6th Annual International IEEE EMBS Conference on Neural Engineering
San Diego, California, 6 - 8 November, 2013

A Rapid Algorithm Prototyping Tool for Bi-Directional Neural Interfaces

Benjamin Isaacson, Siddharth Dani, Sharanya Arcot Desai, *Member, IEEE,* Tim Denison, *Senior Member, IEEE,* Pedram Afshar, *Member IEEE*

Abstract—**Architectures capable of using an algorithm to modify actuation based on measured signals are often called "closed-loop" systems. While such systems are traditionally thought to rely on algorithms residing in device firmware, these may also reside outside the device in a host processor located physically nearby, or on a cloud-based architecture. In order to serve the potentially broad array of data processing modalities, we have developed an application programming interface (API). The API enables access to the sensing and stimulation capabilities of an implantable bi-directional neural interface. Systems using the API on different hardware/software platforms could measure neural signals, process signals in real-time, and modulate stimulation parameters using a variety of algorithms. This flexibility allows increased algorithm access and enables rapid prototyping for potentially improved technology solutions. The system performance was characterized using a signal generator to input square wave pulses to a Simulink model via the API. Closed-loop stimulation latencies of around 600ms were achieved.**

I. Need for Prototyping Systems for Closed-Loop Neuromodulation

Neurostimulation is used to treat a variety of neurological diseases such as Parkinson's disease, essential tremor, urinary incontinence, and chronic pain. To function properly, these technologies require both accurate hardware placement (e.g., placing leads in the correct nervous system location) as well as therapy parameter setting optimization (e.g. electrode selection, stimulation amplitude, pulse width, and frequency).

Selection of optimal parameters is largely an empirical process that may involve multiple, time-consuming device programming sessions spread apart by weeks or months. Outside of these sessions, the ability to make stimulation parameter adjustments is generally limited.

Algorithms for parameter adjustment might provide an opportunity to improve efficiency, efficacy, and access to parameter modifications in neurostimulation. Examples of algorithmic automation concepts in healthcare devices can be found in cardiac pacing [1], diabetes [2], and respiration [3]. The exploration of closed-loop therapies is motivated by a number of potential advantages:

1. Reduced intervention response time: This is the ability to respond to an episodic disease state (e.g., seizure onset in epilepsy) without manual intervention

2. Personalized parameter adjustment: This is the use of the subject's particular natural history of disease to drive intervention rather than relying on potentially imperfect subject population groupings to determine optimal parameters.

3. Reduced healthcare burden: This is the ability to change parameters to adjust to a subject's evolving disease state with less clinical burden. Also, titrating therapy parameters that are delivered to the patient might optimize implant energy usage, thereby potentially increasing device longevity and reducing the need for burdensome battery replacement surgery.

A closed-loop design (Fig. 1) relies on the following critical components: sensors to measure biomarkers of disease such as inertial (e.g., posture and activity) or bioelectric measurements (e.g., local field potentials (LFP), electrocorticography (ECoG), and electromyography (EMG)); classifiers and control policies to provide an algorithm to interpret the biomarkers and determine an appropriate response, and effectors to deliver a response that potentially modifies the disease state.

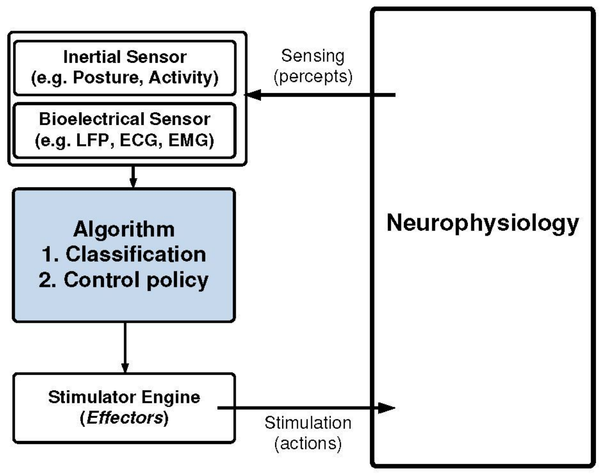


Figure 1: Closed-loop neural interface elements

An important step in developing these systems is the ability to determine algorithms: both appropriate classifiers and control policies. These algorithms may have myriad forms, including requiring spectral energy extraction and processing [4], coherence calculation [5], phase-amplitude coupling calculation [6], and evoked-potential measurement [7]. Furthermore, these algorithms may require the ability to access persistent data storage for signal averaging, learning algorithms, and machine learning classification. And there may be more, yet undiscovered, signal relationships that could be relevant for neurologic diseases.

