Annals of Biomedical Engineering, Vol. 29, pp. 897–907, 2001
Printed in the USA. All rights reserved.

0090-6964/2001/29(10)/897/11/$15.00
Copyright © 2001 Biomedical Engineering Society

Real-Time Linux Dynamic Clamp: A Fast and Flexible Way to Construct Virtual Ion Channels in Living Cells

ALAN D. DORVAL,[1] DAVID J. CHRISTINI,[2] AND JOHN A. WHITE[1]

[1]Department of Biomedical Engineering, Center for BioDynamics, Boston University, Boston, MA
[2]Division of Cardiology, Weill Medical College of Cornell University, 520 E. 70th Street, New York, NY

(Received 4 April 2001; accepted 18 July 2001)

Abstract—We describe a system for real-time control of biological and other experiments. This device, based around the Real-Time Linux operating system, was tested specifically in the context of dynamic clamping, a demanding real-time task in which a computational system mimics the effects of nonlinear membrane conductances in living cells. The system is fast enough to represent dozens of nonlinear conductances in real time at clock rates well above 10 kHz. Conductances can be represented in deterministic form, or more accurately as discrete collections of stochastically gating ion channels. Tests were performed using a variety of complex models of nonlinear membrane mechanisms in excitable cells, including simulations of spatially extended excitable structures, and multiple interacting cells. Only in extreme cases does the computational load interfere with high-speed ''hard'' real-time processing (i.e., real-time processing that never falters). Freely available on the worldwide web, this experimental control system combines good performance, immense flexibility, low cost, and reasonable ease of use. It is easily adapted to any task involving real-time control, and excels in particular for applications requiring complex control algorithms that must operate at speeds over 1 kHz. *© 2001 Biomedical Engineering Society.* [DOI: 10.1114/1.1408929]

Keywords—Stochastic, Real-time Computing, Experimental control, Electrophysiology.

INTRODUCTION

Electrophysiologists have long relied on two methods to probe the electrical mechanisms at work within excitable cells: the current clamp and the voltage clamp. In current clamp, a transmembrane electrode provides current to a cell while the resulting membrane potential is recorded. In voltage clamp, advanced circuitry controls the membrane potential, usually at a constant level, while transmembrane current is recorded. The current clamp technique reveals some of the electrical behaviors exhibited by cells. The voltage clamp technique allows the experimentalist to build accurate mathematical models of some of the biophysical properties responsible for these behaviors.

More recently, a third electrophysiological technique has been developed.[16,18,19] In this technique, often called the ''dynamic clamp'' method, the experimental system controls membrane conductance rather than transmembrane current or potential. Because the dynamic clamp method allows one to mimic biological effects in the living cells, it allows the experimentalist to test new kinds of specific, quantitative hypotheses directly in experiments. For example, dynamic clamping has been used to verify the hypothetical effects of neuromodulators;[18] show how particular conductances can account for changes in neural firing modes,[8,12] spike widths,[14] or spike timing;[13] search for input characteristics that optimize some aspect of postsynaptic behavior;[9,15] and study how particular parameters affect synchronization properties in ''virtual networks'' consisting of one or more biological excitable cells interacting with each other, or with modeled counterparts, in user-specified ways.[3,4,20]

Dynamic clamp is fundamentally more difficult to implement than its forebears. Current clamp requires only a high-quality current source. Voltage clamp requires a fast control system that minimizes the difference between measured membrane potential and a user-supplied ''command'' potential. A dynamic clamp system must calculate the value of a time-varying artificial membrane conductance. Often, this conductance comes from the real-time solution to a set of nonlinear differential equations that depend on time and membrane potential. Next, the system must calculate and output the amount of current that would have passed through this virtual conductance. This calculation also depends on the instantaneous value of membrane potential, and thus must be calculated in real time.

Some potential experiments require an even greater level of realism, and consequently even more complex real-time algorithms. For example, it is well established

Address correspondence to John A. White, PhD, Department of Biomedical Engineering, Boston University, 44 Cummington St., Boston, MA 02215. Electronic mail: jwhite@bu.edu

that the voltage-gated ion channels responsible for electrical behaviors in excitable cells ‘‘gate’’ (open and close) in an apparently stochastic manner.[10,17] In most cases, neuronal modelers assume implicitly that the conductance fluctuations caused by this random gating can be ignored, but in some cases this implicit assumption may not be accurate.[26] The complexity of algorithms required to represent stochastic channel gating is much higher than for deterministic excitable cells. Again most of these complex calculations must be made in real time.

Thus far, efforts to develop dynamic clamp systems have been somewhat isolated, with many labs devising problem- and hardware-specific solutions. Because of this, our goal for the present work was to develop an experimental control system for general use in dynamic clamp and other types of demanding real-time experiments. In this work, we were able to satisfy five crucial design criteria. First, the system is inexpensive. It runs under Real-Time Linux, an open-source modification of the Linux operating system that is free for nonprofit applications.[2] All required software (including ours) are free and open source; the only required hardware include a personal computer (PC), data acquisition card (DAQ) with A/D and D/A capabilities, and (for dynamic clamping) an intracellular current-clamp amplifier. Second, the system is fast. Using inexpensive hardware, the system can solve complex stochastic differential equations in real time at frequencies above 10 kHz (fast enough for most dynamic clamp applications) while simultaneously reading and writing data from and to the DAQ. Third, the system is flexible. An experimentalist with limited programming knowledge can modify our sample code to solve a wide range of problems quite easily. Fourth, our system is user friendly. A modifiable graphical user interface (GUI), adapted from previous work,[5] allows even naïve end users to control sophisticated experiments. Fifth, and most importantly, the system gives ‘‘hard real time’’ performance. A hard real-time task is one that never misses a deadline. To ensure this, all other processes (e.g., keyboard input, monitor refreshes, network communications, other programs, etc.) are suspended when the hard real-time task needs the CPU. These other processes are unpaused only after the hard real-time task returns CPU control.

In this paper, we first describe our system, which we refer to as the ‘‘Real-Time Linux Dynamic Clamp’’ (RTLDC). We then test its basic performance characteristics in an ‘‘open-loop’’ configuration, in which voltage-dependent currents are calculated and output, but have no influence on the input voltage signal. Next, we demonstrate the efficacy of the RTLDC in a series of ‘‘closed-loop’’ mock experimental designs, in which membrane voltage and membrane current influence each other in real time. Examples include models of spiking membranes with spatially uniform and nonuniform membrane potential, as well as deterministic and stochastic membrane conductances. In the last example, a virtual neuronal network is constructed and characterized. Some of this work has appeared previously in preliminary form.[23,24]

METHODS

System Design

The dynamic clamp prototype that we have created is intended to be generally useful for a large range of real-time experiments. With this in mind, much of our hardware can be substituted by similar components from other vendors. Thus, after we introduce our specific components we will henceforth refer to them in generic terms.

All instances of the dynamic clamp require a high frequency current clamp amplifier. We used the Axon Instruments, AxoClamp 2B. Used in bridge mode this amplifier allowed for simultaneous voltage recording and current application through a single electrode. The amplifier must connect to a personal computer (PC) controlled, data acquisition board (DAQ). Our amplifier was connected to a National Instruments, PCI-MIO-16XE-50 data acquisition board. This DAQ boasts 16 channel, 16 bit analog-to-digital input (A/D) and 2 channel, 12 bit digital-to-analog output (D/A), both running at a maximum of 20 kHz.

