Hybrid DNA and Enzyme Based Computing for Address Encoding, Link Switching and Error Correction in Molecular Communication

Frank Walsh[2], Sasitharan Balasubramaniam[1,2], Dmitri Botvich[1,2], Tatsuya Suda[3], Tadashi Nakano[3], Stephen F. Bush[4], and Mícheál Ó Foghlú[1,2]

[1] Telecommunication Software and Systems Group
{sasib,dbotvich,mofoghlu}@tssg.org
[2] Waterford Institute of Technology
Carriganore Campus, Ireland
{fwwalsh}@wit.ie
[3] Information and Computer Science
University of California, Irvine, CA, USA
{suda,tnakano}@ics.uci.edu
[4] Computing and Decision Sciences
GE Global Research
Niskayuna, NY, USA
bushsf@research.ge.com

Abstract. This paper proposes a biological cell-based communication protocol to enable communication between biological nanodevices. Inspired by existing communication network protocols, our solution combines two molecular computing techniques (DNA and enzyme computing), to design a protocol stack for molecular communication networks. Based on computational requirements of each layer of the stack, our solution specifies biomolecule address encoding/decoding, error correction and link switching mechanisms for molecular communication networks.

Keywords: Molecular communication, molecular computing, communication protocols.

1 Introduction

In common with networked computing devices, biological cells have the ability to transmit, receive and process information through signaling networks and signal transduction mechanisms that interact in a complex biochemical system [6][7]. Just as modular silicon components are used to compose digital electronic circuits, the mechanisms that underpin biological systems are now being investigated to create a library of molecular components that can be used to engineer biological based nano (*bio-nano*) scale systems. One good example is Molecular Computing [12], which manipulates biomolecules to engineer biochemical based computing systems. By combining Molecular Computing and Molecular Communication [1], a new research

M. Cheng (Ed.): NanoNet 2008, LNICST 3, pp. 28–38, 2009.

domain that investigates bio-nano communication, the necessary computing mechanisms can be provided to create communication protocols for bio-nano devices (in the rest of this document, this will be referred to as *nanodevice*). Just as data communication protocols resulted in the rapid growth and ubiquity of networked computing devices and applications, the development of communication protocols for nano-based networks will stimulate groundbreaking future applications of bio-nano devices. The potential applications of these combined technologies are vast, particularly in the medical field where nano-scale devices can perform surgical procedures [14] or ensure accurate drug delivery to specific parts of organs and tissues.

Biological cells contain various components that can play vital roles in networked communication. These include, for example network interfaces (receptors, gap junctions), computing processes (regulatory networks, enzymatic signaling pathways) and memory capabilities (nucleic acids). In this paper, we propose a cell-based communication platform that uses these functional complexities to create protocols necessary for molecular communication networks. Our proposed hybrid solution includes DNA as well as enzyme based computing, where each contributes to specific protocol functions. We will describe how we will re-use protocols from communication networks, and transfer their mechanisms to a cell-based environment. In particular, we will show how our molecular communication protocol stack can support addressing, error correction, and link switching.

The paper is constructed as follows: Section 2 reviews the background of molecular communication and computing. Section 3 investigates protocols for data communication and how we reuse some of these concepts for our proposed protocols for molecular communication. Section 4 presents a simple connectionless communication solution using biological cells as a communication platform for address encoding, error correction, and link switching. Finally, section 5 presents conclusions and future work.

2 Background

2.1 Molecular Communication

Molecular Communication uses encoded molecules as information carriers to engineer biochemical-based communication systems. In [9], Moritani et al define a Molecular Communication Interface that uses vesicles embedded with gap junction proteins to transport message-encoded molecules. The vesicles that embed the information molecules (e.g. this could be represented as metabolites, or small nucleotides) will then be used as signal carriers between the sender and receiver nano-devices. Another form of molecular communication exploits the current calcium signalling that occurs between cells. For example, in [10] Nakano et al showed that distant nanodevices can communicate by encoding information through the frequency and amplitude of inter-cellular calcium waves.

