Application of the Hindmarsh-Rose Neural Model in Electronic Circuits

A Thesis Presented
by

Daniel Terence DeBolt

to

The Department of Electrical and Computer Engineering

in partial fulfillment of the requirements for the degree of

Master of Science

in

Electrical Engineering

Northeastern University
Boston, Massachusetts

May 2011
Acknowledgements

I would like to thank my advisor Professor Yong-Bin Kim for his help and support throughout my graduate school experience. My parents, Dan and Linda DeBolt, who instilled a desire to learn about the world and encouraged me to pursue a degree in engineering, without them I would not be where I am today. Finally my loving girlfriend, Cassie Schmitz, without her daily encouragement and support the final revision would have not been possible.

Daniel T. DeBolt
daniel.debolt@gmail.com
May 2011
Abstract

Biomemetic control of robots has been an increasing topic of research over the past decade. This approach allows life-like robust movements by mimicking the control mechanisms observed in nature. Electronic neurons and synapses can be mapped together to make motor programs for walking, swimming, and flying. This thesis focuses on the application of the Hindmarsh-Rose (HR) neural model to replicate biologically meaningful movement in a novel, bio-hybrid micro-robot. The model’s mathematical background is studied and three separate implementations are produced. First, an analog computer method is investigated. Issues solved for this method include scaling for below one volt operation and scaling the model in time for accurate output. A set of tools and a verification board were produced to facilitate a CMOS design. The second method was an implementation of digital neurons on low cost, low pin-count microprocessors. Magnitude scaling of the digital map-based neurons was one technique created for easier processing using only integer variables. An implementation was programmed and proved on a physical demonstration board using 16-bit fixed point variables. The final method was a 65 nm CMOS implementation using an oscillator and an integrator circuit to produce a lower order approximation of the HR model. Benefits include considerably fewer transistors and resistors compared to a previous CMOS circuit. All three serve as a foundation for future research into the area of simple central pattern generators for micro-robots.
# Contents

1 Introduction .................................................. 1
   1.1 Overview of Neural Models ............................................ 1
   1.2 Synapses ....................................................... 3
       1.2.1 Electrical .................................................... 3
       1.2.2 Chemical .................................................... 3
   1.3 Central Pattern Generators .......................................... 3
   1.4 Involved Time Scales ................................................ 4
   1.5 Neuromorphic Robotics .............................................. 5
   1.6 Cyberplasm Microrobot .............................................. 5

2 Hindmarsh-Rose Neuron .............................................. 8
   2.1 Introduction and Biological Background ............................. 8
   2.2 Three Coupled Differential Equations ............................... 8
   2.3 Equation Analysis .................................................. 10
       2.3.1 Equilibrium Analysis & Spiking Mechanism ....................... 10
       2.3.2 Bursting Mechanism ............................................ 12

3 Hindmarsh-Rose Analog Computers .................................... 14
   3.1 Background on Analog Computers .................................... 14
   3.2 Theory of Operation ............................................... 15
   3.3 Magnitude and Time Scaling ........................................ 15
       3.3.1 Magnitude Scaling .......................................... 16
       3.3.2 Time Scaling ............................................... 17
   3.4 Final HR analog computer circuit ................................... 17
       3.4.1 Switched Capacitor Integrators ................................ 19
   3.5 Rescaling of Constants for sub 1V / Single Supply Operation ....... 20
   3.6 Microsoft Excel™ HR calculation tool ................................. 21
   3.7 Discrete Component Implementation .................................. 22
       3.7.1 Input Photodiode – Output LED Characterization ............... 25
   3.8 Limitations ....................................................... 26

4 Digital Neurons .................................................... 29
   4.1 Introduction ....................................................... 29
   4.2 Theory of Operation ............................................... 29
       4.2.1 The Membrane Potential .................................... 30
       4.2.2 Synaptic Connections ........................................ 31
4.2.3 Magnitude Scaling the DTM Neuron .............................................. 32
4.3 Implementation in Hardware ......................................................... 34
  4.3.1 Selected Microcontrollers ....................................................... 34
  4.3.2 Fixed Point Mathematics ....................................................... 35
  4.3.3 DTM neuron Demonstration board ........................................... 36

5 Reduced Complexity Hindmarsh-Rose Circuit .................................. 40
  5.1 Introduction ............................................................................. 40
  5.2 Overview of Operation ............................................................ 41
  5.3 Simulink Model ..................................................................... 42
  5.4 CMOS Design ....................................................................... 44
    5.4.1 Schmitt Trigger ............................................................... 44
    5.4.2 Nonlinear Load ............................................................... 45
    5.4.3 Connection of the first two blocks ..................................... 46
    5.4.4 Adaption Integrator ......................................................... 47
  5.5 CMOS Design Results ............................................................... 48

6 Conclusion .................................................................................... 51
List of Figures

1.1 Robots created at the Northeastern University Marine Science Center. ............... 5
1.2 Diagram of the proposed Cyberplasm micro-robot. ........................................... 6
1.3 Map of the swim pattern generator for the Cyberplasm micro-robot. ................. 7

2.1 Hindmarsh-Rose neuron output for I=2.5. ..................................................... 9
2.2 Graph of the nullclines for the x and y variables. .......................................... 11
2.3 Two Dimensional HR output. ................................................................. 12
2.4 Three Dimensional HR output. ................................................................. 13
2.5 3D plot of the phase plane of the HR equations, I = 2.5. ................................. 13

3.1 Standard operational amplifier integrator circuit. ........................................ 15
3.2 Hindmarsh Rose neuron high level schematic. ............................................. 18
3.3 Switched capacitor integrator circuit. ...................................................... 19
3.4 Screen shot of the HR calculator in Microsoft ExcelTM ................................. 21
3.5 Block diagram of the AD633. ................................................................. 22
3.6 Schematic capture of the analog demonstration board. .................................. 23
3.7 Completed art work of the analog demonstration board. .................................. 24
3.8 Photograph of the completed analog demonstration board. ........................... 24
3.9 Oscilloscope output from the analog demonstration board. ............................ 25
3.10 Comparasion of measured LED performance with datasheet values. ............ 26
3.11 Comparison of several time scaling values. ............................................... 27
3.12 Resistance verse Capacitance for several time scaling factors. ..................... 28

4.1 DTM neuron output and update map ......................................................... 31
4.2 DTM neuron output and bifurcation diagram ............................................ 31
4.3 Magnitude scaled map comparison ........................................................... 33
4.4 Magnitude scaled DTM neuron comparison .............................................. 33
4.5 DTM demonstration board schematic capture. .......................................... 37
4.6 DTM demonstration board art. ................................................................. 38
4.7 Single neuron output from the DTM board ............................................... 39
4.8 Synaptic coupling on the DTM board. ...................................................... 39

5.1 Block diagram of the reduced order neural model. ................................... 41
5.2 A graph of the nonlinear load and the constant output from the current schmitt trigger. ................................................................. 42
5.3 Overview of the Simulink model. ............................................................ 43
5.4 Comparison between the Simulink model and the actual solved HR equations. .... 43
5.5 Circuit diagram of the schmitt trigger. ..................................................... 44
5.6  Circuit diagram of the nonlinear load. ........................................ 46
5.7  Voltage sweep of the first two building blocks. ............................ 47
5.8  Bias circuit for the other building blocks. .................................. 47
5.9  Circuit diagram for the adaption integrator (z variable term). .......... 48
5.10 Waveform output for the reduced order model for several inputs. ....... 50
List of Tables

2.1 Typical Hindmarsh Rose Neural Coefficients ................................. 9
3.1 Resistor Equations for Figure 3.2 .............................................. 18
3.2 Equations for the switched capacitor implementation. ....................... 20
3.3 Rescaled Resistor Equations for Figure 3.2 .................................... 20
4.1 Selected microcontroller comparison chart ........................................ 34
4.2 Values for each fixed point bit position. ........................................ 35
4.3 Examples of fixed point values. .................................................... 35
4.4 Example of multiplying fixed point numbers ................................. 36
Chapter 1

Introduction

Research in the biomimetic robot field has increased over the past decade [1]. State machine control of robots may perform poorly in adverse conditions due to a large number of edge cases, because only a limited number of sequences can be programmed or anticipated for. Mimicking nature is seen as a way to make robots more robust to environmental changes and variations as animals have evolved to adapt perfectly to their environments over thousands of years [2]. Invertebrate nervous systems are simple, easily identifiable and accessible using *in vitro* analysis. This has allowed researchers to make a one-for-one correspondence between an identified neuron or central pattern generator to a behavioral output [3]. Of the many neural models created to replicate neural membrane potential, the Hindmarsh-Rose model was chosen for application in the novel micro-robot, the Cyberplasm. This model was analyzed and implemented using three different electronic architectures.