The RTLDC was structured around an x86 architecture computer. Our DAQ board was connected to a 450 MHz, Pentium II, Dell PC with 128MB RAM. As a minimum for this dynamic clamp setup, we recommend a 586 or higher PC, with at least 64 MB RAM. The PC runs a free, open source extension to the Linux operating system, known as Real Time Linux (RTL) (http://rtlinux.com). RTL is a ‘‘hard’’ real time operating system, which means that commands will always be executed in a known amount of time. RTL provides high temporal precision on a PC, while maintaining the full functionality of the now widely supported parent operating system, Linux.[2] RTL controls the DAQ through another product of the open source community, COMEDI (http://stm.lbl.gov/comedi). A data acquisition board driver package, COMEDI provides a simple application program interface (API) for most commercially available DAQs.[5] We have written a Real-Time Linux Dynamic Clamp (RTLDC) software package that incorporates hard real-time processing with a rich graphical user interface (GUI). The interface is written using Qt (http://trolltech.com/products/qt), a set of open source libraries that can be called on any modern Linux distribution and desktop.[6] A large fraction of the original GUI was taken from a previously designed open source project, the Real

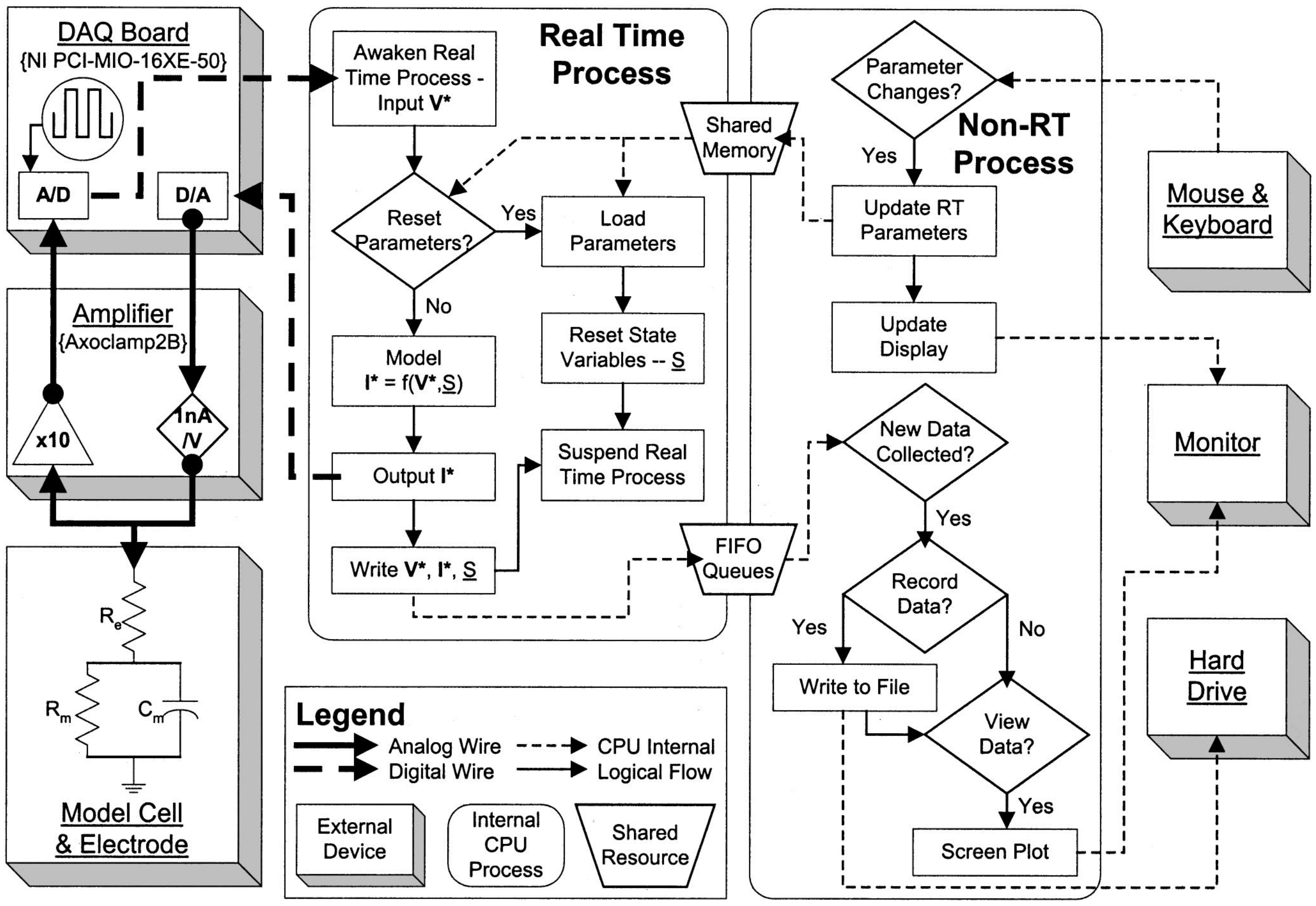
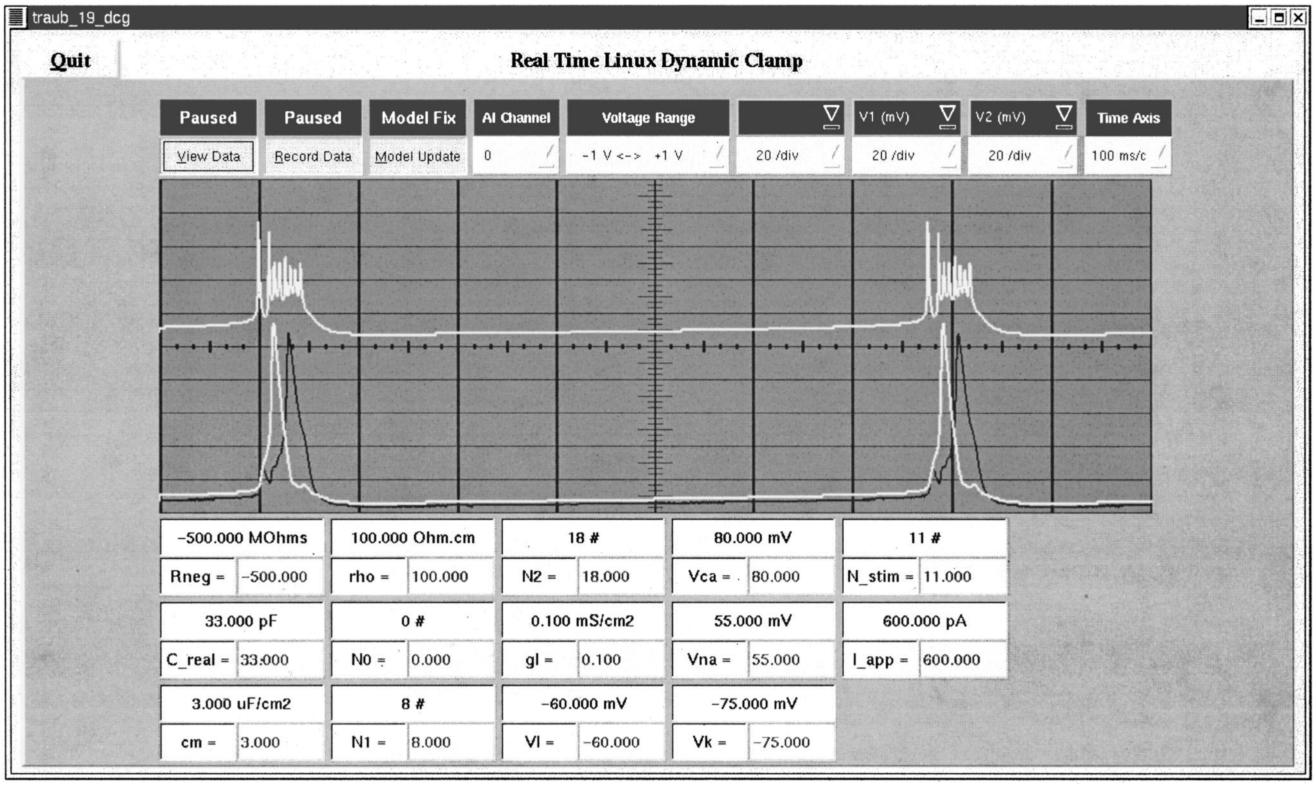


FIGURE 1. Amplifier reads the analog membrane potential from the electrode and model cell. The DAQ samples this voltage, and passes a discrete representation to the RT thread at a fixed frequency. The RT thread uses that voltage, an internal neuronal model, and parameter values from shared memory, to calculate a current. A representation of that current is passed back to the DAQ, converted to an actual current in the amplifier, and provided to the model cell. Meanwhile, the RT thread buffers the voltage, current, and other model data, in the FIFO. When RTL detects spare clock cycles, the non-RT GUI process awakens and reads the buffered data, which it displays to the monitor and saves to disk. When parameters are changed in the GUI, they are immediately stored in shared memory, from where they are used for subsequent RT thread calculations.