All authors: Medtronic Neuromodulation, Minneapolis, MN 55432 USA
(phone: email:)

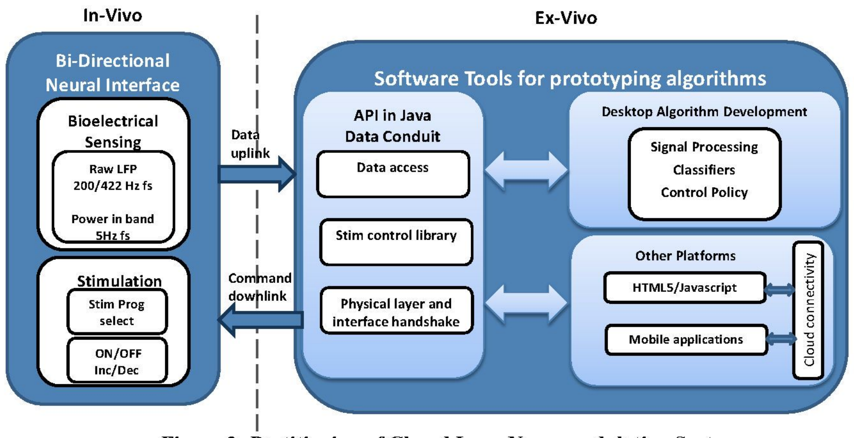


Figure 2: Partitioning of Closed Loop Neuromodulation Systems

While it is possible to store data and perform computation inside a device [8,9], it may not be feasible to experiment with a broad range of algorithms rapidly due to complexity involved in implementing these algorithms in device firmware.

One method to accelerate algorithm development for closed loop systems is by performing processing and storage of data external to the implantable device. This approach enables algorithms to be more rapidly prototyped in order to better characterize the trade-offs between algorithmic complexity, system performance, power consumption, etc. With this understanding, appropriate algorithms could be embedded in device firmware for chronic operation.

II. Interface Features for Algorithm prototyping

There are 3 key desirable features in designing system interfaces that enable data processing and storage external to an implantable device.

1. Platform independence: Allow for use with multiple computational packages such as MATLAB/Simulink, LabVIEW, mobile applications, etc.

2. Low communication latency: A usable interface must add minimal latency from signal detection to actuation.

3. Data security and privacy: The system architecture will need to consider requirements for the confidentiality, integrity, availability, authentication and authorization of data when outside the implantable device, and also disallow a host application from misusing the implantable device.

One way to realize these features is by using an application programming interface (API). The API resides on a host processor (e.g., desktop computer, mobile device) and serves as a communication layer between the host processor and the hardware that communicates with the implantable device (Fig. 2). Its function is to translate commands from the host processor into low-level commands interpreted by the implantable device.

In addition to addressing the features discussed above, the API hides complexity involved in low-level hardware communication from algorithm developers. It can also be updated independently of host processor applications. A similar approach has been adopted in developing APIs focused on similar pre-clinical research [10].

III. System Design

The prototype system designed to explore rapid algorithm prototyping has the following components:

A. *Bi-Directional Neural Interface*

The bi-directional neural interface (BNI) used has the capability of streaming data at 200 or 422 Hz in time domain, and at 5Hz when transmitting power in a pre-defined frequency band. For the results in this paper, 422 Hz sampling was used. Details of the bi-directional neural interface may be found in Stanslaski et al., 2012 [12].