1.1 Overview of Neural Models

Neurons are a core component of the nervous system. Organized internally similar to other cells, they are specialized for intercellular communication by way of their membrane potential [4]. Neural models attempt to replicate this potential either through direct observation of mechanisms or from high-level behavior analysis. According to reference [5], there are twenty identified types of neural outputs, of which not all models are capable of producing. Two of
these outputs are the most important for central pattern generation: tonic spiking (shown in section 2.3.1) and tonic bursting (shown in section 2.3.2).

The following models are listed in order of their capacity to produce a wider variety of outputs:

**Integrate and Fire** The simplest model, consisting of only a single differential equation. A summation of currents charges (or discharges) a capacitor. When a threshold voltage is reached the capacitors voltage is reset and the process repeats [6]. It is limited in the variety of outputs it can produce, usually only to tonic spiking. Another variation of this model type is the *self-resetting neuron* described by Carver Mead [7] and a similar model in reference [8]. In both, a capacitor is charged until a threshold voltage is reached at which point an output spike is produced and the capacitor voltage is reset.

**Resonate and Fire** Exhibits more dynamical output compared to the previous model but cannot produce tonic bursting. Consists of a single equation (using complex variables) or two simpler equations [9].

**FitzHugh-Nagumo** Consists of two coupled differential equations and based on Van der Pol’s equation of a relaxation oscillator. Cannot output tonic bursting. Created as a simpler representation of excitable-oscillatory outputs compared to the more complex Hodgkin-Huxley model [10].

**Morris-Lecar** Based on voltage oscillations observed in barnacle muscle fibers. An accurate bio-physical model, meaning each variable has a biological connection. Output is capable of spiking but not bursting. Consists of two differential equations, based on a conductance model [11].

**Hindmarsh-Rose** The selected neural model for implementation, description and analysis of this model is given in chapter 2.

**Hodgkin-Huxley** One of the most complex conductance models and considered the most biologically meaningful. Can accurately account for the many ionic currents across the
cellular membrane. Consists of 4 differential equations (expanded to over 7 by later researchers [12]) and has many tunable coefficients [13].

1.2 Synapses

Communication and coordination between neurons is made possible by synapses. When a synapse connects two neurons together the postsynaptic neuron receives the ionic current while the presynaptic neuron is the source. Synapses can be divided into two types: electrical and chemical. Chemical synapses are the more commonly found type, but the electrical type is usually found in the nervous system of all animals [4].

1.2.1 Electrical

Electrical synapses are usually bi-directional but in some cases can operate heavier in one direction than the other, or only in one direction [14]. They are created when the pre and post-synaptic membrane meet and gap junction channels of each align. Ions flow between each neuron proportionally to the potential differential between them. Electrical synapses are quick message carriers when compared to chemical synapses described in the next section.

1.2.2 Chemical

Chemical synapses are more commonly found in biology. Unlike the electrical synapse, there is no direct connection between the pre and postsynaptic neurons. No ionic current flows between neurons, but neurotransmitters secreted by the presynaptic neuron open receptors for specific molecules [4]. These molecules allowed to flow through the receptors have a postsynaptic potential response which either increases (excitatory coupling) or decreases (inhibitory coupling) the membrane potential.

1.3 Central Pattern Generators

Rhythmic behavior is created by central pattern generators (CPG) in both vertebrates and invertebrates. CPGs coordinate motion for walking, swimming, breathing and other motor func-
tions [15]. These patterns are created from networks of neurons and synapses and operate even in the absence of descending commands from the brain or feedback from the senses [14]. CPGs can either control a single muscle, limb or another CPG. For example a CPG controls a leg in an insect, another controller would then pass timing control parameters to each leg CPG to coordinate walking or running [16]. Insects contain the simplest networks and can be easily studied and adapted for robotic use [17].

1.4 Involved Time Scales

Time scales seen in neurobiology are much slower than those seen in most CMOS applications. Burst length times on the order of 0.3–10.0 seconds are common while fast spiking neurons are on the order of 1–10 milliseconds [14] [4]. A capacitor is used to store the state variable in analog designs. The speed at which this capacitor charges and discharges sets the neuron’s overall output time constant. A 1 \( \text{pF} \) capacitor charged by a 10 \( \text{nA} \) current charges to 1 \( \text{V} \) in 0.0001 seconds, much faster than the desired spike frequency. Reference [18] creates a spiking neuron in CMOS at a realistic output frequency of 33 Hz. This was accomplished using subthreshold operation and by locating the state capacitor off-chip. This allows a larger capacitor to be used than what is available (and costly in terms of area) in standard CMOS process.

Many proposed CPG models simply operate too fast to control an electromechanical system such as a micro-robot. For example, a CPG circuit for rhythmic chewing is proposed in [19]. The time between chewing is shown to be 0.2 \( \mu \text{S} \), which is orders of magnitude faster than its biological equivalent due in part to the sizing of the 60 \( \text{pF} \) capacitor. Reference [20] proposes a neuron based on the Izhikevich model [21]. The burst length reported is on the order of a few microseconds, which is much faster than the hundreds of milliseconds of its biological counterpart. This increase in speed is suitable for computational neural networks such as machine vision or learning but operates too quickly for operation of a physical electromechanical device. Capacitor sizing is the main limiting factor for biologically accurate analog CMOS implementations.


caption

Figure 1.1: Robots created at the Northeastern University Marine Science Center.

1 Photo credit NU Marine Science Center and Brian Tucker Bresnahan Photography.
2 More and updated information available at http://www.cyberplasm.net
CHAPTER 1. INTRODUCTION

sensors and muscles. Biological sensors are used for their small area and dynamic sensitivity. For example, chemical sensors created from yeast cells may be sensitive to a number of chemical compounds such as those used in aquatic mines. Artificial muscles, sensitive to blue light, will be grown over an organic LED on a Kapton substrate. The muscles will be coordinated to underlie forward movement by articulating an anatomical tail. The overall micro-robot is to be powered by a chemical battery created using bacteria [26]. The entire system will be tied together with an analog implementation of a biologically realistic nervous system. Multiple electronic CPGs will coordinate the output muscles to create a rhythmic swimming pattern. An overall component diagram is shown in figure 1.2.

Figure 1.2: Diagram of the proposed Cyberplasm micro-robot.³

A map for the electric nervous system is shown in figure 1.3. Connections with circles are inhibitory synapses; triangles are excitatory. Input from chemical sensors coordinate the desired direction while the photoreceptor activates the overall map. Bilaterally pared CPGs (the contra-lateral connections between Lseg and Rseg) create out-of-phase oscillations while excited. Intersegment (ipsi-lateral) connections between Seg1 and Seg2 propagate oscillations from front to rear, creating waves of muscle contractions for forward swimming.

³Figure created by Joseph Ayers at the Northeastern University Marine Science Center.
Figure 1.3: Map of the swim pattern generator for the Cyberplasm micro-robot.\textsuperscript{4}

\textsuperscript{4}Figure created by Joseph Ayers at the Northeastern University Marine Science Center.
Chapter 2

Background of the Hindmarsh-Rose Neural Model

2.1 Introduction and Biological Background

The Morris-Lecar [11] and Hodgkin-Huxley [13] models try to replicate the electrophysiological process of biological neurons [27] based on conductance models with many governing equations and coefficients. The Hindmarsh-Rose (HR) model is based on the global behavior of the neuron and its underlying operation is removed from the actual biological process. Despite being simpler with less governing equations and coefficients, the HR model is accurate to neurons seen in biology and was created to accurately follow the bursting output seen in mollusks [28]. Discrete electronic versions have also been compared to those seen in the crustacean stomatogastric system where a high degree of similarity was observed [29]. Electronic versions have even replaced neurons removed from a central pattern generator of a live lobster and have restored functionality [30].