Time Linux Cardiac Pacing Control System[5] (http://cardiodyn.med.cornell.edu/~dchristi/software_index.html).

The RTLDC consists of two interdependent programs: a real-time thread and a non-real-time user process (Fig. 1). After initialization, the real-time thread enters an infinitely repeating loop that begins with a command to suspend the thread. The DAQ is programmed to read a voltage and report it to the real-time thread at some fixed frequency, currently less than or equal to 20 kHz. The passing of this voltage level, **V***, awakens the real-time thread, which feeds the voltage representation through a preselected model to calculate some output current value, **I***. The output current value is sent to the DAQ, which passes it on to the amplifier. The amplifier then provides the corresponding current to the cell. The real-time thread stores the voltage, current and other dynamic variables in a first-in–first-out queue (FIFO). Having reached the end of the loop, the real-time thread returns to the beginning of the loop where it is again suspended until reawakened by the DAQ on the next time step. Nothing limits the system to either reading just one voltage or writing just one current. The system readily handles two or more voltages, potentially from two different cells or two spatially remote locations on the same cell, calculating and supplying voltage-dependent currents to the two distinct locations.

The other half of the RTLDC software, a non-real-time user process, allows the experimenter to interact with the real-time thread while reporting and recording data. In the spare clock cycles between DAQ triggers, the user process reads data from the FIFO. The data being collected are displayed on a virtual oscilloscope and saved to disk. Displayed data can include transmembrane potential, the total membrane current, and any model internal value, such as the membrane voltage profile along a theoretical dendrite, the gating variables of a virtual ion channel population, or the artificial ligand



AI Channel	Voltage Range		V1 (mV)	V2 (mV)	Time Axis
0	-1 V <-> +1 V	20 /div	20 /div	20 /div	100 ms/c

-500.000 MOhms	100.000 Ohm.cm	18 #	80.000 mV	11 #
Rneg = -500.000	rho = 100.000	N2 = 18.000	Vca = 80.000	N_stim = 11.000
33.000 pF	0 #	0.100 mS/cm2	55.000 mV	600.000 pA
C_real = 33.000	N0 = 0.000	gl = 0.100	Vna = 55.000	I_app = 600.000
3.000 uF/cm2	8 #	-60.000 mV	-75.000 mV	
cm = 3.000	N1 = 8.000	Vl = -60.000	Vk = -75.000	

FIGURE 2. Graphical user interface is used to instantiate experimental recordings, change model parameter settings, and view data. The top row of widgets includes, from left to right, three GUI, or RTL thread commands, two DAQ controls, and four virtual oscilloscope settings. The boxes at the bottom are adjustable parameter values of the running model – a 19 compartment CA3 pyramidal cell, in which each compartment corresponds to a different location in the cell arborization (Ref. 21). The virtual oscilloscope shows the voltage traces of three different compartments of the bursting neuron, overlaid with a 20 mV by 100 ms grid. The apical- and basal-dendrite tip compartment membrane potentials, shown in white and black, respectively, are shifted 100 mV below the soma membrane potential, also shown in white. In this example, the basal dendrite tip membrane potential (i.e., the black trace) is the voltage across a real world RC circuit; the other potentials are CPU internal state values (see Fig. 8 for model and implementation details).

concentration adjacent to the tip of a simulated dendritic spine. The parameters of all models can be modified in soft real time, using the mouse and keyboard. Immediately after the experimenter changes a parameter, the user process informs the real-time thread, via a block of shared memory, that a model update is needed. Upon the subsequent reawakening of the real-time thread, the new parameters are loaded into the model. The experimenter interacts with the user process via the GUI, which includes the virtual oscilloscope and parameter control boxes (Fig. 2).

Initial experiments were conducted in an open-loop configuration, in which a signal generator was used to provide voltage to the DAQ [Fig. 3(A)]. Later experiments were performed in a closed-loop configuration, in which a physical RC circuit was used to integrate current provided by the amplifier. In the RC circuit, a 10 MΩ resistor simulated the electrode, and a 2 nS conductor (g_m) in parallel with a 33 pF capacitor (c_m) served as the passive cell membrane. In all closed-loop configurations, the cell membrane exists as the parallel combination of both physical and virtual components. In our mock experiments with the RC circuit, we allowed the capacitance, c_m, dictate the cell surface area. None of the models we explored included a circuit branch equivalent to g_m: a 2 nS conductance with a reversal potential of 0.0 mV. Thus, in all models we added a negative conductance, $-g_m$, to the virtual membrane, thereby canceling out the effects of the physical g_m.

Implemented models include: both deterministic- and stochastic-ion channel conductance based cells, expansive multicompartment neurons, and simple two cell networks. All of these models required the solving of differential equations—deterministic and/or stochastic, ordinary and/or partial—in real time. Because of the periodic nature of the DAQ triggering mechanism that awakened the real-time thread, solutions to the first order differential equations associated with each model were calculated by discrete time step approximations. Deterministic differential equations were approximated using first-order numerical methods which assumed a constant voltage between steps. Solutions to the stochastic differ-

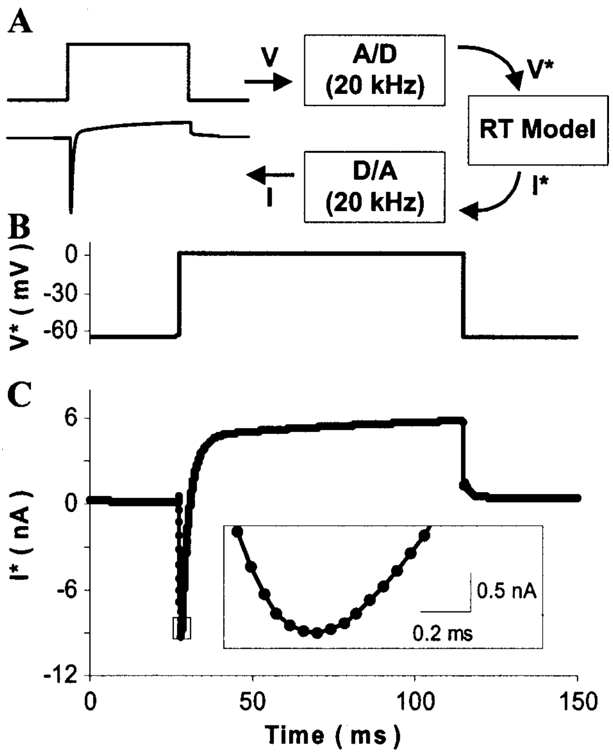


FIGURE 3. Initial experiments were performed in an open-loop configuration. (A) Diagram showing the open-loop configuration of the real-time Linux dynamic clamp system. The input signal, V, from a function generator is digitized at 20 kHz by the 16-bit A/D channel to create V*. The real-time algorithm calculates (for this example) and sums four non-linear membrane currents, generated from five first-order deterministic ODEs (Ref. 25), to create the virtual membrane current I* which is passed through the 12-bit D/A converter to give the continuous-time signal, I. (B) Plot of the internal value of applied voltage V* stepping from the model's resting potential of −65 to 0 mV. (C) The real-time membrane current I* (symbols), and the ideal membrane current (solid line), calculated analytically from the ODEs and V*. Note that I* is identical to the analytic solution at each time step.

ential equations were calculated via a first order discrete time step approximation to the Langevin description, requiring, in its most efficient implementation, only one uniformly distributed random number per time step.[7] Additional models, with arbitrarily complex integration methods, can be added by adjusting two C files for which examples have been created as templates from which to work. No deep understanding of Linux or real-time programming is required to implement new experimental paradigms using the RTLDC, available for free download from http://bme.bu.edu/ndl/rtldc.html.

