5-31-1992

A computer implementation of a multi-neuron expandable network

Mary Ellen Ellen Aleksza
New Jersey Institute of Technology

Copyright Warning & Restrictions

The copyright law of the United States (Title 17, United States Code) governs the making of photocopies or other reproductions of copyrighted material.

Under certain conditions specified in the law, libraries and archives are authorized to furnish a photocopy or other reproduction. One of these specified conditions is that the photocopy or reproduction is not to be "used for any purpose other than private study, scholarship, or research." If a, user makes a request for, or later uses, a photocopy or reproduction for purposes in excess of "fair use" that user may be liable for copyright infringement,

This institution reserves the right to refuse to accept a copying order if, in its judgment, fulfillment of the order would involve violation of copyright law.

Please Note: The author retains the copyright while the New Jersey Institute of Technology reserves the right to distribute this thesis or dissertation

Printing note: If you do not wish to print this page, then select "Pages from: first page # to: last page #" on the print dialog screen

ABSTRACT
A Computer Implementation of a Multi-Neuron Expandable Network

by
Mary Ellen Aleksza

The primary goal of this thesis is to develop a small, expandable neural network that oscillates when presented with an appropriate stimulus. This model will be used in the future as a base to create larger networks. Future studies of these larger networks may be used in the simulation of neuronal activity such biological oscillators.

This thesis encompasses a study of neuron activity, neural networks, neuron models and biological oscillators. The focus of the thesis is the modification and expansion of a single neuron simulation into a simulation of an interconnected multi-neuron oscillatory network.

A single neuron model based on the work of Hodgkin-Huxley is used to develop a simulation of a generalized, expandable, 3X3 (9 neuron) interconnected network.

The model is general enough to be expanded, include multiple neurons (inhibitory and excitatory) with multiple inputs and outputs, and be able to handle any set of random interconnections.

A COMPUTER IMPLEMENTATION OF A MULTI-NEURON EXPANDABLE NETWORK

by
Mary Ellen Aleksza

This Thesis
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfilment of the Requirements for the Degree
of Master of Science in Biomedical Engineering
May 1992

APPROVAL PAGE
A Computer Implementation of a Multi-Neuron Expandable Network

by
Mary Ellen Aleksza

4/27/92

Dr. S. Reisman, Thesis Adviser
Professor and Associate Chairman of Graduate Studies, Electrical and Computer Engineering Department

4/29/92

Dr. D. Kristol, Committee Member
Professor of Chemistry and Director, Biomedical Engineering Program

4/27/92

Dr. J. Carpinelli, Committee Member
Assistant Professor Electrical and Computer Engineering Department

BIOGRAPHICAL SKETCH

Author: Mary Ellen Aleksza

Degree: Master of Science in Biomedical Engineering

Date: May, 1992

Date of Birth:

Place of Birth:

Undergraduate and Graduate Education

* Master of Science in Biomedical Engineering, New Jersey Institute of Technology, Newark, NJ, 1992
* Bachelor of Science in Biomedical Engineering, Milwaukee School of Engineering, Milwaukee, WI, 1988
* Associate of Science in Electrical Engineering, Middlesex County College, Edison, NJ, 1986
* Associate of Applied Science in Electrical Technology, Middlesex County College, Edison, NJ, 1986

Major: Biomedical Engineering

Presentations and Publications:

Aleksza, Mary Ellen. "Computer Implementation of a Multi-Neuron Expandable Network." Presented in Kingston, Rhode Island, March, 1992, at the Eighteenth Annual I.E.E.E. Northeast Bioengineering Conference. Kingston, Rhode Island: 1992.

Hause, L. L., F. Ganyon and M. E. Aleksza. "Impedance Measurements During Simulated Blood Flow." Presented in Seattle, Washington, November, 1989, at the I.E.E.E. Engineering in Medicine and Biology Society Eleventh Annual International Conference. Seattle, Washington: 1989.

This thesis is dedicated

In loving memory of:

Stephen Aleksza

To give up would have been

My only true failure!

I stuck it out for you!

ACKNOWLEDGMENT

The author wishes to express her sincere gratitude to her adviser Dr. S. S. Reisman, for his never ending patience, tolerance, guidance and support throughout this thesis.

Special thanks to Dr. D. Kristol and Dr. J. Carpinelli for serving as members of the committee.

The author would like to thank Dr. W. Tapp for the basic instructions and concepts he provided.

The author would also like to acknowledge the Department of Veteran's Affairs Medical Center, East Orange for the financially supporting this thesis.

And finally, a thank you to the author's husband, family and friends for enduring the wait.

TABLE OF CONTENTS

TABLE OF CONTENTS (continued)

Page

LIST OF TABLES

Table **Page**

LIST OF FIGURES

CHAPTER 1
BACKGROUND

1.1 Introduction

This chapter provides an overview of neural networks and physiological theory of neurons. A section of this chapter is dedicated to the Hodgkin-Huxley neuron model. The simulations developed in this thesis are based on this model. The final section of this chapter discusses a computer implementation of the Hodgkin-Huxley model (1).

1.2 Neuron Physiology (3)

The nervous system is the body's control center and communications network responsible for three basic functions: sensory, integrative and motor. This system is comprised of two major cell groups, neurons and neuroglia. Neuroglial cells are the support and connecting tissue around the neurons. Neurons are the functional unit of the nervous system. These specialized cells are used to convert stimuli to nerve impulses. The neuron cell itself is divided into three sections: the dendrites, for incoming stimuli, the cell body, for processing and the axon, for outgoing stimuli.

The electrical characteristics of a neuron are based on the membrane potential. The membrane potential is caused by the differences between the inside and the outside charges across the membrane. This difference causes a potential difference across the membrane (4).

The charges are caused by unequal concentrations of sodium and potassium ions on opposite sides of the membrane. Also, there are large negatively charged non-diffusible ions trapped within the membrane that facilitate this ion concentration difference. An electric field along with a drift current are products of the concentration differences (4). The resting potential of a neuron is approximately 70 mV with respect to the inside of the cell. A neuron in this state, i.e. not producing impulses, is known as polarized.

A stimulus is any condition within the environment that would cause alterations in the membrane potential. Stimuli may have an inhibitory or excitatory influence on the membrane. A neuron's ability to respond to stimuli and convert it to a nerve impulse is called excitability. Neurons have inherent thresholds that must be exceeded in order to produce a response from a stimulus. A stimulus strong enough to produce an action potential is call a threshold stimulus. Sub-threshold stimuli are those that do not produce an action potential. A series of sub-threshold stimuli have a summing effect and the total, over a given period of time, may produce an action potential.

The membrane becomes permeable to sodium ions when a stimulus or a combination of stimuli is strong enough to exceed the threshold potential (4). This causes the membrane to depolarize, initiating an action potential.

The inside membrane potential begins to increase from -70 mV to 0 mV until the membrane potential is reversed. After this, the inside membrane potential continues to increase to about 30 mV, at which time the membrane permeability changes towards potassium and the membrane repolarizes. The depolarization/repolarization "wave" self-propagates along the outside of the membrane. This action is then transferred via an electro-chemical process to another neuron or muscle. This "stimulus" will either have an excitatory (promote firing) or inhibitory (suppresses firing) characteristic on the next neuron/muscle. The speed of the impulse is dependent upon the temperature and the diameter of the fiber.

A neuron can not generate another action potential until the cell has repolarized. This period of time is called the refractory period. During the absolute refractory period, the cell can not regenerate an action potential regardless of the amplitude of the stimulus present. During the relative refractory period, the cell will produce an action potential only if the amplitude of the stimulus is greater than the normal value.

Neurons are organized in definite patterns called neuron pools. The pools can consist of thousands and/or millions of neurons. The pools are also arranged in organized combinations of different patterns. Neurons in series (one neuron directly affects one neuron) is the simplest pattern. The patterns may be diverging (one

neuron affecting many) or converging (many affecting one). A diverging neuron pattern that eventually converges is known as a parallel network. A reverberating pattern is one that has feedback.

1.3 Introduction to Neural Networks (5)

Neural networks are models that simulate biological neurons. These models are studied because of the possibility of achieving human-like performance from a computer simulation for systems such as biological oscillators. The neural net concept consists of a mass parallel network made up of interconnected computational elements (6). The network is very dependent upon how the individual neurons are interconnected. The connections are usually a set of variable weighted links. The weights control the strength of the incoming signal to the network. The basic unit of a neural network consists of a weighted sum that is compared to a threshold value to see if a response is to be generated.

1.4 Hodgkin-Huxley Neuron Model

The Hodgkin and Huxley model (HH) is a widely known and accepted model of a neuron. This model was generated from data obtained from experiments that were performed on a giant squid axon. The results from the experiments lead to the analogy between the electrical behavior of the membranes and a parallel resistive-capacitive network (1),

(2), (4). The network is shown below in Figure 1.1.

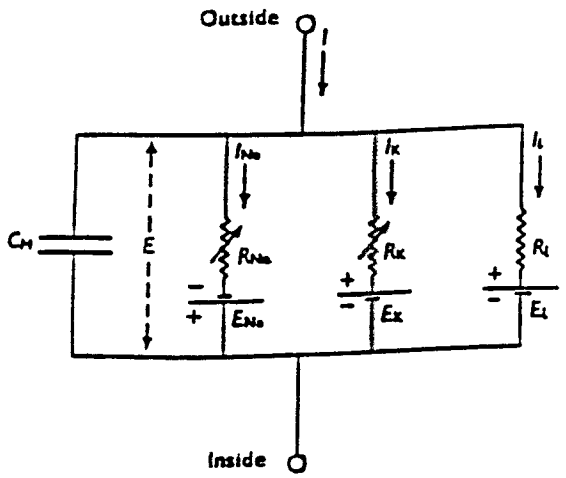

Figure 1.1: Hodgkin and Huxley Electrical Circuit Equivalent of a Neuron Membrane (1).

Ionic current is the current caused by the movement of ions through the resistances. Ina and Ik are currents respectively associated with the sodium and potassium ions. A small leakage current (Il) is caused by chloride and other ions. By using Kirchoff's current laws, the total ionic current is (7): (All equations are adapted from the Hodgkin and Huxley reference (1))

$$Ii = Ina + Ik + Il \qquad (1.0)$$

where $$Ina = gna * (E - Ea) \qquad (1.1)$$

$$Ik = gk * (E - Ek) \qquad (1.2)$$

and $$Il = gl * (E - El) \qquad (1.3)$$

The conductances are gna (for sodium), gk (for potassium) and gl (for the leakage). The equilibrium potentials are Ea (for sodium), Ek (for potassium) and El (the potential at which the leakage current is zero). E is the membrane potential.

The total membrane current (I) (the sum of the capacitor and ionic currents) is given by:

$$I = Cm\ dV/dt + Ii \qquad (1.4)$$

Cm is the membrane capacitance per unit area (V/cm^3). [^ is used to signify something is raised to a power] V is the displacement of the membrane potential from its resting value in millivolts. Ii is the initial value of the capacitor current. The time variable is t.

The equations for the ionic currents can be modified to reflect potential measured directly as displacements from the resting potentials versus the absolute value of the resting potentials. These equations for sodium, potassium and leakage respectively are:

$$Ina = gna * (V - Vna) \qquad (1.5)$$

$$\text{where} \quad V = E - Er \qquad (1.6)$$

$$\text{and} \quad Vna = Ena - Er \qquad (1.7)$$

$$Ik = gk * (V - Vk) \qquad (1.8)$$

$$\text{where} \quad Vk = Ek - Er \qquad (1.9)$$

$$Il = gl * (V - Vl) \qquad (1.10)$$

$$\text{where} \quad Vl = El - Er \qquad (1.11)$$

Er is the resting potential.

Through experimentation, Hodgkin & Huxley found that Ena, Ek, Ei, Cm and gl may be taken as constants, but gna and gk are functions of time and membrane potential (5). Thus, the potassium conductance is given by:

$$gk = gk' * n^4 \qquad (1.12)$$

"gk'" is a constant (mS/cm^3). "n" is a dimensionless

variable between 0 and 1 and is obtained by:

$$dn/dt = an\ (1 - n) - Bn * n \qquad (1.13)$$

The rate constant of potassium ions being transferred from the outside to the inside of the cell (an) is:

$$an = 0.01 * (V+10) / [\exp((V+10)/10) - 1 \qquad (1.14)$$

where $n = an / (an + Bn)$ (1.15)

and $= 1 / (an + Bn)$ (1.16)

The rate constant for potassium ions being transferred from the inside to the outside of the cell (Bn) is:

$$Bn = -0.125 * \exp(V/80) \qquad (1.17)$$

The Sodium Conductance is given by:

$$gna = m^3 * h * gna' \qquad (1.18)$$

"gna'" is a constant (mS/cm^2). "h" is the proportion of inactivating molecules outside and is obtained by:

$$dh/dt = ah * (1-h) - Bh * h \qquad (1.19)$$

The proportion of activating molecules inside the cell is m. The m values are obtained by:

$$dm/dt = am\ (1-m) - Bm \qquad (1.20)$$

The rate constant for sodium ions being transferred from the outside to the inside of the cell (am) is:

$$am = 0.1*(V+25) / (\exp((V+25)/10) - 1 \qquad (1.21)$$

The rate constant of sodium ions being transferred from the inside to the outside of the cell (Bm) is given as:

$$Bm = 4 * \exp(V / 18) \qquad (1.22)$$

The rate constants for the inactivation process are (ah &

Bh):

$$ah = 0.07 * \exp(V / 20) \quad (1.23)$$

$$Bh = 1 / (\exp\{[V+30/10) + 1\} \quad (1.24)$$

The total membrane current (I) as a function of time and voltage is:

$$I = Cm * dV/dt + gk' * n^4 * (V-Vk) + gna' * m^3 * h * (V-Vna) + gl' * (V-Vl) \quad (1.25)$$

This final equation is the basis of the simulation.

1.5 Computer Simulation of the Hodgkin-Huxley Model

1.5.1 Introduction

The computer simulation for an axon action potential is based on a HH computer model implementation (2). The simulation solves for transmembrane potentials, major ionic currents and for the membrane conductances to sodium and potassium. The model is initiated by using the amplitude and duration of two electrical stimuli separated by a time delay and is developed for ease of documentation but not for the minimization of run time.

The simulation determines the membrane conductances as voltage and time dependent variables described by non-linear differential equations. The membrane currents are obtained by using the conductances and a "driving force." The net rate at which the charged particles are crossing the membrane is equal to the sum of the inward and outward currents and the input stimulus (if any.) The membrane potential is updated according to the net charge which is

transferred during the finite computation interval.

1.5.2 Outline of Computer Simulation

The simulation is divided into seven sections for ease of documentation. The first section contains the initialization of variables, the input stimulus set up and the iteration loop. The program is set so the total current flowing during a time increment is used to calculate the new membrane voltage.

The second section is a subroutine that calculates the initial membrane conductances and currents. The values are determined from the dimensionless variables computed from the initial membrane potentials and the electrochemical driving force.

The third section is a subroutine that calculates the rate constants for the membrane voltage. The conductances are assumed to respond instantly to the voltage.

The fourth section is a subroutine that calculates the new conductances. Rectangular integration approximations are used.

The fifth section is a subroutine that calculates the currents during a time interval. The input stimulus, if any, is taken into consideration in this section. The stimulus is added directly to the total current.

The sixth section is a subroutine that updates the membrane voltage. The model accounts for both the voltage expressed as a displacement from the resting value (V) and

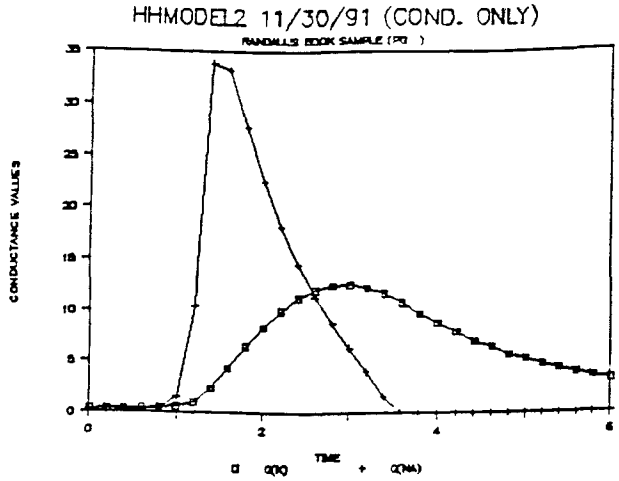

FIGURE 1.2.A

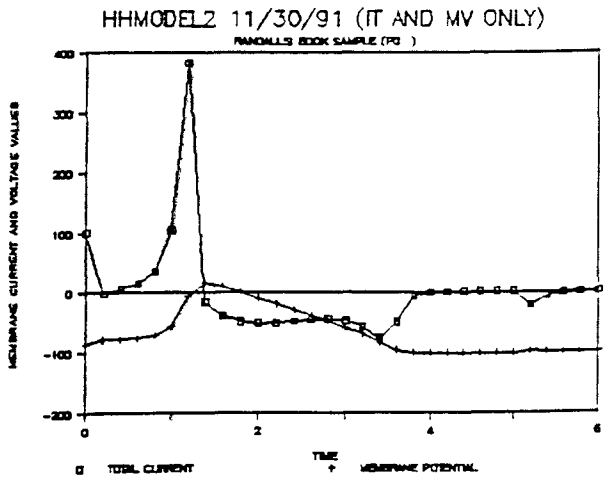

FIGURE 1.2.B

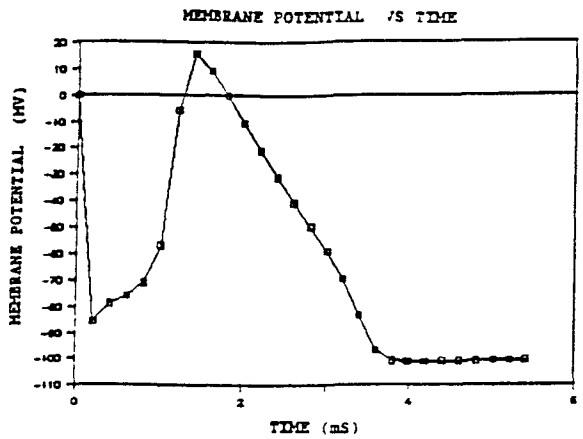

FIGURE 1.2.C

FIGURE 1.2: Sample Run From the Randall Model (8): Values for Stimulus 1 is 100 @ .1, Stimulus 2 is 100 @ .1 and a delay of 5ms. A: Conductances; B: Total Current and Membrane Potential; C: Action potential only.

the absolute voltage (MV). V is the conventional voltage presented by Hodgkin & Huxley. Randall arbitrarily chose a resting membrane potential of -90 mV. This value can range from -70 to -90 mV.

The seventh section is a subroutine that is used to print/plot the output. The program is designed to produce graphic output to the monitor. The program is listed in Appendix A under the HHMODEL series with the name HHMODEL1 and HHMODEL2. The output for HHMODEL1 is not given because it is coded for a graphics display only. The model was adapted for a printed output (HHMODEL2) and is listed in Appendix A. Sample output for HHMODEL2 is shown in FIGURE 1.2, on the facing page.

The HHMODEL1 model was used as a means to became familiar with the capabilities of the model. The graphics display made it easier to see how the amplitude, duration and delay values affect the response of the model.

CHAPTER 2
SCOPE

The primary goal of this thesis is to develop a small, expandable neural network that would oscillate when presented with an appropriate stimulus. This model would then be used in the future as a base to create a larger network. Future studies of this larger network may show that it is a viable option to be used to simulate neuronal activity such as synchronous bursting and biological oscillators.

This thesis encompasses a study of neuron activity, neural networks, neuron models and biological oscillators. The focus of the thesis is the modification and expansion of a single neuron simulation into a simulation of an interconnected multi-neuron oscillatory network.

This thesis demonstrates the development of a single neuron model simulation to a generalized, expandable, 3X3 interconnected network. The neuron simulation is based on the Hodgkin-Huxley model (1).

The final model is a multi-neuron (3X3, or 9 neurons) network. Each neuron has multiple inputs and outputs. These neurons can be either inhibitory or excitatory. The model is general enough to be expandable. The model should can handle any set of random interconnections.

CHAPTER 3
MODEL DEVELOPMENT

3.1 General Model and Simulation Concepts

The simulations are written in BASIC (ProDos 3.3). The programs are run on an Apple 2E enhanced personal computer with extended memory.

The development of the neuron pool starts with a single neuron simulation and continues to an expandable 3X3 interconnected network with multi-input/output capabilities. It is assumed that once a basic multi-neuron network is developed, it can be expanded to any size network.