2.2 Three Coupled Differential Equations

The three coupled equations of the Hindmarsh Rose neural model are shown in (2.1), (2.2) and (2.3). Reference [31] has added a fourth equation, $w$, to the HR equation set that models
extremely slow dynamics (such as calcium exchanges). The fourth equation expands regions where chaotic behavior is present compared to just varying the first three equation coefficients. This fourth equation is not covered in this chapter due to the extra complexity it brings to circuit implementation with little observable benefit to central pattern generation.

\[
\frac{dx}{dt} = ay + bx^2 - cx^3 - dz + I \quad (2.1)
\]

\[
\frac{dy}{dt} = e - f x^2 - y \quad (2.2)
\]

\[
\frac{dz}{dt} = \mu (S(x + h) - z) \quad (2.3)
\]

Table 2.1: Typical Hindmarsh Rose Neural Coefficients

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>e</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>1</td>
<td>0.99</td>
<td>1.01</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>f</th>
<th>μ</th>
<th>S</th>
<th>h</th>
<th>I</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.0128</td>
<td>0.0021</td>
<td>3.966</td>
<td>1.605</td>
<td>0-3.024</td>
</tr>
</tbody>
</table>

Where \(x\) is the membrane potential, \(y\) is the spiking variable (also known as the recovery current) and \(z\) is the bursting variable (also known as the adaption current) [28] [31]. Coefficient \(h\) is the leftmost stable point without adaption. Typical coefficient values are outlined in table 2.1.

Figure 2.1: Hindmarsh-Rose neuron output for \(I=2.5\).
2.3 Equation Analysis

The analysis of the HR equations and similar systems have been a topic of interest in research. With respect to the chaotic operation of circuits, references [32] and [33] give an excellent primer into their operation. The simplest known circuit to exhibit chaotic operation, Chua’s Circuit, is analyzed. Its construction and analysis is similar to that of implementations of the Hodgkin-Huxley neural model [18] [13]. The original two dimensional HR equations were modeled after the Fitzhugh neural model [34]. The third dimension was added later to include triggered bursts in order to more accurately model neurons seen in nature.

2.3.1 Equilibrium Analysis & Spiking Mechanism

Equations (2.1) and (2.2) are similar to the Fitzhugh neural model, except the HR equations have two additional equilibrium points in the nullclines graph [10]. This change gave the two dimensional HR model a more accurate frequency-current relationship [34]. A graph of the $x$ and $y$ nullclines is found by setting equations (2.1) and (2.2) to zero and graphing $y$ with respect to $x$. The two dimensional equations are (2.4) and (2.5). The $z$ variable of the HR equations is assumed to be zero during this analysis as it is purposely set extremely slow compared to the $x$ and $y$ variables. Figure 2.2 shows the graphed nullclines for several values of $I$. Increasing $I$ lowers the $x$ nullcline causing the stable points to merge together then vanish. This drives the system into an unstable region and limit cycle, causing output spiking [35].

\begin{align}
y &= \frac{1}{a} \left( c x^3 - b x^2 - I \right) \\
y &= e - f x^2
\end{align} \tag{2.4, 2.5}
The intersection points can be found by setting (2.4) and (2.5) equal to each other, and then solving equation (2.6) for $x$. The left most result is the resting point and is utilized by (2.3) by setting coefficient $h$ to the negative of this value. Figure 2.3(b) shows the phase plane output of the two dimensional HR neuron during limit cycle spiking. It is transversed clockwise and shows it takes a longer path to get from left to right than right to left on the $x$ variable.

$$\frac{c}{a}x^3 + \frac{f - b}{a}x^2 - e = 0$$  \hspace{1cm} (2.6)
2.3.2 Bursting Mechanism

Neurons in nature do not spike or burst indefinitely and often remain silent until activated [14]. The $z$ variable in the HR equations is what provides this action to the overall HR model. As the input is raised, the stable equilibrium points vanish and cause limit cycling (the spiking action potentials shown in the previous section). Each spike increases the $z$ variable, which in turn raises the $x$ nullcline. This lifting of the nullcline counteracts the lowering of it by the input $I$. This action causes the output spikes to slow down in time, then finally cease. The system has returned to the 3 point equilibrium state. Because the $x$ variable goes low, the $z$ variable in turn falls, allowing the system to return to the single point unstable system and for spiking to resume.
Figure 2.4: Output of the three dimensional HR equations: (a) is the membrane potential ($x$ variable) with $I=1.5$, (b) is the phase plane of the $x$ and $y$ variables.

Figure 2.4 shows the output of the three dimensional HR equations. The phase plane analysis is shown, and is again transversred clockwise. Each semi-circle originates from an individual spike from the overall output burst. The dip observed is the quiet period seen between bursts. Figure 2.5 shows a three dimensional plot of the phase plane with the individual spikes more visible. It is transversed clockwise looking from the top, same as shown before in figure 2.3(b).

Figure 2.5: 3D plot of the phase plane of the HR equations, $I = 2.5$. 
Chapter 3

Implementation of the Hindmarsh-Rose equations using Analog Computers

3.1 Background on Analog Computers

Before the advent of digital computers, analog computers were used to solve many types of mathematical problems. Mechanical analog computers were employed before the creation of transistors. Two meshed gears of different sizes act as gain elements while integration, differentiation, and logarithmic functions were found by mechanical look-up tables. The most well known application was the firing computer used on naval ships which was capable of pivoting a large cannon, while moving, to track a moving target [36]. Electronic analog computers operated in a similar manor, did not suffer from mechanical fatigue, and gave more precise answers. Created from networks of summing circuits, integrators and trim pots for coefficient adjustments, they could be wired for a single purpose or rewired in a lab to solve a variety of simulation problems [37]. Such applications include process control, motor control, and electric grid control [38]. Differential equations have a one-for-one relationship in an analog computer. Reference [31] gives an overview of the implementation in a discrete case but does not expand on scaling factors involved or provide a schematic diagram. Reference [39] covers an implemen-
tation into 0.13 μm CMOS but ignores the biological time scales relevant for robotic operation. Section 1.4 covers the time scales seen in biological neurons.

### 3.2 Theory of Operation

As stated before, analog computers mainly consist of summing junctions and integrators created with operational amplifiers. Trimpots are used to set coefficients from given equations. An operational amplifier RC integrator utilizes the current voltage relationship of a capacitor (Equation 3.1) by placing it in the negative feedback path. The resistor $R$ converts an input voltage to a proportional current as the $V_-$ terminal is considered a virtual ground. Figure 3.1 shows a basic RC integrator while equation 3.2 is its governing equation.

$$I = C \frac{dv}{dt} \quad (3.1)$$

$$V_{out} = -\frac{1}{RC} \int V_{in} \, dt + V_o \quad (3.2)$$

### 3.3 Magnitude and Time Scaling

The output of an analog computer is no longer considered valid if an integrator becomes saturated to the rails voltage. This was a particular challenge to earlier designers as the output of a particular set of equations could be unknown. One way to overcome this was with high
supply voltages: ±50 or ±100 volts was not uncommon [37]. High supply voltages are limited, thus it is not feasible if a model or process deals with higher magnitudes. Magnitude scaling can overcome this limitation. Scaling in time speeds up or slows down the speed at which a set of equations are solved. For example, the HR equations given in 2.2 do not represent biological speed in their output and must be scaled to become relevant. Another example is the modeling of orbital dynamics or other large time scale systems: it would not be practical to have a computer solve the solution at the slower speed observed as opposed to a computer produced faster time scale. Both types of scaling are done with a simple variable substitution.

