing maximum M3C current production for all input combinations. (From [5] with permission.)

building bacterial logic gates because the electron transfer pathways are not fully understood and cannot easily be transferred to other organisms at this time. Similar to *P. aeruginosa* PA14 used here, another electrode respiring bacterium, *Shewanella oneidensis*, shuttles electrons between cells and electrodes via riboflavin, and is an alternative option [29]. We propose to use different types of electrochemically active bacteria to develop logic gates for various applications

14.3 Conclusion

Cellular computing circuits are in an early stage of development, although increased speed of development is likely with emerging computer-aided design. Based on the current implementations discussed here, we expect development of cellular computing circuits for application in the near future. Cellular computing circuits are likely to be used for biosensing and biocontrol in environments, such as the human gut, industrial fermenters, and in the natural environment. The development of cellular computing circuits with a direct electrical output signal will be particularly useful in biosensing applications, however, that do not require immediate response times. We predict that combining this concept with cutting

edge strain design will lead to great improvement in biosensing and biocontrol through cellular computing circuits.

Acknowledgment

This work was supported through an NSF Career Grant # 0939882 to LTA.

References

1. Yokobayashi, Y., Weiss, R., and Arnold, F.H. (2002) *Proc. Natl. Acad. Sci.*, **99**, 16587–16591.
2. Weiss, R. (2001) *Cellular Computation and Communications Using Engineered Genetic Regulatory Elements*. PhD thesis. Massachusetts Institute of Technology, Cambridge, MA.
3. Tabor, J.J., Salis, H.M., Simpson, Z.B., Chevalier, A.A., Levskaya, A. *et al.* (2009) *Cell*, **137**, 1272–1281.
4. Tamsir, A., Tabor, J.J., and Voigt, C.A. (2011) *Nature*, **469**, 212–215.
5. Li, Z.J., Rosenbaum, M.A., Venkataraman, A., Tam, T.K., Katz, E., and Angenent, L.T. (2011) *Chem. Commun.*, **47**, 3060–3062.
6. Ramalingam, K.I., Tomshine, J.R., Maynard, J.A., and Kaznessis, Y.N. (2009) *Biochem. Eng. J.*, **47**, 38–47.
7. Anderson, J.C., Voigt, C.A., and Arkin, A.P. (2007) *Mol. Syst. Biol.*, **3**, 133.
8. Baumgardner, J., Acker, K., Adefuye, O., Crowley, S.T., Deloache, W. *et al.* (2009) *J. Biol. Eng.*, **3**, 11.
9. Voigt, C.A. (2006) *Curr. Opin. Biotechnol.*, **17**, 548–557.
10. Anderson, J.C., Clarke, E.J., Arkin, A.P., and Voigt, C.A. (2006) *J. Mol. Biol.*, **355**, 619–627.
11. You, L., Cox, R.S., Weiss, R., and Arnold, F.H. (2004) *Nature*, **428**, 868–871.
12. Friedland, A.E., Lu, T.K., Wang, X., Shi, D., Church, G., and Collins, J.J. (2009) *Science*, **324**, 1199–1202.
13. Miller, M.B. and Bassler, B.L. (2001) *Annu. Rev. Microbiol.*, **55**, 165–199.
14. Danino, T., Mondragon-Palomino, O., Tsimring, L., and Hasty, J. (2010) *Nature*, **463**, 326–330.
15. Elowitz, M.B. and Leibler, S. (2000) *Nature*, **403**, 335–338.
16. Gardner, T.S., Cantor, C.R., and Collins, J.J. (2000) *Nature*, **403**, 339–342.
17. Tabor, J.J., Levskaya, A., and Voigt, C.A. (2011) *J. Mol. Biol.*, **405**, 315–324.
18. Brenner, K., Karig, D.K., Weiss, R., and Arnold, F.H. (2007) *Proc. Natl. Acad. Sci.*, **104**, 17300–17304.
19. Basu, S., Mehreja, R., Thiberge, S., Chen, M.-T., and Weiss, R. (2004) *Proc. Natl. Acad. Sci.*, **101**, 6355–6360.
20. Basu, S., Gerchman, Y., Collins, C.H., Arnold, F.H., and Weiss, R. (2005) *Nature*, **434**, 1130–1134.
21. Purnick, P.E.M. and Weiss, R. (2009) *Nat. Rev. Mol. Cell Biol.*, **10**, 410–422.
22. Batt, G., Yordanov, B., Weiss, R., and Belta, C. (2007) *Bioinformatics*, **23**, 2415–2422.
23. Purcell, O., di Bernardo, M., Grierson, C.S., and Savery, N.J. (2011) *PLoS ONE*, **6**, e16140.
24. Willner, I. and Katz, E. (2003) *J. Am. Chem. Soc.*, **125**, 6803–6813.
25. Venkataraman, A., Rosenbaum, M., Arends, J.B.A., Halitschke, R., and Angenent, L.T. (2010) *Electrochem. Commun.*, **12**, 459–462.
26. Rabaey, K., Boon, N., Siciliano, S.D., Verhaege, M., and Verstraete, W. (2004) *Appl. Environ. Microbiol.*, **70**, 5373–5382.
27. Venkataraman, A., Rosenbaum, M.A., Perkins, S.D., Werner, J.J., and Angenent, L.T. (2011) *Energy Environ. Sci.*, **4**, 4550–4559.
28. Fornero, J.J., Rosenbaum, M., and Angenent, L.T. (2010) *Electroanalysis*, **22**, 832–843.
29. Lies, D.P., Hernandez, M.E., Kappler, A., Mielke, R.E., Gralnick, J.A., and Newman, D.K. (2005) *Appl. Environ. Microbiol.*, **71**, 4414–4426.