The HH computer simulation by Randall (described in Section 1.5.2) is used as the starting block for the network. All the assumptions made by Hodgkin & Huxley and Randall are kept. The Randall simulation was generalized for ease of expansion. The computational sections remained the same except for the code used for the addition of the stimulus values. The program is modified so that any time the simulation time falls between a chosen delay and duration time, the appropriate stimulus would be added.

All stimuli values (amplitude, duration and delay) are chosen by the user. These values remain constant throughout the simulation but have to be continuously adjusted to coincide with the simulation time. "Time adjustment" code is required so that the appropriate

neurons are stimulated after the respective neuron fires. [Example: For a given stimulus, the duration chosen is 0.2 ms and the delay chosen is 2.0 ms. If an action potential occurs at 5 ms for a particular neuron, the stimulus will reach the next neuron at 7.0 ms and will last until 7.2 ms. The program does not calculate initial stimuli that start at 0.0 ms. (Inherent to the Randall program.) A small initial delay of 0.04 ms (or one simulation time interval) is arbitrarily chosen to compensate for this.

The stimulus values are set equal to dummy variables at the start of the program. The values are transferred to working variables after the appropriate neuron fires. This prevents the stimuli from being accounted for prematurely.

The models use predominately large excitatory values and times to ensure that the neurons fire. Negative amplitudes are used to indicate an inhibitory stimulus.

3.2 Neuron Model and Simulation

3.2.1 Introduction

The goal of this thesis is to develop a oscillatory, multi-neuron interconnected network simulation. The Randall simulation (discussed in Section 1.5) accommodated a single neuron with two input stimuli. The Randall simulation had to be adapted to accommodate for multiple neurons with multiple input stimuli. The program had to be generalized for ease of expansion. This section

describes how a neuron is simulated.

3.2.2 Simulation of the Neuron

The neuron is described by three characteristics: the conductances (Na, K and leakage), the ionic currents (Na, K and leakage) and the membrane potential (2). The associated equations are outlined in Sections 1.4 and 1.5. This computational group of equations predominantly remain the same throughout the simulation development. The stimulus addition sequence (see Appendix A.1 lines 4070-4100) are modified to account for multiple stimuli that may be present at any given time as described in Section 3.1. A stimulus will be added as long as the simulation time is greater than the associated delay time but less then the associated duration time. This group is a major subroutine in the simulation as outlined in Section 3.5. It is generalized so that no matter which neuron is being activated the same code is called. Matrices keep track of which neuron and associated stimulus is being calculated. The matrix assignments for the 2X2 and the 3X3 model are shown in TABLES 3.1 and 3.2.

3.2.3 Neuron Interconnections

The interconnection pattern for the network is chosen by the user prior to running the simulation. (Example: FIGURES 3.2, 3.3 and 3.4) The simulation is developed based on which neurons are affected by the "firing" of a particular neuron. (i.e. What effect does the firing of N1 have on the rest of the pool?)

Stimulus values are used for connecting the neuron. The values are triggered by an action potential of another neuron. These values are chosen by the user. An arbitrary threshold value is chosen to signify when an action potential has occurred. The membrane potential is compared to this value and a stimulus is activated if the threshold is exceeded. An arbitrary threshold value of 0.0 mV was chosen.

The stimuli that are used for the interconnections must be adjusted to the simulation time during the simulation run as described in Sections 3.1 and 3.2.2. A flag is used to prevent the program from recalculating the updated stimulus values during an action potential.

The flag used to prevent the program from recalculating the time adjustment values during the action potential period must be reset so that the multi-neuron network can oscillate. This is addressed by using an arbitrary reset threshold value. The membrane potential will be compared to this value. The flag will be reset if the membrane potential is less then this value. The reset value was first set to -100.0 and a trial run, using standard values showed that the network would not oscillate easily. The value was increased to -95.0 which proved to be sufficient.

The interconnections are achieved by assigning values to a stimulus set (amplitude, duration and delay) for a particular neuron. That is, values assigned to

(1,4) matrix block indicate that when neuron 4 "fires" a stimulus will be sent to neuron 1. Initial values are assigned to the (*,1) matrix block values. (* indicates any neuron) In other words, the matrix assignments reflect which neurons are influenced by the respective neuron. The models assume that neurons can receive stimuli from an external source and/or from any of the other neurons in the matrix. The models do not consider neurons that feedback directly onto themselves. "Time adjustment" code is written for each neuron connected to another.

The "time adjustment" code consists of converting the dummy variables to working variables. The stimulus amplitude is a direct relation. (dummy = working) The working delay value is the sum of the simulation time (at which the respective action potential occurred) and the dummy variable. The working duration value is the sum of the working delay value and the dummy value.

The final simulation program is outlined in section 3.5 (including a flow chart) and the programs are listed in Appendix B.

3.3 Series Two-Neuron Pool Model and Simulation

The initial model development is to set up the simplest neuron pattern, a two neuron series model. The model is shown on the next page in FIGURE 3.1 This model would serve as the base for the extended multi-neuron pools.

The computational code from the Randall single neuron model is used to signify a neuron. The code is set as one subroutine and is called by each neuron. Matrices are used to keep track of the neurons.

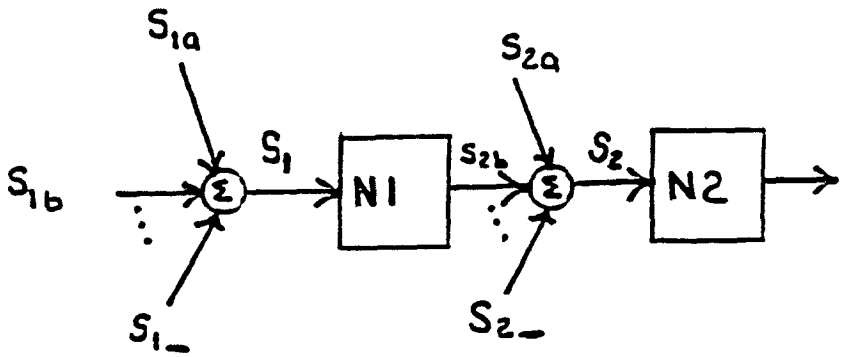

FIGURE 3.1: Series 2 Neuron Model Connections
The code used for the series 2 neuron simulation is listed in Appendix A and the program is named HHMOD6D.

3.4 Multi-Neuron Pool Model and Simulation

3.4.1 General Model and Simulation Concepts

This series of models includes a 2X2 (4 neuron pool, [N1-N4) and a 3X3 (9 neuron pool, [N1-N9) network. Matrices are used to keep track of which stimulus comes from which neuron. The assigned values are shown below in TABLE 3.1 and TABLE 3.2 on the next page.

ALL S,D,Y(*,1) VALUES ARE INITIAL STATES

N1 VALUES	N2 VALUES	N3 VALUES	N4 VALUES
(1,2)= 2>1	(2,2)= 1>2	(3,2)= 1>3	(4,2)= 1>4
(1,3)= 3>1	(2,3)= 3>2	(3,3)= 2>3	(4,3)= 2>4
(1,4)= 4>1	(2,4)= 4>2	(3,4)= 4>3	(4,4)= 3>4

TABLE 3.1: 2X2 Matrix Assignments.

ALL S.D.Y(*,1) VALUES ARE INITIAL STATES

N1 VALUES	N2 VALUES	N3 VALUES
(1,2)= 2>1	(2,2)= 1>2	(3,2)= 1>3
(1,3)= 3>1	(2,3)= 3>2	(3,3)= 2>3
(1,4)= 4>1	(2,4)= 4>2	(3,4)= 4>3
(1,5)= 5>1	(2,5)= 5>2	(3,5)= 5>3
(1,6)= 6>1	(2,6)= 6>2	(3,6)= 6>3
(1,7)= 7>1	(2,7)= 7>2	(3,7)= 7>3
(1,8)= 8>1	(2,8)= 8>2	(3,8)= 8>3
(1,9)= 9>1	(2,9)= 9>2	(3,9)= 9>3
N4 VALUES	**N5 VALUES**	**N6 VALUES**
(4,2)= 1>4	(5,2)= 1>5	(6,2)= 1>6
(4,3)= 2>4	(5,3)= 2>5	(6,3)= 2>6
(4,4)= 3>4	(5,4)= 3>5	(6,4)= 3>6
(4,5)= 5>4	(5,5)= 4>5	(6,5)= 4>6
(4,6)= 6>4	(5,6)= 6>5	(6,6)= 5>6
(4,7)= 7>4	(5,7)= 7>5	(6,7)= 7>6
(4,8)= 8>4	(5,8)= 8>5	(6,8)= 8>6
(4,9)= 9>4	(5,9)= 9>5	(6,9)= 9>6
N7 VALUES	**N8 VALUES**	**N9 VALUES**
(7,2)= 1>7	(8,2)= 1>8	(9,2)= 1>9
(7,3)= 2>7	(8,3)= 2>8	(9,3)= 2>9
(7,4)= 3>7	(8,4)= 3>8	(9,4)= 3>9
(7,5)= 4>7	(8,5)= 4>8	(9,5)= 4>9
(7,6)= 5>7	(8,6)= 5>8	(9,6)= 5>9
(7,7)= 6>7	(8,7)= 6>8	(9,7)= 6>9
(7,8)= 8>7	(8,8)= 7>8	(9,8)= 7>9
(7,9)= 9>7	(8,9)= 9>8	(9,9)= 8>9

TABLE 3.2: 3X3 Matrix Assignments

3.4.2 2X2 Pool

The 2X2 pool arbitrary interconnections are shown below in FIGURE 3.2. The connections were arbitrarily chosen. The program is named MULTIN1A and is displayed in Appendix B.

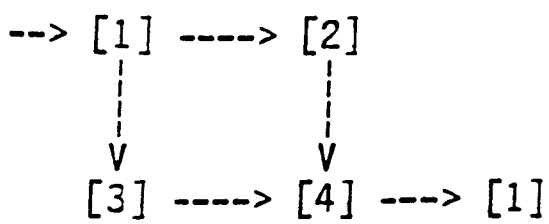

FIGURE 3.2: MULTIN1/1A Interconnections

The 2X2 model, shown in FIGURE 3.2, is used for trial runs prior to using the values on any of the 3X3 models. It is used due to the small number of neurons, thus, achieving a faster run time. This network is used as the base for the 3X3 model.

3.4.3 3X3 Pool

The 3X3 basic model simulation is generalized to accommodate most combinations of the nine neurons. The 3X3 model is arbitrarily connected in two different formats. The first format (simple) is shown below in FIGURE 3.3. This program is named MULTIN2A. (see Appendix B)

The second arbitrary format is slightly more complex and is shown in the below in FIGURE 3.4. This program is named MULTIN2B. (see Appendix B)

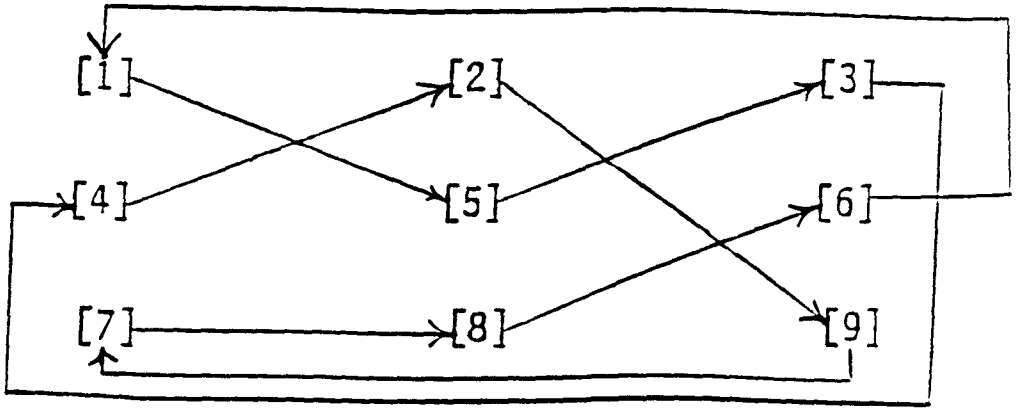

FIGURE 3.3: MULTIN2A (3X3, simple) Interconnections

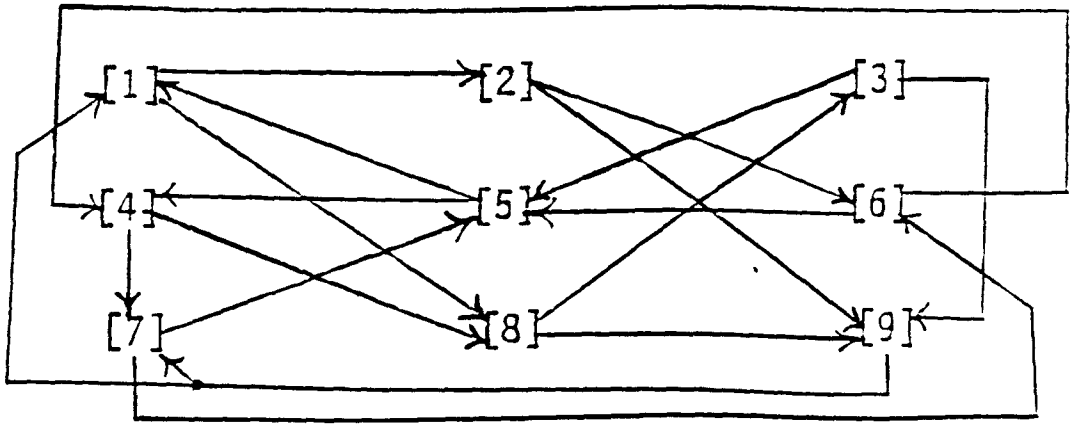

FIGURE 3.4: MULTIN2B (3X3, complex) Interconnections

3.5 Multi-Neuron Network Simulation Program

The generalized final simulation code is broken down into eight sections as shown in Appendix B. The first section consist of REM statements that identify the program and provide general user information.

The second section is used to initialize values, set up dimension statements and obtain user input values. The user input values are obtained by a subroutine described below under section 4.

The third section is the main routine. This section controls the flow of the subroutines and also tracks the simulation time. Initial values are transferred from holding variables to working variables in this section.

The fourth section is a subroutine called from section 2. This subroutine is used to obtain user input. The user values are the date of the run, the number of the simulation that was run that day (i.e. 1,2,3...), the amplitude, duration and delay values for each of the neurons stimuli and the total simulation time.

The fifth section consists of two subroutines that are called from the main program (section 3). The first routine routes the program to the initial value computational section. The second directs the program to the value updating computational section and is also used to reset the time adjustment overrun flag (= 0).

The sixth section is a subroutine that is used to convert the dummy variables into working variables. This

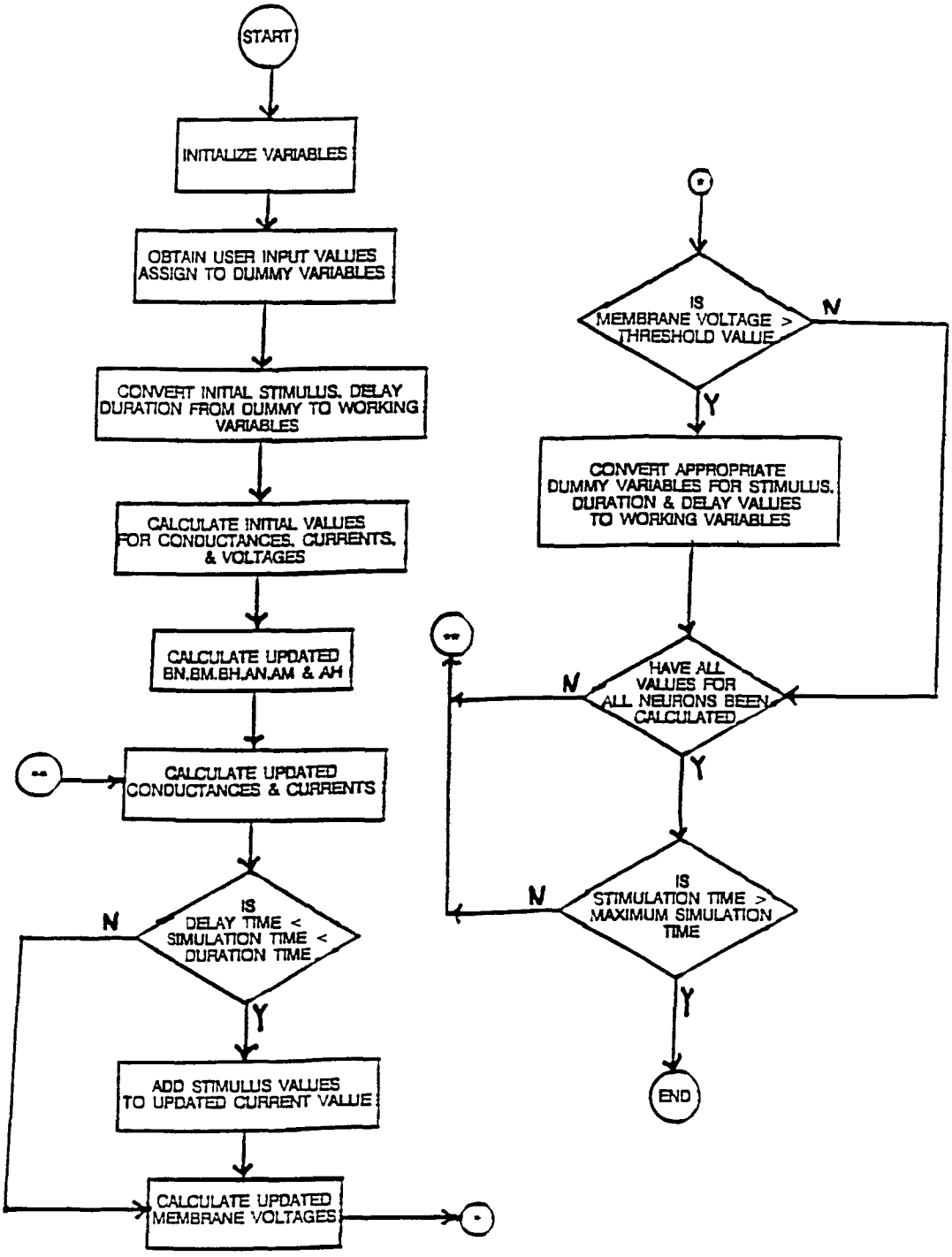

FIGURE 3.5: Flow Chart For the Multi-Neuron Simulation Program

section is used for interconnecting the neurons. Only neurons affecting other neurons have time adjustment code. The flag used to prevent this section from recalculating during an action potential is set (= 1). This is further discussed in section 3.2.

The seventh section is the computational portion of the program for both the initial calculations and the updated calculations. This is further discussed in sections 1.5.1 and 3.2.

The eighth section is used to print the output.

The program flow chart is shown on the facing page in FIGURE 3.5.

Chapter 4
SIMULATION RESULTS

4.1 General Information

Results are displayed on the following pages. These are selected samples of the data obtained. The membrane potential is used as the demonstration variable for the output, although the simulation calculates all the characteristics of the neuron. The results obtained are printed numerical values that are run specific. The data was manually transferred from the printed information to LOTUS 123, release 2.01, for graphing. The simulation programs are written in BASIC (ProDos) and run on an APPLE 2E enhanced personal computer.

Two types of graphs are used to present the data. The first type (membrane potential versus time) is one that displays the entire cycle of neuron activity including the polarized state, the action potential and the repolarization. This graph shows how stimuli cause the membrane potential to change but may not fully trigger an action potential. This graph can be used to compare the simulation results to theory. This graph became "crowded" as the number of neurons increased.

The second type (neuron number versus time) indicates when an action potential occurred. This graph is used to demonstrate the oscillatory characteristics of the simulation.