### 3.3.1 Magnitude Scaling

\[ x = X \cdot x_{ms} \cdot y = Y \cdot y_{ms} \cdot z = Z \cdot z_{ms} \]

Plugging in the previous scaling factor into equations (2.1), (2.2), (2.3) gives:

\[
\frac{d(X \cdot x_{ms})}{dt} = a(Y \cdot y_{ms}) + b(X \cdot x_{ms})^2 - c(X \cdot x_{ms})^3 - d(Z \cdot z_{ms}) + I \tag{3.3}
\]

\[
\frac{d(Y \cdot y_{ms})}{dt} = e - f(X \cdot x_{ms})^2 - (Y \cdot y_{ms}) \tag{3.4}
\]

\[
\frac{d(Z \cdot z_{ms})}{dt} = \mu (S ((X \cdot x_{ms}) + h) - (Z \cdot z_{ms})) \tag{3.5}
\]

Simplifying (3.3), (3.4), (3.5) yields:

\[
\frac{dX}{dt} = a \frac{y_{ms}}{x_{ms}} Y + b \frac{x_{ms}}{x_{ms}} X^2 - c \frac{x_{ms}}{x_{ms}} X^3 - d \frac{z_{ms}}{x_{ms}} Z + \frac{1}{x_{ms}} I \tag{3.6}
\]

\[
\frac{dY}{dt} = e \frac{1}{y_{ms}} - f \frac{x_{ms}^2}{y_{ms}} X^2 - Y \tag{3.7}
\]

\[
\frac{dZ}{dt} = \mu \left( S \left( \frac{x_{ms}}{z_{ms}} X + h \frac{1}{z_{ms}} \right) - Z \right) \tag{3.8}
\]

If each rescale factor is set to one, then (3.6), (3.7), (3.8) become the form of the original equations. Scaling factors larger than one decrease output magnitude; the opposite is true for factors less than one. Choosing appropriate factors is easily done graphically. For example, if ±4 volt supply was required to operate the HR neurons, \( x_{ms} = 1 \), \( y_{ms} = 3 \), \( z_{ms} = 1 \) would work. There is no single answer and is highly dependent on the given constraints of the design.
3.3.2 Time Scaling

\[ T = t \cdot T_s \]

Plugging in the previous scaling factor into equations (3.6), (3.7), (3.8) gives:

\[
\frac{dX}{dT} = \left( \frac{1}{T_s} \right) \left( \frac{y_{ms}}{x_{ms}} Y + b x_{ms} X^2 - c x_{ms}^2 X^3 - d \frac{z_{ms}}{x_{ms}} Z + \frac{1}{x_{ms}} I \right) \tag{3.9}
\]

\[
\frac{dY}{dT} = \left( \frac{1}{T_s} \right) \left( e \frac{1}{y_{ms}} - f \frac{x_{ms}^2}{y_{ms}} X^2 - Y \right) \tag{3.10}
\]

\[
\frac{dZ}{dT} = \left( \frac{1}{T_s} \right) \left( \mu \left( \frac{x_{ms}}{z_{ms}} X + h \frac{1}{z_{ms}} \right) - Z \right) \tag{3.11}
\]

The original equations are left if \( T_s = 1 \). For this particular set of differential equations, \( T_s \) will always be much less than 1. In reference [39], \( T_s = 5E-6 \), which is a trade-off to reduce the size of resistors when the integration capacitors are small (picofarad level). While reference [31] had \( T_s \approx 2.2E-3 \), which gives a much more biological meaningful timescale. Variations around the second value are a good starting point for biologically accurate time scaling of the HR model.

3.4 Final HR analog computer circuit

The final schmatic for a Hindmarsh-Rose neuron using analog integrators is shown in 3.2 and the resistor equations are given in table 3.1. The design process starts with choosing a supply voltage and is followed by scaling the output magnitudes such that the output stays within the supply limits. Next, an appropriate time scale is chosen: \( T_s = 2.2E-3 \) is a common starting point for a biologically meaningful output. Both the \( x \) and \( z \) variable integrators invert the output. The reason the \( y \) variable integrator gives an inverted output is because each side of equation (3.10) is multiplied by -1. Because the standard op-amp integrator is inverting, the sign change is accounted for on the left, but is carried on the right hand side. This is why the constant \( e \) is tied to the negative reference and not to the positive one. Additionally, it is the reason why the \( x^2 \) term is not inverted before the second integrator input as one would assume from looking at the \( y \) variable equation.
Figure 3.2: Hindmarsh Rose neuron high level schematic.

Table 3.1: Resistor Equations for Figure 3.2

<table>
<thead>
<tr>
<th>X Variable</th>
<th>Y Variable</th>
<th>Z Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>$R_{xx1} = \frac{x_{\text{ms}} T_x}{C_x}$</td>
<td>$R_{yy} = \frac{T_y}{C_y}$</td>
<td>$R_{zz} = \frac{T_z}{\mu C_z}$</td>
</tr>
<tr>
<td>$R_{xy} = \frac{x_{\text{ms}} T_y}{y_{\text{ms}} C_x}$</td>
<td>$R_{yx2} = \frac{y_{\text{ms}} T_x}{x_{\text{ms}} C_y}$</td>
<td>$R_{zx} = \frac{z_{\text{ms}} T_y}{\mu S_x C_z}$</td>
</tr>
<tr>
<td>$R_{xx2} = \frac{x_{\text{ms}} T_y}{d z_{\text{ms}} C_x}$</td>
<td>$R_{ye} = \frac{y_{\text{ms}} T_x}{c C_y}$</td>
<td>$R_{zh} = \frac{z_{\text{ms}} T_y}{\mu S_h C_z}$</td>
</tr>
<tr>
<td>$R_{xx3} = \frac{T_x}{b x_{\text{ms}} C_x}$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$R_{xx3} = \frac{T_y}{c x_{\text{ms}} C_x}$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
3.4.1 Switched Capacitor Integrators

Switched capacitor integrators can overcome the intrinsic limit on the sizing of resistors in CMOS [40]. A switched capacitor integrator is shown in figure 3.3. The overall function of the switched capacitor integrator is similar to equation (3.2). The only difference is that resistor \( R \) is replaced by a switched capacitor equivalent which is shown in equation 3.12. The final equation is (3.13) where \( f_{\text{clk}} \) is the switching frequency and \( C_a \) is the switched capacitor value in Farads. Generally, the switching frequency must be 100 times greater than the highest signal frequency [41]. This is because the equivalent resistance is based on the assumption that the switching time is considered negligible. Another advantage of the switched capacitor integrator is that the coefficient is a ratio of capacitance, which can be precise despite manufacturing variations. A disadvantage of this circuit topology is clock feed-through from the switches, which leads to the need for low pass filtering the output of each integrator [40] [41], adding more complexity.

![Switched capacitor integrator circuit.](image)

\[
R = \frac{1}{f_{\text{clk}}C_a} \quad (3.12)
\]

\[
V_{out} = f_{\text{clk}} C_a \int V_{in} dt + V_0 \quad (3.13)
\]

The final equations are shown in table 3.2.
CHAPTER 3. HINDMARSH-ROSE ANALOG COMPUTERS

Table 3.2: Equations for the switched capacitor implementation.

<table>
<thead>
<tr>
<th>X Variable</th>
<th>Y Variable</th>
<th>Z Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_{xi} = \frac{i \cdot C_x}{x_{ms} \cdot T_{clk}}$</td>
<td>$C_{yi} = \frac{C_y}{T_{clk}}$</td>
<td>$C_{zi} = \frac{\mu \cdot C_z}{T_{clk}}$</td>
</tr>
<tr>
<td>$C_{xy} = \frac{a \cdot y_{ms} \cdot C_x}{x_{ms} \cdot T_{clk}}$</td>
<td>$C_{yx} = \frac{f \cdot x_{ms} \cdot C_y}{y_{ms} \cdot T_{clk}}$</td>
<td>$C_{zx} = \frac{\mu \cdot S \cdot C_z}{x_{ms} \cdot T_{clk}}$</td>
</tr>
<tr>
<td>$C_{xz} = \frac{d \cdot z_{ms} \cdot C_z}{x_{ms} \cdot T_{clk}}$</td>
<td>$C_{yz} = \frac{c \cdot C_y}{y_{ms} \cdot T_{clk}}$</td>
<td>$C_{zh} = \frac{\mu \cdot S \cdot h \cdot C_z}{x_{ms} \cdot T_{clk}}$</td>
</tr>
<tr>
<td>$C_{xx2} = \frac{b \cdot x_{ms} \cdot C_x}{T_{clk}}$</td>
<td>$C_{xx3} = \frac{c \cdot x_{ms} \cdot C_y}{T_{clk}}$</td>
<td></td>
</tr>
</tbody>
</table>