2.2 Molecular Computing

This section will describe two common molecular computing techniques which include DNA and enzyme based computing. A summary and the characteristic differences between the two types of computation are also described.

2.2.1 DNA Based Computing

DNA is the universal "information molecule" and has a number of advantages in the computing world, such as encoding information as sequence of biochemical symbols as well as using these symbols to perform computing operations. In [3], Benenson et al present a programmable autonomous finite state automaton consisting entirely of biomolecules. The authors' design consists of a long DNA input molecule that is processed repeatedly by a restriction enzyme, and short DNA "rule" molecules that control the operation of the restriction enzyme. This concept forms the basis for a nanoscale computing machine that diagnoses disease and releases treatment molecules based on several disease-indicating inputs [11]. In [17], Liu et al extended the molecular automaton presented in [11] to design a "DNA-based Killer Automaton" that can release cytotoxic molecules which propagate to neighboring cells via gap junction channels.

2.2.2 Enzyme Based Computing

Markevich et al [4] created a bistable switch using a cell-based Kinase-Phophatase signaling cascade (MAPK) that is highly conserved in eukaryotic cells. In doing so, the author demonstrates the use of ultra-sensitive cell-based enzyme signaling pathways to perform digital logic computation. Similarly, in [5] Stetter et al uses the bistable nature of biochemical enzymatic reactions to create a reusable, "easy to engineer" architecture that forms the basis of several Boolean logic functions such as AND, and OR gates. This small enzyme-based circuit can act as a sub-component in composing more complex functions.

There are a number of differences between the two types of cell-based computing, where each has certain disadvantages and advantages with respect to computing for communication protocols. Firstly, the computational complexity and speed associated with DNA computing is, as yet, not attainable using enzyme based computing [16]. Also, the parameter characterization effort required to achieve enzyme computing increases dramatically relative to circuit complexity [13]. This makes enzyme computing more suitable for relatively simpler circuits that require short computation time. On the other hand, DNA-based computing can support larger computing requirements. The other difference between enzyme and DNA computing is that enzymatic reactions are intrinsic in cytosolic cell signaling pathways [7]. Therefore, this allows closer interaction with cell membrane components such as receptors and gap junctions. This makes it particularly suitable to simpler, responsive computing involving extra-cellular input and output.

3 Defining Protocols for Molecular Communication

In this section we will first describe the core characteristics of communication network protocols, and how these protocols will be re-used to support nanodevices.

3.1 Communication Network Protocols

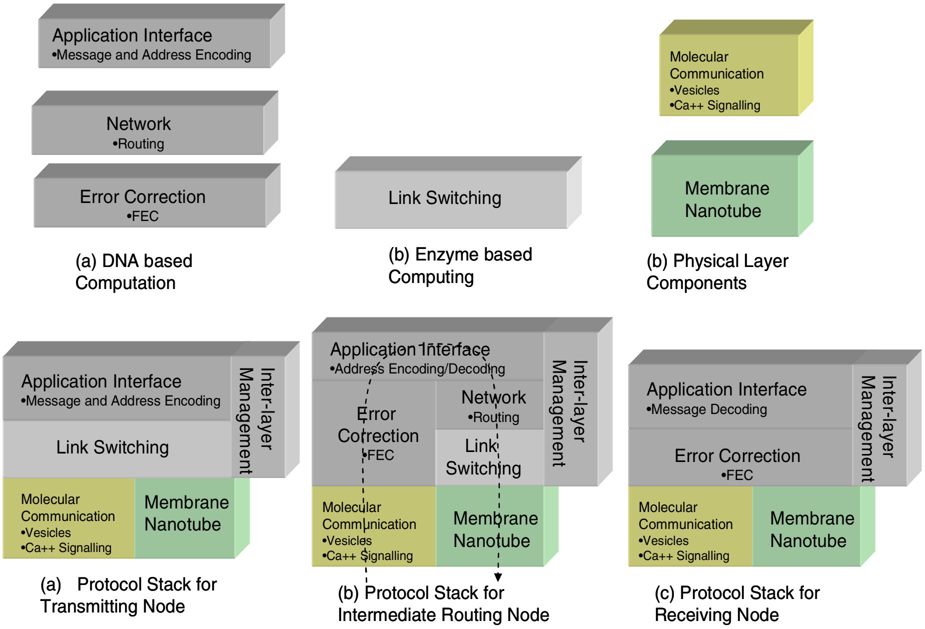
Communication networks consist of protocols that exhibit the following properties; access mechanisms to physical communication interfaces, encoding and addressing mechanisms, error detection/correction techniques, and routing of packets between connected nodes. Physical interface controllers provide connection to physical transmission media and include mechanisms such as modulation and channel coding. The link layer functions manage access to the underlying physical layer, while flow control and acknowledgment mechanisms are usually implemented in higher layer protocols such as TCP. Communication can be connectionless or connection-oriented, where connectionless communication have lower data overhead, and are suitable for energy efficient networks such as wireless sensor networks. Another common protocol used in communication network is error correction, where techniques such as Forward Error Correction (FEC) can ensure that end devices can recover from any data corruption incurred during transmission. One approach is through inclusion of redundancy in channel encoding process.