Stimulus amplitude values ranged for 50 to 200

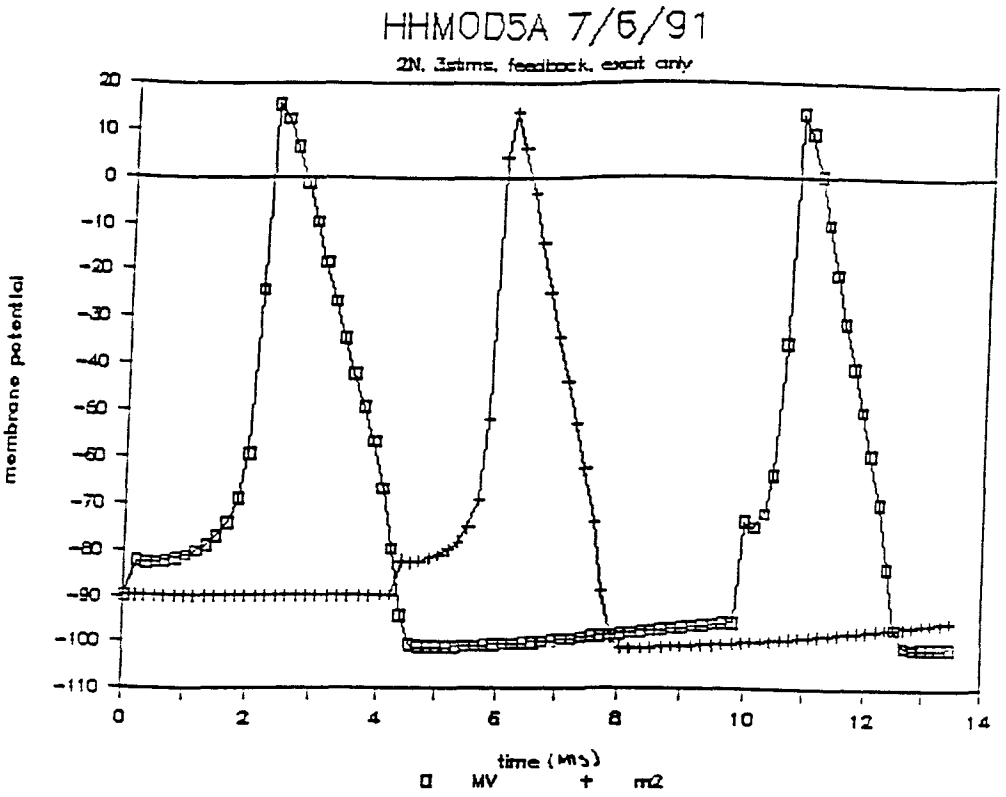

FIGURE 4.1: (HHMOD5A) Sample Results: Series 2 neuron model simulation using excitatory values only

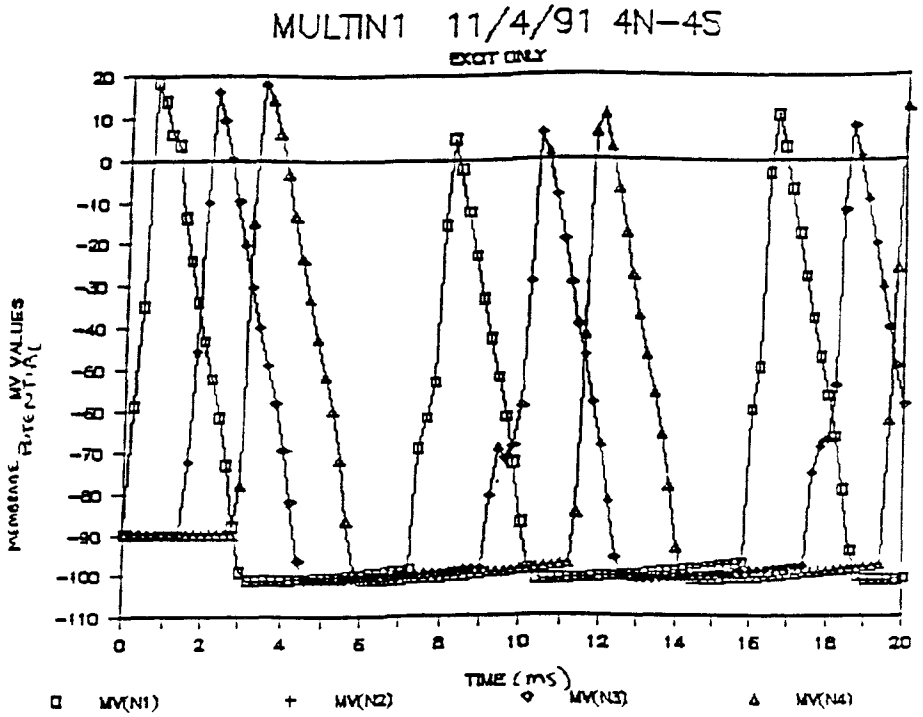

FIGURE 4.2: (MULTIN1) Initial Results: 2X2 simulation run. Only excitatory values are used.

A/cm^2. Amplitude values between 50 and 100 A/cm^2 are used to show a combination effect if more than one stimulus influenced a neuron. The duration values ranged from 0.24 - 0.5 ms. The delays varied between 0.5 and 5 ms, except for the initial delay of 0.04 ms. Delay times of 2 ms or greater are used to ensure that the system would oscillate. The delay has to be greater than the absolute refractory period in order for the oscillation to occur.

The average run time over a simulated 50 ms interval for the 2X2 model is 2.5 hours and for the 3X3 is 4.5 hours.

4.2 Series Two Neuron Simulation

FIGURE 4.1, on the facing page, shows the final run of the series 2 neuron model (FIGURE 3.1) prior to converting to a matrix system. Only excitatory values are used. The simulation is set up such that each neuron has three input stimuli and N1 triggers N2. N2 then feeds back to N1 using the third stimulus value. The system can not oscillate because N2 flags are not programmed to be reset.

4.3 Multi-Neuron Pools

4.3.1 2X2 Pools

In FIGURE 4.2, on the facing page, are the results from the first trial of the 2X2 network shown in FIGURE 3.2. N1 is initially stimulated by an external source. N2 and

N3 are stimulated by N1. The stimulus values for each are the same. N4 is stimulated by both N2 and N3. N4 then feedbacks to N1. Large stimulus values are chosen to ensure oscillations occur.

In FIGURE 4.3, below, are results from the 2X2 model (FIGURE 3.2). The stimulus values were decreased to see the effect. This trial did not oscillate. N2 was triggered, the second time, by a stimulus that was not strong enough to produce another action potential. N3's stimulus alone was not strong enough for N4 to produce another action potential. Thus, N1 was not triggered by N4. Therefore, the system can not oscillate. All the neuron eventually returned to the resting potential of -89.9 (not shown on the graph).

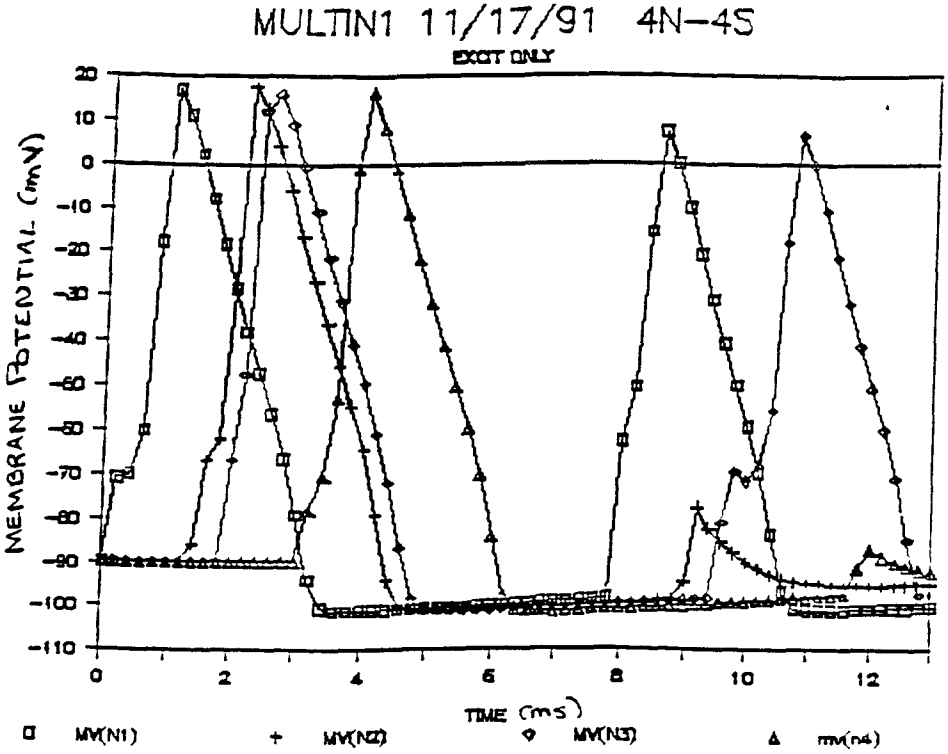

FIGURE 4.3: (MULTIN1) 2X2 simulation using excitatory values only. The system failed to oscillate.

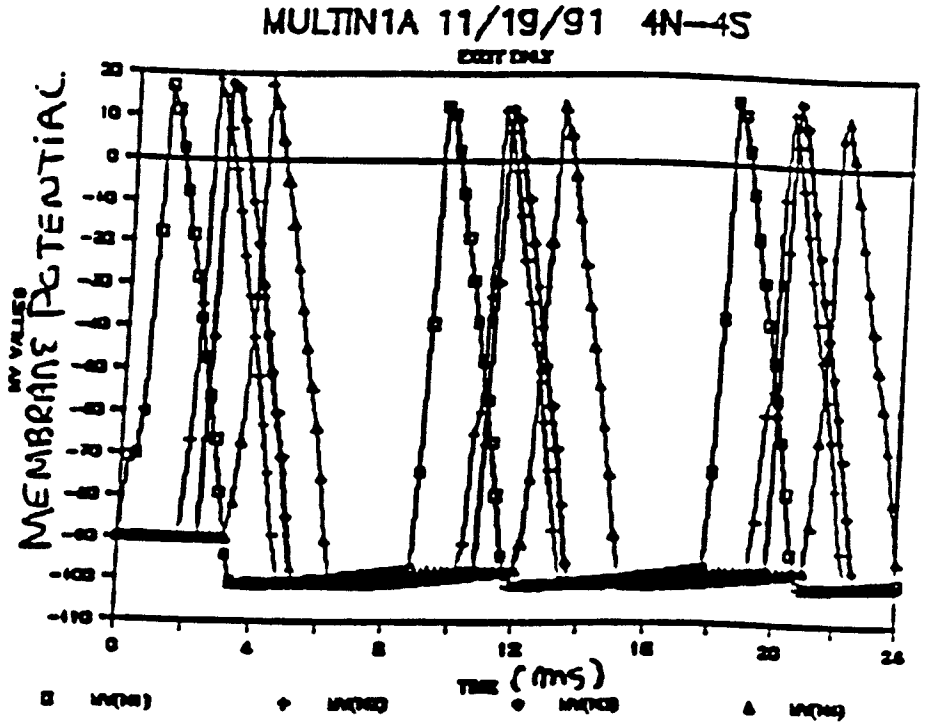

FIGURE 4.4.1: (MULTIN1) 2X2 model simulation using excitatory values only. The system achieves oscillations.

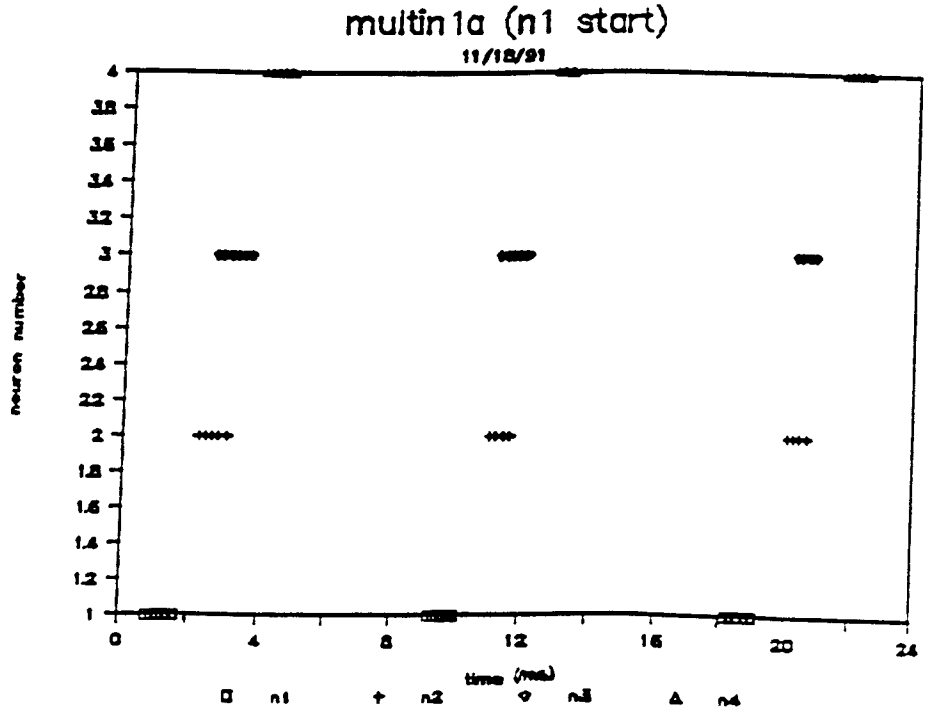

FIGURE 4.4.2: (MULTIN1) FIGURE 4.4.1 information in neuron number versus time format.

In FIGURE 4.4.1, shown on the facing page, are results from the 2X2 model simulation (FIGURE 3.5) that achieved oscillations. N1 is initiated by an outside source. N2 and N3 are triggered by N1 but with a difference in the stimulus delays. N4 is again triggered by a combination of N2 and N3. The delay between N4 and N1 was increased to produce oscillations.

FIGURE 4.4.2, shown on the facing page, is used for a comparison of the two graph types as described in Section 4.1. This graph displays the same results, as used in Figure 4.4.1, but in neuron number versus time format.

4.3.2 3X3 Pools

4.3.2.1 3X3 Model (Simple)

In FIGURE 4.5, shown facing page 26, are results from the initial run of the first 3X3 network (FIGURE 3.4). N1, N4 and N7 are triggered initially by an external stimulations. Only excitatory values are used. Stimulus amplitude values range between 100 and 150 with short delays (no greater than 2 ms). N1, N4 and N7 did not receive any further stimuli because N4, N5, N6, N8 and N9 did not produce a second action potential. Therefore, the system did not oscillate.

FIGURE 4.6, shown on facing page 27, is similar to that shown in FIGURE 4.5 except the delay values were increased (no greater than 5ms) along with the amplitude values (100 to 200). The system achieved oscillations.

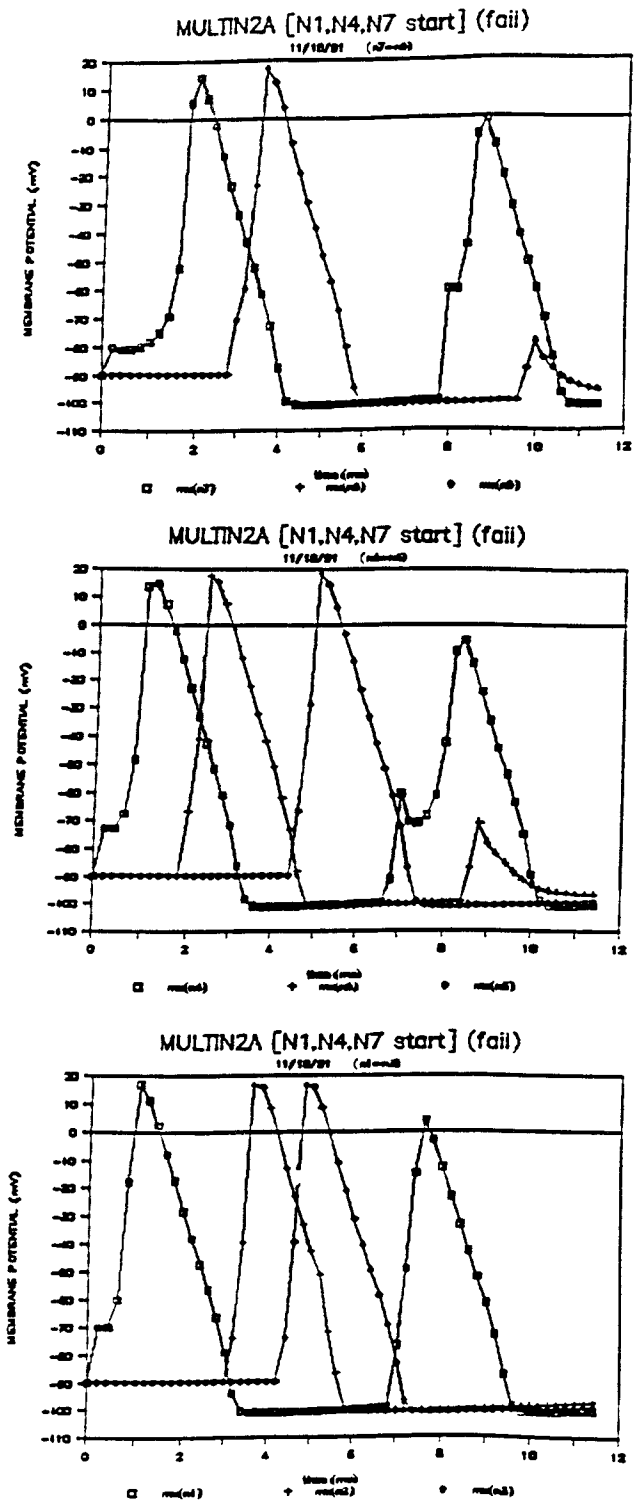

FIGURE 4.5: {(MULTIN2A) 3X3 network} Only excitatory values used with short delay times. N1, N4 and N7 initiated the network. No oscillations.

4.3.2.2 3X3 Model (Complex)

FIGURE 4.7, on page 27, displays the initial run of the second 3X3 network (FIGURE 3.5). N1, N3 and N4 are initially stimulated by an external source. All excitatory values are used. The simulation achieved oscillations.

FIGURE 4.8, shown on facing page 28, also displays results from the second 3X3 network shown in FIGURE 3.5. A single neuron (N3) is used to initiate the system. Only excitatory values were used. The simulation period is extended to show how the system responds over a long time period. The delay between N5 and N6 appears to become negligible as the simulation evolved. It is as if the two neurons became synchronized with each other.

FIGURE 4.9, shown on page 28, displays results from the 3X3 network shown in FIGURE 3.5. N9 is used as the starting neuron. N5 and N9 have three input stimuli whereas the rest of the neurons have only 1 or 2. This allowed one of the stimuli to be inhibitory. The value from N6 to N5 (6,6) and the value from N3 to N9 (9,4) are chosen as the inhibitory values. The graph is in the membrane potential versus time format to show the overall network response. The system achieved oscillations.

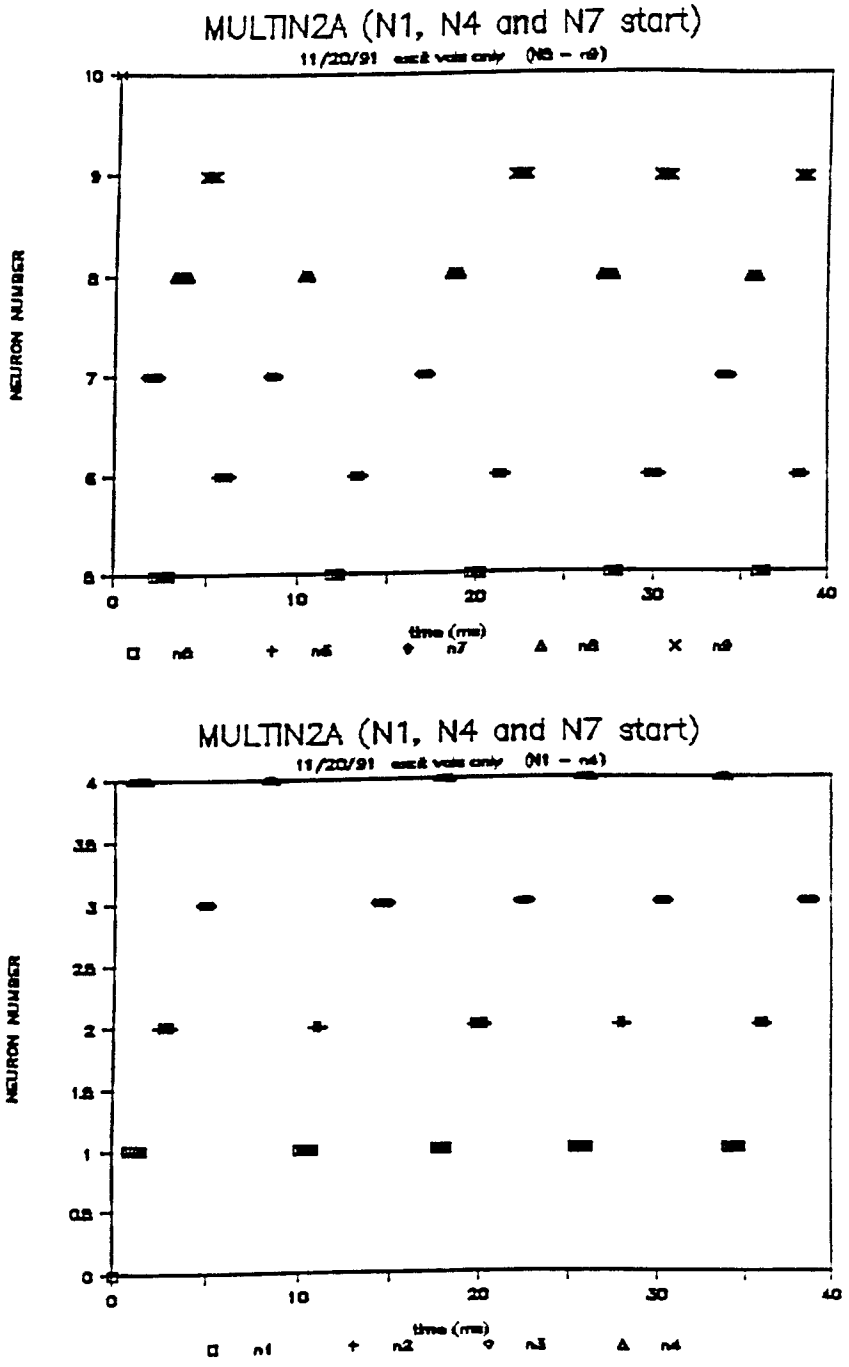

FIGURE 4.6: {(MULTIN2A) 3X3 network} Only excitatory values used. Delay times increase compared to Figure 10. N1, N4 and N7 initiated the system.