3.5 Rescaling of Constants for sub 1V / Single Supply Operation

Constants in equations are instantiated by tying a resistor to ±1V reference in an analog computer. This poses a challenge in supply limited systems and results in extra complexity to create the reference voltages in an integrated circuit. Creating a computer to run from a single supply instead of a split supply means the references are ±1 volt around the common voltage. Applying a similar method shown in section 3.3.1, one can rescale the reference voltage to the supply rails or any arbitrary voltage. For example, we want to instantiate a neuron running from an +8 volt supply with a +4 volt common. This has shifted the output voltage swing to between 0 and 8 volts. Negative constants must be tied to +3 volts while positive constants must be tied to +5 volts. The references can shift to 0 or +8 volts with a rescale factor. Equation (3.14) shows an example for the reference voltage calculation, where $y_{rs}$ is the rescale value and $y_{ms}$ is the current magnitude scaling factor.

$$V_{ref} = V_{com} \pm \frac{y_{rs}}{y_{ms}}$$ (3.14)

Table 3.3: Rescaled Resistor Equations for Figure 3.2

<table>
<thead>
<tr>
<th>X Variable</th>
<th>Y Variable</th>
<th>Z Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>$R_{xi} = \frac{z \cdot T}{e \cdot C_x}$</td>
<td>$R_{ye} = \frac{y \cdot T}{e \cdot C_y}$</td>
<td>$R_{zh} = \frac{z \cdot T}{e \cdot S \cdot h \cdot C_z}$</td>
</tr>
</tbody>
</table>
3.6 Microsoft Excel\textsuperscript{TM} HR calculation tool

To automate the calculation of the many coefficient resistors, a spreadsheet tool was created. Every assigned parameter in the Hindmarsh Rose equations can be varied along with scaling factors, rescaling factors and the multiplier coefficients. The multiplier coefficient is required as most discrete and integrated-circuit analog multipliers have an output scaling factor dictated by their topology. In the generic equation 3.15, the value $\alpha$ is plugged into the spreadsheet.

$$z = \frac{1}{\alpha} (x \cdot y)$$  \hspace{1cm} (3.15)

![Figure 3.4: Screen shot of the HR calculator in Microsoft Excel\textsuperscript{TM}.]
3.7 Discrete Component Implementation

A demonstration board was constructed to verify scaling and simulation results. The TL082 dual operational amplifier made by STMicroelectronics and the AD633 analog multiplier from Analog Devices were used in the design. The TL082 is a mid-end op-amp with low input bias current (typically 20 pA) due to its JFET input stage and a high slew rate of 16 V/µs [42]. Only a handful of companies still make analog multipliers as digital signal processing or application specific chips (such as variable filters) have become more widely accepted and cheaper. The AD633 is low cost (∼$7 individually), has high impedance differential inputs, and has a 10 volt scaled output [43]. Figure 3.5 shows the block diagram of the AD633 and equation 3.16 shows the governing output equation.

\[
W = \frac{(X_1 - X_2)(Y_1 - Y_2)}{10V} + Z
\]  

Figure 3.5: Block diagram of the AD633.\(^1\)

The scaled output of the multiplier must be accounted for when selecting the resistor values to set the coefficients of the HR equations. Section 3.6 shows an entry into the calculation tool to account for this value. Precision capacitors (2% tolerance) and resistors (1% tolerance) were selected to keep coefficient drift low due to process variations. Layout of the circuit was done with display and future expansion in mind. Easy access for external voltage / current inputs and the membrane potential output is done through a terminal block. An on-board input can

\(^1\)Figure is Adapted from [43].
be selected and modulated with a wheel potentiometer. Finally, each variable in the solved HR equations has a test pin point for easy oscilloscope access. Figure 3.6 shows the schematic and figure 3.7 shows the completed board artwork. The autorouter was not used. Each trace was routed by hand to ensure the shortest possible signal paths. Figure 3.8 is a photograph of the completed analog demonstration boards.

Figure 3.6: Schematic capture of the analog demonstration board.
Figure 3.7: Completed art work of the analog demonstration board.

Figure 3.8: Photograph of the completed analog demonstration board.
3.7.1 Input Photodiode – Output LED Characterization

Photodiode Test Platform

The environmental sensors of the Cyberplasm project (Section 1.6) were previously planed to be engineered cells that emit a certain wavelength of light when exposed to a certain chemical compound. The smallest commercially available sensor was required with the Vishay TEMD6200FX01 being chosen. It is in an 0805 package with a footprint of 2 mm × 1.25 mm and a height of 0.85 mm. The peak sensitivity is located at 540 nm which is close to the ~570 nm originally required by the project [44]. To test the sensitivity, the photodiode was placed on a test board and wired to a transimpedence amplifier with selectable gain. Several LEDs with a range of wavelengths close to the outlined specification were to then be used as a controllable input. The requirement for photodiode inputs was the basis of a capacitive feedback transimpedance amplifier work done in [45] by another member of the team. Work was halted as the direction of the Cyberplasm project shifted away from the need of photoreceptors and toward the use of electro-chemical sensors.
Output LED Characterization

Organic LEDs printed with “E-Jet” technology [46] is Cyberplasm project’s current method of approach for activating the synthetic muscles. Earlier work was toward the use of commercially available surface mount LEDs. The ROHM SMLP12BC7T was the smallest available in an 0402 package (1 mm × 0.6 mm × 0.2 mm), had a 465 nm peak output wavelength and a nominal forward voltage of 2.9V @ 5 mA [47]. LEDs have a turn-on point of several volts. A test bench was created to measure the forward voltage and current through several ROHM LEDs using an adjustable current source. The Cyberplasm project is intended to be run with a chemical battery that outputs less than 1V. An accurate current voltage relationship of the LED for cases outside of the specification was required. Figure 3.10 shows the measured relationship between LED current consumption and forward voltage. Even with the switch to organic LEDs, a boost converter will be a future research topic due to their high turn-on voltages [48].

![Figure 3.10: Comparison of measured LED performance with datasheet values.](image)

(a) Figure adapted from [44].

(b) Measured performance from test bench.

3.8 Limitations

A CMOS analog computer implementation is limited by the sizing of the resistor-capacitor combination shown in equation (3.2). Time scaling of the Hindmarsh-Rose equations for biological
realistic output means this RC ratio must be much less than one. Figure 3.11 shows the output for three separate time scaling factors. The resistor $R_{yy}$, shown in table 3.1, depends only on the time scaling factor which illustrates the resistor-capacitor sizing easily. If $T_s = 2.2 \times 10^{-3}$, $R_{yy}$ must be $2.2 \, \text{G}\Omega$ but only $5 \, \text{M}\Omega$ if $T_s = 5 \times 10^{-6}$, it is advantageous from an implementation standpoint to operate faster to reduce the size of the required resistors and capacitors. Figure 3.12 shows the reduction of needed resistor values with an increase in time scaling.

![Figure 3.11: Comparison of several time scaling values.](image-url)
Figure 3.12: Resistance verse Capacitance for several time scaling factors.
Chapter 4

Digital Neurons

4.1 Introduction

Neural models are described by coupled differential equations. Solving these digitally involves numerical integration, which is computationally intensive and limits the number of concurrent neurons that can be instantiated on low end processors. Reference [5] gives an overview of several neural models and outlines the number of floating point operations per 1 mS of neuron operation. For the Hindmarsh-Rose, the author states a low end of 120 operations per millisecond of operation. An alternative to numerical integration is to use discrete time–map based neurons [49]. In depth analysis and application of this digital neuron is the subject of this chapter. Another advantage to using digital neurons is the ease of analysis of large neural maps [49] and the ease of reconfiguration to test many different theories [50]. Large maps of neurons require updating synaptic weights many times to fine tune operation, which is done much easier in code than in a hardware implementation [51].

4.2 Theory of Operation

The discrete time–map based neural model is formed using a two-dimensional look-up function and a second equation which acts similarly to the $z$ variable in the Hindmarsh-Rose model (Section 2.3.2). Because the model is discrete, the output is updated at a certain frequency.
This has to be chosen correctly when implemented in hardware to ensure the output has a biologically meaningful timescale.