3.2 Protocols for Molecular Communication

As described earlier, our intention is to be able to re-use protocols from conventional communication networks for molecular communication. Fig. 2 illustrates the components of our protocol stack and the protocols for different operations of the nanodevice (e.g. Transmitting node, Receiving Node, Intermediate Routing Node). Our approach is based on interconnection of loose protocol components, where each component is performed by a specific molecular computing technique. The reason that we have not embed all components into a generic protocol stack, is to prevent unnecessary increase in computational complexity. Although, the components of each layer is mapped from conventional protocols used in communication networks, the layers of our protocol stack is re-organised to suit a number of characteristics found in molecular communication. For example, propagation of information in molecular communication is typically characterized as low speed and in an environment where the interconnecting links between nano devices use biological signaling mechanisms that are highly variable compared to standard communication networks [1][2]. These characteristics have repercussions for the design of protocols of molecular communication systems. Slow diffusion-based processes do not support the creation of high-speed switching functions common in conventional network devices that will require complex queuing mechanisms for packets. At the same time, due to high variability and harsh biological environment, the use of acknowledgements and retransmission of messages in the event of loss or corrupt packets may not lead to improved performance.

We anticipate two types of information transmissions used in molecular communications, which includes sensory data (data collected from nanodevices) and command data (instructions for nanodevices). Therefore, the transmission mechanism and protocols to be used will be highly dependent on the nature of the information. For example, for sensor data, we may use single paths with UDP-like transmission with no error correction. However, command information or high priority sensor data will be transmitted through redundant paths with error correction capabilities (e.g. FEC).

Since protocols can usually be defined through a Finite State Machine (FSM), we adopt a nano-logic circuit that is translated from a FSM to represent the different types of protocols. We then map the specific protocol to either DNA or enzyme based computing. Since each technique has its own characteristics, we apply and select the right



(a) DNA based Computation

(b) Enzyme based Computing

(b) Physical Layer Components

(a) Protocol Stack for Transmitting Node

(b) Protocol Stack for Intermediate Routing Node

(c) Protocol Stack for Receiving Node

Fig. 1. Molecular Communication protocol stack

techniques based on two factors which includes, (i) the sequence of operation for the protocol, and (ii) complexity of computation required for the protocol. The DNA based computing is used for *Application Interface*, *Network*, and *Error Correction* layers, while the enzyme based computing is used for the *Link Switching* layer. The Application Interface, Network, and Error Correction layers will require higher complexity computation and is usually not required to be time sensitive. Such computations will include FEC, addressing, and information encoding/decoding. Enzyme based computing, due to its limited time requirement, is most suitable in performing small size logic circuit with high-speed computation. Therefore, this is most ideal for switching of information biomolecules between the links. The underlying physical layer can be based on solutions by [1] [10] for molecular communication, where the molecular communication can be guided through membrane nanotubes [19]. We select membrane nanotubes as a physical layer communication mechanism between cells, essentially providing the guided channels interconnecting each node in the bio-nano network. Unlike intercellular communication mechanisms that broadcast chemical signals to all neighboring cells via intercellular space, these nano-tubular structures can create a network of communication links between distinct cells that can support intercellular transfer of cytosolic molecules, vesicles and organelles. A notable work is by Önfelt et al who demonstrated a membrane network that transports tagged vesicles from cell to cell [19]. Therefore, the membrane nanotubes could be used in conjunction with a suitable molecular communication mechanism such as [9] that uses vesicles to transport message molecules or [10] to guide modulated calcium “waves” from sending cell to receiving cell.