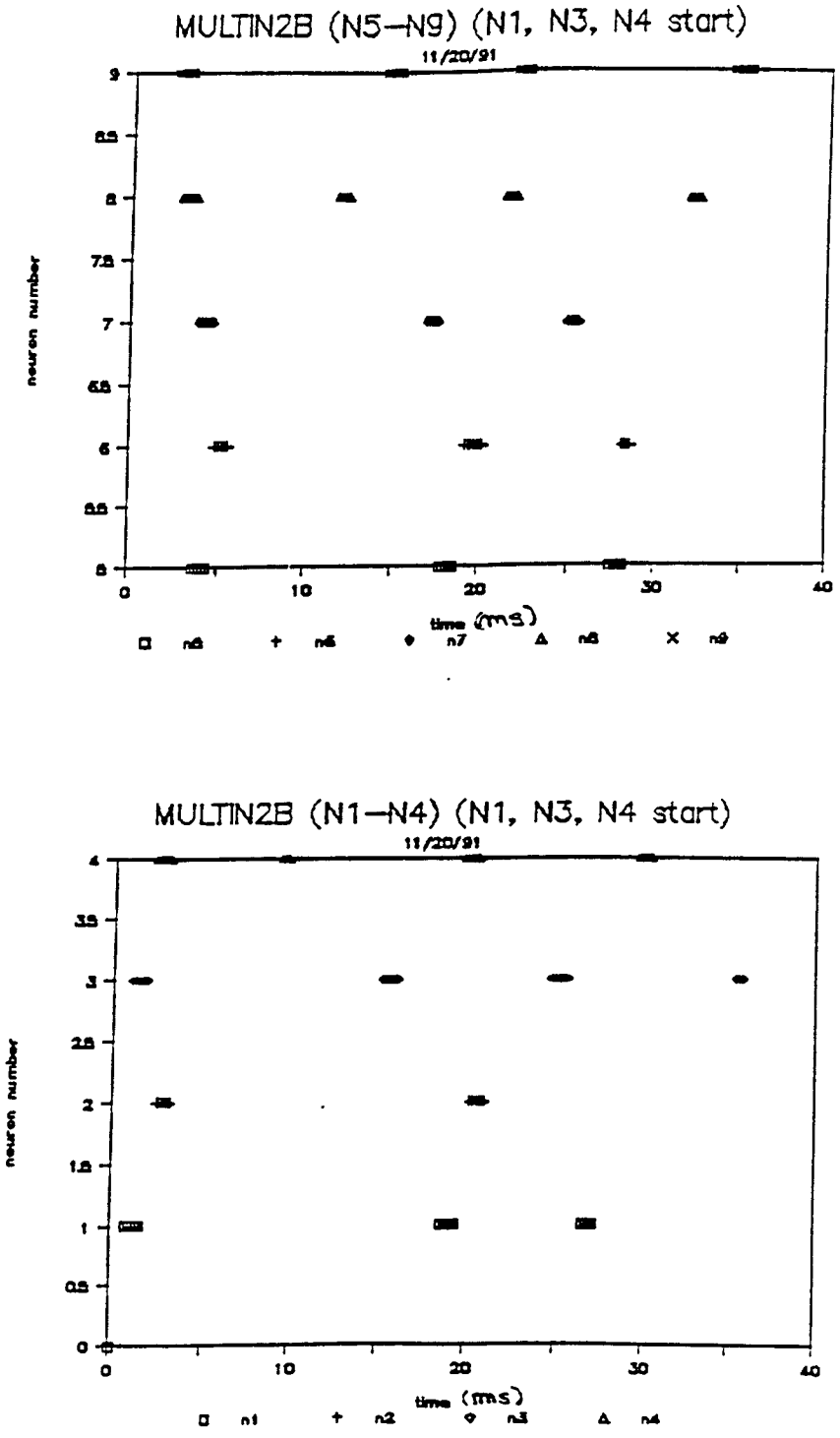

FIGURE 4.7: {(MULTIN2B) 3X3 network} Only excitatory values used. N1, N3 and N4 initiate the system.

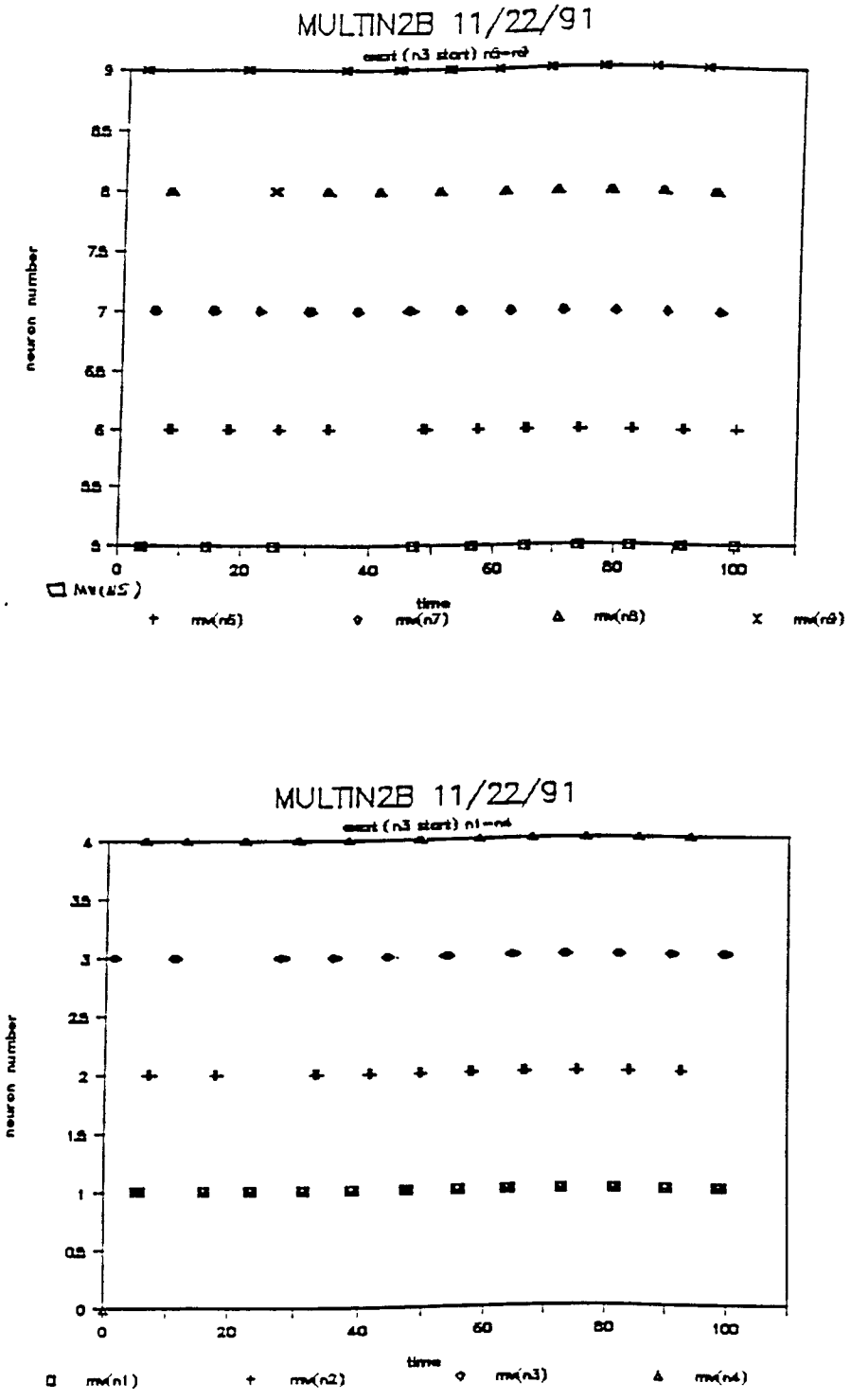

FIGURE 4.8: {(MULTIN2B) 3X3 network} Only excitatory values used. N3 used to initiate system.

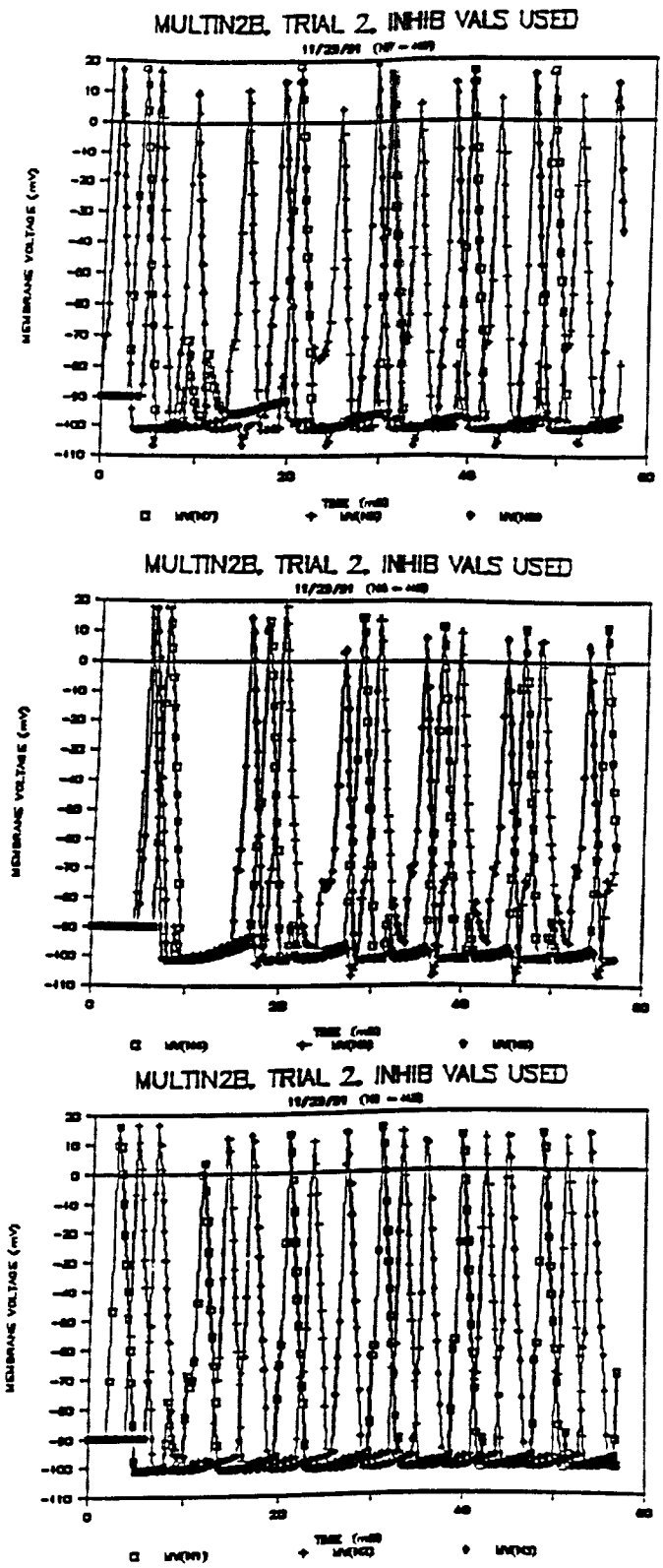

FIGURE 4.9: {(MULTIN2B) 3X3 network} N9 is used to initiate the system. Inhibitory values are assigned to the stimuli (6,6) and (9,4).

4.4 Other Results

Sample runs were done using a different threshold value for the flag (-1.0) as discussed in Section 3.4.1. The data used for FIGURE 4.9 were used for one of the test runs. The results were the same as shown in FIGURE 4.9. Other sample runs (results not shown) showed that if the delays are long enough the threshold for the flag does not matter. Future studies are required to determine the full impact of this value on the simulation.

4.5 DISCUSSION

The neural network models have been programmed to be very adaptable for future expansions. The primary change should be the translation of the program from BASIC to a complied language such as FORTRAN, C or compiled BASIC. This would allow for the data to enter and leave the program through other files, reducing run time. The output can be stored for future use (printed/plotted) versus having run specific output and manually transferring the information to a separate plotting program. The run time of a compiled program will be reduced.

The current 9 neuron (3X3) models are expandable to 10 neurons but the program can be renumbered to accommodate for different numbers of neurons, limited by only the capabilities of the computer. This expansion is done by changing the programming associated with the

neuron interconnections and adjusting for the number of neurons accordingly. The model can then be adapted to take into account neurons that directly feedback onto themselves by adding another matrix series.

A future model possibility is a 10 X 10 matrix (100 neuron) model. A 10 X 10 model is described by Traub, et. al. (8) (9) The Traub model is capable of simulating hippocampal responses such as synchronous bursting. The future model(s) and the Traub model(s) could be compared to produce other possible expansion.

The interconnections between the neurons can also be expanded. The model would be able to accommodate possible functions that link the previous neuron's activities to the activities of the next. The automatic setup of neuron connections should also be considered. Traub, et. al. set up the connections through a random process. This process automatically determines the number of inhibitory to excitatory neurons which would be present and how each neuron affects the others.

CHAPTER 5
INTRODUCTION TO BIOLOGICAL OSCILLATORS

5.1 Introduction

Future possibilities of oscillatory neural networks are to simulate biological oscillators (BO). This chapter is an introduction to BOs. It provides a very basic understanding of what they are; how they work and BO's importance in biological (human) systems.

The integrate and fire model (demonstrated in Section 5.3) is for example purposes. It is used to help visualize how simplified BOs function.

5.2 General Theory

Biological oscillators are inherent cycles of biological systems. Human BOs include, but are not limited to, temperature regulation, breathing, menstrual cycle, heart rate variability and circadian rhythms (10). The oscillations appear to be either self-initiated (i.e. pacemaker cells or electrically coupled isopotential cells) or controlled by the nervous system. The period of the oscillation is dependent upon many things some of which are not fully understood.

Biological oscillators can be affected by oscillations within that particular system or by oscillations external to that system (4). The interactions may lead to entrainment. Entrainment is also known as synchronization or phase locking (11). This

phenomenon allows for a biological system to "latch - on" to its environment (i.e. circadian rhythms entraining to the day-night cycle).

5.3 Integrate and Fire Model

One of the basic introductory models for phase locked biological oscillators is the integrate and fire (I&F) model (10), (11), (12). This model is used due to its ease of implementation.

The I&F model assumes a variable (the activity) increases linearly with time until a threshold is reached. Upon reaching this threshold the activity instantly and discontinuously resets to zero. This process is then repeated.

The ratio between the synchronized mode of the frequency of modulation and the oscillator is known as N:M phase locking. The simplest ratio is N:1 phase locking.

The example used in this chapter is adapted from the Glass & Mackey model in order to be implemented on the Apple 2E enhanced personal computer. The threshold is sinusoidally modulated (SMT) in time. This allows for the oscillator to become synchronized to the sinusoidal stimulus. The activity reaches the threshold at definite fixed phases of the SMT.

The associated mathematical equations are (5.1), (5.2) and (5.3) as shown on the next page.

The activity (X) is:

$$X = XO + (PC * T) \qquad (5.1)$$

XO is the initial activity value. PC is a positive constant. T is time.

The SMT (SS) is given by:

$$SS = SO + K * SIN ((W * T) + PA) \qquad (5.2)$$

SO is the initial SMT value. K is the amplitude of the SMT. K must be greater than 0 but less then the initial SMT value ($0 <= K < SO$). The angular frequency (W) and the phase angle (PA) must be greater than 0. After the threshold is reached, for the first time, the activity becomes:

$$X = PC * (T - TP) \qquad (5.3)$$

TP is the time at which the activity and the SMT are equal.

Dimensionless parameters are used in the simulation to expedite numerical computations. The final dimensionless equations for the activity (XX) and the SMT (S3) respectively are:

$$XX = X2 + (LA * TT) \qquad \{ TT > 0\} \qquad (5.4)$$

$$S3 = S2 + KK * SIN (6.28319 * TT + PA)$$

$$\{ 0 <= KK < 1 \} \qquad (5.5)$$

The variable X2, S2, LA, KK and TT are the dimensionless equivalent to those shown above in equations 5.2 and 5.3.

The ratio of the frequency of the SMT to the frequency of the unperturbed oscillator (R) when K = KK =

0 is given by:

$$R = 1 / LA \qquad (5.6)$$

where $1/LA = (W * SO) / (6.28319) * PC)$ (5.7)

Results from the I&F simulation are shown below in FIGURE 5.1. The program associated with the integrate and fire model and the program variables are listed in Appendix C and is named PROG3. The simulation program is further discussed in that section.

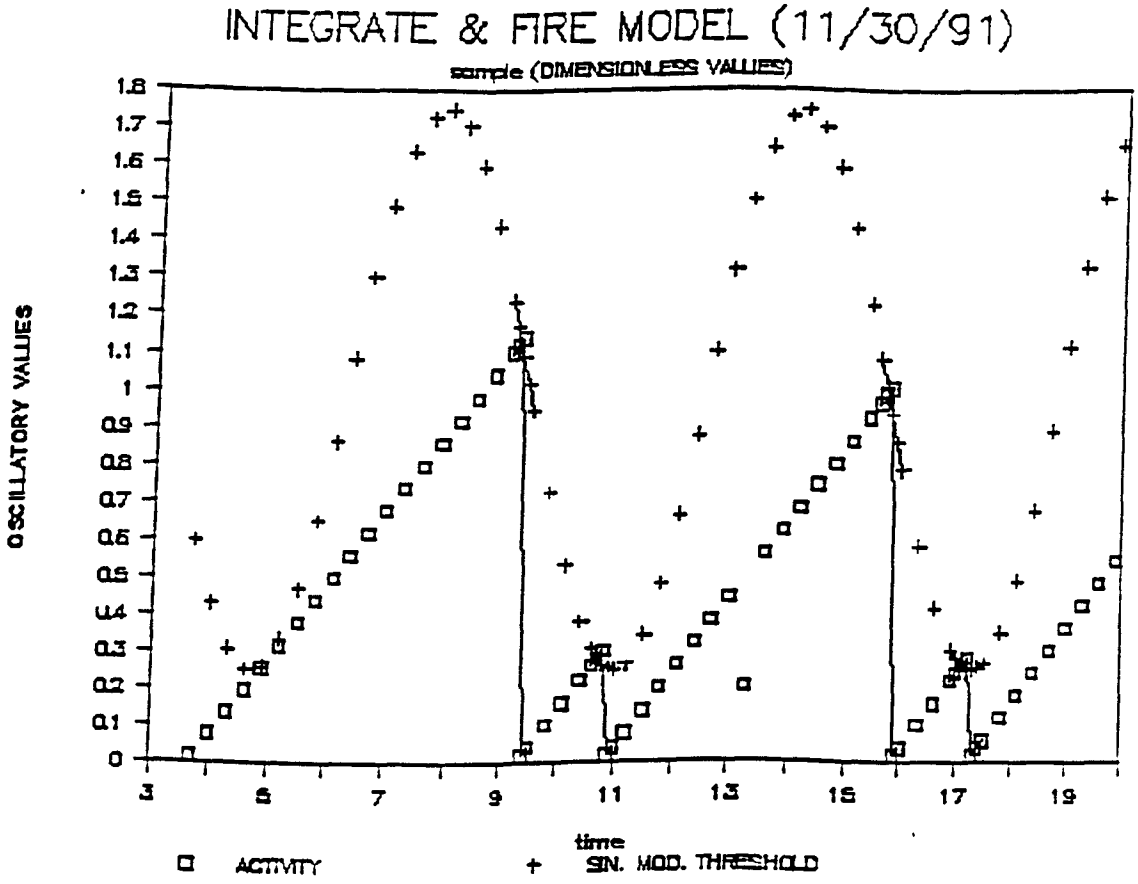

FIGURE 5.1: Sample Results From the Integrate & Fire Model: The values used are as follows: XO=0, SO=W=1, PC=DT=.1, K=.75, T3=75 and n=2.