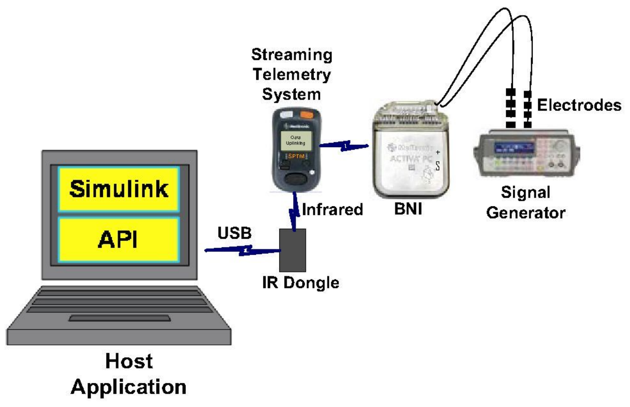


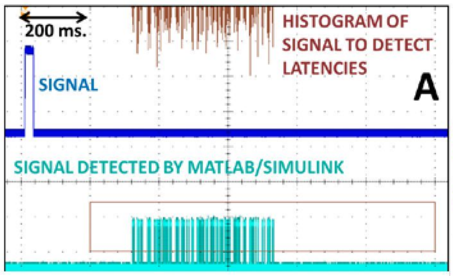
Figure 3: Test Setup. The system was tested by processing signals input into the bi-directional neural interface (BNI) from a signal generator which were then communicated via the API to the host processor running MATLAB and Simulink

B. *Telemetry device*

Data packets from the BNI were transmitted through a low-frequency proximal (~4 cm) inductive telemetry device to the host processor. Stimulation programs on the BNI were setup by the standard clinician programmer (Medtronic, Inc. model 8840). The closed-loop system could only vary stimulation parameters within the bounds set by the clinician programmer.

C. *Application Programming Interface (API)*

The API, written in Java, utilized a multithreaded telemetry execution engine to process the transfer and receipt of commands over a generic connection interface



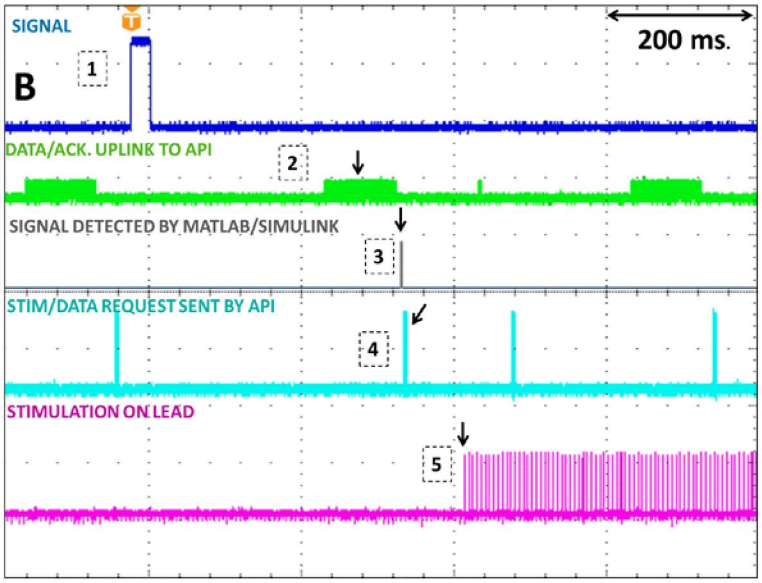


Figure 4: Oscilloscope screenshots. (A) Cyan traces(detection of pulses by Simulink) and histogram shows that the signal occurrence to detect latency depends on position of pulse (blue trace) within the 400 ms. data packet. (B) A random trial of full closed-loop system latency computation with the Inc. command. (1) Single square pulse is input to the BNI; (2) the pulse is sent within a 400 ms. packet to the API; (3) the pulse is detected by Simulink; (4) Stim (Inc) command is sent by API when pulse is detected; (5) Stimulation appears on lead.

which was implemented using various physical layers (e.g., USB, Bluetooth, etc.). The host processor instantiated an instrument instance, requested data from and sent stimulation updates to the BNI.

D. Host Processor and Algorithm

For this work, a PC-based host processor running MATLAB and Simulink (Mathworks) for algorithm development was selected. The algorithm consisted of a classifier to identify states of interest from the input signal and a control policy to modify stimulation based on identified state. The classifier was an empirically-derived linear threshold on the amplitude of the raw signal. The corresponding control policy toggled stimulation OFF (0V) and ON (6V) based on the classifier output.

IV. Test Setup

The system was validated using a test setup as illustrated in Fig 3. Signals from a signal generator were input to the bi-directional neural interface (BNI) and telemetry was used to stream data from the BNI to the host computer running the API and algorithm software. For the experiments performed in this paper, data from the BNI were sent in packets of 400 ms to the host processor.

Sixty-six square pulses from a signal generator, each 20 Hz at 50% duty cycle and 60μV were input to the BNI and retrieved by the host processor through the API. The pulses were distributed over 10 minutes and the spacing between them was chosen to randomly position them in the 400 ms data packets sent from the BNI to the host processor through telemetry (Oscilloscope screenshot in Fig 4 A shows the distribution of the pulse positions within the 400 ms data packets).