In between the two layers will be the *Inter-layer protocol management*, which will coordinate the different computation of each layer of the protocols and the location where this will happen in the cell. Fig. 2 illustrates our solution that combines a subset of our proposed protocol to support transmission on a single link.

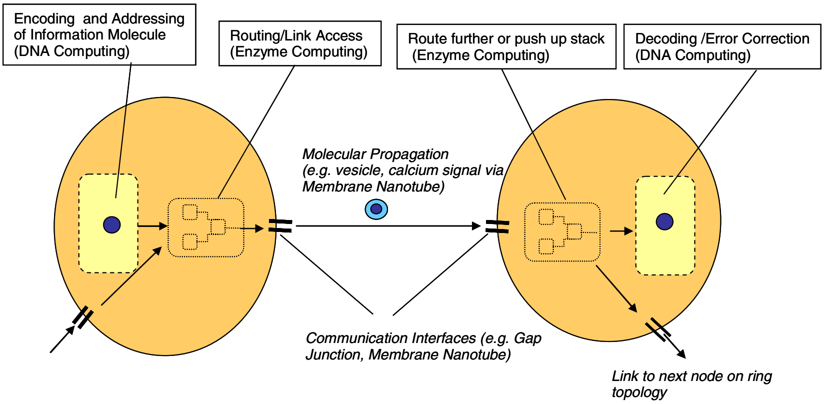

Encoding and Addressing of Information Molecule (DNA Computing)
Routing/Link Access (Enzyme Computing)
Route further or push up stack (Enzyme Computing)
Decoding /Error Correction (DNA Computing)
Molecular Propagation (e.g. vesicle, calcium signal via Membrane Nanotube)
Communication Interfaces (e.g. Gap Junction, Membrane Nanotube)
Link to next node on ring topology

Fig. 2. Mechanism of transmission for single link molecular communication

The flow of operation between the layers is as follows. For the Transmitting Node (Fig. 1(a)), the Application layer Interface will perform the message encoding for the information biomolecules. The encoded biomolecule is then further encoded with the specific address of the intended destination using an address table. In our proposed protocol stack, we have left our application layer open, where the cells can interface to a physical device or we can have artificial cells with embedded functionalities (e.g. the cell also acts as the device). Once the encoding process is performed, the information biomolecule is ready for transmission and submitted to the Link Switching layer, which selects the correct gap junction for transmission. In the Intermediate Routing Nodes case (Fig. 1(b)), when the biomolecule is received by the cell, the error correction is first performed on the biomolecule. This is then followed by the address decoding and encoding process based on the routing table for the next node. Once this is performed, the link switching operation follows and transmits the biomolecule to the underlying link. Once the information biomolecule is received at the receiving device (Fig. 1(c)), the information biomolecule is once again passed through the Error Correction layer to perform any necessary error correction, which is then followed by the message decoding at the Application Interface layer.

4 Proposed Solution

In this section, we will describe the molecular computing operations for information encoding and addressing, link switching, as well as error correction.

4.1 Encoding and Addressing

Fig. 3(a) illustrates the encoding process. Similar to the model proposed by Liu et al in [17], our solution uses Benenson's and Shapiro's work in [3] to create a DNA-based automaton that produces a *single strand DNA* (*ssDNA*) message molecules for intercellular communication. Each ssDNA message is encoded as a unique sequence of nucleotide bases as demonstrated in [4]. For simplicity, only three addressable nano-device nodes are considered and each encoded ssDNA message is 'framed' to include addressing information.