### 4.2.1 The Membrane Potential

Despite consisting of a simple update map, the DTM neural model can exhibit a wide variety of neural outputs. An expanded DTM neuron that includes sub-threshold oscillations is shown in reference [52]. The Discrete time–map based neuron consists of two variables: a *fast* ($x$) variable and a *slow* ($y$) variable. (4.1)-(4.3) are the governing equations of the DTM model. Coefficient $\mu$ is usually set to 0.001, similar to the slow variable in the Hindmarsh-Rose model, and $\alpha$ is known as the map control parameter. The update map, equation (4.3), is what sets the limit cycle for spiking. As long as the $y$ variable is not great enough to keep the value of $x$ under 0, limit cycling can occur. The $y$, or slow variable, is the bursting mechanism of the model as it can inhibit spiking, and set by $\mu$. Figure 4.1 shows a bursting neuron and a graph of the map function for a fixed value of $y$. The neuron is in a bursting regime. Furthermore, the output of the $y$ variable is much like the slow variable ($z$) in the HR model. Figure 4.2 gives an example of how $\sigma$ can be varied for several output types. The bifurcation diagram shows how $\alpha$ and $\sigma$ dictate the neuron’s output type.

\[
x_{n+1} = f(x_n, y_n) \tag{4.1}
\]

\[
y_{n+1} = y_n - \mu(x_n + 1) + \mu \sigma \tag{4.2}
\]

\[
f(x, y) = \begin{cases} 
\frac{\alpha}{\alpha - y} + y & x \leq 0 \\
\alpha + y & 0 < x < \alpha + y \\
-1 & x \geq \alpha + y
\end{cases} \tag{4.3}
\]
4.2.2 Synaptic Connections

Connections between map neurons are also discrete in time, their output being updated before the neuron’s output. The coupling type is that of an electrical synapse: current flowing between two neurons. In the coupling equations, $g_{ji}$ is the coupling constant between neuron $j$ and the
to be updated neuron $i$. Neurons cannot be coupled to themselves, meaning $j \neq i$. Positive values of $g$ indicate excitatory coupling while negative values indicate inhibitory coupling. $\beta^e$ and $\sigma^e$ are used to control the individual coupling for each equation and can be usually set to 1.

$$x_{n+1} = f(x_n, y_n + \beta_i) \quad (4.4)$$

$$y_{n+1} = y_n - \mu(x_n + 1) + \mu \sigma_i + \mu \sigma_i \quad (4.5)$$

$$\beta_i = g_{ji}\beta^e(x_{j,n} - x_{i,n}) \quad (4.6)$$

$$\sigma_i = g_{ji}\sigma^e(x_{j,n} - x_{i,n}) \quad (4.7)$$

### 4.2.3 Magnitude Scaling the DTM Neuron

The DTM neural model may be scaled in magnitude much like the equations of the Hindmarsh-Rose model. By introducing a scaling factor, $z$, the update map and the slow variable $y$ can be rescaled. Values greater than 1 increase the output magnitude. This method of scaling can be used to avoid utilizing floating point variables if the scaling is great enough. This is because the round up/down during variable update calculations becomes negligible. The lowest value found for $z$ with acceptable error was 1500. With that scaling value, the $x$ variable ranged from -3000 to 9000 and the $y$ variable ranged from -6000 to 0. These values could easily be represented by a signed 16-bit number (-32,768 to 32,767), while $z = 5400$ was the largest. The updated equations are shown in (4.8), (4.9), (4.10). The $y$ variable does not require direct scaling as it is directly dependent on the magnitude of the $x$ variable. Figure 4.3 shows how the $z$ variable stretches the update map in both directions. Figure 4.4 compares the unscaled DTM neural output to that of a scaled one. There is no change in the output frequency, only in the output magnitude.

$$x_{n+1} = f(x_n, y_n) \quad (4.8)$$

$$y_{n+1} = y_n - \mu(x_n + z) + z \mu \sigma \quad (4.9)$$
Chapter 4. Digital Neurons

\[ f(x, y) = \begin{cases} 
\frac{z^2 \alpha + y}{z - x} & x \leq 0 \\
\frac{z \alpha + y}{z} & 0 < x < z \alpha + y \\
-1 \cdot z & x \geq z \alpha + y 
\end{cases} \] \quad (4.10)

Figure 4.3: Comparison of unscaled and scaled lookup function, \( \alpha = 6 \) and \( y \) is fixed to -3, -30 respectively.

Figure 4.4: Comparison of unscaled and scaled DTM neurons, \( \alpha = 6 \) and \( \sigma = -0.1 \).
4.3 Implementation in Hardware

To confirm the operation of coupled DTM neurons in low cost hardware, a demonstration board was designed and programmed. The board also offered a visual showpiece for simple central pattern generators by the output LEDs or by viewing on an oscilloscope from the output headers from each microcontroller.

4.3.1 Selected Microcontrollers

Two popular 8-bit and 16-bit microcontroller platforms were chosen: the Atmel ATmega88PA [53] and the Microchip PIC24FJ65GA002 [54]. The ATmega has good compiler and library support. The open source GNU gcc compiler targets the AVR (ATtiny/ATmega) and utilizes the open source avr-libc library. The PIC has a free, but low feature compiler available from the manufacturer while full feature compilers are for sale from various suppliers. The advantage the PIC has over the ATmega is its 16 instead of 8-bits, which speeds up the 16-bit math required for running the DTM model. Both have a hardware multiplier which also speeds up mathematical operations compared to artificial multiply and divide commands provided by the compiler. While the compiler can support floating point variables in software, the routines are quite slow and require too much looping. Real time updating would not be possible if using floating point variables. Instead, values were represented using fixed point notation as outlined in the next section.

<table>
<thead>
<tr>
<th>Specification</th>
<th>ATmega88PA</th>
<th>PIC24FJ65GA002</th>
</tr>
</thead>
<tbody>
<tr>
<td>V\text{in}</td>
<td>1.8-5.5</td>
<td>2-3.6</td>
</tr>
<tr>
<td>Word Len.</td>
<td>8-bit</td>
<td>16-bit</td>
</tr>
<tr>
<td>Prog. Memory</td>
<td>8k</td>
<td>64k</td>
</tr>
<tr>
<td>RAM</td>
<td>1k</td>
<td>8k</td>
</tr>
<tr>
<td>Timers</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>PWM Outputs</td>
<td>6</td>
<td>5</td>
</tr>
<tr>
<td>HW Multiplier</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>HW Divider</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>ADC</td>
<td>6/8(pkg. dep.)</td>
<td>10</td>
</tr>
</tbody>
</table>
CHAPTER 4. DIGITAL NEURONS

4.3.2 Fixed Point Mathematics

Fixed point numbers are a simpler representation of fractional numbers using a limited number of bits. Fixed point math is a requirement where processors do not have floating point units, such as cell phones and embedded platforms [55]. Just as the name implies, the decimal point is always fixed and cannot change during runtime in comparison to floating point numbers. Fixed point representations can only store a limited range of values but the arithmetic is extremely quick compared to utilizing floats [56]. A Q4.12 representation was chosen: 4-bits for whole numbers and 12-bits for the fractional part. Table 4.2 show each bits decimal representation for this format. This is because the DTM model’s maximum values are +6 and -5.5 and it also fits into a 16-bit value for easier representation in C. furthermore, programming is easier, as 16-bit integers are already defined by the C standard while using 24-bits, for example, would require more careful tracking of non-equivalent types in C (such as an int and a char), and an increase in overhead on the low power processor. The largest values that can be represented are \( \sim 7.99 \) and \( \sim -8.99 \) while the smallest is \( 1/2^{12} \approx 2.44 \times 10^{-4} \). Table 4.3 shows several decimal values and their binary fixed point representation.