Chapter 6
CONCLUSIONS

The Randall simulation as described in Section 1.5 is used to find out minimum input values of both amplitude and duration, in order to produce an action potential. It is found that the duration of the input stimulus and the amplitude of the stimulus play an important role in the response of the neuron. The smaller the threshold the longer duration is required for a response and vice versa.

The delay between the stimuli is also a key factor. Depending on the delay, the second stimulus would either be added to the first or it would not have any effect on the neuron at all. The second stimulus could even act as a separate initial stimulus, if the delay is long enough.

The model is found to have a long repolarization period. During this period the simulation is in a state of hyperpolarization of at least 11 mV below the resting potential. It was found that the model had to repolarize to at least 5 mV below the resting potential before an average stimulus would produce a response. The simulation is based on constant input stimuli values. Thus, the long refractory period becomes important when choosing a delay value for the next stimulus.

It was found that the timing from one neuron to the initiation of the next neuron is very important. A neuron will not produce a response if it is in the refractory period. This caused many attempts not to self oscillate.

The delay times are critical in order to properly stimulate the neuron for the program to continue.

The reset value plays an important role in the oscillating capabilities of the overall network. The value can appear to "extend" the refractory period if the delay values chosen are not long enough. The value has no effect on the output if the delay values are long enough. This value is a weak link in the model and should be a focal point for future developments.

The overall general characteristics of the network appear to be comparable to theory. The neuron did not fire if the input stimulus was not greater than the threshold value. This and delays became very important factors when trying to obtain oscillations. The action potential responses remained similar to theoretical values as compared to the Randall model.

Appendix A
HHMOD SERIES PROGRAMS

This series uses Randall's simulation of the Hodgkin-Huxley model. The program variables are listed below.

HHMOD SERIES	SYMBOLS AND DEFINITONS (ALPHABETICAL ORDER)
AH(10)	ALPHA-H, NA+ OFF-ON
AM(10)	ALPHA-M, NA+ OFF-ON
AN(10)	ALPHA-N, K+ OFF-ON
AS	Y/N REPLY TO RESTART THE PROGRAM
BH(10)	BETA-A, NA+ ON-OFF
BM(10)	BETA-M, NA+ ON-OFF
BN(10)	BETA-N, K+ ON-OFF
DL(10,10)	WORKING VALUE FOR THE STIMULUS DELAY
DT	CHANGE IN TIME
DU(10,10)	WORKING VALUE FOR THE STIMULUS DURATION
F1 - F6	FLAGS USED FOR PREVENTING DUEL CALCULATIONS
GK(10)	K+ CONDUCTANCE
GN(10)	NA+ CONDUCTANCE
H(10)	NA+ INACTIVATION
IK(10)	K+ CURRENT
IL(10)	LEAKAGE CURRENT
IN(10)	NA+ CURRENT
IT(10)	TOTAL CURRENT
MV(10)	MEMBRANCE VOLTAGE (INSIDE - OUTSIDE)
M(10)	NA+ ACTIVATION
N1	NN HOLDING VARIABLE
NN	NUMBER OF NEURONS IN THE MATRIX
N(10)	K+ ACTIVATION
Q, Q(1-7)	FOR-NEXT HOLDING VARIABLE FOR NN AND Z1
SA(10,10)	WORKING VALUE FOR THE STIMULUS AMPLITUDE
T	TIME (MSEC)
T1-T6,TN	TIMES USED TO ADJUST FOR DELAYS
TP	PRINT TIME
V(10)	MEMBRANE VOLTAGE (DISPLACEMENT FROM REST)
Y(10,10)	HOLDING VALUE FOR STIMULUS DELAY
Z1	NUMBER OF STIMULI PER NEURON

A1. HHMODEL1 - Randall's model adapted for the Apple IIe using the graphic display option. (pp 38-41)

A2. HHMOD6D - (8/6,18,25/91, 9/8,11,17/91) The HHMOD6D is the final 2 neuron series simulation. The model is general and is used as the base for the MULTIN series. (pp 42-45)

A.1 HHMODEL1 Program

```
10  REM  AXON ACTION POTENTIAL
20  REM       HODGKIN AND HUXLEY
30  REM       J.OF PHYSIO 177:500 (1952)
40  REM  THIS VERSION COMPUTES EXP() AS USED
50  REM       ABSOLUTE CONDUCTANCES
60  REM       NO ORDINATE LABELS
65  REM  PROGRAM NAMED HHMODEL1
70  :
80  :
99  REM  ////////////////////////////////
100  REM  FET STIM; SET STEADY STATE VALUES
150  TEXT : HOME :
151  REM  CLEAR SCREEN
199  REM  GET STIMULUS PARAMETERS
200  GOSUB 2000
300  LET T = 0
301  LET DT = 1 / 25
400  LET MV =  - 90
499  REM  SET ALPHA AND BETA
500  GOSUB 3000
599  REM  REM S-S N,M,H
600  GOSUB 3200
699  REM  S-S G,I'S
700  GOSUB 3500
709  REM  AXES LABELS
710  GOSUB 6000
720  :
730  :
799  REM  ************************
800  REM  LOOPS HERE EACH ITERATION
899  REM  DISPLAY VARIABLES
900  GOSUB 4000
999  REM  SET ALPHA AND BETA
1000  GOSUB 3000
1099  REM  UPDATE G'S
1100  GOSUB 5000
1199  REM  UPDAT I'S
1200  GOSUB 5500
1299  REM  UPDATE VOLTS
1300  GOSUB 5700
1400  LET T = T + DT
1499  REM  NEXT ITERATION
1500  GOTO 800
1600  :
1610  :
1999  REM  ############################
2000  : REM  GET STIMULUS VALUES
2001  TEXT : HOME : VTAB 5: HTAB 5
2002  PRINT "ENTER STIMULUS VALUES"
2003  PRINT : PRINT
2100  INPUT "STIM #1";SA(1)
2200  INPUT " DURAT ";SD(1)
2300  INPUT "DELAY ";DLY
```

A.1 HHMODEL1 Continued

```
2400  INPUT "STIM#2";SA(2)
2500  INPUT " DURAT ";SD(2)
2600  RETURN
2700 :
2999  REM  ****************************
3000  REM  RATE CONSTANTS FOR VOLTS
3020  LET V =  - MV - 90
3029  REM  AVOID 1/0
3030  IF V + 25 = 0 THEN V =  - 25.1
3040  IF V + 10 = 0 THEN V =  - 10.1
3050 :
3060  LET BN = 0.125 *  EXP (V / 80)
3070  LET BM = 4 *  EXP (V / 18)
3080  LET BH = 1 / ( EXP ((V + 30) / 10) + 1)
3100  LET AN = 0.01 * (V + 10) / ( EXP ((V + 10) / 10) - 1)
3110  LET AM = 0.1 * (V + 25) / ( EXP ((V + 25) / 10) - 1)
3120  LET AH = 0.07 *  EXP (V / 20)
3140  RETURN
3199  REM  &&&&&&&&&&&&&&&&&&&&&&&&&&&&
3200  REM  STEADY STATE VARIABLES
3210  LET N = AN / (AN + BN)
3220  LET M = AM / (AM + BM)
3230  LET H = AH / (AH + BH)
3240  RETURN
3250 :
3499  REM  >>>>>>>>>>>>>>>>>>>>>>>>>>
3500  REM  INITIAL G'S AND I'S
3520  LET GK = 36 * N * N * N * N
3530  LET GNA = 120 * M * M * M * H
3550  LET IK = GK * (V - 12)
3560  LET INA = GNA * (V + 115)
3570  LET IL = 0.3 * (V + 10.6)
3590  LET IT = IK + INA + IL
3610  RETURN
3998  REM  ****************************
3999 :
4000  REM  APPLE II GRAPHICS DISPLAY
4020  LET X = 25 + (T / DT)
4025  REM  OR X=25+250*(T/10)
4030  HPLOT X,20 - (MV / 1.5)
4039  REM  ABSOLUTE CONDUCTANCE
4040  HPLOT X,150 - 1.5 * GK
4050  HPLOT X,150 - 1.5 * GNA
4070  REM  STIMULUS IF ANY
4080  LET ST = 0
4090  IF T < SD(1) THEN ST = SA(1)
4100  IF T =  > DLY AND T < (DLY + SD(2)) THEN ST = ST + SA(2)
4110  HPLOT X,110 - (ST / 20)
4129  REM  SOFTWARE HALT
4130  IF X =  > 275 THEN  GOTO 4130
4140  RETURN
4198  REM  &&&&&&&&&&&&&&&&&&&&&&&&&&&&
```

A.1 HHMODEL1 Continued

```
4199 :
5000  REM  UPDATE CONDUCTANCES
5020  LET DN = AN * (1 - N) - BN * N
5030  LET DM = AM * (1 - M) - BM * M
5040  LET DH = AH * (1 - H) - BH * H
5060  LET N = N + DN * DT
5070  LET M = M + DM * DT
5080  LET H = H + DH * DT
5100  LET GK = 36 * N * N * N * N
5110  LET GNA = 120 * M * M * M * H
5130  RETURN
5139  REM  *************************
5140 :
5500  REM  NEW CURRENTS
5520  LET IK = GK * (V - 12)
5530  LET INA = GNA * (V + 115)
5540  LET IL = 0.3 * (V + 10.6)
5550  LET IT = IK + IN + IL
5560 :
5570  REM  IS THERE A STIMULUS TO ADD
5580  IF T < SD(1) THEN IT = IT + SA(1)
5590  IF T =  > (DLY + SD(2)) THEN  RETURN
5600  IF T =  > DLY THEN IT = IT + SA(2)
5620  RETURN
5699  REM  ////////////////////////////
5700  REM  NEW VOLTAGE
5710  LET DV = IT * DT
5720  LET V = V - DV
5730  LET MV =  - V - 90
5740  RETURN
5750 :
5760 :
5999  REM  ++++++++++++++++++++++++++++
6000  REM  PLOT AXES AND LABELS
6010  HGR : HOME
6020  HPLOT 25,151 TO 275,151
6030  FOR I = 0 TO 10
6040  HPLOT 25 + I * 25,152 TO 25 + I * 25,154
6050  NEXT I
6060  HPLOT 20,0 TO 20,86
6070  FOR I = 0 TO 13
6080  HPLOT 17,I * 6.67 TO 19,I * 6.67
6090  NEXT I
6100  HPLOT 15,20 TO 275,20
6110  HOME : VTAB 21
6120  HTAB 4: PRINT "0";
6130  HTAB 22: PRINT "5";
6140  HTAB 39: PRINT "10";
6150 :
6155  REM  DISPLAY STIMULUS PARAMETERS AT BOTTOM OF SCREEN
```

A.1 HHMODEL1 Continued

```
6160  VTAB 24: HTAB 1
6170  PRINT "#1";SD(1);"@";SA(1);
6180  HTAB 17: PRINT "DELAY=";DL;
6190  HTAB 29
6200  PRINT "#2=";SD(2);" @ ";SA(2);
6205 :
6210  REM  PLOT INITIAL STEADY STATE VALUES
6220  FOR X = 21 TO 24
6230  HPLOT X,20 - (MV / 1.5)
6240  HPLOT X,110
6250  HPLOT X,150 - 1.5 * GK
6260  HPLOT X,150 - 1.5 * GN
6270  NEXT X
6280  RETURN
6290 :
9999  END
```

A.2 HHMOD6D Program

```
10  REM  AXON ACTION POTENTIAL
20  REM       HODGKIN AND HUXLEY
30  REM       J.OF PHYSIO 177:500 (1952)
40  REM  THIS VERSION COMPUTES EXP() AS USED
50  REM       ABSOLUTE CONDUCTANCES
55  REM  GENERALIZED MODEL 1 NEURON SET WITH INDIVIDUAL NEURON MAIN
     NE
57  REM  N1 = # OF NEURONS
59  REM  J1 & J2 ARE EXCITE FROM N2 TO N1
65  REM  THIS IS A MODIFIED HH MODEL
66  REM  NO GRAPHICS PRINTS TEXT
67  REM  UPDATE 8/6,8/18,8/25,9/8,9/11,9/17/91
68  REM  HHMOD6D
69  REM  COMBINES N1 AND N2 VIA SYNAPES
70  REM  OUTPUT N1 TO INPUT N2 WHERE
71  REM  A USER CHOSEN SIMULUS IS USED
75  REM  ALL SUB ROUTINES ARE AT THE END OF THE PROGRAM
77  REM  STIM 3 OF N1 COMES FROM N2
78  REM  TIME DLY USED FROM N2 TO N1
80  REM  MULTI STIMULUS INPUTS
82  REM  INHIBITORY VALUES TO BE USED
84  REM  PRINTS EVERY 4TH VALUE ONLY
86  REM  LOOPS USED FOR MULTI STIMS
87  REM  ADDS STIMS AS LONG AS THERE IS ONE
99  REM  ////////////////////////////////
100  REM  INITIALIZE VALUES
200  DIM SA(10,10),DU(10,10),DL(10,10)
204  DIM V(10),MV(10)
206  DIM N(10),M(10),H(10)
208  DIM GK(10),GN(10)
210  DIM IK(10),IN(10),IT(10),IL(10)
212  DIM DN(10),DM(10),DH(10)
214  DIM BN(10),BM(10),BH(10)
216  DIM AN(10),AM(10),AH(10)
300 T = 0.00
301  LET DT = 1 / 25
305 BN(N1) = BM(N1) = BH(N1) = 0
310  REM  CHANGE TV TO SET THRESHOLD VALUE
312 TN = T4 = T5 = T6 = TP = 0
313 F3 = 0
314 F1 = 0
315 F2 = 0
316 Z5 = ZT = ZP = 0
318 MV(1) =  - 90
319 MV(2) =  - 90
320 N1 = 2
325 J1 = 200
420  REM  <//>\<//>\<//>\<//>\<//>\<//>\<//>\>//>\>
421  REM  START OF PROGRAMMING
425  REM  GET USER VALUES
427  GOSUB 2000
489 N1 = 1
```

A.2 HHMOD6D Continued

```
499  REM  SET ALPHA AND BETA
500  REM  SS N,M,H,G,I
505  GOSUB 3000
799  REM  ****************************************
800  REM  LOOPS HERE EACH ITERATION
998 N1 = 1
999  REM  SET ALPHA AND BETA
1000  REM  UPDATE G,I,V
1005  GOSUB 3050
1305  IF F1 >  = 1 THEN  GOTO 1515
1310  IF MV(1) >  = 0.00 THEN  GOSUB 1400
1320  IF F3 >  = 1 THEN  GOTO 1450
1350  IF T >  = TM THEN  GOTO 19000
1355 T = T + DT
1360  REM  NEXT ITERATION
1365  GOTO 800
1400  REM  SET UP COUNTER BETWEEN N1 AND N2
1402 F3 = 1
1408  FOR C = 1 TO Z1 STEP 1
1410 DL(2,C) = DL(2,C) + T
1412 DU(2,C) = DU(2,C) + DL(2,C) + T
1414  NEXT C
1420  REM  /-\/-\/-\/-\/-\/-\/-\/-\/-\/-\/-\/-\
1450  REM  START OF N2 PROGRAMMING
1460 F1 = 1
1465 TT = 0
1480  REM  SET UP ALPHA AND BETA
1481 N1 = 2
1482  PRINT "N2 PROGRAMMING STARTED"
1485  REM  SS N,M,H,G,I
1490  GOSUB 3000
1510  REM ++++++++++++++++++++++++++++++++++++
1515  REM  LOOPS HERE EACH ITERATION
1518 N1 = 2
1520  REM  UPDATE G,I,V
1525  GOSUB 3050
1595  IF F2 >  = 1 THEN  GOTO 1350
1600  IF MV(2) >  = TV THEN  GOSUB 6000
1989  REM  NEXT ITERATION
1990  GOTO 1350
1999  REM  ############################
2000 : REM  GET STIMULUS VALUES
2002  PRINT "ENTER STIMULUS VALUES"
2005  INPUT "HOW MANY STIMS PER N (MX=3)";Z1
2008  FOR Q0 = 1 TO N1 STEP 1
2010  FOR Q = 1 TO Z1 STEP 1
2015  PRINT "STIM VALUES FOR N=";Q0;"AND STIM=";Q
2020  INPUT "STIM VALUE";SA(Q0,Q)
2022  INPUT "DURAT VALUE";DU(Q0,Q)
2024  INPUT "DELAY VALUE";DL(Q0,Q)
2026  NEXT Q
2027  NEXT Q0
```

A.2 HHMOD6D Continued

```
2527  INPUT "DLY FROM N2 TO N1";Z5
2528  REM  DLY GREATER THEN REFRACT PERIOD TO STIM N1
2530  INPUT "MAX SIMULATION TIME";TM
2700  RETURN
2999  REM  ***************************
3000  REM  ++++++++++++++#########################++++++++++++++
     ###+#+#
3001  REM  FIRST GOSUB SUB SET
3010  GOSUB 8000
3015  GOSUB 8200
3020  RETURN
3050  REM  ()()()()()()()()()()())()()()()()()()()()()()()()
     ()()
3051  REM  SECOND GOSUB SET
3055  GOSUB 8000
3060  GOSUB 9000
3065  RETURN
6000  REM  /-\/-\/-\/-\/-\/-\/-\/-\/-\/-\/-\/-\/-\/-\/
6001  REM  OBTAIN STIM 3 N1 VALUES
6005 SA(1,3) = J1
6007 DL(1,3) = T + Z5
6010 DU(1,3) = T + Z5 + J2
6015 F2 = 1
6017  PRINT "REACHED FEEDBACK VALUES"
6025  RETURN
7999  REM  ++++++++++++++++++++++++++++++++++
8000  REM  RATE CONSTANTS FOR VOLTS
8020 V(N1) =  - MV(N1) - 90
8029  REM  AVOID 1/0
8030  IF V(N1) + 25 = 0 THEN V(N1) =  - 25.1
8040  IF V(N1) + 10 = 0 THEN V(N1) =  - 10.1
8060 BN(N1) = 0.125 *  EXP (V(N1) / 80)
8070 BM(N1) = 4 *  EXP (V(N1) / 18)
8081 BH(N1) = 1 / ( EXP ((V(N1) + 30) / 10) + 1)
8100 AN(N1) = 0.01 * (V(N1) + 10) / ( EXP ((V(N1) + 10) / 10) - 1)
8110 AM(N1) = 0.1 * (V(N1) + 25) / ( EXP ((V(N1) + 25) / 10) - 1)
8120 AH(N1) = 0.07 *  EXP (V(N1) / 20)
8130  RETURN
8200  REM  STEADY STATE VARIABLES
8210 N(N1) = AN(N1) / (AN(N1) + BN(N1))
8220 M(N1) = AM(N1) / (AM(N1) + BM(N1))
8230 H(N1) = AH(N1) / (AH(N1) + BH(N1))
8500  REM  INITIAL G AND I
8520  LET GK(N1) = 36 * N(N1) * N(N1) * N(N1) * N(N1)
8525  LET GN(N1) = 120 * M(N1) * M(N1) * M(N1) * H(N1)
8530 IK(N1) = GK(N1) * (V(N1) - 12)
8535 IN(N1) = GN(N1) * (V(N1) + 115)
8540 IL(N1) = 0.3 * (V(N1) + 10.6)
8545 IT(N1) = IK(N1) + IN(N1) + IL(N1)
8550  RETURN
8999  REM  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
9000  REM  UPDATE CONDUCTANCES
```