Fig. 3(a) illustrates how nucleotide encoded messages are assembled in sequence of long input double stranded DNA message molecule with each message separated by a 'spacer' sequence. The upper leftmost "sticky end" represents the current state of the machine. During the address encoding process, the DNA message molecule is cut by a restriction enzyme, which releases the leftmost segment of the molecule. Thus the <*address, message*> pairing represented by the current state of the encoding automaton is released as an ssDNA segment through the restriction process. Fig. 3(b) illustrates how each address state and transition corresponds to actual encoded message molecule. Each state transition is enacted by a corresponding DNA "rule" molecule and enzyme complex that cleaves the corresponding nucleotide sequences. A key characteristic of address encoding is the precise cleaving of input message molecule that encodes or "frames" the message.

Fig. 4 illustrates a rule execution transition from Address 2 to Address 3. Each rule molecule has a recognition site to which a restriction enzyme can bind. As described earlier, the number of nucleotide bases between the restriction enzyme and the sticky end of the rule molecule determines the precise locations of the message molecule

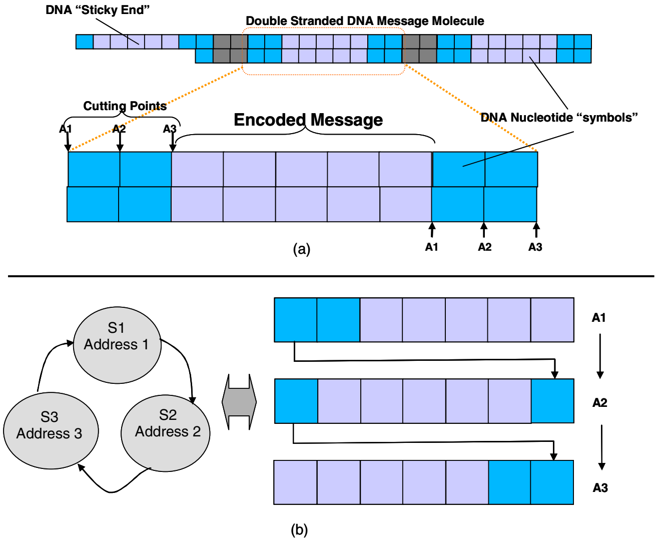


Fig. 3. (a) Double Stranded DNA message molecule indicating restriction cut points for address encoding, (b) State representation of address encoding transitions

cleave. In this example, the restriction enzyme complex combines with the message molecule and cuts at fourteen nucleotides on the top and twenty-one nucleotides at the bottom. The resulting new sticky end reveals the next state of the automaton. More importantly, the segment that is cut away separates into two ssDNA molecules. The lower ssDNA molecule indicated in Fig. 4 is the encoded message molecule with its rightmost end complementary to the new sticky end of the DNA message molecule.

Similar to techniques used in [17] and [11], the nanodevice can control computation by releasing molecules (e.g. mRNA) that selectively activate DNA *"rule"* molecules. The results of the computation can provide input to other parallel computational functions, which was proposed in [3]. In our solution, the cleaved ssDNA message molecules are released into the cytosol and provide the input to the molecular interface control function of the network layer. Theoretically, this mechanism can be extended to encode a multitude of unique address locations and any number of messages during computation.