RESULTS

Our dynamic clamp configuration functions at up to 20 kHz, the maximum frequency allowed by our DAQ. Sufficiently complex models that impose heavy computational loads and thus must run at slower speeds are explored below. During operation, the CPU reads a voltage from the DAQ at precisely timed periodic intervals, with some small jitter. This OS independent jitter stems from the time it takes the CPU to handle the interrupt triggered by the DAQ board read command, and is a fundamental limitation of the mother board and processor. For our system, jitter is less than 5 μs in the vast majority of cases, with maximum jitter of 5–25 μs occurring less than once per 100 000 cycles.

Without model calculations, an input/output (I/O) latency of 837 ns average, and less than 5 μs in the worst case, is required for thread overhead and writing to the DAQ. The total I/O latency depends on the number of computations the RTLDC must perform each interval. The maximum operational frequency for a given model is set by the worst case I/O latency of that model. For example, a model with a maximum I/O latency of 100 μs could not run faster than 10 kHz. We explore some of the system's capabilities and inabilities via example RTLDC models, the implementations and results of which are described in the following subsections.

Point Process Models

A conductance based, deterministic point process model of a medial entorhinal cortex (MEC) stellate neuron was constructed in the RTLDC. This model consists of four nonlinear conductance sources, generated from five first-order differential equations, and a leak conductance.[1] Experiments were performed with this model in an open-loop configuration at 20 kHz, with a signal generator provided, voltage step function serving as membrane potential (Fig. 3). To test that the output current, **I**, was being generated correctly, a second set of experiments were performed in which **I** was resampled on a second A/D channel of the DAQ (Fig. 4). Because the RTLDC was sampling two channels, the maximum speed of those experiments on our DAQ was 10 kHz. The resampled current, **I****, closely matches the theoretical current, $\mathbf{I}_{\mathbf{th}}$, on both short- and long-time scales. **I**** is nearly identical to the real-time calculated current, **I***, that the DAQ should provide, delayed by one time step [Fig. 4(C), inset]. The "true" current level, **I**, coming out of the DAQ is not precisely known, but is at all times between the values of **I*** and **I****.

To test the system for speed versus computational complexity, the Hodgkin–Huxley model of a squid giant axon membrane patch was implemented. This model consists of two nonlinear conductances generated from three first order, ordinary differential equations, and a leak conductance.[11] To observe that the system was functioning appropriately, we moved to the closed-loop configuration that includes the current clamp amplifier and physical world components of the model passive cell [Fig. 5(A)]. The maximum frequency at which the squid

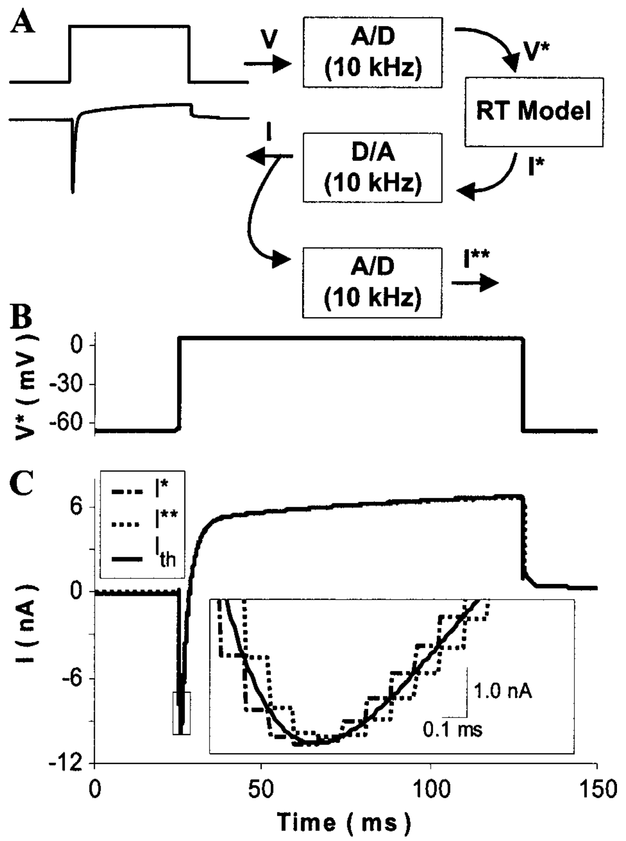


FIGURE 4. In some open-loop experiments, the output signal was resampled to verify that D/A and A/D conversion ran smoothly. (A) Diagram showing the open-loop resample configuration of the real-time Linux dynamic clamp. The DAQ board could read two A/D channels at a maximum frequency of 10 kHz. (B) Plot of the internal value of the applied voltage V* stepping from a rest potential of near −65 to 0 mV. (C) Three representations of the transmembrane current, summed from four nonlinear conductances (Ref. 25). I* is the real time, discretized current value sent to the DAQ. I** is the resampled current level. I_{th} is the theoretical current, determined analytically. I** is nearly identical to I*, but delayed by one time step. The actual current level I jumps between the levels seen in the inset, and although the precise jump time is not predictable, I is always bounded between I* and I**.

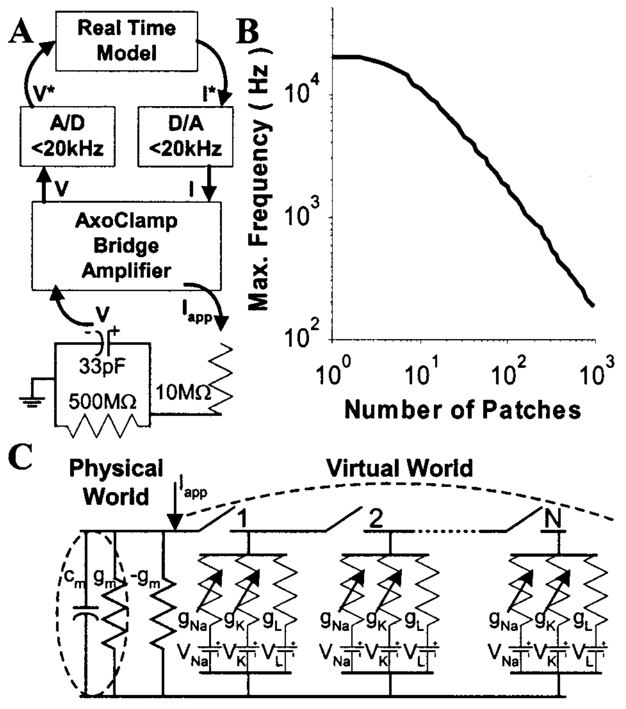


FIGURE 5. Experiments were performed to determine the maximum safe operating frequency as a function of computational load. (A) Schematic diagram of the closed-loop system, in which the virtual membrane current I is converted to a physical current I_{app} and integrated by a physical RC circuit representing realistic values for electrode and cell membrane characteristics. (B) Log–log plot of the number of Hodgkin–Huxley membrane patches vs the operating frequency at which the RTLDC can simulate them. The log–log relationship is approximately linear except for at low membrane patch numbers where the frequency plateaus at the maximum DAQ sampling rate of 20 kHz. (C) Circuit diagram of the virtual squid giant axon. This model consists of physical components in parallel with "virtual" components that exist only algorithmically. The virtual components include the negative conductance, $-g_m$, which cancels out the physical conductance, g_m. Each patch consists of a leak conductance, g_L, and two nonlinear conductances, g_{Na} and g_K, from three independent gating variables, each of which is based on two voltage dependent rate constants (Ref. 11). Each data point in B was collected by adding more patches [i.e., closing more switches in (C)] and then finding the safe operational frequency.

giant axon model would run was determined, and the theoretical membrane was then split into an increasing number of patches, in order to increase the computational load in a well-controlled manner without changing the results [Fig. 5(C)]. The conductances through each patch were calculated independently, and their corresponding currents were summed to create the output provided to the model cell. The results span from one membrane patch at 20 kHz to 1000 patches at 200 Hz [Fig. 5(B)]. These data show that the log of the maximum operating speed is approximately inversely proportional to the log of the computational complexity, except for where limited by the peak-sampling rate.