A.2 HHMOD6D Continued

```
9020 DN(N1) = AN(N1) * (1 - N(N1)) - BN(N1) * N(N1)
9030 DM(N1) = AM(N1) * (1 - M(N1)) - BM(N1) * M(N1)
9040 DH(N1) = AH(N1) * (1 - H(N1)) - BH(N1) * H(N1)
9060 N(N1) = N(N1) + DN(N1) * DT
9070 M(N1) = M(N1) + DM(N1) * DT
9080 H(N1) = H(N1) + DH(N1) * DT
9100 GK(N1) = 36 * N(N1) * N(N1) * N(N1) * N(N1)
9110 GN(N1) = 120 * M(N1) * M(N1) * M(N1) * H(N1)
9200 IK(N1) = GK(N1) * (V(N1) - 12)
9210 IN(N1) = GN(N1) * (V(N1) + 115)
9215 IL(N1) = 0.3 * (V(N1) + 10.6)
9220 IT(N1) = IK(N1) + IN(N1) + IL(N1)
9554  FOR Q3 = 1 TO Z1 STEP 1
9556  IF T > DL(N1,Q3) AND T < DU(N1,Q3) THEN IT(N1) = IT(N1) +
      SA(N1,Q3)
9558  NEXT Q3
9700  REM  NEW VOLTAGE
9710 DV(N1) = IT(N1) * DT
9720 V(N1) = V(N1) - DV(N1)
9730 MV(N1) =  - V(N1) - 90
9732  PRINT "MV OF";N1;"=";MV(N1);"    ";"V OF";N1;"=";V(N1)
9733  PRINT "TIME = ";T
9740  RETURN
9899  REM ++++++++++++++++++++++++++++++++++++++++++++
19000  INPUT "RESTART SIM. Y/N";A$
19002  IF A$ = "Y" THEN  GOTO 300
19999  END
```

Appendix B
MULTIN SERIES PROGRAMS

This series simulates the 2X2 (4 neuron) and the 3X3 (9 neuron) pools. The program variables are listed below:

MULTIN SERIES SYMBOLS AND DEFINITONS (ALPHABETICAL ORDER)

AH(10)	ALPHA-H, NA+ OFF-ON
AM(10)	ALPHA-M, NA+ OFF-ON
AN(10)	ALPHA-N, K+ OFF-ON
AS	Y/N REPLY TO RESTART THE PROGRAM
BH(10)	BETA-A, NA+ ON-OFF
BM(10)	BETA-M, NA+ ON-OFF
BN(10)	BETA-N, K+ ON-OFF
DL(10,10)	WORKING VALUE FOR THE STIMULUS DELAY
DT	CHANGE IN TIME
DU(10,10)	WORKING VALUE FOR THE STIMULUS DURATION
D(10,10)	HOLDING VALUE FOR STIMULUS DURATION
F(10)	FLAGS USED FOR PREVENTING DUEL CALCULATIONS
GK(10)	K+ CONDUCTANCE
GN(10)	NA+ CONDUCTANCE
H(10)	NA+ INACTIVATION
IK(10)	K+ CURRENT
IL(10)	LEAKAGE CURRENT
IN(10)	NA+ CURRENT
IT(10)	TOTAL CURRENT
MV(10)	MEMBRANCE VOLTAGE (INSIDE - OUTSIDE)
M(10)	NA+ ACTIVATION
N1	NN HOLDING VARIABLE
NN	NUMBER OF NEURONS IN THE MATRIX
N(10)	K+ ACTIVATION
Q, Q(1-7)	FOR-NEXT HOLDING VARIABLE FOR NN AND Z1
RS	THRESHOLD VALUE TO INITIATE FLAG RESET
RES	TODAYS DATE
SA(10,10)	WORKING VALUE FOR THE STIMULUS AMPLITUDE
S(10,10)	HOLDING VALUE FOR STIMULUS AMPLITUDE
T	TIME (MSEC)
TP	PRINT TIME
V(10)	MEMBRANE VOLTAGE (DISPLACEMENT FROM REST)
XX	RUN NUMBER
Y(10,10)	HOLDING VALUE FOR STIMULUS DELAY
Z1	NUMBER OF STIMULI PER NEURON

B.1 MULTIN1: (11/2,3,12,13/91) This is the 2X2 pool (FIGURE 3.2). (pp. 47-50)

B.2 MULTIN2: (11/12/91) This is the generalized base 3X3 simulation program. (pp 51-54)

B.3 MULTIN2A: (11/12,17/91) This is the simplified 3X3 simulation program. (FIGURE 3.3) (pp 55 - 59)

B.4 MULTIN2B: (11/12/17/91) This is the complex 3X3 pool simulation program. (FIGURE 3.5)(pp 60-64)

B.1 MULTIN1 Program

```
10  REM  AXON ACTION POTENTIAL
20  REM      HODGKIN AND HUXLEY
30  REM      J.OF PHYSIO 177:500 (1952)
40  REM  THIS VERSION COMPUTES EXP() AS USED
50  REM      ABSOLUTE CONDUCTANCES
52  REM  THIS IS A MODIFIED HH MODEL
54  REM  NO GRAPHICS PRINTS DATA ONLY
56  REM  USER DETERMINED NETWORK USING
57  REM  NEURON VALUES (S, D, Y)
58  REM  MODEL NAME:  MULTIN1
60  REM  UPDATE 11/2-3/91
62  REM  2 BY 2 NETWORK
64  REM  4 STIMS PER N, NO FEEDBACK ONTO ITSELF
66  REM  STIMS FROM OTHER N'S AND 1 EXTRN
99  REM  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%
100  REM  INITIALIZE VALUES
109  REM  NN = NUMBER OF NEURONS
110 NN = 4
111  REM  RS = RESET THRESHHOLD VAL.
112 RS =  - 95.0
114  REM  Z1=NUMBER OF STIMS/N
116 Z1 = 4
190 T = 0.00
192 DT = 1 / 25
195  DIM S(10,10),D(10,10),Y(10,10)
200  DIM SA(10,10),DU(10,10),DL(10,10)
204  DIM V(10),MV(10),F(10)
206  DIM N(10),M(10),H(10)
208  DIM GK(10),GN(10)
210  DIM IK(10),IN(10),IT(10),IL(10)
212  DIM DN(10),DM(10),DH(10)
214  DIM BN(10),BM(10),BH(10)
216  DIM AN(10),AM(10),AH(10)
225  FOR Q6 = 1 TO NN STEP 1
227  FOR Q7 = 1 TO Z1 STEP 1
229 SA(Q6,Q7) = 0.0
231 DL(Q6,Q7) = 0.0
233 DU(Q6,Q7) = 0.0
235  NEXT Q7
237 MV(Q6) =  - 90
239 F(Q6) = 0
241  NEXT Q6
305 BN(N1) = BM(N1) = BH(N1) = 0
310  REM  CHANGE TV TO SET THRESHOLD VALUE
312 TN = T4 = T5 = T6 = TP = 0
316 Z5 = ZT = ZP = 0
420  REM  </>\</>\</>\</>\</>\</>\>/>\>
421  REM  START OF PROGRAMMING
425  REM  GET USER VALUES
427  GOSUB 2000
430  FOR Q8 = 1 TO NN STEP 1
432 SA(Q8,1) = S(Q8,1)
434 DU(Q8,1) = D(Q8,1)
```

B.1 MULTIN1 Continued

```
 436 DL(Q8,1) = Y(Q8,1)
 438  NEXT Q8
 499  REM  SET ALPHA AND BETA
 500  REM  SS N,M,H,G,I
 505  GOSUB 3000
 799  REM  ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
 800  REM  LOOPS HERE EACH ITERATION
 999  REM  SET ALPHA AND BETA
1000  REM  UPDATE G,I,V
1005  GOSUB 3050
1350  IF T >  = TM THEN  GOTO 19000
1355 T = T + DT
1360  REM  NEXT ITERATION
1365  GOTO 800
1999  REM  ############################
2000 : REM  GET STIMULUS VALUES
2002  PRINT "ENTER STIMULUS VALUES"
2008  FOR Q0 = 1 TO NN STEP 1
2010  FOR Q = 1 TO Z1 STEP 1
2015  PRINT "STIM VALUES FOR N=";Q0;"AND STIM=";Q
2020  INPUT "STIM VALUE = ";S(Q0,Q)
2022  INPUT "DURAT VALUE = ";D(Q0,Q)
2024  INPUT "DELAY VALUE = ";Y(Q0,Q)
2026  NEXT Q
2027  NEXT Q0
2530  INPUT "MAX SIMULATION TIME";TM
2700  RETURN
3000  REM  ++++++++++++++++############################+++
      ###+#+#
3001  REM  FIRST GOSUB SUB SET
3005  FOR Q4 = 1 TO NN STEP 1
3007 N1 = Q4
3010  GOSUB 8000
3015  GOSUB 8200
3017  NEXT Q4
3020  RETURN
3050  REM  ()()()()()()()()()()()()()()()()()()()()()()()()
      ()()
3051  REM  SECOND GOSUB SET
3053  FOR Q5 = 1 TO NN STEP 1
3054 N1 = Q5
3055  GOSUB 8000
3060  GOSUB 9000
3062  IF MV(Q5) <  = RS THEN F(Q5) = 0
3064  IF F(Q5) = 1 THEN  GOTO 3068
3066  IF MV(Q5) >  = 0.0 THEN  GOSUB 4000
3068  NEXT Q5
3070  RETURN
4000  REM  ----------------------------------------
4001  REM  ASSIGN ACTUAL NETWORK VALUES
4002  REM  PROGRAM HERE TO CONTROL INTERCONNECTIONS
4010  IF N1 = 1 THEN  GOTO 4020
4012  IF N1 = 2 THEN  GOTO 4037
```

B.1 MULTIN1 Continued

```
 4014  IF N1 = 3 THEN  GOTO 4050
 4016  IF N1 = 4 THEN  GOTO 4060
 4018  RETURN
 4020  REM  PROGRAMMING FOR N=1
 4021 SA(2,2) = S(2,2)
 4023 DL(2,2) = Y(2,2) + T
 4025 DU(2,2) = D(2,2) + Y(2,2) + T
 4027 SA(3,2) = S(3,2)
 4029 DL(3,2) = Y(3,2) + T
 4031 DU(3,2) = D(3,2) + Y(3,2) + T
 4032 F(1) = 1
 4033  GOTO 4018
 4037  REM  PROGRAMMING FOR N=2
 4039 SA(4,3) = S(4,3)
 4041 DL(4,3) = Y(4,3) + T
 4043 DU(4,3) = D(4,3) + Y(4,3) + T
 4044 F(2) = 1
 4045  GOTO 4018
 4050  REM  PROGRAMMING FOR N=3
 4052 SA(4,4) = S(4,4)
 4054 DL(4,4) = Y(4,4) + T
 4056 DU(4,4) = D(4,4) + Y(4,4) + T
 4057 F(3) = 1
 4058  GOTO 4018
 4060  REM  PROGRAMMING FOR N=4
 4062 SA(1,4) = S(1,4)
 4064 DL(1,4) = Y(1,4) + T
 4066 DU(1,4) = D(1,4) + Y(1,4) + T
 4068 F(4) = 1
 4070  GOTO 4018
 7999  REM  ++++++++++++++++++++++++++++++++++
 8000  REM  RATE CONSTANTS FOR VOLTS
 8020 V(N1) =  - MV(N1) - 90
 8029  REM  AVOID 1/0
 8030  IF V(N1) + 25 = 0 THEN V(N1) =  - 25.1
 8040  IF V(N1) + 10 = 0 THEN V(N1) =  - 10.1
 8060 BN(N1) = 0.125 *  EXP (V(N1) / 80)
 8070 BM(N1) = 4 *  EXP (V(N1) / 18)
 8081 BH(N1) = 1 / ( EXP ((V(N1) + 30) / 10) + 1)
 8100 AN(N1) = 0.01 * (V(N1) + 10) / ( EXP ((V(N1) + 10) / 10) - 1)
 8110 AM(N1) = 0.1 * (V(N1) + 25) / ( EXP ((V(N1) + 25) / 10) - 1)
 8120 AH(N1) = 0.07 *  EXP (V(N1) / 20)
 8130  RETURN
 8200  REM  STEADY STATE VARIABLES
 8210 N(N1) = AN(N1) / (AN(N1) + BN(N1))
 8220 M(N1) = AM(N1) / (AM(N1) + BM(N1))
 8230 H(N1) = AH(N1) / (AH(N1) + BH(N1))
 8500  REM  INITIAL G AND I
 8520  LET GK(N1) = 36 * N(N1) * N(N1) * N(N1) * N(N1)
 8525  LET GN(N1) = 120 * M(N1) * M(N1) * M(N1) * H(N1)
 8530 IK(N1) = GK(N1) * (V(N1) - 12)
 8535 IN(N1) = GN(N1) * (V(N1) + 115)
```

B.1 MULTIN1 Continued

```
8540 IL(N1) = 0.3 * (V(N1) + 10.6)
8545 IT(N1) = IK(N1) + IN(N1) + IL(N1)
8550  RETURN
8999  REM  ////////////////////////////////////////////////
9000  REM  UPDATE CONDUCTANCES
9020 DN(N1) = AN(N1) * (1 - N(N1)) - BN(N1) * N(N1)
9030 DM(N1) = AM(N1) * (1 - M(N1)) - BM(N1) * M(N1)
9040 DH(N1) = AH(N1) * (1 - H(N1)) - BH(N1) * H(N1)
9060 N(N1) = N(N1) + DN(N1) * DT
9070 M(N1) = M(N1) + DM(N1) * DT
9080 H(N1) = H(N1) + DH(N1) * DT
9100 GK(N1) = 36 * N(N1) * N(N1) * N(N1) * N(N1)
9110 GN(N1) = 120 * M(N1) * M(N1) * M(N1) * H(N1)
9200 IK(N1) = GK(N1) * (V(N1) - 12)
9210 IN(N1) = GN(N1) * (V(N1) + 115)
9215 IL(N1) = 0.3 * (V(N1) + 10.6)
9220 IT(N1) = IK(N1) + IN(N1) + IL(N1)
9554  FOR Q3 = 1 TO Z1 STEP 1
9556  IF T > DL(N1,Q3) AND T < DU(N1,Q3) THEN IT(N1) = IT(N1) +
      SA(N1,Q3)
9558  NEXT Q3
9700  REM  NEW VOLTAGE
9710 DV(N1) = IT(N1) * DT
9720 V(N1) = V(N1) - DV(N1)
9730 MV(N1) =  - V(N1) - 90
9732  PRINT "MV OF";N1;"=";MV(N1);"    ";"V OF";N1;"=";V(N1)
9733  PRINT "TIME = ";T
9740  RETURN
9899  REM ++++++++++++++++++++++++++++++++++++++++++++
19000  INPUT "RESTART SIM. Y/N";A$
19002  IF A$ = "Y" THEN  GOTO 300
19999  END
```

B.2 MULTIN2 Program

```
10  REM  AXON ACTION POTENTIAL
20  REM       HODGKIN AND HUXLEY
30  REM       J.OF PHYSIO 177:500 (1952)
40  REM  THIS VERSION COMPUTES EXP() AS USED
50  REM       ABSOLUTE CONDUCTANCES
52  REM  THIS IS A MODIFIED HH MODEL
54  REM  NO GRAPHICS PRINTS DATA ONLY
56  REM  USER DETERMINED NETWORK USING
57  REM  NEURON VALUES (S, D, Y)
58  REM  MODEL NAME: MULTIN2
60  REM  UPDATE 11/12/91
63  REM  3 BY 3 NETWORK
64  REM  9 STIMS PER N, NO FEEDBACK ONTO ITSELF BUT VCAN BE ADDED
     R
66  REM  STIMS FROM OTHER N'S AND 1 EXTRN
99  REM  '////////////////////////////////
100   REM   INITIALIZE VALUES
109   REM   NN = NUMBER OF NEURONS
110 NN = 9
111   REM   RS = RESET THRESHHOLD VAL.
112 RS =  - 95.0
114   REM   Z1=NUMBER OF STIMS/N
116 Z1 = 9
118 TP = 0.0
190 T = 0.00
192 DT = 1 / 25
195   DIM S(10,10),D(10,10),Y(10,10)
200   DIM SA(10,10),DU(10,10),DL(10,10)
204   DIM V(10),MV(10),F(10)
206   DIM N(10),M(10),H(10)
208   DIM GK(10),GN(10)
210   DIM IK(10),IN(10),IT(10),IL(10)
212   DIM DN(10),DM(10),DH(10)
214   DIM BN(10),BM(10),BH(10)
216   DIM AN(10),AM(10),AH(10)
225   FOR Q6 = 1 TO NN STEP 1
227   FOR Q7 = 1 TO Z1 STEP 1
229 SA(Q6,Q7) = 0.0
231 DL(Q6,Q7) = 0.0
233 DU(Q6,Q7) = 0.0
235   NEXT Q7
237 MV(Q6) =  - 90
239 F(Q6) = 0
241   NEXT Q6
305 BN(N1) = BM(N1) = BH(N1) = 0
310   REM   CHANGE TV TO SET THRESHOLD VALUE
420   REM   </>\</>\</>\</>\</>\</>\>/>\>
421   REM   START OF PROGRAMMING
425   REM   GET USER VALUES
427   GOSUB 2000
430   FOR Q8 = 1 TO NN STEP 1
432 SA(Q8,1) = S(Q8,1)
434 DU(Q8,1) = D(Q8,1)
```

B.2 MULTIN2 Continued

```
436 DL(Q8,1) = Y(Q8,1)
438  NEXT Q8
499  REM  SET ALPHA AND BETA
500  REM  SS N,M,H,G,I
505  GOSUB 3000
799  REM  *******************************************
800  REM  LOOPS HERE EACH ITERATION
999  REM  SET ALPHA AND BETA
1000  REM  UPDATE G,I,V
1005  GOSUB 3050
1350  IF T >  = TM THEN  GOTO 19000
1355 T = T + DT
1360  REM  NEXT ITERATION
1365  GOTO 800
1999  REM  ###########################
2000 : REM  GET STIMULUS VALUES
2001  PRINT "MODEL NAME - MULTIN2"
2002  PRINT "ENTER STIMULUS VALUES"
2008  FOR Q0 = 1 TO NN STEP 1
2010  FOR Q = 1 TO Z1 STEP 1
2015  PRINT "STIM VALUES FOR N=";Q0;"AND STIM=";Q
2020  INPUT "STIM VALUE = ";S(Q0,Q)
2022  INPUT "DURAT VALUE = ";D(Q0,Q)
2024  INPUT "DELAY VALUE = ";Y(Q0,Q)
2026  NEXT Q
2027  NEXT Q0
2530  INPUT "MAX SIMULATION TIME";TM
2700  RETURN
3000  REM  ++++++++++++++######################+++++++++
     ###+#+#
3001  REM  FIRST GOSUB SUB SET
3005  FOR Q4 = 1 TO NN STEP 1
3007 N1 = Q4
3010  GOSUB 8000
3015  GOSUB 8200
3017  NEXT Q4
3020  RETURN
3050  REM  ()()()()()()()()()()())()()()()()()()()()()()
     ()()
3051  REM  SECOND GOSUB SET
3053  FOR Q5 = 1 TO NN STEP 1
3054 N1 = Q5
3055  GOSUB 8000
3060  GOSUB 9000
3062  IF MV(Q5) <  = RS THEN F(Q5) = 0
3064  IF F(Q5) = 1 THEN  GOTO 3068
3066  IF MV(Q5) >  = 0.0 THEN  GOSUB 4000
3068  NEXT Q5
3070  RETURN
4000  REM  ---------------------------------------
4001  REM  ASSIGN WORKING VALUES HERE
4002  REM  CONTROLS INTERCONECTIONS BY CHOICE OF S,D AND Y
```