<table>
<thead>
<tr>
<th>Bit Position</th>
<th>15</th>
<th>14</th>
<th>13</th>
<th>12</th>
<th>11</th>
<th>10</th>
<th>9</th>
<th>8</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value</td>
<td>(-2^3)</td>
<td>(2^2)</td>
<td>(2^1)</td>
<td>(2^0)</td>
<td>(\frac{1}{2^7})</td>
<td>(\frac{1}{2^6})</td>
<td>(\frac{1}{2^5})</td>
<td>(\frac{1}{2^4})</td>
<td>(\frac{1}{2^3})</td>
<td>(\frac{1}{2^2})</td>
<td>(\frac{1}{2^1})</td>
<td>(\frac{1}{2^0})</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Decimal</th>
<th>Fixed Point</th>
<th>Binary</th>
<th>Actual Dec. Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>0x1000</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>-1.0</td>
<td>0xf000</td>
<td>-1.0</td>
<td></td>
</tr>
<tr>
<td>0.125</td>
<td>0x0200</td>
<td>0.125</td>
<td></td>
</tr>
<tr>
<td>0.38</td>
<td>0x0614</td>
<td>0.379883</td>
<td></td>
</tr>
<tr>
<td>0.01</td>
<td>0x0029</td>
<td>0.0100098</td>
<td></td>
</tr>
<tr>
<td>-4.374</td>
<td>0xba04</td>
<td>-4.37402</td>
<td></td>
</tr>
<tr>
<td>2.0E-4</td>
<td>0x0001</td>
<td>2.44141E-4</td>
<td></td>
</tr>
<tr>
<td>1.0E-4</td>
<td>0x0000</td>
<td>0.0</td>
<td></td>
</tr>
</tbody>
</table>

The following C code defines are used to multiply and divide fixed point values:

```c
#define N 12
```
```c
#define ONE (int)0x1000
#define NEGATIVEONE (int)0xF000
#define MUL(a,b) (int)(((long)a * (long)b) >> N)
#define DIV(a,b) (((long)a << N) / b)
```

N is the number of bits representing the fractional part of the number. Defines ONE and NEGATIVEONE are used to increase the readability of written code as these values are needed throughout the program. Integer values $a$ and $b$ have to be type casted to a long to avoid overflow when multiplying them together. The result is then shifted by the number of bits of the fractional portion and interpreted as a 16-bit signed fixed point number. Table 4.4 gives an example of the multiply operation for fixed point numbers.

<table>
<thead>
<tr>
<th>Decimal</th>
<th>Fixed Point</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.5</td>
<td>0x1800</td>
</tr>
<tr>
<td>$\times$4.25</td>
<td>$\times$0x4400</td>
</tr>
<tr>
<td></td>
<td>0x6600000 Shift right 12-bits.</td>
</tr>
<tr>
<td>6.375</td>
<td>0x6600 Which match.</td>
</tr>
</tbody>
</table>

### 4.3.3 DTM neuron Demonstration board

Processing and signal generation is performed on a single microcontroller. Each of the microcontroller outputs are low pass filtered, which is required to view the low frequency output when using pulse width modulation as a digital to analog method. Output LEDs are arranged in a pattern to create undulations down an anatomical spine, giving a visual reference to the swim pattern generator [4]. Each output is tied to a header pin for viewing on an oscilloscope or connecting to the output LEDs. Two voltage regulators are included on the board, +5V and +3.3V for the ATmega and the PIC respectively.
CHAPTER 4. DIGITAL NEURONS

Figure 4.5: DTM demonstration board schematic capture.
Software is the same for each platform with the only difference being setup routines. Both use timers for pulse width modulation for the analog output. Another timer is set to produce a software interrupt 500 times a second. This routine calculates each neuron’s updated value and sets the output timers appropriately. The main part of the program consists of setting up each neuron’s coefficients, then a forever loop. All processing is done in the interrupt routine after initialization is completed.
Figure 4.7: Output from the DTM board with $\alpha = 6$ and $\sigma = -0.1$. (b) is a zoomed in view of (a).

Figure 4.8: Comparison of two separate neuron outputs from the DTM board, (a) is where the neurons are uncoupled, (b) shows the same neurons with excitatory coupling.
Chapter 5

Reduced Complexity

Hindmarsh-Rose Circuit

5.1 Introduction

The completed analog integrator implementation uses at least three operational amplifiers, two multipliers and 13 resistors (see figure 3.2). As most multipliers are differential in operation, extra circuitry is needed to condition signals to and from the multiplier blocks [57]. This leads to a standard integrator design that uses a large number of transistors and area and is a poor choice for a vast array on CMOS for robotic operation. A potential solution is to use either simpler or digital models. Another possible solution is to use a reduced model that can still exhibit select dynamics from the higher order model. Reference [35] outlines a CMOS implementation of a burster similar to the Hindmarsh-Rose model in output. Consisting of three independent blocks, the entire circuit uses only 25 transistors and two capacitors. Designed in 1996, this model works, but leaves much to be desired in terms of the number of transistors, output dynamics, and operating supply voltage. Updating to current CMOS technology and improving the output dynamics is the topic of this chapter.
5.2 Overview of Operation

As shown in chapter 2, the two dimensional HR equations enter a limit cycle with an increase in input. The limit cycle spiking can be approximated using a relaxation oscillator [40]. This is accomplished by charging and discharging a capacitor with a voltage input / current output schmitt trigger. With this single block the oscillator output will be a triangle wave; which is not the shape of a single action potential spike seen in models or nature. If the relative output of the hysteresis element is set for slow charging and rapid discharging, a saw-tooth waveform is created. This is an improvement but can be improved with an extra transconductance block; the nonlinear load. This block is always a current sink from the charging capacitor. A third block adds an adaption term and completes the overall model. It acts the same as the third variable of the HR equation; a ‘slow’ variable to limit the emission of spikes and create bursts of them. The overall process is to integrate the difference between the current value of the \( x \) variable and the resting (equilibrium) point. The overall block diagram is shown in figure 5.1. The input is a current, \( I_{in} \), the output is a voltage, \( V_x \). \( I_{nl} \) and \( I_{adp} \) are always current sinking from the capacitor while \( I_{st} \) is either sourcing to or sinking current from the capacitor depending on the output voltage.

![Figure 5.1: Block diagram of the reduced order neural model.](image-url)
5.3 Simulink Model

A Simulink model was created in order to establish a baseline of operation. Operation was based on the previous papers implementation and the Hindmarsh-Rose model. The schmitt trigger points were set to $x = 2$ and 4, with an output of 2.25 and -15 respectively. The equation for the nonlinear load was chosen based on the shape between $x = 0$ and 4, a much steeper slope between 0 and 1.2 than between $x = 1.2$ to 4. The shape is what gives rise to the spike shape of the neural output. Because the nonlinear load is always discharging, the integration capacitor and the schmitt trigger are supplying a constant charging current, both will balance and thus an equilibrium point will be reached. Solving (5.1) for $x$ gives three solutions, with the left most point the equilibrium point, which is $x \approx 1$. The lower part of figure 5.3 is the adaption integrator. The constant is set to 1 because it is the equilibrium point found earlier. The 0.055 term sets the adaption time constant while the 0.020 term models losses in the capacitor due to the current mirror output of the original and updated circuit. Figure 5.4 shows a comparison between the model and solved HR equations with the same inputs. The magnitude and time labels are not included as they have little relevance when modeling for verification only.

\[
\frac{1}{2}x(x - 4)^2 = 2.25
\]  

(5.1)

Figure 5.2: A graph of the nonlinear load and the constant output from the current schmitt trigger.
CHAPTER 5.  REDUCED COMPLEXITY HINDMARSH-ROSE CIRCUIT

Figure 5.3: Overview of the Simulink model.

Figure 5.4: Comparison between the Simulink model and the actual solved HR equations.
5.4 CMOS Design

The CMOS design utilized the 65 nm Predictive Technology Model from ASU [58]. The target voltage was 0.8 volts and minimized the number of components for a single neuron. In the design, many transistors operate in subthreshold / weak inversion. A mosfet operating in the subthreshold region is governed by equation (5.2) for the specific case of $V_{ds} > 2V_T$. Where $V_T = kT/q \approx 26 mV$ at room temperature, $n$ is a fitting parameter (often close to the value of 1) and $I_{o,n}$ is considered the ‘reverse’ current.

$$I = I_{o,n} \frac{w}{l} e^{\frac{V_{gs}-V_{in,n}}{nVT}}$$  \hspace{1cm} (5.2)

5.4.1 Schmitt Trigger

The schmitt trigger is formed with five transistors. Four of them act as inverters with the fifth acting as a current source when the output is low [59]. This current source (transistor $M_5$) is the reason for the hysteresis of the overall circuit. It increases the current through $M_2$ when active thus transistor $M_1$ must conduct more in order for the inverter to switch [40]. The switching points are determined by analysis of the inverter with or with out the current source active. The overall circuit is shown in figure 5.5.