Recent studies in our lab have found that stellate cells in the MEC exhibit qualitative behavior determined by the variance in the conductance of a population of persistent sodium channels.[25] Specifically, at some current levels these cells autonomously toggle between two states: stable periodic spiking and subthreshold oscillations. A stochastic model of these cells (model from Acker,[1] stochastic method from Fox[7]) was implemented using RTLDC [Fig. 6(A)]. This model consists of a leak conductance, three nonlinear ordinary differential equation dependent conductances and a nonlinear stochastic differential equation dependent conductance. The RTLDC implementation shows both periodic spiking and

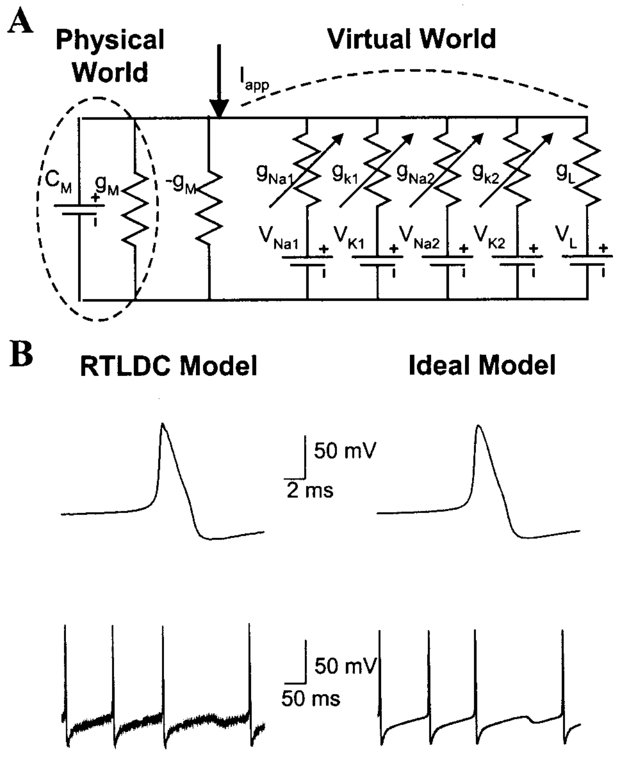


FIGURE 6. Example of a neuronal model that requires stochastic dynamics. (A) Circuit diagram of the model MEC stellate cell. The model consists of physical components in parallel with virtual components. Five biophysical conductances are modeled: a leak conductance, g_L, three nonlinear, ODE dependent conductances—g_{Na1}, g_{K1}, and g_{K2}—and a nonlinear, stochastic differential equation conductance, g_{Na2}. As in other models, the virtual —g_m is introduced to remove the physical g_m. (B) Responses of the MEC stellate cell to applied current (3.8 nA/cm^2) match ideal responses (generated using a detailed algorithm with a 10 μs discrete time step) very closely on both fine and gross time scales.

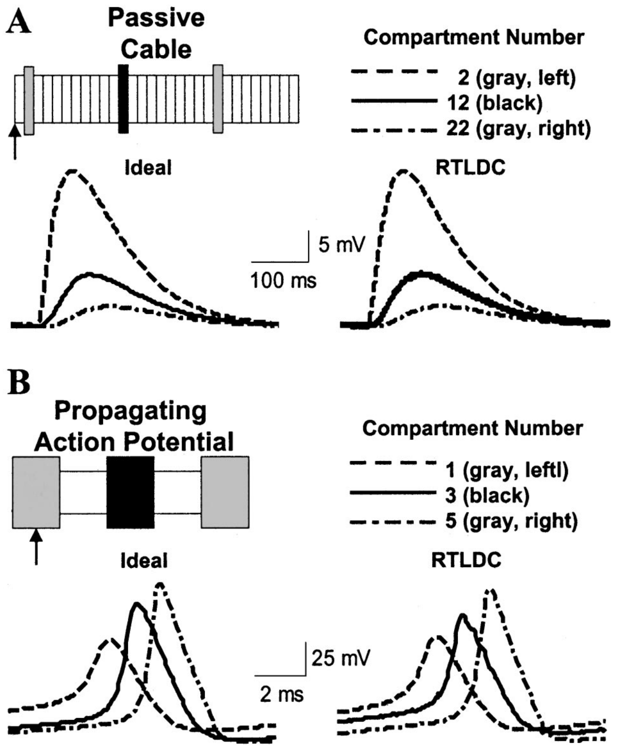


FIGURE 7. Simple spatially expansive models were created to demonstrate the feasibility of artificial compartment addition to real neurons. (A) and (B) detail two different, spatially extended models, and share the following format. Top left: Cartoon depiction of the models' spatial components—each box represents a spatially distinct compartment. The black compartments illustrate the position of the RC circuit within the cable and the gray compartments illustrate the position of the specific virtual compartments being examined. Top right: Legend that links the compartment location with the plotted data, below. Bottom: Voltage traces of the highlighted compartments for purely computational simulations (left), and RTLDC simulations (right), of the same mathematical models. The data on each graph are temporally aligned. (A) Real-time responses of a 30-compartment, linear virtual dendrite (length = 1 cm, compartment electrotonic length = 0.1, τ_m = 50 ms) to a conductance input (arrow), G_s. {$G_s = G_{\max} e^{-t/\tau d}(1-e^{-t/\tau r})$, where $G_{\max}$ is the maximum input conductance, and τ_r and τ_d are the rising and decaying time constants of 5 and 20 ms, respectively. Ideal responses are virtually identical. (B) Propagating action potential in a five-compartment virtual axon (length = 1 cm, compartment electrotonic length = 0.3, DC current (arrow) = 25 μA/cm^2). The differences in action potential magnitudes in the three compartments reflect boundary conditions. The RTLDC results closely mimic the ideal computational models.

subthreshold oscillations, and confirms that a stochastic, bistable system can be accurately simulated with the RTLDC [Fig. 6(B)].

Spatially Extended Models

Historical considerations of extended neurons held that dendrites could be treated as passive cables, such as in the Rall model,[22] while axons could be properly modeled by the Hodgkin–Huxley equations.[11] In keeping with this tradition, our first two attempts at attaching extensive artificial compartments to a real world neuronal counterpart were with these two models: passive cable and Hodgkin–Huxley squid giant axon.

In both cases, a theoretical cable was constructed and a physical world circuit, with realistic electrode and passive membrane characteristics, was inserted as a compartment in the middle of the virtual cable. The membrane potentials of the other compartments were simulated as state variables, along with the HH gating variables of every compartment. The passive cable was built with 30 compartments, each with an electrotonic length of 0.1. Artificial synaptic conductance, in the form of a rising and a falling biexponential function (see Fig. 7), was provided at one end of the cable [Fig. 7(A)]. The results were nearly indistinguishable from theoretical predictions.