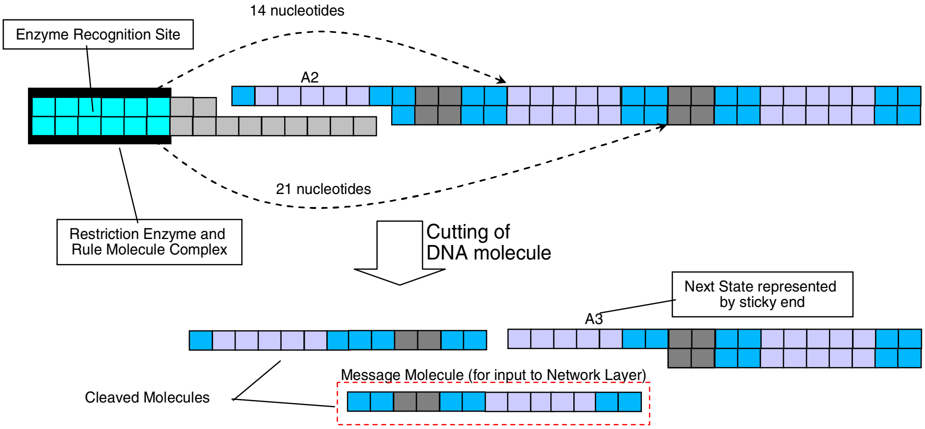
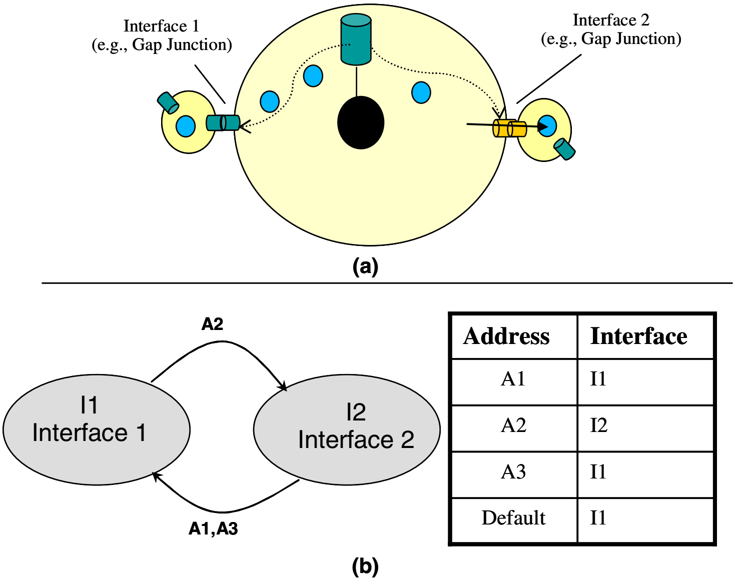

14 nucleotides
Enzyme Recognition Site
A2
21 nucleotides
Restriction Enzyme and Rule Molecule Complex
Cutting of DNA molecule
Next State represented by sticky end
A3
Message Molecule (for input to Network Layer)
Cleaved Molecules

Fig. 4. Mechanism of State Transition from Address 2 to Address 3 using Benenson's Molecular Automata [3]

4.2 Molecular Interface Control

As described earlier, the operation of our molecular communication is through a membrane nanotube network. Fig. 5 illustrates a cell with two distinct molecular communication interfaces (e.g. distinct gap junctions). Each addressable location is switched through the corresponding interface according to the addressing state diagram shown in Fig. 5 (b). For communication involving the transfer of message molecules through gap junctions, our solution is based on results in [18] which demonstrate the diffusion of synthetic oligonucleotides through gap junction channels. In our case, instead of oligonucleotides, we diffuse our encoded ssDNA from the previous section.

In this study, interface selection is achieved using the "real world" implementation of the logical recurrent architecture as described by Stetter et al in [5]. The switching circuit releases/alters a corresponding chemical signal that "switches" the ssDNA to



Address	Interface
A1	I1
A2	I2
A3	I1
Default	I1

Fig. 5. (a) Schematic diagram of cell with two distinct molecular communication interfaces, (b) Address/Interface state diagram and switching table

the correct interface. In the case of gap junction interfaces, the output of the enzyme-based circuit will control the permeability of gap junction channels. Gap junction permeability is affected by the connexin phosphorylation [10] via specific concentration of phosphorylation reagents.

Thus Stetter's circuit can be used to effectively switch on and off each molecular communication interface by controlling the degree of phosphorylation of gap junction connexins. This in turn will allow the ssDNA to be pushed through only a single link (or multiple links if multicasting is used). Using this technique, several communication links can be controlled simultaneously via compartmentalized enzymatic functions [8]. The Inter-layer protocol we will be responsible for triggering the enzymatic computation, once the operation from the Application Interface layer is complete (the operation of this mechanism is subject to future work).