V. Results

For the system described in this paper, latencies were introduced by numerous individual system elements such as, telemetry, detection, API latency, etc. Latency was calculated separately for sensed data retrieval commands and for stimulation control. Sensing latency was computed by measuring the time elapsed between the occurrence of the signal on the electrodes and its detection by the host processor (time between 1 and 3 in Fig 4B). Stimulation latency was computed by measuring the time between detection/issuing a command to change stimulation and the stimulation change at the electrode (time between 3 and 5 in Fig 4B). All latency measurements and statistics computations were automated using a Tektronix MSO 4034 Mixed Signal Oscilloscope.

Table 1 shows the latencies for the system from signal input to stimulation on the lead. Section A in table 1 contains Sensing latency statistics and Section B contains Stimulation latency statistics. To get the entire closed-loop system latency, numbers in Section A should be added to the appropriate latency numbers in Section B. For example, when the Inc/Dec (increment/decrement) command is used to adjust stimulation amplitude, the mean closed loop system latency from Signal to Stimulation would be 508.6 ms + 80.86 ms ~ 590 ms. Fig 4 B is an oscilloscope screenshot illustrating the different latency steps from signal to stimulation in a randomly selected trial. Table 2 shows the latencies that are introduced by the API alone. Note that the API has minimal contribution to the overall system latency.

Table 1 (A): Sensing Latency

Command	Description	Mean (ms)	Min (ms)	Max (ms)	Std (ms)
Get Data	Get data packets from the INS	508.6	311.2	716.9	116.2

Table 1 (B): Stimulation Latency

Command	Description	Mean (ms)	Min (ms)	Max (ms)	Std (ms)
Therapy On	Switch ON stim engine	354.6	343.9	372.4	7.1

Therapy Off	Switch OFF stim engine	52.4	45.1	61.1	4.3
Group Switch	Change multiple stim parameters	495.9	490.5	505.1	3.7
Inc/Dec (Amp,PW,Freq.)	Increase/decrease stimulation amplitude, pulse width or frequency	80.9	76.0	84.6	2.6

Note: To get full closed-loop system latency, add Sensing latency to latency associated with appropriate command from the Stimulation Latency table.

Table 2: API only latency

Command	Mean (ms)	Min (ms)	Max (ms)	Std (ms)
Get Data	3.3	2.6	8.6	0.7
Therapy On	2.9	1.8	2.7	0.3
Therapy Off	2.9	2.1	3.2	0.2
Group Switch	2.7	1.3	6.8	0.6
Inc/Dec (Amp; PW;Freq)	2.9	1.8	4.0	0.3

VI. Discussion and Conclusion

For the system described in this paper, the closed-loop latency from signal occurrence to stimulation at electrode is ~600 ms. The main contribution to the total latency comes from the packaging of data samples in 400 ms packets and transmission of these packets via telemetry (sensing latency). The large variance in this sensing latency (~ 400 ms; table 1A) is due to the variance in the position of the signal of interest (pulse signal in this study) within the 400 ms data packets (Fig. 4A). Pulse signals occurring at the very end of the data packet result in total closed-loop system latency of ~ 390 ms when the Inc/Dec command is used. On the other hand, pulse signals occurring at the very beginning of the data packet result in total latency of ~ 790 ms.

Although the telemetry used introduced communication latency, the leveraging of an existing platform simplified the development process of the API and host application. The telemetry technology used in this prototype concept is leveraged from systems with CE mark approval (not approved for commercial use in the U.S.; investigational use only). A future improvement could be to update device firmware to have shorter data packets or implement faster telemetry hardware and thus gain an advantage on signal detection latency.

In spite of telemetry latency, the ability to run an algorithm on an entire data packet, accelerated with a powerful external computer, can help compensate for some of the latency disadvantage. In the current system an algorithm that performs batch processing on incoming data packets can output a result every 400ms (data packet length), which is comparable to pilot data on closed-loop neurostimulation systems that perform a moving average of 400ms sensed signals [11].The API described in this work provides simplicity, flexibility, and security for researchers interested in prototyping algorithms. This approach allows for rapid and broad algorithm development. Algorithms that meet performance needs for closed-loop systems may then be implemented in device firmware optimizing for computation and power consumption. Proximal telemetry, the host application modifying stimulation parameters only within boundaries set by clinical programmers, and the ability of the API to detect data loss or corruption make for a robust setting for developing closed-loop neurostimulation algorithms

Biomarkers extracted from multiple signal sources representative of disease state or through other processing methods are convenient to explore with the flexibility on the host processor that the API enables. The Java implementation of the API also opens the possibility for mobile applications to interact with devices so long as data security and privacy are maintained, which in the future could allow for greater collaboration between researchers and reduce the learning time for understanding neural mechanisms.