The HH squid giant axon model was constructed with five compartments, each with an electrotonic length of

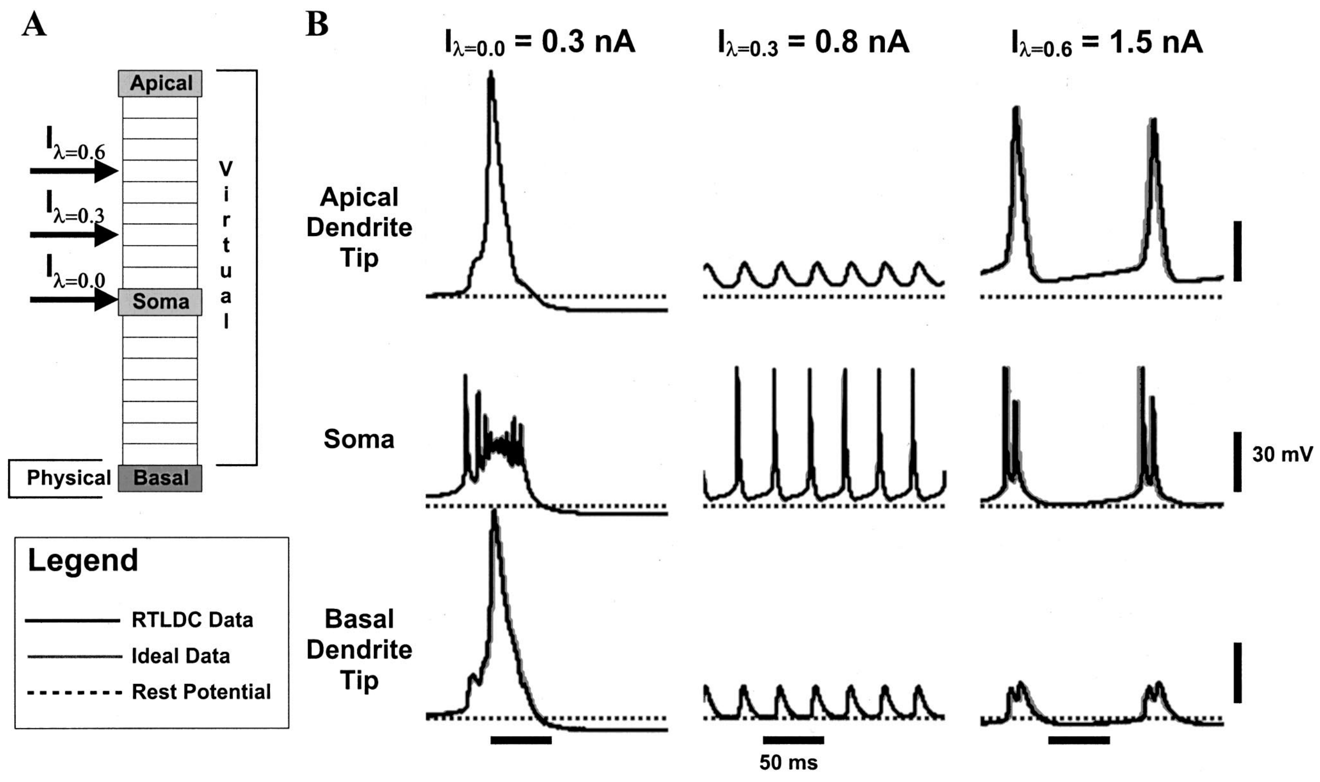


FIGURE 8. Complex model of the CA3 pyramidal cell was implemented with the RTLDC (Ref. 21), and run at a refresh frequency of 11 kHz. The compartments of this model have up to six nonlinear conductances apiece. (A) A schematic map of the CA3 model compartments: a soma, ten apical dendrite compartments and eight basal dendrite compartments. Virtual conductances were added to a RC circuit to create the basal tip—the 18 other compartments were completely virtual. In experiments, a simulated dc current was applied to only one compartment at a time. As depicted, three currents were examined: application to the apical compartments that were 0.3 and 0.6 length constants from the soma, $I_{\lambda=0.3}$ and $I_{\lambda=0.6}$, respectively, and to the soma itself, $I_{\lambda=0.0}$. The soma and apical- and basal-dendrite tips are labeled; these correspond to the three compartments for which data are shown. (B) Data collected from the experiments described above. Dynamic clamp data plotted in black; ideal computational models run with the same time step plotted in gray. On most graphs, the gray ideal results are completely hidden beneath the identical dynamic clamp data; left: somatic current injection leads to low frequency bursting of ~1 Hz (burst frequency not shown). The initial somal depolarization induces large calcium spikes in both the apical and basal dendrites, which lead to prolonged depolarization of the soma, and thus a somatic burst; middle: proximal apical dendrite stimulation leads to repetitive firing of ~35 Hz. The higher amplitude current leads to a more rapid somatic depolarization and repolarization, prohibiting the distal compartments from fully depolarizing; right: distal apical dendrite stimulation leads to repeating doublets at ~8 Hz. The more distal current application leads to full apical dendrite calcium spikes, which cause two complete de- and repolarization cycles in the soma. The somatic activity however, is inadequate to elicit calcium spikes in the basal dendrite, thus prohibiting the doublets from becoming full scale bursts. As shown, results are nearly indistinguishable from those generated by purely computational models.

0.3, the middle of which included the real world RC circuit. Depolarizing direct current was provided to one end of the cable at amplitude sufficient to create periodic action potential propagation [Fig. 7(B)]. The speed of propagation was well matched to analytical predictions. As in ideal models, the action potential shape was slightly modified as the wave progressed through the different compartments, due to the boundary conditions of the relatively short cable. Real-time results are extremely close to purely computational models.

A third multicompartment model was implemented that may be more relevant to modern electrophysiological experiments. This 19-compartment model of a CA3 pyramidal cell consists of six nonlinear conductances, which are expressed in varying concentrations in each compartment.[21] Five of these conductances are voltage dependent and two are calcium concentration dependent. Thus, in addition to the nine gating variables, membrane potential and calcium concentration serve as the eleven state variables for each compartment. The massive computational load imposed by this model limited the RTLDC to run at 11 kHz. As expected, the application of direct current to different compartments at different amplitudes led to the three qualitatively distinct behaviors seen in the ideal model: periodic spiking, doublet firing, and bursting (Fig. 8). As in previous cases, results from real-time simulations are difficult to distinguish from traditional computational models.