B.2 MULTIN2 Continued

```
4003  REM  EACH SET PROGRAMS WHAT IS EFFECTED BY THAT PARICULAR N
     T EFFECTS IT
4008  IF N1 = 1 THEN  GOTO 4030
4009  IF N1 = 2 THEN  GOTO 4060
4010  IF N1 = 3 THEN  GOTO 4090
4011  IF N1 = 4 THEN  GOTO 4120
4012  IF N1 = 5 THEN  GOTO 4150
4013  IF N1 = 6 THEN  GOTO 4180
4014  IF N1 = 7 THEN  GOTO 4210
4015  IF N1 = 8 THEN  GOTO 4240
4016  IF N1 = 9 THEN  GOTO 4270
4020  RETURN
4030  REM  PROGRAMMING FOR N1=1
4058 F(1) = 1
4059  GOTO 4020
4060  REM  PROGRAMMING FOR N1=2
4088 F(2) = 1
4089  GOTO 4020
4090  REM  PROGRAMMING FOR N1=3
4118 F(3) = 1
4119  GOTO 4020
4120  REM  PROGRAMMING FOR N1=4
4148 F(4) = 1
4149  GOTO 4020
4150  REM  PROGRAMMING FOR N1=5
4178 F(5) = 1
4179  GOTO 4020
4180  REM  PROGRAMMING FOR N1=6
4208 F(6) = 1
4209  GOTO 4020
4210  REM  PROGRAMMING FOR N1=7
4238 F(7) = 1
4239  GOTO 4020
4240  REM  PROGRAMMING FOR N1=8
4268 F(8) = 1
4269  GOTO 4020
4270  REM  PROGRAMMING FOR N1=9
4298 F(9) = 1
4299  GOTO 4020
7999  REM  ++++++++++++++++++++++++++++++++++
8000  REM  RATE CONSTANTS FOR VOLTS
8020 V(N1) =  - MV(N1) - 90
8029  REM  AVOID 1/0
8030  IF V(N1) + 25 = 0 THEN V(N1) =  - 25.1
8040  IF V(N1) + 10 = 0 THEN V(N1) =  - 10.1
8060 BN(N1) = 0.125 *  EXP (V(N1) / 80)
8070 BM(N1) = 4 *  EXP (V(N1) / 18)
8081 BH(N1) = 1 / ( EXP ((V(N1) + 30) / 10) + 1)
8100 AN(N1) = 0.01 * (V(N1) + 10) / ( EXP ((V(N1) + 10) / 10) - 1)
8110 AM(N1) = 0.1 * (V(N1) + 25) / ( EXP ((V(N1) + 25) / 10) - 1)
8120 AH(N1) = 0.07 *  EXP (V(N1) / 20)
8130  RETURN
```

B.2 MULTIN2 Continued

```
 8200  REM  STEADY STATE VARIABLES
 8210 N(N1) = AN(N1) / (AN(N1) + BN(N1))
 8220 M(N1) = AM(N1) / (AM(N1) + BM(N1))
 8230 H(N1) = AH(N1) / (AH(N1) + BH(N1))
 8500  REM  INITIAL G AND I
 8520  LET GK(N1) = 36 * N(N1) * N(N1) * N(N1) * N(N1)
 8525  LET GN(N1) = 120 * M(N1) * M(N1) * M(N1) * H(N1)
 8530 IK(N1) = GK(N1) * (V(N1) - 12)
 8535 IN(N1) = GN(N1) * (V(N1) + 115)
 8540 IL(N1) = 0.3 * (V(N1) + 10.6)
 8545 IT(N1) = IK(N1) + IN(N1) + IL(N1)
 8550  RETURN
 8999  REM  ////////////////////////////////////////////////////
 9000  REM  UPDATE CONDUCTANCES
 9020 DN(N1) = AN(N1) * (1 - N(N1)) - BN(N1) * N(N1)
 9030 DM(N1) = AM(N1) * (1 - M(N1)) - BM(N1) * M(N1)
 9040 DH(N1) = AH(N1) * (1 - H(N1)) - BH(N1) * H(N1)
 9060 N(N1) = N(N1) + DN(N1) * DT
 9070 M(N1) = M(N1) + DM(N1) * DT
 9080 H(N1) = H(N1) + DH(N1) * DT
 9100 GK(N1) = 36 * N(N1) * N(N1) * N(N1) * N(N1)
 9110 GN(N1) = 120 * M(N1) * M(N1) * M(N1) * H(N1)
 9200 IK(N1) = GK(N1) * (V(N1) - 12)
 9210 IN(N1) = GN(N1) * (V(N1) + 115)
 9215 IL(N1) = 0.3 * (V(N1) + 10.6)
 9220 IT(N1) = IK(N1) + IN(N1) + IL(N1)
 9554  FOR Q3 = 1 TO Z1 STEP 1
 9556  IF T > DL(N1,Q3) AND T < DU(N1,Q3) THEN IT(N1) = IT(N1) +
       SA(N1,Q3)
 9558  NEXT Q3
 9700  REM  NEW VOLTAGE
 9710 DV(N1) = IT(N1) * DT
 9720 V(N1) = V(N1) - DV(N1)
 9730 MV(N1) =  - V(N1) - 90
 9732  IF T >  = TP THEN  GOTO 9735
 9733  RETURN
 9734  REM  PRINT PROGRAMMING
 9735  PRINT "TIME= ";T;" N= ";N1;
 9737  PRINT "MV= ";MV(N1);"V= ";V(N1)
 9739  IF N1 = 9 THEN TP = TP + 0.2
 9740  GOTO 9733
 9899  REM ++++++++++++++++++++++++++++++++++++++++++++
19000  INPUT "RESTART SIM. Y/N";A$
19002  IF A$ = "Y" THEN  GOTO 300
19999  END
```

B.3 MULTIN2A Program

```
10  REM  AXON ACTION POTENTIAL
20  REM      HODGKIN AND HUXLEY
30  REM      J.OF PHYSIO 177:500 (1952)
40  REM  THIS VERSION COMPUTES EXP() AS USED
50  REM      ABSOLUTE CONDUCTANCES
52  REM  THIS IS A MODIFIED HH MODEL
54  REM  NO GRAPHICS PRINTS DATA ONLY
56  REM  USER DETERMINED NETWORK USING
57  REM  NEURON VALUES (S, D, Y)
58  REM  MODEL NAME: MULTIN2A
59  REM  TRAIL 1 INTERREALATED SINGLE PATHWAYS
60  REM  UPST 11/12,17/91
63  REM  3 BY 3 NETWORK
64  REM  9 STIMS PER N, NO FEEDBACK ONTO ITSELF BUT VCAN BE ADDED
     R
66  REM  STIMS FROM OTHER N'S AND 1 EXTRN
99  REM  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%
100  REM  INITIALIZE VALUES
109  REM  NN = NUMBER OF NEURONS
110 NN = 9
111  REM  RS = RESET THRESHHOLD VAL.
112 RS =  - 95.0
114  REM  Z1=NUMBER OF STIMS/N
116 Z1 = 9
192 DT = 1 / 25
195  DIM S(10,10),D(10,10),Y(10,10)
200  DIM SA(10,10),DU(10,10),DL(10,10)
204  DIM V(10),MV(10),F(10)
206  DIM N(10),M(10),H(10)
208  DIM GK(10),GN(10)
210  DIM IK(10),IN(10),IT(10),IL(10)
212  DIM DN(10),DM(10),DH(10)
214  DIM BN(10),BM(10),BH(10)
216  DIM AN(10),AM(10),AH(10)
220 T = 0.0
222 TP = 0.0
225  FOR Q6 = 1 TO NN STEP 1
227  FOR Q7 = 1 TO Z1  STEP 1
229 SA(Q6,Q7) = 0.0
231 DL(Q6,Q7) = 0.0
233 DU(Q6,Q7) = 0.0
235  NEXT Q7
237 MV(Q6) =  - 90
239 F(Q6) = 0
241  NEXT Q6
305 BN(N1) = BM(N1) = BH(N1) = 0
310  REM  CHANGE TV TO SET THRESHOLD VALUE
420  REM  </>\</>\</>\</>\</>\</>\>/>\>
421  REM  START OF PROGRAMMING
425  REM  GET USER VALUES
427  GOSUB 2000
430  FOR Q8 = 1 TO NN STEP 1
432 SA(Q8,1) = S(Q8,1)
```

B.3 MULTIN2A Continued

```
 434 DU(Q8,1) = D(Q8,1)
 436 DL(Q8,1) = Y(Q8,1)
 438  NEXT Q8
 499  REM  SET ALPHA AND BETA
 500  REM  SS N,M,H,G,I
 505  GOSUB 3000
 799  REM  ****************************************
 800  REM  LOOPS HERE EACH ITERATION
 999  REM  SET ALPHA AND BETA
1000  REM  UPDATE G,I,V
1005  GOSUB 3050
1350  IF T >  = TM THEN  GOTO 19000
1355 T = T + DT
1360  REM  NEXT ITERATION
1365  GOTO 800
1999  REM  ############################
2000 : REM  GET STIMULUS VALUES
2001  PRINT "MOD NAME MULTIN2A"
2002  PRINT "ENTER STIMULUS VALUES"
2003  PRINT "11/18/91 - TRIAL1 DATA"
2008  FOR Q0 = 1 TO NN STEP 1
2010  FOR Q = 1 TO Z1 STEP 1
2015  PRINT "STIM VALUES FOR N=";Q0;"AND STIM=";Q
2020  INPUT "SA= ";S(Q0,Q)
2022  INPUT "DU= ";D(Q0,Q)
2024  INPUT "DL= ";Y(Q0,Q)
2026  NEXT Q
2027  NEXT Q0
2530  INPUT "MAX SIMULATION TIME";TM
2700  RETURN
3000  REM  ++++++++++++++++##############################++
      ###+#+#
3001  REM  FIRST GOSUB SUB SET
3005  FOR Q4 = 1 TO NN STEP 1
3007 N1 = Q4
3010  GOSUB 8000
3015  GOSUB 8200
3017  NEXT Q4
3020  RETURN
3050  REM  ()()()()()()()()()()()()()()()()()()()()()()()()(
      )()()
3051  REM  SECOND GOSUB SET
3053  FOR Q5 = 1 TO NN STEP 1
3054 N1 = Q5
3055  GOSUB 8000
3060  GOSUB 9000
3062  IF MV(Q5) <  = RS THEN F(Q5) = 0
3064  IF F(Q5) = 1 THEN  GOTO 3068
3066  IF MV(Q5) >  = 0.0 THEN  GOSUB 4000
3068  NEXT Q5
3070  RETURN
4000  REM  ----------------------------------------
4001  REM  ASSIGN WORKING VALUES HERE
```

B.3 MULTIN2A Continued

```
4002  REM  CONTROLS INTERCONECTIONS BY CHOICE OF S,D AND Y
4003  REM  EACH SET PROGRAMS WHAT IS EFFECTED BY THAT PARICULAR N
     T EFFECTS IT
4008  IF N1 = 1 THEN  GOTO 4030
4009  IF N1 = 2 THEN  GOTO 4060
4010  IF N1 = 3 THEN  GOTO 4090
4011  IF N1 = 4 THEN  GOTO 4120
4012  IF N1 = 5 THEN  GOTO 4150
4013  IF N1 = 6 THEN  GOTO 4180
4014  IF N1 = 7 THEN  GOTO 4210
4015  IF N1 = 8 THEN  GOTO 4240
4016  IF N1 = 9 THEN  GOTO 4270
4020  RETURN
4030  REM  PROGRAMMING FOR N1=1
4032 SA(5,2) = S(5,2)
4034 DL(5,2) = Y(5,2) + T
4036 DU(5,2) = D(5,2) + Y(5,2) + T
4058 F(1) = 1
4059  GOTO 4020
4060  REM  PROGRAMMING FOR N1=2
4062 SA(9,3) = S(9,3)
4064 DL(9,3) = Y(9,3) + T
4066 DU(9,3) = D(9,3) + Y(9,3) + T
4088 F(2) = 1
4089  GOTO 4020
4090  REM  PROGRAMMING FOR N1=3
4092 SA(4,4) = S(4,4)
4094 DL(4,4) = Y(4,4) + T
4096 DU(4,4) = D(4,4) + Y(4,4) + T
4118 F(3) = 1
4119  GOTO 4020
4120  REM  PROGRAMMING FOR N1=4
4122 SA(2,4) = S(2,4)
4124 DL(2,4) = Y(2,4) + T
4126 DU(2,4) = D(2,4) + Y(2,4) + T
4148 F(4) = 1
4149  GOTO 4020
4150  REM  PROGRAMMING FOR N1=5
4152 SA(3,5) = S(3,5)
4154 DL(3,5) = Y(3,5) + T
4156 DU(3,5) = D(3,5) + Y(3,5) + T
4178 F(5) = 1
4179  GOTO 4020
4180  REM  PROGRAMMING FOR N1=6
4182 SA(1,6) = S(1,6)
4184 DL(1,6) = Y(1,6) + T
4186 DU(1,6) = D(1,6) + Y(1,6) + T
4208 F(6) = 1
4209  GOTO 4020
4210  REM  PROGRAMMING FOR N1=7
4212 SA(8,8) = S(8,8)
4214 DL(8,8) = Y(8,8) + T
```

B.3 MULTIN2A Continued

```
4216 DU(8,8) = D(8,8) + Y(8,8) + T
4238 F(7) = 1
4239  GOTO 4020
4240  REM  PROGRAMMING FOR N1=8
4242 SA(6,8) = S(6,8)
4244 DL(6,8) = Y(6,8) + T
4246 DU(6,8) = D(6,8) + Y(6,8) + T
4268 F(8) = 1
4269  GOTO 4020
4270  REM  PROGRAMMING FOR N1=9
4272 SA(7,9) = S(7,9)
4274 DL(7,9) = Y(7,9) + T
4276 DU(7,9) = D(7,9) + Y(7,9) + T
4298 F(9) = 1
4299  GOTO 4020
7999  REM  ++++++++++++++++++++++++++++++++++
8000  REM  RATE CONSTANTS FOR VOLTS
8020 V(N1) =  - MV(N1) - 90
8029  REM  AVOID 1/0
8030  IF V(N1) + 25 = 0 THEN V(N1) =  - 25.1
8040  IF V(N1) + 10 = 0 THEN V(N1) =  - 10.1
8060 BN(N1) = 0.125 *  EXP (V(N1) / 80)
8070 BM(N1) = 4 *  EXP (V(N1) / 18)
8081 BH(N1) = 1 / ( EXP ((V(N1) + 30) / 10) + 1)
8100 AN(N1) = 0.01 * (V(N1) + 10) / ( EXP ((V(N1) + 10) / 10) - 1)
8110 AM(N1) = 0.1 * (V(N1) + 25) / ( EXP ((V(N1) + 25) / 10) - 1)
8120 AH(N1) = 0.07 *  EXP (V(N1) / 20)
8130  RETURN
8200  REM  STEADY STATE VARIABLES
8210 N(N1) = AN(N1) / (AN(N1) + BN(N1))
8220 M(N1) = AM(N1) / (AM(N1) + BM(N1))
8230 H(N1) = AH(N1) / (AH(N1) + BH(N1))
8500  REM  INITIAL G AND I
8520  LET GK(N1) = 36 * N(N1) * N(N1) * N(N1) * N(N1)
8525  LET GN(N1) = 120 * M(N1) * M(N1) * M(N1) * H(N1)
8530 IK(N1) = GK(N1) * (V(N1) - 12)
8535 IN(N1) = GN(N1) * (V(N1) + 115)
8540 IL(N1) = 0.3 * (V(N1) + 10.6)
8545 IT(N1) = IK(N1) + IN(N1) + IL(N1)
8550  RETURN
8999  REM  //////////////////////////////////////////////////
9000  REM  UPDATE CONDUCTANCES
9020 DN(N1) = AN(N1) * (1 - N(N1)) - BN(N1) * N(N1)
9030 DM(N1) = AM(N1) * (1 - M(N1)) - BM(N1) * M(N1)
9040 DH(N1) = AH(N1) * (1 - H(N1)) - BH(N1) * H(N1)
9060 N(N1) = N(N1) + DN(N1) * DT
9070 M(N1) = M(N1) + DM(N1) * DT
9080 H(N1) = H(N1) + DH(N1) * DT
9100 GK(N1) = 36 * N(N1) * N(N1) * N(N1) * N(N1)
9110 GN(N1) = 120 * M(N1) * M(N1) * M(N1) * H(N1)
9200 IK(N1) = GK(N1) * (V(N1) - 12)
9210 IN(N1) = GN(N1) * (V(N1) + 115)
```

B.3 MULTIN2A Continued

```
9215 IL(N1) = 0.3 * (V(N1) + 10.6)
9220 IT(N1) = IK(N1) + IN(N1) + IL(N1)
9554  FOR Q3 = 1 TO Z1 STEP 1
9556  IF T > DL(N1,Q3) AND T < DU(N1,Q3) THEN IT(N1) = IT(N1) +
      SA(N1,Q3)
9558  NEXT Q3
9700  REM  NEW VOLTAGE
9710 DV(N1) = IT(N1) * DT
9720 V(N1) = V(N1) - DV(N1)
9730 MV(N1) =  - V(N1) - 90
9732  IF T >  = TP THEN  GOTO 9735
9733  RETURN
9734  REM  PRINT PROGRAMMING
9735  PRINT "TIME= ";T;" N= ";N1;
9737  PRINT "MV= ";MV(N1);"V= ";V(N1)
9739  IF N1 = 9 THEN TP = TP + 0.2
9740  GOTO 9733
9899  REM ++++++++++++++++++++++++++++++++++++++++++++
19000  INPUT "RESTART SIM. Y/N";A$
19002  IF A$ = "Y" THEN  GOTO 220
19999  END
```

B.4 MULTIN2B Program

```
10  REM  AXON ACTION POTENTIAL
20  REM      HODGKIN AND HUXLEY
30  REM      J.OF PHYSIO 177:500 (1952)
40  REM  THIS VERSION COMPUTES EXP() AS USED
50  REM      ABSOLUTE CONDUCTANCES
52  REM  THIS IS A MODIFIED HH MODEL
54  REM  NO GRAPHICS PRINTS DATA ONLY
56  REM  USER DETERMINED NETWORK USING
57  REM  NEURON VALUES (S, D, Y)
58  REM  MODEL NAME: MULTIN2B
59  REM  TRIAL 2 RANDOM EXCIT CONNECTIONS
60  REM  UPDATE 11/12,17/91
63  REM  3 BY 3 NETWORK
64  REM  9 STIMS PER N, NO FEEDBACK ONTO ITSELF BUT VCAN BE ADDED
     R
66  REM  STIMS FROM OTHER N'S AND 1 EXTRN
99  REM  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
100  REM  INITIALIZE VALUES
109  REM  NN = NUMBER OF NEURONS
110 NN = 9
111  REM  RS = RESET THRESHHOLD VAL.
112 RS =  - 95.0
114  REM  Z1=NUMBER OF STIMS/N
116 Z1 = 9
192 DT = 1 / 25
195  DIM S(10,10),D(10,10),Y(10,10)
200  DIM SA(10,10),DU(10,10),DL(10,10)
204  DIM V(10),MV(10),F(10)
206  DIM N(10),M(10),H(10)
208  DIM GK(10),GN(10)
210  DIM IK(10),IN(10),IT(10),IL(10)
212  DIM DN(10),DM(10),DH(10)
214  DIM BN(10),BM(10),BH(10)
216  DIM AN(10),AM(10),AH(10)
220 T = 0.0
222 TP = 0.0
225  FOR Q6 = 1 TO NN STEP 1
227  FOR Q7 = 1 TO Z1 STEP 1
229 SA(Q6,Q7) = 0.0
231 DL(Q6,Q7) = 0.0
233 DU(Q6,Q7) = 0.0
235  NEXT Q7
237 MV(Q6) =  - 90
239 F(Q6) = 0
241  NEXT Q6
305 BN(N1) = BM(N1) = BH(N1) = 0
310  REM  CHANGE TV TO SET THRESHOLD VALUE
420  REM  </>\</>\</>\</>\</>\</>\>/>\>
421  REM  START OF PROGRAMMING
425  REM  GET USER VALUES
427  GOSUB 2000
430  FOR Q8 = 1 TO NN STEP 1
```

B.4 MULTIN2B Continued

```
 432 SA(Q8,1) = S(Q8,1)
 434 DU(Q8,1) = D(Q8,1)
 436 DL(Q8,1) = Y(Q8,1)
 438  NEXT Q8
 499  REM  SET ALPHA AND BETA
 500  REM  SS N,M,H,G,I
 505  GOSUB 3000
 799  REM  ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
 800  REM  LOOPS HERE EACH ITERATION
 999  REM  SET ALPHA AND BETA
1000  REM  UPDATE G,I,V
1005  GOSUB 3050
1350  IF T >  = TM THEN  GOTO 19000
1355 T = T + DT
1360  REM  NEXT ITERATION
1365  GOTO 800
1999  REM  ############################
2000 : REM  GET STIMULUS VALUES
2001  PRINT "MODEL NAME - MULTIN2B"
2002  PRINT "ENTER STIMULUS VALUES"
2003  PRINT "NAME: MULTIN2B 3X3"
2004  INPUT "TODAY'S DATE IS (EG:11/22/91) ";R$
2005  INPUT "RUN NUMBER IS ";XX
2008  FOR Q0 = 1 TO NN STEP 1
2010  FOR Q = 1 TO Z1 STEP 1
2015  PRINT "STIM VALUES FOR N=";Q0;"AND STIM=";Q
2020  INPUT "STIM VALUE = ";S(Q0,Q)
2022  INPUT "DURAT VALUE = ";D(Q0,Q)
2024  INPUT "DELAY VALUE = ";Y(Q0,Q)
2026  NEXT Q
2027  NEXT Q0
2530  INPUT "MAX SIMULATION TIME";TM
2700  RETURN
3000  REM  +++++++++++++++####################################+++++++
     ###+#+#
3001  REM  FIRST GOSUB SUB SET
3005  FOR Q4 = 1 TO NN STEP 1
3007 N1 = Q4
3010  GOSUB 8000
3015  GOSUB 8200
3017  NEXT Q4
3020  RETURN
3050  REM  ()()()()()()()()()()())()()()()()()()()()()()()()()()
     ()()
3051  REM  SECOND GOSUB SET
3053  FOR Q5 = 1 TO NN STEP 1
3054 N1 = Q5
3055  GOSUB 8000
3060  GOSUB 9000
3062  IF MV(Q5) <  = RS THEN F(Q5) = 0
3064  IF F(Q5) = 1 THEN  GOTO 3068
3066  IF MV(Q5) >  = 0.0 THEN  GOSUB 4000
```