4.3 DNA Decoding and Forward Error Correction

As already stated, prioritized messages require error detection and correction. Invariably, errors will occur in the encoding and transmission process of ssDNA molecules due to the imprecise nature of the associated complex biochemical reactions [15]. By including redundancy in the encoding process, error correction mechanisms can be incorporated into the decoding process. Our solution combines the nucleotide redundancy concept presented in [16] with DNA automata design in [11] to create an autonomous error correction mechanism. In our proposed technique, each ssDNA molecule is composed of several repeated, identical nucleotide sequences.

In [11] Benenson uses "protector strands" to control the operation of an enzyme based state machine by separating the constituent DNA strands of message molecules (see Fig. 6). In our solution, the protector strands are designed to have a strong affinity for a specific received ssDNA. The ssDNA molecules cause the corresponding protector strand to separate from the transition strand and hybridize with the message molecule allowing the formation, and thus activation, of a double stranded transition molecule (a similar mechanism to the encoding process). The resulting transition molecule and restriction enzyme complex cleaves the corresponding decoding DNA molecule and releases the decoded DNA molecule (in Fig. 6, this is represented as the end DNA hairpin) with no errors. Our assumption of this approach is mainly for finite instruction messages, where our end device will contain as many Decoding DNA molecule as the number of possible instructions. Hybridization can also occur even though both the protector strand and the received ssDNA molecule are not exactly complementary.

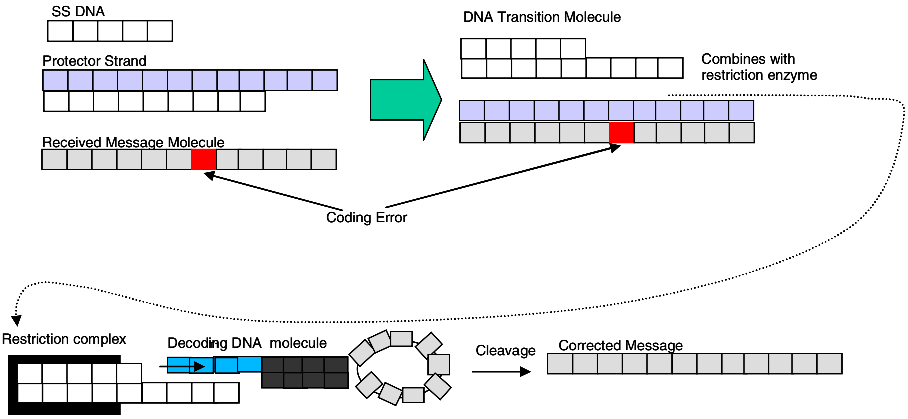


Fig. 6. Forward Error Correction Mechanism

5 Conclusion and Future Work

Inspired by protocols for communication networks, we have presented a molecular communication protocol stack that successfully combines molecular computing and molecular communication techniques. We describe how the core characteristics of communication network protocols are re-used to design bio-nano device communication protocols. Our proposed protocol stack presents the address encoding/decoding, link switching, and error correction functions that are developed using molecular computing techniques. The solution demonstrates the necessity of matching the characteristics of each molecular computing technique to the computational requirements of each layer of the proposed protocol stack. Our future work will investigate the feasibility of our design initially through simulation of chemical circuits for molecule encoding/decoding, link switching and error correction.

Acknowledgements. This work has received support from the Higher Education Authority in Ireland under the PRTLI Cycle 4 programme, in a project Serving Society: Management of Future Communications Networks and Services.