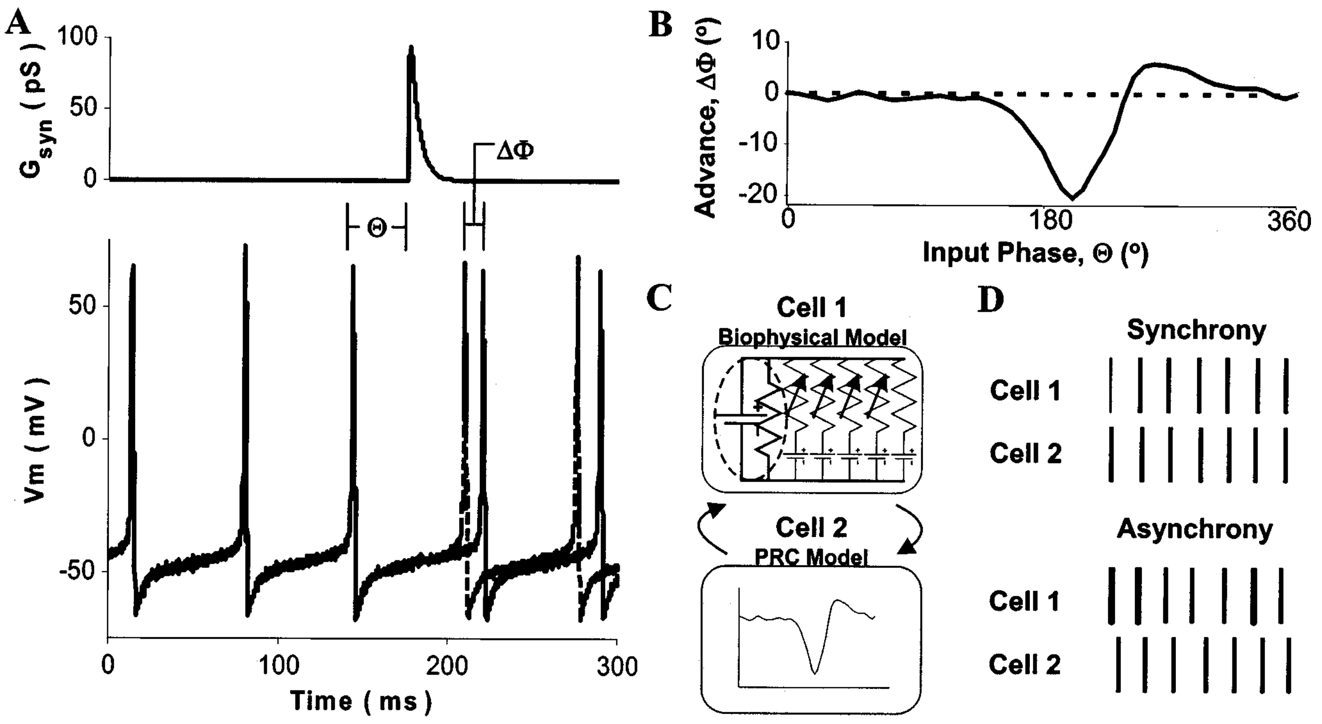


FIGURE 9. Noisy model MEC stellate cell (Ref. 25) was implemented with the RTLDC. Depolarizing current, sufficient to induce periodic firing, was provided to the cell, 12 uA/cm^2. The RTLDC was configured to collect the synapse phase response curve (synPRC) of the spiking cell. The resulting synPRC was used to simulate a second cell, bidirectionally coupled to the first, creating a 2 cell network. (A) Top: the conductance input provided to the biophysical model cell. The artificial conductance has rising and falling time constants of 0.5 and 5.0 ms, respectively. Bottom: The voltage trace of the noisy model cell. An artificial synaptic conductance change is initiated at a phase of Θ after the third action potential. This leads to a phase advance, ΔΦ, in the firing of the next action potential. The dotted line is the same cell under the same conditions, without the input. (B) Phase advances were measured for 50 phases (and low pass filtered) to construct the phase response curve. The synPRC shows the phase advance ΔΦ of the next action potential, as a function of the phase Θ, at which the artificial synaptic input was provided. Results are as expected for this noisy MEC model (Ref. 1). (C) Cartoon of the 2 cell network. Cell 1 is the same noisy MEC model, and is provided with the conductance change plotted at the top of (A), each time cell 2 fires. Cell 2 is a periodically firing cell (same frequency as cell 1, ~13 Hz), whose response to synaptic input from cell 1 is a phase advance or delay, as governed by the synPRC. (D) Raster plots of the network: Top: after the system has run for a few seconds, it always falls into synchronous firing. Bottom: under appropriate initial conditions, the cells can be made to teeter around antisynchrony (firing 180° out of phase) before converging toward synchrony. Theoretical results show that the antisynchronous mode is stable for noiseless cells of this type, but with noise the cells eventually switch to synchrony (Ref. 1).

Small Network Models

Beyond membrane mechanisms and neuronal processes, the RTLDC allows for network simulations in which real neurons could be synaptically coupled to completely artificial ones. To this end, experiments were conducted in which the synaptic phase response curve (synPRC)[1] of a model MEC stellate cell[25] was determined by the RTLDC. The synPRC characterizes the phase change of a periodically spiking MEC stellate cell due to a synaptic conductance, thereby allowing for behavioral predictions of small networks of these cells.

The noisy MEC stellate model instantiated with the RC circuit inclusion via the RTLDC was used to simulate a real neuron. Depolarizing direct current was provided such that the model cell exhibited regular, periodic spiking. The RTLDC was configured to provide pseudosynaptic input in the form of a biexponential conductance change at evenly distributed phases in the stellate cell period [Fig. 9(A)]. Specifically, the RTLDC provided 25 pseudosynaptic inputs at each of 50 phases, and measured the subsequent spike time advance or delay. The 25 spike time changes from each pseudosynaptic input at a given phase were averaged, and a synPRC map was constructed [Fig. 9(B)]. These maps align well with previous analytic and computational models examined by our lab.[1]

In a subsequent set of experiments, the noisy MEC stellate model with the real world, RC circuit was used to simulate one cell. The previously constructed synPRC map was used to simulate a second cell with no real world component. These two cells were synaptically coupled, bidirectionally, to create a two-cell network [Fig. 9(C)]. Depending on initial firing phases, the two-cell network exhibited synchronous (in phase) or nearly antisynchronous (180° out of phase) firing [Fig. 9(D)]. As previously seen by our group for this noisy MEC stellate model, antisynchronous behavior was less stable than synchronous.[1] As expected, the intrinsic noise eventually caused the networks teetering around antisynchrony to shift to synchrony.

DISCUSSION

In this work, we have described and tested an inexpensive and flexible system, based around Real-Time Linux (RTL), for high-speed, real-time experimental control. The system was designed to strike a balance among the crucial constraints of performance, cost, ease-of-use, and flexibility. Our applications involve the technique of dynamic clamping,[16,18,19] a relatively new method for testing specific hypotheses regarding how particular membrane conductances affect neuronal firing patterns. However, this design has been used for other real-time biomedical experiments,[5] and is easily modifiable for any real-time task with clock speeds up to ~50 kHz. With a relatively modern PC, computational operations are not rate limiting, even for very complex algorithms. This property makes our system particularly appropriate for computationally intensive real-time experimental tasks.

In choosing or developing a system for real-time control, one must first classify one's task as requiring either a ''soft'' or a ''hard'' real-time operating system (RTOS). Most OSs meet soft real-time requirements, in which commands are characterized by their average completion times. A soft RTOS framework is useful in communication applications, such as video conferencing, where audio and video feeds occur at a given frequency, but in which occasional cycle skips are acceptable. In contrast, applications like dynamic clamping require a hard RTOSs, characterized by worst-case performance, and thus guaranteed to complete assigned tasks on time, every time. The need for hard real-time performance constrains one's choices of operating systems considerably. For example, hard real-time performance can be attained in Windows NT/95/98/2000 only by installing specialized cards and/or purchasing expensive OS extensions. While many proprietary hard RTOS options exist, we chose RTL as an open source project which is robust, inexpensive, and guaranteed to be available for decades to come. For a more complete description of the suitability of RTL for biomedical applications, see Christini *et al.*[5]

For electrophysiologists to make use of the RTLDC, the RC circuit in the examples must be replaced by an electrode in series with a living cell. Wet experiments tend to be highly nonstationary: the parameters of the recording configuration drift over time. While these changes amount to manageably small errors in traditional electrophysiological experiments, they can yield significant miscalculations or even spawn devastating instabilities in a dynamic clamp. Therefore, experimentalists must strive for stable recordings in which the access resistance and electrode capacitance are held nearly constant. The electronics within the amplifier, specifically the resistance compensation and capacitance neutralization circuits, are thus able to remove electrical artifacts attributable to these components; hence, enabling the passage of the true cell membrane potential to the RTLDC model under all conditions. Complex experiments may require certain cell-intrinsic parameters (e.g., cell surface area, ion reversal potentials, channel activation time constants, etc.) to be measured or approximated, and subsequently incorporated into a RTLDC model. In these instances, it is recommended to (re)measure the parameter values in question at the end of the experiment, to verify that they did not change and/or were approximated correctly.[14] Finally, many conceivable dynamic clamp experiments require extremely fine model tuning. For example, it is possible to create a virtual analog of a particular population of ion channels to subtract out the physical conductance those channels provide. Slight differences between the physical channel properties and the model channel properties however, can lead to instabilities that might quickly destroy a neuron.[19] In contrast, eliminating channels with pharmacological blockers and replacing them with virtual conductances of known properties, would be quite stable.