B.4 MULTIN2B Continued

```
3068  NEXT Q5
3070  RETURN
4000  REM  ---------------------------------------
4001  REM  ASSIGN WORKING VALUES HERE
4002  REM  CONTROLS INTERCONECTIONS BY CHOICE OF S,D AND Y
4003  REM  EACH SET PROGRAMS WHAT IS EFFECTED BY THAT PARICULAR N
     T EFFECTS IT
4008  IF N1 = 1 THEN  GOTO 4030
4009  IF N1 = 2 THEN  GOTO 4060
4010  IF N1 = 3 THEN  GOTO 4090
4011  IF N1 = 4 THEN  GOTO 4120
4012  IF N1 = 5 THEN  GOTO 4150
4013  IF N1 = 6 THEN  GOTO 4180
4014  IF N1 = 7 THEN  GOTO 4210
4015  IF N1 = 8 THEN  GOTO 4240
4016  IF N1 = 9 THEN  GOTO 4270
4020  RETURN
4030  REM  PROGRAMMING FOR N1=1
4032 SA(2,2) = S(2,2)
4034 DL(2,2) = Y(2,2) + T
4036 DU(2,2) = D(2,2) + Y(2,2) + T
4038 SA(8,2) = S(8,2)
4040 DL(8,2) = Y(8,2) + T
4042 DU(8,2) = D(8,2) + Y(8,2) + T
4058 F(1) = 1
4059  GOTO 4020
4060  REM  PROGRAMMING FOR N1=2
4062 SA(6,3) = S(6,3)
4064 DL(6,3) = Y(6,3) + T
4066 DU(6,3) = D(6,3) + Y(6,3) + T
4068 SA(9,3) = S(9,3)
4070 DL(9,3) = Y(9,3) + T
4072 DU(9,3) = D(9,3) + Y(9,3) + T
4088 F(2) = 1
4089  GOTO 4020
4090  REM  PROGRAMMING FOR N1=3
4092 SA(5,4) = S(5,4)
4094 DL(5,4) = Y(5,4) + T
4096 DU(5,4) = D(5,4) + Y(5,4) + T
4098 SA(9,4) = S(9,4)
4100 DL(9,4) = Y(9,4) + T
4102 DU(9,4) = D(9,4) + Y(9,4) + T
4118 F(3) = 1
4119  GOTO 4020
4120  REM  PROGRAMMING FOR N1=4
4122 SA(7,5) = S(7,5)
4124 DL(7,5) = Y(7,5) + T
4126 DU(7,5) = D(7,5) + Y(7,5) + T
4128 SA(8,5) = S(8,5)
4130 DL(8,5) = Y(8,5) + T
4132 DU(8,5) = D(8,5) + Y(8,5) + T
4148 F(4) = 1
```

B.4 MULTIN2B Continued

```
 4149  GOTO 4020
 4150  REM  PROGRAMMING FOR N1=5
 4152 SA(1,5) = S(1,5)
 4154 DL(1,5) = Y(1,5) + T
 4156 DU(1,5) = D(1,5) + Y(1,5) + T
 4158 SA(4,5) = S(4,5)
 4160 DL(4,5) = Y(4,5) + T
 4162 DU(4,5) = D(4,5) + Y(4,5) + T
 4178 F(5) = 1
 4179  GOTO 4020
 4180  REM  PROGRAMMING FOR N1=6
 4182 SA(4,6) = S(4,6)
 4184 DL(4,6) = Y(4,6) + T
 4186 DU(4,6) = D(4,6) + Y(4,6) + T
 4188 SA(5,6) = S(5,6)
 4190 DL(5,6) = Y(5,6) + T
 4192 DU(5,6) = D(5,6) + Y(5,6) + T
 4208 F(6) = 1
 4209  GOTO 4020
 4210  REM  PROGRAMMING FOR N1=7
 4212 SA(5,7) = S(5,7)
 4214 DL(5,7) = Y(5,7) + T
 4216 DU(5,7) = D(5,7) + Y(5,7) + T
 4218 SA(6,7) = S(6,7)
 4220 DL(6,7) = Y(6,7) + T
 4222 DU(6,7) = D(6,7) + Y(6,7) + T
 4238 F(7) = 1
 4239  GOTO 4020
 4240  REM  PROGRAMMING FOR N1=8
 4242 SA(3,8) = S(3,8)
 4244 DL(3,8) = Y(3,8) + T
 4246 DU(3,8) = D(3,8) + Y(3,8) + T
 4248 SA(9,9) = S(9,9)
 4250 DL(9,9) = Y(9,9) + T
 4252 DU(9,9) = D(9,9) + Y(9,9) + T
 4268 F(8) = 1
 4269  GOTO 4020
 4270  REM  PROGRAMMING FOR N1=9
 4272 SA(1,9) = S(1,9)
 4274 DL(1,9) = Y(1,9) + T
 4276 DU(1,9) = D(1,9) + Y(1,9) + T
 4278 SA(7,9) = S(7,9)
 4280 DL(7,9) = Y(7,9) + T
 4282 DU(7,9) = D(7,9) + Y(7,9) + T
 4298 F(9) = 1
 4299  GOTO 4020
 7999  REM  ++++++++++++++++++++++++++++++++
 8000  REM  RATE CONSTANTS FOR VOLTS
 8020 V(N1) =  - MV(N1) - 90
 8029  REM  AVOID 1/0
 8030  IF V(N1) + 25 = 0 THEN V(N1) =  - 25.1
 8040  IF V(N1) + 10 = 0 THEN V(N1) =  - 10.1
```

B.4 MULTIN2B Continued

```
8060 BN(N1) = 0.125 *  EXP (V(N1) / 80)
8070 BM(N1) = 4 *  EXP (V(N1) / 18)
8081 BH(N1) = 1 / ( EXP ((V(N1) + 30) / 10) + 1)
8100 AN(N1) = 0.01 * (V(N1) + 10) / ( EXP ((V(N1) + 10) / 10) - 1)
8110 AM(N1) = 0.1 * (V(N1) + 25) / ( EXP ((V(N1) + 25) / 10) - 1)
8120 AH(N1) = 0.07 *  EXP (V(N1) / 20)
8130  RETURN
8200  REM  STEADY STATE VARIABLES
8210 N(N1) = AN(N1) / (AN(N1) + BN(N1))
8220 M(N1) = AM(N1) / (AM(N1) + BM(N1))
8230 H(N1) = AH(N1) / (AH(N1) + BH(N1))
8500  REM  INITIAL G AND I
8520  LET GK(N1) = 36 * N(N1) * N(N1) * N(N1) * N(N1)
8525  LET GN(N1) = 120 * M(N1) * M(N1) * M(N1) * H(N1)
8530 IK(N1) = GK(N1) * (V(N1) - 12)
8535 IN(N1) = GN(N1) * (V(N1) + 115)
8540 IL(N1) = 0.3 * (V(N1) + 10.6)
8545 IT(N1) = IK(N1) + IN(N1) + IL(N1)
8550  RETURN
8999  REM  ////////////////////////////////////////////////
9000  REM  UPDATE CONDUCTANCES
9020 DN(N1) = AN(N1) * (1 - N(N1)) - BN(N1) * N(N1)
9030 DM(N1) = AM(N1) * (1 - M(N1)) - BM(N1) * M(N1)
9040 DH(N1) = AH(N1) * (1 - H(N1)) - BH(N1) * H(N1)
9060 N(N1) = N(N1) + DN(N1) * DT
9070 M(N1) = M(N1) + DM(N1) * DT
9080 H(N1) = H(N1) + DH(N1) * DT
9100 GK(N1) = 36 * N(N1) * N(N1) * N(N1) * N(N1)
9110 GN(N1) = 120 * M(N1) * M(N1) * M(N1) * H(N1)
9200 IK(N1) = GK(N1) * (V(N1) - 12)
9210 IN(N1) = GN(N1) * (V(N1) + 115)
9215 IL(N1) = 0.3 * (V(N1) + 10.6)
9220 IT(N1) = IK(N1) + IN(N1) + IL(N1)
9554  FOR Q3 = 1 TO Z1 STEP 1
9556  IF T > DL(N1,Q3) AND T < DU(N1,Q3) THEN IT(N1) = IT(N1) +
      SA(N1,Q3)
9558  NEXT Q3
9700  REM  NEW VOLTAGE
9710 DV(N1) = IT(N1) * DT
9720 V(N1) = V(N1) - DV(N1)
9730 MV(N1) =  - V(N1) - 90
9732  IF T >  = TP THEN  GOTO 9735
9733  RETURN
9734  REM  PRINT PROGRAMMING
9735  PRINT "TIME= ";T;" N= ";N1;
9737  PRINT "MV= ";MV(N1);"V= ";V(N1)
9739  IF N1 = 9 THEN TP = TP + 0.2
9740  GOTO 9733
9899  REM ++++++++++++++++++++++++++++++++++++++++++
19000  INPUT "RESTART SIM. Y/N";A$
19002  IF A$ = "Y" THEN  GOTO 220
19999  END
```

APPENDIX C
PROG SERIES PROGRAMS

This series is the simulation of the Glass & Mackey Phase Lock Biological Oscillation Model. The program variable are listed below:

TABLE:	PROG SERIES (NOT DIMENSIONLESS)
SYMBOL	VARIABLE DEFINITION
DT	CHANGE IN TIME
K	SINUSOID AMPLITUDE
PA	PHASE ANGLE
PC	POSITIVE CONSTANT
R	RATIO
RA	Y/N TRY AGAIN
SO	INITIAL VALUE OF SINUSOID
SS	VALUE OF SINUSOID
T	TIME
T3	MAXIMUM TIME FOR RUNNING THE PROGRAM
TP	TIME WHEN X = SS
VV	VALUE OF SIN(VV)
W	ANGULAR FREQUENCY
X	ACTIVITY
XO	INITIAL VALUE ACTIVITY
	(DIMENSIONLESS VARIABLES)
A	STABILITY TIME
DV	DUMMY VARIABLE FOR HOLDING DATA
K2	ABS (LA * (N-1)
KK	K VALUE
LA	POSITIVE CONSTANT
N	PHASE RATIO
S2	INITIAL SS VALUE
TT	TIME
TU	TIME UNSTABLE
X2	INITIAL ACTIVITY VALUE
XX	ACTIVITY VALUE

PROG3B (shown on the next page) is the integrate and fire simulation described in Chapter 5.3. The program is broken down as follows:

(1 - 155) - REM statements that describe the program and give user information;
(200 - 280) - User input values entered here;
(299 - 325) - Initial Dimensionless values calculated
(335 - 405) - " " Activity and SMT values calculated and printed;
(406 - 475) - Updated values are calculated and printed;
(500 - 1004) - Retry Option

C.1 PROG3B Program

```
1   REM  THESIS PROJECT
2   REM  MARY ELLEN ALEKSZA (NJIT)
3   REM  BIOLOGICAL OSCILLATORS
4   REM  GLASS AND MACKEY'S SIMPLE MODEL
5   REM  SEE THESIS LOG FOR DETAILS
7   REM  MODEL STARTED 09 APRIL 1990
8   REM  DR S REISMAN DR W TAPP
10  REM  DIMENSIONLESS PARAMETER MODEL
15  REM  PROGRAM NAMED PROG#
100 REM  START OF THE PROGRAM
105 REM  ************************
110 REM  PART 1 PROGRAM VARIABLES
112 REM  K = AMP OF SINUSOID
113 REM  SO = INIT. VALUE OF SINE
114 REM  T = TIME
115 REM  DT = CHANGE IN TIME
116 REM  T2 = TIME XX=S3
117 REM  X = INCREASING ACTIVITY
118 REM  SS = SINUSOID
119 REM  XO = INIT.VALUE ACTIVITY
120 REM  PC = POSITIVE CONSTANT
122 REM  W=FREQUENCY
124 REM  PA = PHASE ANGLE = 0
126 REM  VV = THE VALUE OF THE SIN()
128 REM  T3 = TIME TO RUN PROGRAM TOO
132 REM  0.0 <= X < SS
134 REM  0<=K<SS
135 REM  W>0 AND PA>0
138 REM  R = F(SIN MOD) ; F(UNFTD OSC)
140 REM  X2= DIMLESS INITIAL X VALUE
142 REM  S2= DIMLESS INITIAL SS VALUE
144 REM  KK= DIMLESS K VALUE
146 REM  LA= DIMLESS PC VALUE
148 REM  S3= DIMLESS SS VALUE
150 REM  TT= DIMLESS T VALUE
155 REM  XX = DIMLESS VALUE OF X
157 REM  N= PHASE RATION
159 REM  X2= ABS (LA * N -1)
161 REM  A = STABILITY TIME (ALPHA)
163 REM  TU = TIME UNSTABLE
165 REM  TS = TIME STABLE
167 REM  DU = DUMMY VARIABLE
170 REM  KK >= (LA=N -1)
200 REM  ************************
201 REM  PART 2 INPUT DATA
205 PRINT "WHAT IS XO VALUE ";
210 INPUT XO
215 PRINT "WHAT IS SO VALUE ";
220 INPUT SO
225 PRINT "WHAT IS PC VALUE ";
230 INPUT PC
245 PRINT "WHAT IS W VALUE ";
250 INPUT W
255 PRINT "WHAT IS K VALUE ";
260 INPUT K
265 PRINT "WHAT IS DT VALUE";
270 INPUT DT
275 PRINT "WHAT IS T3 VALUE ";
280 INPUT T3
282 PRINT "WHAT IS THE N VALUE ";
284 INPUT N
285 LET T = 0
298 REM ************************
299 REM  PART 3 DIMLESS CALC. INITIAL
300 LET X2 = XO / SO
305 LET S2 = 1
310 LET KK = K / SO
315 LET LA = (6.28319 * PC) / (W * SO)
320 REM  R = 1/LA
322 LET R = (W * SO) / (6.28319 * PC)
325 PRINT "R = ";R
327 LET X2 = ABS (LA * N - 1)
330 IF KK > = X2 THEN  GOTO 350
333 PRINT "KK IS NOT LARGE ENOUGH. TRY AGAIN ";
335 INPUT K
337 GOTO 310
345 REM  ****************************
346 REM  PART 4 XX AND S3 CALCS
350 REM
351 LET T = DT
352 LET NN = 1 / N
355 LET TT = (W * T) / 6.28319
360 LET XX = X2 + (NN * LA * TT)
365 LET VV = (6.28319 * TT)
370 LET SS = 1.0 + KK *  SIN (VV)
375 IF XX > = SS THEN  GOTO 395
380 LET T = T + DT
385 IF T > = T3 THEN  GOTO 999
390 GOTO 355
395 PRINT "XX VALUE = ";XX;"  ";
400 PRINT "SS VALUE = ";SS;"  ";
404 PRINT "T VALUE = ";T;"  ";
405 PRINT "TT VALUE = ";TT
406 REM  ************************
407 REM  PART 5 XX AND S3 FINAL CALC.
410 LET T2 = TT
415 LET XX = 0.0
420 LET T = T + DT
425 IF T > = T3 THEN  GOTO 999
430 LET TT = (W * T) / 6.28319
435 LET XX = (NN * LA) * (TT - T2)
440 LET VV = 6.28319 * TT
445 LET SS = 1 + KK *  SIN (VV)
450 IF XX > = S3 THEN  GOTO 460
455 GOTO 420
460 PRINT "XX= ";XX;"  ";
465 PRINT "S3= ";S3;"  ";
470 PRINT "TT= ";TT;"  ";
472 PRINT "T= ";T
475 GOTO 410
500 REM  ************************
501 REM  END OF CALCULATIONS
999 PRINT "RUN AGAIN (1=Y,2=N)";
1000 INPUT RA
1002 IF RA = 1 THEN  GOTO 100
1004 END
```

APPENDIX D
INSTRUCTIONS FOR EXECUTING PROGRAMS

All programs are written in BASIC and are to be run on an Apple 2E enhanced (128K) using ProDOS 3.3. The printer used is an IMAGEWRITER I. All programs are started the same way as follows:

1. Insert program disk into disk drive.
2. Turn on machine
3. Make sure CAPS LOCK on
4. Type CATALOG to see list of programs on disk
5. Load desired model (e.g. LOAD MULTIN1A)
6. Type LIST to view programming
7. Type RUN to execute model
8. CONTL S - controls the screen scrolling
9. CONTL C - stops the program (BREAK)
10. Program will display on the screen automatically
11. For printer display
    a. Type PR#1 prior to running the program
    b. Type RUN - program will display on printer only
    c. Type PR#0 when complete to return to screen

I. MULTIN SERIES:

12. Enter the date after first prompt; (e.g. TODAYS DATE: 5/5/92)
13. Enter run number next; (The run number represents a number assigned to the data set run on a particular day. This number is for record keeping purposes only.)
14. Enter neuron information next: stimulus amplitude (S/SA), duration (D/DU) and the delay (Y/DL); (The header will indicate which neuron and stimulus number is being entered)

D. **Program Instructions Continued:**

15. Enter the maximum run time next;

16. Enter Y/N to continue after the program is completed.

II. **HHMOD SERIES:**

[HHMOD6D]

12. Enter the number of stimuli/neuron;

13. Enter stimuli information (amp, duration, delay);

14. Enter the N2 to N1 delay;

15. Enter the maximum simulation time;

16. Enter y/n to restart the simulation.

[HHMODEL1]

12. Enter stimulus 1 amplitude;

13. Enter stimulus 1 duration;

14. Enter delay time between stimuli;

15. Enter stimulus 2 amplitude;

16. Enter stimulus 2 duration;

17. CONTL C is used to end program.

REFERENCES

1. Hodgkin, A. L. and A. F. Huxley. "A Quantitative Description of Membrane Current and its Application to Conduction and Excitation in Nerve". The Journal of Physiology, Vol. 117, March 10, 1952.

2. Randall, J. E. Microcomputers and Physiological Simulation. Addison-Wesley Pub. Co., Cambridge, MA. 1983.

3. Tortora, G. and N. P. Anagnostakos. Principles of Anatomy & Physiology, 5th. ed. Harper & Row, Pub., New York, NY. 1987.

4. Bahill, T. A. Bioengineering Biomedical, Medical and Clinical Engineering. Prentice-Hall, Inc., Englewood Cliffs, NJ. 1981.

5. Stanley, J. and S. Luedeking. Introduction to Neural Networks: Computer Simulations of Biological Intelligence. California Scientific Software, CA. 1989.

6. Lippmann, R. P. "An Introduction To Computing With Neural Nets". IEEE ASSP Magazine. April, 1987.

7. Nilsson, J.W. Electric Circuits. Addison-Wesley Pub. Co., Cambridge, MA. 1983.

8. Traub, R.D. and R. Llina. "Hippacampal Pyramidial Cells: Significance of Dendrite Ionic Conductances for Neuronal Functions and Epilepogenesis". Journal of Neurophysiology, Vol. 42, No. 2, March, 1979.

9. Traub, R.D. and K. S. Wong. "Synchronized Burst Discharge in Disinhibited Hippocampal Slice: Model of Cellular Mechanics". Journal of Neurophysiology, Vol. 49, No. 2, February, 1983.

10. Glass, L., C. Graves, G. Petrillo and M. C. Mackey. "Unstable Dynamics of a Periodically Driven Oscillator in the Presence of Noise". The Journal of Theoretical Biology, Vol. 86, 1980.

11. Glass, L. and M. C. Mackey. From Clocks to Chaos: The Rhythms of Life. Princeton University Press, Inc., Princton, NJ. 1988.

12. Glass, L. and M. C. Mackey. "A Simple Model For Phase Locking of Biological Oscillators". Journal of Mathematical Biology, Vol. 7, 1979.