VII. Acknowledgements

The authors acknowledge help provided by Dave Linde, Dave Carlson, Randy Jensen and Scott Stanslaski, of Medtronic Neuromodulation in debugging and testing the system.

VIII. References

[1] P. Palmisano *et al.*, "Closed-loop cardiac pacing vs. conventional dual-chamber pacing with specialized sensing and pacing algorithms for syncope prevention in patients with refractory vasovagal syncope: results of a long-term follow-up," *Europace* (2012) 14 (7): 1038-1043, doi: 10.1093/europace/eur419.

[2] S. D. Patek *et al.*, "Modular closed-loop control of diabetes," *IEEE Transactions on Biomedical Engineering*, vol. 59, no. 11, November 2012, pp.2986-2999.

[3] N. Claure, MSc PhD *et al*, "Automated closed loop control of inspired oxygen concentration," *Respiratory Care*, January 1, 2013 vol. 58 no. 1 151-161.

[4] A. Priori, G. Foffani, L. Rossi, S. Marceglia, "Adaptive deep brain stimulation (aDBS) controlled by local field potential oscillations," *Experimental Neurology*, 2013, vol 245, pp. 77-86.

[5] S.N. Kalitzin, D.N. Velis, F.H. Lopes de Silva, "Stimulation-based anticipation and control of state transitions in the epileptic brain," *Epilepsy and Behavior*, 2010, vol 17, p.p. 310-323.

[6] C. de Hemptinne, E.S. Ryapolova-Webb, E.L. Air, P.A. Garcia, K.J. Miller, J.G. Ojemann, J.L. Ostrem, N.B. Galifianakis, P.A. Starr, "Exaggerated phase-amplitude coupling in the primary motor cortex in Parkinson disease," *Proc National Acad Sci*, 2013, vol 110(12), pp. 4780-5.

[7] D.R. Freestone, S.N. Long, S. Frey, P.H. Syptulkowski, J.E. Giftakis, M.J. Cook, "A Method for Actively Tracking Excitability of Brain Networks using a Fully Implantable Monitoring System," *6th International IEEE EMBS Neural Engineering*.

[8] D. Carlson, D. Linde, B. Isaacson, P. Afshar, D. Bourget, S. Stanslaski, P. Stypulkowski, and T. Denison, "A Flexible Algorithm Framework for Closed-Loop Neuromodulation Research Systems," 35th Annual International Conference of the IEEE EMBS, Osaka, Japan, 3 - 7 July, 2013

[9] P. Afshar, A. Khambati, S. Stanslaski, D. Carlson, R. Jensen, D. Linde, S. Dani, M. Lazarewicz, P. Stypulkowski, T. Denison, "A translational platform for prototyping closed-loop neuromodulation systems," *Frontiers in Neural Circuits*, 2012, vol 7(117).

[10] Newman, J.P., Zeller-Townson, R., Fong, M.F., Arcot Desai, S., Gross, R.E., and Potter, S.M.: 'Closed-Loop, Multichannel Experimentation Using the Open-Source NeuroRighter Electrophysiology Platform', Frontiers in neural circuits, 2012, 6, pp. 98

[11] Little, S., Pogosyan, A., Neal, S., Zavala, B., Zrinzo, L., Hariz, M., Foltynie, T., Limousin, P., Ashkan, K., FitzGerald, J., Green, A. L., Aziz, T. Z. and Brown, P. (2013), Adaptive deep brain stimulation in advanced Parkinson disease. Ann Neurol.. doi: 10.1002/ana.23951

[12] S. Stanslaski, P.Cong, D.Carlson, W. Santa, R. Jensen, G. Molnar, W. Marks, A. Shafquat, and T. Densison, "An implantable bi-directional brain-machine interface system for chronic neuroprosthesis research," IEEE EMBC 2009, pp. 5494-5497. DOI:10.1109/IEMBS.2009.5334562