Previous dynamic clamp implementations have explored the contributions of specific ion channel populations to cellular activity by adding virtual synapses,[13,16,19] and subtracting the effects of physical conductances.[14] They have also examined the roles certain conductances play in small network dynamics.[12,19] Future work will focus on a broad range of phenomena which include those studied previously. Purely computational models, made with assumptions such as electrotonic compactness or ion channel homogeneity, can be incorporated into living neurons to assess the predictions of the models and test the reasonableness of the assumptions. Ion channels, whose density changes have been implicated in diseased tissue malfunction, can be added to or removed from diseased neurons. In such experiments, the potential ability of the virtual channels to restore normal neuronal function would verify the role of the implicated channels in the disease state. As a final example, the electrical effects of computer designed drugs could be tested on healthy or diseased tissue before the pharmaceutical agents are even created.

To maximize power and flexibility we have built a dynamic clamp system based around Real-Time Linux. The biggest disadvantage of using RTL for real-time experimental control is that no commercial vendor has stepped in to create a user-friendly, turnkey solution. Our goal has been to bridge this gap for the end user. To this end, we have written a number of user-modifiable, open-source ''template'' programs for graphical user interfaces and specific dynamic clamp experiments (including the ones discussed in this paper). We have also compiled a systematic set of instructions for installation of Linux, RTL, the Comedi device drivers, and the Qt graphics libraries. This information can be found on the web at http://bme.bu.edu/ndl/rtldc.html.

ACKNOWLEDGMENTS

The authors are grateful to Dr. L. Abbott, Dr. R. Butera, Dr. B. Connors, Dr. Y. Manor, Dr. E. Marder, and Dr. F. Nadim for invaluable discussions regarding the technique of dynamic clamping. We would like to thank V. Abramson for his expert assistance in programming the RTLDC. We also thank D. Schleef, Dr. V. Yodaiken, and the rest of the RTL community for help with technical issues related to use of Real-Time Linux. This work was supported by a Grant to J. A. W. from the National Science Foundation (BES 0085177).

REFERENCES

[1] Acker, C. D. Synchronization of strongly coupled excitatory neurons: Relating biophysics to network behavior. M.S. Thesis, Boston University, Department of Biomedical Engineering, 2000.

[2] Barabanov, M., and V. Yodaiken. Real-time linux. *Linux J.* 34:19–23, 1997.

[3] Bartos, M., Y. Manor, F. Nadim, E. Marder, and M. P. Nusbaum. Coordination of fast and slow rhythmic neuronal circuits. *J. Neurosci.* 19:6650–6660, 1999.

[4] Bertram, R., J. Previte, A. Sherman, T. A. Kinard, and L. S. Satin. The phantom burster model for pancreatic beta-cells. *Biophys. J.* 79:2880–2892, 2000.

[5] Christini, D. J., K. M. Stein, S. M. Markowitz, and B. B. Lerman. Practical real-time computing system for biomedical experiment interface. *Ann. Biomed. Eng.* 27:180–186, 1999.

[6] Eng, E., QT GUI toolkit. *Linux J.* 31:32–43, 1996.

[7] Fox, R. J., Stochastic versions of the Hodgkin-Huxley equations. *Biophys. J.* 72:2068–2074, 1997.

[8] Gramoll, S., J. Schmidt, and R. L. Calabrese. Stitching in the activity state of an interneuron that controls coordination of the hearts in the medicinal leech (Hirudo-medicinalis). *J. Exp. Biol.* 186:157–171, 1994.

[9] Harsch, A., and H. P. C. Robinson. Postsynaptic variability of firing in rat cortical neurons: The roles of input synchronization and synaptic NMDA receptor conductance. *J. Neurosci.* 20:6181–6192, 2000.

[10] Hille, B. Ionic Channels of Excitable Membranes, 2nd ed. Sutherland, MA: Sinauer, 1992.

[11] Hodgkin, A. L., and A. F. Huxley. A quantitative description of membrane current and its application to conduction and excitation in nerve. *J. Physiol. (London)* 117:500–544, 1952.

[12] Hughes, S. W., D. W. Cope, and V. Crunelli. Dynamic clamp study of I-h modulation of burst firing and delta oscillations in thalamocortical neurons *in vitro*. *Neuroscience (Oxford)* 87:541–550, 1998.

[13] Jaeger, D., and J. M. Bower. Synaptic control of spiking in cerebellar Purkinje cells: Dynamic current clamp based on model conductances. *J. Neurosci.* 19:6090–6101, 1999.

[14] Ma, M., and J. Koester. The role of K^+ currents in frequency-dependent spike broadening in Aplysia R20 neurons: A dynamic-clamp analysis. *J. Neurosci.* 16:4089–4101, 1996.

[15] Reyes, A. D., E. W. Rubel, and W. J. Spain. *In vitro* analysis of optimal stimuli for phase-locking and time-delayed modulation of firing in avian nucleus laminaris neurons. *J. Neurosci.* 16:993–1007, 1996.

[16] Robinson, H. P. C., and N. Kawai. Injection of digitally synthesized synaptic conductance transients to measure the integrative properties of neurons. *J. Neurosci. Methods* 49:157–165, 1993.

[17] Sakmann, B., and E. Neher. Single-Channel Recording, 2nd ed. New York, NY: Plenum, 1995.

[18] Sharp, A. A., M. B. O'Neil, L. F. Abbott, and E. Marder. Dynamic clamp: Computer-generated conductances in real neurons. *J. Neurophysiol.* 69:992–995, 1993.

[19] Sharp, A. A., M. B. O'Neil, L. F. Abbott, and E. Marder. The dynamic clamp: Artificial conductances in biological neurons. *Trends Neurosci.* 16:389–394, 1993.

[20] Sharp, A. A., F. K. Skinner, and E. Marder. Mechanisms of oscillation in dynamic clamp constructed two-cell half-center circuits. *J. Neurophysiol.* 76:867–883, 1996.

[21] Traub, R. D., R. K. S. Wong, R. Miles, and H. Michelson. A model of a CA3 hippocampal pyramidal neuron incorporating voltage-clamp data on intrinsic conductances. *J. Neurophysiol.* 66:635–650, 1991.

[22] Weiss, T. Cellular Biophysics. Cambridge, MA: MIT Press, 1996.

[23] White, J. A., and A. D. Dorval. Effects of channel noise in the hippocampal formation. *Ann. Biomed. Eng.* 28:s109, 2000.

[24] White, J. A., J. S. Haas, and A. D. Dorval. Stochastic dynamic clamping as a method for studying the effects of biological noise sources. In: Proceedings of the 1999 IEEE Engineering in Medicine and Biology 21st Annual Conference and the 1999 Fall Meeting of the Biomedical Engineering Society, 1999, p. 880.

[25] White, J. A., R. Klink, A. Alonso, and A. R. Kay. Noise from voltage-gated ion channels may influence neuronal dynamics in the entorhinal cortex. *J. Neurophysiol.* 80:262–269, 1998.

[26] White, J. A., J. T. Rubinstein, and A. R. Kay. Channel noise in neurons. *Trends Neurosci.* 23:131–137, 2000.