![Schmitt Trigger Circuit Diagram](image)

Figure 5.5: Circuit diagram of the schmitt trigger.

The switching point from output high-to-low ($V_{10}$) is found as follows. Assume the output is high; the center point, $A$, must be low and $V_{in}$ must be above the switching point. The current
source formed by $M_5$ is not active. The switching point is determined by the inverter formed by $M_1$ and $M_2$. This point is found by solving equation (5.3) for $V_{10}$.

$$I_{o,n}\frac{w_2}{l_2}e^{\frac{V_{10}-V_{th,n}}{nV_T}} = I_{o,p}\frac{w_1}{l_1}e^{\frac{V_{dd}-V_{10}-V_{th,p}}{nV_T}}$$  \(5.3\)

$$V_{10} = V_{th,n} - V_{th,p} + V_{dd} + n V_T \ln \left( \frac{I_{o,p}(w_1/l_1)}{I_{o,n}(w_2/l_2)} \right)$$  \(5.4\)

The switching point from low to high is found in a similar manner. Assume the output is low; point $A$ is high and the input must be below the low-to-high ($V_{01}$) switching point. Because the current source is constant, we can simply add an intermediate switching point to the value of $V_{10}$. This intermediate point ($V_{sp}$) is found by relating the current source formed by $M_5$ and the point where transistor $M_2$ is balanced. This process is shown in equations (5.5) and (5.6).

$$I_{o,n}\frac{w_2}{l_2}e^{\frac{V_{sp}-V_{th,n}}{nV_T}} = K_p\frac{w_5}{l_5} (V_{dd} - V_{th,p})^2$$  \(5.5\)

$$V_{sp} = n V_T \ln \left( \frac{K_p(w_5/l_5)}{I_{o,n}(w_2/l_2)}(V_{dd} - V_{th,p})^2 \right) + V_{th,n}$$  \(5.6\)

The overall switching point can be found using:

$$V_{01} = V_{10} + V_{sp}$$  \(5.7\)

The output of the schmitt trigger is converted to a current by using transistors $M_6$, $M_7$ as switches and $M_8$, $M_9$ as current sources. Voltages $V_{bias,n}$ and $V_{bias,p}$ are biases created in another circuit.

### 5.4.2 Nonlinear Load

The nonlinear load should act similarly to that in figure 5.2. Figure 5.6 shows the completed circuit, the input is a voltage, the output a current, both at the same point. When the input voltage is low, transistor $M_1$ is fully on, but the output current is still low. This is because the drain-source voltage of transistor $M_3$ is below its saturation voltage, an increase in the input voltage (up to the saturation voltage) will increase the current output. Meanwhile, the increasing input voltage is actually decreasing the gate-source voltage of transistor $M_1$, causing
the output current to fall. The aspect ratio of $M_1$ sets the current flowing to the mirror formed by transistors $M_2$ and $M_3$. Transistor $M_1$ acts as a variable resistor when in the triode region of operation. Resistors $R_a$ and $R_b$ are used to linearize the mosfet resistor by adding half of the drain-source voltage to the gate [41]. These resistors must be large to limit current consumption, their values are not critical but should closely match. Pinch resistors would be a good candidate for a CMOS implementation [60].

Figure 5.6: Circuit diagram of the nonlinear load.

5.4.3 Connection of the first two blocks

It is the shape of this nonlinear load that gives the overall shape to each spike. As the voltage across the capacitor increases the nonlinear loads sinks less current. These two blocks are a close approximation of the first two variables of the Hindmarsh-Rose equations. The schmitt trigger block was set to switch at 0.3 and 0.6 volts, with a current output of 3.5 and -10 $\mu$A respectively. The nonlinear load peaks at 3.8 $\mu$A @ 0.16 volts and its left most intersection point is at 0.11 volts. Figure 5.7 is a voltage sweep of the first two blocks and shows the current/voltage relationship for each. The dotted line is the high to low switching sweep, while the dot-dash line is the low to high switching sweep.
Figure 5.7: Voltage sweep of the first two building blocks. The left most intersection is the equilibrium point.

Figure 5.8: Bias circuit for the other building blocks.

Figure 5.8 shows the bias circuit used in each building block of the entire circuit. Two current mirrors create n-type and p-type transistor bias voltages. The current element $I_{\text{bias}}$ in an actual implementation could be a band-gap referenced or peaking current source [60]. For simulation simplicity of this circuit, an ideal current source was used.

5.4.4 Adaption Integrator

The adaption integrator is based on a transconductance amplifier. The current output is proportional to the difference between the two inputs [61]. The capacitor, $C_z$, is then charged or
discharged by the current from the amplifier creating the output voltage $V_z$. A capacitors current is proportional to the time derivative of the voltage across it; the circuit is an integrator [7]. The adaption subtraction current term is created by the current mirror formed by $M_6$ and $M_7$. The time constant of the overall integrator is set by the ratio of transistor $M_5$ and the size of $C_z$.

![Circuit diagram for the adaption integrator (z variable term).](image)

5.5 CMOS Design Results

The output from the reduced order model has similar dynamics to that seen in the solved equations of the Hindmarsh-Rose model. Capacitors were set small for simpler simulation, 10 pF for $C$ and 50 pF for $C_z$, which is why the output time scale is small. Increasing the size of these capacitors increases the output timescale and would have to be located off-chip for longer timescales. With no input, the model’s output is low; with an increasing input, the equilibrium point vanishes and the model outputs tonic bursting. If the input is above a certain threshold, the adaption cannot compensate enough to cease bursting, the output becomes a train of spikes.

Future work for this design includes the layout of this circuit in CMOS or the translation of the design to printed transistors using “E-Jet” technology. Also a synapse circuit must be designed to facilitate coordination between neurons for the creation of central pattern generators. Finally an improved nonlinear circuit operating at a few nano-amps would improve overall
operation and the supply voltage can be lowered; 0.8 volts was chosen, as it was at the time, the output of the chemical battery of the Cyberplasm micro-robot.
(a) $I_{in} = 0.0 \ \mu A$

(b) $I_{in} = 0.5 \ \mu A$

(c) $I_{in} = 1.0 \ \mu A$

(d) $I_{in} = 1.5 \ \mu A$

(e) $I_{in} = 2.0 \ \mu A$

(f) $I_{in} = 2.75 \ \mu A$

Figure 5.10: Waveform output for the reduced order model for several inputs.
Chapter 6

Conclusion

The application of neural models to form central pattern generators allows robots to have life-like movement. Most complex neural models are based on conductance models which require the implementation of a large number of circuits but give deep insight into the individual dynamics of the entire neuron. However, the Hindmarsh-Rose model, based on global dynamics, has the benefit of being relatively simple yet able to produce a wide variety of outputs that closely follow those observed in nature. This model serves as a building block for central pattern generators (CPGs) when combined with synapses. When coupled in certain configurations, many different motor programs can be created, such as walking and swimming. Three methods to implement the HR model were investigated. Analog computers can accurately solve differential equations in real-time. Previous implementations ignore the time scales relevant for use in robotics and operated from high supply voltages. Scaling in magnitude, time and references were investigated to facilitate the creation in nano-meter CMOS. Tools to calculate resistor values from scaling parameters and a reference board were created. The second method utilized discrete time – map based neurons, a mathematical model of neural dynamics not derived from differential equations. They have been applied in previous robots built at the Northeastern University Marine Science Center. Work focused on finding the smallest application processor and investigated the scaling of the model’s magnitude or utilizing fixed point variables to allow for simpler integer calculations. Verification was done by producing a demonstration board and by programming physical processors rather than relying on simulations. The final implementa-
tion method was a reduced complexity model. By reducing the first two variables to a single oscillator, and using a simple adaption integrator, an approximation of the HR equations could be created using less than 25 transistors. It was simulated in 65 $nm$ CMOS using a 0.8 volt supply. Each method of implementation provides a framework for possible future research to create a CPG network for micro-robots.
Bibliography