References

1. Hiyama, S., Moritani, Y., Suda, T., Egashera, R., Enomoto, A., Moore, M., Nakano, T.: Mole-cular Communication. In: Proc. NSTI Nanotech Conference and Trade Show, Anaheim, vol. 3, pp. 391–394 (May 2005)
2. Alfano, G., Morandi, D.: On Information Transmission Among Nanomachines. In: 1st International Conference on Nano-Networks (Nano-Net), Lousanne, Switzerland (2006)
3. Benenson, Y., Shapiro, E.: Molecular Computing Machines. In: Dekker Encyclopaedia of Nanoscience and Nanotechnology, pp. 2043–2055. Marcel Dekker, New York (2004)
4. Markevich, N., Hoek, J., Kholodenko, B.: Signalling switches and bistability arising from multisite phosphorylation in protein kinase cascades. J. Cell Biology 164(3), 353 (2004), `http://www.jcb.org/cgi/reprint/164/3/353`
5. Stetter, M., Schurmann, B., Hofstetter, M.: Logical Nano-Computation in Enzyme Reaction Networks. In: Proceedings of BIONETICS 2006, Cavalese, Italy (2006)
6. Sauro, H., Kholodenko, B.: Quantitative analysis of signalling networks. Prog, in Biophysics & Mol. Biology 86, 5–43 (2004)
7. Cooper, G.: The cell, a molecular approach, pp. 545–549, ISBN 0-87893-119-8
8. Komarova, N., Zou, X., Nie, Q., Bardwell, L.: A theoretical framework for specificity in cell signalling. Mol. Syst. Biol. (October 2005)
9. Moritani, Y., Hyama, S., Suda, T.: A Communication Interface Using Vesicles Embedded with Channel Forming Proteins in Molecular Communication. In: Proceedings of BIONETICS 2007, Budapest, Hungary (2007)
10. Nakano, T., Suda, T., Koujin, T., Haraguchi, T., Hiraoka, Y.: Molecular Communication through Gap Junction Channels: System Design, Experiments and Modeling. In: Proceedings of 2nd International Conference on Bio-inspired models of Network, Information, and Computing Systems, Budapest, Hungary (2007)
11. Benenson, Y., Gil, B., Ben-Dor, U., Adar, R., Shapiro, E.: An Autonomous Molecular Computer for Logical Control of Gene Expression. Nature 429, 423–429 (2004)
12. De Silva, A.P., Uchiyama, S.: Molecular Logic and Computing. Nature Nanotechnology 2 (July 2007)
13. Niazof, T., Baron, R., Katz, E., Ioubashevski, O., Willner, I.: Concatenated logic gates using four coupled biocatalysts operating in series. In: PNAS, Niazov, November 14 (2006)
14. Cavalcanti, R., Freitas Jr., A.: Nanorobotics Control Design: A Collective behaviour approach for medicine. IEEE Transaction on NanoBioscience 4(2) (June 2005)
15. Abelson, H., Allen, D., Coore, D., Hanson, C., Homsy, G., Knight, T.F., Nagpal, R., Rauch, E., Sussman, G., Weiss, R.: Amorphous Computing. Communications of the ACM 43(5) (May 2001)
16. Fedichkin, L., Katz, E., Privman, V.: Error Correction and Digitalization Concepts in Biochemical Computing. J. Comput. Theor. Nanosci. 5, 36–43 (2008)
17. Liu, S., Gaudiot, J.: DNA-based Killer Automaton: the Innovative Nanomedicine. In: Proc. NSTI Nanotech Conference and Trade Show, Boston, Massachusetts (May 2006)
18. Valiunas, V., Polosina, Y.Y., Miller, H., Potapova, I.A., Valiuniene, L., Doronin, S., Mathias, R.T., Robinson, R.B., Rosen, M.R., Cohen, I.S., Brink, P.R.: Connexin-specific cell-to-cell transfer of short interfering RNA by gap junctions. J. Physiol. 568(2), 459–468 (2005)
19. Önfelt, B., Nedvetzki, S., Benninger, R.K.P., Purbhoo, M.A., Sowinski, S., Hume, A.N., Seabra, M.C., Neil, M.A.A., French, P.M.W., Davis, D.M.: Structurally Distinct Membrane Nanotubes between Human Macrophages Support Long-Distance Vesicular Traffic or Surfing of Bacteria. FEBS Lett. 581(11), 2194–2201 (2007)